Recherches en vision par ordinateur

Peter Sturm

Recueil d'articles

présenté par

Peter Franz STURM

pour obtenir le grade de
Habilitation à diriger des Recherches
de l'**INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE**
Spécialité : **Informatique**

———

Recherches en vision par ordinateur

———

Présenté publiquement le 16 mai 2006

devant le jury composé de

Président :	M. Roger Mohr	INPG
Rapporteurs :	M. Kostas Daniilidis	University of Pennsylvania
	M. Michel Dhome	LASMEA
	M. Jean Ponce	Beckman Institute et University of Illinois
Examinateurs :	M. Luc Van Gool	Katholieke Universiteit Leuven et ETH Zürich
	M. Richard Hartley	The Australian National University
	M. Radu Horaud	INRIA
	M. Long Quan	Hong Kong University of Science and Technology

Avant-propos

Ce recueil d'articles accompagne le document de synthèse des travaux et activités scientifiques. La structure de ce document-ci est identique à celle des parties II à V du document de synthèse.

Foreword

This collection of papers accompanies the document containing the synthesis of my research and related activies ("synthèse des travaux et activités scientifiques"). Its structure is identical to that of parts II to V of the latter.

Contents

Part II

Calibration and Self-Calibration of Perspective Cameras

Chapter 4

Camera Calibration

4.1 Plane-Based Calibration

Paper 1 [29]: P. Sturm and S. Maybank. On plane-based camera calibration: A general algorithm, singularities, applications. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Fort Collins, Colorado, USA*, pages 432–437, June 1999.

Paper 2 [20]: P. Sturm. Algorithms for plane-based pose estimation. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Hilton Head Island, South Carolina, USA*, pages 1010–1017, June 2000.

4.2 Using Linear Calibration Objects

Paper 3 [13]: P. Hammarstedt, P. Sturm, and A. Heyden. Closed-form solutions and degenerate cases for camera calibration with one-dimensional objects. In *Proceedings of the 10th International Conference on Computer Vision, Beijing, China*, October 2005.

4.3 Calibration of Zoom Lenses

Paper 4 [38]: M. Urbanek, R. Horaud, and P. Sturm. Combining off- and on-line calibration of a digital camera. In *Proceedings of the Third International Conference on 3D Digital Imaging and Modeling, Québec City, Canada*, pages 99–106, May 2001.

On Plane-Based Camera Calibration: A General Algorithm, Singularities, Applications

Peter F. Sturm and Stephen J. Maybank
Computational Vision Group, Department of Computer Science, The University of Reading
Whiteknights, PO Box 225, Reading, RG6 6AY, United Kingdom
{P.F.Sturm, S.J.Maybank}@reading.ac.uk

Abstract

We present a general algorithm for plane-based calibration that can deal with arbitrary numbers of views and calibration planes. The algorithm can simultaneously calibrate different views from a camera with variable intrinsic parameters and it is easy to incorporate known values of intrinsic parameters. For some minimal cases, we describe all singularities, naming the parameters that can not be estimated. Experimental results of our method are shown that exhibit the singularities while revealing good performance in non-singular conditions. Several applications of plane-based 3D geometry inference are discussed as well.

1 Introduction

The motivations for considering planes for calibrating cameras are mainly twofold. First, concerning calibration in its own right, planar calibration patterns are cheap and easy to produce, a laser printer output for example is absolutely sufficient for applications where highest accuracy is not demanded. Second, planar surface patches are probably the most important twodimensional "features": they abound, at least in man-made environments, and if their metric structure is known, they carry already enough information to determine a camera's pose up to only two solutions in general [4]. Planes are increasingly used for interactive modeling or measuring purposes [1, 10, 11].

The possibility of calibrating cameras from views of planar objects is well known [7, 12, 14]. Existing work however, restricts in most cases to the consideration of a single or only two planes (an exception is [8], but no details on the algorithm are provided) and cameras with constant calibration. In addition, the study of singular cases is usually neglected (besides in [12] for the simplest case, calibration of the aspect ratio from one view of a plane), despite their presence in common configurations.

It is even possible for cameras to self-calibrate from views of planar scenes with unknown metric structure [13], however several views are needed (Triggs recommends up to 9 or 10 views of the same plane for reliable results) and the "risk" of singularities should be greater compared to calibration from planes with known metric structure.

In this paper, we propose a general algorithm for calibrating a camera with possibly variable intrinsic parameters and position, that copes well with an arbitrary number of calibration planes and camera views. Calibration is essentially done in two steps. First, the 2D-to-2D projections of planar calibration objects onto the image plane(s) are computed. Each of these projections contributes to a system of homogeneous linear equations in the intrinsic parameters, which are hence easily determined. Calibration can thus be achieved by solving linear equations, but can of course be enhanced by subsequent non linear optimization.

In §2, we describe our camera model and projections of planar objects. In §3, we introduce the principle of plane-based calibration. A general algorithm is proposed in §4. Singularities are revealed in §5. Experimental results are presented in §6, and some applications described in §7.

2 Background

Camera Model. We use perspective projection to model cameras. A projection may be represented by a 3×4 projection matrix P that incorporates the so-called extrinsic and intrinsic camera parameters:

$$\mathsf{P} \sim \mathsf{K}\mathsf{R}(\quad \mathtt{I}_3 \quad | - \mathbf{t}) \ . \tag{1}$$

Here, $\sim$ means equality up to a non zero scale factor, $\mathtt{I}_3$ is the 3×3 identity matrix, R a 3×3 orthogonal matrix representing the camera's orientation, $\mathbf{t}$ a 3-vector representing its position, and K the 3×3 calibration matrix:

$$\mathsf{K} = \begin{pmatrix} \tau f & s & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{pmatrix} \ .$$

In general, we distinguish 5 intrinsic parameters for perspective projection: the (effective) focal length f, the aspect ratio τ, the principal point (u_0, v_0) and the skew factor s accounting for non rectangular pixels. The skew factor is usually very close to 0 and we ignore it in the following.

Calibration and Absolute Conic. Our aim is to calibrate a camera, i.e. to determine its intrinsic parameters or its

calibration matrix K (subsequent pose estimation is relatively straightforward). Instead of directly determining K, we will try to compute the symmetric matrix $\mathsf{K}\mathsf{K}^\mathsf{T}$ or its inverse, from which the calibration matrix can be computed uniquely using Cholesky decomposition [5]. This leads to simple and, in particular, linear calibration equations. Furthermore, the analysis of singularities of the calibration problem is greatly simplified: the matrix $\omega \sim (\mathsf{K}\mathsf{K}^\mathsf{T})^{-1}$ represents the image of the Absolute Conic whose link to calibration and metric scene reconstruction is exposed for example in [2]. This geometrical view helps us with the derivation of singular configurations (cf. §5).

Planes, Homographies and Calibration. We consider the use of one or several planar objects for calibration. When we talk about *calibration planes*, we mean the supports of planar calibration objects. The restriction of perspective projection to points (or lines) on a specific plane takes on the simple form of a 3×3 homography that depends on the relative position of camera and plane and the camera's intrinsic parameters. Without loss of generality, we may suppose that the calibration plane is the plane $Z = 0$. This way, the homography can be derived from the projection matrix P by dropping the third column in equation (1):

$$\mathsf{H} \sim \mathsf{K}\mathsf{R} \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix} \quad . \tag{2}$$

The homography can be estimated from four or more point or line correspondences. It can only be sensibly decomposed as shown in equation (2), if the metric structure of the plane is known (up to scale is sufficient), i.e. if the coordinates of points and lines used for computing H are given in a metric frame.

Equation (2) suggests that the 8 coefficients of H (9 minus 1 for the arbitrary scale) might be used to estimate the 6 pose parameters R and $\mathbf{t}$, while still delivering 2 constraints on the calibration K. These constraints allow us to calibrate the camera, either partially or fully, depending on the number of calibration planes, the number of images, the number of intrinsic parameters to be computed and on singularities.

3 Principle of Plane-Based Calibration

Calibration will be performed via the determination of the image of the Absolute Conic (IAC), $\omega \sim \mathsf{K}^{-\mathsf{T}}\mathsf{K}^{-1}$, using plane homographies. As mentioned previously, we consider pixels to be rectangular, and thus the IAC has the following form (after appropriate scaling):

$$\omega \sim \begin{pmatrix} 1 & 0 & -u_0 \\ 0 & \tau^2 & -\tau^2 v_0 \\ -u_0 & -\tau^2 v_0 & \tau^2 f^2 + u_0^2 + \tau^2 v_0^2 \end{pmatrix} \quad . \tag{3}$$

The calibration constraints arising from homographies can be expressed and implemented in several ways. For example, it follows from equation (2) that:

$$\mathsf{H}^\mathsf{T}\omega\,\mathsf{H} \sim \mathsf{H}^\mathsf{T}\mathsf{K}^{-\mathsf{T}}\mathsf{K}^{-1}\mathsf{H} \sim \begin{pmatrix} 1 & 0 & -t_1 \\ 0 & 1 & -t_2 \\ -t_1 & -t_2 & \mathbf{t}^\mathsf{T}\mathbf{t} \end{pmatrix} \quad .$$

The camera position $\mathbf{t}$ being unknown and the equation holding up to scale only, we can extract exactly two different equations in ω that prove to be homogeneous linear:

$$\mathbf{h}_1^\mathsf{T}\,\omega\,\mathbf{h}_1 - \mathbf{h}_2^\mathsf{T}\,\omega\,\mathbf{h}_2 = 0 \qquad \mathbf{h}_1^\mathsf{T}\,\omega\,\mathbf{h}_2 = 0 \ , \tag{4}$$

where $\mathbf{h}_i$ is the ith column of H. These are our basic calibration equations. If several calibration planes are available, we just include the new equations into a linear equation system. It does not matter if the planes are seen in the same view or in several views or if the same plane is seen in several views, provided the calibration is constant (this restriction is relaxed in the next section). The equation system is of the form $\mathsf{A}\mathbf{x} = \mathbf{0}$, with the vector of unknowns $\mathbf{x} = (\omega_{11}, \omega_{22}, \omega_{13}, \omega_{23}, \omega_{33})^\mathsf{T}$. After having determined $\mathbf{x}$, the intrinsic parameters are extracted via:

$$\begin{aligned} \tau^2 &= \frac{\omega_{22}}{\omega_{11}} \qquad u_0 = -\frac{\omega_{13}}{\omega_{11}} \qquad v_0 = -\frac{\omega_{23}}{\omega_{22}} \\ f^2 &= \frac{\omega_{11}\omega_{22}\omega_{33} - \omega_{22}\omega_{13}^2 - \omega_{11}\omega_{23}^2}{\omega_{11}\omega_{22}^2} \end{aligned} \tag{5}$$

4 A General Calibration Algorithm

We describe now how the basic principle can be extended in two important ways. First, we show that prior knowledge of intrinsic parameters can be easily included. Second, and more importantly, we show how the scheme can be applied for calibrating cameras with variable intrinsic parameters.

4.1 Prior Knowledge of Intrinsic Parameters

Let $\mathbf{a}_i$ be the ith column of the design matrix A of the linear equation system described in the previous section. We may rewrite the equation system as:

$$\omega_{11}\mathbf{a}_1 + \omega_{22}\mathbf{a}_2 + \omega_{13}\mathbf{a}_3 + \omega_{23}\mathbf{a}_4 + \omega_{33}\mathbf{a}_5 = \mathbf{0} \ .$$

Prior knowledge of, e.g. the aspect ratio τ, allows us via equation (5) to eliminate one of the unknowns, say ω_{22}, leading to the reduced linear equation system:

$$\omega_{11}(\mathbf{a}_1 + \tau^2\mathbf{a}_2) + \omega_{13}\mathbf{a}_3 + \omega_{23}\mathbf{a}_4 + \omega_{33}\mathbf{a}_5 = \mathbf{0} \ .$$

Prior knowledge of u_0 or v_0 can be dealt with similarly. The situation is different for the focal length f, due to the complexity of equation (5): prior knowledge of f allows to eliminate unknowns only if the other parameters are known, too. However, this is not much of an issue – it is rarely the case that the focal length is known beforehand while the other intrinsic parameters are unknown.

4.2 Variable Intrinsic Parameters

We make two assumptions that are not very restrictive but eliminate useless special cases to deal with. First, we consider the aspect ratio to be constant for a given camera. Second, the principal point may vary, but only in conjunction with the focal length. Hence, we consider two modes of variation: only f varies or f, u_0 and v_0 vary together.

If we take into account the calibration equations arising from a view for which it is assumed that the intrinsic parameters have changed with respect to the preceding view (e.g. due to zooming), we just have to introduce additional unknowns in $\mathbf{x}$ and columns in A. If only the focal length is assumed to have changed, a new unknown ω_{33} is needed. If in addition the principal point is supposed to have changed, we add also unknowns for ω_{13} and ω_{23} (cf. equation (3)). The corresponding coefficients of the calibration equations have to be placed in additional columns of A.

Note that the consideration of variable intrinsic parameters does not mean that we have to assume different values for *all* views, i.e. there may be views sharing the same intrinsics, sharing only the aspect ratio and principal point, or sharing the aspect ratio alone.

4.3 Complete Algorithm

The complete algorithm consists of the following steps:

1. Compute plane homographies from feature correspondences.
2. Construct the equation matrix A according to the directions outlined in §§3,4.1 and 4.2.
3. Ensure good numerical conditioning of A (see below).
4. Solve the equation system to least squares by any standard method and extract the intrinsic parameters from the solution as shown in equation (5).

Conditioning. We may improve the conditioning of A by the standard technique of rescaling rows and columns [5]. In practice, we omit row-wise rescaling for reasons explained below. Columns are rescaled such as to have equal norms. The coefficients of the solution vector of the modified equation system have to be scaled accordingly to obtain the solution of the original problem. In our experiments, this rescaling proved to be crucial to obtain reliable results.

As for rescaling rows, this proves to be delicate in our case, since occasionally there are rows with all coefficients very close to zero. Rescaling these rows will hugely magnify noise and lead to unreliable results.

Comments. The described calibration algorithm requires mainly the least squares solution of a single linear equation system. Naturally, the solution may be optimized subsequently using non linear least squares techniques. This optimization should be done simultaneously for the calibration and the pose parameters, that may be initialized in a straightforward manner from the linear calibration results. For higher accuracy, estimation of optical distortion parameters should be included.

Minimal Cases. Each view of a calibration object provides two calibration equations. Hence, in the absence of singularities, the following minimal calibration schemes may be realized: with a single view of a single plane, we might calibrate the aspect ratio and focal length, provided the principal point is given. With two views of a single plane, or one view of two planes we can fully calibrate the camera. Three views of a single plane, taken by a zooming camera, enable calibration of the 3 different focal lengths, as well as the constant aspect ratio and principal point.

5 Singularities

The successful application of any algorithm requires awareness of singularities. This helps avoiding situations where the result is expected to be unreliable or restricting the problem at hand to a solvable one. We describe here the singularities of calibration from one or two planes.

Due to lack of space, we are only able to give a sketch of the derivations. A first remark is that only the relative orientation of planes and camera is of importance for singularities, i.e. the position and the actual intrinsic parameters do not influence the existence of singularities. A second observation is that planes that are parallel to each other provide exactly the same information as a single plane with the same orientation (except that more feature correspondences may provide a higher robustness in practice). So, as for the case of two calibration planes, we omit dealing with parallel planes and instead refer to the one-plane scenario.

Since the calibration equations are linear, singularities imply the existence of a linear family of solutions for the IAC ω. Hence, there is also a degenerate conic ω', i.e. a conic consisting of the points on two lines only. Let us note that any conic that satisfies the calibration equations (4), contains the projections of the circular points of the calibration planes. Naturally, this is also valid for ω'. If we exclude the planes of being parallel to each other (cf. the above discussion), the two lines making up ω' are nothing else than the vanishing lines of the calibration planes. There is one point left to consider: since we are considering rectangular pixels, the IAC is required to be of the form (3), i.e. its coefficient ω_{12} is zero. Geometrically, this is equivalent to the conic being symmetric with respect to a vertical and a horizontal line (this is referred to as "reflection constraint" in table 2). Based on these considerations, it is a rather mechanical task to derive all possible singularities.

All singularities for one- and two-plane calibration and for different levels of prior knowledge are described in ta-

bles 1 and 2. We reveal which of the intrinsic parameters can/can't be estimated uniquely. The tables contain columns for τf and f which stand for the calibrated focal length, measured in horizontal and vertical pixel dimensions respectively. In some cases it is possible to compute, e.g. τf, but not to compute τ or f individually.

A general observation is that a plane parallel to the image plane, allows to estimate the aspect ratio, but no other parameters. Generally speaking, the more regular the geometric configuration is, the more singularities may occur.

Prior	Pos. of cal. plane	τ	τf	f	u_0	v_0
u_0, v_0	Parallel to image pl.	+	-	-	+	+
	Perpend. to image pl.					
	parallel to u axis	-	+	-	+	+
	parallel to v axis	-	-	+	+	+
	else	-	-	-	+	+
	Else					
	parallel to u axis	-	-	-	+	+
	parallel to v axis	-	-	-	+	+
	else	+	+	+	+	+
τ	Parallel to u axis	+	-	-	+	-
	Parallel to v axis	+	-	-	-	+
	Else	+	-	-	-	-
τ, u_0, v_0	Parallel to image pl.	+	-	-	+	+

Table 1. Singularities of calibration from one plane. Here, parallelism to the image plane's u or v axis means parallelism in 3-space.

6 Experimental Results

We performed a number of experiments with simulated and real data, in order to quantify the performance of our method, to motivate its use in applications described in the following section and to exhibit singularities.

6.1 Simulated Experiments

For our simulated experiments, we used a diagonal calibration matrix with $f = 1000$ and $\tau = 1$. Calibration is performed using the projections of the 4 corner points of squares of size 40cm. The distance of the calibration squares to the camera is chosen such that the projections roughly fill the 512×512 image plane. The projections of the corner points are perturbed by centered Gaussian noise of 0 to 2 pixels variance.

We only display graphs showing the behavior of our algorithm with respect to other parameters than noise; note however that in all cases, the behavior with respect to noise is nearly perfectly linear. The data in the graphs shown stem from experiments with a noise level of 1 pixel. The errors shown are absolute ones (scaled by 1000 for the aspect ratio). Each point in a graph represents the *median* error of 1000 random experiments. The graphs of the mean errors are similar but less smooth.

One plane seen in one view. The scenario and results are shown in the upper part of figure 1. Calibration is performed for different orientations of the square, ranging from $0°$ (parallel to the image plane) to $90°$ (perpendicular to the image plane). Given the principal point, we calibrated the aspect ratio and the focal length. An obvious observation is the presence of singularities: the error of the aspect ratio increases considerably as the calibration square tends to be perpendicular to the image plane ($90°$). The determination of the focal length is impossible for the extreme cases of parallelism and perpendicularity. Note that these observations are all predicted by table 1. In the range of $[30°, 70°]$, the relative error for the focal length is below 1%, while the aspect ratio is estimated correctly within 0.01%.

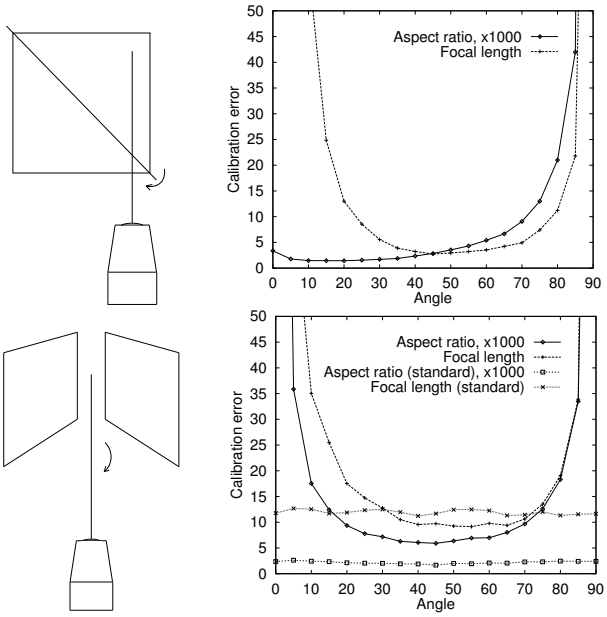


Figure 1. Simulation scenarios and results.

Two planes seen in one view. Calibration is performed with a camera rotating about its optical axis by $0°$ to $90°$. Two planes with an opening angle of $90°$ are observed (cf. lower part of figure 1). Plane-based calibration is now done without any prior knowledge of intrinsic parameters. For comparison, we also calibrate with a standard method [3], using full 3D coordinates of the corner points as input.

The standard calibration approach is insensitive to rotation about the optical axis. As for the plane-based method, the singularities for the estimation of the aspect ratio and the focal length for angles of $0°$ and $90°$ are predicted by table 2. As for the intermediate range of orientations, the estimation of the aspect ratio by the plane-based method is 3 to 4 times worse than with the standard approach, although it is still quite accurate. As for the focal length, the plane-based estimate is even slightly better between $30°$ and $70°$. The error graphs for u_0 and v_0 are not shown; for both methods

Prior	Position of calibration planes	τ	τf	f	u_0	v_0
None	One plane is parallel to the image plane	cf. case of known τ in table 1				
	General case of planes satisfying reflection constraint (see caption)	-	-	-	-	-
	Both planes are parallel to the u axis	-	-	-	+	-
	Same absolute incidence angle with respect to image plane	-	-	-	+	+
	Both planes are parallel to the v axis	-	-	-	-	+
	Same absolute incidence angle with respect to image plane	-	-	-	+	+
	Vanishing lines intersect "above" the principal pt. i.e. at a point $(u_0, v, 1)$	-	-	-	+	-
	Vanishing lines intersect at a point $(u, v_0, 1)$	-	-	-	-	+
	Both planes are perpendicular to image (and satisfy reflection constraint)	-	-	-	+	+
u_0, v_0	At least one plane is parallel to the image plane	cf. case of known τ, u_0, v_0 in table 1				
	Both planes are perpendicular to the image (and satisfy reflection constr.)	-	-	-	+	+
τ	One plane is parallel to the image plane	cf. case of known τ in table 1				
τ, u_0, v_0	One plane is parallel to the image plane	cf. case of known τ, u_0, v_0 in table 1				

Table 2. Singularities of calibration from two planes. The cases of parallel planes are not displayed, but may be consulted in the appropriate parts of table 1 on one-plane calibration. In all configurations not represented here, all intrinsic parameters can be estimated. By "reflection constraint" we mean that the vanishing lines of the two planes are reflections of each other by both a vertical and a horizontal line in the image.

they are nearly horizontal (i.e. there is no singularity), the errors of the plane-based estimation being about 30% lower than with the standard approach.

6.2 Calibration Grid

We calibrated a camera from images of a 3D calibration grid with targets arranged in three planes (cf. figure 2). For comparison, calibration was also carried out using a standard method [3]. We report the results of two experiments. First, 4 images were taken from different positions, but with fixed calibration. The camera was calibrated from single views in different modes: standard calibration using all points or points from two planes only, plane-based calibration from one, two or three planes with different levels of prior knowledge (cf. table 3). Prior values were taken from the results of standard calibration.

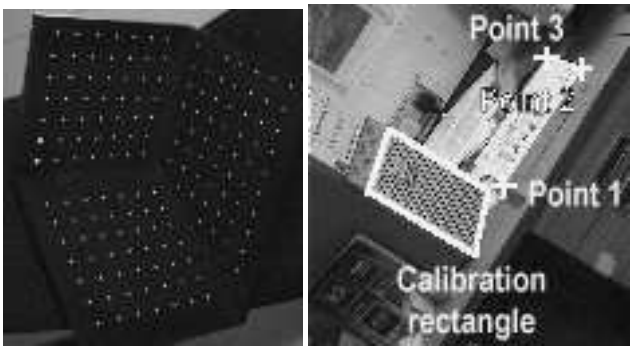


Figure 2. Calibration grid and lab scene.

Table 3 shows the mean and standard deviation of the results for the focal length, computed over the 4 views and over all combinations of planes. We note that even the one-plane method gives results very close to those of the standard method that uses all points and their full 3D coordinates. The precision of the plane-based results is lower than for full standard calibration, though comparable to standard calibration using two planes. The results are very accurate despite the proximity to singular configurations. This may be attributed to the high accuracy of target extraction.

Method	f
Standard calibration from three planes	1041.4 ± 0.6
Standard calibration from two planes	1042.1 ± 3.3
One plane, u_0, v_0 known	1044.5 ± 9.0
One plane, τ, u_0, v_0 known	1041.2 ± 3.7
Two planes, nothing known	1043.6 ± 4.7
Two planes, τ known	1040.7 ± 2.7
Two planes, u_0, v_0 known	1040.2 ± 2.5
Two planes, τ, u_0, v_0 known	1040.3 ± 2.1
Three planes, nothing known	1039.9 ± 0.7

Table 3. Results for calibration grid.

Method	Focal lengths				
Standard	714.7	1041.4	1386.8	1767.4	2717.2
Planes	709.9	1042.7	1380.2	1782.8	2702.0

Table 4. Results for variable focal length.

For the second experiment, we took images at 5 different zoom positions. The camera was calibrated using the 5×3 planes simultaneously, where for each zoom position an individual focal length and principal point were estimated. Table 4 shows the results for the focal lengths (a value of 1000 corresponds to about 7.5mm), compared to those of standard calibration, averaged over single views. The deviation increases with the focal length but stays below 1%.

6.3 Lab Scene

A pair of images of an ordinary lab scene were taken. A rectangular part of a computer tower (cf. figure 2) was

used for calibration. Subsequently, the pose of the views with respect to the calibration plane was determined. The three points shown in figure 2 were triangulated and their 3D distances measured and compared to hand-measured ones. The differences for the pairs (1,2), (1,3) and (2,3) were 4mm, 3mm and 0mm respectively, for absolute distances of 275mm, 347mm and 214mm. These results are about as good as we might expect: the edges of the rectangular patch are rounded, thus not reliably extracted in the images. The measured point distances are "extrapolated" from this rectangle, thus amplifying the errors of edge extraction. From the views' calibration and pose, we computed the epipolar geometry and found that the distance of points to corresponding epipolar lines was about 1 pixel, even at the borders of the images.

This simple experiment highlights two issues. First, besides calibrating the views, we readily obtain their pose in a *metric* 3D frame. Second, we obtain reasonable estimates of matching constraints, potentially for distant views.

7 Applications

Cheap Calibration Tool. Planar patterns are easy to produce, while enabling a reasonably reliable calibration.

Ground Plane Calibration. We have successfully performed experiments with images of traffic scenes. Ground plane calibration from road markers is used to restrict the pose of vehicles to be detected and tracked.

Reconstruction of Piecewise Planar Objects from Single Views. Using geometrical constraints such as coplanarity, parallelism, right angles etc., 3D objects may be reconstructed from a single view (see e.g. [10]). Our calibration method requires knowledge of the metric structure of planes. This requirement may be relaxed by simultaneously determining calibration and plane structure, e.g. one view of a rectangle allows to determine the focal length and the ratio of the rectangle's edge lengths. We are using this in combination with the mentioned geometrical constraints to reconstruct objects from a single image.

Reconstruction of Indoor Scenes. Our calibration method is the central part of ongoing work on a system for interactive multi-view 3D reconstruction of indoor scenes, similar in spirit to the approaches presented in [10, 11]. The main motivation for using plane-based calibration is to make a compromise between requirements on flexibility, user interaction and implementation cost. We achieve flexibility by not requiring off-line calibration: our calibration patterns, planar objects, are omnipresent in indoor scenes. The amount of user interaction is rather little: we usually use rectangles as calibration objects; they have to be delineated in images and their edge lengths measured. By identifying planar patterns across distant views, we not only can simultaneously calibrate many views but also compute a global initial pose of many views to bootstrap, e.g. wide baseline matching. This scheme relies on methods that are relatively simple to implement and might provide a useful alternative to completely automatic techniques such as [9] that are more flexible but more difficult to realise.

Augmented Reality. A nice and useful application of plane-based calibration and pose estimation is presented in [6]. Rectangular plates are used to mark the position of non planar objects to be added to a video sequence, which is in some way a generalisation of "overpainting" planar surfaces in videos by homography projection of a desired pattern. Plane-based methods may also be used for blue screening; attaching calibration patterns on the blue screen allows to track camera pose and calibration and thus to provide input for positioning objects in augmented reality.

8 Conclusion

We presented a general and easy to implement plane-based calibration method that is suitable for calibrating variable intrinsic parameters and that copes with any number of calibration planes and views. Experimental results are very satisfactory. For the basic cases of one or two planes, we gave an exhaustive list of singularities. Several applications of plane-based calibration were described. An analytical error analysis might be fruitful, i.e. examining the influence of feature extraction errors on calibration accuracy.

An extended version of this paper can be retrieved at `http://www.cvg.cs.reading.ac.uk/~pfs/plane.ps.gz`.

References

[1] A. Criminisi, I. Reid, A. Zisserman, "Duality, Rigidity and Planar Parallax," *ECCV*, pp. 846-861, 1998.

[2] O. Faugeras, "Stratification of Three-Dimensional Vision: Projective, Affine and Metric Representations," *Journal of the Optical Society of America A*, 12, pp. 465-484, 1995.

[3] O. Faugeras, G. Toscani, "Camera Calibration for 3D Computer Vision," *Int. Workshop on Machine Vision and Machine Intelligence*, pp. 240-247, 1987.

[4] R.J. Holt, A.N. Netravali, "Camera Calibration Problem: Some New Results," *CVIU*, 54 (3), pp. 368-383, 1991.

[5] A. Jennings, J.J. McKeown, *Matrix Computation,* 2nd edition, Wiley, 1992.

[6] M. Jethwa, A. Zisserman, A. Fitzgibbon, "Real-time Panoramic Mosaics and Augmented Reality," *BMVC*, pp. 852-862, 1998.

[7] R.K. Lenz, R.Y. Tsai, "Techniques for Calibration of the Scale Factor and Image Center for High Accuracy 3-D Machine Vision Metrology," *PAMI*, 10 (5), pp. 713-720, 1988.

[8] F. Pedersini, A. Sarti, S. Tubaro, "Multi-Camera Acquisitions for High-Accuracy 3D Reconstruction," *SMILE Workshop*, pp. 124-138, 1998.

[9] M. Pollefeys, R. Koch, M. Vergauwen, L. Van Gool, "Metric 3D Surface Reconstruction from Uncalibrated Image Sequences," *SMILE Workshop*, pp. 139-154, 1998.

[10] H.-Y. Shum, R. Szeliski, S. Baker, M. Han, P. Anandan, "Interactive 3D Modeling from Multiple Images Using Scene Regularities," *SMILE Workshop*, pp. 236-252, 1998.

[11] R. Szeliski, P.H.S. Torr, "Geometrically Constrained Structure from Motion: Points on Planes," *SMILE Workshop*, pp. 171-186, 1998.

[12] R.Y. Tsai, "A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses," IEEE *Journal of Robotics and Automation*, 3 (4), pp. 323-344, 1987.

[13] B. Triggs, "Autocalibration from planar scenes," *ECCV*, pp. 89-105, 1998.

[14] G.-Q. Wei, S.D. Ma, "A Complete Two-Plane Camera Calibration Method and Experimental Comparisons," *ICCV*, pp. 439-446, 1993.

Algorithms for Plane-Based Pose Estimation

Peter Sturm*
INRIA Rhône-Alpes
655 Avenue de l'Europe
38330 Montbonnot St Martin, France
Peter.Sturm@inrialpes.fr

Abstract

We present several methods for the estimation of relative pose between planes and cameras, based on projections of sets of coplanar features in images. While such methods exist for simple cases, especially one plane seen in one or several views, the aim of this paper is to propose solutions for multi-plane multi-view situations, possibly with little overlap. We propose a factorization-based method for the general case of n planes seen in m views. A mechanism for computing missing data, i.e. when one or several of the planes are not visible in one or several of the images, is described. Experimental results for real images are shown.

1. Introduction

The work presented in this paper is part of a project on multi-view 3D modeling based on scene regularities. Scene regularities like coplanarity of points or lines, perpendicularity and parallelism of lines or planes etc. are increasingly used for interactive 3D modeling of man-made scenes [1, 2, 3, 12, 13, 15]. Typically, the entire scene (often a building) is depicted by hand in one or several images; geometric constraints representing scene regularities enable a 3D reconstruction of the scene. This approach is feasible and gives good results if the scene consists of a limited number of "primitives" and if its geometry can be described well enough by geometric constraints like the ones mentioned.

If the environment to be modeled is large and cluttered, it is usually not feasible to depict all the primitives needed for a 3D model. Also, useful geometric constraints might often only be provided for a fraction of the environment. In such circumstances, one natural solution for 3D modeling is triangulation, based on feature correspondences obtained by image matching. Beside the matching, camera calibration and relative camera pose have to be obtained. A complete automatization of this process is of course desirable, but it is questionable if current systems are performing well enough for cluttered large scenes. Also, there will always persist a certain failure rate; so, a user might prefer to trade a limited amount of interaction for a higher reliability of the results.

*This work is partially supported by the EPSRC funded project GR/K89221 (Vector).

We follow a different approach, as described in the following. Given a set (or a sequence) of images of an environment, we first want to use scene regularities (and associated features depicted in images by a user) to calibrate the views and estimate their relative pose. Once this is achieved, the calibration and pose information give us multi-view constraints for automatically matching and triangulating other features than those used to capture the scene regularities.

Attractive "primitives" for calibration and pose estimation are planar objects with known metric structure: each image of such an object provides two constraints on calibration and, if calibration can be fully determined, relative pose up to two solutions in general [6]. Especially rectangles are very useful since determining their metric structure is done by simply measuring their edge lengths and since they abound in man-made environments.

Pose estimation from planar objects turns out to be rather harder than camera calibration: calibration constraints based on the projection of planar objects with known metric structure [8, 9, 14, 17, 18, 19] can be accumulated over many images. As for pose estimation however, the goal is to obtain relative camera (and plane) pose in a *global* reference frame: estimation of relative pose of two cameras seeing the same plane is rather easy (see e.g. [6] and references therein for algorithms), but estimating simultaneously the pose of m cameras, each one seeing one or only a few of n planes, is not trivial. We are not aware of general methods for this task in the literature, although it is quite probable that developments have been made in the photogrammetric community. However, photogrammetric techniques are often designed for strong camera network geometries or for situations where at least approximate pose information is already available.

The paper is organized as follows. The problem of multi-view multi-plane pose estimation is formulated in §2. A method for the basic one-view one-plane case is given in §3. A factorization-based method for the multi-view multi-plane situation is presented in §4. Experimental results are shown in §5, followed by conclusions.

Notation. As already mentioned above, for a 3×3 matrix A, $\bar{\mathsf{A}}$ is the 3×2 submatrix consisting of its first two columns. The sign $\sim$ means equality up to scale (for vectors or matrices). The symbol $\mathtt{I}$ denotes the identity matrix.

2. Problem Formulation

The problem at hand is to estimate the relative pose of m cameras and n planes, based on projections of the planes (i.e. features on the planes) in (some of) the cameras. In the following, we only deal with point features, but our ideas may be extended to other features. We suppose that the metric structure of the planes is known, i.e. that the coordinates of points on a plane are known in some Euclidean reference frame attached to the plane. Using this information, the cameras may be calibrated, e.g. using our algorithm described in [14]. In the following we thus suppose that the cameras are calibrated.

We now describe the coordinate transformations that lead from 2D point coordinates of points on a plane to the coordinates of their projections in an image. Let $\mathbf{Q}_{jk}$ be the kth point on the jth plane, given by coordinates (X_{jk}, Y_{jk}). Let the position and orientation of the jth plane (in the sequel simply called the plane's *pose*) be given by a rotation matrix S_j and a translation vector $\mathbf{v}_j$ with respect to some global 3D world reference frame, such that the coordinates of $\mathbf{Q}_{jk}$ in that global frame are:

$$\mathbf{Q}^w_{jk} = \begin{pmatrix} \mathsf{S}_j & \mathbf{v}_j \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \begin{pmatrix} X_{jk} \\ Y_{jk} \\ 0 \\ 1 \end{pmatrix} \quad .$$

Let the pose of camera i be given by R_i and $\mathbf{t}_i$, such that the coordinates of $\mathbf{Q}_{jk}$ in the local camera frame are:

$$\mathbf{Q}^c_{ijk} = \begin{pmatrix} \mathsf{R}_i & \mathbf{t}_i \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \mathbf{Q}^w_{jk} \quad .$$

The camera model used throughout the paper is perspective projection, i.e. the coordinates of the projected point are:

$$\begin{aligned} \mathbf{q}_{ijk} &\sim \begin{pmatrix} \mathsf{K}_i & \mathbf{0} \end{pmatrix} \mathbf{Q}^c_{ijk} \\ &\sim \mathsf{K}_i \begin{pmatrix} \mathsf{R}_i\mathsf{S}_j & \mathsf{R}_i\mathbf{v}_j + \mathbf{t}_i \end{pmatrix} \begin{pmatrix} X_{jk} \\ Y_{jk} \\ 0 \\ 1 \end{pmatrix} \quad (1) \end{aligned}$$

where K_i is the *calibration matrix* of view i.

The aim of the algorithms presented in this paper is to determine camera and plane pose, i.e. the $\mathsf{R}_i, \mathbf{t}_i, \mathsf{S}_j$ and $\mathbf{v}_j$, from the calibration matrices K_i, the metric structure of the planes, represented by the (X_{jk}, Y_{jk}), and the image points $\mathbf{q}_{ijk}$. The computations are based on homographies for camera–plane pairs that represent the perspective projections of the planes onto the image planes. The homography H_{ij} for camera i and plane j is (this is simply the matrix of equation (1), without the third column):

$$\mathsf{H}_{ij} \sim \mathsf{K}_i \begin{pmatrix} \left(\mathsf{R}_i\bar{\mathsf{S}}_j\right)_{3\times 2} & \left(\mathsf{R}_i\mathbf{v}_j + \mathbf{t}_i\right)_{3\times 1} \end{pmatrix}$$

where $\bar{\mathsf{S}}_j$ is the 3×2 submatrix of S_j consisting of its first two columns. Since calibration is known, we may compute

$$\mathsf{M}_{ij} \sim {\mathsf{K}_i}^{-1}\mathsf{H}_{ij} \sim \begin{pmatrix} \left(\mathsf{R}_i\bar{\mathsf{S}}_j\right)_{3\times 2} & \left(\mathsf{R}_i\mathbf{v}_j + \mathbf{t}_i\right)_{3\times 1} \end{pmatrix} \quad .$$

The algorithms described in the following determine pose using these homographies M_{ij}. The basic constraint used is that the first two columns of any M_{ij} are the first two columns of a rotation matrix, up to scale.

The homographies are computed from point matches between planes and the images, by a linear method analogous to the 8-point method for the fundamental matrix [4].

3. Basic Case: One Plane Seen in One View

Suppose the view is calibrated and the homography H (we omit the subscripts in this section) has been computed. As shown above, we can compute the matrix

$$\mathsf{M} \sim \begin{pmatrix} \left(\mathsf{R}\bar{\mathsf{S}}\right)_{3\times 2} & \left(\mathsf{R}\mathbf{v} + \mathbf{t}\right)_{3\times 1} \end{pmatrix}$$

Of course, we can only compute *relative* pose, i.e. a rotation matrix T and a vector $\mathbf{w}$ such that:

$$\begin{aligned} \mathsf{T} &= \mathsf{R}\mathsf{S} && (2) \\ \mathbf{w} &= \mathsf{S}^\mathsf{T}\mathbf{v} + \mathsf{S}^\mathsf{T}\mathsf{R}^\mathsf{T}\mathbf{t} && (3) \end{aligned}$$

in which case we have:

$$\mathsf{M} \sim \mathsf{T} \begin{pmatrix} 1 & 0 & \\ 0 & 1 & \mathbf{w} \\ 0 & 0 & \end{pmatrix} \quad .$$

In the absence of noise, the solution for T and $\mathbf{w}$, given M, is simple: first, scale M such that its first two columns have unit norm (there are two such scale factors, linked by sign reversal). The first two columns of M are then adopted as the first two columns of T. The third column of T is computed as the cross product of the first two (plus possibly a scaling by -1 to ensure that $\det \mathsf{T} = +1$). Then, $\mathbf{w}$ is obtained as the third column of $\left(\mathsf{T}^\mathsf{T}\right)\mathsf{M}$ (M scaled as above). There are two solutions in general (due to the existence of two scale factors for M), which are related as follows:

$$\mathsf{T}' = \mathsf{T} \begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad \mathbf{w}' = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \mathbf{w}$$

In practice, it is easy to disambiguate between these two: the two solutions for $\mathbf{w}$ correspond to optical centers on either side of the plane. Thus, it is sufficient to know which side of a plane is visible (assuming that a planar object has only one visible side). We achieve this e.g. by giving the coordinates of points on the plane in a reference frame whose *positive Z* axis (the axis perpendicular to the plane) shows toward the "visibility half-space", and then choosing the pose whose $\mathbf{w}$ has a negative third coefficient, or vice versa.

If noise is present, M will not be exactly of the form shown above, and we have to determine some "best" T and $\mathbf{w}$. As usual in this case, the criterion used is the Frobenius matrix norm $\|\cdot\|_F$ (root of sum of squared matrix coefficients). Concretely, the problem may be formulated as:

$$\min_{\mathsf{T},\mathbf{w},\lambda} \|\lambda\mathsf{M} - \mathsf{T}\begin{pmatrix}1 & 0 & \\ 0 & 1 & \mathbf{w} \\ 0 & 0 & \end{pmatrix}\|_F^2 \quad \text{subject to } \mathsf{T}^\mathsf{T}\,\mathsf{T} = \mathtt{I} \quad (4)$$

It is easy to show, along the lines of [5], that the optimal solution for the rotation T can be obtained independently from λ and $\mathbf{w}$, and that these may then be obtained from T.

The optimal solution for T is obtained by solving the following subproblem:

$$\min_{\bar{\mathsf{T}}} \|\bar{\mathsf{M}} - \bar{\mathsf{T}}\|_F^2 \quad \text{subject to} \quad \left(\bar{\mathsf{T}}^\mathsf{T}\right)\bar{\mathsf{T}} = \mathtt{I}_2 \ . \quad (5)$$

In words, we determine the rotation matrix T whose first two columns are closest to those of M, in the sense of the Frobenius norm. Note that this is different from the formulation chosen by Zhang [19], who finds the rotation matrix closest to the 3×3 matrix consisting of $\bar{\mathsf{M}}$ and a third column computed (more or less) as the cross product of these first two columns. It can be shown that this approach does not solve the original problem (4) optimally.

Problem (5) is easily solved using Singular Value Decomposition (SVD). Let $\bar{\mathsf{M}}_{3\times2} = \mathsf{U}_{3\times2}\Sigma_{2\times2}\mathsf{V}_{2\times2}^\mathsf{T}$ be the SVD of $\bar{\mathsf{M}}$. The optimal "amputated" rotation $\bar{\mathsf{T}}$ is then:

$$\bar{\mathsf{T}} = \mathsf{U}\mathsf{V}^\mathsf{T} \ .$$

The third column of T may then be computed in the same manner as described above for the noise free case.

Having solved for T, the optimal scale factor λ and vector $\mathbf{w}$ are obtained as:

$$\lambda = \frac{\mathrm{trace}(\bar{\mathsf{T}}^\mathsf{T}\ \bar{\mathsf{M}})}{\mathrm{trace}(\bar{\mathsf{M}}^\mathsf{T}\ \bar{\mathsf{M}})} = \frac{\sum_{i=1}^{3}\sum_{j=1}^{2} T_{ij}M_{ij}}{\sum_{j=1}^{2} M_{ij}^2}$$

$$\mathbf{w} = \left(\mathsf{T}^\mathsf{T}\right)\mathsf{M}\begin{pmatrix}0\\0\\\lambda\end{pmatrix} \ .$$

Again, there are two solutions in general which can be disambiguated as discussed for the noise free case.

We do not claim that the method presented in this section is original, but describe it here since it is an important part of the method described in the next section.

4. Multi-View Multi-Plane Pose

The method of the previous section may be used to determine the relative pose for m cameras observing a single plane or n planes being observed by a single camera, by applying it for individual camera–plane pairs and stitching together the results. However, if more than one camera observe more than one plane, the situation is more complicated. In the following, we present a method that uses the relative pose information obtained for individual camera–plane pairs simultaneously to determine global relative pose of cameras and planes. We first assume that all planes are visible in all cameras. The case of missing data is dealt with in §4.2.

In the following, we first compute the rotational part of the pose, followed in §4.3 by the translations.

4.1. Rotational Part of Pose

Let T_{ij} represent the rotational part of the relative pose between camera i and plane j, as computed using the method of §3. We may group all equations of type (2) for camera–plane pairs in one single equation system:

$$\underbrace{\begin{pmatrix}\mathsf{T}_{11} & \mathsf{T}_{12} & \cdots & \mathsf{T}_{1n}\\ \mathsf{T}_{21} & \mathsf{T}_{22} & \cdots & \mathsf{T}_{2n}\\ \vdots & \vdots & \ddots & \vdots\\ \mathsf{T}_{m1} & \mathsf{T}_{m2} & \cdots & \mathsf{T}_{mn}\end{pmatrix}}_{\mathsf{W}} = \underbrace{\begin{pmatrix}\mathsf{R}_1\\ \mathsf{R}_2\\ \vdots\\ \mathsf{R}_m\end{pmatrix}}_{\mathsf{R}} \underbrace{\begin{pmatrix}\mathsf{S}_1 & \mathsf{S}_2 & \cdots & \mathsf{S}_n\end{pmatrix}}_{\mathsf{S}} \quad (6)$$

This equation motivates the idea of solving for the R_i and S_j by factorization: the matrix W is (in absence of noise) of rank 3 and its three non zero singular values are all equal. If noise is present, we may estimate the matrix W' with these properties that is closest to W in the sense of the Frobenius norm, as follows. Let $\mathsf{W} = \mathsf{U}\Sigma\mathsf{V}^\mathsf{T}$ be the SVD of W. Let U' (V') be the matrix consisting of the first three columns of U (V). The optimal W' is then given by $\mathsf{W}' = \mathsf{U}'\mathsf{V}'^\mathsf{T}$.

Since U' and V' have the same dimensions as R and S in equation (6), we may try to extract the rotation matrices R_i and S_j from them. The factorization does not guarantee that the 3×3 submatrices of U' and V' are valid rotations. Thus, we determine the R_i and S_j as the rotation matrices that are closest to the according submatrices in U' and V'. This is described in [5].

One issue to discuss is the possibility of ambiguities in the factorization, i.e. the existence of matrices A such that $(\mathsf{U}'\mathsf{A})\left(\mathsf{A}^{-1}\mathsf{V}'^\mathsf{T}\right)$ is a valid solution for our problem. Since the two matrices resulting from the factorization have to be collections of rotation matrices, it can be shown that the only possible ambiguities correspond to A being a rotation matrix. This is no problem here, since naturally the R_i and S_j can only be determined up to a global rotation.

Another important issue is numerical condition. The matrix to be factorized is a collection of rotation matrices, thus automatically well balanced, i.e. its coefficients are in average of the same magnitude. Also, the three non zero singular values are equal (in the noise free case), suggesting that even in the noisy case the condition should be good.

4.2. Computing Missing Data

Our method suffers, as all factorization approaches, from the problem of missing data: in practice we will often meet the case where several planes are not visible in several cameras, thus the matrix to be factorized is not entirely defined. Solutions to this problem have been proposed [7, 11, 16]; these are either of an ad hoc or heuristic nature or rely on an initialization by some (unclear) means. We propose another ad hoc approach for our problem. Our situation is not too bad, since the missing entries in the matrix to be factorized are 3×3 rotation matrices, thus providing some useful constraints for their determination.

The computation of the missing rotation T_{ij} between a camera i and a plane j is based on the following observation. If we know, for some i' and j', the rotations $\mathsf{T}_{i'j'}$, $\mathsf{T}_{i'j}$ and $\mathsf{T}_{ij'}$, then we can compute T_{ij} as (cf. equation (2)):

$$\mathsf{T}_{ij} = \mathsf{T}_{ij'} \left(\mathsf{T}_{i'j'}^{\mathsf{T}}\right) \mathsf{T}_{i'j}$$

If several such combinations are available, we may compute T_{ij} as their "average". To do so, we simply add up the individual estimations of T_{ij} to a matrix A and compute T_{ij} as the rotation matrix that best approximates A, in the sense of the Frobenius norm (see [5]).

Computation of missing data has usually to be done in a cumulative manner, i.e. some of the T_{ij} can only be computed using other matrices that were missing at the outset but have been computed as shown above.

4.3. Translational Part of Pose

Having computed the rotational part of camera and plane pose, the translational part may be determined as follows. Let $\mathbf{w}_{ij}$ represent the translational part of the relative pose between camera i and plane j, as computed using the method in §3. From equation (3), we have:

$$\mathbf{w}_{ij} = \mathsf{S}_j^{\mathsf{T}} \mathbf{v}_j + \mathsf{S}_j^{\mathsf{T}} \mathsf{R}_i^{\mathsf{T}} \mathbf{t}_i \quad .$$

Since we know S_j, we may compute

$$\mathbf{w}'_{ij} = \mathsf{S}_j \mathbf{w}_{ij} = \mathbf{v}_j + \mathsf{R}_i^{\mathsf{T}} \mathbf{t}_i \quad .$$

A cost function for estimating the $\mathbf{v}_j$ and $\mathbf{t}'_i = \mathsf{R}_i^{\mathsf{T}} \mathbf{t}_i$ is then:

$$c = \sum_{i,j} \|\mathbf{w}'_{ij} - \mathbf{v}_j - \mathbf{t}'_i\|^2 \qquad (7)$$

where summation is over all available camera–plane pairs.

The partial derivatives of criterion (7) with respect to the kth entry of $\mathbf{v}_j$ and the pth entry of $\mathbf{t}'_i$ respectively are:

$$\frac{\partial c}{\partial v_{jk}} = 2 \sum_i \left(w'_{ijk} - v_{jk} - t'_{ik}\right) \qquad (8)$$

$$\frac{\partial c}{\partial t'_{ip}} = 2 \sum_j \left(w'_{ijp} - v_{jp} - t'_{ip}\right) \quad . \qquad (9)$$

Criterion (7) may be minimized by solving for the common roots of the partial derivatives. This can be done by solving the following simple linear equation system (shown for the case where all planes are seen in all views):

$$\left(\begin{array}{ccc|ccc} m\mathtt{I} & & & \mathtt{I} & \cdots & \mathtt{I} \\ & \ddots & & \vdots & \ddots & \vdots \\ & & m\mathtt{I} & \mathtt{I} & \cdots & \mathtt{I} \\ \hline \mathtt{I} & \cdots & \mathtt{I} & n\mathtt{I} & & \\ \vdots & \ddots & \vdots & & \ddots & \\ \mathtt{I} & \cdots & \mathtt{I} & & & n\mathtt{I} \end{array}\right) \left(\begin{array}{c} \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_n \\ \hline \mathbf{t}'_1 \\ \vdots \\ \mathbf{t}'_m \end{array}\right) = \left(\begin{array}{c} \sum_i \mathbf{w}'_{i1} \\ \vdots \\ \sum_i \mathbf{w}'_{in} \\ \hline \sum_j \mathbf{w}'_{1j} \\ \vdots \\ \sum_j \mathbf{w}'_{mj} \end{array}\right)$$

We use a special method to solve this sparse system. The solution is of course only determined up to translation: adding a 3-vector to all the $\mathbf{v}_j$ and subtracting it from all the $\mathbf{t}'_i$ does not affect criterion (7).

4.4. Complete Algorithm

1. Compute homographies between planes and images.
2. If the cameras are not calibrated yet, calibrate them using one of the methods in [8, 14, 17, 18, 19].
3. Estimate relative pose between pairs of planes and cameras as described in §3.
4. Compute missing data as described in §4.2.
5. Estimate the rotational part of global relative pose by factorization as described in §4.1.
6. Estimate the translational part of pose (cf. §4.3).
7. Optional, but recommended: simultaneous (non-linear) optimization of pose and calibration parameters (including distortion). Not explained in detail here (lack of space), but rather straightforward to implement.

5. Experimental Results

We have tested our methods with image sequences of different types. First, images of a calibration grid were used to evaluate their performance with respect to the number of images used. Second, planar patterns printed on paper were attached to all the walls of a room. This scene is a test for our methods in the case of a high amount of missing data. The third image sequence is of the same type as the second one, however the planar objects used for calibration and pose estimation were part of the scene (rectangular objects like windows, doors, computer screens etc.).

5.1. Calibration Grid

Images of a calibration grid (see figure 2) were taken with a Canon MV-1 Camcorder. For different zoom positions, 4 images each were taken from different positions. The input to our methods were the coordinates of circular targets in each of the three planes of the grid, and the corresponding image coordinates. For each zoom setting, a total of 12 homographies could be computed. From these, the camera was calibrated and pose estimated using the methods in §3 and §4, followed by non-linear optimization.

In figure 1, some results are presented for the zoom position corresponding to shortest focal length (and largest optical distortion). The upper two curves show the absolute errors (in degrees) of the angles between the three planes of the grid, computed from the estimated pose. With the minimum case of a single view, the error is about 1.4° for both the "linear" method (§4) and after optimization ("Linear+LM" in the graph). Adding views leads to an error of about 1° for the linear method (which seems to be a limit here, maybe due to the neglection of optical distortion) and a linear decrease of the error after optimization, reaching a tenth of a degree when four views are used.

The lower two curves show the average distance errors for the full 3D reconstruction of the calibration grid. Since we know the coordinates of the targets in each of the three planes of the object, and we estimate the pose of the planes, we can obtain a full 3D reconstruction, i.e. full 3D coordinates of the targets. The error (residuals after alignment with the ground truth by rigid transformation) is practically constant and equal to a tenth of a percent, regardless of the number of images and optimization.

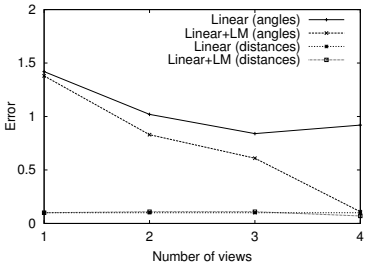


Figure 1. Errors of pose estimation. Absolute errors (degrees) for angles (upper curves). Relative errors (in %) for distances, see text.

5.2. Indoor Scenes

We took a set of about 400 images of an indoor scene (see examples in figure 2). The edges of 14 rectangular objects in the scene (windows, drawers, a door, blackboard, computer screens, etc., cf. figure 3) were measured, giving their metric structure. In 151 of the images, one or more of these planar objects were visible and in 84, two or more

Figure 2. Images of grid and indoor scene.

objects. In these images, the 4 corners of the objects were marked by hand. This is the input to our algorithms.

In a first step, the calibration method of [14] was applied to calibrate the 84 views simultaneously. Then, relative pose between each view and the objects seen in it was computed using the method of §3. From the totality of $84 \times 14 = 1176$ image–plane pairs, the relative pose of 218 pairs could be determined from the available images, i.e. the amount of missing data was about 81 %. Global pose was estimated using the algorithm of §4.4. The result was used to obtain a textured VRML model of the planar objects used for calibration and pose estimation (figure 3 shows a rendering).

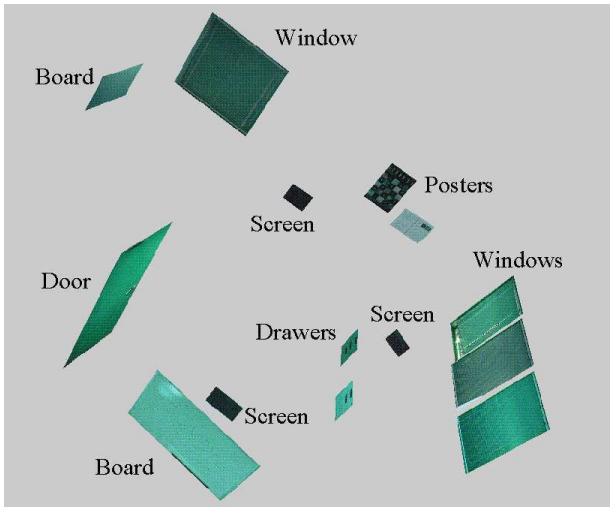


Figure 3. Rendering of a textured 3D model of the planar objects used for calibration and pose estimation.

Qualitatively, the reconstruction captures very well the shape of the room in which the images where taken. The accuracy of the reconstruction is not very high: angles between neighboring planar objects (the angles between the infinite planes supporting the objects) are in average estimated with an error of about 6°. This is rather weak as for photogrammetric standards. However, the global pose is good enough to think of using it for wide-baseline matching using adaptive windows: approximately knowing the relative pose between views, matching windows can be transferred via projective mappings computed from pose and calibration, based on the assumption of locally planar object surfaces. Initial matching experiments are encouraging.

Overall, we consider this experiment as a really hard test: the input data is rather minimal (4 points per plane) and poor (some of the objects were not really planar, extraction of features in the images was quite inaccurate, the objects appear usually very small in the images); the imaging geometry is weak (~ 80 % of missing data); no special illumination was used, etc. So, the accuracy of our results might be as good as one might expect under these conditions.

In a second experiment, a few planar patterns, printed on paper using a laser printer, were attached to the walls of a room. Owing to higher accuracy in feature extraction, the average error of angles between neighboring patterns was about 3°. The amount of missing data was again over 80 % and the patterns occupied only about 3 % of the images.

6. Conclusion and Perspectives

We have presented methods for plane-based pose estimation. Beside a method for the basic one-view one-plane case, a factorization-based method for the multi-view multi-plane case was presented.

Our experimental results suggest that our method may be applied successfully even when the amount of missing data is very high. In "calibration scenarios" the estimated pose can certainly be used as starting point for optimizing calibration and pose. However, the global goal of our work is not calibration but the 3D reconstruction of complicated man-made environments. Our thread of thought is that the process should be initialized by a limited amount of user interaction, followed by automatic processes. The type of user interaction described in this paper (depicting some salient objects in the images) enables a good camera calibration and an approximate global pose estimation. The recovered pose might be good enough to be used for wide baseline matching using adaptive windows (according to initial experiments). This is what we are currently working on. Our hope (and conviction) is that a few additional matches per image (beside the hand picked ones) should be enough to increase the quality of the pose by a sufficient amount in order to make e.g. voxel coloring approaches [10] for 3D reconstruction feasible.

Acknowledgement. We thank Bill Triggs for his help with the feature extraction for one of the experiments.

References

[1] R. Cipolla, D.P. Robertson, E.G. Boyer, "Photobuilder – 3D models of architectural scenes from uncalibrated images," *Conf. Multimedia Computing and Systems*, 25-31, 1999.

[2] A. Criminisi, I. Reid, A. Zisserman, "Duality, Rigidity and Planar Parallax," *ECCV*, pp. 846-861, June 1998.

[3] P.E. Debevec, C.J. Taylor, J. Malik, "Modeling and Rendering Architecture from Photographs: a Hybrid Geometry-and Image-Based Approach," *SIGGRAPH*, August 1996.

[4] R. Hartley, "In Defence of the 8-Point Algorithm," *ICCV*, pp. 1064-1070, June 1995.

[5] B.K.P. Horn, H.M. Hilden, S. Negahdaripour, "Closed-Form Solution of Absolute Orientation Using Orthonormal Matrices," *Journal Opt. Soc. America A*, Vol. 5, 1127-1135, 1988.

[6] R.J. Holt, A.N. Netravali, "Camera Calibration Problem: Some New Results," *CVIU*, Vol. 54, No. 3, 368-383, 1991.

[7] D. Jacobs, "Linear Fitting with Missing Data: Applications to Structure-from-Motion and to Characterizing Intensity Images," *CVPR*, pp. 206-212, June 1997.

[8] R.K. Lenz, R.Y. Tsai, "Techniques for Calibration of the Scale Factor and Image Center for High Accuracy 3-D Machine Vision Metrology," *PAMI*, Vol. 10, 713-720, 1988.

[9] D. Liebowitz, A. Zisserman, "Combining Scene and Auto-calibration Constraints," *ICCV*, 1999.

[10] S.M. Seitz, C.R. Dyer, "Photorealistic Scene Reconstruction by Voxel Coloring," *CVPR*, pp. 1067-1073, June 1997.

[11] H.Y. Shum, K. Ikeuchi, R. Reddy, "Principal Component Analysis with Missing Data and its Application to Polyhedral Object Modeling," *PAMI*, Vol. 17, 854-867, 1995.

[12] H.-Y. Shum, R. Szeliski, S. Baker, M. Han, P. Anandan, "Interactive 3D Modeling from Multiple Images Using Scene Regularities," *SMILE Workshop, Freiburg, Germany*, pp. 236-252, June 1998.

[13] P. Sturm, S.J. Maybank, "A Method for Interactive 3D Reconstruction of Piecewise Planar Objects from Single Images," *BMVC*, pp. 265-274, September 1999.

[14] P. Sturm, S.J. Maybank, "On Plane-Based Camera Calibration: A General Algorithm, Singularities, Applications," *CVPR, Fort Collins, CO*, pp. 432-437, June 1999.

[15] R. Szeliski, P.H.S. Torr, "Geometrically Constrained Structure from Motion: Points on Planes," *SMILE Workshop, Freiburg, Germany*, pp. 171-186, June 1998.

[16] C. Tomasi, T. Kanade, "Shape and Motion from Image Streams under Orthography: A Factorization Method," *International Journal on Computer Vision*, Vol. 9, No. 2, pp. 137-154, 1992.

[17] R.Y. Tsai, "A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses," IEEE *Journal of Robotics and Automation*, Vol. 3, No. 4, pp. 323-344, 1987.

[18] G.-Q. Wei, S.D. Ma, "A Complete Two-Plane Camera Calibration Method and Experimental Comparisons," *ICCV*, pp. 439-446, 1993.

[19] Z. Zhang, "Flexible Camera Calibration By Viewing a Plane From Unknown Orientations," *ICCV*, pp. 666-673, 1999.

Degenerate Cases and Closed-form Solutions for Camera Calibration with One-Dimensional Objects

Pär Hammarstedt†, Peter Sturm‡, Anders Heyden†

† Applied Mathematics Group, School of Technology and Society
Malmo University, 20506 Malmo, Sweden
par.hammarstedt@ts.mah.se, heyden@ts.mah.se

‡ MOVI group, INRIA Rhône-Alpes
38330 Montbonnot St Martin, France
peter.sturm@inrialpes.fr

Abstract

Camera Calibration with one-dimensional objects is based on an algebraic constraint on the image of the absolute conic. We will give an alternative derivation to this constraint, allowing a geometrical interpretation. From this we derive the degenerate cases, or critical motions, where the calibration algorithm will fail. We also show that constraints on the intrinsic parameters lead to simplified closed-form solutions and a reduced set of critical motions. A simulation and a real data experiment is performed to evaluate the accuracy of the calibration result for motions close to being critical.

1. Introduction

In computer vision, metric 3D reconstruction from images requires the camera to be calibrated. The main camera calibration techniques can be classified into five groups. In *3D reference object calibration* an object with known geometry is used [12, 3]. In *2D plane based calibration* planar patterns are used [10, 13]. *1D object calibration* is discussed in this paper. The remaining two groups are self-calibration, where point correspondences between images of an unknown scene are used [7, 6, 5, 3], and *motion constrained calibration*, where the camera is confined to some special kind of motion [1, 4, 8]. In some cases of camera motion, known as critical motions, the calibration algorithms will fail. This has been studied in detail for *3D reference object calibration* in [2] and for self-calibration in [9].

In this paper we aim to complete the theory of 1D object calibration by identifying the critical motions. We show how to reduce them when partial knowledge of the cameras calibration parameters is given. Camera calibration using one-dimensional (1D) objects was recently proposed in [14]. Here, the calibration object consists of a set of at least three collinear points. The motion of the object is constrained by one point being fixed. One advantage of using 1D objects for calibration are that 1D objects with known geometry are easy to construct. Another advantage is that in a multi-camera environment, all cameras can observe the entire calibration object simultaneously, which is a prerequisite for calibration and hard to obtain with 3- and 2-dimensional calibration objects. In practice, the 1D object can be constructed by marking three points on a stick.

The paper is organized as follows: In Section 2 a brief review of camera calibration with 1D objects is given. In Section 3 a geometrical interpretation of the calibration constraint is presented, from which the critical motions are identified in Section 4. Section 5 describes how simplified closed-form solutions reduce the critical motions. Section 6 validates the theoretical results by two sets of experiments.

2 Preliminaries

2.1 Notation

We will use the standard pin-hole camera model:

$$\lambda_p \underbrace{\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}_{\widetilde{\mathbf{m}}} = \underbrace{\underbrace{\begin{bmatrix} \gamma f & sf & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{bmatrix}}_{K} [\, R \,|\, -Rt \,]}_{P} \underbrace{\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}}_{\widetilde{M}} . \quad (1)$$

Here, f denotes the focal length, γ the aspect ratio, s the skew and (u_0, v_0) the principal point. These are called the *intrinsic parameters* and are contained in the upper-triangular *calibration matrix* K. Furthermore, R and t denote the relation between the camera coordinate system and the object coordinate system, where R is a rotation matrix and t a translation vector, i.e. a Euclidean transformation. P is the camera matrix and λ_p is the projective depth of $\widetilde{\mathbf{m}}$. A 2D point is denoted by either $\mathbf{m} = [x, y]^T$ or $\widetilde{\mathbf{m}} = [x, y, 1]^T$. A 3D point is denoted by either $M = [X, Y, Z]^T$ or $\widetilde{M} = [X, Y, Z, 1]^T$.

2.2 Camera Calibration with 1D Objects

We will now give a brief review of the theory for camera calibration with one-dimensional objects, following [14]. In the following, we often call the one-dimensional calibration object a "stick", for simplicity.

Refer to Figure 1 where point O is the camera center. Point A is fixed relative to the camera, and the length of the stick AB is

$$L = \|B - A\|. \tag{2}$$

The position of point C is given by

$$C = \lambda_A A + \lambda_B B, \tag{3}$$

where λ_A and λ_B are known. Without loss of generality we choose $R = I$ and $t = 0$, which implies that the optical center O is at the origin. Let the unknown depths of A, B and C be z_A, z_B and z_C, respectively. According to (1) we have $A = z_A K^{-1}\widetilde{\mathbf{a}}$ and similarly for B and C, so equation (3) gives

$$z_C\widetilde{\mathbf{c}} = z_A\lambda_A\widetilde{\mathbf{a}} + z_B\lambda_B\widetilde{\mathbf{b}}. \tag{4}$$

By performing cross products on both sides of (4) with $\widetilde{\mathbf{c}}$ and scalar products with $(\widetilde{\mathbf{b}} \times \widetilde{\mathbf{c}})$ we obtain

$$z_B = -z_A \frac{\lambda_A(\widetilde{\mathbf{a}} \times \widetilde{\mathbf{c}}) \cdot (\widetilde{\mathbf{b}} \times \widetilde{\mathbf{c}})}{\lambda_B(\widetilde{\mathbf{b}} \times \widetilde{\mathbf{c}}) \cdot (\widetilde{\mathbf{b}} \times \widetilde{\mathbf{c}})}. \tag{5}$$

From (2) we have

$$\|K^{-1}(z_B\widetilde{\mathbf{b}} - z_A\widetilde{\mathbf{a}})\| = L \tag{6}$$

and by substituting z_B by (5) in this equation we get

$$z_A\|K^{-1}\mathbf{h}\| = L \tag{7}$$

where

$$\mathbf{h} = [h_1, h_2, h_3]^T = \frac{(z_B\widetilde{\mathbf{b}} - z_A\widetilde{\mathbf{a}})}{z_A} = \tag{8}$$

$$= \widetilde{\mathbf{a}} + \frac{\lambda_A(\widetilde{\mathbf{a}} \times \widetilde{\mathbf{c}}) \cdot (\widetilde{\mathbf{b}} \times \widetilde{\mathbf{c}})}{\lambda_B(\widetilde{\mathbf{b}} \times \widetilde{\mathbf{c}}) \cdot (\widetilde{\mathbf{b}} \times \widetilde{\mathbf{c}})}\widetilde{\mathbf{b}}. \tag{9}$$

Equation (7) is equivalent to

$$z_A^2\mathbf{h}^T\omega\mathbf{h} = L^2 \tag{10}$$

where

$$\omega = K^{-T}K^{-1} = \tag{11}$$

$$= \begin{bmatrix} \frac{1}{f^2\gamma^2} & -\frac{s}{f^2\gamma^2} & \frac{sy_0 - x_0}{f^2\gamma^2} \\ -\frac{s}{f^2\gamma^2} & \frac{s^2}{f^2\gamma^2} + \frac{1}{f^2} & -\frac{s(sy_0 - x_0)}{f^2\gamma^2} - \frac{y_0}{f^2} \\ \frac{sy_0 - x_0}{f^2\gamma^2} & -\frac{s(sy_0 - x_0)}{f^2\gamma^2} - \frac{y_0}{f^2} & \frac{(sy_0 - x_0)^2}{f^2\gamma^2} + \frac{y_0^2}{f^2} + 1 \end{bmatrix} \tag{12}$$

is the image of the absolute conic [3]. Let ω_{ij} be the element of ω at row i and column j. Then ω, which is symmetric, can be defined by

$$\mathbf{d} = [\omega_{11}, \omega_{12}, \omega_{22}, \omega_{13}, \omega_{23}, \omega_{33}]^T.$$

With $\mathbf{x} = z_A^2\mathbf{d}$ and

$$\mathbf{u} = [h_1^2, 2h_1h_2, h_2^2, 2h_1h_3, 2h_2h_3, h_3^2]^T,$$

equation (10) becomes

$$\mathbf{u}^T\mathbf{x} = L^2,$$

giving one constraint on z_A and the intrinsic parameters in K per image. In the most general case with six unknowns, we need at least six observations of the stick for calibration.

Given N images, the solution to (10) is found by solving a linear system of one equation per image, such that symmetry of ω is enforced:

$$\mathbf{U}\mathbf{x} = L^2\mathbf{1} \tag{13}$$

where $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_N]^T$ and $\mathbf{1} = [1, \ldots, 1]^T$. The least squares solution is then given by

$$\mathbf{x} = L^2(\mathbf{U}^T\mathbf{U})^{-1}\mathbf{U}^T\mathbf{1}.$$

K and z_A can then be found by Cholesky decomposition of $z_A^2\omega$ (which is given by $\mathbf{x}$).

3 Geometrical Interpretation

In order to identify the critical motions of the stick for which calibration will fail, we will now interpret equation (10) in geometrical terms. Refer to Figure 2. Let the line through A and B be l_{AB}. The intersection of l_{AB} and the plane at infinity π_∞ is given by $X_\infty = \widetilde{B} - \widetilde{A}$. Projecting this point onto the image we obtain the vanishing point

$$\mathbf{v} = [v_1, v_2, v_3]^T$$

of the line l_{AB}:

$$\begin{aligned} \mathbf{v} &= PX_\infty = K[I|0](\widetilde{B} - \widetilde{A}) = K(B - A) \\ &= z_B[x_B, y_B, 1]^T - z_A[x_A, y_A, 1]^T \\ &= z_B\widetilde{\mathbf{b}} - z_A\widetilde{\mathbf{a}}. \end{aligned}$$

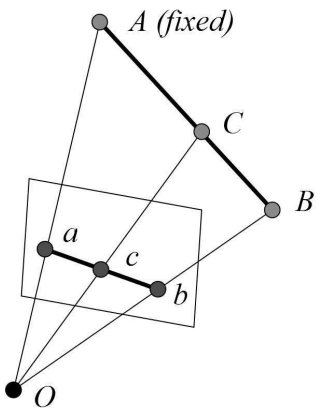


Figure 1. Illustration of 1D calibration objects

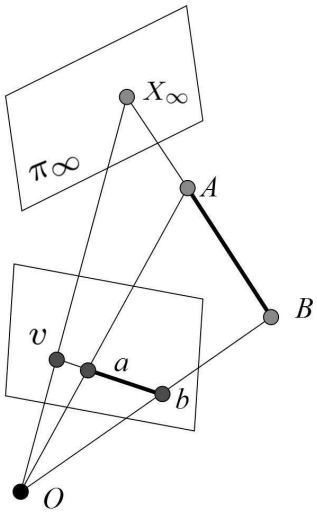


Figure 2. Geometrical interpretation of calibration from 1D objects

Using (8) we have

$$\mathbf{v} = z_A \mathbf{h}. \tag{14}$$

Alternatively, let $X = B - A$. With $\mathbf{v} = KX$ we get

$$1 = \frac{X^T X}{X^T X} = \frac{X^T K^T K^{-T} K^{-1} K X}{\|X\|^2} = \frac{\mathbf{v}^T \omega \mathbf{v}}{L^2} \Rightarrow$$

$$\mathbf{v}^T \omega \mathbf{v} = L^2, \tag{15}$$

so that (14) holds, since (15) $\Leftrightarrow$ (10). We can now interpret (15) as follows: the algebraic distance between the vanishing point of the stick and the image of the absolute conic equals L^2.

Notice that for calibration only, the actual length of the stick does not have to be known; using the constraint (10) will give us z_A in units of L (i.e. z_A will be the unit-less ratio of stick length and the actual metric depth of A), and always the correct calibration K. This is the typical scale-depth ambiguity in reconstruction; a change in scale can be compensated by a change in depth without changing the calibration matrix.

4 Degenerate cases

A motion of the stick is critical if and only if (15) has more than one solution. Given a number of observations of the stick, let $\mathbf{v}_i$ be the vanishing point in image i. The motion is now critical when the vanishing points of the stick $\mathbf{v}_i$ lie on a conic ω' so that

$$\mathbf{v}_i^T \omega' \mathbf{v}_i = 0 \quad \forall i,$$

since then, if ω is a solution to (15), $\omega + k\omega'$, $k \in \mathbf{R}$, is also a solution by

$$\mathbf{v}_i^T (\omega + k\omega') \mathbf{v}_i = \mathbf{v}_i^T \omega \mathbf{v}_i + k \mathbf{v}_i^T \omega' \mathbf{v}_i = L^2.$$

When solving for ω in (13), the actual solution $\omega + k\omega'$ is constrained to a symmetric matrix, therefore ω' must also be symmetric. If additional constraints are placed on ω, such that ω is of a more constrained form, then ω' must also be of the same, more constrained, form. This is done by incorporating knowledge on the intrinsic parameters as will be described in section 5.

Note that equation (10) would have no solutions (with $L \neq 0$) if $\mathbf{v}_i$ would lie on ω such that $\mathbf{v}_i^T \omega \mathbf{v}_i = 0$. Since ω is a virtual conic and the vanishing points are real (from $\mathbf{v} = P(\widetilde{B} - \widetilde{A})$), this however only happens if $\mathbf{v}_i = 0 \ \forall i$. This corresponds to the uninteresting case where A and B both lie on an optical ray of the camera in all images so that $\mathbf{a}$ and $\mathbf{b}$ coincide.

4.1 Critical motions

We now want to identify the critical motions of the stick that give rise to the degenerate cases where the vanishing points lie on a conic ω' in the image plane.

Assume $\mathbf{v}_i \omega' \mathbf{v}_i = 0$. Let D_i be any point on the stick in image i and $E_i = D_i - A$ the same point expressed in a coordinate system with origin translated to A. With $P = K[I|0]$ and $\mathbf{v}_i = K[I|0](\widetilde{D_i} - \widetilde{A}) = KE_i$ we get

$$\mathbf{v}_i^T \omega' \mathbf{v}_i = 0 \quad \Leftrightarrow \quad E_i^T K^T \omega' K E_i = 0 \quad \Leftrightarrow$$

$$E_i^T \omega'' E_i = 0 \quad \Leftrightarrow \quad \widetilde{E_i^T} \begin{bmatrix} \omega'' & 0 \\ 0 & 0 \end{bmatrix} \widetilde{E_i} = 0 \tag{16}$$

where ω'' is symmetric. Equation (16) tells us that all points

on the stick in all positions lie on a quadric of rank less than or equal to 3, in this case a cone, centered at A.

In other words: the motion is critical if and only if the vanishing points of the stick lie on a conic ω'. Since we deal with perspective projection, this is exactly the case if the stick's point at infinity traces out a conic on the plane at infinity during the motion (which can be a degenerate conic, e.g. consisting of 2 straight lines). This in turn means that the stick, when seen as an infinite line, traces out a cone, with the fixed point A as vertex and the above conic as "generator". Note that the cone does not need to be circular, i.e. the locus of an individual point on the stick does not need to be a planar circle for the degeneracy to occur. Furthermore, as mentioned above, the generating conic may be degenerate, e.g. consisting of 2 straight lines. As for the stick's motion, this means that it is waved in 2 different planes.

Note that critical motions do not depend on the actual position of the stick's fixed point A; they only depend on the stick's orientation (and in special cases, see below, on its orientation with respect to the camera).

In [14] some partial results on critical motions are given; the case of a circular cone. This is of course degenerate, but there are many more critical motions, as we have seen.

4.2 Safe motions

In practice, all critical motions should of course be avoided. From the above said, we observe that this can be achieved by for example moving the stick in three or more non-parallel planes, which may be realized by some zig-zag motion. Many other examples can be found, e.g. moving the stick in a spiral.

5 Closed-Form Solutions

We will now look at the closed-form solutions for the cases where some of the intrinsic parameters of the camera are known and show what degeneracies there are in these cases. We also show that the number of images required for calibration using these closed-form solutions will be smaller than in the general case.

5.1 Unknown focal length

Assume that only the focal length of the camera is unknown. The image coordinate system can then be transformed such that $s = 0$, $\gamma = 1$ and $(x_0, y_0) = (0, 0)$. Then

$$\omega = \begin{bmatrix} \frac{1}{f^2} & 0 & 0 \\ 0 & \frac{1}{f^2} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

so that the calibration problem reduces to solving equation (13) where (in the minimal case of only two images)

$$\mathbf{U} = \begin{bmatrix} h_{11}^2 + h_{21}^2 & h_{31}^2 \\ h_{12}^2 + h_{22}^2 & h_{32}^2 \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} \frac{z_A^2}{f^2} \\ z_A^2 \end{bmatrix}$$

and h_{ji} is h_j in image i. We observe that here, only two images are needed for calibration since then $\mathbf{U}$ is invertible. Modifying the calibration algorithm in this way fixes known camera parameters to their correct value and reduces the set of critical motions to the case where the vanishing points all lie on a circle centered in the image.

This can also be verified by noting that (13) has a unique solution if and only if $det(\mathbf{U}) \neq 0$. Now, denoting v_j in image i by v_{ji},

$$det(\mathbf{U}) = h_{32}^2(h_{11}^2 + h_{21}^2) - h_{31}^2(h_{12}^2 + h_{22}^2) = 0 \quad \Leftrightarrow$$

$$v_{32}^2(v_{11}^2 + v_{21}^2) - v_{31}^2(v_{12}^2 + v_{22}^2) = 0, \qquad (17)$$

since $\mathbf{v} = z_A\mathbf{h}$ (by (14)) and $z_A \neq 0$ since all depths are positive. The condition for a critical motion (17) is fulfilled if $v_{3i} = 0 \ \ \forall i$, which corresponds to the case where the vanishing point of the stick is a point at infinity so that the stick is moving in a plane parallel to the image plane, or if

$$(\frac{v_{11}}{v_{31}})^2 + (\frac{v_{21}}{v_{31}})^2 = (\frac{v_{12}}{v_{32}})^2 + (\frac{v_{22}}{v_{32}})^2 \quad \Leftrightarrow$$

$$v_{x1}^2 + v_{y1}^2 = v_{x2}^2 + v_{y2}^2$$

where v_{xi} and v_{yi} are the x- and y- coordinates of the vanishing point in image i (since $\mathbf{v}$ is expressed in homogeneous coordinates), meaning that the vanishing points lie on a centered circle. Now equation (16) gives that the stick lies on a quadric of the form

$$\begin{bmatrix} a & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & b & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

centered at A, where $a, b \in \mathbf{R}$, which is a circular cone whose axis of symmetry is parallel to the z axis (see Figure 3). Waving the stick in a plane parallel to the image plane is then also a degenerate motion, since it is a special case of a circular cone (it's like a cone that is squashed to a plane). In this case, the vanishing points of the stick are points at infinity of the image plane. The line at infinity of the image plane is a (degenerate) conic, of the required form (centered circle).

5.2 Unknown focal length and aspect ratio

In this case

$$\omega = \begin{bmatrix} \frac{1}{f^2\gamma^2} & 0 & 0 \\ 0 & \frac{1}{f^2} & 0 \\ 0 & 0 & 1 \end{bmatrix} ..$$

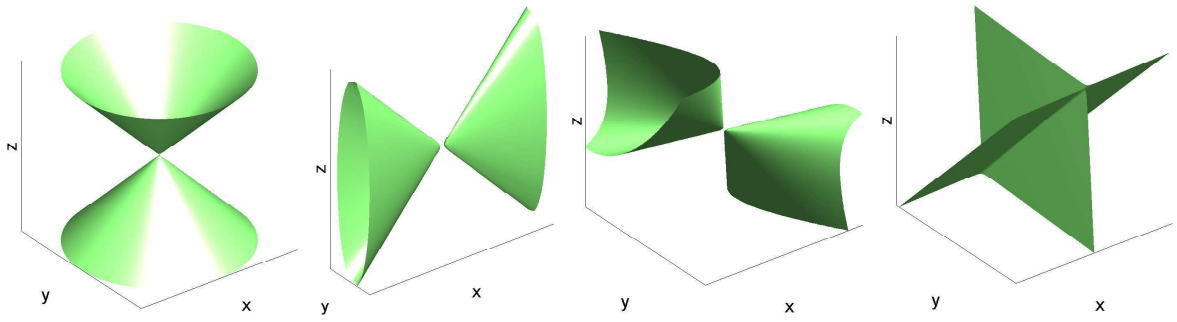


Figure 3. Examples of critical quadric surfaces. If only the focal length is unknown, the critical surface is a circular cone with axis of symmetry is parallel to the z axis (far left). With also the aspect ratio unknown the surface is an elliptical cone with main axis parallel to any two coordinate axes, and axis of symmetry parallel to the third one (center left). Examples of general quadrics representing critical surfaces in the general case (right). The camera has the optical axis coinciding with the z-axis and the image plane coordinate axes coinciding with the x- and y-axis

The calibration problem reduces to solving equation (13) where (in the minimal case of three images)

$$\mathbf{U} = \begin{bmatrix} h_{11}^2 & h_{21}^2 & h_{31}^2 \\ h_{12}^2 & h_{22}^2 & h_{32}^2 \\ h_{13}^2 & h_{23}^2 & h_{33}^2 \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} \frac{z_A^2}{f^2\gamma^2} \\ \frac{z_A^2}{f^2} \\ z_A^2 \end{bmatrix}$$

and h_{ji} is h_j in image i, which has a unique solution if and only if $\det(\mathbf{U}) \neq 0$. Now $\det(\mathbf{U}) = 0$ if and only if

$$v_{11}^2 v_{22}^2 v_{33}^2 + v_{21}^2 v_{32}^2 v_{13}^2 + v_{31}^2 v_{22}^2 v_{13}^2 - v_{31}^2 v_{22}^2 v_{13}^2 - v_{21}^2 v_{12}^2 v_{33}^2 - v_{11}^2 v_{23}^2 v_{32}^2 = 0 \qquad (18)$$

which is the condition for a critical motion. It is fulfilled either if $v_{j_0 i} = 0 \ \forall i$ and for some fixed j_0, corresponding to a motion of the stick in any of the two image coordinate axis planes ($v_1 = 0$ or $v_2 = 0$) or in a plane parallel to the image plane (v_3=0), or (by rewriting (18) by dividing with $v_{31}^2 v_{32}^2 v_{33}^2$, renaming $\frac{v_{1i}}{v_{3i}}$ to v_{xi} and $\frac{v_{2i}}{v_{3i}}$ to v_{yi}, which then are the image coordinates of the vanishing point) if

$$v_{x1}^2 v_{y2}^2 + v_{y1}^2 v_{x3}^2 + v_{x2}^2 v_{y3}^2 - v_{y2}^2 v_{x3}^2 - v_{y1}^2 v_{x2}^2 - v_{x1}^2 v_{y3}^2 = 0.$$

This means that the vanishing points are on a ellipse centered in the image, with axes coinciding with the image x and y axes. Equation (16) gives that the stick then moves on the surface of an elliptical cone with main axis parallel to any two coordinate axis, and axis of symmetry parallel to the third one, see Figure 3.

5.3 Unknown focal length and principal point

Another frequently occurring condition in camera calibration is that of $s = 0$ and $\gamma = 1$. In this case we find the simplified closed-form solution by observing that

$$\omega = \begin{bmatrix} \frac{1}{f^2} & 0 & -\frac{x_0}{f^2} \\ 0 & \frac{1}{f^2} & -\frac{y_0}{f^2} \\ -\frac{x_0}{f^2} & -\frac{y_0}{f^2} & \frac{x_0^2}{f^2} + \frac{y_0^2}{f^2} + 1 \end{bmatrix} ..$$

This reduces the problem to solving equation (13) where (in the minimal case of four images)

$$\mathbf{U} = \begin{bmatrix} h_{11}^2 + h_{21}^2 & 2h_{11}h_{31} & 2h_{21}h_{31} & h_{31}^2 \\ h_{12}^2 + h_{22}^2 & 2h_{12}h_{32} & 2h_{22}h_{32} & h_{32}^2 \\ h_{13}^2 + h_{23}^2 & 2h_{13}h_{33} & 2h_{23}h_{33} & h_{33}^2 \\ h_{14}^2 + h_{24}^2 & 2h_{14}h_{34} & 2h_{24}h_{34} & h_{34}^2 \end{bmatrix},$$

$$\mathbf{x} = z_A^2 \left[\tfrac{1}{f^2}, \quad -\tfrac{x_0}{f^2}, \quad -\tfrac{y_0}{f^2}, \quad \tfrac{x_0^2}{f^2} + \tfrac{y_0^2}{f^2} + 1 \right]^T$$

and h_{ji} is h_j in image i. The critical motions are according to equation (16) reduced to quadrics of the form

$$\begin{bmatrix} a & 0 & c & 0 \\ 0 & a & d & 0 \\ c & d & b & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

centered at A, where $a, b, c, d \in \mathbf{R}$. Other cases where a subset of the intrinsic parameters is known can be treated similarly.

6 Experiments

6.1 Simulation

In order to evaluate the calibration accuracy for motions close to being critical, an experiment on simulated data was

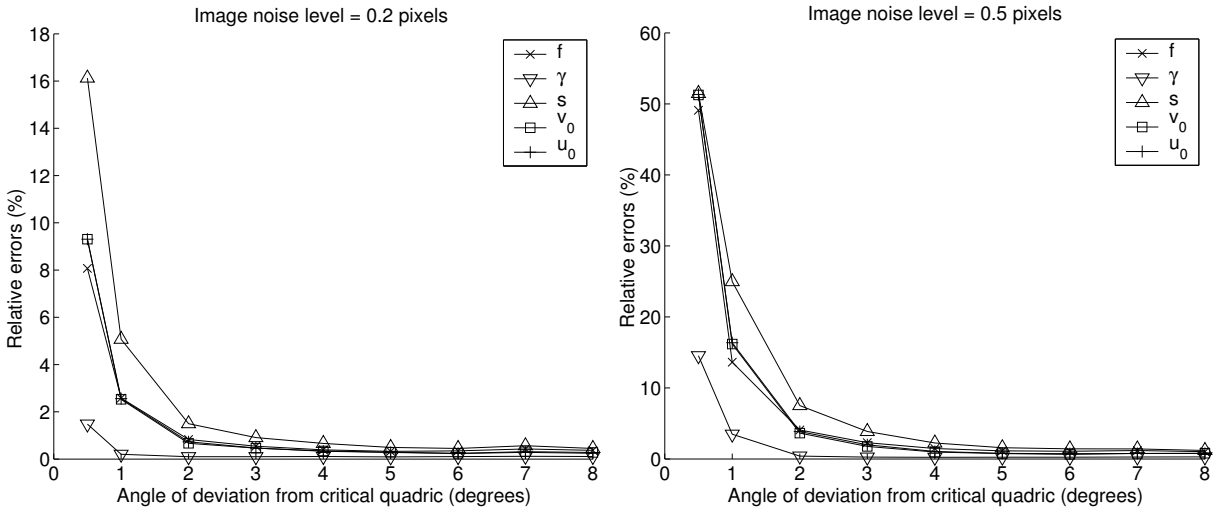


Figure 4. Calibration errors with respect to angle of deviation of the 1D objects from a critical quadric

performed. The simulated camera had $f = 1000$, $\gamma = 1$, $s = 0$ and $(x_0, y_0) = (320, 240)$. A stick of length $L = 70$ with $\lambda_A = \lambda_B = 0.5$ and fixed point $A = [0, 35, 150]^T$ was placed in 100 equally spaced positions on a critical cone. Gaussian noise with mean 0 and varying standard deviation was added to the angle between the stick and the axis of symmetry of the critical cone as illustrated in Figure 5. Gaussian noise with mean 0 and varying standard deviation was added to the obtained image points.

The calibration algorithm for the general case where all the intrinsic parameters are assumed to be unknown was used. We measure the relative accuracy of the focal length $|\Delta f/f|$ and the dimensionless quantities $|\Delta\gamma|$, $|\Delta s|$, $|\Delta u_0/f|$ and $|\Delta v_0/f|$ since errors in these contribute about equally to the overall geometric accuracy in scene reconstruction [11]. Results are given in Figure 4 for two different levels of image noise.

We note that the calibration results are very inaccurate for small angles of deviation from the critical surface as expected. The improvement in accuracy is very dramatic when increasing from close to 0° deviation from the cone, to a few degrees. After around 5° there is no big improvement and the results are quite good from this point on.

The fact that we get more accurate calibration results than in [14] is probably due to the stick being far from parallel to the optical axis of the camera. Since the endpoints of the image of the stick then are far apart, the results are less affected by noise.

Errors (%)	Sequence 1	Sequence 2
f	1.3566	20.3945
γ	1.5918	23.7308
s	0.7971	1.1993
u_0	4.6013	5.2164
v_0	0.6743	3.7431

Table 1. Experimental results for calibration from real data. In sequence 1, the stick is moving randomly. In sequence 2, the motion of the stick is such that it is close to a critical quadric surface

6.2 Real Data Experiment

To evaluate the sensitivity of the calibration algorithm in a real world scenario, a digital camera was calibrated using two separate image sequences containing images of a stick moving in two different patterns. The image resolution was 640×480 pixels. In the first sequence the stick was moved randomly. In the second sequence the stick was moved close to a critical surface, as illustrated in Figure 6. The camera was in both cases calibrated using the closed form solution for calibration from one dimensional objects given no knowledge of the intrinsic parameters, as described above. To be able to compare the results, the camera was also calibrated using the standard algorithm for calibration from planar patterns [13], including nonlinear minimization of the cameras intrinsic parameters from reprojection errors, resulting in a very precise calibration. The results are given

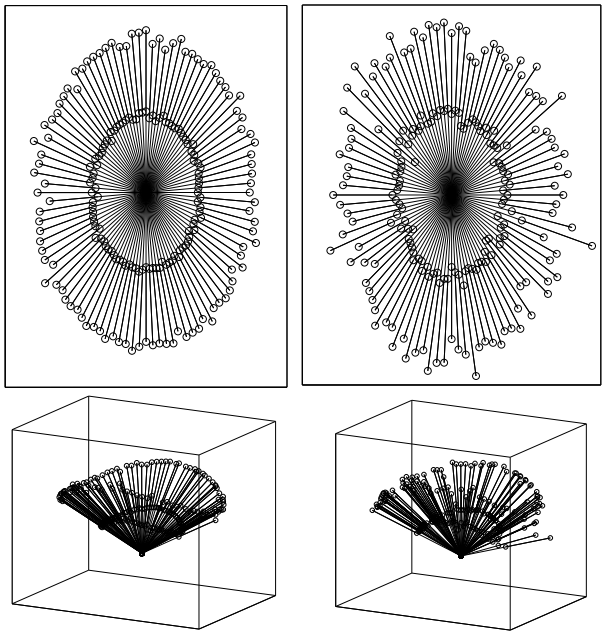


Figure 5. Simulation of sticks on a degenerate surface with added angular noise with a standard deviation of 2° (left) and 5° (right)

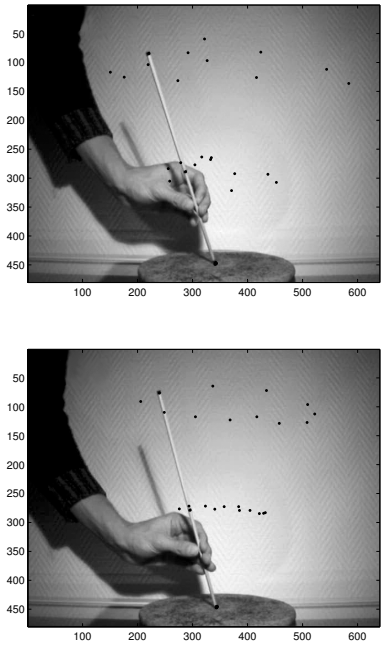


Figure 6. Two images from two image sequences, each consisting of 12 images. On each of the two images, tracked points from the entire sequence are superimposed. In sequence 1, the stick is moving in a random fashion (top). In sequence 2, the motion of the stick is such that it is close to a critical quadric surface in each image (bottom)

in Table 1, where the errors in the intrinsic parameters from each of the two calibration results are given with respect to the calibration result from the planar patterns. The errors from the sequence with the degenerate stick movement is generally much larger than for the random movement sequence, suggesting that close-to-critical motions of the stick has to be avoided in practice.

7. Summary and Conclusions

Based on a geometrical interpretation of the constraint used in camera calibration with one-dimensional objects, we have identified the critical motions where the calibration algorithm will fail. We have shown that constraints on the intrinsic parameters of the camera lead to simplified closed-form-solutions and a reduced set of critical motions, and also proposed some safe non-critical motions that will guarantee the success of the calibration algorithm in practice. A simulation and a real data experiment was performed to evaluate the calibration accuracy for motions close to being critical, showing the sensitivity of the algorithm to these motions.

References

[1] M. Armstrong, A. Zisserman, and P. Beardsley. Euclidean structure from uncalibrated images. In E. Hancock, editor, *Proceedings of the 5th British Machine Vision Conference, York, England*, volume 2, pages 509–518, September 1994.

[2] T. Buchanan. The twisted cubic and camera calibration. *Computer Vision, Graphics and Image Processing*, 42:130–132, 1988.

[3] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[4] R. I. Hartley. Self-calibration from multiple views with a rotating camera. In *Proc. European Conf. on Computer Vision*, pages 471–478, Stockholm, Sweden, 1994.

[5] A. Heyden and K. Åström. Flexible calibration: Minimal cases for auto-calibration. In *Proc. Int. Conf. on Computer Vision*, pages 350–355, Kerkyra, Greece, 1999.

[6] A. Heyden and K. Åström. Euclidean Reconstruction from Image Sequences with Varying and Unknown Focal Length and Principal Point. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 438–443, San Juan, Puerto Rico, 1997.

[7] S. J. Maybank and O. D. Faugeras. A theory of self calibration of a moving camera. *Int. Journal of Computer Vision*, 8(2):123–151, 1992.

[8] M. Pollefeys, L. Van Gool, and M. Proesmans. Euclidean 3d reconstruction from image sequences with variable focal lengths. In *Proc. European Conf. on Computer Vision*, pages 31–42, Cambridge, UK, 1996.

[9] P. Sturm. Critical motion sequences for monocular self-calibration and uncalibrated Euclidean reconstruction. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 1100–1105, San Juan, Puerto Rico, 1997.

[10] P. Sturm and S. Maybank. On plane-based camera calibration: A general algorithm, singularities, applications. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Fort Collins, Colorado, USA*, pages 432–437, June 1999.

[11] B. Triggs. Autocalibration from planar scenes. In *Proc. European Conf. on Computer Vision*, volume I, pages 89–105, Freiburg, Germany, 1998.

[12] R.Y. Tsai. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. IEEE *Journal of Robotics and Automation*, 3(4):323–344, August 1987.

[13] Z. Zhang. A flexible new technique for camera calibration. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.

[14] Z. Zhang. Camera calibration with one-dimensional objects. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 26(7):892–899, 2004.

Combining Off- and On-line Calibration of a Digital Camera

Magdalena Urbanek, Radu Horaud and Peter Sturm
INRIA Rhône-Alpes
655, avenue de l'Europe, 38334 Saint Ismier Cedex, France
{Magdalena.Urbanek, Radu.Horaud, Peter.Sturm}@inrialpes.fr

Abstract

We introduce a novel outlook on the self-calibration task, by considering images taken by a camera in motion, allowing for zooming and focusing. Apart from the complex relationship between the lens control settings and the intrinsic camera parameters, a prior off-line calibration allows to neglect the setting of focus, and to fix the principal point and aspect ratio throughout distinct views. Thus, the calibration matrix is dependent only on the zoom position. Given a fully calibrated reference view, one has only one parameter to estimate for any other view of the same scene, in order to calibrate it and to be able to perform metric reconstructions. We provide a close-form solution, and validate the reliability of the algorithm with experiments on real images. An important advantage of our method is a reduced - to one - number of critical camera configurations, associated with it. Moreover, we propose a method for computing the epipolar geometry of two views, taken from different positions and with different (spatial) resolutions; the idea is to take an appropriate third view, that is "easy" to match with the other two.

1. Introduction

The problem of recovering the Euclidean structure of a scene is strongly associated with the estimation of the camera internal parameters, i.e. calibration. When no calibration knowledge provided, one can reconstruct only a projective model of the scene [6, 10].

1.1. Previous work

The most basic solution to compute the internal parameters employs a calibration grid or planes, and performs an off-line calibration. However the restriction of keeping an identical camera state (including zooming and focusing) while shooting subsequent images can hardly be fulfilled in practice.

Another idea is to self-calibrate an entire sequence. Existing approaches follow several directions. One is to assume invariance of unknown intrinsic parameters throughout distinct views [12, 15, 1, 8], thus not to allow for zooming/focusing, which is quite a strong constraint. Given a stereo pair of an arbitrary scene, one cannot vary but *the magnification parameter* (we use that term, to avoid confusion of associating different meanings to *the focal length*, in vision and optics), while having the other ones known [7]. Other methods [3, 13] allow the retrieval of varying magnification parameter and fixed principal point. Furthermore, if provided with at least 9 views, it is possible to fix only one camera internal parameter and let the other ones vary [13, 11].

In reality, such a general calibration problem cannot be solved reliably. On the other hand, one can quite easily provide some prior information, which simplifies the task. Our approach belongs to such a group of techniques.

1.2. Motivation

All considered cases of self-calibration, which allow magnification parameter variation throughout distinct views, suffer from a significant number of critical camera configurations [14]. It is therefore much "safer" not to change the camera settings.

Let us combine one fully calibrated image (*the reference image*) with an uncalibrated one, taken from a different viewpoint. Then, one has only one magnification parameter to estimate. What about the other intrinsic parameters? The complex relationship between calibration and camera lens control settings [16] does not allow straight-forward simplifications.

To summarize, we are interested in the following issues:

- Are there any conditions that enable the use of a priori knowledge of the intrinsic parameters?
- Can one allow for zooming/focusing, while still maintain a small family of critical situations?
- What can be done with stereo pairs, if one camera/view is fully calibrated?

1.3. Contribution

We combine off-line and on-line methods in order to calibrate a digital camera with a zoom lens and auto-focus.

We introduce a novel outlook on the self-calibration problem, by reducing to one the number of intrinsic parameters to be estimated. We provide a close-form solution for the method. Also, one has to account for only a single family of critical camera configurations [14].

By studying the behaviour of the camera intrinsic parameters as a function of variable zoom and focus, we derive approximate values for the aspect ratio and the principal point. We identify a small influence of focus upon calibration, which becomes negligible for settings larger than 2.5m. We conclude, that once a camera is calibrated for a known zoom setting, one can re-use those values any time that zoom is set. Therefore, we recommend employing minimally or maximally zoomed-in images as the reference ones, since those zoom settings can be reliably reproduced.

Furthermore, we simplify the computation of the epipolar geometry for stereo images of different resolutions, omitting a direct matching between them. The problem of matching two images of different zoom and viewpoint is therefore decomposed into two simpler matching problems: a wide baseline matching with the same zoom [2], and matching images with different zoom, shot from the same viewpoint [5].

The proposed method of "combined calibration" estimates the intrinsic parameters with even 2%-accuracy, from real images, leading to a reliable Euclidean reconstruction.

2. Camera modeling

2.1. The model

We assume the perspective camera model with the projection matrix of the form:

$$\mathsf{P} = \mathsf{K}\begin{pmatrix}\mathsf{R} & \mathbf{t}\end{pmatrix} \tag{1}$$

where R and $\mathbf{t}$ represent the orientation and the position of the camera with respect to the world coordinate system, and K is the calibration matrix:

$$\mathsf{K} = \begin{pmatrix} k\alpha & 0 & u_0 \\ 0 & \alpha & v_0 \\ 0 & 0 & 1 \end{pmatrix}$$

with the principal point (u_0, v_0), the magnification parameter α and the aspect ratio k. We assume a zero-skew.

A scene point $\mathbf{M}$ is projected onto the image onto a point $\mathbf{m}$ via $\mathbf{m} = \mathsf{P}\mathbf{M}$.

2.2. Off-the-shelf digital camera

Most often one is provided with digital cameras, which allow mechanical setting of both zoom and focus. One can specify the area of interest (and thus, its depth on the image) and focus on chosen features within the area.

We have worked with the Olympus Camedia C-2500L digital camera. It provides both auto-focus and manual-focus with discretized values from 0.3m until 15m and ∞ to be set. The zoom, on the contrary, has a continuous range and a manual drive, which makes the reproducibility of different settings difficult (with notable exceptions for the minimal and the maximal zooms).

Each $(zoom, focus)$ setting corresponds to a physical configuration of lenses, inside the camera. Since their functional dependencies are complex, we cannot specify the exact camera state, which makes the estimation of camera internal parameters difficult. When using auto-focus, the only camera settings that we are able to reproduce (and to expect the same calibration results, for an arbitrary image, taken with the same settings) are: $(zoom, focus) = (zoom_{min}, \infty)$ and $(zoom, focus) = (zoom_{max}, \infty)$.

The question is how do the entries of the calibration matrix K change with variations of zoom and focus. Experiments described in the following section suggest conditions, under which the internal camera calibration can be assumed invariant, for different $(zoom, focus)$ settings.

3. Off-line stability study of calibration

We study the stability of the camera internal parameters, under change in the camera mechanical settings, zoom and focus. We point out the parameters that do not vary much, and can be assumed invariant. We find a small influence of focus on calibration, if the camera is far enough from the scene. Finally, we provide calibration knowledge for particular zoom settings, which is to be used a priori, in self-calibration.

3.1. A way to calibrate

We extract the calibration matrix K from the projection matrix P, estimated from correspondences between non-coplanar 3D points and their 2D images.

The form of $\mathsf{P} = (\bar{\mathsf{P}} \quad \mathbf{p})$ and (1) imply: $\bar{\mathsf{P}} = \mathsf{KR}$. Since $\bar{\mathsf{P}}\bar{\mathsf{P}}^\mathsf{T} = \mathsf{KRR}^\mathsf{T}\mathsf{K}^\mathsf{T} = \mathsf{KK}^\mathsf{T}$, we can simply obtain K from the Cholesky decomposition of $\bar{\mathsf{P}}\bar{\mathsf{P}}^\mathsf{T}$.

In order to estimate P, we run a non-linear algorithm, which minimizes the reprojection error

$$C = \sum_{i=1}^{n} (u_i - u_{mi})^2 + (v_i - v_{mi})^2 \tag{2}$$

of n image points (u_i, v_i) and reprojections (u_{mi}, v_{mi}) of the corresponding 3D points $\mathbf{M}_i$.

3.2. Optical distortion

Since imperfect camera lenses give rise to non-perspective image distortion, it is often necessary to optimize (2) using additional distortion parameters. In some cases, this extended projection model causes over-parameterization, resulting in instabilities in the estimation of all intrinsic parameters.

Based on the observation, that the bigger the zoom used, the less distortion is present in the image, we can point out experimentally a "critical" zoom, for which the *estimated* distortion coefficient does not decrease with the increase of zoom. Therefore, we omit the distortion parameters in the optimization, if a zoom is bigger than the "critical" one.

We only estimate the first term D_r of the radial distortion, which proved sufficient to provide reliable results. Overall, the employed calibration method is described in [4].

3.3. Experiments

We stepped the lens through the full range of focus, while the zoom was examined in two positions: the minimal and the maximal ones. At each step, we performed a full camera calibration (images of a calibration grid were considered). To ensure the stability of calibration, we considered only images with a sufficiently large number of control points clearly visible.

We used manual focusing. For each $(zoom, focus)$ setting, we took several images with slightly different orientations of the calibration grid. The distance camera-grid was kept identical to the value of the set focus.

We considered focus values between 1m and 5m. The images were of size 640×512 pixels. The obtained estimates of the internal camera parameters are listed separately: for the minimal zoom (Table 1), for the maximal zoom (Table 2).

Focus[m]	k[1]	α[pix]	u_0[pix]	v_0[pix]	D_r[1]
1	0.9993	700	321	268	-0.2393
	0.9991	698	321	267	-0.2423
	0.9999	700	317	267	-0.2363
1.2	0.9992	695	314	277	-0.2598
	0.9996	702	320	269	-0.2405
	0.9997	728	294	238	-0.1468
1.5	0.9998	731	316	232	-0.1574
	1.0007	710	325	269	-0.2469
	0.9998	723	318	234	-0.1601
2	1.0007	736	295	269	-0.1523
	1.0002	699	319	274	-0.2970
2.5	1.0001	722	318	268	-0.2207

Table 1. Calibration results: the $minimal$ zoom and varying focus.

Focus[m]	k[1]	α[pix]	u_0[pix]	v_0[pix]	D_r[1]
1	0.9996	921	316	268	-0.0976
	0.9994	920	318	268	-0.0945
	0.9992	918	319	269	-0.1037
1.2	1.0010	1133	320	266	0.0445
	1.0009	1122	317	274	0.0075
	1.0008	1128	318	271	0.0218
1.5	1.0015	1384	310	297	0
	1.0005	1386	291	296	0.0294
	1.0013	1391	320	292	0
2	1.0020	1749	312	313	0
	1.0020	1740	311	310	0
	1.0013	1745	289	303	0
2.5	1.0030	1969	301	324	0.1470
	1.0008	1944	255	316	0.0306
3.5	1.0024	1959	314	356	0.0295
	1.0012	1965	290	336	0
5	1.0016	1999	301	344	0.0182

Table 2. Calibration results: the $maximal$ zoom and varying focus.

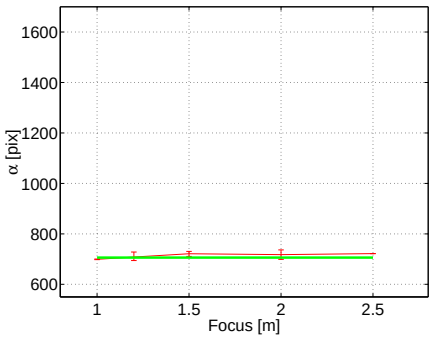
3.4. Dependencies

What information can be extracted from Tables 1 and 2?

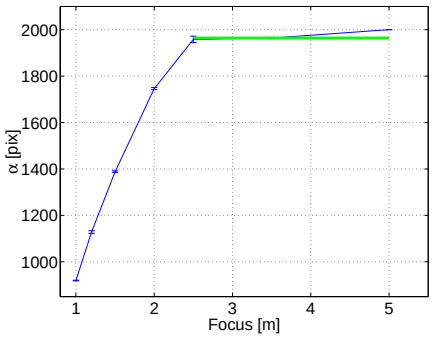
Aspect ratio (k). It is close to unity. The equality $k = 1$ is valid for any $(zoom, focus)$ setting, with a relative error smaller than 0.2%.

Magnification parameter (α). For the minimal zoom, α stays constant relative to focusing. For the maximal zoom, the same is observed as soon as the distance camera-object is bigger than 2.5m (see Figure 1). Hence, for a chosen zoom, it is possible to represent the relevant α with a single value (e.g. the median of the estimates): $\alpha_{min} = 706$ (with 2%-relative error) for $(zoom, focus) = (zoom_{min}, focus \geq 0.3\text{m})$, and $\alpha_{max} = 1965$ (with 1%-relative error) for $(zoom, focus) = (zoom_{max}, focus \geq 2.5\text{m})$.

Principal point (u_0, v_0). Overall, it concentrates near the image centre (see Figure 2). Since in general, the exact position of the principal point does not have a big impact upon the quality of reconstruction, it is possible to employ approximate statistical values, obtained from the Student's reliability test: $u_0 = 311 \pm 21$, $v_0 = 280 \pm 42$, with a factor of risk 0.1 . Further on, we will use the approximation: $(u_0, v_0) = (311, 280)$.



(a) Minimal zoom: median $\alpha_{min} = 706$.



(b) Maximal zoom: median $\alpha_{max} = 1965$.

Figure 1. Possibilities to approximate the magnification parameter α with its median.

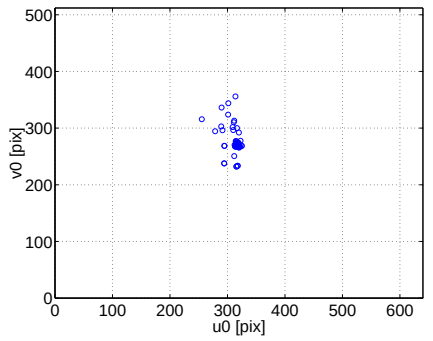


Figure 2. Principal point concentrates near the image centre.

Auto-focusing. For a fixed zoom, the setting of focus does not influence calibration significantly. We can use auto-focusing, and still be capable to employ calibration results for the examined zooms. We only have to keep in mind the requirement concerning the maximal zoom: the distance camera-scene has to be larger than 2.5m.

3.5. Final results to be used in self-calibration

A view taken with the minimal/maximal zooming. We are provided with calibration matrices of reference: K_{min} for the minimal zoom case (for any focus value), and K_{max} for the maximal zoom case (for focus ≥ 2.5m).

A view taken with an arbitrary (unknown) zooming. One is provided with the values of k and (u_0, v_0). Hence, α remains the only calibration parameter to determine.

A summary is given in Table 3.

Zoom	Focus[m]	k[1]	α[pix]	u_0[pix]	v_0[pix]
min	≥ 0.3	1	706	311	280
max	≥ 2.5	1	1965	311	280
?	**?**	1	**?**	311	280

Table 3. Results of off-line calibration (the Olympus Camedia C-2500L digital camera).

4. Self-calibration

We consider a stereo pair: a calibrated reference image and an image taken with an unknown zoom. (In practice, we obtain the calibration for the reference image simply by taking it using the minimal or the maximal zoom, and adopting the according intrinsic parameters, obtained by the off-line calibration.) We are thus provided with calibration matrices: K_{ref}, fully known, for the reference image, and K, defined up to unknown α, for the other image. Due to Kruppa's equations [9], we derive a close form solution for α. Also, we reveal stereo configurations, for which our self-calibration algorithm fails.

4.1. Kruppa's equations

Finding the matrix K associated with a camera is equivalent to finding the image ω of the absolute conic, taken by that camera. Since $\omega^{-1} \sim \mathsf{K}\mathsf{K}^{\mathsf{T}}$, let us denote $\mathsf{C} = \mathsf{K}\mathsf{K}^{\mathsf{T}}$ for the camera to be calibrated, and $\mathsf{C}_{ref} = \mathsf{K}_{ref}{\mathsf{K}_{ref}}^{\mathsf{T}}$ for the camera, that took the reference image.

The link between images of the absolute conic and the epipolar geometry can be expressed as follows ([17]; F is the fundamental matrix of the stereo; $\mathbf{e}$ is the epipole on the image, taken by the uncalibrated camera):

$$\mathsf{F}\mathsf{C}_{ref}\mathsf{F}^{\mathsf{T}} \sim [\mathbf{e}]_{\times}\mathsf{C}{[\mathbf{e}]_{\times}}^{\mathsf{T}} \tag{3}$$

Having separated the known entries of matrix K (k, u_0, v_0) from the unknown one (α):

$$\mathsf{K} = \begin{pmatrix} k & 0 & u_0 \\ 0 & 1 & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \alpha & 0 & 0 \\ 0 & \alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} = \mathsf{K}_0 \mathsf{K}_\alpha$$

and multiplying (3) from the left by ${\mathsf{K}_0}^\mathsf{T}$, and from the right by K_0, we obtain:

$${\mathsf{K}_0}^\mathsf{T} \mathsf{F} \mathsf{K}_{ref} {\mathsf{K}_{ref}}^\mathsf{T} \mathsf{F}^\mathsf{T} \mathsf{K}_0 \sim {\mathsf{K}_0}^\mathsf{T} [\mathbf{e}]_\times \mathsf{K}_0 \underbrace{\mathsf{K}_\alpha \mathsf{K}_\alpha^\mathsf{T}}_{\mathsf{C}_\alpha} {\mathsf{K}_0}^\mathsf{T} [\mathbf{e}]_\times^\mathsf{T} \mathsf{K}_0 \tag{4}$$

with C_α of the following form:

$$\mathsf{C}_\alpha = \begin{pmatrix} \alpha^2 & 0 & 0 \\ 0 & \alpha^2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{5}$$

Let us denote:

$$\underline{\mathsf{E}} = {\mathsf{K}_0}^\mathsf{T} \mathsf{F} \mathsf{K}_{ref} \tag{6}$$

which moves F to a "semi-calibrated" space. From a property (valid for any matrix A and vector $\mathbf{v}$)

$$\mathsf{A}^{-\mathsf{T}} [\mathbf{v}]_\times \sim [\mathsf{A}\mathbf{v}]_\times \mathsf{A} \tag{7}$$

we have:

$${\mathsf{K}_0}^\mathsf{T} [\mathbf{e}]_\times \mathsf{K}_0 \sim [{\mathsf{K}_0}^{-1} \mathbf{e}]_\times \underbrace{{\mathsf{K}_0}^{-1} \mathsf{K}_0}_{\mathtt{I}} \tag{8}$$

Thus, (6) and (8) enable us to write (4) as:

$$\underline{\mathsf{E}} \underline{\mathsf{E}}^\mathsf{T} \sim [{\mathsf{K}_0}^{-1} \mathbf{e}]_\times \mathsf{C}_\alpha {[{\mathsf{K}_0}^{-1} \mathbf{e}]_\times}^\mathsf{T} \tag{9}$$

Let us use the Singular Value Decomposition of $\underline{\mathsf{E}}$:

$$\underline{\mathsf{E}} = \mathsf{U}\ \mathrm{diag}(r, s, 0)\ \mathsf{V}^\mathsf{T} \tag{10}$$

Introducing (10) into (9), and moving U and U^T to the opposite side of the formula, result in:

$$\mathrm{diag}(r, s, 0) \underbrace{\mathsf{V}^\mathsf{T} \mathsf{V}}_{\mathtt{I}} \mathrm{diag}(r, s, 0) \sim \mathsf{U}^\mathsf{T} [{\mathsf{K}_0}^{-1} \mathbf{e}]_\times \mathsf{C}_\alpha [{\mathsf{K}_0}^{-1} \mathbf{e}]_\times \mathsf{U} \tag{11}$$

(remind a property: ${[\mathbf{v}]_\times}^\mathsf{T} = -[\mathbf{v}]_\times$, for any vector $\mathbf{v}$). Using (7), we can write (11) in the form:

$$\mathrm{diag}(r^2, s^2, 0) \sim [\mathsf{U}^\mathsf{T} {\mathsf{K}_0}^{-1} \mathbf{e}]_\times \mathsf{U}^\mathsf{T} \mathsf{C}_\alpha \mathsf{U} [\mathsf{U}^\mathsf{T} {\mathsf{K}_0}^{-1} \mathbf{e}]_\times \tag{12}$$

Let us notice, that

$$[\mathsf{U}^\mathsf{T} {\mathsf{K}_0}^{-1} \mathbf{e}]_\times = \left[\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \right]_\times = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

If we denote with $(\mathbf{u}_1\ \mathbf{u}_2\ \mathbf{u}_3)$ columns of matrix U, (12) writes as follows:

$$\begin{pmatrix} r^2 & 0 \\ 0 & s^2 \end{pmatrix} \sim \begin{pmatrix} {\mathbf{u}_2}^\mathsf{T} \mathsf{C}_\alpha \mathbf{u}_2 & -{\mathbf{u}_1}^\mathsf{T} \mathsf{C}_\alpha \mathbf{u}_2 \\ -{\mathbf{u}_1}^\mathsf{T} \mathsf{C}_\alpha \mathbf{u}_2 & {\mathbf{u}_1}^\mathsf{T} \mathsf{C}_\alpha \mathbf{u}_1 \end{pmatrix} \tag{13}$$

Equalities between ratios of coefficients of the matrices in (13) form Kruppa's equations. However, only the following equality can contribute positively to the solution:

$$\frac{r^2}{s^2} = \frac{{\mathbf{u}_2}^\mathsf{T} \mathsf{C}_\alpha \mathbf{u}_2}{{\mathbf{u}_1}^\mathsf{T} \mathsf{C}_\alpha \mathbf{u}_1} \tag{14}$$

The other possible equation $({\mathbf{u}_1}^\mathsf{T} \mathsf{C}_\alpha \mathbf{u}_2 = 0)$ leads always to a solution $\mathsf{C}_\alpha = \mathtt{I}$, and thus $\alpha = 1$.

Remembering the form of C_α (5), one can retrieve from (14) the unknown α, by solving a quadratic equation (since the numerator and denominator of (14) are linear expressions in entries of matrix C_α):

$$\alpha = \sqrt{\frac{s^2 u_{32}^2 - r^2 u_{31}^2}{r^2 (u_{11}^2 + u_{21}^2) - s^2 (u_{12}^2 + u_{22}^2)}} \tag{15}$$

where u_{ij} are entries of matrix U, and r, s - the singular values, given in (10).

4.2. Outline of the algorithm

Step 0: Perform off-line calibration of the camera, obtaining $\alpha_{min}, \alpha_{max}, k, (u_0, v_0)$ - thus full calibration matrices: K_{min} and K_{max} for reference images, and a calibration matrix K (associated with any other image) defined up to α.

Then, for a stereo pair (a reference image and an image of an unknown zooming), given the matching:

Step 1: Compute the fundamental matrix F.

Step 2: Move F to a "semi-calibrated" space, obtaining a new matrix $\underline{\mathsf{E}}$ - see (6).

Step 3: Apply the SVD on $\underline{\mathsf{E}}$ - see (10).

Step 4: Use entries of the matrices obtained in Step 3 to compute the unknown internal parameter α of matrix K - see (15).

4.3. Critical motions

As it has been fully studied in [14], a solution for the unknown magnification parameter is not always uniquely defined. In our case, since we consider to know all intrinsic parameters of one camera, there exist only one family of camera configurations that is critical, which is significantly less than with more general cases of self-calibration.

Let us consider a stereo pair of cameras: $\mathbf{C}_{ref}$ (fully calibrated) and $\mathbf{C}$ (with an unknown α). The algorithm is

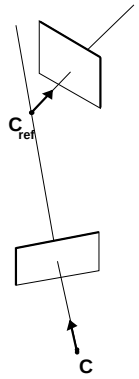


Figure 3. Critical configuration of cameras

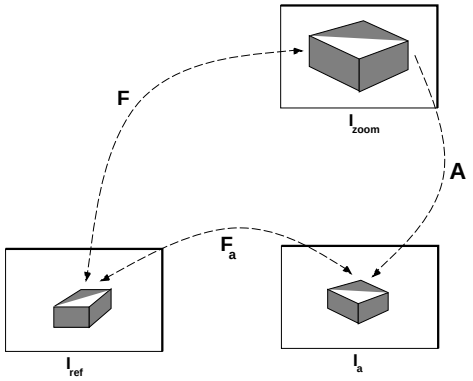


Figure 4. Connections between images: fundamental matrix F_a between I_{ref} and I_a, affine transformation A between I_a and I_{zoom}, fundamental matrix F between I_{ref} and I_{zoom}

singular if the centre of camera $\mathbf{C}_{ref}$ lies on the optical axis of camera $\mathbf{C}$ (Figure 3). This kind of configuration is connected with a camera movement (starting from the reference position), that consists of any rotation, followed by a translation along the optical axis of the camera. There is no constraint on the orientation of camera $\mathbf{C}_{ref}$.

The case has been analyzed along the lines of [14]. Here, we omit its explanation, due to the lack of space.

In practice, any camera configuration that is close to the critical one, can cause problems in self-calibration, giving rise to inaccurate results.

5. Matching

We are interested in running our self-calibration algorithm on pairs of images of different spatial resolutions (different magnifications). Being aware of problems concerning matching such views, we propose a way to avoid it, by introducing an additional view, that allows to match the two original ones.

5.1. Difficulties

Existing direct techniques for automatic matching of two images taken from different viewpoints and with different resolutions do not give satisfactory results. Since a big area on the zoomed-in image is to be correlated with a small area on the zoomed-out image, accuracy of computed epipolar lines is weak.

Even when dealing with images very similarly zoomed-in, very few algorithms cope with matching them, if the camera movement between the two views is not small. On the other hand, once one decreases the baseline between cameras (so that it would be appropriate for correlation techniques), the scene reconstruction becomes less reliable.

5.2. Our method

To avoid a manual specification of corresponding points, we combine two techniques:

- matching two images of *the same resolution*, taken from *different viewpoints*
- matching two images of *different resolutions*, taken from *the same viewpoint*

Hence, we assume being provided with an additional view I_a, of the same resolution as the reference image I_{ref}, but taken from the same camera position as the zoomed-in one I_{zoom} (see Figure 4).

Having performed an automatic matching [2] between I_{ref} and I_a, we compute the fundamental matrix F_a of that stereo pair. Thus for any image points $\mathbf{m}_{ref}, \mathbf{m}_a$ (related to I_{ref} and I_a respectively):

$$\mathbf{m}_a{}^{\mathsf{T}} \mathsf{F}_a \mathbf{m}_{ref} = 0 \tag{16}$$

On employing method [5], we match I_a with I_{zoom}, and estimate an affine transformation A between them (due to the lack of space, we omit a derivation of this property) - for any image points $\mathbf{m}_a, \mathbf{m}_{zoom}$ (related to I_a and I_{zoom} respectively):

$$\mathbf{m}_a = \mathsf{A}\mathbf{m}_{zoom} \tag{17}$$

Now, (16) and (17) let us find out the fundamental matrix F of the stereo pair of interest - I_{ref} and I_{zoom}:

$$(\mathsf{A}\mathbf{m}_{zoom})^{\mathsf{T}} \mathsf{F}_a \mathbf{m}_{ref} = 0$$

$$\mathbf{m}_{zoom}{}^{\mathsf{T}} \underbrace{\mathsf{A}^{\mathsf{T}} \mathsf{F}_a}_{\mathsf{F}} \mathbf{m}_{ref} = 0$$

"True α"	Self-calibration for α			
	Calibration grid		Arbitrary object	
	min ref.	max ref.	min ref.	max ref.
708	701±1%	763±8%	731±3%	691±2%
847	814±4%	887±5%	881±4%	829±2%
1018	960±6%	1044±3%	1040±2%	1012±1%
1250	1171±6%	1273±2%	1298±4%	1242±1%
1486	1387±7%	1508±1%	1592±7%	1442±3%
1729	1621±6%	1751±1%	1886±9%	1779±3%
1905	1772±7%	1933±1%	2072±9%	1862±2%

Table 4. Self-calibration results. The same α (in a row, related to a zoom setting) is estimated from stereo pairs of two different objects, with both kind of reference images each. "True α" comes from calibration.

Thus

$$\mathsf{F} = \mathsf{A}^{\mathsf{T}}\mathsf{F}_a \tag{18}$$

Equation (18) enables us to compute the epipolar geometry between images of different resolutions (I_{ref} and I_{zoom}), without being given matches between them. It is sufficient to specify correspondences between each of those images and a special additional one (I_a), and result in connections written as functions of F_a and A.

6. Experiments

Input data. We took images of an arbitrary object (a toy house), with the minimal and the maximal zoom settings, from a reference viewpoint. Then, from another camera position, we shot a number of images, of variable zooming. We also took images of a calibration grid, every time a view of the house was registered. Hence, we had two sets with corresponding images (of different features), taken with identical camera settings (Figure 5).

Separately for each photographed object, we combined our images in stereo pairs of a reference image and an image taken with an unknown zoom. Each image of an unknown zoom was put into 2 stereo pairs: with a minimal and with a maximal zoom reference image. Having employed results from off-line calibration (the constraint: distance camera-scene ≥ 2.5m had been fulfilled), we ran the self-calibration algorithm for each stereo pair, obtaining estimations for α, related to every considered zooming (see Table 4).

Discussion. The algorithm recovers the unknown magnification parameter with a high accuracy. However, there are some cases, where the relative error grows up to 9%. They show up for stereo pairs, which combine images of significantly different resolutions (e.g. a minimal zoom reference view with a strongly zoomed-in one; and vice-versa).

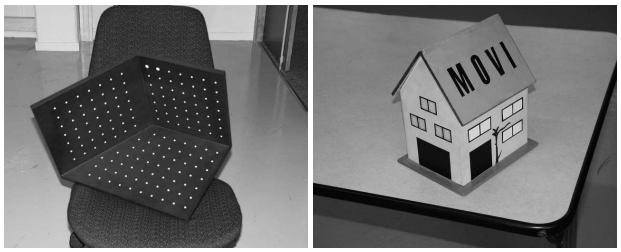


Figure 5. Images of different features, taken with identical camera settings.

The reason could be related to the fact, that the considered self-calibration step does not take into account any distortion model, and thus, its results are not always consistent with the off-line calibration (see Section 3.2). In particular: a distortion model, considered for the minimal zoom reference image, is "forwarded" by self-calibration to α, estimated for the other image of the stereo pair. If that image has been taken with a relatively big zoom setting, the no-distortion model has to be considered then, in order to avoid over-parameterization. For the opposite case: not taking distortion into account for the maximal zoom reference case, implies the no-distortion model for the other image, as well, which is not always correct (zoomed-out images).

A way to cope with the described inconsistence would be to employ a non-linear optimization. The self-calibration step, along with a linear structure from motion method, provides an initial guess for camera parameters (internal and external ones). Then, it would be sufficient to use an extended projection model (including distortion) in a bundle-adjustment setting.

Overall, the experiments validate that our self-calibration method is reliable, for any stereo pair. The unknown magnification parameter can be recovered with even 2%-accuracy, provided that the stereo pair is composed of images of similar resolutions. Therefore, it is more convenient to use a minimal zoom reference view to self-calibrate zoomed-out images, and a maximal zoom reference one, for more zoomed-in images.

7. 3D reconstruction

We applied the described technique on a stereo pair of images of a chimney (Figure 6). The only knowledge we had, was that both images were taken with our camera, and that one of them was taken with the minimal possible zoom. Self-calibration provided us with an estimation for the unknown magnification parameter for the second setting of the camera: $\alpha = 980$.

Reliability of the obtained reconstruction of the chim-

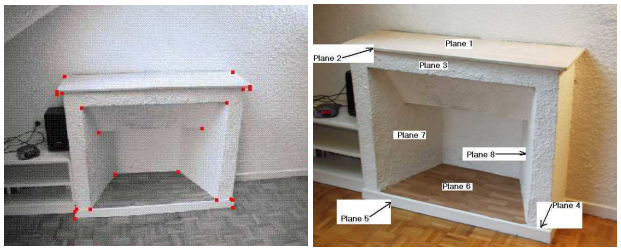


Figure 6. Stereo pair of a chimney (the reference image on the left).

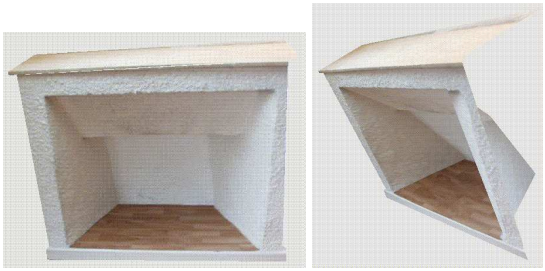


Figure 7. Reconstructed chimney.

ney (Figure 7), with correctly retrieved depth and angles between specified planes (Table 5), certifies a high quality of the performed calibration, and thus, capability to recover the Euclidean structure.

8. Conclusion

We have presented a method to simplify the self-calibration process of a zooming camera, based only on information of a boundary (minimal or maximal) zoom, used for taking one of the images. Due to the off-line calibration preprocessing, the on-line self-calibration step has only one parameter to estimate, and thus, only one family of critical motion sequences for cameras to deal with (a situation that is not valid for more complex cases of self-calibration). We provide a close-form solution for the problem and present experiments on real images that validate the stability and reliability of our method.

The proposed combined calibration technique can be easily used in various applications, as quite often one is provided with at least one reference image. The complex problem of dealing with wide, differently zoomed views of a scene, is decomposed into several simpler tasks, which is an important advantage of the presented approach.

Acknowledgements. We would like to thank Frederik Schaffalitzky from the Visual Geometry Group in Oxford, for making accessible a matching software.

Plane	*3*	*5*	*6*	*7*	*8*
1	88 (90)	88 (90)	2 (0)	89 (90)	89 (90)
3	-	2 (0)	91 (90)	71 (70)	114 (115)
5	-	-	90 (90)	70 (70)	113 (115)
6	-	-	-	88 (90)	90 (90)
7	-	-	-	-	43 (45)

Table 5. Angles (in [deg]) between chosen planes of the chimney: retrieved values, and the real ones (in parentheses).

References

[1] M. Armstrong, A. Zisserman, and R. Hartley, "Self-calibration from image triplets", *ECCV*, pp.3-16, 1996.

[2] A. Baumberg, "Reliable feature matching across widely separated views", *CVPR*, pp. 774-781, 2000.

[3] S. Bougnoux, "From projective to euclidean space under any practical situation, a criticism of self-calibration", *ICCV*, pp. 790-796, 1998.

[4] E. Boyer, *Reconstruction de surfaces d'objets courbes en vision par ordinateur*, PhD thesis, INP de Lorraine, 1996.

[5] Y. Dufournaud, C. Schmid, and R. Horaud, "Matching images with different resolutions", *CVPR*, 2000.

[6] O. Faugeras, "What can be seen in three dimensions with an uncalibrated stereo rig?", *ECCV*, pp. 563-578, 1992.

[7] R.I. Hartley, "Estimation of relative camera positions for uncalibrated cameras", *ECCV*, pp. 579-587, 1992.

[8] R.I. Hartley, "Self-calibration from multiple views with a rotating camera", *ECCV*, pp. 471-478, 1994.

[9] R.I. Hartley, "Kruppa's equations derived from the fundamental matrix", *PAMI*, 19(2), pp. 133-135, 1997.

[10] R.I. Hartley, R. Gupta, and T. Chang, "Stereo from uncalibrated cameras", *CVPR*, pp. 761-764, 1992.

[11] A. Heyden and K. Åström, "Minimal conditions on intrinsic parameters for euclidean reconstruction", *ACCV*, vol. II, pp. 169-176, 1998.

[12] E. Malis and R. Cipolla, "Multi-view constraints between collineations: Application to self-calibration from unknown planar structures", *ECCV*, pp. 610-624, 2000.

[13] M. Pollefeys, R. Koch, and L. Van Gool, "Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters", *ICCV*, pp. 90-95, 1998.

[14] P. Sturm, "Critical motion sequences for the self-calibration of cameras and stereo systems with variable focal length", *BMVC*, pp. 63-72, 1999.

[15] B. Triggs, "Autocalibration from planar scenes", *ECCV*, 1998.

[16] R.G. Willson and S.A. Shafer, "Modeling and calibration of zoom lenses", In *Camera Calibration and Orientation Determination*, Springer-Verlag, 1993.

[17] C. Zeller and O. Faugeras, "Camera self-calibration from video sequences: the Kruppa equations revisited", Technical report 2793, INRIA, 1996.

Chapter 5

Camera Self-Calibration

5.1 Crititcal Motions for Kruppa Equations

Paper 5 [21]: P. Sturm. A case against Kruppa's equations for camera self-calibration. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 22(10):1199–1204, October 2000.

5.2 Focal Length Self-Calibration

Paper 6 [23]: P. Sturm. Critical motion sequences for the self-calibration of cameras and stereo systems with variable focal length. *Image and Vision Computing*, 20(5-6):415–426, 2002.

Paper 7 [28]: P. Sturm, Z.L. Cheng, P.C.Y. Chen, and A.N. Poo. Focal length calibration from two views: Method and analysis of singular cases. *Computer Vision and Image Understanding*, 99(1):58–95, July 2005.

5.3 Self-Calibration for Planar Motions

Paper 8 [10]: O. Faugeras, L. Quan, and P. Sturm. Self-calibration of a 1d projective camera and its application to the self-calibration of a 2d projective camera. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 22(10):1179–1185, October 2000.

5.4 Plane-Based Self-Calibration

Paper 9 [12]: P. Gurdjos and P. Sturm. Methods and geometry for plane-based self-calibration. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Madison, Wisconsin, USA*, volume 1, pages 491–496, June 2003.

5.5 Optimal Fundamental Matrix Estimation

Paper 10 [3]: A. Bartoli and P. Sturm. Non-linear estimation of the fundamental matrix with minimal parameters. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 26(4):426–432, 2004.

IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 22, NO. 10, OCTOBER 2000 1199

A Case Against Kruppa's Equations for Camera Self-Calibration

Peter Sturm

Abstract—We consider the self-calibration problem for perspective cameras and especially the classical Kruppa equation approach. It is known that for several common types of camera motion, self-calibration is degenerate, which manifests itself through the existence of ambiguous solutions. In a previous paper, we have studied these *critical motion sequences* and have revealed their importance for practical applications. Here, we reveal a type of camera motion that is not critical for the generic self-calibration problem, but for which the Kruppa equation approach fails. This is the case if the optical centers of all cameras lie on a sphere and if the optical axes pass through the sphere's center, a very natural situation for 3D object modeling from images. Results of simulated experiments demonstrate the instability of numerical self-calibration algorithms in near-degenerate configurations.

Index Terms—Self-calibration, calibration, euclidean reconstruction, Kruppa equations, critical motions, degeneracy, absolute conic.

1 Introduction

We consider the self-calibration problem for perspective cameras. By self-calibration, we mean the recovery of a camera's intrinsic parameters by only using information contained in images taken by this camera. Explicitly, no information on camera motion or on the 3D structure of the environment is used.

It has been shown by Maybank and Faugeras that, if the camera's calibration remains fixed over an image sequence, self-calibration is in general possible [7]. This result is based on the so-called *Kruppa equations*, that link the camera's intrinsic parameters with the epipolar geometry of pairs of views taken by the camera. The epipolar geometry can be estimated from sole image point correspondences, so Kruppa's equations put constraints on the intrinsic parameters and can thus be used for self-calibration. Several practical self-calibration approaches based on Kruppa's equations have subsequently been proposed by Faugeras and students of his [1], [5], [6], [16]. For other self-calibration approaches, see, for example, [2], [4], [9], [10], [15].

It is known that several types of camera motion exist, for which self-calibration is a degenerate problem, i.e., there exist ambiguous solutions. In [12], [13], we report on a complete study of the *critical motion sequences*. The problem of degeneracy must be taken into account in practical self-calibration since several very common imaging situations are indeed critical.

In Section 4, we describe a configuration, that is not critical for generic self-calibration, but for which approaches based on Kruppa's equations fail. Concretely, we show that this is the case if all optical centers lie on a sphere and if the optical axes pass through the sphere's center—a situation that appears frequently in 3D object modeling from photographs or image sequences.

In Section 2, we briefly introduce a theory of self-calibration on which the rest of the paper is based. Kruppa's equations are reviewed in Section 3. In Section 5, we examine the instability of Kruppa equation-based approaches for self-calibration in near-degenerate situations, through numerical experiments. In Section 6, results are shown that underline the fact that other approaches may perform well even in situations that are exactly degenerate for Kruppa's equations. Section 7 discusses some special cases of the degeneracy considered in this paper, whereas Section 8 provides a general discussion on why certain types of methods suffer from certain types of degeneracies.

• *The author is with INRIA Rhône-Alpes, 655 Avenue de l'Europe, 38330 Montbonnot, St Martin, France. E-mail: Peter.Sturm@inrialpes.fr*

Manuscript received 14 Aug. 1998; revised 5 May 2000; accepted 10 May 2000.
Recommended for acceptance by M. Black.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 107989.

2 Self-Calibration and Euclidean Reconstruction

We consider camera (self-) calibration as an intermediate step to the recovery of metric 3D structure, also called Euclidean reconstruction. In geometrical terms, obtaining a Euclidean reconstruction is equivalent to determining the position of the absolute conic Ω_∞ [11]. The calibration of a camera's intrinsic parameters is equivalent to the determination of the absolute conic's *projection*. The absolute conic is characterized as being the only conic in three-space that is invariant to Euclidean transformations. A consequence of this is that even under arbitrary camera displacements, the projection of Ω_∞ remains fixed, if the camera's calibration does not change during the displacement. This property gives us a constraint for the determination of the absolute conic and its projection and, thus, for Euclidean reconstruction and self-calibration.

Euclidean reconstruction or self-calibration can, thus, be formulated as the determination of the unique conic in three-space, whose projections are identical in all views of a given image sequence [13]. In this paper, we consider perspective projection as camera model, whose intrinsic parameters are described in the next section. Most practical self-calibration approaches start with a global projective reconstruction and try to identify the absolute conic in 3D; approaches based on Kruppa's equations rely on the epipolar geometry of pairs of views, trying to identify the image of the absolute conic in 2D using local information.

3 Kruppa's Equations

Kruppa's equations can be considered as an epipolar matching constraint for the projections of quadrics or conics. Consider Fig. 1, where the case of a quadric's projection in two views is illustrated. Two epipolar planes are tangent to the quadric and the induced epipolar lines in the images are, thus, tangent to the conics obtained by projection of the quadric. Hence, an epipolar line that is tangent to an image conic corresponds to an epipolar line that is tangent to the conic in the other image. The same kind of epipolar constraint is valid if the conics in the images are obtained by projection of a *conic* in three-space, instead of a quadric.

If we consider the projections of the absolute conic, we obtain a special case of this conic matching constraint, since the image conics are identical when the images are taken by a camera with fixed intrinsic parameters (cf. Section 2). Let ω be the projection of the absolute conic. The matching constraint can be expressed in the following form [16]:

$$\mathbf{F}\,\omega^{-1}\,\mathbf{F}^{\mathrm{T}} \;\sim\; [\mathbf{e}']_\times\,\omega^{-1}\,[\mathbf{e}']_\times\,, \tag{1}$$

where $\mathbf{F}$ is the fundamental matrix of the two views, $\mathbf{e}'$ is the second epipole (the first epipole $\mathbf{e}$ is not used in this formula), $\sim$ means equality up to scale (we work in homogeneous coordinates) and $[\mathbf{e}']_\times$ is the skew-symmetric matrix associated with the cross-product of $\mathbf{e}'$. Equation (1) is one formulation of Kruppa's equations. It links the intrinsic parameters of the camera (represented by the image ω of the absolute conic) with the epipolar geometry (represented by $\mathbf{F}$ and $\mathbf{e}'$). Since the epipolar geometry can be estimated from sole image correspondences, Kruppa's equations can be used for self-calibration.

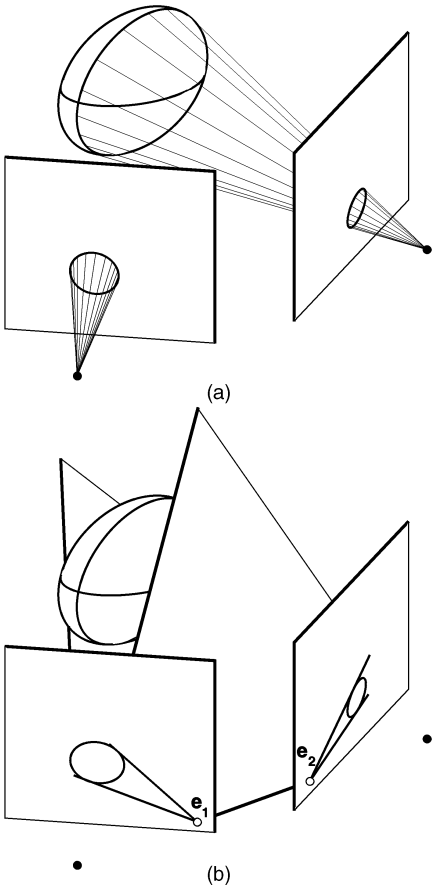


Fig. 1. (a) The image of a quadric is the projection of its silhouette, as seen from the optical center of the camera. (b) Two epipolar planes are tangent to the quadric; the associated epipolar lines are tangent to the images of the quadric.

Once the image ω of the absolute conic has been determined using Kruppa's equations or any other approach, the intrinsic parameters are determined straightforwardly, as described in the following. Let the calibration matrix of the camera be given by:

$$\mathtt{K} = \begin{pmatrix} \tau\alpha & -\tau\alpha\cot\Theta & \mathtt{u}_0 \\ 0 & \alpha/\sin\Theta & \mathtt{v}_0 \\ 0 & 0 & 1 \end{pmatrix},$$

where τ is the aspect ratio, α the focal length (in pixels), (u_0, v_0) the principal point, and Θ the skew angle between pixel axes. The image of the absolute conic in a view with calibration $\mathtt{K}$ is, independently of the view's extrinsic parameters, given by:

$$\omega \sim \mathtt{K}^{-\mathrm{T}}\mathtt{K}^{-1}.$$

Hence, once ω is known, the intrinsic parameters are easily determined by Cholesky decomposing ω, using the property of the calibration matrix being upper triangular. Equivalently, but slightly more conveniently, we may decompose the dual of the absolute conic's image:

$$\omega^{-1} \sim \mathtt{K}\mathtt{K}^{\mathrm{T}}. \tag{2}$$

4 A Degenerate Case for Kruppa's Equations

In this section, we consider the case where the camera to be self-calibrated moves on a sphere while its optical axis points towards the sphere's center. This type of camera motion is not critical for the generic self-calibration problem, but Kruppa's equations are degenerate, which is demonstrated in the following.[1]

Let $\mathbf{C}$ be the center of the viewing sphere (i.e., the sphere of camera positions). Consider an arbitrary sphere Φ that is also centered in $\mathbf{C}$. Obviously, since the camera is always pointed towards $\mathbf{C}$, the sphere Φ is perceived in the same way in all views, i.e., its projections are identical. Let ϕ be the conic representing the sphere's projections. The fact that ϕ is the identical projection of a quadric into all views means nothing else than that ϕ satisfies Kruppa's equations for each pair of views. Hence, ϕ gives us a mathematically valid, but wrong, solution for the self-calibration problem. We did not constrain the radius of sphere Φ, which means that there is a whole family of ambiguous solutions for self-calibration. Note that this degeneracy is independent of scene structure: The sphere Φ does not have to exist in the real world—it is a purely algebraic object, like the absolute conic.

In the following, we examine the nature of the ambiguous solutions, i.e., which intrinsic parameters are affected in which way. Let the camera's distance from $\mathbf{C}$ be d and denote spheres centered in $\mathbf{C}$ and with (possibly imaginary) radius r by Φ_r. It is easy to verify that the dual of the image of Φ_r is identical in all views and is given by the matrix ϕ_r^{-1}:

$$\phi_r^{-1} \sim \mathtt{K}\begin{pmatrix} \frac{r^2}{r^2-d^2} & 0 & 0 \\ 0 & \frac{r^2}{r^2-d^2} & 0 \\ 0 & 0 & 1 \end{pmatrix}\mathtt{K}^{\mathrm{T}}. \tag{3}$$

Since ϕ_r is satisfying Kruppa's equations (1), we may try to extract intrinsic parameters from it, as described in Section 3, for the true image of the absolute conic. By Cholesky decomposing ϕ_r^{-1}, we obtain an upper triangular calibration matrix $\mathtt{K}_r$ satisfying $\phi_r^{-1} \sim \mathtt{K}_r\mathtt{K}_r^{\mathrm{T}}$. From (2) and (3), it follows that $\mathtt{K}_r$ is given by:

$$\mathtt{K}_r = \mathtt{K}\begin{pmatrix} \frac{r}{\sqrt{r^2-d^2}} & 0 & 0 \\ 0 & \frac{r}{\sqrt{r^2-d^2}} & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

With regard to the following decomposition of $\mathtt{K}$, separating the focal length α from the other intrinsic parameters:

$$\mathtt{K} = \begin{pmatrix} \tau & -\tau\cot\Theta & \mathtt{u}_0 \\ 0 & 1/\sin\Theta & \mathtt{v}_0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} \alpha & 0 & 0 \\ 0 & \alpha & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

the ambiguous solution for the calibration matrix, due to ϕ_r, is given by:

$$\mathtt{K}_r = \begin{pmatrix} \tau & -\tau\cot\Theta & u_0 \\ 0 & 1/\sin\Theta & v_0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} \frac{r}{\sqrt{r^2-d^2}}\alpha & 0 & 0 \\ 0 & \frac{r}{\sqrt{r^2-d^2}}\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

We observe that the intrinsic parameters given by $\mathtt{K}_r$ are identical with the true parameters in $\mathtt{K}$, with the exception of the focal length, which is given by:

$$\alpha_r = \frac{r}{\sqrt{r^2-d^2}}\,\alpha. \tag{4}$$

This result has two implications: First, the focal length is usually the intrinsic parameter one is most interested in when self-calibrating (the aspect ratio being constant and often precisely known, the principal point being close to the image center) and it is embarassing that exactly this parameter is obstructed. Second, even prior knowledge of the other intrinsic parameters (aspect ratio, principal point, skew angle) does not help in resolving the ambiguity in focal length estimation!

1. Triggs already pointed out that this configuration is degenerate for the quasi-linear approach proposed in [15]. Relationships between degeneracies for different types of self-calibration approaches are discussed in Section 8.

We now have a closer look at the factor $s = \frac{r}{\sqrt{r^2 - d^2}}$ in (4). Depending on the radius r of the sphere Φ_r, this factor takes imaginary or real values. The focal length being a real number, we are interested in the cases when s is real. It is easy to check that this is valid exactly in the following two cases:

- r is real and larger or equal than d, i.e., Φ_r is a real sphere of equal or larger radius than the viewing sphere. In this case, we have $s \in [1, \infty]$.
- r is a multiple of $I = \sqrt{-1}$, i.e., Φ_r is a sphere of imaginary points only. In this case, we have $s \in (0, 1]$.

In conclusion, all nonzero real values are mathematically valid solutions for the ambiguous focal length α_r.

The reason for Kruppa's equations having ambiguous solutions in a configuration that is not critical for the generic self-calibration problem, may be resumed by the following phrase. Namely, Kruppa's equations are constraints on the *image* of the absolute conic, but they do not enforce the *planarity* of the absolute conic in 3D, which is exactly why, in our case, the projections of *spheres* are admitted as solutions. This issue is discussed in more detail in Section 8.

The degeneracy of Kruppa's equations we revealed in this section has been observed in experiments by Zeller and Faugeras [16]: In their self-calibration approach, a global optimization stage is initialized by a robust fit of estimates of the aspect ratio and focal length, obtained from pairs of views and with the principal point being supposed known. The two-view results of the focal length were reported to be extremely unstable, whereas the aspect ratio is estimated reliably. Zeller and Faugeras were not aware that the reason for this seems to be the numerical instability caused by the camera configuration used for their experiments, which is close to the configuration dealt with in this paper. However, during the global optimization stage, when all views are taken into account simultaneously, the instability seems to be reduced enough to obtain good results, which is possible since the camera configuration is only near to, but not exactly, degenerate.

5 Instability for Near-Degenerate Camera Configurations

In the following, we report on numerical simulations that have been designed to reveal the instability caused by near-degenerate camera configurations. It is not intended to give a complete quantitative analysis of the problem, but to demonstrate the effect of near-degeneracy on numerical algorithms.

The basic simulated camera setup is as follows: The scene consists of 50 3D points that are randomly chosen in a sphere of radius 100, centered in the origin. We place the camera at arbitrary positions on the sphere of radius 200, that is also centered in the origin. The camera's calibration is fixed to:

$$\mathtt{K} = \begin{pmatrix} 1,000 & 0 & 256 \\ 0 & 1,000 & 256 \\ 0 & 0 & 1 \end{pmatrix}$$

and the camera is rotated such that it focuses the origin (the viewing sphere's center).

The following variations and perturbations are applied in various combinations:

- The camera's orientation is changed such that the camera focuses a randomly chosen point within a given distance from the origin (a different point for each view). This distance will be referred to as "optical axis offset" and varies between zero and 10.
- The camera is translated off the viewing sphere, toward the origin. Its distance from the origin is reduced from 200 to 180. The number of translated cameras will be referred to as "number of views off viewing sphere" in the following.
- Gaussian noise of standard deviation one pixel is added to the coordinates of the image points.

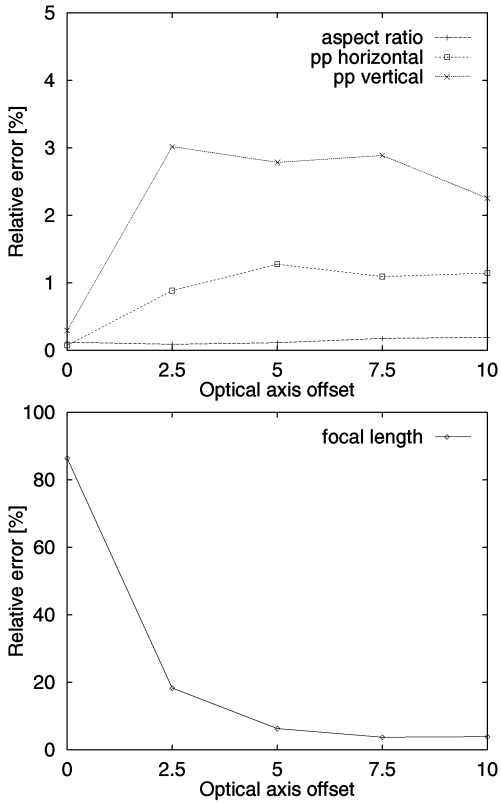


Fig. 2. The camera is placed exactly on the viewing sphere, but the optical axis is rotated away from the sphere's center by various amounts. The labels "pp horizontal" and "pp vertical" refer to the principal point coordinates u_0 and v_0. Note the different scales in the graphs.

The first two actions move the configuration away from being critical for Kruppa's equations.

The estimation of the intrinsic parameters is carried out as follows. First, fundamental matrices between pairs of views are estimated by a quasi-linear method [3]. The intrinsic parameters are estimated by a Levenberg-Marquardt type optimization scheme, minimizing a criterion based on Kruppa's equations. The intrinsic parameters are initialized with their *true* values. Thus, large errors in the estimated parameters indicate large instabilities caused by near-degenerate configurations.

For each setup, we carried out 20 different experiments. In the following graphs, median relative errors for the focal length, the aspect ratio and the coordinates of the principal point are shown. For all of the results shown, eight views have been used for self-calibration.

Fig. 2 shows the situation when all cameras are placed exactly on the viewing sphere, but their optical axes are rotated away from the center by various amounts. As the theory in the previous section suggests, the aspect ratio and the principal point are estimated quite reliably even in the exactly degenerate situation when there is no optical axis offset. The error on the focal length however, is, as expected, very large for the degenerate case (note the different scales in the upper and lower parts of Fig. 2) and only reaches the same level as for the other parameters, when the optical axis offset becomes significant.

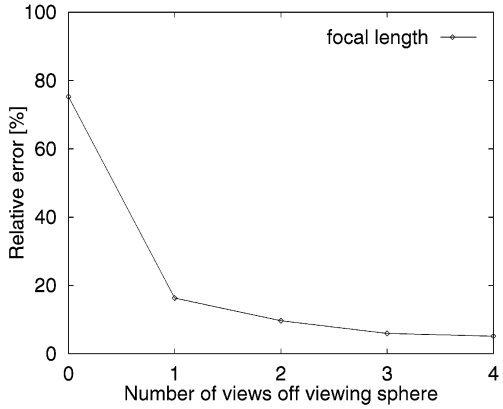


Fig. 3. Between zero and four of the eight views are translated away from the viewing sphere. There is no optical axis offset.

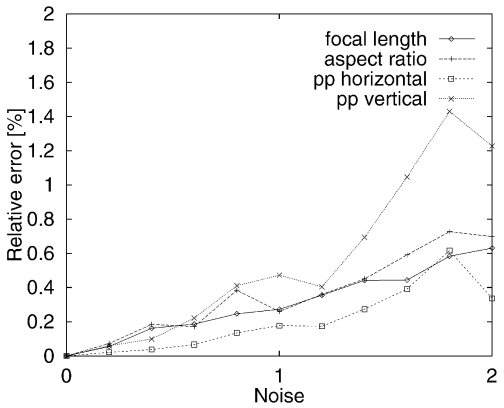


Fig. 5. Relative errors for intrinsic parameters with respect to noise in the image points.

Fig. 3 shows the situation when between zero and four of the eight views are translated away from the viewing sphere, but there is no optical axis offset. Again, the aspect ratio and the principal point are estimated with less than five percent of error (graph not shown). As for the focal length estimation, it is interesting to note that even with only one view taken from a position not on the viewing sphere, the error decreases dramatically, but half of the views have to be translated in order to come close to the five percent error level.

Finally, Fig. 4 shows the case of both optical axis offset and views translated away from the viewing sphere. The optical axis offset is such that the cameras focus points within five units distance of the viewing sphere's center. It is worth noting that the focal length error drops to less than the half when half of the views are translated off the viewing sphere.

6 Solving the Present Case

As we mentioned, the camera configuration discussed in this paper is not inherently degenerate for self-calibration, but for approaches based on Kruppa's equations (and others, cf. Section 8). Nonlinear methods which include the planarity constraint for the absolute conic, e.g., [2], [4], [9], [15], will in general succeed in self-calibration (cf. also Section 8). To demonstrate this, we designed a simple method to resolve the ambiguity introduced by Kruppa's equations. Since the other intrinsic parameters beside the focal length are estimated well by a Kruppa equation approach, we adopt these and, in a second step, apply a method for estimating the focal length only. We want to use the same input as for Kruppa's equations, i.e., fundamental matrices for pairs of views. An alternative is the method proposed in [10], which is based on a global projective reconstruction.

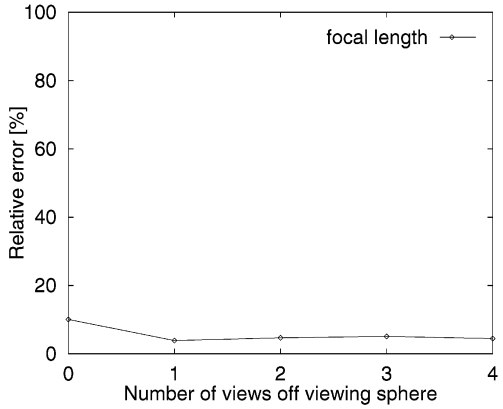


Fig. 4. Between zero and four of the eight views are translated away from the viewing sphere. The optical axes are offset.

In general, the focal length can be computed from a single pair of views, given the epipolar geometry. Our camera configuration, however, is degenerate for this problem [8]. For triplets of views, focal length estimation is no longer degenerate in general [14]. For each triplet, it is possible to obtain 12 equations of degree four in the focal length, with coefficients depending on the three fundamental matrices. These equations can be solved individually and their solutions combined in a robust manner to provide an estimate for the focal length. Details of this method are omitted due to lack of space, please contact the author for further information.

Fig. 5 shows the relative errors, with respect to the amount of Gaussian noise added to the image points, for the aspect ratio and the principal point (estimated using Kruppa's equations) and the focal length (estimated subsequently). The low error for the focal length confirms that our camera configuration is not inherently critical for self-calibration.

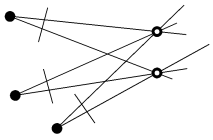
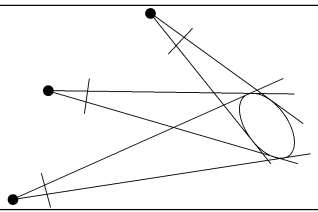
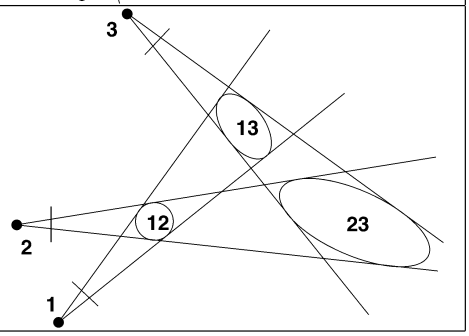
7 Spurious Solutions

It can be shown that in general, the ambiguous solutions for the focal length described in Section 4 represent the only degeneracies for Kruppa's equations with the considered camera configuration. There are, however, special cases where further solutions exist, prohibiting the estimation of other intrinsic parameters as well. Additional ambiguities arise, for example, if there exist quadrics other than spheres whose projections are identical in all views. As we discuss in Section 8, it would be very difficult to give an exhaustive list of all additional degeneracies and, thus, to derive conditions under which certain intrinsic parameters can be estimated for sure. However, it is possible to derive some sufficient conditions under which certain parameters can *not* be estimated without ambiguity. In the following, we describe such conditions for three special cases of the camera configuration discussed in this paper.[2]

The first case concerns cameras located on a circle and fixating this circle's center. We suppose that the cyclotorsion (rotation about the optical axis) is the same for all the views. This configuration is inherently degenerate for self-calibration, giving rise to a two-degree-of-freedom family of solutions [13]. For

2. Results are just summarized here; for details contact the author.

TABLE 1
Different Levels of Degeneracy, Affecting Different Types of Self-Calibration Methods

Methods	Reason for degeneracy	2D sketch of the reason for degeneracy
Any	Existence of a single conic (represented by two points here), different from the absolute conic, with identical projections in all views	
Linear methods	Existence of a single quadric (represented by an ellipse here) with identical projections in all views	
Kruppa equations	Existence of one quadric per pair of views, the projections of which in the associated pair of views being identical with those of the other quadrics in the respective views	

For easier understanding, degeneracies are illustrated in 2D, i.e., conics are shown as two points, while quadrics are shown as conics. Each method inherits the degeneracies from the levels above it.

Kruppa's equations, the ambiguity has at least three degrees of freedom. The following conditions hold for cameras with rectangular pixels (i.e., $\Theta = 90°$; similar, but more complicated, conditions can be derived for nonrectangular pixels). The coordinates of the principal point can never be estimated both at the same time. If the cameras are not upright (i.e., none of the two pixel axes lies in the plane of motion), then none of the coordinates of the principal point can be estimated. If the cyclotorsion is such that none of the pixel axes forms a 45 degree angle with the plane of motion, then the aspect ratio can not be estimated. The focal length can never be estimated since we are in a special case of Section 4.

The second special case consists again of a camera moving on a circle, but fixating an arbitrary point on the circle's axis, i.e., the optical axes do not lie in the plane of motion. The third case is an extension of this, considering two such sets of cameras, arranged symetrically (i.e., the locus of camera positions is the union of two "parallel" circles of the same size). The ambiguity conditions here are essentially the same as in the first case, except for the aspect ratio, where the 45 degree cyclotorsion constraint has to be adapted appropriately to account for the inclination of the optical axes with respect to the plane(s) of motion.

8 Levels of Degeneracy

We briefly explain that self-calibration methods may be divided into at least three groups, suffering from increasing levels of degeneracy, i.e., for which increasingly many critical motions exist. Inherent degeneracies, i.e., degeneracies concerning *any* method, occur exactly if there is a proper virtual conic (i.e., a conic with no real points) in three-space, different from the absolute conic, whose projections are identical in all views of an image sequence [13].

Methods that do not enforce the *planarity*[3] of the absolute conic, suffer from additional degeneracies: camera configurations for which there is a *quadric*, whose projections are identical in all views, are degenerate. Kruppa's equations are one example: The epipolar matching constraint they represent can not distinguish between quadrics and conics in three-space. Another example is the linear method proposed by Triggs [15]: The planarity of the absolute conic is a nonlinear constraint, thus omitted in the linear approach, causing the degeneracy.

In the following, we explain that there exist even more degeneracies for Kruppa's equations, which do not concern the other self-calibration methods cited in this paper. Suppose that ω is an ambiguous solution for Kruppa's equations. This means that for each pair of views i and j, there exists a quadric Φ_{ij}, which projects to ω in both views. However, Kruppa's equations do not constrain these quadrics to be the same for any pair of views!

The three levels of degeneracies are summarized in Table 1. Clearly, from top to bottom, there is more and more room for the existence of degenerate configurations. It seems to be quite difficult to describe all the degeneracies for Kruppa's equations explicitly, which is why in Section 7 we only give some sufficiency conditions for the existence of spurious solutions.

9 Conclusion

In this paper, we have considered the camera self-calibration problem and the classical practical approach, based on Kruppa's equations. We have revealed that for one of the most natural imaging situations (camera moving on a sphere while focusing the

3. Planarity of the absolute conic is equivalent to rank-three-ness of the "absolute quadric" used in [15].

sphere's center) this approach fails, in spite of self-calibration being possible in general. Precisely the focal length can not be estimated, even when the other intrinsic parameters are known.

The occurrence of serious numerical instabilities due to near-degenerate camera configurations has been demonstrated by experiment. However, our results suggest that it should be relatively easy to avoid this problem in practice, either by introducing sufficient variation in the camera placement or by using a method that does not suffer from the degeneracy.

We have shown informally that Kruppa's equations suffer from more degeneracies than other known self-calibration methods, which is a prize paid for using local information (fundamental matrices), as opposed to starting with a global projective reconstruction. In general, this paper contributes to the understanding of how to successfully apply self-calibration, which needs good algorithms, but also, awareness of degenerate situations.

ACKNOWLEDGMENTS

The author would like to thank Bill Triggs for fruitful discussions. This work is partly supported by the EPSRC funded project GR/K89221 (Vector).

REFERENCES

[1] O.D. Faugeras, Q.T. Luong, and S.J. Maybank, "Camera Self-Calibration: Theory and Experiments," *Proc. European Conf. Computer Vision,* pp. 321-334, May 1992.

[2] R.I. Hartley, "Euclidean Reconstruction from Uncalibrated Views," *Proc. Workshop Applications of Invariants in Computer Vision,* pp. 187-202, Oct. 1993.

[3] R.I. Hartley, "In Defence of the 8-Point Algorithm," *Proc. Int'l Conf. Computer Vision,* pp. 1,064-1,070, June 1995.

[4] A. Heyden and K. Åström, "Euclidean Reconstruction from Constant Intrinsic Parameters," *Proc. Int'l Conf. Pattern Recognition,* vol. 1, pp. 339-343, Aug. 1996.

[5] Q.T. Luong, "Matrice Fondamentale et Autocalibration en Vision par Ordinateur," doctoral thesis, Université de Paris-Sud, Orsay, France, 1992.

[6] Q.T. Luong and O.D. Faugeras, "An Optimization Framework for Efficient Self-Calibration and Motion Determination," *Proc. Int'l Conf. Pattern Recognition,* pp. 248-252, Oct. 1994.

[7] S.J. Maybank and O.D. Faugeras, "A Theory of Self Calibration of a Moving Camera," *Int'l J. Computer Vision,* vol. 8, no. 2, pp. 123-151, 1992.

[8] G.N. Newsam, D.Q. Huynh, M.J. Brooks, and H.P. Pan, "Recovering Unknown Focal Lengths in Self-Calibration: An Essentially Linear Algorithm and Degenerate Configurations," *Proc. ISPRS-Congress,* vol. 31, part B3, pp. 575-580, 1996.

[9] M. Pollefeys and L. Van Gool, "A Stratified Approach to Metric Self-Calibration," *Proc. Conf. Computer Vision and Pattern Recognition,* pp. 407-412, June 1997.

[10] M. Pollefeys, R. Koch, and L. Van Gool, "Self-Calibration and Metric Reconstruction in Spite of Varying and Unknown Internal Camera Parameters," *Proc. Int'l Conf. Computer Vision,* pp. 90-95, Jan. 1998.

[11] J.G. Semple and G.T. Kneebone, *Algebraic Projective Geometry.* Oxford Science Publication, 1952.

[12] P. Sturm, "Critical Motion Sequences for Monocular Self-Calibration and Uncalibrated Euclidean Reconstruction," *Proc. Conf. Computer Vision and Pattern Recognition,* pp. 1,100-1,105, June 1997.

[13] P. Sturm, "Vision 3D Non calibrée: Contributions a la Reconstruction Projective et Étude des Mouvements Critiques pour L'Auto-Calibrage," doctoral thesis, INPG, Grenoble, France, 1997.

[14] P. Sturm, "Critical Motion Sequences for the Self-Calibration of Cameras and Stereo Systems with Variable Focal Length," *Proc. British Machine Vision Conf.,* pp. 63–72 Sept. 1999.

[15] B. Triggs, "Autocalibration and the Absolute Quadric," *Proc. Conf. Computer Vision and Pattern Recognition,* pp. 609-614, June 1997.

[16] C. Zeller and O. Faugeras, "Camera Self-Calibration from Video Sequences: the Kruppa Equations Revisited," Research Report 2793, INRIA, Feb. 1996.

ELSEVIER

Image and Vision Computing 20 (2002) 415–426

image AND vision COMPUTING

www.elsevier.com/locate/imavis

Critical motion sequences for the self-calibration of cameras and stereo systems with variable focal length

Peter Sturm*

INRIA Rhône-Alpes, 655 Avenue de l'Europe, 38330 Montbonnot St Martin, France

Received 16 October 2000; accepted 18 December 2001

Abstract

We consider the self-calibration problem for a moving camera whose intrinsic parameters are known except for the focal length which may vary freely across different views. The conditions, under which the determination of the focal length's values for an image sequence is not possible, are derived. These depend only on the camera's motion. We give a complete catalogue of the so-called *critical motion sequences*. This is then used to derive the critical motion sequences for stereo systems with variable focal lengths. © 2002 Published by Elsevier Science B.V.

1. Introduction

One of the major goals of computer vision is the recovery of spatial information about the environment. Classical approaches assume that the cameras are *calibrated* beforehand but a great interest in *uncalibrated* vision and on-line calibration has arisen during the last decade. A key result is that even with completely uncalibrated cameras, spatial information—*projective structure*—can be obtained: the scene can be reconstructed up to an unknown projective transformation [6,9]. Furthermore, a moving camera can *self-calibrate*, i.e. the calibration parameters can be estimated solely from feature correspondences between several images [11,13]. This allows the projective ambiguity in the reconstruction to be reduced to a Euclidean one (up to a similarity transformation) and we speak of *uncalibrated Euclidean reconstruction*.

It is known that several types of camera motion prevent self-calibration, i.e. the calibration parameters cannot be determined uniquely. Accordingly, Euclidean structure cannot be obtained, although reconstruction at some level between projective and Euclidean is generally possible. These degeneracies are inherent, i.e. they cannot be resolved by any algorithm without additional knowledge. Sequences of camera motions that imply such degeneracies will be referred to as *critical motion sequences*. By 'sequences' we mean that not only the motion between two successive views but that over a complete image sequence is critical.

For the basic self-calibration scenario, a moving camera with *fixed* calibration, we derived the critical motion sequences in [17,18]. In this paper, we study the case of a moving camera with *variable* and unknown focal length, but whose other intrinsic parameters are known. A practical self-calibration algorithm was proposed by Azarbayejani and Pentland [1]. Algorithms and closed-form solutions for the two-view case are given, e.g. in Refs. [3–5,8,12]. Newsam et al. derived the critical motions for the two-view case [12]. In this paper, we derive a complete characterization of critical motion *sequences* for any number of views and the critical motions for stereo systems. This paper is an extended version of [19].

The paper is organized as follows. In Section 2 we provide some theoretical background for our approach. The problem of deriving critical motion sequences is formulated in Section 3. The critical motion sequences are derived in Section 4. A summary of the derivations is given in Section 5 and comments are made in Section 6. The critical motions for stereo systems are derived in Section 7 and conclusions are drawn in Section 8.

2. Background

The definitions in this section are mainly taken from Refs. [2,16]. Some of the results for general quadrics are presented only for central conics.

2.1. Notation

We refer to the *plane at infinity* as the *ideal plane* and

* Tel.: +33-4-76-61-52-32; fax: +33-4-76-61-54-54.
E-mail address: peter.sturm@inrialpes.fr (P. Sturm).

0262-8856/02/$ - see front matter © 2002 Published by Elsevier Science B.V.
PII: S0262-8856(02)00012-4

denote it by Π_∞. $\mathscr{P}^n$ is the n-dimensional projective space and $\sim$ means equality up to a scalar factor accounting for the use of homogeneous coordinates. We use the abbreviation PVC for proper virtual conics (see description later).

2.2. Pinhole camera model

We use perspective projection to model cameras. A projection may be represented by a 3×4 projection matrix P that maps points of 3-space to points in 2-space: $q \sim PQ$. We consider only the case of perfect perspective projection, i.e. the projection center does not lie on Π_∞.

With regard to physical cameras, the projection matrix may be decomposed into a *calibration matrix* K and a *pose matrix*. The pose matrix represents the position and orientation of the camera with respect to some world coordinate frame. In general, we distinguish five intrinsic parameters for the perspective projection model: the (effective) focal length f, the aspect ratio τ, the principal point (u_0, v_0) and a skew factor accounting for non rectangular pixels. The skew factor is usually very close to 0 and we ignore it in the following. The calibration matrix may be written as:

$$K = \begin{pmatrix} \tau f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{pmatrix}$$

We decompose the projection matrix as follows:

$$P = K \underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} R & -Rt \\ 0^{\mathrm{T}} & 1 \end{pmatrix}}_{P_c}$$

The matrix P_c is the *canonical projection* and we call its destination the *metric image plane*. The canonical projection depends only on the camera's extrinsic parameters—a rotation matrix R representing its orientation and a 3-vector t representing its position. The calibration matrix K describes an invertible affine transformation from the metric image plane to pixel coordinates.

2.3. Quadrics and conics

A *quadric* in $\mathscr{P}^n$ is a set of points satisfying a quadratic equation in their homogeneous coordinates. Each quadric can be represented by a symmetric $(n + 1) \times (n + 1)$ matrix. A *proper quadric* is a quadric whose matrix has a non zero determinant. *Conics* are planar quadrics; we will not distinguish between a conic and its matrix. A conic in $\mathscr{P}^3$ or *3D conic* is defined by its *supporting plane* and the conic's equation in that plane.

2.4. Virtual quadrics

A *virtual quadric* is a quadric with no real point. All proper virtual conics (PVC) are central [2] and hence can be transformed to *Euclidean normal form* by a Euclidean transformation (principal axis transformation). The Euclidean normal form of a virtual conic is a diagonal matrix of the conic's eigenvalues, which all have the same sign.

2.5. Cones

By *cones* we mean rank-3 quadrics in $\mathscr{P}^3$ with vertex not on Π_∞. A cone is uniquely defined by its vertex and any (conic) section by a plane not containing the vertex. Cones are used in this paper through the notion of the *projection cone* of a 3D conic, i.e. the cone formed by the projection rays of the perspective projection of the conic. The Euclidean normal form of a cone is a diagonal matrix $\mathrm{diag}(\lambda_1, \lambda_2, \lambda_3, 0)$ with non zero λ_i. If the λ_i are all distinct then the cone is an *elliptic cone*. If exactly two of the λ_i are equal the cone is *circular* (or *right*). For an *isotropic cone*, all three λ_i are equal. Each isotropic cone contains the absolute conic (see description later).

A circular cone is invariant to arbitrary rotation about a single line passing through its vertex. This line is called the cone's *axis*. An isotropic cone is invariant to any rotation about its vertex.

2.6. Absolute quadric and absolute conic

The *absolute quadric* of $\mathscr{P}^n$ is defined by the equations $x_1^2 + \ldots + x_n^2 = x_{n+1} = 0$. The *absolute conic* Ω is the absolute quadric of P^3. Ω is a proper virtual conic in the ideal plane whose position uniquely defines the Euclidean structure of 3-space. The calibration of a camera is equivalent to determining the image ω of Ω, respectively, its dual ω^* [7,11]. From the relation $\omega^* \sim KK^{\mathrm{T}}$, the calibration matrix K can uniquely be recovered by Cholesky decomposition [15].

3. Problem formulation

We consider a sequence of n views, generally taken from different positions and with different orientations. The focal lengths for the views may all be different and the other intrinsic parameters (aspect ratio and principal point) are known (they need not be equal for all the views). The problem at hand is to perform focal length self-calibration, i.e. to determine the n different values for the focal length, which allows in general to obtain a Euclidean reconstruction of the scene. In the following, we describe this problem in geometrical terms, in analogy to Ref. [18].

First, *calibration* of a camera is equivalent to the determination of the image of the absolute conic, as 'produced' by that camera. *Self-calibration* means the same but with the conotation that information used to calibrate does not stem from, e.g. known metric 3D structure. *Euclidean reconstruction* is equivalent to the determination of the position of the absolute conic in 3D. The problem of Euclidean reconstruction is slightly more general than that of self-calibration: degeneracy of self-calibration implies degeneracy of Euclidean reconstruction while the reciproque is not

always true (e.g. self-calibration of a camera rotating about its optical center is in general possible while any level of 3D reconstruction is impossible, including Euclidean reconstruction). The derivations that follow refer to degeneracies of Euclidean reconstruction.

To determine the position of the absolute conic, some constraints are needed. We will describe these constraints in the following paragraph but first we give a straightforward informal definition for degeneracy of Euclidean reconstruction: the Euclidean reconstruction problem is degenerate exactly if there is a conic in 3D not identical with the absolute conic that satisfies the mentioned constraints [17,18]. All such conics will be called *potential absolute conics*.

We now describe the constraints that may be used to determine the absolute conic. First, the absolute conic must be a proper virtual conic. Second, the image of the absolute conic by any perfect perspective projection is also a proper virtual conic. Third, the knowledge of some intrinsic parameters constrains the *projections* of the absolute conic in a given set of views, which in turn gives us constraints on the absolute conic itself. For the scenario considered here we make these constraints explicit in the following.

The image of the absolute conic, as 'produced' by a camera with calibration matrix K is given by:

$$\omega \sim K^{-\mathrm{T}}K^{-1} \sim \begin{pmatrix} 1 & 0 & -u_0 \\ 0 & \tau^2 & -\tau^2 v_0 \\ -u_0 & -\tau^2 v_0 & u_0^2 + \tau^2 v_0^2 + \tau^2 f^2 \end{pmatrix} \quad (1)$$

Since f may vary and we know the other intrinsic parameters, there is, for each view, exactly one family of possible images of the absolute conic. Consider now a conic Φ in 3D and its projection Φ in one view. For ϕ being a potential absolute conic its projection ϕ must be of the form Eq. (1) for some non zero real value a possibly different from the true f (remember that we suppose that the other intrinsic parameters are known):

$$\phi \sim \begin{pmatrix} 1 & 0 & -u_0 \\ 0 & \tau^2 & -\tau^2 v_0 \\ -u_0 & -\tau^2 v_0 & u_0^2 + \tau^2 v_0^2 + \tau^2 a^2 \end{pmatrix}$$

It is easy to show that a conic has this form exactly if, in the metric image plane, the conic is a virtual circle, centered in the origin. To see this, we map ϕ from pixel coordinates to the metric image plane using the true calibration matrix:

$$\phi_m \sim K^{\mathrm{T}}\phi K \sim \begin{pmatrix} f^2 & 0 & 0 \\ 0 & f^2 & 0 \\ 0 & 0 & a^2 \end{pmatrix}$$

For any non zero real value of a, this represents a proper virtual circle whose center is the coordinate origin. It is important to note that this statement is independent of the actual true value f of the focal length. Since all other intrinsic parameters are known the only important parameters for the consideration of degeneracy of Euclidean reconstruction are the extrinsic parameters of the views in a given sequence.

We now summarize the discussion.

Proposition 1. *Consider a sequence of n views with known aspect ratio and principal point but unknown and possibly different values for the focal length. Let P_{ci} be the canonical projection for view $i, i = 1, \ldots, n$.*

Euclidean reconstruction is degenerate exactly if there is at least one 3D conic Φ not identical with the absolute conic such that:

- *Φ is a proper virtual conic*;
- *the ϕ_i, $i = 1, \ldots, n$, where ϕ_i is the projection of Φ by P_{ci} are proper virtual circles centered in the origin. This is equivalent to the ϕ_i being represented by diagonal matrices whose diagonal elements are all non zero real values of the same sign the first two elements being equal.*

Definition 1. Any Φ as defined in Proposition 1 is called a *potential absolute conic*.

Consider a sequence of n views. Let (R_i, t_i) be the extrinsic parameters of view $i, i = 1, \ldots, n$. If Euclidean reconstruction is degenerate for the sequence of views, we say that $\{(R_i, t_i) | i = 1, \ldots, n\}$ is a *critical motion sequence* for Euclidean reconstruction.

The aim of the following section is to derive all generic critical motion sequences, i.e. all configurations where there is no unique solution to Euclidean reconstruction.

4. Derivation of the critical motion sequences

In this section the critical motion sequences are derived based on Proposition 1 and Definition 1. We proceed in a constructive manner: given a generic proper virtual 3D conic Φ, we determine all possible extrinsic parameters that form a critical motion sequence with respect to Φ, i.e. for which Φ is a potential absolute conic. The derivations are divided into two parts considering potential absolute conics Φ which lie/do not lie on Π_∞. The results are summarized in Section 5.

4.1. Potential absolute conics on Π_∞

Let Φ be a PVC on the ideal plane. Its canonical projection ϕ by a camera with extrinsic parameters (R, t) is given by:

$$\phi \sim R\Phi R^{\mathrm{T}} \quad (2)$$

Like it is the case for all geometric entities on the ideal plane the projection depends only on the orientation of the camera not on its position.

We now determine all orientations R for which Φ is a proper virtual circle centered in the origin. This implies that ϕ is a diagonal matrix of the form: $\phi \sim \mathrm{diag}(b, b, 1)$. If we choose the free scale factor for Φ such that $\det \Phi = b^2 = \det \phi$, the $\sim$ in Eq. (2) can be replaced by an equality sign:

$$\begin{pmatrix} b & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & 1 \end{pmatrix} = R\Phi R^{\mathrm{T}} \tag{3}$$

Eq. (3) implies that Φ has a double eigenvalue b and the single eigenvalue 1. The case $b = 1$ is of no interest here because this would mean that Φ is the *true* absolute conic (the absolute conic is the only conic on the ideal plane with a triple eigenvalue).

From Eq. (3), we derive:

$$R^{\mathrm{T}} \begin{pmatrix} b & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \Phi R^{\mathrm{T}} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

and further:

$$R^{\mathrm{T}} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \Phi R^{\mathrm{T}} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

Hence the vector

$$v_R = R^{\mathrm{T}} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{4}$$

is an eigenvector of Φ to the eigenvalue 1. Since 1 is a single eigenvalue, all its associated eigenvectors are equal up to scale. This means that for all rotation matrices R in a critical motion sequence the vectors v_R must be equal up to scale.

It is easy to show that $(v_R^{\mathrm{T}}, 0)^{\mathrm{T}}$ is nothing else than the ideal point of the optical axis for a camera with orientation R. All v_R being equal up to scale is thus equivalent to the optical axes of all views being parallel. This is thus a necessary condition for critical motion sequences with respect to a conic on the ideal plane.

We now show that this is also a sufficient condition. Remember that eigenvectors of symmetric matrices that are associated to different eigenvalues are mutually orthogonal [2]. Thus, the eigenspace of Φ for the double eigenvalue b consists of all vectors orthogonal to v_R. Let r_i^{T} be the row vector representing the ith row of the rotation matrix R. From Eq (4) we have $r_3 = v_R$. Since R is an orthogonal matrix, we have $r_1 \perp v_R$ and $r_2 \perp v_R$, which means that r_1 and r_2 are eigenvectors of Φ associated to the eigenvalue b. We thus obtain:

$$R\Phi R^{\mathrm{T}} = \begin{pmatrix} r_1^{\mathrm{T}} \\ r_2^{\mathrm{T}} \\ r_3^{\mathrm{T}} \end{pmatrix} \Phi \begin{pmatrix} r_1 & r_2 & r_3 \end{pmatrix} = \begin{pmatrix} r_1^{\mathrm{T}} \\ r_2^{\mathrm{T}} \\ r_3^{\mathrm{T}} \end{pmatrix} \begin{pmatrix} br_1 & br_2 & r_3 \end{pmatrix}$$

$$= \begin{pmatrix} b & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

In conclusion, the projection of Φ is a virtual centered circle exactly if the ideal point of the camera's optical axis is $(v^{\mathrm{T}}, 0)^{\mathrm{T}}$ where v is the eigenvector of Φ associated to its single eigenvalue. Hence, a motion sequence is critical with respect to a PVC on the ideal plane if and only if the optical axes of all the views are parallel.

4.2. Potential absolute conics not on Π_∞

Contrary to conics on the ideal plane the projection of conics not on Π_∞ depends on both camera position and orientation. First, we deal with position then with orientation.

4.2.1. Position

Let Φ be a PVC *not* on the ideal plane. Consider a view with optical center at position t. Let ϕ be the canonical projection of Φ. Let Λ be the projection cone of Φ (cf. Section 2.5).

One condition for Φ being a potential absolute conic is that ϕ is a circle, centered in the origin of the metric image plane. Note that the origin of the metric image plane is the camera's principal point, i.e. the intersection of the optical axis with the image plane. Since the optical axis is perpendicular to the image plane the projection cone Λ must be circular and its axis is the camera's optical axis. The vertex of the projection cone being the optical center t, we obtain constraints on the possible camera positions in a critical motion sequence: for a potential absolute conic Φ all possible camera positions are the vertices of circular cones that contain Φ. These are summarized in the following (proofs are given in Appendix A).

If Φ is a virtual circle then the locus of possible camera positions in a critical motion sequence is the line L perpendicular to the circle's supporting plane and passing through the circle's center (Fig. 1(a)). If Φ is a virtual ellipse then the locus of camera positions is the union of a real ellipse Ψ_{e} and a real hyperbola Ψ_{h} (Fig. 1(b)). The supporting planes of the three conics Φ, Ψ_{e} and Ψ_{h} are mutually perpendicular (there are further relations between these conics, cf. Appendix A).

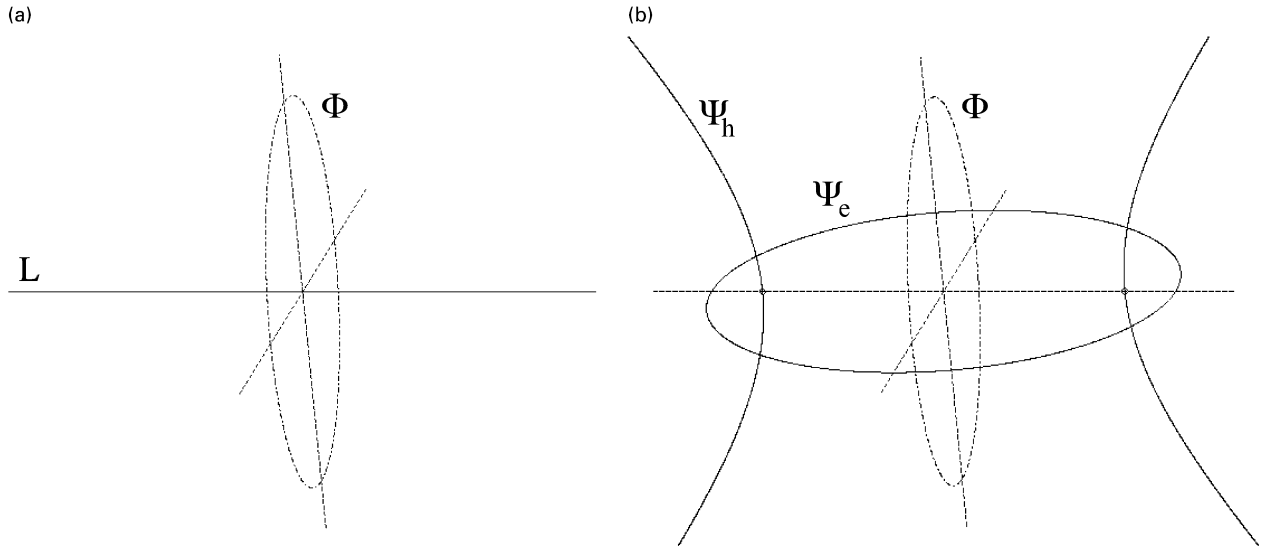


Fig. 1. Locus of camera positions in a motion sequence critical with respect to a conic Φ not on the ideal plane. The conic Φ is shown in dotted style to illustrate that it is virtual and can in fact not be drawn. The figures are further explained in the text. (a) The case of the PVC Φ being a circle. (b) The case of the PVC Φ being an ellipse.

4.2.2. Orientation

The results of the previous paragraph are necessary conditions that hold for camera *positions* in a critical motion sequence. To obtain sufficient conditions, we now consider the *orientation* of cameras. First note that rotations about the optical axis are not important here: if the projection of Φ is a circle centered in the origin (the principal point) then any rotation about the optical axis will preserve this property. Hence, the only part of camera orientation that matters is the *direction of the optical axis*.

For the camera positions derived in the previous paragraph, we have to determine the directions of the optical axis for which Φ is projected onto a circle centered in the origin. The proofs of the following statements are given in Appendix A.

If Φ is a virtual circle (cf. Fig. 1(a)) then the optical axes of all views have to coincide with L for the camera configuration to be critical. The only exception is that at two camera positions on L the optical axis might be oriented arbitrarily. These positions are symmetric with respect to the supporting plane of Φ; their distance d from that plane is related to the radius r of Φ by $d = \mathrm{Im}(r)$ (the imaginary part of the radius, which is complex due to Φ being a virtual circle). For more details, see Section B.1.

To summarize, critical motion sequences with respect to a *circle* Φ consist of collinear optical centers and optical axes passing through all optical centers except that at two positions the optical axis may be oriented arbitrarily (several views might be taken from these two positions by a camera rotating about its optical center).

If Φ is a virtual ellipse (cf. Fig. 1(b)) and the optical center lies on the ellipse Ψ_e (respectively the hyperbola Ψ_h) then the optical axis has to be the tangent of Ψ_e (respectively Ψ_h) at the optical center in order for the camera configuration to be critical.

5. Summary of critical motion sequences for a moving camera

The following camera positions/orientations constitute critical motion sequences for Euclidean reconstruction:

Case 1: Arbitrary position of optical centers but parallel optical axes. This means that camera motions are pure translations possibly combined with an arbitrary rotation about the optical axis and a reversal of the gaze direction.

Case 2: Collinear optical centers. The optical axes at two positions may be oriented arbitrarily, all others coincide with the line joining the optical centers. This means that camera motions are pure forward translations with two exceptions where the translation may be followed by an arbitrary rotation about the optical center.

Case 3: The optical centers lie on an ellipse/hyperbola pair as shown in Fig. 1(b). At each position, the optical axis is tangent to the ellipse/hyperbola. A necessary condition derived from this is: the views may be partitioned into at most two sets for which the centers and optical axes are all coplanar. In addition, these two sets define planes which are perpendicular to each other.

If we consider only one of the two conics Ψ_e or Ψ_h we

can describe the critical motion sequences as follows: a camera moving on a trajectory that may be described as (arc of) a conic and always gazing in the direction of motion, i.e. the current tangent direction. An example might be a camera mounted on a vehicle gazing in driving direction.

Newsam et al. derived degenerate configurations for two-view self-calibration [12]. Their results are of course contained in the earlier list.

Our results were reported in Ref. [19]. Kahl considered the problem in Ref. [10] but his results are not complete. As for case 2, he only obtains two subcases:

- two optical centers and arbitrary orientation;
- collinear optical centers and all optical axes aligned with the optical centers.

Whereas the 'union' of these cases (case 2 described earlier) also describes critical motion sequences.

5.1. Degree of ambiguity

We now discuss the resulting degree of ambiguity in the Euclidean reconstruction or the ego-motion estimation of the camera for the earlier cases of critical motion sequences.

5.1.1. Case 1

We only consider the case of unaligned optical centers. Aligned optical centers are discussed in Section 5.1.2.

All potential absolute conics lie on the ideal plane. Let $(Q^{\mathrm{T}}, 0)^{\mathrm{T}}$ be the ideal point of the optical axes in the critical motion sequence. It can be shown that the potential absolute conics form exactly the following 1-degree-of-freedom family:

$$\Phi(\lambda) \sim I + \lambda Q Q^{\mathrm{T}}$$

i.e. the family of conics spanned by the absolute conic (represented by the identity matrix I) and the degenerate conic QQ^{T}.

Since all potential absolute conics lie on the ideal plane the ideal plane can be recovered uniquely, which means that affine reconstruction is possible [18,10]. This implies, e.g. that relative camera displacements in the gazing direction can be estimated. However, the 1-dof-ambiguity for Euclidean reconstruction does not allow measuring angles correctly. For example, the direction of translation between different viewing positions (with respect to, e.g. the gazing direction) cannot be determined or analogously the direction of a detected obstacle's location.

5.1.2. Case 2

Different subcases have to be discussed. First, if all the optical axes are aligned (i.e. are identical with the line L shown in Fig. 1(a)) then the motion sequence is also critical according to case 1. There is a 2-dof-family of potential absolute conics: the conics described in Section 5.1.1 and all circles whose centers lie on L and whose supporting planes are orthogonal to L. Compared to case 1, affine reconstruction is not possible here. This would usually cause a wrong estimation of relative displacements, e.g. the time of impact with respect to an obstacle might be wrongly estimated.

If all the optical axes are aligned with one exception (i.e. at one viewing position t along the line L the camera gazes at other directions than along L) then there remain a 1-dof-family of potential absolute conics. These are the (true) absolute conic and one virtual circle per plane that is orthogonal to L (the circles' centers lie on L). These conics are the intersections of the planes orthogonal to L and the isotropic cone with vertex t. Affine reconstruction is not possible and as before relative displacements and angles cannot be estimated correctly.

The third subcase occurs when the camera gazes in other directions than along L at exactly two viewing positions. Only two potential absolute conics remain, which means that there are only two different solutions for Euclidean reconstruction. The potential absolute conics are the two intersections of the isotropic cones with vertices at the two exceptional viewing positions (one intersection is the true absolute conic of course, the second one lies on the equidistance plane of the two viewing positions). The wrong solution can often be ruled out in practice by imposing that the reconstructed scene lies in front of all the views.

5.1.3. Case 3

This case is difficult to explain when the motion sequence comprises three or fewer viewing positions (please contact the author for results). For more than three viewing positions, however, it is easy to see that only two potential absolute conics are possible: the true absolute conic and the virtual ellipse shown in Fig. 1(b). Thus, although case 3 (motion on conic arcs) seems to be relevant in practice the existence of only two solutions for Euclidean reconstruction is reassuring. As noted by Pollefeys the wrong solution can often be ruled out because it would lead to unrealistic estimates of the focal length [14].

6. Comments

In this section, we discuss a few special cases.

6.1. Two cameras in general position

From case 2 in the Section 5.1.2, it follows that Euclidean reconstruction is always degenerate from only two views independently of their position and orientation. In fact, it is known that in this case there is a two fold ambiguity for the absolute conic: let ω_1 and ω_2 be the projections of the absolute conic in the two views. The projection cones of ω_1 and ω_2 intersect of course in the absolute conic but in general also in a second conic Φ hence the ambiguity. However, self-calibration can in general be achieved since

Φ has the same projections as the absolute conic (it lies on the same projection cones). This illustrates that Euclidean reconstruction and self-calibration are not exactly equivalent problems. As mentioned before the ambiguous solution for Euclidean reconstruction can often be ruled out in practice by imposing that the reconstructed scene lies in front of both cameras.

6.2. Camera rotating about its optical center

A camera rotating about its optical center while possibly changing its focal length can always be calibrated from two views whose optical axes do not coincide. This is briefly explained in the following. The two views are critical for Euclidean reconstruction according to case 2 (cases 1 and 3 allow only one optical axis per camera position in a critical motion sequence). The position t of the two views is one of the two exceptional points. The cone defined by Φ and t contains the absolute conic (it is an isotropic cone, cf. Section B.1). Hence, the projections of Φ are identical with the image of the absolute conic, which means that (self-) calibration has a unique solution.

6.3. Fixation

Consider cameras fixating a finite point (i.e. the point lies on the optical axes of the cameras). For two cameras, the configuration is always critical whereas for more than two cameras (with different optical centers) it is always non critical.

The first statement is easy to understand. There is a one-dimensional family of possibilities to realize case 3, i.e. a one-dimensional family of 'motion conics' Ψ: the conic has to contain the optical centers and has the two optical axes as tangents. This gives four constraints on the five degrees of freedom for a conic which means that there remains one degree of freedom for Ψ and thus for the potential absolute conic Φ.

Consider now an additional camera that fixates the same finite point as the two others. Suppose the configuration were still critical. From Section A.2.1 we know that optical axes for optical centers on Ψ_e and Ψ_h are mutually skew hence it follows that the three optical centers have to lie on either Ψ_e or Ψ_h and the optical axes are tangents to the same conic. This would imply that the fixated point lies on three different tangents to the same conic, which is not possible [16]. Hence, the assumption that the configuration is critical, is contradicted.

7. Derivation of the critical motion sequences for stereo systems

The results presented in the previous sections allow us to study degeneracies of Euclidean reconstruction for stereo systems. Here, a stereo system consists of two cameras with coplanar optical axes and symmetric but possibly variable vergence angles α (cf. Fig. 2). The distance between the two cameras is fixed. The focal lengths of the two cameras are not constrained to be equal and they may vary freely between different images. We define the *axis of a stereo system* as the line perpendicular to the baseline and passing through its midpoint.

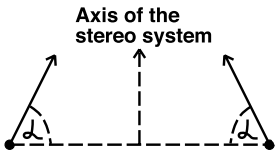


Fig. 2. The type of stereo system discussed in this paper.

A single pair of images taken by such a stereo system is of course critical (cf. Section 6.3). In the following, we reveal the conditions for two stereo pairs to be critical. They are critical if the set of individual views constitute a critical motion sequence as described in the previous sections. We consider several cases:

- coplanar stereo pairs with identical vergence angles;
- coplanar stereo pairs with variable vergence angles;
- non coplanar stereo pairs.

By coplanar stereo pairs we mean that all the optical centers and optical axes are located in the same plane.

Before examining the different cases, we give an introductory remark concerning case 3 of Section 5.

7.1. Concerning case 3

For a given stereo system, we want to establish constraints on the possible locations of 'motion conics' Ψ. Since the optical axes of the two cameras in the stereo system are coplanar the two optical centers have to lie on either the ellipse Ψ_e or the hyperbola Ψ_h but cannot be distributed on both (cf. Section A.2.1). It is now easy to show that due to symmetry of the vergence angles, Ψ must be symmetric with respect to the axis of the stereo system. Since Ψ cannot be a circle (cf. Section A.2.1) it has exactly two symmetry lines (which are perpendicular to each other). This means that the axis of the stereo system coincides with one of these two symmetry lines of Ψ.

7.2. Coplanar stereo pairs with identical vergence angles

For the trivial case of a stationary stereo system any number of stereo pairs are critical. Non-trivial cases are discussed in the following.

7.2.1. Parallel optical axes

If the vergence angles are of 90°, i.e. if the two optical axes are parallel then each stereo pair is critical according to cases 1 and 3. According to case 1 the combination of two stereo pairs is only critical if the stereo system undergoes a pure translation or a rotation by 180° in the plane of motion,

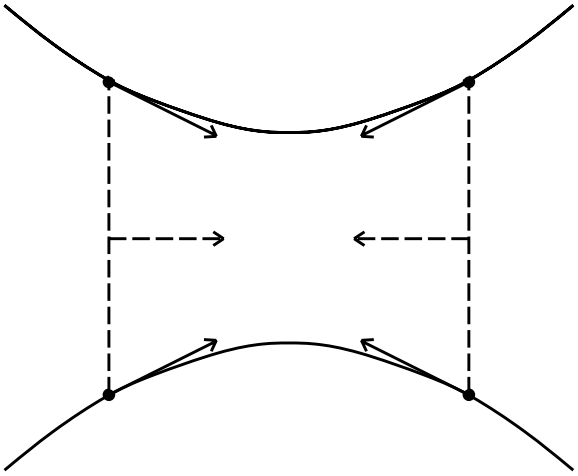


Fig. 3. Two stereo systems with opposite gazing direction but identical axes constitute a critical motion sequence according to case 3. The hyperbola shown is the 'motion conic' Ψ.

i.e. a reversal of gaze direction. Case 3 is dealt with as in the following paragraph.

7.2.2. Convergent (non parallel) optical axes

The stereo pairs can only be critical according to case 3 and this only if there is an ellipse or a hyperbola Ψ containing all four optical centers and having all optical axes as tangents. From Section 7.1, we conclude that the axes of the two stereo pairs do either coincide or are perpendicular to each other. It can be shown that in case they are perpendicular, there is no possibility to place the two stereo pairs in a way that a conic Ψ as described earlier exists (since the distance between the two cameras and the vergence angles are fixed). In case the axes coincide a conic Ψ exists exactly if the two stereo pairs have opposite gaze direction as shown in Fig. 3.

7.2.3. Summary

Two coplanar stereo pairs with identical vergence angles are critical in exactly the following situations. If the optical axes of the stereo system are parallel then the stereo pairs are critical if they are related by a pure translation possibly followed by a reversal of gaze direction. If the optical axes are convergent then the stereo pairs are only critical if they gaze in opposite directions and if their axes are identical. The only case of practical importance is pure translation of a stereo system with parallel optical axes.

7.3. Coplanar stereo pairs with variable vergence angles

Due to varying vergence angles at least one stereo pair has convergent optical axes. Hence, the combination of the stereo pairs can only be critical according to case 3. As stated in Section 7.1, the conic Ψ must be symmetric with respect to the axes of the stereo pairs, which implies that these are either identical or perpendicular to each other. It can be shown that if they are identical no conic Ψ as described earlier can exist. For perpendicular axes, there is a one-dimensional family of possibilities for placing the stereo pairs and setting their vergence angles relative to each other such that the combination of the stereo pairs is critical.

In summary, if the vergence angles for the two stereo

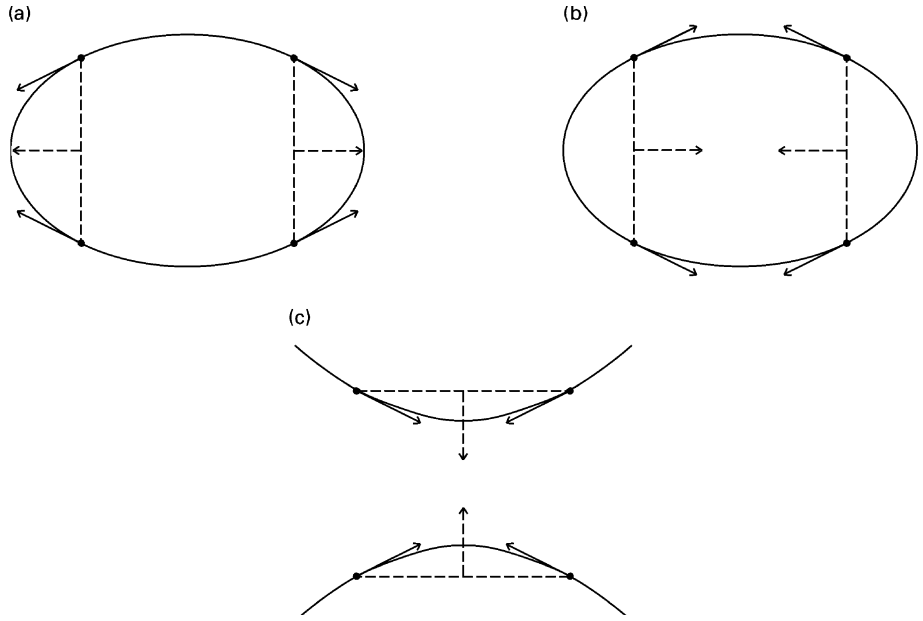


Fig. 4. Possible arrangements of stereo systems and 'motion conics' (described in the text). (a) Possible arrangement of stereo systems and motion ellipse Ψ_e, as described by case 3 in Section 5. For vergence angles $\alpha < 90°$ the stereo system gazes away from the ellipse's center. (b) Only for vergence angles $\alpha > 90°$ the stereo system gazes towards the ellipse's center. (c) Possible arrangement of stereo systems and motion hyperbola Ψ_h.

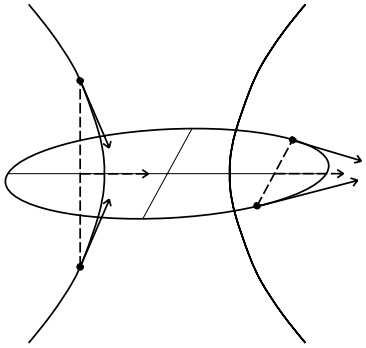


Fig. 5. One possible non coplanar critical motion sequence for stereo systems.

pairs are different the configuration can only be critical if the axes of the stereo pairs are perpendicular to each other. The exact conditions for being critical are complicated and omitted here since they do not really contribute to the understanding of this discussion.

7.4. Non coplanar stereo pairs

Non coplanar stereo pairs can only be critical according to cases 1 or 3. Case 1 is only possible if the optical axes are all parallel (vergence angle $\alpha = 90°$). Critical motions are exactly pure translations possibly followed by rotations about axes parallel to the optical axes or by reversals of the gazing direction.

As for case 3, the non coplanarity implies that one stereo pair is located on the ellipse Ψ_e and the other on the hyperbola Ψ_h. We assume that practical vergence angles are inferior (or equal to) 90°. This means that the axis of the stereo system located on the ellipse Ψ_e is directed away from the ellipse's center as shown in Fig. 4(a), i.e. a case as in Fig. 4(b) is not possible. As for the hyperbola the contrary is valid, i.e. the stereo system's axis will be directed towards the hyperbola's center (cf. Figs. 3 and 4(c)).

By comparing this discussion with Fig. 1(b), it is clear that for the only cases relevant in practice the axes of the two stereo pairs must be identical as shown in Fig. 5 (for the other potential arrangements, one stereo pair would fixate a point that is behind the other stereo pair, i.e. the common field of view would be either empty or very small making image matching and thus self-calibration inherently impossible). The relative position of the stereo pairs is thus as follows. Their axes coincide and their 'supporting planes' are orthogonal to each other.

In conclusion, the only critical situations that might be encountered with non coplanar stereo systems are:

- a stereo system with parallel optical axes undergoing pure translations (possibly in several different directions), possibly followed by rotations about axes parallel to the optical axes and reversals of the gazing direction.
- a stereo system with parallel or convergent optical axes that is rotated by 90° about its axis followed by a translation along its axis and possibly a reversal of gazing direction.

8. Conclusions

We have derived all motion sequences that are critical for Euclidean reconstruction from image sequences with variable and unknown focal length whose other intrinsic parameters are known. The critical motion sequences are described geometrically. Our results are rather encouraging since only few cases exist that are likely to be met in practice. The most important cases are pure translation and especially pure forward motion. One also has to be aware of the fact that motion on a conic arc while gazing in the direction of motion is critical although in general there are only two ambiguous solutions for Euclidean reconstruction and self-calibration. Another important result is that an image sequence taken by a camera that fixates a finite point is always critical when two views only are used but never critical with three or more views.

Both pure translation and motion on a conic are also critical for self-calibration when all intrinsics are constant but unknown [18]. In that case, however, the degree of ambiguity in the solution is higher. Moreover, general planar motions are always critical, which is not the case for the situation dealt with in this paper.

Our results allowed us to study the critical motions of stereo systems. We did this for various situations, stereo systems undergoing planar or non planar motion and having fixed or variable vergence angles. Our results show that it should be rather easy to avoid critical motions in practice: it suffices to guarantee that the axes of the different stereo pairs in a sequence are neither parallel nor perpendicular to each other.

Acknowledgements

This work was done while the author stayed with the Computational Vision Group of the University of Reading and was supported by the EPSRC funded project GR/K89221 (Vector).

Appendix A. Locus of vertices of circular cones containing a conic section

Consider a proper virtual conic Φ. We want to determine the locus of all real points C such that the cones formed by Φ and with C as vertex are circular. Without loss of generality, we can choose simple coordinates for the problem as follows: let the supporting plane of Φ be the plane $Z = 0$ and let the conic be centered in the origin and with axes aligned with the X and Y axes. Hence, the conic's matrix is

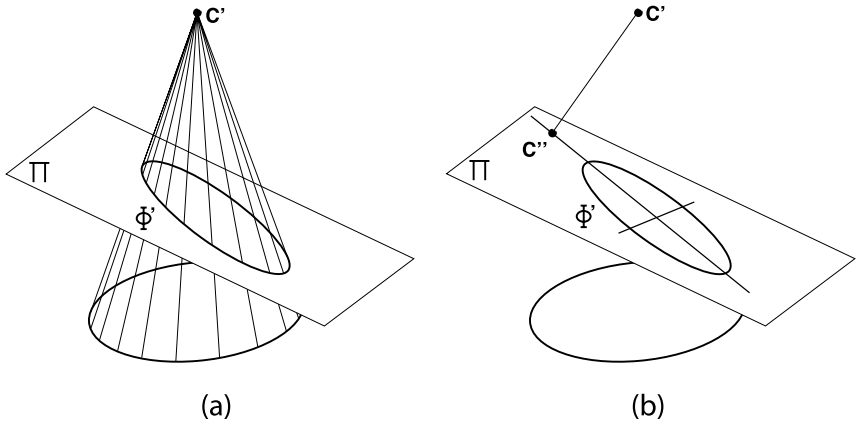


Fig. A1. For any circular cone and any plane, the orthogonal projection of the cone's vertex on that plane lies on one of the two symmetry lines of the conic section induced by the cone and the plane (see text for more details).

of the form:

$$\Phi \sim \begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & -1 \end{pmatrix}$$

Since Φ is a proper virtual conic we have $a, b < 0$.

Let now $C = (X, Y, Z, 1)^{\mathrm{T}}$ be the vertex of a cone Λ that contains Φ (with $Z \neq 0$). The cone's matrix is given by:

$$\Lambda \sim \begin{pmatrix} a & 0 & -a\frac{X}{Z} & 0 \\ 0 & b & -b\frac{Y}{Z} & 0 \\ -a\frac{X}{Z} & -b\frac{Y}{Z} & \frac{aX^2 + bY^2 - 1}{Z^2} & \frac{1}{Z} \\ 0 & 0 & \frac{1}{Z} & -1 \end{pmatrix}$$

We want to establish C for which the cone Λ is circular. Cones with finite vertex are circular exactly if the conic obtained by intersection with the ideal plane has a double eigenvalue [2]. This condition is explored for the two different cases of Φ being a virtual circle or an ellipse (there are no other cases for PVC [2]).

A.1. Φ is a virtual circle

This case occurs when $a = b$. For Λ to be a circular cone, its vertex C must lie on the line passing through the center of Φ and being orthogonal to its supporting plane. Here, this is the Z-axis.

A.2. Φ is a virtual ellipse

This case occurs when $a \neq b$.

First, we show that C must lie in one of the symmetry planes of Φ, which are defined as follows. A symmetry plane of an ellipse is a plane orthogonal to the ellipse's supporting plane, which contains one of the ellipse's two symmetry lines. Besides the supporting plane itself the two symmetry planes are the only planes which conserve Φ by reflection.

Consider a circular virtual cone with vertex C'. Let Π be a plane with $C' \notin \Pi$ and let Φ' be the conic cut out from the cone by Φ (Fig. A1(a)). Let C'' be the orthogonal projection of the cone's vertex C' on Φ. It is easy to show that C'' lies on one of the two symmetry lines of Φ' (Fig. A1(b)). Since C'' is the orthogonal projection of C' on Π the plane spanned by C' and that symmetry line is a symmetry plane of Φ'. We conclude that for all conic sections of a circular cone the cones vertex lies on an associated symmetry plane.

In our case, the symmetry planes of Φ are the planes $X = 0$ and $Y = 0$. From the earlier discussion it follows that the vertex C of the circular cone Λ must lie in one of these planes. In the following, we examine the case $X = 0$ (the other case can be treated in an analogous way). The cones matrix is thus simplified to:

$$\Lambda \sim \begin{pmatrix} a & 0 & 0 & 0 \\ 0 & b & -b\frac{Y}{Z} & 0 \\ 0 & -b\frac{Y}{Z} & \frac{bY^2 - 1}{Z^2} & \frac{1}{Z} \\ 0 & 0 & \frac{1}{Z} & -1 \end{pmatrix}$$

As stated earlier, the cone is circular exactly if the conic obtained by intersection with the ideal plane has a double eigenvalue. This conic Φ_∞ is given by the upper left 3×3

submatrix of Λ:

$$\Phi_\infty \sim \begin{pmatrix} a & 0 & 0 \\ 0 & b & -b\frac{Y}{Z} \\ 0 & -b\frac{Y}{Z} & \frac{bY^2-1}{Z^2} \end{pmatrix}$$

The three eigenvalues of Φ_∞ are:

$$a \text{ and } \frac{b(Y^2+Z^2)-1 \pm \sqrt{b^2(Y^2+Z^2)^2+2b(Z^2-Y^2)+1}}{2Z^2}$$

Equating the second and third eigenvalues leads to subcases of equating the first eigenvalue a with the second or third eigenvalue. Equating the first eigenvalue a with the second or third eigenvalue leads to the following constraint on Y and Z (after some manipulations using MAPLE):

$$abY^2 + (b-a)(aZ^2+1) = 0 \quad \text{(A1)}$$

Eq. (A1) can be represented by the following matrix equation:

$$(Y,Z,1)\begin{pmatrix} ab & 0 & 0 \\ 0 & a(b-a) & 0 \\ 0 & 0 & b-a \end{pmatrix}\begin{pmatrix} Y \\ Z \\ 1 \end{pmatrix}$$

The matrix

$$\Psi \sim \begin{pmatrix} ab & 0 & 0 \\ 0 & a(b-a) & 0 \\ 0 & 0 & b-a \end{pmatrix}$$

represents a conic in the plane $X = 0$ whose type is:

- a real ellipse, if $b < a$. Note that Ψ cannot be a circle (this would be the case if $\Psi_{11} = \Psi_{22}$ which is equivalent to $a = b$ in contradiction to the assumption that $a \neq b$).
- a hyperbola, if $a < b$.

If we consider the case $Y = 0$, we obtain just the reciproque result.

Hence the locus of vertices of circular cones containing a virtual ellipse Φ is the union of a real ellipse Ψ_e and a real hyperbola Ψ_h (cf. Fig. 1(b)). The following relations hold between Φ, Ψ_e and Ψ_h. Their supporting planes are mutually orthogonal and each of them is a symmetry plane for the two other conics. The hyperbola Ψ_h passes through the focii of Ψ_e.

A.2.1. Further observations

We note that Ψ cannot be a circle (property used in Section 7). It would be a circle if (cf. Eq. (A2)) $\Psi_{11} = \Psi_{22}$, i.e. if $ab = a(b-a)$, hence if $a = 0$. This is in contradiction with the fact that a is an eigenvalue of the *proper* virtual conic Φ and thus non zero.

Another property that is used in Section 6 is easy to show. Namely, all real tangents of Ψ_e and of Ψ_h are mutually skew, i.e. there is no tangent of Ψ_e that has a real intersection point with any tangent of Ψ_h.

Appendix B. Axis of circular cones containing a conic section

In the following we prove the statements made in Section 4.2.2. Remember that the optical axis is the axis of the circular cone Λ (cf. Section 4.2.1). As in Appendix A we consider the two cases of Φ being a virtual circle or a virtual ellipse. We use the same simple coordinates as in Appendix A.

B.1. Φ is a virtual circle

We already know that the vertex of a circular cone Λ containing Φ is a point with coordinates $(0,0,Z,1)^T$, i.e. Λ is given by:

$$\Lambda \sim \begin{pmatrix} a & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & -\frac{1}{Z^2} & \frac{1}{Z} \\ 0 & 0 & \frac{1}{Z} & -1 \end{pmatrix}$$

The intersection of Λ with the ideal plane is a conic given by:

$$\Phi_\infty \sim \begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & -\frac{1}{Z^2} \end{pmatrix}$$

For $Z^2 \neq -1/a$, Φ_∞ has a double and a single eigenvalue, i.e. Λ is a circular cone with the Z-axis as axis. However, for $Z = \pm 1/\sqrt{-a}$ (these are real numbers since $a < 0$), Φ_∞ is the absolute conic (given by the identity matrix) which means that Λ is an isotropic cone. Isotropic cones are invariant to rotation about their vertex. Hence, the projection of an isotropic cone by a camera located at its vertex is always a centered virtual circle, regardless of the camera's orientation.

B.2. Φ is a virtual ellipse

It is easy to show that in this case, Λ cannot be an isotropic cone, i.e. at each possible camera position, there is only one possible direction for the optical axis (given by the cone's axis), such that the camera configuration is critical. In the following we prove that the axis of Λ is the tangent line to Ψ at C that lies in the supporting plane of Ψ (notation as in Appendix A).

Another definition of the axis of a circular cone as that given in Section 4.2.1, is as follows: exactly planes orthogonal to the axis cut the cone in circles, i.e. conics containing circular points.

The tangent line to Ψ at C is given by (coordinates in the plane $X = 0$):

$$T \sim \Psi \begin{pmatrix} Y \\ Z \\ 1 \end{pmatrix} \sim \begin{pmatrix} abY \\ a(b-a)Z \\ b-a \end{pmatrix}$$

The planes orthogonal to the line T are given by:

$$\Pi \sim \begin{pmatrix} 0 \\ a(a-b)Z \\ abY \\ d \end{pmatrix}$$

for real d. The two ideal intersection points of Π and Λ are:

$$Q \sim \begin{pmatrix} \pm\sqrt{a(a^2+b^2-2ab-a^2bY^2)} \\ abY \\ a(b-a)Z \\ 0 \end{pmatrix}$$

They are circular points since:

$$Q_1^2 + Q_2^2 + Q_3^2 = a(b-a)(abY^2 + (b-a)(aZ^2+1))$$

which is, cf. Eq. (A1), equal to 0.

References

[1] A. Azarbayejani, A.P. Pentland, Recursive estimation of motion, structure and focal length, IEEE Transactions on Pattern Analysis and Machine Intelligence 17 (1995) 562–575.

[2] W. Boehm, H. Prautzsch, Geometric concepts for geometric design (A.K. Peters, 1994).

[3] S. Bougnoux, From projective to Euclidean space under any practical situation, a criticism of self-calibration, Proceedings of the International Conference on Computer Vision, Bombay, India, 1998, pp. 790–796.

[4] M.J. Brooks, L. de Agapito, D.Q. Huynh, L. Baumela, Direct methods for self-calibration of a moving stereo head. Proceedings of the European Conference on Computer Vision, Cambridge, 1996, pp. 415–426.

[5] L. de Agapito, D.Q. Huynh, M.J. Brooks, Self-calibrating a stereo head: an error analysis in the neighbourhood of degenerate configurations, Proceedings of the International Conference on Computer Vision, Bombay, India, 1998, pp. 747–753.

[6] O. Faugeras, What can be seen in three dimensions with an uncalibrated stereo rig? Proceedings of the European Conference on Computer Vision, Santa Margherita Ligure, Italy, 1992, pp. 563–578.

[7] O. Faugeras, Stratification of three-dimensional vision: projective, affine and metric representations, Journal of the Optical Society of America A 12 (1995) 465–484.

[8] R.I. Hartley, Estimation of relative camera positions for uncalibrated cameras, Proceedings of the European Conference on Computer Vision, Santa Margherita Ligure, Italy, 1992, pp. 579–587.

[9] R. Hartley, R. Gupta, T. Chang, Stereo from uncalibrated cameras, Proceedings of the European Conference on Computer Vision, Santa Margherita Ligure, Italy, 1992, pp. 761–764.

[10] F. Kahl, Critical motions and ambiguous Euclidean reconstructions in auto-calibration, Proceedings of the International Conference on Computer Vision, Kerkyra, Greece, 1999, pp. 469–475.

[11] S.J. Maybank, O.D. Faugeras, A theory of self calibration of a moving camera, International Journal on Computer Vision 8 (1992) 123–151.

[12] G.N. Newsam, D.Q. Huynh, M.J. Brooks, H.P. Pan, Recovering unknown focal lengths in self-calibration: an essentially linear algorithm and degenerate configurations. Part B3 of the proceedings of the XVIII ISPRS-Congress, Vienna, Austria, 1996, pp. 575–580.

[13] M. Pollefeys, R. Koch, L. Van Gool, Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters, Proceedings of the International Conference on Computer Vision, Bombay, India, 1998, pp. 90–95.

[14] M. Pollefeys, Self-calibration and metric 3D reconstruction from uncalibrated image sequences, PhD Thesis, Department ESAT, Katholieke Universiteit Leuven, Belgium, 1999.

[15] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, Numerical recipes in C, Cambridge University Press, Cambridge, New York, 1992.

[16] J.G. Semple, G.T. Kneebone, Algebraic Projective Geometry, Oxford Science Publication, 1952.

[17] P. Sturm, Critical motion sequences for monocular self-calibration and uncalibrated Euclidean reconstruction, Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, Puerto Rico, 1997, pp. 1100–1105.

[18] P. Sturm, Vision 3D non calibrée: contributions à la reconstruction projective et étude des mouvements critiques pour l'auto-calibrage, PhD Thesis, INPG, France, 1997.

[19] P. Sturm, Critical motion sequences for the self-calibration of cameras and stereo systems with variable focal length. Proceedings of the British Machine Vision Conference, Nottingham, 1999, pp. 63–72.

Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computer Vision and Image Understanding
ELSEVIER
Computer Vision and Image Understanding 99 (2005) 58–95
www.elsevier.com/locate/cviu

Focal length calibration from two views: method and analysis of singular cases

P. Sturm[a], Z.L. Cheng[b,*,1], P.C.Y. Chen[c,*], A.N. Poo[b]

[a] *INRIA Rhône-Alpes, 38330 Montbonnot, St. Martin, France*
[b] *Mechanical Engineering Department, National University of Singapore, 119260 Singapore, Singapore*
[c] *Bachelor of Technology Programme, Faculty of Engineering, National University of Singapore, 119260 Singapore, Singapore*

Received 20 October 2003; accepted 1 November 2004
Available online 18 December 2004

Abstract

We consider the problem of estimating the focal length of a camera from two views while the focal length is not varied during the motion of the camera. An approach based on Kruppa's equations is proposed. Specifically, we derive two linear and one quadratic equations to solve the problem. Although the three equations are interdependent in general, each one may be singular for different configurations. We study in detail the generic singularities of the problem and the actual singularities of the individual calibration equations. Results of our experiments using synthetic and real data underline the effect that singular configurations may have on self-calibration. However, these results are stable once the singularities are avoided.

Keywords: Camera calibration; Kruppa's equations; 3D reconstruction

* Corresponding authors. Fax: +65 6777 3525 (P.C.Y. Chen).
E-mail addresses: chengz@rpi.edu (Z.L. Cheng), engchenp@nus.edu.sg (P.C.Y. Chen).
[1] Present address: 23, 13th Street, 3rd floor, Troy, NY 12180, USA.

1077-3142/$ - see front matter © 2004 Elsevier Inc. All rights reserved.
doi:10.1016/j.cviu.2004.11.002

1. Introduction

Camera self-calibration has been studied for various scenarios. In the original scenario [3], the case of a camera with constant but completely unknown intrinsic parameters is considered. Since then, this has been extended to cases where all but one of the intrinsic parameters may be varying [16,12]. Reports on recent advances and general overviews of the topic can be found in [11,4].

In parallel to the proposition of new algorithms, research has been conducted on "critical motions," where camera configurations or trajectories will render self-calibration impossible in theory and unstable in practice, see e.g. [1,13,14,18,19,21,23].

In this paper, we consider what may be the simplest self-calibration scenario: two views of an unknown static scene are taken by a camera with constant parameters, with the assumption that all intrinsic parameters except the focal length are known. Although very simple, we believe that this is a very useful scenario in practice. It has been shown that it is even possible to calibrate a *varying* focal length from two views [6]. Simple algorithms for this purpose were proposed in [1,2,15,16]. One of the drawbacks of this scenario is that the problem is unsolvable whenever the optical axes of the two views are coplanar [14,15,21], which is always approximately the case for stereo systems. Other less likely critical configurations are also described in [14,15,21].

In this paper, we show that the assumption of a *constant* focal length reduces the number of critical configurations. The generic critical configurations (which we will also refer to as *singularities* or *degeneracies*) of the problem are given: the problem is unsolvable whenever the optical axes of the two views are parallel or if they intersect at a finite point equidistant from both optical centers.

We show that two linear and one quadratic equations can be derived from the singular value decomposition (SVD) of the fundamental matrix. All critical configurations for the individual equations are then revealed in detail. Especially, it is shown that the quadratic equation degenerates only in the generic cases, or in some cases when the focal length is equal to ± 1, whereas the linear equations' critical configurations are the same as for the above problem of estimating a *varying* focal length.

We believe that such a study of critical configurations is important, since it indicates which configurations to avoid in general, and explains why certain algorithms may still fail (see e.g., a study on Kruppa equations [19]).

The performance of the calibration equations is evaluated using synthetic and real data. In both cases, we are interested in investigating the camera setups close to critical configurations. As for the real images, we show that, when the critical configurations are avoided, the results are of acceptable accuracy and stability.

This paper is an extended version of [20], and contains more experimental results and a more in-depth theoretical study.

Organization. The problem is formulated in Section 2 and the calibration equations are derived in Section 3. Generic and equation-specific singularities are summarized in Sections 4 and 5. Experimental results are provided in Section 6 and the paper is concluded in Section 7. The appendices contain all proofs for the equation-specific singularities, organized in several sections in a logical sequence.

Notations. In this paper, matrices are represented in sans serif font (e.g., K), vectors in bold face (e.g., $\mathbf{q}$), and scalars in italics. Coefficients of a matrix U (respectively, a vector $\mathbf{v}$) are denoted by U_{ij} (respectively, v_i). Equality of matrices or vectors, up to scale, is denoted by $\sim$. For any vector $\mathbf{v}$, $[\mathbf{v}]_\times$ represents the skew-symmetric matrix associated with the cross product, i.e., $\mathbf{v} \times \mathbf{w} = [\mathbf{v}]_\times \mathbf{w}$. Transposition of a vector $\mathbf{v}$ is denoted as $\mathbf{v}^T$, and the inverse of the transpose of a matrix A as A^{-T}. In complex equations, we often use the shorthand notations $c_\alpha = \cos\alpha$, $s_\alpha = \sin\alpha$, and $t_\alpha = \tan\alpha$.

2. Problem formulation

Throughout this paper, we use perspective projection as the camera model, with the following intrinsic parameters: the focal length f, the aspect ratio τ, and the principal point (u_0, v_0). A 3D point $\mathbf{Q}$ is projected to an image point $\mathbf{q}$ via

$$\mathbf{q} \sim \mathsf{P}\mathbf{Q} \sim \mathsf{K}\mathsf{R}(\mathtt{I} - \mathbf{t})\mathbf{Q},$$

where the rotation matrix R and the vector $\mathbf{t}$ represent the camera's orientation and position, respectively. The calibration matrix K is defined as

$$\mathsf{K} = \begin{pmatrix} \tau f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{pmatrix}.$$

In the following, assume that two images of a static scene are available and that a projective reconstruction is possible or, equivalently, that the fundamental matrix can be computed. Without loss of generality, assume that the first camera is located at the origin and that its rotation matrix is the identity matrix. With R and $\mathbf{t}$ being the extrinsic and K' the intrinsic parameters of the second camera, the fundamental matrix of two images is given by [11]

$$\mathsf{F} \sim \mathsf{K}'^{-T}\mathsf{R}[\mathbf{t}]_\times \mathsf{K}^{-1}.$$

We assume that the aspect ratio and the principal point are known for both images and that their focal lengths are identical. We can thus move from a completely uncalibrated space to a "semi-calibrated" one, by computing an intermediate between the fundamental matrix and the essential matrix ($\mathsf{R}[\mathbf{t}]_\times$ in the above equation)

$$\mathsf{G} \sim \begin{pmatrix} \tau' & 0 & 0 \\ 0 & 1 & 0 \\ u'_0 & v'_0 & 1 \end{pmatrix} \mathsf{F} \begin{pmatrix} \tau & 0 & u_0 \\ 0 & 1 & v_0 \\ 0 & 0 & 1 \end{pmatrix} \sim \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & f \end{pmatrix} \mathsf{R}[\mathbf{t}]_\times \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & f \end{pmatrix}. \qquad (1)$$

We call G the *semi-calibrated fundamental matrix*.

3. Calibration equations

Let the singular value decomposition [5] of G be given by

$$\mathsf{G} = \mathsf{U}\Sigma\mathsf{V}^{\mathrm{T}},$$

with $\Sigma = \mathrm{diag}(a, b, 0)$ being the diagonal matrix of singular values ($a, b > 0$) and U and V orthogonal matrices. We denote by $\mathbf{u}_i$ and $\mathbf{v}_j$ the ith and jth column of U and V, respectively. Note that the second epipole $\mathbf{e}'$ of G is its left null space, i.e., $\mathbf{e}' \sim \mathbf{u}_3$. It can be shown [9,23] that Kruppa's equations can be reinterpreted by the following relationship in terms of fundamental matrix and the epipole:

$$\mathsf{G}\begin{pmatrix} f^2 & 0 & 0 \\ 0 & f^2 & 0 \\ 0 & 0 & 1 \end{pmatrix}\mathsf{G}^{\mathrm{T}} \sim [\mathbf{e}']_\times \begin{pmatrix} f^2 & 0 & 0 \\ 0 & f^2 & 0 \\ 0 & 0 & 1 \end{pmatrix}[\mathbf{e}']_\times.$$

In terms of the SVD of G, this can be written as

$$\mathsf{U}\Sigma\mathsf{V}^{\mathrm{T}}\begin{pmatrix} f^2 & 0 & 0 \\ 0 & f^2 & 0 \\ 0 & 0 & 1 \end{pmatrix}\mathsf{V}\Sigma\mathsf{U}^{\mathrm{T}} \sim [\mathbf{u}_3]_\times \begin{pmatrix} f^2 & 0 & 0 \\ 0 & f^2 & 0 \\ 0 & 0 & 1 \end{pmatrix}[\mathbf{u}_3]_\times.$$

Multiplying the equation by U^{T} from the left and U from the right gives, due to the orthogonality of U

$$\begin{aligned}\Sigma\mathsf{V}^{\mathrm{T}}\begin{pmatrix} f^2 & 0 & 0 \\ 0 & f^2 & 0 \\ 0 & 0 & 1 \end{pmatrix}\mathsf{V}\Sigma &\sim \begin{pmatrix} \mathbf{u}_1^{\mathrm{T}} \\ \mathbf{u}_2^{\mathrm{T}} \\ \mathbf{u}_3^{\mathrm{T}} \end{pmatrix}[\mathbf{u}_3]_\times \begin{pmatrix} f^2 & 0 & 0 \\ 0 & f^2 & 0 \\ 0 & 0 & 1 \end{pmatrix}[\mathbf{u}_3]_\times (\,\mathbf{u}_1 \quad \mathbf{u}_2 \quad \mathbf{u}_3\,) \\ &\sim \begin{pmatrix} \mathbf{u}_2^{\mathrm{T}} \\ -\mathbf{u}_1^{\mathrm{T}} \\ \mathbf{0}^{\mathrm{T}} \end{pmatrix}\begin{pmatrix} f^2 & 0 & 0 \\ 0 & f^2 & 0 \\ 0 & 0 & 1 \end{pmatrix}(\,\mathbf{u}_2 \quad -\mathbf{u}_1 \quad \mathbf{0}\,).\end{aligned}$$

The last row and the last column of this matrix equation are zero vectors, so we concentrate on the upper left 2×2 part of the equation

$$\begin{pmatrix} a\mathbf{v}_1^{\mathrm{T}} \\ b\mathbf{v}_2^{\mathrm{T}} \end{pmatrix}\begin{pmatrix} f^2 & 0 & 0 \\ 0 & f^2 & 0 \\ 0 & 0 & 1 \end{pmatrix}(\,a\mathbf{v}_1 \quad b\mathbf{v}_2\,) \sim \begin{pmatrix} \mathbf{u}_2^{\mathrm{T}} \\ -\mathbf{u}_1^{\mathrm{T}} \end{pmatrix}\begin{pmatrix} f^2 & 0 & 0 \\ 0 & f^2 & 0 \\ 0 & 0 & 1 \end{pmatrix}(\,\mathbf{u}_2 \quad -\mathbf{u}_1\,).$$

Making use of the fact that the vectors $\mathbf{v}_1$, etc., have unit norm, we can further simplify the above equation to obtain

$$\begin{aligned}&\begin{pmatrix} a^2(f^2 + V_{31}^2(1-f^2)) & abV_{31}V_{32}(1-f^2) \\ abV_{31}V_{32}(1-f^2) & b^2(f^2 + V_{32}^2(1-f^2)) \end{pmatrix} \\ &\quad\sim \begin{pmatrix} f^2 + U_{32}^2(1-f^2) & -U_{31}U_{32}(1-f^2) \\ -U_{31}U_{32}(1-f^2) & f^2 + U_{31}^2(1-f^2) \end{pmatrix}.\end{aligned}$$

The equality (up to scale) of these two symmetric matrices gives rise to three individual quadratic equations in f^2 (by forming the cross-product of the vectors containing the three different coefficients of each matrix). Two of these have the trivial solution[2] $f^2 = 1$. Factoring this out, we thus obtain two linear equations and a quadratic one:

$$f^2\{aU_{31}U_{32}(1-V_{31}^2)+bV_{31}V_{32}(1-U_{32}^2)\}+U_{32}V_{31}(aU_{31}V_{31}+bU_{32}V_{32})=0, \tag{2}$$

$$f^2\{aV_{31}V_{32}(1-U_{31}^2)+bU_{31}U_{32}(1-V_{32}^2)\}+U_{31}V_{32}(aU_{31}V_{31}+bU_{32}V_{32})=0, \tag{3}$$

$$\begin{aligned} &f^4\{a^2(1-U_{31}^2)(1-V_{31}^2)-b^2(1-U_{32}^2)(1-V_{32}^2)\}\\ &\quad+f^2\{a^2(U_{31}^2+V_{31}^2-2U_{31}^2V_{31}^2)-b^2(U_{32}^2+V_{32}^2-2U_{32}^2V_{32}^2)\}\\ &\quad+\{a^2U_{31}^2V_{31}^2-b^2U_{32}^2V_{32}^2\}=0. \end{aligned} \tag{4}$$

These are our self-calibration equations. They are of course algebraically dependent, but we will see in the following sections that they may be singular in different conditions.

3.1. Calibration algorithm

A simple calibration algorithm can be formulated as follows:

(1) Estimate the fundamental matrix between the two views (algorithms with good performance are given in [22]).
(2) "Undo" the known intrinsic parameters, as shown in Eq. (1).
(3) Compute the SVD of G and extract the coefficients U_{31}, U_{32}, V_{31}, and V_{32}, as well as the non-zero singular values a and b.
(4) Construct and solve any of the Eqs. (2)–(4). In practice, we only solve the quadratic equation. The spurious solution can either be ruled out using the linear equations, or usually by simply taking the solution closest to a reasonable guess (in simulations, the spurious solution was always observed to be far off the true one).
(5) Optionally, the result can be improved by bundle adjustment, after having estimated the relative pose of the cameras.

3.2. On standardization

It is often advisable to work in "standardized" image coordinates [8], which is usually achieved by translating and scaling image coordinates appropriately. The transformation applied in step (2) of the above algorithm, mainly amounts to such

[2] The case where the true squared focal length equals 1, is discussed in Appendix D; this might occur if working in standardized coordinates.

a translation, and one might also apply an additional scaling. Usually, the range of feasible focal lengths is well known, and one might apply a scaling with the inverse of a feasible focal length value f_0 (standardization based on image point coordinates as in [8] amounts usually to such a scaling). The semi-calibrated fundamental matrix would be transformed according to

$$\begin{pmatrix} f_0 & 0 & 0 \\ 0 & f_0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \mathsf{G} \begin{pmatrix} f_0 & 0 & 0 \\ 0 & f_0 & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{5}$$

The rest of the algorithm will be the same, except that the estimated focal length, has to be multiplied by f_0 at the end.

In Section 5 and in the appendix, we show that if f_0 happens to be equal to the true focal length, then the calibration equations may become degenerate. Thus, with f_0 close to the true focal length, one may expect an instable focal length estimation. In Section 6.1.4, this is shown to occur in some situations. On the other hand, when applying no such scaling, instabilities were observed in other situations. A rule of thumb that we apply in practice is thus to apply a scaling by a value f_0 significantly larger than the maximum expected focal length. This (admittedly ad hoc) procedure gave always good performance.

4. Generic singularities

Before discussing singularities associated with the above calibration equations, we describe the generic singularities of the underlying problem, i.e., those that cannot be overcome by *any* algorithm. They can be obtained rather directly by specializing the results obtained for *varying* focal lengths [14,15,17,21].

The only critical configurations for the (self-) calibration of a constant focal length from two views are:

- the optical axes are parallel to each other, or
- the optical axes intersect at a finite point and the optical centers are equidistant from this point. We refer to this configuration as the *equidistance configuration*. We may consider that it subsumes the case of parallel optical axes: although the optical axes intersect at a point at infinity, we may consider that the intersection point is equidistant from the optical centers (at infinite distance).

In both these cases, there is an infinite number of solutions for f^2.

Kahl and Triggs [13] have derived critical configurations. However, their results are not as clearly stated as above, and seem slightly incomplete. For example, their "turntable" rotation about the intersection point of the optical axes cannot produce all possible cyclotorsions of the two cameras, i.e., rotations about

their optical axes (which do not affect the self-calibration problem discussed in this paper).

Coplanarity of the optical axes is a necessary condition for a singular configuration with *equal* focal lengths, whereas it is already sufficient if two *different* focal lengths have to be estimated [14,15,17,21]. We will see in the following section that the quadratic Eq. (4) is nearly only degenerate in the generic singular cases (with the exception of $f = \pm 1$). On the other hand, the linear equations are degenerate when the two optical axes are coplanar, and in a particular case of little practical importance.

The stability of calibration in near-degenerate situations should be better for the equal focal length case.

5. Singularities of the calibration equations

It is useful to examine the singularities of the above calibration equations. Here we will determine under what conditions the individual equations become singular. This will allow us to see if they suffer from non-generic singularities and possibly to determine which equation to use under what condition, or to determine a single equation that should always be used.

The equations are said to be singular if they lead to invalid solutions. Such solutions may arise when there is an infinite number of choices for the coefficients of the equations' unknowns, or when the coefficients are equal to zero. If the SVD of G is unique (up to sign or swapping the columns of U and V and corresponding singular values), the forms of (4), (2), and (3) are unique. Otherwise, there may be invalid solutions. In the absence of noise, the true squared focal length is necessarily a solution of the equations. For the quadratic equation, there is in general a second, spurious solution. In most cases, this is a negative value and can thus be discarded (since we are looking for the *squared* focal length). In some cases, however, the equations may have an infinite number of solutions: for certain singular *relative camera poses*, all coefficients of our polynomial equations vanish, implying an infinite number of solutions for f. In the following, all singular *relative camera poses* are summarized. Proofs for the following statements are given in the appendices.

All three equations vanish of course in the generic singular conditions given in Section 4, i.e., their coefficients all become zero here. For the quadratic equation, there are, in general, no further singularities (unlike the general Kruppa equations that are subject to non-generic singularities). The only exception occurs when the true focal length equals ± 1, which means that the semi-calibrated fundamental matrix is a fully calibrated fundamental matrix, i.e., an essential matrix. This can happen if the fundamental matrix is expressed in perfectly standardized coordinates, meaning that the coordinate scaling recommended in [8] happens to be done by the inverse of the focal length. The essential matrix has two equal non-zero singular values, which means that its SVD is not unique: there is a one-degree-of-freedom family of possible SVDs. It is shown in Appendix D.2 that, depending on which

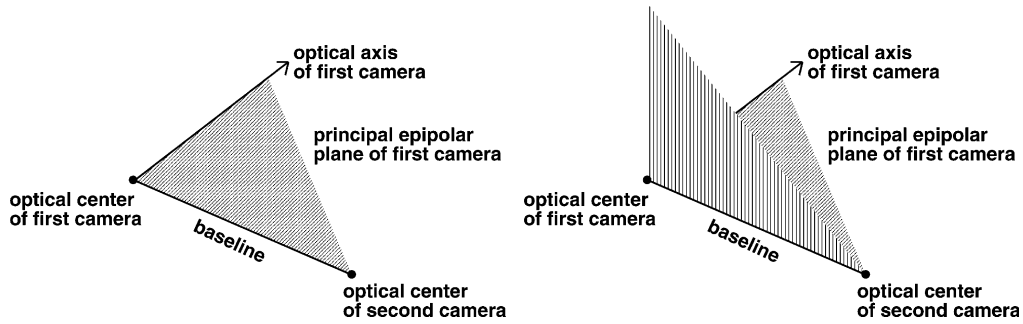


Fig. 1. Example of a singular case for the linear equations when the optical axes are not coplanar. (Left) The notion of principal epipolar plane is illustrated (plane spanned by the optical centers and one optical axis). (Right) If the optical axis of the second camera lies anywhere in the plane Π, which is orthogonal to the first camera's principal epipolar plane, then the linear equations become degenerate. In that case the two principal epipolar planes are orthogonal to one another (unless the optical axis points towards the first camera's optical center, in which case the principal epipolar plane of the second camera is not defined).

of the ambiguous SVDs one happens to compute in practice,[3] the quadratic equation's coefficients may vanish, even for a camera configuration that is generically non-singular. We show in Appendix D.2 that only a finite number, among the infinite number of possible SVDs, cause such a singularity. It is thus unlikely to encounter exactly such a case. However, when working in standardized coordinates (or, when scaling with approximately the true inverse focal length), one may get close enough, in which case noise in the data may create instabilities. This effect is studied using simulations, cf. Section 6.1.4, and conclusions are stated above in Section 3.2.

For the linear equations, there is degeneracy in two cases. The first case is when the optical axes are coplanar. The other case is best explained as follows. The family of epipolar planes consists of the pencil of planes that contain the cameras' baseline, i.e., the line joining the two optical centers. We define a *principal epipolar plane* associated with a camera as the epipolar plane that contains its optical axis, cf. the left part of Fig. 1. This is uniquely defined unless the optical axis coincides with the baseline, in which case, at least one camera looks straight at the other one. The non-generic singularities of the two linear calibration equations can be described, using the principal epipolar planes of the two cameras, in the following scenarios:

- Neither of the two principal epipolar planes is uniquely defined. This means that the two optical axes are identical, which implies of course that they are parallel (and coplanar). This is a generic singular case, and naturally all three equations become degenerate.
- One of the principal epipolar planes is not uniquely defined. This is a special case of coplanar optical axes. The linear equations degenerate, whereas the quadratic one does not in general.

[3] This depends on the implementation used for SVD computation and the outcome is possibly non-deterministic.

- The principal epipolar planes are identical. This means that the optical axes are coplanar. The linear equations degenerate. The quadratic equation degenerates only if, in addition, the equidistance configuration is present. Otherwise, its spurious solution is always zero (cf. Section E.3), i.e., the true solution can be obtained without ambiguity.
- The principal epipolar planes are orthogonal to each other. In this case, the linear equations degenerate. The quadratic equation does not degenerate, and its spurious solution is always negative or zero (cf. Section F.2), i.e., the true solution can be obtained without ambiguity. This situation is illustrated in the right part of Fig. 1.

Summary. The quadratic equation is degenerate practically only in generic singular configurations. In addition, whenever the linear equations degenerate in generic non-singular configurations, the quadratic one gives a unique admissible solution for the squared focal length.

It is interesting to note that the non-generic singularities for the linear equations (coplanar optical axes and orthogonal principal epipolar planes) correspond to generic singular camera configurations for the case of *different* focal lengths [14,15,17,21].

6. Experimental results

We conducted various experiments with our algorithm, to evaluate its performance with respect to several factors. Specifically, we studied its behavior in the proximity of singular configurations. This was done systematically using both simulated data and real data to give some intuition on how much effort has to be spent in avoiding singularities in practice. We also evaluated the performance with respect to the level of noise in the data and with respect to errors in the assumption of the location of the principal point. Experiments with real images were carried out for images of a calibration grid and also for images of a few generic scenes.

6.1. Simulated data

We conducted simulated experiments to assess the sensitivity of the calibration equations in close-to-singular situations. Fig. 2 shows the simulated scenarios. The starting position of the cameras is depicted on the left. It is the typical stereo situation, with symmetric vergence angles α. This situation is singular: the optical axes are coplanar and the optical centers are equidistant from the intersection point of the optical axes.

In the first scenario, the second camera rotates away from the plane spanned by the initial position of the optical axes, by an angle between 0° and 5° (“elevation angle”). In Fig. 2, this rotation would be towards the reader.

In the second scenario (shown on the right of Fig. 2), the second camera moves along its optical axis. The optical axes stay coplanar, but the distances of the optical centers to the intersection point of the optical axes are no longer equal. Hence, the

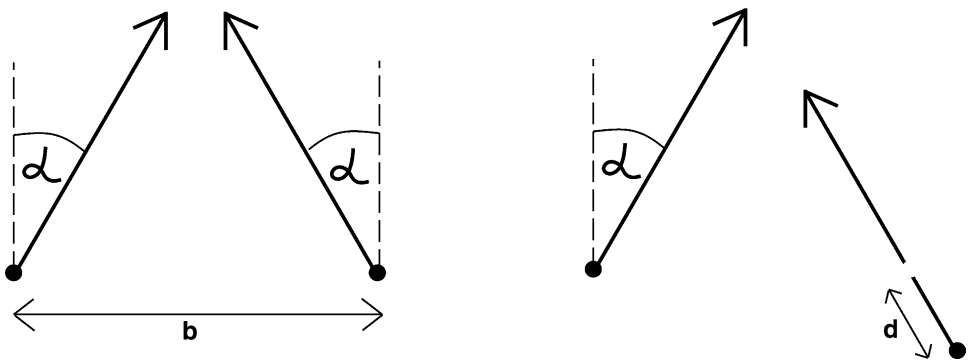


Fig. 2. Simulation scenarios. Shown are the optical centers and optical axes. (Left) Initial camera pose; b is the distance between the optical centers and α the vergence angle of the optical axes. (Right) Second simulation scenario; the second camera is moved along its optical axis by the distance d.

scenario is not singular any more (generically, and for the quadratic equation), besides for the case of a zero vergence angle (parallel optical axes). The baseline of the system is $b = 1000$ U, and the displacement of the second camera is by $d = -250, -200, \ldots, 250$ U.

For both scenarios, experiments are done with different vergence angles, with α between 0° (parallel optical axes in the initial position) and 30°. Three dimensional scene points are created randomly as follows: their coordinates are drawn from a uniform distribution inside a rectangular volume in front of the cameras, whose depth is 10 times the baseline. Only points inside the field of view of both cameras are used. Cameras are simulated with a focal length of 1000 pixels and a field of view of 28.7°, corresponding to images of size 512×512. By default, 100 points are used in each experiment, unless otherwise stated. The 3D points are projected to the images, and centered Gaussian noise (with a standard deviation between 0 and 1 pixels), is added to the image point coordinates. These image points are the input to the algorithm.

The following figures show mainly results for the quadratic equation. Results for the linear equations are not shown here, however, they are discussed in the text. Displayed are the median values of the relative errors on the focal length (ratio of the difference between true and estimated focal length, and the true focal length); each data point in the graphs is the result of 1000 random experiments. In all simulated experiments, the 8-point method of [8] is used to compute the fundamental matrix, i.e., no non-linear optimization was done.

6.1.1. First scenario: off-plane rotation

Fig. 3 shows results for this scenario. The upper left part is relative to a zero vergence angle (i.e., with an elevation angle of 0°, the optical axes are parallel and the configuration is singular), and the upper right part is relative to a vergence angle of 5°. For zero vergence, it can be seen that even for a 3° rotation off the base plane, the errors are below 10% for realistic noise levels. Slight vergence of the cameras significantly improves the results (compare the upper right with the upper left part of Fig. 3).

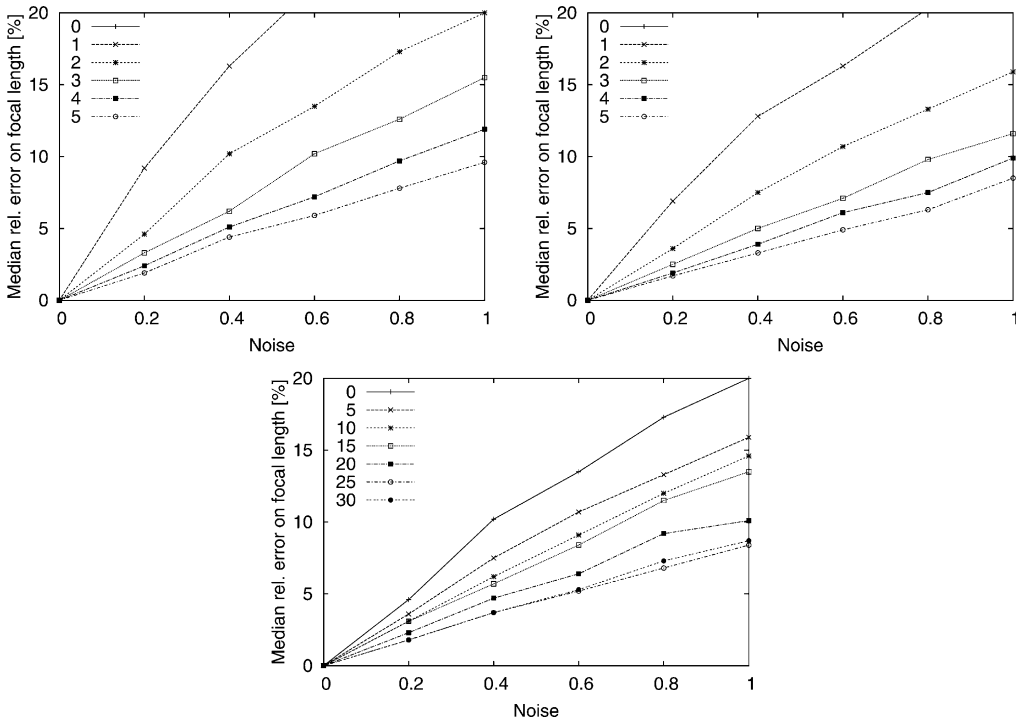


Fig. 3. First scenario. (Top) Results are shown for different elevation angles (one curve per elevation angle, from 0° to 5°, cf. the graphs' legends). The curves for 0° elevation are outside the graphs (this situation is singular, and the results reflect this). (Upper left) Vergence fixed to 0°. (Upper right) Vergence fixed to 5°. (Bottom) Elevation angle fixed to 2°, results shown for different vergence angles (one curve per vergence angle, 0°, 5°, ..., 30°, cf. legend).

In the lower part of Fig. 3, the elevation angle is kept fixed to 2°, to illustrate the influence of the vergence angle α. It is intuitive that with a vergence angle of 0°, the configuration is "closer" to the degenerate situation of parallel optical axes, thus the focal length estimation less stable, compared to larger vergence angles. This is reflected in the graph: the error in the estimated focal length decreases with increasing vergence angle, although above 25° vergence, there is no further significant improvement.

It is worthy to note that the linear equations gave nearly identical results to the quadratic one in this scenario. Since two linear equations are available, the average of their results is taken as estimated focal length, unless one of the two gave a negative solution for f^2, in which case only the solution of the other equation was used of course.

6.1.2. Second scenario: displacement of the second camera

Fig. 4 shows results for the second scenario. The upper part of the figure shows the influence of the vergence angle for a fixed, relatively small displacement (5% of the baseline) of the second camera. For a vergence angle of 0°, the optical axes

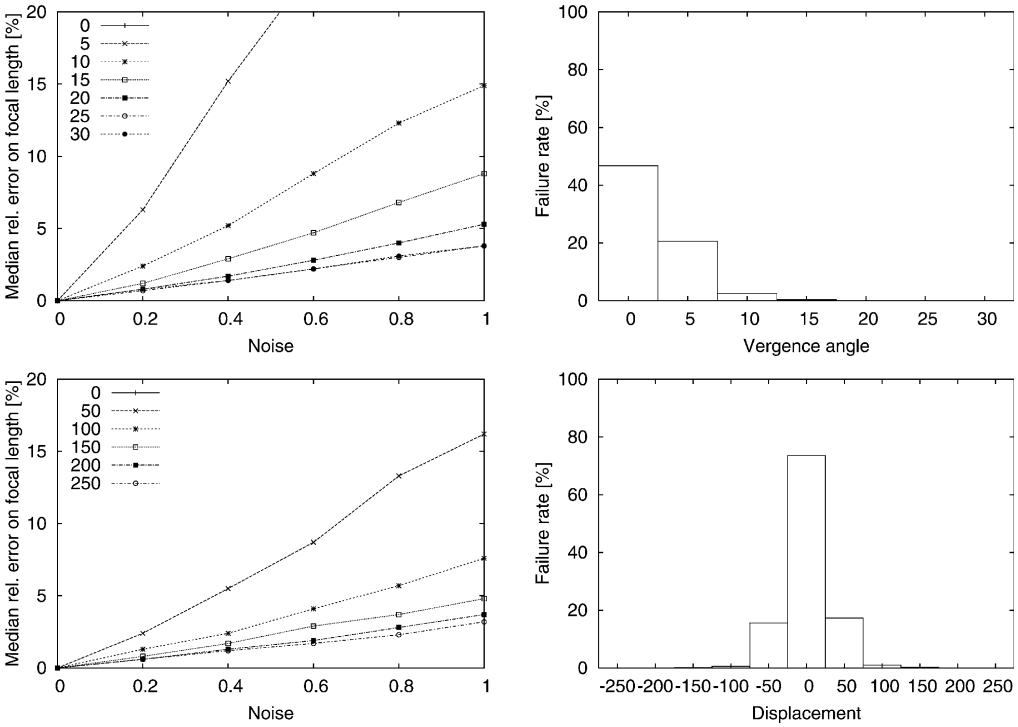


Fig. 4. Second scenario. (Top) Fixed displacement $d = -50$. (Upper left) Relative errors on estimated focal length for different vergence angles. (Upper right) Failure rates (see text) for a noise level of 0.6 pixels and different vergence angles. (Bottom) Fixed vergence angle of 10°. (Lower left) Relative errors on estimated focal length for different displacements (for $d = 0, 50, \ldots, 250$ U, cf. graph's legend). (Lower right) Failure rates for a noise level of 1 pixel.

are parallel and the situation remains singular for any displacement, which is reflected by the fact that the corresponding curve is outside the graph. The figure shows that close to 0° vergence, the results are heavily affected by the near-singularity and noise, but they stabilize with increasing vergence angle. This is shown by the error on the focal length, which decreases significantly with increasing vergence angle (upper left part of Fig. 4), as well as by the decreasing failure rate (upper right). Failure was declared whenever the quadratic equation did not admit a positive solution.

The lower part of Fig. 4 shows the results with respect to varying displacement, for a fixed vergence angle of 10°. The curve for zero displacement is outside the graph (this corresponds to the singular equidistance configuration). With increasing displacement, the performance increases as expected, both in terms of relative error on the estimated focal length and failure rate. The graphs for displacements towards the scene (negative d) are not plotted in the lower left part of Fig. 4, for the sake of clarity; note that the graph for a value of $-d$ is very similar to that for d.

As for the linear equations, this scenario is singular (coplanar optical axes). This is reflected by experimental results (not shown here), where relative errors are sometimes above 100%, and nearly always above 70% (besides a high failure rate).

6.1.3. Influence of the number of point correspondences

In Fig. 5, we show results on the influence of the number of point correspondences used for computing the fundamental matrix. As expected, performance increases with the number of points, with an asymptotic behavior.

6.1.4. Influence of standardization

As discussed in Sections 3.2 and 5, the use of standardized coordinates (in our case, a scaling) has to be considered more closely. Here, we show results obtained with different scalings. The x-axis of the graphs in Figs. 6 and 7 shows the inverse scale factor applied to the fundamental matrix according to Eq. (5) (remember that the true focal length is 1000). The graphs show the percentage of random experiments where the focal length was estimated (positive solution for f^2) and was within 10% of the ground truth value. Results are shown for both, quadratic and linear equations.

Fig. 6 shows results for the first scenario. All graphs show a clear "performance hole" when scaling is done with a factor close to the actual inverse focal length. With decreasing elevation angle (bottom to top) and increasing noise (left to right), the instability caused by scaling with the inverse focal length, gets combined with the increasing instability due to getting closer to the singular equidistance case. In the least favorable case (upper right), the success rate drops to an average of around 30%. Overall, the linear equations are much more sensitive to the scale factor, compared to the quadratic equation, which has close to 100% success in the favorable case on the lower left, even when scaling is done with approximately the true inverse focal length.

Fig. 7 shows results for the second scenario. The linear equation is degenerate here, and the results are always bad, as stated in Section 6.1.2. As for the quadratic equation, the same performance hole as above around the true focal length can be observed. Interestingly, performance also drops significantly for scale factors below 50 (extreme left side of the graphs); the only explanation we can think of is that in this special case, round-off error becomes too large.

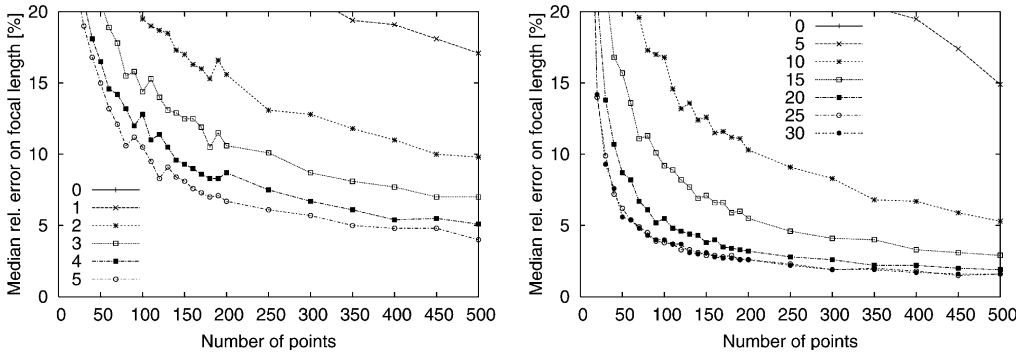


Fig. 5. (Left) First scenario (cf. Section 6.1.1), vergence fixed to 0°, noise level of 1 pixel, results for different elevation angles. (Right) Second scenario (cf. Section 6.1.2), displacement of −50, noise level of 1 pixel, results for different vergence angles.

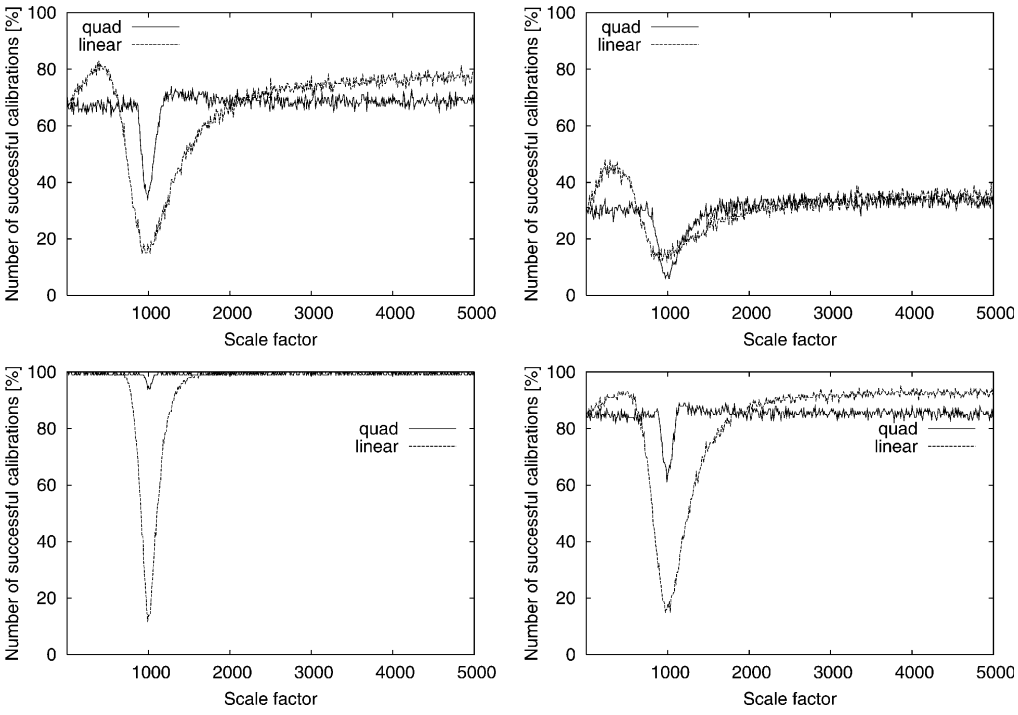


Fig. 6. First scenario, with vergence fixed to 30°. (Top) Elevation angle fixed to 1°. (Bottom) Elevation angle fixed to 5°. (Left column) Noise level of 0.4 pixels. (Right column) Noise level of 1 pixel.

Based on these observations, we decided to scale by a factor much lower than the inverse of the maximum expected focal length, as stated already in Section 3.2. In all other simulated experiments, a scale factor of 1/5000 was thus used, which always gave good results.

6.2. Real images of a calibration grid

Using real images of a calibration grid, we attempted to evaluate the algorithm's performance with respect to proximity to singular configurations and its sensitivity to the assumption of the principal point's position.

6.2.1. Experimental setup

It is relatively easy to avoid singular configurations in practice. Especially, one should avoid the case of coplanar optical axes. There are multiple ways to achieve this goal. One approach is as follows. Before taking the second image, point the camera to the same point in the scene as in the first image (this is simple to do with a viewfinder). Then, tilt the camera slightly upwards or downwards, and take the second image. Determining by how much one should tilt the camera is one of the goals of this experiment.

We took a total of 10 images of a calibration grid with a handheld camera. Fig. 8 shows some sample images. They were taken from 10 different positions, covering a

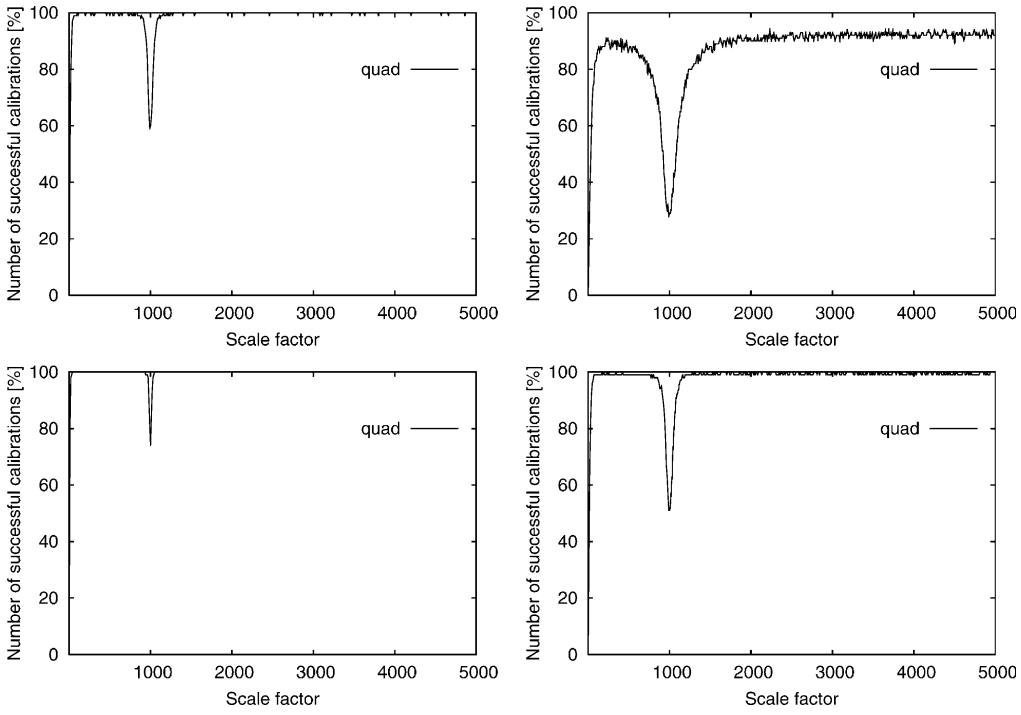


Fig. 7. Second scenario, with vergence fixed to 30°. (Top) Displacement of −50. (Bottom) Displacement of −100. (Left) Noise level of 0.4 pixels. (Right) Noise level of 1 pixel.

Fig. 8. Some images of the calibration grid.

roughly circular path around the grid (i.e., most pairs of views are close to the singular equidistance configuration, cf. Section 4). From each position, we applied a small tilt angle and then took one image as described above. Thus, among the 45 possible image pairs, some have approximately coplanar optical axes while some do not.

For this experiment and the ones in the next section, we used a Sony DSC-P31 digital camera with 5 mm focal length and chose a moderate image resolution of 640×480.

The camera was calibrated, including radial lens distortion, using all 10 images of the grid, by a photogrammetric calibration algorithm. The resulting focal length of

625 pixels is used as "ground truth" in the following experiments. The images were corrected for distortion before applying our algorithm. The extracted image positions of the grid's targets were used by our algorithm to compute the fundamental matrix.

6.2.2. Effect of principal point estimation on focal length calibration

As described above, our focal length calibration algorithm is based on the assumption that we know the other intrinsic parameters. Here, we show that an error on the assumed location of the principal point has little effect on the computed focal length. For one pair of images, we estimated the focal length repeatedly, changing (in steps of 5 pixels) the assumed coordinates of the principal point by up to $\pm$ 25 pixels from the image center in both directions.

Among the 121 different computed focal lengths, the maximum relative error with respect to the true focal length was 4.16%. The mean relative error was 0.2%. The standard deviation of the computed focal lengths was 11.7 pixels, i.e., only about 1.8% of the focal length. We conclude that realistic errors in the assumption of the principal point's position have little effect on our algorithm, at least concerning the range of accuracy that one can expect in our minimal scenario. Hence it is usually safe to assume that the principal point is at the image center when we use this algorithm for focal length calibration.

6.2.3. Stability of the algorithm

Here, we evaluate the algorithm's performance, with respect to how close the optical axes are to being coplanar. The calibration of our images, using a photogrammetric approach that makes use of the known geometry of the calibration grid, tells us the position of the optical centers and the optical axes for our 10 images. To measure how close the optical axes associated with two images are to being coplanar, we proceed as illustrated in the left part of Fig. 9: we compute the two principal epipolar planes $p1$ and $p2$ (cf. Section 5). The "middle plane" is the plane that "bisects" $p1$ and $p2$. The angle c between the middle plane and $p1$ (or, equivalently, $p2$) is our measure for the deviation from the case of coplanar optical axes. In addition, we also considered a measure for how close the two optical axes are from being parallel, but which was found to be less significant for the following evaluation.

We applied our algorithm to all 45 possible image pairs formed by our 10 input images. The estimated values of the focal length are plotted in the right part of Fig. 9, over the value of the angle c for the corresponding image pair.

We observe three groups of results:

- For $c > 1.5°$, the calibrated focal lengths are quite precise and accurate. Their average is 627.6, which is nearly identical to the ground truth. Their standard deviation is about 6.5 pixels, i.e., about 1.1% of the focal length.
- For $c < 1°$, the results are not at all stable. Errors range from 25 to 280 pixels.
- For $1° < c < 1.5°$, the results are not very precise but become reasonably accurate.

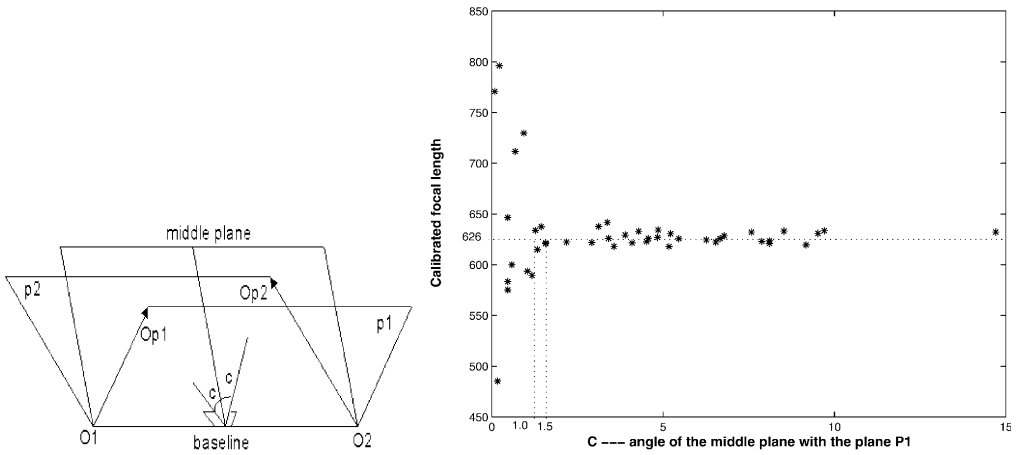


Fig. 9. (Left) The angle c used for measuring by how much an image pair deviates from having coplanar optical axes. (Right) Sensitivity of focal length with respect to the angle c.

We conclude that for the type of images tested, it is safe to run our algorithm whenever the angle c exceeds 1.5°. This corresponds to tilting the camera between two image acquisitions by about 10% of its opening angle, which seems to be reasonably achievable in practice. However, with a lower accuracy in image point extraction, this value will increase. In Section 6.3, we thus test our algorithm with real images of generic scenes.

6.2.4. 3D reconstruction results using the calibrated focal length

Having calibrated the focal length, we can estimate the relative position of the two considered images [11] and carry out a 3D reconstruction of the matched image points [10]. We did this for several image pairs. To evaluate the quality of the 3D reconstruction, we compare it to the known geometry of the calibration grid. We take two steps to achieve this objective. First, we fit planes to the three subsets of coplanar points (cf. Fig. 8). Here, we design a relative distance to evaluate the coplanarity of points. Specifically, we first measure the distances of points to the fitted plane. Next, we compute the largest distance between pairs of the considered points. The distances of the points to the plane are then normalized by this largest distance. The obtained distances (in percent) are the so-called relative distances. Second, we measure the angles between each pair of planes and compare it to the "ground truth": one of the grid's planes forms 90° angles with the two others, which themselves form a 120° angle.

The results of our evaluation are displayed in Table 1. They are shown for five pairs, which share one common image. Note that from left to right, the baseline (the distance between optical centers) decreases. Row f contains the calibrated focal lengths. The rows A_{ij} (*with* $i,j \in [1,2,3]$) show the angles between pairs of planes. The rows Std_i (*with* $i \in [1,2,3]$) show, for the 3 planes, the standard deviation of the

Table 1
Reconstruction results using calibrated focal length

	Ground truth	Pair 1	Pair 2	Pair 3	Pair 4	Pair 5
f	625.0	622.3	633.0	632.0	628.4	623.6
A12	90.0	90.17	89.75	91.12	90.49	89.89
A13	90.0	89.65	89.34	92.18	91.36	88.87
A23	120.0	119.79	119.88	120.32	120.57	118.56
Std1	0.0	1.3e−4	1.7e−4	2.4e−4	3.1e−4	3.2e−4
Std2	0.0	3.4e−4	3.5e−4	2.6e−4	3.8e−4	2.8e−4
Std3	0.0	2.8e−4	3.1e−4	4.9e−4	5.3e−4	3.8e−4

relative distances as described above, which is useful to evaluate the coplanarity of points.

We observe that for the two image pairs with the largest baselines, the angles are all within 0.3° from their true values. With decreasing baseline, the errors generally increase, both for the angles and the coplanarity measure, although they still stay relatively small.

6.3. Real images of generic scenes

For the images of the calibration grid, image point matching was provided due to the easy identification of the targets. Here, we consider images of two generic scenes. Interest point extraction and matching is done automatically using the available software[4] (see also [22]). The same camera zoom setting as in Section 6.2 was used, which provides the "ground truth" value for the focal length in Tables 2 and 3.

6.3.1. An outdoor scene

We took five images of a building of the National University of Singapore (see Fig. 10, for examples). The distance between the camera and the building is about 25 m. The results for several image pairs are presented in Table 2 (camera configurations close to the coplanar case give poor results which are not shown here). After calibration, we also reconstructed the building. We chose the median of the seven calibrated results as shown in Table 2, and used the result to reconstruct the building's two faces with the right angle. We found that the reconstructed results (about 85°) are roughly close to the ground truth (the relative error is about 5%).

When analyzing the results of Table 2, we need to consider the following issue. Although the same zoom setting was used as for the images of the calibration grid, the camera focused on a scene at a different distance. Hence, comparatively large relative errors of several percent may be expected. Here, the maximum relative error is about 10%, which seems reasonable for this experiment.

[4] http://www-sop.inria.fr/robotvis/personnel/~zzhang/softwares.html.

Table 2
Results for image pairs of the building, cf. Fig. 10

Image pair	Ground truth	12	14	15	23	25	34	35
f	625.0	643.2	654.3	604.7	688.6	689.8	592.4	657.7

The label "12" in the first row stands for the pair of images 1 and 2, and analogously for the other labels.

Table 3
Results for image pairs of the 3 cups, cf. Fig. 11

Image pair	Ground truth	12	13	14	23	24	34
f	625.0	602.4	604.8	596.9	621.3	612.7	623.7

The label "12" in the first row stands for the pair of images 1 and 2, and analogously for the other labels.

Fig. 10. Some images of the building.

6.3.2. An indoor scene

We took four images of a simple indoor scene as Fig. 11 shows. Interest points were mainly extracted on the three cups and just a few on the plug in the background, i.e., the scene is relatively "flat."

The estimated focal lengths, for all 6 possible image pairs, are shown in Table 3. Again, the same camera setting as in Section 6.2 was used. The maximum relative error is about 6.5%, and the average relative error is less than 5%.

Fig. 11. Images of 3 cups.

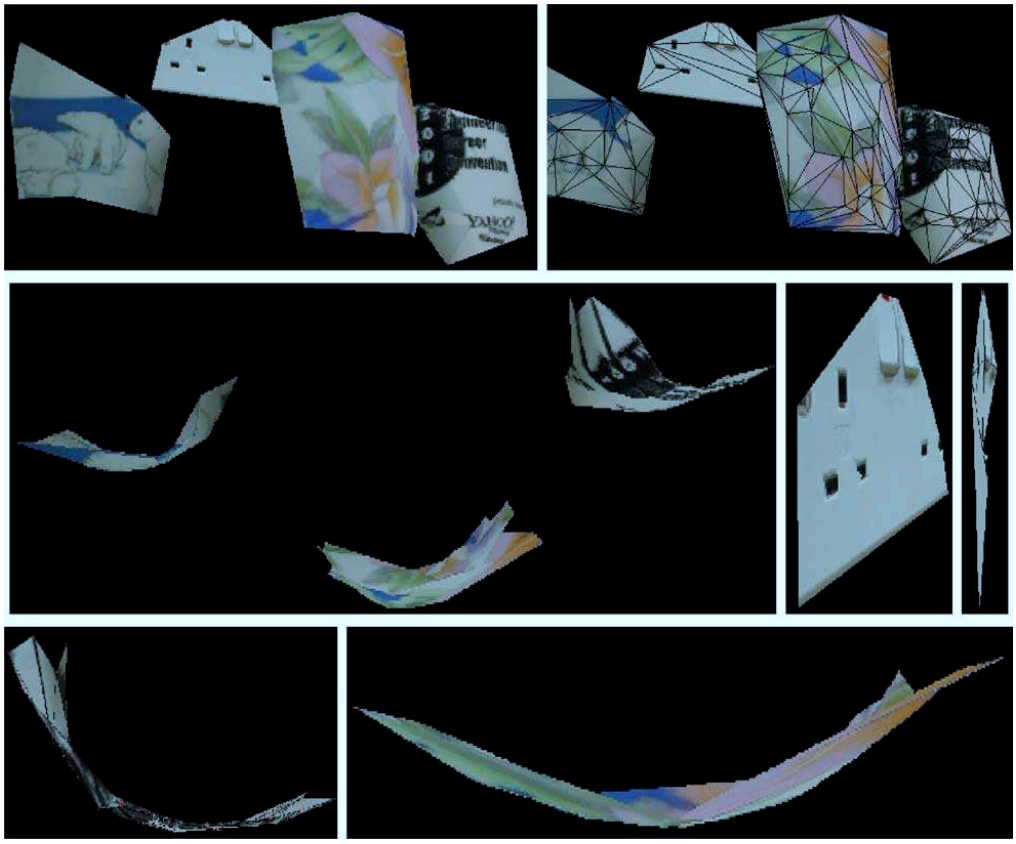


Fig. 12. Rendering of the reconstructed cup scene. (First row) General appearance of the scene, once with overlaid triangular mesh. (Second row) Rough top view of cups and two close-ups of the plug in the background (rightmost image shows the near coplanarity of the reconstruction). (Third row) Top views of two of the cups, showing that their cylindrical shape has been recovered.

As we did for the images of the calibration grid, we performed a 3D reconstruction of the scene using the calibration result. A triangular mesh is semi-automatically adjusted to the reconstructed 3D points, and used to create textured VRML models. A few renderings of one of the models are shown in Fig. 12. Due to the sparseness of the extracted interest points, the reconstruction of the scene is not complete. However, Fig. 12 shows that it is qualitatively correct, as explained in the caption of the figure.

7. Conclusions

We have analyzed the problem of focal length calibration from two views of an unknown scene, given their epipolar geometry and the assumption that the views have identical focal length. Closed form solutions have been derived, which

consist of one quadratic and two linear equations (which are algebraically interdependent). We have studied critical camera configurations in detail. Our experimental results suggest that in practice such configurations are relatively easy to avoid. Acceptably accurate results can be obtained when these singular configurations are avoided.

Acknowledgment

Z.L. Cheng is very grateful for the scholarship granted by the National University of Singapore.

Appendix A. Background

We describe a few known results about matrix decompositions that will be used in the following sections. Let the SVD of a 3×3 matrix M of rank 2 be given as

$$\mathsf{M} \overset{SVD}{=} \mathsf{U}\Sigma\mathsf{V}^{\mathrm{T}} = (\,\mathbf{u}_1 \quad \mathbf{u}_2 \quad \mathbf{u}_3\,) \begin{pmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \mathbf{v}_1^{\mathrm{T}} \\ \mathbf{v}_2^{\mathrm{T}} \\ \mathbf{v}_3^{\mathrm{T}} \end{pmatrix}.$$

The right null-vectors of M are equal (up to scale) to the third column $\mathbf{v}_3$ of V. As for the left null-vectors of M, they are equal (up to scale) to the third column $\mathbf{u}_3$ of U.

In the following, we suppose that $\sigma_1 \neq \sigma_2$. Consider the symmetric matrix $\mathsf{M}^{\mathrm{T}}\mathsf{M}$. It has 0, σ_1^2, and σ_2^2 as eigenvalues. The eigenvectors of $\mathsf{M}^{\mathrm{T}}\mathsf{M}$ to the eigenvalue σ_i^2 $(i = 1, 2)$ are equal (up to scale) to the ith column $\mathbf{v}_i$ of V.

Similarly, the eigenvectors of $\mathsf{M}\mathsf{M}^{\mathrm{T}}$ to the eigenvalue σ_i^2 $(i = 1, 2)$ are equal (up to scale) to the ith column $\mathbf{u}_i$ of U.

Appendix B. Parameterization of relative pose

In the following sections, we derive singular camera configurations. A geometric description is most useful. (Non-) Singularity only depends on the relative pose of the two views (and, in some very special cases, on the actual value of the focal length). Since only *relative* pose matters, we assume, without loss of generality, that the optical center of the first camera is the origin. Furthermore, we assume that its optical axis coincides with the Z-axis. Hence, the rotational part of its pose consists of a rotation $\mathsf{R}_{Z,1}$ about the Z-axis (cyclotorsion). This may, again without loss of generality, be chosen such that the optical center of the second camera lies in the plane $X = 0$, i.e., its coordinates are $(0, Y, Z)$. Without loss of generality, we may fur-

thermore impose that the distance between the two cameras is equal to 1. Hence, the second camera's position may be parameterized by an angle γ

$$\begin{pmatrix} 0 \\ \cos\gamma \\ \sin\gamma \\ 1 \end{pmatrix}.$$

Let the second camera's orientation be given by three elementary rotation matrices: $\mathsf{R}_2 = \mathsf{R}_{Z,2}\mathsf{R}_Y\mathsf{R}_X$. The semi-calibrated fundamental matrix for this parameterization is then given by

$$\mathsf{G} \sim \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & f \end{pmatrix} \mathsf{R}_{Z,2}\mathsf{R}_Y\mathsf{R}_X \left[\begin{pmatrix} 0 \\ \cos\gamma \\ \sin\gamma \end{pmatrix} \right]_\times \mathsf{R}_{Z,1} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & f \end{pmatrix}. \tag{B.1}$$

Note that the rotations $\mathsf{R}_{Z,1}$ and $\mathsf{R}_{Z,2}$ have the following special form:

$$\mathsf{R}_{Z,i} = \begin{pmatrix} \cdot & \cdot & 0 \\ \cdot & \cdot & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Hence, Eq. (B.1) can be rewritten as

$$\mathsf{G} \sim \mathsf{R}_{Z,2} \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & f \end{pmatrix} \mathsf{R}_Y\mathsf{R}_X \left[\begin{pmatrix} 0 \\ \cos\gamma \\ \sin\gamma \end{pmatrix} \right]_\times \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & f \end{pmatrix}}_{\mathsf{H}} \mathsf{R}_{Z,1}. \tag{B.2}$$

Due to the special form of $\mathsf{R}_{Z,1}$ and $\mathsf{R}_{Z,2}$ and the orthogonality of the left and right singular matrices of an SVD, G and H have the same singular values and the third rows of their respective matrices U and V are equal to one another (up to sign at least). Specifically, this means that the SVDs of G and H lead to the same calibration equations.[5]

Hence, we may analyze the singularities of the calibration equations by studying the SVD of H, which allows us to express algebraic singularity conditions relatively easily in geometric terms, i.e., in terms of relative pose.

The matrix H, defined in (B.2) is given explicitly as

$$\mathsf{H} \sim \begin{pmatrix} (\sin\gamma\sin\alpha - \cos\gamma\cos\alpha)\sin\beta & -\sin\gamma\cos\beta & f\cos\gamma\cos\beta \\ \sin\gamma\cos\alpha + \cos\gamma\sin\alpha & 0 & 0 \\ f(\sin\gamma\sin\alpha - \cos\gamma\cos\alpha)\cos\beta & f\sin\gamma\sin\beta & -f^2\cos\gamma\sin\beta \end{pmatrix}. \tag{B.3}$$

Here, α and β are the angles of R_X and R_Y, respectively.

[5] In fact, this illustrates that cyclotorsion (rotation about the optical axis) does not influence focal length calibration.

In the following, we express conditions for coplanar or parallel optical axes, etc., in terms of the relative pose parameters α, β, and γ.

The *optical axis* of the second camera has the direction

$$\mathbf{D} \sim \begin{pmatrix} -\sin\beta \\ \sin\alpha\cos\beta \\ \cos\alpha\cos\beta \\ 0 \end{pmatrix}. \tag{B.4}$$

Since the direction of the first optical axis is given by $(0,0,1,0)^{\mathrm{T}}$, the optical axes are *parallel* exactly if

$$\sin\alpha = \sin\beta = 0. \tag{B.5}$$

The two optical axes are *coplanar* exactly if $H_{33} = 0$, hence if $\cos\gamma = 0$ or $\sin\beta = 0$ (cf. Eq. (B.3)). The case $\cos\gamma = 0$ means that the second camera's optical center lies on the first camera's optical axis.

Let us express these conditions in terms of the principal epipolar planes, defined in section 5. The two *principal epipolar planes* are computed as

$$\Pi_1 \sim \begin{pmatrix} \cos\gamma \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad \Pi_2 \sim \begin{pmatrix} \cos\beta(\cos\alpha\cos\gamma - \sin\alpha\sin\gamma) \\ -\sin\beta\sin\gamma \\ \sin\beta\cos\gamma \\ 0 \end{pmatrix}.$$

The optical axes are coplanar if one or both principal epipolar planes are not defined (algebraically, if all their coefficients are zero) or if they are identical. Naturally, we find the same conditions as above: when $\cos\gamma = 0$, Π_1 is not defined (the second camera's optical center lies on the first optical axis). A necessary condition for Π_2 not being defined is $\sin\beta = 0$. In that case, we observe that Π_1 and Π_2 are identical (their coordinate vectors are equal up to scale), thus the optical axes are coplanar.

Besides the different conditions for coplanar optical axes, another configuration is relevant: *mutually orthogonal principal epipolar planes* (cf. Section 5). This means that the scalar product of their normals (the upper 3-subvectors of Π_1 and Π_2) vanishes, which happens exactly if $\Pi_{2,1} = 0$ (we exclude $\cos\gamma = 0$ since Π_1 is assumed to be defined):

$$\cos\beta(\cos\alpha\cos\gamma - \sin\alpha\sin\gamma) = 0.$$

Let us now consider the *equidistance configuration*: the optical axes are coplanar (but not parallel) and the optical centers are at the same distance from the intersection point of the optical axes. Let us develop this case for the two conditions of coplanar optical axes:

- $\cos\gamma = 0$. In that case, the second optical center is the intersection point of the two optical axes, hence equidistance is excluded.

- $\sin\beta = 0$. We exclude parallel optical axes, hence: $\sin\alpha \neq 0$. The intersection point of the optical axes is

$$\begin{pmatrix} 0 \\ 0 \\ \sin\gamma - \cos\gamma\frac{\cos\alpha}{\sin\alpha} \\ 1 \end{pmatrix}.$$

The squared distances to the optical centers are thus equal if

$$\frac{(\sin\gamma\sin\alpha - \cos\gamma\cos\alpha)^2}{\sin^2\alpha} = \frac{\cos^2\gamma}{\sin^2\alpha}$$

which (since $\sin\alpha \neq 0$) is equivalent to (after some trigonometric manipulations)

$$\sin\alpha(\cos^2\gamma - \sin^2\gamma) + 2\cos\alpha\cos\gamma\sin\gamma = 0. \tag{B.6}$$

The last case of interest is that of the angles between optical axes and baseline (line joining the optical centers) being equal. Note that this subsumes the equidistance configuration, but is more general. The condition for this case is given in the last row of the table.

All special cases of relative pose that are relevant in the following sections, are summarized in the table below.

Summary of relevant special cases for relative camera pose

Coplanar optical axes	$\cos\gamma = 0$ or $\sin\beta = 0$
2nd optical center on 1st optical axis	$\cos\gamma = 0$
1st optical center on 2nd optical axis	$\sin\beta = \cos\alpha\cos\gamma - \sin\alpha\sin\gamma = 0$
Parallel optical axes	$\sin\alpha = \sin\beta = 0$
Orthogonal principal epipolar planes	$\cos\beta(\cos\alpha\cos\gamma - \sin\alpha\sin\gamma) = 0$
Equidistance	$\sin\beta = \sin\alpha(\cos^2\gamma - \sin^2\gamma) + 2\cos\alpha\cos\gamma\sin\gamma = 0$
Equal angles between optical axes and baseline	$\sin^2\gamma = \cos^2\beta(\sin\alpha\cos\gamma + \cos\alpha\sin\gamma)^2$

Appendix C. Proofs for singularities of the calibration equations

Let us first define the meaning of singularity of the equations, based on observations made in Section 5: they are singular if all their coefficients vanish. In the following, we first derive conditions for singularity in terms of the elements of the SVD of G, respectively, H, concretely, in terms of the singular values a and b and the coefficients U_{31}, U_{32}, V_{31}, and V_{32} that show up in Eqs. (2)–(4). We then establish the corresponding geometrical configurations based on the proposed parameterization of relative pose.

The analysis of singularities is tricky due to the possibility that the SVD of the (semi-calibrated) fundamental matrix may not be unique. Note that the SVD is never

unique for any matrix: e.g., simultaneously scaling corresponding columns of U and V by -1 gives another valid SVD. Such manipulations lead to the same calibration equations, as may be verified by checking Eqs. (2)–(4). Thus, in the following, we speak of *ambiguous SVD* if there are *infinitely many* possible SVDs for a matrix. In our case, this is exactly the case if H has two equal non-zero singular values $a = b$: if

$$\mathsf{H} = \mathsf{U} \begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & 0 \end{pmatrix} \mathsf{V}^{\mathrm{T}}$$

is an SVD of H, then also

$$\mathsf{H} = \underbrace{\mathsf{U} \begin{pmatrix} \cos\rho & \sin\rho & 0 \\ -\sin\rho & \cos\rho & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{\mathsf{U}'} \begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & 0 \end{pmatrix} \underbrace{\begin{pmatrix} \cos\rho & -\sin\rho & 0 \\ \sin\rho & \cos\rho & 0 \\ 0 & 0 & 1 \end{pmatrix} \mathsf{V}^{\mathrm{T}}}_{\mathsf{V}'^{\mathrm{T}}}$$

for any angle ρ.

In the following two sections, we first analyze the case of ambiguous SVDs, followed by that of a unique one.

Appendix D. Singularities in the case of ambiguous SVDs

D.1. Cases of ambiguous SVDs

In the following, we derive all cases in which the singular values of H are equal. The singular values of H are the square roots of the eigenvalues of $\mathsf{H}^{\mathrm{T}}\mathsf{H}$

$$\mathsf{H}^{\mathrm{T}}\mathsf{H} \sim \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & f \end{pmatrix} \left[\begin{pmatrix} 0 \\ \cos\gamma \\ \sin\gamma \end{pmatrix} \right]_{\times} \mathsf{R}_X^{\mathrm{T}} \mathsf{R}_Y^{\mathrm{T}} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & f^2 \end{pmatrix} \mathsf{R}_Y \mathsf{R}_X \left[\begin{pmatrix} 0 \\ \cos\gamma \\ \sin\gamma \end{pmatrix} \right]_{\times} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & f \end{pmatrix}.$$

We want to find the conditions for which $\mathsf{H}^{\mathrm{T}}\mathsf{H}$ has two equal non-zero eigenvalues (and one that is zero). In that case, its characteristic polynomial must be of the form

$$\lambda(\lambda - a)^2 = \lambda^3 - 2a\lambda^2 + a^2\lambda.$$

Hence, if we denote by x_i the coefficient of λ^i, we must have

$$4x_1 - x_2^2 = 0. \tag{D.1}$$

Let us formulate this condition for the characteristic polynomial of $\mathsf{H}^{\mathrm{T}}\mathsf{H}$. In the following, we at times use the following compact notation: $c_\alpha = \cos\alpha$ and $s_\alpha = \sin\alpha$, and analogously for other angles.

The expression in (D.1) can be factorized in three factors:

$$-(f^2-1)^2, \tag{D.2}$$

$$(f^2-1)s_\beta^2 c_\gamma^2 + c_\gamma^2 + s_\beta^2 + c_\beta^2(c_\alpha c_\gamma - s_\alpha s_\gamma)^2 + 2c_\beta c_\gamma(c_\alpha c_\gamma - s_\alpha s_\gamma), \tag{D.3}$$

$$(f^2-1)s_\beta^2 c_\gamma^2 + c_\gamma^2 + s_\beta^2 + c_\beta^2(c_\alpha c_\gamma - s_\alpha s_\gamma)^2 - 2c_\beta c_\gamma(c_\alpha c_\gamma - s_\alpha s_\gamma). \tag{D.4}$$

If any one of the expressions (D.2)–(D.4) is equal to zero, then $\mathsf{H}^\mathsf{T}\mathsf{H}$ has two equal non-zero eigenvalues, and the SVD of H is not unique. The trivial case is obviously $f^2 = 1$ (from Eq. (D.2)). This will be dealt with in detail in Section D.2.

As for $f^2 \neq 1$, we will show in the following that expressions (D.3) or (D.4) are equal to zero exactly in generic singular configurations. We consider three cases: $c_\gamma = 0$, $s_\beta = 0$, and $c_\gamma, s_\beta \neq 0$.

- $c_\gamma = 0$. The expressions in (D.3) and (D.4) are identical in this case: $s_\beta^2 + c_\beta^2 s_\alpha^2$. This is zero exactly if $s_\alpha = s_\beta = 0$. This means exactly, cf. the table in Appendix B, that the second camera lies on the optical axis of the first one ($\cos\gamma = 0$) and that their optical axes are identical (since they are parallel, due to $s_\alpha = s_\beta = 0$). Hence, we are in a special case of parallel optical axes, which is of course a generic degenerate situation.
- $s_\beta = 0$. The expressions in (D.3) and (D.4) become ("+" for (D.3) and "−" for (D.4))

  $$c_\gamma^2 + (c_\alpha c_\gamma - s_\alpha s_\gamma)^2 \pm 2c_\gamma(c_\alpha c_\gamma - s_\alpha s_\gamma) = (c_\gamma \pm (c_\alpha c_\gamma - s_\alpha s_\gamma))^2.$$

  This is zero (for either "+" or "−") exactly if

  $$c_\gamma^2 = (c_\alpha c_\gamma - s_\alpha s_\gamma)^2.$$

  Using trigonometric manipulations, this can be transformed into:

  $$s_\alpha^2\left(s_\alpha\left(c_\gamma^2 - s_\gamma^2\right) + 2c_\alpha c_\gamma s_\gamma\right)^2 = 0.$$

  This holds if $s_\alpha = 0$ or $s_\alpha(c_\gamma^2 - s_\gamma^2) + 2c_\alpha c_\gamma s_\gamma = 0$. The first condition corresponds to parallel optical axes and the second one to the equidistance configuration, cf. the table in Appendix B. Hence, as above, the expressions (D.3) and (D.4) can only be zero (for $f^2 \neq 1$) in generic degenerate situations.

- $c_\gamma, s_\beta \neq 0$. We show in the following that under these assumptions, the expressions (D.3) and (D.4) cannot be zero for *positive* values of f^2. Expressions (D.3) or (D.4) being zero leads to (division by $s_\beta^2 c_\gamma^2$ is allowed since this is assumed to be non-zero here)

  $$f^2 = \frac{-s_\beta^2 s_\gamma^2 - c_\gamma^2 - c_\beta^2(c_\alpha c_\gamma - s_\alpha s_\gamma)^2 \pm 2c_\beta c_\gamma(c_\alpha c_\gamma - s_\alpha s_\gamma)}{s_\beta^2 c_\gamma^2}.$$

  Here, "+" corresponds to (D.3) and "−" to (D.4). We develop this equation:

$$f^2 = \frac{-s_\beta^2 s_\gamma^2 - \left(c_\gamma \mp c_\beta (c_\alpha c_\gamma - s_\alpha s_\gamma)\right)^2}{s_\beta^2 c_\gamma^2}.$$

The right-hand side of this equation can obviously never be positive. Hence, the equation can never be true for real values of f, meaning that expressions (D.3) and (D.4) can not be zero (for $f^2 \neq 1$ and under the assumption $c_\gamma, s_\beta \neq 0$).

Summary. The semi-calibrated fundamental matrix has equal non-zero singular values exactly in the case $f = \pm 1$ (expression (D.2)) or if the cameras are in equidistance configuration (includes the case of parallel optical axes). In the first case, the fundamental matrix is actually the essential matrix of the camera pair. In practice, $f = \pm 1$ can happen if one works in standardized image coordinates [8] (which often comes down to scaling the images by approximately the inverse focal length), which is usually recommended for numerical reasons. As for the second case, equidistance, this represents a generic singularity, hence the calibration equations become singular anyway. In the following section, we thus only analyze the case $f = \pm 1$.

D.2. The case $f = \pm 1$

In the following, we only consider the case $f = +1$; as for $f = -1$, the equations are analogous, with only sign changes in appropriate places. The matrix H is now given by

$$\mathsf{H} \sim \mathsf{R}_Y \mathsf{R}_X \left[\begin{pmatrix} 0 \\ \cos\gamma \\ \sin\gamma \end{pmatrix} \right]_\times .$$

As proven above, H has two equal singular values, i.e. its SVD is not unique. In practice, the SVD one obtains depends on the actual numerical implementation used to compute it. We want to investigate if our calibration equations may be singular for some SVDs and non-singular for others, or if they are (non-) singular irrespective of the actual SVD.

We write H in detail

$$\begin{aligned} \mathsf{H} &\sim \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} 0 & -\sin\gamma & \cos\gamma \\ \sin\gamma & 0 & 0 \\ -\cos\gamma & 0 & 0 \end{pmatrix} \\ &= \begin{pmatrix} \sin\beta(\sin\alpha\sin\gamma - \cos\alpha\cos\gamma) & -\cos\beta\sin\gamma & \cos\beta\cos\gamma \\ \cos\alpha\sin\gamma + \sin\alpha\cos\gamma & 0 & 0 \\ \cos\beta(\sin\alpha\sin\gamma - \cos\alpha\cos\gamma) & \sin\beta\sin\gamma & -\sin\beta\cos\gamma \end{pmatrix}. \end{aligned}$$

We now establish the possible SVDs of H. Since H is the product of two orthonormal matrices and another one, we can derive its SVDs from those of that other matrix. This is a skew-symmetric matrix, and all its SVDs can be shown to be of the following form, for some value of ρ (and up to changing signs for entire

columns or rows of the orthogonal matrices involved; this does not matter for our analysis):

$$\begin{pmatrix} 0 & -s_\gamma & c_\gamma \\ s_\gamma & 0 & 0 \\ -c_\gamma & 0 & 0 \end{pmatrix} \overset{SVD}{=} \begin{pmatrix} 0 & 1 & 0 \\ -s_\gamma & 0 & c_\gamma \\ c_\gamma & 0 & s_\gamma \end{pmatrix} \begin{pmatrix} c_\rho & s_\rho & 0 \\ -s_\rho & c_\rho & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$
$$\times \begin{pmatrix} c_\rho & -s_\rho & 0 \\ s_\rho & c_\rho & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} -1 & 0 & 0 \\ 0 & -s_\gamma & c_\gamma \\ 0 & c_\gamma & s_\gamma \end{pmatrix}$$
$$= \begin{pmatrix} -s_\rho & c_\rho & 0 \\ -c_\rho s_\gamma & -s_\rho s_\gamma & c_\gamma \\ c_\rho c_\gamma & s_\rho c_\gamma & s_\gamma \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} -c_\rho & s_\rho s_\gamma & -s_\rho c_\gamma \\ -s_\rho & -c_\rho s_\gamma & c_\rho c_\gamma \\ 0 & c_\gamma & s_\gamma \end{pmatrix},$$

where we use, as above, the shorthand notation $c_\alpha = \cos\alpha$ and $s_\alpha = \sin\alpha$, and analogously for other angles.

Hence, the SVDs of H are parameterized by the same angle ρ, and are of the following form:

$$\underbrace{\mathsf{R}_Y \mathsf{R}_X \begin{pmatrix} -s_\rho & c_\rho & 0 \\ -c_\rho s_\gamma & -s_\rho s_\gamma & c_\gamma \\ c_\rho c_\gamma & s_\rho c_\gamma & s_\gamma \end{pmatrix}}_{\mathsf{U}} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \underbrace{\begin{pmatrix} -c_\rho & s_\rho s_\gamma & -s_\rho c_\gamma \\ -s_\rho & -c_\rho s_\gamma & c_\rho c_\gamma \\ 0 & c_\gamma & s_\gamma \end{pmatrix}}_{\mathsf{V}^{\mathsf{T}}}$$

with U explicitly of the form:

$$\mathsf{U} = \begin{pmatrix} c_\beta & 0 & s_\beta \\ 0 & 1 & 0 \\ -s_\beta & 0 & c_\beta \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & c_\alpha & -s_\alpha \\ 0 & s_\alpha & c_\alpha \end{pmatrix} \begin{pmatrix} -s_\rho & c_\rho & 0 \\ -c_\rho s_\gamma & -s_\rho s_\gamma & c_\gamma \\ c_\rho c_\gamma & s_\rho c_\gamma & s_\gamma \end{pmatrix}.$$

Let us call $X = c_\beta(c_\alpha c_\gamma - s_\alpha s_\gamma)$. From the above SVD, we identify the values used in the calibration equations:

$$a = b,$$

$$U_{31} = s_\beta s_\rho + c_\rho X,$$

$$U_{32} = -s_\beta c_\rho + s_\rho X,$$

$$V_{31} = -s_\rho c_\gamma,$$

$$V_{32} = c_\rho c_\gamma.$$

Note that $X = 0$ is the condition for orthogonal principal epipolar planes (cf. the table in Appendix B). Let us further define:

$$Y = U_{31}^2 V_{31}^2 - U_{32}^2 V_{32}^2,$$

$$Z = U_{32}^2 - U_{31}^2 + V_{32}^2 - V_{31}^2.$$

The quadratic equation can now be written as (we factor out $a = b$):

$$f^4(Y + Z) - f^2(2Y + Z) + Y = 0.$$

Its coefficients vanish all exactly if $Y = Z = 0$. Let us go into details (we use the relationship $s_\rho^4 - c_\rho^4 = s_\rho^2 - c_\rho^2$):

$$Y = U_{31}^2 V_{31}^2 - U_{32}^2 V_{32}^2 = s_\beta c_\gamma^2 (2 c_\rho s_\rho X + s_\beta (s_\rho^2 - c_\rho^2)) = 0, \tag{D.5}$$

$$Z = U_{32}^2 - U_{31}^2 + V_{32}^2 - V_{31}^2 = -4 c_\rho s_\rho s_\beta X + (s_\beta^2 + c_\gamma^2 - X^2)(c_\rho^2 - s_\rho^2) = 0. \tag{D.6}$$

In the following, we consider two questions:

- for which relative camera poses do (D.5) and (D.6) hold *whatever* value ρ has?
- do values for ρ exist for any relative camera pose, such that (D.5) and (D.6) hold?

D.2.1. Relative camera poses for which (D.5) and (D.6) hold for every ρ

Let us consider any value of ρ different from 0. Dividing (D.5) and (D.6)by c_ρ^2 gives:

$$t_\rho^2 (s_\beta^2 c_\gamma^2) + 2 t_\rho (s_\beta c_\gamma^2 X) - s_\beta^2 c_\gamma^2 = 0,$$

$$t_\rho^2 (X^2 - s_\beta^2 - c_\gamma^2) - 4 t_\rho (s_\beta X) + (s_\beta^2 + c_\gamma^2 - X^2) = 0,$$

where $t_\rho = \tan\rho$. The equations hold for every value of ρ exactly if the coefficients of powers of t_ρ all vanish, hence if all the following equations hold (we leave out the ones occurring twice):

$$s_\beta^2 c_\gamma^2 = 0,$$

$$s_\beta c_\gamma^2 X = 0,$$

$$X^2 - s_\beta^2 - c_\gamma^2 = 0,$$

$$s_\beta X = 0.$$

If $s_\beta = 0$, then the third equation holds if $X^2 - c_\gamma^2 = 0$ (we will examine this case just below). If $s_\beta \neq 0$, then the first and fourth equation imply that $c_\gamma = X = 0$. In that case, however, the third equation would not be satisfied. Hence, the only possible case is $s_\beta = X^2 - c_\gamma^2 = 0$. Let us examine it in detail.

The term $X^2 - c_\gamma^2$ can be expanded as follows:

$$\begin{aligned} -c_\gamma^2 + (c_\alpha c_\gamma - s_\alpha s_\gamma)^2 &= c_\gamma^2 (c_\alpha^2 - 1) + s_\gamma^2 s_\alpha^2 - 2 c_\alpha s_\alpha c_\gamma s_\gamma \\ &= (s_\gamma^2 - c_\gamma^2) s_\alpha^2 - 2 c_\alpha s_\alpha c_\gamma s_\gamma \\ &= s_\alpha \Big((s_\gamma^2 - c_\gamma^2) s_\alpha - 2 c_\alpha c_\gamma s_\gamma \Big). \end{aligned}$$

It is equal to zero if $s_\alpha = 0$ or if $(s_\gamma^2 - c_\gamma^2)s_\alpha - 2c_\alpha c_\gamma s_\gamma = 0$. The first case, together with the assumption $s_\beta = 0$, corresponds to the case of parallel optical axes (cf. the table in Appendix B). The second case, corresponds to the equidistance condition. Hence, both cases correspond to generic singular configurations.

We conclude that for $f = \pm 1$, the quadratic calibration vanishes *whichever* SVD one happens to compute (whatever value ρ has) only in the generic singular configurations.

D.2.2. For which relative camera poses can (D.5) and (D.6) hold?

Note that in the following, only generic non-singular configurations are of interest. Let us now consider the question for different cases:

- $\sin\beta = 0$. Eq. (D.5) holds and (D.6) becomes

  $$(c_\gamma^2 - X^2)(c_\rho^2 - s_\rho^2) = 0.$$

  As shown in Section D.2.1, the first possibility, $c_\gamma^2 - X^2 = 0$, corresponds to generic singular configurations, hence is not of interest here. As for the second possibility, $c_\rho^2 - s_\rho^2 = 0$, it tells us that for all relative camera poses with $\sin\beta = 0$, there exist four different values for ρ (separated by 90°), for which the quadratic calibration equation vanishes.
- $\sin\beta \neq 0$, $\cos\gamma = 0$, $\sin^2\beta - \cos^2\beta\sin^2\alpha = 0$. Eq. (D.5) holds and (D.6) becomes

  $$c_\rho s_\rho c_\beta s_\beta s_\alpha = 0.$$

  Hence, for all relative camera poses corresponding to the assumptions made here, there again exist four different values for ρ (separated by 90°), for which the quadratic calibration equation vanishes.
- $\sin\beta \neq 0$, $\cos\gamma = 0$, $\sin^2\beta - \cos^2\beta\sin^2\alpha \neq 0$. Eq. (D.5) holds and (D.6) becomes (the $\pm$ corresponds to $\sin\gamma = \pm 1$)

  $$\pm 4c_\rho s_\rho c_\beta s_\beta s_\alpha + (s_\beta^2 - c_\beta^2 s_\alpha^2)(c_\rho^2 - s_\rho^2) = 0.$$

  Let us first note that for $c_\rho = 0$, this equation cannot hold, due to the assumption that $s_\beta^2 - c_\beta^2 s_\alpha^2 \neq 0$. We may thus divide the equation by c_ρ^2. After some modifications, this leads to:

  $$t_\rho^2(c_\beta^2 s_\alpha^2 - s_\beta^2) \pm 4t_\rho c_\beta s_\beta s_\alpha - (c_\beta^2 s_\alpha^2 - s_\beta^2) = 0.$$

  It is easy to verify that, whatever values α and β have (if compatible with the assumptions made here), there exist exactly two solutions for $t_\rho = \tan\rho$. Hence, for all relative camera poses corresponding to the assumptions made here, there again exist four different values for ρ (separated by 90°), for which the quadratic calibration equation vanishes.

  Hence, for all relative camera poses corresponding to the assumptions made here, there again exist four different values for ρ, for which the quadratic calibration equation vanishes.
- $\sin\beta \neq 0$, $\cos\gamma \neq 0$. For (D.5) to hold, we must have

$$2c_\rho s_\rho X + s_\beta(s_\rho^2 - c_\rho^2) = 0.$$

Multiplying this equation with $2s_\beta$ and adding this to (D.6) gives a *necessary* condition for the vanishing of the quadratic calibration equation

$$(-s_\beta^2 + c_\gamma^2 - X^2)(c_\rho^2 - s_\rho^2) = 0.$$

It is easy to verify that $(-s_\beta^2 + c_\gamma^2 - X^2) = 0$ is equivalent to the condition of equal angles between optical axes and baseline (cf. the table in Appendix B) and that under the assumption $\sin\beta \neq 0$, $\cos\gamma \neq 0$, there exist four different values for ρ for which (D.5) and (D.6) hold.
As for $(-s_\beta^2 + c_\gamma^2 - X^2) \neq 0$, the necessary condition is $c_\rho^2 - s_\rho^2 = 0$. Substituting this into (D.5) and (D.6), leads to the condition $s_\beta c_\gamma^2 X = 0$. Since here we assume that $\sin\beta \neq 0$ and $\cos\gamma \neq 0$, we thus conclude that for $X = 0$ (orthogonal principal epipolar planes, see above), four different values for ρ exist (due to $c_\rho^2 - s_\rho^2 = 0$), for which the quadratic calibration equation vanishes.

D.2.3. Summary

The quadratic equation vanishes of course in generic degenerate conditions. The only other case where it may vanish is when $f = \pm 1$. This may happen because the SVD of the fundamental matrix is ambiguous. For $f = \pm 1$, the coefficients of the quadratic equation may all be zero, depending on which SVD one happens to compute in practice (which angle ρ). This can happen in exactly the following non-generic singular configurations: (i) the optical axes are coplanar, (ii) the principal epipolar planes are mutually orthogonal, or (iii) the angles between the optical axes and the baseline, are equal. In each of these cases, only four different values of ρ (four among the infinitely many ambiguous SVDs) exist for which the quadratic calibration equation vanishes. Hence, the chances for the quadratic equation to vanish in generical non-singular configurations, are small. Nevertheless, instabilities may indeed occur in cases close to $f = \pm 1$, i.e., when working in nearly perfectly standardized coordinates, as illustrated in Section 6.1.4.

Appendix E. Singularities in the case of a unique SVD

We now consider the cases where the semi-calibrated fundamental matrix has a unique SVD (up to switching entire columns or rows or changing signs for entire columns or rows), i.e., different non-zero singular values a and b.

E.1. Quadratic equation

Zeroing the three coefficients of Eq. (4) leads to the following equations:

$$a^2(1 - U_{31}^2 - V_{31}^2 + U_{31}^2 V_{31}^2) = b^2(1 - U_{32}^2 - V_{32}^2 + U_{32}^2 V_{32}^2), \tag{E.1}$$

$$a^2(U_{31}^2 + V_{31}^2 - 2U_{31}^2V_{31}^2) = b^2(U_{32}^2 + V_{32}^2 - 2U_{32}^2V_{32}^2), \tag{E.2}$$

$$a^2U_{31}^2V_{31}^2 = b^2U_{32}^2V_{32}^2. \tag{E.3}$$

Substituting (E.3) into (E.1) and (E.2), we get:

$$a^2(1 - U_{31}^2 - V_{31}^2) = b^2(1 - U_{32}^2 - V_{32}^2), \tag{E.4}$$

$$a^2(U_{31}^2 + V_{31}^2) = b^2(U_{32}^2 + V_{32}^2). \tag{E.5}$$

Adding these two equations together, leads to $a^2 = b^2$. This is in contradiction with our assumptions (unique SVD). We conclude that the quadratic equation is never degenerate when the SVD is unique, i.e., when the cameras are not in an equidistance configuration (including parallel optical axes) and if $f \neq \pm 1$.

Further below, we examine special cases where one of its coefficients vanishes, and especially a case where the quadratic equation becomes linear.

E.2. Linear equations

It is easy to show that *both* linear equations degenerate if any one of the following conditions holds:

$$U_{32} = V_{31} = 0, \tag{E.6}$$

$$U_{32} = V_{32} = 0, \tag{E.7}$$

$$U_{31} = V_{31} = 0, \tag{E.8}$$

$$U_{31} = V_{32} = 0. \tag{E.9}$$

The only other singularities occur, for Eq. (2), if

$$V_{31} = \pm U_{32} \quad \text{and} \quad aU_{31} = \mp bV_{32} \tag{E.10}$$

and, for Eq. (3), if

$$V_{32} = \pm U_{31} \quad \text{and} \quad aV_{31} = \mp bU_{32}. \tag{E.11}$$

Any one of the conditions (E.6), (E.9), (E.10), and (E.11) implies that the optical axes are coplanar (they imply that $H_{33} = 0$, cf. Appendix B). Only the conditions (E.7) and (E.8) may correspond to non-coplanar optical axes. In the following section, we consider the case of coplanar optical axes, and show that this always implies the degeneracy of the linear equations. We then consider the case of non-coplanar axes and examine cases where the linear equations degenerate.

E.3. Coplanar optical axes

As shown in Appendix B, the optical axes are coplanar if $\cos\gamma = 0$ or $\sin\beta = 0$. We examine the two cases in the following.

E.3.1. $\cos\gamma = 0$

This means that the optical center of the second camera is the point $(0, 0, \sin\gamma, 1)$, i.e., it lies on the optical axis of the first camera (the Z-axis). In this case, the first epipole has coordinates $(0, 0, 1)^T$. Since the first epipole is the null-vector of the fundamental matrix H, it is equal (up to sign) to the third *column* $\mathbf{v}_3$ of the matrix V in its SVD. Due to the orthogonality of V, this implies that its third *row* is also given as $(0, 0, \pm 1)$, hence we have: $V_{31} = V_{32} = 0$. Hence, the quadratic equation (4) becomes

$$f^2\left\{f^2\left(a^2(1-U_{31}^2) - b^2(1-U_{32}^2)\right) + \left(a^2U_{31}^2 - b^2U_{32}^2\right)\right\} = 0.$$

The spurious solution of that equation is $f = 0$, and can thus be always rejected, meaning the quadratic equation gives a unique admissible solution.

Consider now the symmetric matrix $\mathsf{H}\mathsf{H}^T = \mathsf{U}\,\mathrm{diag}(a^2, b^2, 0)\mathsf{U}^T$. The columns of U are the eigenvectors of $\mathsf{H}\mathsf{H}^T$. It can be shown that

$$\begin{pmatrix} \sin\alpha \\ \cos\alpha\sin\beta \\ 0 \end{pmatrix}$$

is an eigenvector of $\mathsf{H}\mathsf{H}^T$ to a non-zero eigenvalue (thus, a^2 or b^2). Hence, this vector must be equal (up to scale) to one of the first two columns of U, which means that $U_{31} = 0$ or $U_{32} = 0$. Together with the condition $V_{31} = V_{32} = 0$ shown above, this implies that at least one of (E.6)–(E.9) is true, hence both linear equations, (2) and (3), are degenerate.

E.3.2. $\sin\beta = 0$

In this case, both $\mathsf{H}^T\mathsf{H}$ and $\mathsf{H}\mathsf{H}^T$ have $(1, 0, 0)^T$ as an eigenvector with non-zero eigenvalue. Hence, one of the first two columns of U and one of first two columns of V have this form. It can be shown that if the first column of U has that form, then the second column of V is of the same form, and vice versa. This means that either $U_{31} = V_{32} = 0$ or $U_{32} = V_{31} = 0$, which implies that both linear equations vanish and that the quadratic one becomes

$$f^2\left\{f^2\left(a^2(1-U_{31}^2) - b^2(1-V_{32}^2)\right) + \left(a^2U_{31}^2 - b^2V_{32}^2\right)\right\} = 0 \tag{E.12}$$

if $U_{32} = V_{31} = 0$ or

$$f^2\left\{f^2\left(a^2(1-V_{31}^2) - b^2(1-U_{32}^2)\right) + \left(a^2V_{31}^2 - b^2U_{32}^2\right)\right\} = 0 \tag{E.13}$$

if $U_{31} = V_{32} = 0$. Hence, as in Section E.3.1, the quadratic equation gives a single admissible solution.

E.3.3. Summary

Whenever the optical axes are coplanar, the two linear equations (2) and (3) vanish and the quadratic equation (4) gives in general a single admissible solution. The latter one vanishes completely exactly in the equidistance configuration (including parallel optical axes). Hence all singular cases of the quadratic equation in the coplanar case are generic singular cases, with the exception of the special cases for $f = \pm 1$.

Appendix F. Non-coplanar optical axes

F.1. Linear equations

As Section E.2 shows, the singular cases for non-coplanar optical axes are, for the linear equations, given by Eqs. (E.7) and (E.8):

$$U_{32} = V_{32} = 0,$$

$$U_{31} = V_{31} = 0.$$

F.1.1. First case: $U_{32} = V_{32} = 0$

In the following, the SVD of H is considered. The right null-vector (first epipole) of H is easily seen to be $(0, f\cos\gamma, \sin\gamma)^{\mathsf{T}}$ (cf. Eq. (B.3)). As described in Appendix A, this vector is equal, up to scale, to the third column $\mathbf{v}_3$ of V. Hence we have[6]:

$$\begin{aligned}\mathsf{H} &\sim \begin{pmatrix} (\sin\gamma\sin\alpha - \cos\gamma\cos\alpha)\sin\beta & -\sin\gamma\cos\beta & f\cos\gamma\cos\beta \\ \sin\gamma\cos\alpha + \cos\gamma\sin\alpha & 0 & 0 \\ f(\sin\gamma\sin\alpha - \cos\gamma\cos\alpha)\cos\beta & f\sin\gamma\sin\beta & -f^2\cos\gamma\sin\beta \end{pmatrix} \\ &\sim \underbrace{\begin{pmatrix} U_{11} & U_{12} & U_{13} \\ U_{21} & U_{22} & U_{23} \\ U_{31} & 0 & U_{33} \end{pmatrix}}_{\mathsf{U}} \begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & 0 \end{pmatrix} \underbrace{\begin{pmatrix} V_{11} & V_{21} & V_{31} \\ V_{12} & V_{22} & 0 \\ 0 & f\cos\gamma & \sin\gamma \end{pmatrix}}_{\mathsf{V}^{\mathsf{T}}}. \end{aligned} \tag{F.1}$$

From the orthogonality of rows 2 and 3 of V^{T}, it follows that $V_{22} = 0$ and from this, that $V_{11} = 0$. From $H_{22} = H_{23} = 0$, it also follows that $U_{21} = 0$. Hence (F.1) is rewritten as

$$\begin{aligned} &\begin{pmatrix} (\sin\gamma\sin\alpha - \cos\gamma\cos\alpha)\sin\beta & -\sin\gamma\cos\beta & f\cos\gamma\cos\beta \\ \sin\gamma\cos\alpha + \cos\gamma\sin\alpha & 0 & 0 \\ f(\sin\gamma\sin\alpha - \cos\gamma\cos\alpha)\cos\beta & f\sin\gamma\sin\beta & -f^2\cos\gamma\sin\beta \end{pmatrix} \\ &\quad\sim \begin{pmatrix} U_{11} & U_{12} & U_{13} \\ 0 & U_{22} & U_{23} \\ U_{31} & 0 & U_{33} \end{pmatrix} \begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & V_{21} & V_{31} \\ V_{12} & 0 & 0 \\ 0 & f\cos\gamma & \sin\gamma \end{pmatrix} \\ &\quad= \begin{pmatrix} bU_{12}V_{12} & aU_{11}V_{21} & aU_{11}V_{31} \\ bU_{22}V_{12} & 0 & 0 \\ 0 & aU_{31}V_{21} & aU_{31}V_{31} \end{pmatrix}. \end{aligned}$$

From the coefficient (3,1) of that equation, we derive

[6] Here unitary determinant of the orthogonal matrix V is not imposed.

$$(\sin\gamma\sin\alpha - \cos\gamma\cos\alpha)\cos\beta = 0$$

which is thus a necessary condition for non-coplanar singular cases for the linear equations in the first case. Note that this condition is nothing else than that for mutually orthogonal principal epipolar planes, cf. the table in Appendix B.

In the following it is shown that this condition is also a sufficient one. We do this by giving analytical SVDs[7] for H in the two cases $\cos\beta = 0$ and $\sin\gamma\sin\alpha - \cos\gamma\cos\alpha = 0$. Based on these SVDs, the coefficients of the linear calibration Eqs. (2) and (3) can be computed and it will be seen that they all vanish.

- $\cos\beta = 0$. This implies that $\sin\beta = \pm 1$ and H becomes

$$\mathsf{H} \sim \begin{pmatrix} \pm\sin\gamma\sin\alpha \mp \cos\gamma\cos\alpha & 0 & 0 \\ \sin\gamma\cos\alpha + \cos\gamma\sin\alpha & 0 & 0 \\ 0 & \pm f\sin\gamma & \mp f^2\cos\gamma \end{pmatrix}. \tag{F.2}$$

  Its SVD is given by (using the same shorthand notation as further above)

$$\underbrace{\begin{pmatrix} 0 & \pm s_\alpha s_\gamma \mp c_\alpha c_\gamma & \pm c_\alpha s_\gamma \pm s_\alpha c_\gamma \\ 0 & c_\alpha s_\gamma + s_\alpha c_\gamma & c_\alpha c_\gamma - s_\alpha s_\gamma \\ 1 & 0 & 0 \end{pmatrix}}_{\mathsf{U}} \begin{pmatrix} ft_2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \underbrace{\begin{pmatrix} 0 & \pm s_\gamma & \mp f c_\gamma \\ t_2 & 0 & 0 \\ 0 & f c_\gamma & s_\gamma \end{pmatrix}}_{\mathsf{V}^{\mathrm{T}}}, \tag{F.3}$$

  where $t_2 = \sqrt{f^2\cos^2\gamma + \sin^2\gamma}$. It is easy to verify that (F.3) indeed is an SVD of H: the matrices U and V are orthonogonal and the product of the above expression equals H, as given in (F.2).

  We thus have $U_{32} = V_{32} = 0$, which was already shown in Section E.2 to be a sufficient condition for degeneracy of the linear equations.

- $\sin\gamma\sin\alpha - \cos\gamma\cos\alpha = 0$. Note that in this case, we have $\cos\gamma \neq 0$ and $\sin\alpha \neq 0$: the condition $\cos\gamma = 0$ can be excluded since it would imply coplanar optical axes (cf. Appendix B). Concerning $\sin\alpha \neq 0$: if $\sin\alpha = 0$, then $\cos\alpha = \pm 1$ and $\sin\gamma\sin\alpha - \cos\gamma\cos\alpha = \mp\cos\gamma \neq 0$, which is contradictory to our assumption here. We may thus put:

$$\sin\gamma = \frac{\cos\alpha}{\sin\alpha}\cos\gamma.$$

  H becomes

$$\mathsf{H} \sim \begin{pmatrix} 0 & -s_\gamma c_\beta & f c_\gamma c_\beta \\ s_\gamma c_\alpha + c_\gamma s_\alpha & 0 & 0 \\ 0 & f s_\gamma s_\beta & -f^2 c_\gamma s_\beta \end{pmatrix} \sim \begin{pmatrix} 0 & -c_\alpha c_\beta & f s_\alpha c_\beta \\ 1 & 0 & 0 \\ 0 & f c_\alpha s_\beta & -f^2 s_\alpha s_\beta \end{pmatrix}.$$

  An SVD for H is given by:

[7] The analytical SVDs in this section are given up to possible switching of columns of the involved matrices and, for easier expressions, up to scale for the orthogonal matrices U and V.

$$\begin{pmatrix} \cos\beta & 0 & f\sin\beta \\ 0 & t_2 & 0 \\ -f\sin\beta & 0 & \cos\beta \end{pmatrix} \begin{pmatrix} t_1 t_2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & -\cos\alpha & f\sin\alpha \\ t_1 & 0 & 0 \\ 0 & f\sin\alpha & \cos\alpha \end{pmatrix} \tag{F.4}$$

with $t_1 = \sqrt{f^2\sin^2\alpha + \cos^2\alpha}$ and $t_2 = \sqrt{f^2\sin^2\beta + \cos^2\beta}$. Again, we have $U_{32} = V_{32} = 0$, meaning that the linear calibration equations degenerate.

F.1.2. Second case: $U_{31} = V_{31} = 0$

The analysis can be done analogously as above, leading to the same conclusions (the SVDs are the same, up to swapping of the singular values and corresponding columns of U and V). Which one of the cases $U_{32} = V_{32} = 0$ or $U_{31} = V_{31} = 0$ occurs in practice, depends on which one of the singular values is larger.

F.2. Quadratic equation

If we exclude $f = \pm 1$, then non-coplanar optical axes imply that $a \neq b$ (follows from Section D.1) and hence the quadratic equation is non-degenerate. We now consider what happens in the cases where the linear equations degenerate: $\cos\beta = 0$ or $\sin\gamma\sin\alpha - \cos\gamma\cos\alpha = 0$, cf. Section F.1.1.

- $\cos\beta = 0$. The SVD of H in this case is given in Eq. (F.3). We substitute its coefficients in the quadratic equation, and get

  $$-g^4 + g^2 f^2 \sin^2\gamma + f^4\cos^2\gamma = 0,$$

  where f is the true focal length and g the estimated one. Its two solutions are $g^2 = f^2$ and $g^2 = -f^2\cos^2\gamma$. Being always negative (or zero), the second solution can be ruled out, which means that the quadratic equation gives a unique feasible solution here.
- $\sin\gamma\sin\alpha - \cos\gamma\cos\alpha = 0$. Substituting the coefficients of the SVD of H, given in Eq. (F.4), in the quadratic equation, we get

  $$g^4(\cos^2\alpha\,\cos^2\beta - 1) + g^2 f^2(\cos^2\alpha\,\sin^2\beta + \sin^2\alpha\,\cos^2\beta) + f^4\sin^2\alpha\,\sin^2\beta = 0.$$

  Besides f^2, g^2 has the following solution:

  $$-\frac{\sin^2\alpha\,\sin^2\beta}{\sin^2\beta + \sin^2\alpha\,\cos^2\beta}$$

  which is always non-positive.[8] Hence, the quadratic equation has again a unique admissible solution.

[8] Note that the denominator is assured not to be zero, since we exclude $\sin\beta = 0$ (we consider non-coplanar optical axes) and $\sin\alpha = 0$ (cf. Section F.1.1).

F.3. Summary

If the optical axes are non-coplanar, then the quadratic equation is never degenerate (with the exception of the special case $f = \pm 1$ discussed in Section D.2). In addition, in all cases where the linear equations vanish, the spurious solution of the quadratic equation can be ruled out due to being non-positive.

References

[1] S. Bougnoux, From projective to Euclidean space under any practical situation, a criticism of self-calibration, in: Proc. 6th Internat. Conf. on Computer Vision, Bombay, India, January, 1998, pp. 790–796.

[2] M. Brooks, L. De Agapito, D. Huynh, L. Baumela, Towards robust metric reconstruction via a dynamic uncalibrated stereo head, Image Vision Comput. 16 (14) (1998) 989–1002.

[3] O. Faugeras, O. Luong, S. Maybank, Camera self-calibration: theory and experiments, in: Proc. European Conf. on Computer VisionLNCS, vol. 588, Springer-Verlag, Berlin, 1992, pp. 321–334.

[4] O. Faugeras, Q.-T. Luong, The Geometry of Multiple Images, MIT Press, Cambridge, 2001.

[5] G.H. Golub, C.F. van Loan, Matrix computation, The Johns Hopkins University Press, 1989.

[6] R. Hartley, Estimation of relative camera positions for uncalibrated cameras, in: Proc. Eur. Conf. on Computer Vision, 1992, pp. 579–587.

[7] R. Hartley, In defence of the 8-point algorithm, in: Proc. 5th Internat. Conf. on Computer Vision, Boston, USA, 1995, pp. 1064–1070.

[8] R. Hartley, Kruppa's equations derived from the fundamental matrix, IEEE Trans. Pattern Anal. Mach. Intell. 19 (2) (1997) 133–135.

[9] R. Hartley, P. Sturm, Triangulation, Comput. Vision Image Understand. 68 (2) (1997) 146–157.

[10] R. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, Cambridge University Press, Cambridge, 2000.

[11] A. Heyden, K. Åström, Euclidean reconstruction from image sequences with varying and unknown focal length and principal point, in: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 1997, pp. 438–443.

[12] F. Kahl, B. Triggs, Critical motions in Euclidean structure from motion, in: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, vol. 2, 1999, pp. 366–372.

[13] F. Kahl, B. Triggs, K. Astrom, Critical motions for auto-calibration when some intrinsic parameters can vary, J. Math. Imaging Vision 13 (2) (2000) 131–146.

[14] G.N. Newsam, D.Q. Huynh, M.J. Brooks, H.P. Pan, Recovering unknown focal lengths in self-calibration: an essentially linear algorithm and degenerate configurations, ISPRS-Congress XXXI (B3) (1996) 575–580.

[15] M. Pollefeys, R. Koch, L. Van Gool, Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters, in: Proc. 6th Internat. Conf. on Computer Vision, Bombay, India, 1998, pp. 90–96.

[16] M. Pollefeys, Self-calibration and metric 3D reconstruction from uncalibrated image sequences. PhD Thesis, Katholieke Universiteit Leuven, Belgium, 1999.

[17] P. Sturm, Critical motion sequences for monocular self-calibration and uncalibrated Euclidean reconstruction, in: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, Puerto Rico, June, 1997, pp. 1100-1105.

[18] P. Sturm, A case against Kruppa's equations for camera self-calibration, IEEE Trans. Pattern Anal. Mach. Intell. 22 (10) (2000) 1199–1204.

[19] P. Sturm, On focal length calibration from two views, in: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, Kauai, 2001, pp. 145–150.

[20] P. Sturm, Critical motion sequences for the self-calibration of cameras and stereo systems with variable focal length, Image Vision Comput. 20 (5–6) (2002) 415–426.

[21] Z. Zhang, R. Deriche, O. Faugeras, Q. Luong, A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry, Artif. Intell. 78 (1995) 87–119.
[22] C. Zeller, O. Faugeras, Camera self-calibration from video sequences: the Kruppa equations revisited. Rapport de Recherche 2793, INRIA, France, 1996.

IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 22, NO. 10, OCTOBER 2000 1179

Self-Calibration of a 1D Projective Camera and Its Application to the Self-Calibration of a 2D Projective Camera

Olivier Faugeras, Long Quan, and Peter Strum

Abstract—We introduce the concept of self-calibration of a 1D projective camera from point correspondences, and describe a method for uniquely determining the two internal parameters of a 1D camera, based on the trifocal tensor of three 1D images. The method requires the estimation of the trifocal tensor which can be achieved linearly with no approximation unlike the trifocal tensor of 2D images and solving for the roots of a cubic polynomial in one variable. Interestingly enough, we prove that a 2D camera undergoing planar motion reduces to a 1D camera. From this observation, we deduce a new method for self-calibrating a 2D camera using planar motions. Both the self-calibration method for a 1D camera and its applications for 2D camera calibration are demonstrated on real image sequences.

Index Terms—Vision geometry, camera model, self-calibration, planar motion, 1D camera.

1 Introduction

A CCD camera is commonly modeled as a 2D projective device that projects a point in $\mathcal{P}^3$ (the projective space of dimension 3) to a point in $\mathcal{P}^2$. By analogy, we can consider what we call a 1D projective camera which projects a point in $\mathcal{P}^2$ to a point in $\mathcal{P}^1$. This 1D projective camera may seem very abstract, but many imaging systems using laser beams, infrared, or ultrasound acting only on a source plane can be modeled this way. What is less obvious, but more interesting for our purpose, is that in some situations, the usual 2D camera model is also closely related to this 1D camera model. First, one example might be the case of the 2D affine camera model operating on line segments: The direction vectors of lines in 3D space and in the image correspond to each other via this 1D projective camera model [20]. Other cases will be discussed later.

In this paper, we first introduce the concept of self-calibration of a 1D projective camera by analogy to that of a 2D projective camera which is a very active topic [17], [12], [7], [13], [1], [27], [19] since the pioneering work of [18]. It turns out that the theory of self-calibration of 1D camera is considerably simpler than the corresponding one in 2D. It is essentially determined in a unique way by a linear algorithm using the trifocal tensor of 1D cameras. After establishing this result, we further investigate the relationship between the usual 2D camera and the 1D camera. It turns out that a 2D camera undergoing planar motion can be reduced to a 1D camera on the trifocal plane of the 2D cameras. This remarkable relationship allows us to calibrate a real 2D projective camera using the theory of self-calibration of a 1D camera. The advantage of doing so is evident. Instead of solving complicated Kruppa equations for 2D camera self-calibration, an exact linear algorithm can be used for 1D camera self-calibration. The only constraint is that the motion of the 2D camera should be restricted to planar motions. The other applications, including 2D affine camera calibration, are also briefly discussed. Part of this work was also presented in [10].

The paper is organized as follows: In Section 2, we review the 1D projective camera model and its trifocal tensor. Then, an efficient estimation of the trifocal tensor is discussed in Section 3. The theory of self-calibration of a 1D camera is introduced and developed in Section 4. After pointing out some direct applications of the theory in Section 5, we develop in Section 6 a new method of 2D camera self-calibration by converting a 2D camera undergoing planar motions into a 1D camera. Experimental results on both simulated and real image sequences are presented in Section 7. Finally, some concluding remarks and future directions are given in Section 8.

Throughout the paper, vectors are denoted in lower case boldface, matrices and tensors in upper case boldface. Some basic tensor notation is used: covariant indices as subscripts, contravariant indices as superscripts and the implicit summation convention.

2 1D Projective Camera and Its Trifocal Tensor

We will first review the one-dimensional camera which was abstracted from the study of the geometry of lines under affine cameras [20]. Also, we can introduce it directly by analogy to a 2D projective camera.

A 1D projective camera projects a point $\mathbf{x} = (x^1, x^2, x^3)^T$ in $\mathcal{P}^2$ (projective plane) to a point $\mathbf{u} = (u^1, u^2)^T$ in $\mathcal{P}^1$ (projective line). This projection may be described by a 2×3 matrix $\mathbf{M}$ as $\lambda \mathbf{u} = \mathbf{M}_{2\times 3}\mathbf{x}$. Now, we examine the geometric constraints available for points seen in multiple views similar to the 2D camera case [22], [23], [13], [26], [9]. There is a constraint only in the case of three views, as there is no constraint for two views (two projective lines always intersect in a point in a projective plane).

Let three views of the point $\mathbf{x}$ be given as follows:

$$\begin{cases} \lambda \mathbf{u} &= \mathbf{M}\mathbf{x}, \\ \lambda' \mathbf{u}' &= \mathbf{M}'\mathbf{x}, \\ \lambda'' \mathbf{u}'' &= \mathbf{M}''\mathbf{x}. \end{cases} \tag{1}$$

These can be rewritten in matrix form as

$$\begin{pmatrix} \mathbf{M} & \mathbf{u} & \mathbf{0} & \mathbf{0} \\ \mathbf{M}' & \mathbf{0} & \mathbf{u}' & \mathbf{0} \\ \mathbf{M}'' & \mathbf{0} & \mathbf{0} & \mathbf{u}'' \end{pmatrix} (\mathbf{x}, -\lambda, -\lambda', -\lambda'')^T = \mathbf{0}.$$

The vector $(\mathbf{x}, -\lambda, -\lambda', -\lambda'')^T$ cannot be zero, so

$$\begin{vmatrix} \mathbf{M} & \mathbf{u} & \mathbf{0} & \mathbf{0} \\ \mathbf{M}' & \mathbf{0} & \mathbf{u}' & \mathbf{0} \\ \mathbf{M}'' & \mathbf{0} & \mathbf{0} & \mathbf{u}'' \end{vmatrix} = 0. \tag{2}$$

The expansion of this determinant produces a trifocal constraint for the three views

$$T_{ijk} u^i u'^j u''^k = 0, \tag{3}$$

where T_{ijk} is a $2 \times 2 \times 2$ homogeneous tensor whose components T_{ijk} are 3×3 minors (involving all three views) of the following 6×3 joint projection matrix:

$$\begin{pmatrix} \mathbf{M} \\ \mathbf{M}' \\ \mathbf{M}'' \end{pmatrix} = (1, 2, 1', 2', 1'', 2'')^T.$$

The components of the tensor can be made explicit as $T_{ijk} = [\vec{i} \vec{j}' \vec{k}'']$, for $i, j', k'' = 1, 2$, where the bracket $[ij'k'']$ denotes the 3×3 minor of ith, jth, and kth row vector of the above joint

- *O. Faugeras is with INRIA, 2004, route des Lucioles-B.P. 93, 06902 Sophia Antipolis Cedex, France. E-mail: Oliver.Faugeras@sophia.inria.fr.*
- *L. Quan and P. Sturm are with CNRS-INRIA, ZIRST-655 avenue de l'Europe, 38330 Montbonnot, France. E-mail: {Long.Quan, Peter.Sturm}@inrialpes.fr.*

Manuscript received 12 Mar. 1998; revised 01 Mar. 2000; accepted 28 June 2000.
Recommended for acceptance by S. Sarkar.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 107800.

projection matrix and bar $''-''$ in $\bar{i}$, $\bar{j}$, and $\bar{k}$ denotes the mapping $(1,2)\mapsto(2,-1)$. It can be easily seen that any constraint obtained by adding further views reduces to a trilinearity. This proves the uniqueness of the trilinear constraint. Moreover, the $2\times 2\times 2$ homogeneous tensor has $7 = 2\times 2\times 2 - 1$ d.o.f., so it is a minimal parametrization of three views in the uncalibrated setting since three views have exactly $3\times(2\times 3-1)-(3\times 3-1)=7$ d.o.f., up to a projective transformation in $\mathcal{P}^2$.

This result for the one-dimensional projective camera is very interesting. The trifocal tensor encapsulates exactly the information needed for projective reconstruction in $\mathcal{P}^2$. Namely, it is the unique matching constraint, it minimally parametrizes the three views and it can be estimated linearly. Contrast this to the 2D image case in which the multilinear constraints are algebraically redundant and the linear estimation is only an approximation based on over-parametrization.

3 Estimation of the Trifocal Tensor of a 1D Camera

Each point correspondence in three views $\mathbf{u}\leftrightarrow\mathbf{u}'\leftrightarrow\mathbf{u}''$ yields one homogeneous linear equation for the eight tensor components T_{ijk} for $i,j,k=1,2$:

$$(u^1u'^1u''^1, u^1u'^1u''^2, u^1u'^2u''^1, u^1u'^2u''^2,$$
$$u^2u'^1u''^1, u^2u'^1u''^2, u^2u'^2u''^1, u^2u'^2u''^2)\mathbf{t}=0,$$

where $\mathbf{t}=(T_{111},T_{112},T_{121},T_{122},T_{211},T_{212},T_{221},T_{222})^T$. With at least seven point correspondences, we can solve for the tensor components linearly.

A careful normalization of the measurement matrix is nevertheless necessary just like that stressed in [11] for the linear estimation of the fundamental matrix. The points in each image are first translated so that their centroid is the origin of the image coordinates, then scaled so that the average distance of the points from the origin is 1. This is achieved by an affine transformation of the image coordinates in each image: $\bar{\mathbf{u}}=\mathbf{A}\mathbf{u}$, $\bar{\mathbf{u}}'=\mathbf{B}\mathbf{u}'$, and $\bar{\mathbf{u}}''=\mathbf{C}\mathbf{u}''$. With these normalized image coordinates, the normalized tensor components $\bar{T}_{ijk}$ are linearly estimated by SVD from $\bar{T}_{ijk}\bar{u}^i\bar{u}'^j\bar{u}''^k=0$. The original tensor components T_{ijk} are recovered by undoing the normalization transformations: $T_{abc}=\bar{T}_{ijk}A^i_aB^j_bC^k_c$.

4 Self-Calibration of a 1D Camera from Three Views

The concept of camera self-calibration using only point correspondences became popular in the computer vision community following Maybank and Faugeras [18], by solving the so-called Kruppa equations. The basic assumption is that the internal parameters of the camera remain invariant. In the case of the 2D projective camera, the internal calibration (the determination of the five internal parameters) is equivalent to the determination of the image ω of the absolute conic in $\mathcal{P}^3$.

4.1 The Internal Parameters of a 1D Camera and the Circular Points

For a 1D camera represented by a 2×3 projection matrix $\mathbf{M}_{2\times 3}$, this projection matrix can always be decomposed into

$$\mathbf{M}_{2\times 3}=\mathbf{K}_{2\times 2}(\mathbf{R}_{2\times 2}\ \ \mathbf{t}_{2\times 1}),$$

where

$$\mathbf{K}_{2\times 2}=\begin{pmatrix}\alpha & u_0\\ 0 & 1\end{pmatrix}$$

represents the two internal parameters: α, the focal length in pixels and u_0, the position of the principal point; the external parameters are represented by a 2×2 rotation matrix $\mathbf{R}_{2\times 2}$,

$$\mathbf{R}_{2\times 2}=\begin{pmatrix}\cos\theta & \sin\theta\\ -\sin\theta & \cos\theta\end{pmatrix}$$

and the translation vector $\mathbf{t}_{2\times 1}$.

The object space for a 1D camera is a projective plane, and any rigid motion on the plane leaves the two circular points I and J invariant (a pair of complex conjugate points on the line at infinity of the plane). Similarly to the 2D camera case where the knowledge of the internal parameters is equivalent to that of the image of the absolute conic, the knowledge of the internal parameters of a 1D camera is equivalent to that of the image points $\mathbf{i}$ and $\mathbf{j}$ of the circular points in $\mathcal{P}^2$.

The relationship between the image of the circular points and the internal parameters of the 1D camera follows directly by projecting one of the circular points $I=(i,1,0)^T$, where $i=\sqrt{-1}$, by the camera $\mathbf{M}_{2\times 3}$:

$$\lambda\mathbf{i}=e^{-i\theta}\begin{pmatrix}u_0+i\alpha\\ 1\end{pmatrix}=\begin{pmatrix}\alpha & u_0\\ 0 & 1\end{pmatrix}(\,\mathbf{R}_{2\times 2}\quad\mathbf{t}\,)\begin{pmatrix}i\\ 1\\ 0\end{pmatrix}.$$

It clearly appears that the real part of the ratio of the projective coordinates of the image of the circular point $\mathbf{i}$ is the position of the principal point u_0 and the imaginary part is the focal length α.

4.2 Determination of the Images of the Circular Points

Our next task is to locate the circular points in the images. Let us consider one of the circular points, say I. This circular point is projected onto $\mathbf{i}$, $\mathbf{i}'$, and $\mathbf{i}''$ in the three views. As they should be invariant because of our assumption that the internal parameters of the camera are constant, we have:

$$\lambda\mathbf{i}=\lambda'\mathbf{i}'=\lambda''\mathbf{i}''\equiv\mathbf{u},$$

where $\mathbf{u}=(u^1,u^2)^T=\rho(a+ib,1)^T$ for $\lambda,\lambda',\lambda'',\rho\in\mathcal{C}$.

The triplet of corresponding points $\mathbf{i}\leftrightarrow\mathbf{i}'\leftrightarrow\mathbf{i}''$ satisfies the trilinear constraint (3) as all corresponding points do, therefore, $T_{ijk}i^ii'^ji''^k=0$, i.e., $T_{ijk}u^iu^ju^k=0$. This yields the following cubic equation in the unknown $x=u^1/u^2$:

$$T_{111}x^3+(T_{211}+T_{112}+T_{121})x^2+(T_{212}+T_{221}+T_{122})x+T_{222}=0. \tag{4}$$

A cubic polynomial in one unknown with real coefficients has in general either three real roots or one real root and a pair of complex conjugate roots. The latter case of one real and a pair of complex conjugates is obviously the case of interest here. In fact, (4) characterizes all the points of the projective plane which have the same coordinates in three views. This is reminiscent of the 3D case where one is interested in the locus of all points in space that project onto the same point in two views (see Section 6). The result that we have just obtained is that, in the case where the internal parameters of the camera are constant, there are in general three such points: the two circular points which are complex conjugate, and a real point with the following geometric interpretation.

First, consider the case of two views and let us ask the question, what is the set of points such that their images in the two views are the same? This set of points can be called the 2D horopter (h) of the two 1D views. Since the two cameras have the same internal parameters, we can ignore them and assume that we work with the calibrated pixel coordinates. In that case, a camera can be identified to an orthonormal system of coordinates centered at the optical center, one axis is parallel to the retina, the other one is the optical axis. The two views correspond to each other via a rotation

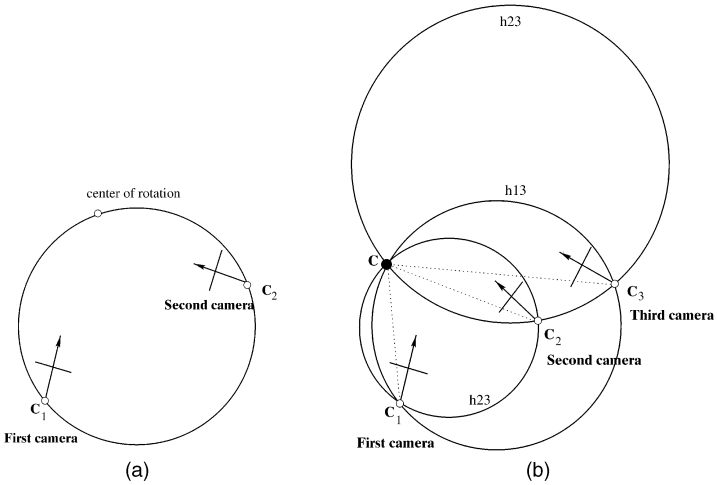


Fig. 1. (a) The two-dimensional horopter which is set of points having the same coordinates in the two views (see text). (b) The geometric interpretation of the real point C which has the same images in all three views (see text).

followed by a translation. This can always be described in general as a pure rotation around a point A whose coordinates can easily be computed from the cameras' projection matrices. A simple computation then shows that the horopter (h) is the circle going through the two optical centers and A, as illustrated in Fig. 1a. In fact, it is the circle minus the two optical centers. Note that since all circles go through the circular points (hence their name), they also belong to the horopter curve, as expected.

In the case of three views, the real point, when it exists, must be at the intersection of the horopter (h_{12}) of the first two views and the horopter (h_{23}) of the last two views. The first one is a circle going through the optical centers C_1 and C_2, the second one is a circle going through the optical centers C_2 and C_3. Those two circles intersect in general at a second point C which is the real point we were discussing, and the third circle (h_{13}) corresponding to the first and third views must also go through the real point C, see Fig. 1b.

We have therefore established the interesting result that the internal parameters of a 1D camera can be uniquely determined through at least seven point correspondences in three views: The seven points yield the trifocal tensor and (4) yields the internal parameters.

5 Applications

The theory of self-calibration of 1D cameras is considerably simpler than the corresponding one in 2D [18] and can be directly used whenever a 1D projective camera model occurs; for instance, self-calibration of some active systems using laser beams, infrared [3], or ultrasound whose imaging system is basically reduced to a 1D camera on the source plane; and partial/full self-calibration of 2D projective cameras using planar motions.

The first type of applications is straightforward. The interesting observation is that the 1D calibration procedure can also be used for self-calibrating a real 2D projective camera if the camera motion is restricted to planar motions. This is discussed in detail in the remainder of this paper.

6 Calibrating a 2D Projective Camera using Planar Motions

A planar motion consists of a translation in a plane and a rotation about an axis perpendicular to that plane. Planar motion is often performed by a vehicle moving on the ground, and has been used for camera self-calibration by Beardsley and Zisserman [4] and by Armstrong et al. [1].

Recall that the self-calibration of a 2D projective camera [8], [18] consists of determining the five unchanging internal parameters of a 2D camera, represented by a 3×3 upper triangular matrix

$$\mathbf{K} = \begin{pmatrix} \alpha_u & s & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{pmatrix}.$$

This is mathematically equivalent to the determination of the image of the absolute conic ω, which is a plane conic described by $\mathbf{x}^T(\mathbf{K}^{-1})^T(\mathbf{K}^{-1})\mathbf{x} = 0$ for image points $\mathbf{x}$. Given the image of the absolute conic $\mathbf{x}^T\mathbf{C}\mathbf{x} = 0$, the calibration matrix $\mathbf{K}$ can be found from $\mathbf{C}$ using the Choleski decomposition.

6.1 Converting 2D images into 1D images

For a given planar motion, the trifocal plane—the plane through the camera centers—of the camera is coincident with the motion plane as the camera is moving on it. Therefore, the image location of the motion plane is the same as the trifocal line which could be determined from fundamental matrices. The determination of the image location of the motion plane has been reported in [1], [4]. Obviously, if restricting the working space to the trifocal plane, we have a perfect 1D projective camera model which projects the points of the trifocal plane onto the trifocal line in the 2D image plane, as the trifocal line is the image of the trifocal plane. In practice, very few or no point at all really lies on the trifocal plane.

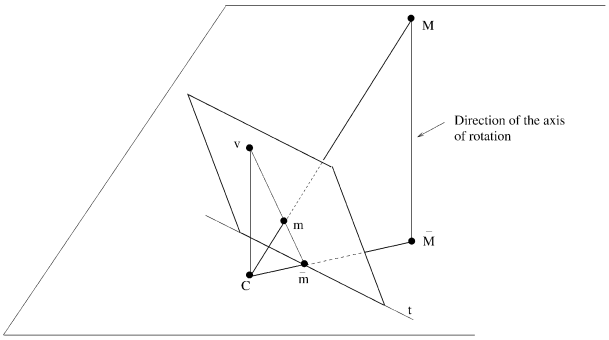


Fig. 2. Creating a 1D image from a 2D image from the vanishing point of the rotation axis and the trifocal line (see text).

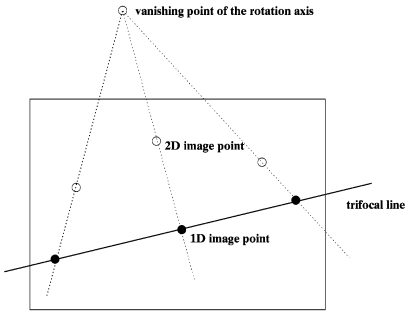


Fig. 3. Converting 2D image points into 1D image points in the image plane is equivalent to a projective projection from the image plane to the trifocal line with the vanishing point of the rotation axis as the projection center.

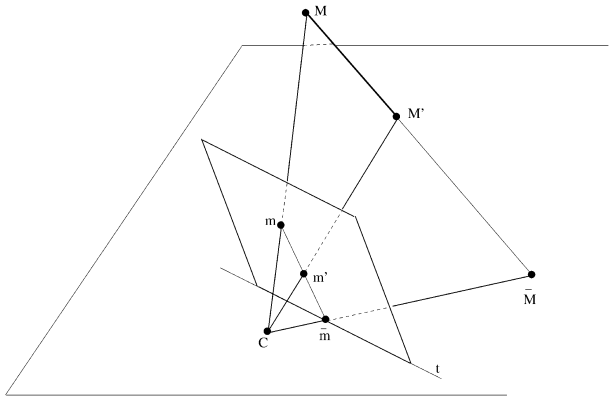


Fig. 4. Creating a 1D image from any pair of points or any line segment (see text).

However, we may virtually project any 3D point onto the trifocal plane, therefore, here comes the central idea of our method: *the 2D images of a camera undergoing planar motion reduce to 1D images by projecting the 2D image points onto the trifocal line*. This can be achieved in at least two ways.

First, if the vanishing point $\mathbf{v}$ of the rotation axis is well-defined. This vanishing point of the rotation axis being the direction perpendicular to the common plane of motion can be determined from fundamental matrices by noticing that the image of the horopter for planar motion degenerates to two lines [1], one of which goes through the vanishing point of the rotation axis; we may refer to [1] for more details.

Given a 3D point M with image m, we mentally project it to $\bar{M}$ in the plane of motion, the projection being parallel to the direction of rotation. The image $\bar{m}$ of this virtual point can be obtained in the image as the intersection of the line through v and m with the trifocal line t. Since the vanishing point v of the rotation axis and the trifocal line t are well defined, this construction, illustrated in Fig. 2, is a well-defined geometric operation.

Note that this is also a perspective projection from $\mathcal{P}^2$ (image plane) to $\mathcal{P}^1$ (trifocal line): $m \mapsto \bar{m}$ as illustrated in Fig. 3.

Alternatively, if the vanishing point is not available, we can nonetheless create the virtual points in the trifocal plane. Given two points M and M' with images m and m', the line (M, M') intersects the plane of motion in $\bar{M}$. The image $\bar{m}$ of this virtual point can be obtained in the image as the intersection of the line (m, m') with the trifocal line t, see Fig. 4.

Another important consequence of this construction is that *2D image line segments can also be converted into 1D image points!* The construction is even simpler, as the resulting 1D image point is just the intersection of the line segment with the trifocal line.

6.1.1 1D Self-Calibration

At this point, we have obtained the interesting result that a 1D projective camera model is obtained by considering only the reprojected points on the trifocal line for a planar motion. The 1D self-calibration method described in Section 4 will allow us to locate the image of the circular points common to all planes parallel to the motion plane.

6.1.2 Estimation of the Image of the Absolute Conic for the 2D Camera

Each planar motion generally gives us two points on the absolute conic, together with the vanishing point of the rotation axes as the pole of the trifocal line w.r.t. the absolute conic. The pole/polar relation between the vanishing point of the rotation axes and the trifocal line was introduced in [1]. As a whole, this provides four constraints on the absolute conic. Since a conic has five d.o.f., at least two different planar motions, yielding eight linear constraints on the absolute conic, will be sufficient to determine the full set of five internal parameters of a general 2D camera by fitting a general conic of the form $\mathbf{x}^T\mathbf{C}\mathbf{x} = au^2 + bv^2 + cuv + du + ev + f = 0$. If we assume a four-parameter model for camera calibration with no image skew (i.e., $s = 0$), one planar motion yielding four constraints is generally sufficient to determine the four internal parameters of the 2D camera. However, this is not true for some very common planar motions such as purely horizontal or vertical motions with the image plane perpendicular to the motion plane. It can be easily proven that there are only three instead of four independent constraints on the absolute conic in these configurations. We need at least two different planar motions for determining the four internal parameters.

Also, this suggests that even if the planar motion is not purely horizontal or vertical, but close, the vanishing point of the rotation axes only constrains loosely the absolute conic. Using only the circular points located on the absolute conic is preferable and numerically stable, but we may need at least three planar motions to determine the five internal parameters of the 2D camera. Note that the numerical instability of the vanishing point for a nearly horizontal trifocal line was already reported by Armstrong in [2].

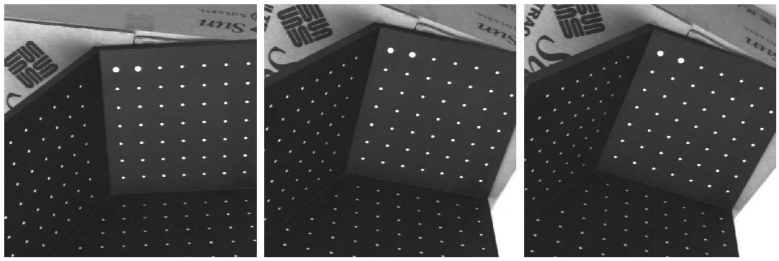


Fig. 5. Three images of the first planar motion.

TABLE 1
Estimated Positions of the Images of the Circular Points by Self-Calibration with Different Triplets of Images of the First Sequence

Image triplet	Fixed point	Circular points by self-calibration	Circular points by calibration
$(16, 19, 22)$	493.7	$290.7 \pm i2779.1$	$310.3 \pm i2650.3$
$(16, 20, 22)$	421.8	$250.1 \pm i2146.3$	$273.9 \pm i2153.5$
$(17, 19, 21)$	533.1	$291.3 \pm i2932.4$	$241.3 \pm i2823.1$
$(16, 18, 20)$	617.8	$238.5 \pm i2597.6$	$238.1 \pm i2791.5$
$(18, 20, 22)$	368.3	$230.6 \pm i2208.2$	$272.1 \pm i2126.2$

The quantities are expressed in the first image pixel coordinate system. The location of circular points by calibration vary as the trifocal line location varies.

TABLE 2
Estimated Positions of the Image of Circular Points with Different Triplets of Images

Image triplet	Circular points	Fixed point
$(16, 18, 20)$	$245.5 \pm i2490.5$	590.0
$(18, 20, 22)$	$221.4 \pm i2717.8$	384.4
$(16, 20, 22)$	$236.2 \pm i2617.3$	452.9
$(16, 19, 22)$	$240.0 \pm i2693.4$	488.0
$(17, 19, 21)$	$304.7 \pm i2722.7$	516.6
known position by calibration	$262.1 \pm i2590.6$	

These quantities vary because the 1D trifocal tensor varies. The trifocal line and the vanishing point of the rotation axes are estimated using seven images of the sequence instead of the minimum of three images.

TABLE 3
Estimated Position of the Image of Circular Points with One Triplet of the Second Image Sequence

Image triplet	Fixed point	Circular points by self-calibration	Circular points by calibration
$(8, 11, 15)$	927.2	$269.7 + i1875.5$	$276.5 + i1540.1$

Obviously, if we work with a three-parameter model with known aspect ratio and without skew, one planar motion is sufficient [1].

As we have mentioned at the beginning of this section, the method described in this section is related to the work of Armstrong et al. [1], but there are some important differences which we now explain.

1. First, our approach gives an elegant insight of the intricate relationship between 2D and 1D cameras for a special kind of motion, called planar motion.
2. Second, it allows us to only use the fundamental matrices of the 2D images and the trifocal tensor of 1D images to self-calibrate the camera instead of the trifocal tensor of 2D images. It is now well-known that fundamental matrices can be very efficiently and robustly estimated [29], [25]. The same is true of the estimation of the 1D trifocal tensor [20] which is a *linear* process. Armstrong et al., on the other hand, use the trifocal tensor of 2D images which, so far, has been hard to estimate due to complicated algebraic constraints to our knowledge. Also, the trifocal tensor of 2D images takes a special form in the planar motion case [1] and the new constraints have to be included in the estimation process.

   It may be worth mentioning that in the case of interest here, planar motion of the cameras, the Kruppa equations become degenerate [28] and the recovery of the internal parameters is impossible from the Kruppa equations. Since it is known that the trifocal tensor of 2D images is algebraically equivalent to the three fundamental matrices plus the restriction of the trifocal tensor to the trifocal plane [14], [15], [9], our method can be seen as an inexpensive way of estimating the full trifocal tensor of 2D images: First, estimate the three fundamental matrices (nonlinear but simple and well-understood), then estimate the trifocal tensor in the trifocal plane (linear).

Although it looks superficially that both the 1D and 2D trifocal tensors can be estimated linearly with at least seven image correspondences, this is misleading since the estimation of the 1D trifocal tensor is exactly linear for seven d.o.f., whereas the linear estimation of the 2D trifocal tensor is only a rough approximation based on a set of 26 auxiliary parameters for its 18 d.o.f. and obtained by neglecting eight complicated algebraic constraints.

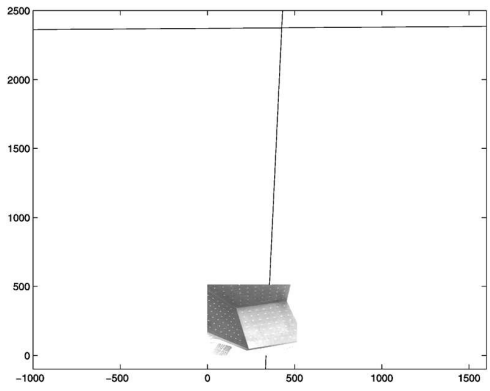


Fig. 6. The image of the motion planes of the two planar motions.

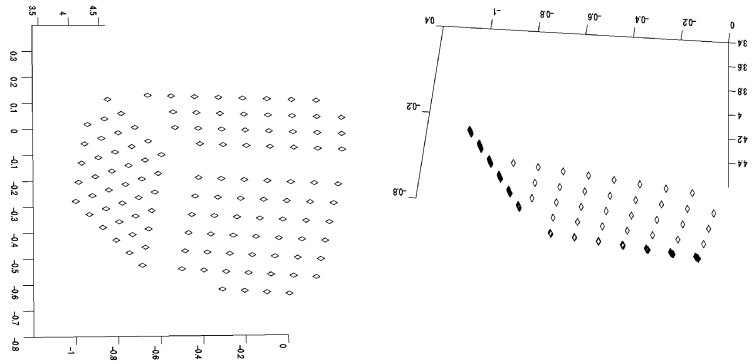


Fig. 7. Two views of the resulting 3D reconstruction by self-calibration.

3. Third, but this is a minor point, our method may not require the estimation of the vanishing point of the rotation axes.

7 Experimental Results

The theoretical results for 1D camera self-calibration and its applications to 2D camera calibration have been implemented and experimented on synthetic and real images. Due to space limitation, we do not present the results on synthetic data, the algorithms generally perform very well. We only show some real examples. Here, we consider a scenario of a real camera mounted on a robot arm. Two sequences of images are acquired by the camera moving in two different planes. The first sequence contains seven (indexed from 16 to 22) images (cf. Fig. 5) and the second contains eight (indexed from 8 to 15).

The calibration grid was used to have the ground truth for the internal camera parameters which have been measured as $\alpha_u = 1534.7$, $\alpha_v = 1539.7$, $u_0 = 281.3$, and $v_0 = 279.0$ using a standard calibration method [6].

We take triplets of images from the first sequence and, for each triplet, we estimate the trifocal line and the vanishing point of the rotation axes using the three fundamental matrices of the triplet. The 1D self-calibration is applied for estimating the images of the circular points along the trifocal lines. To evaluate the accuracy of the estimation, the images of the circular points of the trifocal plane are recomputed in the image plane from the known internal parameters by intersecting the image of the absolute conic with the trifocal line. Table 1 shows the results for different triplets of images of the first sequence.

Since we have more than three images for the same planar motion of the camera, we could also estimate the trifocal line and the vanishing point of the rotation axes by using all the available fundamental matrices of the seven images of the sequence. The results using redundant images are presented for different triplets in Table 2. We note the slight improvement of the results compared with those presented in Table 1.

The same experiment was carried out for the other sequence of images where the camera underwent a different planar motion. Similar results to the first image sequence are obtained. We only give the result for one triplet of images in Table 3 for this sequence.

Now, two sequences of images, each corresponding to a different planar motion, yield four distinct imaginary points on the image plane which must be on the image ω of the absolute conic. Assuming that there is no camera skew, we could fit to those four points an imaginary ellipse using standard techniques and compute the resulting internal parameters. Note that we did not use the pole/polar constraint of the vanishing point of the rotation axes on the absolute conic as it was discussed in Section 6. This constraint is not numerically reliable.

To have an intuitive idea of the planar motions, the two trifocal lines together with one image are shown in Fig. 6.

The ultimate goal of self-calibration is to get 3D metric reconstruction. 3D reconstruction from two images of the sequence is performed by using the estimated internal parameters as illustrated in Fig. 7. To evaluate the reconstruction quality, we did the same reconstruction using the known internal parameters. Two such reconstructions differ merely by a 3D similarity transformation which could be easily estimated. The resulting relative error for normalized 3D coordinates by similarity between the reconstruction from self-calibration and offline calibration is 3.4 percent.

8 Conclusions and Other Applications

First, we have established that the two internal parameters of a 1D camera can be uniquely determined through the trifocal tensor of three 1D images. Since the trifocal tensor can be estimated linearly from at least seven points in three 1D images, the method of the 1D self-calibration is a real linear method (modulo the fact that we have to find the roots of a third degree polynomial in one variable), no over-parameterization was introduced.

Second, we have proven that if a 2D camera undergoes a planar motion, the 2D camera reduces to a 1D camera in the plane of motion. The reduction of a 2D image to a 1D image can be efficiently performed by using only the fundamental matrices of 2D images. Based on this relation between 2D and 1D images, the self-calibration method for 1D cameras can be applied for self-calibrating a 2D camera. Our experimental results based on real image sequences show the good stability of the solutions yielded by the 1D self-calibration method and the accurate 3D metric reconstruction that can be obtained from the internal parameters of the 2D camera estimated by the 1D self-calibration method. The camera motions that defeat the self-calibration method developed in Section 4 are described in [24].

References

[1] M. Armstrong, A. Zisserman, and R. Hartley, "Self-Calibration from Image Triplets," *Proc. Fourth European Conf. Computer Vision,* B. Buxton and R. Cipolla, eds., pp. 3–16, Apr. 1996.

[2] M. Armstrong, "Self-Calibration from Image Sequences," PhD thesis, Univ. of Oxford, 1996.

[3] K. Åström, "Invariancy Methods for Points, Curves, and Surfaces in Computational Vision," PhD thesis, Lund Univ., 1996.

[4] P.A. Beardsley and A. Zisserman, "Affine Calibration of Mobile Vehicles," *Proc. Europe-China Workshop Geometrical Modeling and Invariants for Computer Vision,* R. Mohr and C. Wu, eds., pp. 214–221, Apr. 1995.

[5] T. Buchanan, "The Twisted Cubic and Camera Calibration," *Computer Vision, Graphics, and Image Processing,* vol. 42, no. 1, pp. 130–132, Apr. 1988.
[6] O. Faugeras and G. Toscani, "Camera Calibration for 3D Computer Vision," *Proc. Int'l Workshop Machine Vision and Machine Intelligence,* 1987.
[7] O. Faugeras, "Stratification of Three-Dimensional Vision: Projective, Affine, and Metric Representations," *J. Optical Soc. Am.,* vol. 12, pp. 465–484, 1995.
[8] O. Faugeras and S. Maybank, "Motion from Point Matches: Multiplicity of Solutions," *Int'l J. Computer Vision,* vol. 3, no. 4, pp. 225–246, 1990.
[9] O. Faugeras and B. Mourrain, "About the Correspondences of Points Between n Images," *Proc. Workshop Representation of Visual Scenes,* pp. 37–44, June 1995.
[10] O. Faugeras, L. Quan, and P. Sturm, "Self-Calibration of a 1D Projective Camera and Its Application to the Self-Calibration of a 2D Projective Camera," *Proc. European Conf. Computer Vision,* June 1998.
[11] R. Hartley, "In Defence of the 8-Point Algorithm," *Proc. Fifth Int'l Conf. Computer Vision,* pp. 1,064–1,070, June 1995.
[12] R.I. Hartley, "Euclidean Reconstruction from Uncalibrated Views," *Proc. Darpa-Esprit Workshop Applications of Invariants in Computer Vision,* pp. 187–202, Oct. 1993.
[13] R.I. Hartley, "A Linear Method for Reconstruction from Lines and Points," *Proc. Fifth Int'l Conf. Computer Vision,* E. Grimson, ed., pp. 882–887, June 1995.
[14] A. Heyden, "Geometry and Algebra of Multiple Projective Transformations," PhD thesis, Lund Univ., 1995.
[15] A. Heyden and K. Astrom, "Algebraic Properties of Multilinear Constraints," *Proc. Fourth European Conf. Computer Vision,* B. Buxton and R. Cipolla, eds., pp. 671–682, Apr. 1996.
[16] A. Heyden, "A Common Framework for Multiple-View Tensors," *Proc. Fifth European Conf. Computer Vision,* pp. 3–19, June 1998.
[17] Q.-T. Luong and O. Faugeras, "Self-Calibration of a Moving Camera from Point Correspondences and Fundamental Matrices," *The Int'l J. Computer Vision,* vol. 22, no. 3, pp. 261–289, 1997.
[18] S.J. Maybank and O.D. Faugeras, "A Theory of Self Calibration of a Moving Camera," *Int'l J. Computer Vision,* vol. 8, no. 2, pp. 123–151, 1992.
[19] M. Pollefeys, R. Koch, and L. Van Gool, "Self-Calibration and Metric Reconstruction in Spite of Varying and Unknown Internal Camera Parameters," *Int'l Conf. Computer Vision,* pp. 90–95, 1998.
[20] L. Quan and T. Kanade, "Affine Structure from Line Correspondences with Uncalibrated Affine Cameras," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 19, no. 8, pp. 834–845, Aug. 1997.
[21] J.G. Semple and G.T. Kneebone, *Algebraic Projective Geometry.* Oxford Science Publication, 1952.
[22] A. Shashua, "Algebraic Functions for Recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 17, no. 8, pp. 779–789, Aug. 1995.
[23] M. Spetsakis and J. Aloimonos, "A Unified Theory of Structure from Motion," *Proc. DARPA Image Understanding Workshop,* pp. 271–283, 1990.
[24] P. Sturm, "Vision 3D Non Calibrée: Contributions à la Reconstruction Projective et étude des Mouvements Critiques Pour L'Auto-Calibrage," PhD Thesis, INPG, Dec. 1997.
[25] P.H.S. Torr and A. Zissermann, "Performance Characterization of Fundamental Matrix Estimation Under Image Degradation," *Machine Vision and Applications,* vol. 9, pp. 321–333, 1997.
[26] B. Triggs, "Matching Constraints and the Joint Image," E. Grimson, ed., *Proc. Fifth Int'l Conf. Computer Vision,* pp. 338–343, June 1995.
[27] B. Triggs, "Autocalibration and the Absolute Quadric," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 609–614, June 1997.
[28] Cyril Zeller and Olivier Faugeras, "Camera Self-Calibration from Video Sequences: The Kruppa Equations revisited," Research report 2793, INRIA, Feb. 1996.
[29] Z. Zhang, R. Deriche, O. Faugeras, and Q.T. Luong, "A Robust Technique for Matching Two Uncalibrated Images Through the Recovery of the Unknown Epipolar Geometry," *Artificial Intelligence,* vol. 78, pp. 87–119, 1995.

Methods and Geometry for Plane-Based Self-Calibration*

P. Gurdjos

IRIT-TCI, 118 Route de Narbonne
31062 Toulouse Cedex 4, France
www.irit.fr/∼Pierre.Gurdjos/

P. Sturm

INRIA Rhône-Alpes, 655 Avenue de l'Europe
38330 Montbonnot, France
www.inrialpes.fr/movi/people/Sturm

Abstract. We consider the problem of camera self-calibration, from images of a planar scene with *unknown* metric structure. The general case of possibly varying focal length is addressed. This problem is non-linear in general. One of our contributions is a non-linear approach, that makes abstraction of the (possibly varying) focal length, resulting in a computationally efficient algorithm. In addition, it does not require a good initial estimate of the focal length, unlike previous approaches. As for the initialization of other parameters, we propose a practical approach, that simply requires to take one image in roughly fronto-parallel position. Closed-form solutions for various configurations of unknown intrinsic parameters are provided. Our methods are evaluated and compared to previous approaches, using simulated and real images. Besides our practical contributions, we also provide a detailed geometrical interpretation of the principles underlying our approach.

1. Introduction

Calibration of a camera consists in recovering its metric properties, which are partially encoded as a set of so-called internal parameters. Methods for calibration from 1D [16], 2D [11, 15, 2] and 3D *known* structures can be found in the literature. When the structure of the observed world is *unknown* ("0D" structure), this problem is referred to as "self-calibration". On the other hand, assumptions about the degree of the observed (unknown) world space can sometimes be made. When the observed scene is 2D, *i.e.* consists of a plane, one refers to it as the plane-based self-calibration problem. In addition to the internal parameters, plane-based self-calibration consists in recovering the plane's metric structure. Besides the fact that, in many man-made environments, planes are widely present and easily identifiable, an important advantage of plane-based self-calibration is that it only requires to estimate inter-image homographies (induced by world planes), using stabler and more accurate algorithms than those for inter-image transformations arising from projections of 3D points, *e.g.* the fundamental matrix.

In the rest of this paper, we suppose that we have taken n images of a rigid planar object – the *(world) plane* – whose metric structure is unknown. Further, we suppose that we have at our disposal inter-image homographies H_{ij} (there exist "real-time" algorithms for estimating them, e.g. [4]). The two main goals of our work are to calibrate the camera and to compute the plane's metric structure, allowing to rectify its images. These goals are linked of course: given the plane's metric structure, we know how to calibrate [11, 15], and given the calibration, how to rectify the plane [5]. Self-calibration is based on constraints on the intrinsic parameters. Concretely, we consider the rather general case of zero skew, constant but unknown principal point and a possibly varying unknown focal length.

*We acknowledge support from GdR ISIS (Projets Jeunes Chercheurs).

2. A Stratified Problem Formulation

We first consider the case where we have prior knowledge of the plane's metric structure. The problem reduces to that of camera calibration and linear solutions exist [11, 15], even for varying intrinsics [11, 2]. We briefly outline the geometrical basis of these methods. A plane's metric structure is encoded by the locus of its circular points (cf. §3). Here, this means that we know the images of these points (ICP) in all n views. Our goal here is camera calibration, which can be solved by computing the image of the absolute conic (IAC). The circular points of the plane lie on the absolute conic, and thus the ICP lie on the IAC. Hence, calibration can be seen as fitting a conic (the IAC) to all available ICP. This can be done by solving a linear equation system.

Consider now the general case, where the plane's metric structure is unknown. We introduce unknowns to parameterize the ICP in one of our views, and compute the ICP in the other views using inter-image homographies. The estimation problem becomes non-linear, and iterative methods for its solution have been proposed [13, 6]. One of their drawbacks, common to non-linear problems, is the need for good initial estimates. Another problem is that the number m of unknowns may become relatively large in the case of varying intrinsics, increasing the sensitivity to the initial estimates and computation time (generally with $\mathcal{O}(m^3)$ complexity per iteration). One of our contributions is a parameterization that allows to solve the problem using a fixed number of unknowns (reducing the complexity to $\mathcal{O}(m)$), and that has a nice geometric interpretation (not shown completely in this paper due to lack of space).

Up to now, we have considered two extreme cases: completely known or completely unknown metric plane structure. In the latter case we know at least its projective structure (every image is a projective "model" of the plane). Consider now the obvious intermediate case: known affine structure. The problem remains non-linear, but can be expressed using fewer unknowns and simpler equations. Most importantly, closed-form solutions for interesting minimal cases are now possible (see §5.3). One way of recovering the plane's affine structure is e.g. to identify the projections of two sets of parallel lines on the plane. Another solution, that we use in this paper, is to simply take a fronto-parallel image of the plane. Taking an exactly fronto-parallel image is of course difficult. However, we show that in practice, a roughly fronto-parallel image is sufficient to get good initial estimates using the closed-form solutions. We use them to start a non-linear optimization process, where the assumption of fronto-parallelism can be dropped.

We thus have established a stratification for plane-based calibration: *calibration* relies on the knowledge of the plane's metric structure, whereas *self-calibration* only requires its projective structure. The intermediate case of known affine structure is analogous to using scene constraints in traditional ("3D") self-calibration [5, 14], or, in the case of a fronto-parallel image, to self-calibration based on special motions (typically, pure translations [1, 8]).

3. Background

3.1. Calibration and the Absolute Conic (AC)

It is well known that camera (self-) calibration is equivalent to computing the image of the absolute conic (IAC) [3]. If A is the camera's calibration matrix, then the IAC is given by $\boldsymbol{\omega} = \mathsf{A}^{-\top}\mathsf{A}^{-1}$, with:

$$\boldsymbol{\omega} \sim \begin{pmatrix} \tau^2 & 0 & -\tau^2 u_0 \\ 0 & 1 & -v_0 \\ -\tau^2 u_0 & -v_0 & \tau^2 u_0^2 + v_0^2 + \tau^2 f^2 \end{pmatrix}, \quad (1)$$

with 4 degrees of freedom (d.o.f.): the focal length f, the aspect ratio τ and the principal point (u_0, v_0).

3.2. Representing the Euclidean Structure of a World Plane in Images

Given some world plane Π projected onto an image plane $\mathcal{I}$, the world-to-image homography that maps points $\mathbf{p}$ on Π onto pixels $\mathbf{m}$ on $\mathcal{I}$ is defined by a 3×3 matrix P such that $\mathbf{m} \sim \mathsf{P}\mathbf{p}$. We assume that Π is a Euclidean plane but, *a priori*, the image of Π under P only yields its projective structure. As for the world-to-image homography, we use the following decomposition (*e.g.* see [5, pp. 41-44]):

$$\mathsf{P} = \mathsf{P}_p\mathsf{P}_a\mathsf{P}_s = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \mu & \lambda & 1 \end{pmatrix} \begin{pmatrix} \beta & \alpha & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \mathsf{P}_s, \quad (2)$$

where P_p (resp. P_a) is the projective (resp. affine) component of P, with 4 d.o.f. in all, while the 3×3 matrix P_s is the metric component of P with 4 d.o.f. (*i.e.* P_s is a 2D similarity transformation in Π). The rôle of the scalars $\mu, \lambda, \beta, \alpha$ is discussed below, but note that they let $(\mathsf{P}_p\mathsf{P}_a)^{-1}$ define metric image *rectification*, by mapping points on $\mathcal{I}$ onto Π w.r.t. some "arbitrary" Euclidean coordinate system. This means that μ, λ, β and α are a representation of the world plane's Euclidean structure. In the sequel, we neglect the "arbitrary" metric component of P, so we consider that

$$\mathsf{P} = \mathsf{P}_p\mathsf{P}_a = \begin{pmatrix} \beta & \alpha & 0 \\ 0 & 1 & 0 \\ \mu\beta & \mu\alpha + \lambda & 1 \end{pmatrix}. \quad (3)$$

In the following paragraphs, we establish links between the parameters μ, λ, β and α and the circular points and their images, as well as the world plane's vanishing line, which encode its Euclidean respectively affine structure.

Vanishing Line. Under the world-to-image homography P (see (3)), the world plane's line at infinity is mapped to its vanishing line $\mathbf{v}$ according to

$$\mathbf{v} = \mathsf{P}^{-\top}(0,0,1)^\top = (-\mu, -\lambda, 1)^\top \quad (4)$$

Hence, λ and μ represent the plane's affine structure.

Circular Points (CP). The circular points (CP) of a plane have the following properties: (1) They lie on the absolute conic. (2) Their coordinates $\mathbf{i}_\pm \sim (1, \pm i, 0)^\top$ are the same in every Euclidean coordinate system.

Image of Circular Points (ICP). Under P, the CP transform into the ICP (images of circular points) according to:

$$\mathsf{P}\mathbf{i}_\pm = (\beta, 0, \mu\beta)^\top \pm i\,(\alpha, 1, \mu\alpha + \lambda)^\top. \quad (5)$$

The ICP are another representation, besides P, of the world plane's metric structure. They are defined by the same parameters, $\mu, \lambda, \beta, \alpha$ and lie on the vanishing line $\mathbf{v}$.

Conic Dual to the Circular Points. The conic dual to the circular points (CDCP) is defined by $\mathsf{D}^*_\infty = \mathbf{i}_+\mathbf{i}_-^\top + \mathbf{i}_-\mathbf{i}_+^\top \sim \mathsf{diag}(1,1,0)$. D^*_∞ is of rank 2 and transforms under any rank 3 homography $\mathsf{M} \in \mathbb{R}^{3\times3}$ into the symmetric rank 2 matrix $\mathsf{M}\mathsf{D}^*_\infty\mathsf{M}^\top$. Under P, the CDCP transforms into

$$\boldsymbol{\Sigma} = \begin{pmatrix} \bar{\mathsf{P}}\bar{\mathsf{P}}^\top & \bar{\mathsf{P}}\bar{\mathsf{P}}^\top(\mu,\lambda)^\top \\ (\mu,\lambda)\,\bar{\mathsf{P}}\bar{\mathsf{P}}^\top & (\mu,\lambda)\,\bar{\mathsf{P}}\bar{\mathsf{P}}^\top(\mu,\lambda)^\top \end{pmatrix}, \quad (6)$$

where $\bar{\mathsf{P}}$ is the upper left 2×2 part of P. The conic $\boldsymbol{\Sigma}$ is yet another representation of the world plane's metric structure, which will be used in our self-calibration approach. Note that $\mathrm{null}(\boldsymbol{\Sigma}) = \mathbf{v} = (-\mu, -\lambda, 1)^\top$.

Under A, the CDCP transforms into

$$\boldsymbol{\Lambda} = \mathsf{A}\mathsf{D}^*_\infty\mathsf{A}^\top \sim \mathsf{diag}(1, \tau^2, 0). \quad (7)$$

We also have the following property, used later: image lines $\mathbf{m}_1$ and $\mathbf{m}_2$ are orthogonal iff $\mathbf{m}_1^\top\boldsymbol{\Lambda}\mathbf{m}_2 = 0$.

4. Plane-based Calibration

In this section, we review constraints on the IAC $\boldsymbol{\omega}$ that are used to solve the plane-based calibration problem. We first review the basic equations introduced in [11, 15]. In §4.2, we then describe an approach leading to equations that do not take into account the focal length. This allows the number of unknowns to remain constant, even in the case of a varying focal length. This advantage may be very interesting for the non-linear *self-calibration* problem, and in §5.2, we accordingly extend the approach of §4.2.

4.1. Basic Equations

The CP lie on the AC, and hence the ICP lie on the IAC, which is expressed by $(\mathsf{P}\mathbf{i}_\pm)^\top \boldsymbol{\omega}(\mathsf{P}\mathbf{i}_\pm) = 0$. Requiring that both real and imaginary parts be zero, yields:

$$\mathbf{h}_1^\top \boldsymbol{\omega} \mathbf{h}_1 = \mathbf{h}_2^\top \boldsymbol{\omega} \mathbf{h}_2, \quad \mathbf{h}_1^\top \boldsymbol{\omega} \mathbf{h}_2 = 0, \tag{8}$$

where $\mathbf{h}_1$and $\mathbf{h}_2$ are the first two columns of P. These constraints are linear in the elements of $\boldsymbol{\omega}$.

4.2. The Centre Line Constraint

Equations (8) include the unknown focal length (contained in $\boldsymbol{\omega}$) which might be disadvantageous, as explained above. Thus, we now describe an alternative approach, based on a geometric constraint on the principal point and aspect ratio, the *centre line constraint* [2], that is regardless of the possibly varying focal length. This constraint results from the central projection of a planar object, as an original way of formulating the plane-based calibration problem. This might seem unrelated to the geometrical background stated above but one of our contributions is to give a more convenient matrix representation of the centre line constraint in terms of the imaged CDCP's $\boldsymbol{\Lambda}$ and $\boldsymbol{\Sigma}$ in order to extend it to self-calibration in §5.2. To remind the reader of the geometrical background required to thoroughly understand this approach, we give the following theorem[1] and corollary.

Theorem 1 If one rotates the image plane around its intersection with the world plane, while moving the camera center "adequately" along a circle (called *centre circle*), in a plane perpendicular to this intersection (called *centre plane*), then the world points and the image points remain in homographic correspondence under P.

Corollary 2 By orthogonally projecting the centre circle onto the image plane, the locus of the principal point is a line segment in the image plane called *centre segment* and the line that contains it is called the *centre line* (*cf.* Fig. 1).

What the theorem says is best explained by referring to the animation downloadable at XXX[2]. For a proof, see [9, pp. 511-517]. In our case, the most important issue is given by the corollary: if the world-to-image homography matrix P is known, then the principal point necessarily lies on a certain line, called centre line (*cf.* Fig. 1). In [2], the following properties have been stated.

Properties 3 (1) The centre line coordinates $\boldsymbol{\phi}$ only depend on the aspect ratio τ and the world-to-image homography, *i.e.* are irrespective of the focal length f:

$$\boldsymbol{\phi} = \left(-\tau^2\varphi_1, -\varphi_2, \tau^2\varphi_3 + \varphi_4\right)^\top = \boldsymbol{\phi}(\mathsf{P},\tau), \tag{9}$$

where φ_i denotes the i-th element of the 4-vector

$$\boldsymbol{\varphi} = \begin{pmatrix} (P_{12}P_{31} - P_{11}P_{32})(P_{31}^2 + P_{32}^2) \\ P_{31}(P_{31}^2 + P_{32}^2) \\ (P_{12}P_{31} - P_{11}P_{32})(P_{31}P_{11} + P_{12}P_{32}) \\ P_{31}P_{32} \end{pmatrix}, \tag{10}$$

and P_{ij} is the element (i, j) of P.

(2) The centre line $\boldsymbol{\phi}$ contains the principal point, *i.e.*

$$(u_0, v_0, 1)^\top \boldsymbol{\phi} = 0. \tag{11}$$

This equation is called *centre line constraint*.

(3) The centre line is orthogonal to the vanishing line $\mathbf{v}$, hence (cf. §3), $\boldsymbol{\phi}$ and $\mathbf{v}$ are conjugated w.r.t. $\boldsymbol{\Lambda} = \mathsf{A}\mathsf{C}_\infty^*\mathsf{A}^\top$:

$$\mathbf{v}^\top \boldsymbol{\Lambda} \boldsymbol{\phi} = 0. \tag{12}$$

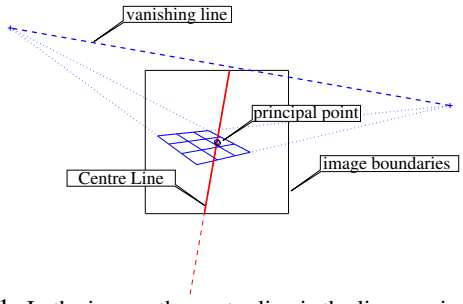


Figure 1: In the image, the centre line is the line passing through the principal point and orthogonal to the vanishing line.

An important aspect is that it the *centre line constraint* (with suitable normalization) can express a geometric error (*i.e.* the distance from the principal point to the centre line [2]) that can be minimized (in the least-squares sense) with regard to the problem of plane-based calibration.

The pencil of centre lines. Consider the definition (9) of the centre line. It depends linearly on the squared aspect ratio, τ^2. We may thus define the locus of the centre line as a linear family, i.e. a line pencil, that is independent of τ. To clarify this, let us rewrite (9):

$$\boldsymbol{\phi} = (0, -\varphi_2, \varphi_4)^\top + \tau^2 (-\varphi_1, 0, \varphi_3)^\top = \mathbf{d}_1 + \tau^2 \mathbf{d}_2.$$

Let us denote this line pencil by $\mathscr{P}$. Given the definition of $\boldsymbol{\varphi}$ in (10) and of P in (3), the two chosen "base lines" of $\mathscr{P}$ can be written in terms of the image of the CDCP:

$$\mathbf{d}_1 = (\boldsymbol{\Sigma}\mathbf{e}_3) \times \mu\mathbf{e}_1, \quad \mathbf{d}_2 = (\boldsymbol{\Sigma}\mathbf{e}_3) \times \lambda\mathbf{e}_2,$$

[1] To our knowledge, until recently this theorem has never been reported in the vision literature, even if G. Sparr in [10] showed algebraically that the camera centre is constrained to an elliptical space curve.

[2] It is worthy of note that, as a result, if one only looks at the image, there exist displacements of the planar object that are totally invisible, *i.e.* for which the image of the object is the same.

where $\mathbf{e}_1^\top = (1,0,0)\,, \mathbf{e}_2^\top = (0,1,0)\,, \mathbf{e}_3^\top = (0,0,1)$. The vertex of $\mathscr{P}$ may be computed as follows:

$$\mathbf{d}_1 \times \mathbf{d}_2 \sim \mathbf{\Sigma}\mathbf{e}_3. \tag{13}$$

By definition, this point belongs to the centre line $\boldsymbol{\phi}$. Together with (11) and (12), we have thus established three points on the centre line: the vertex of $\mathscr{P}$, the principal point $\mathbf{p}_0 = (u_0, v_0, 1)^\top$ and the point given by $\mathbf{\Lambda}\mathbf{v}$. These points being collinear may be expressed as:

$$\det\left(\mathbf{\Sigma}\mathbf{e}_3 \mid \mathbf{\Lambda}\mathbf{v} \mid \mathbf{p}_0\right) = 0. \tag{14}$$

This is an alternative representation of the centre line constraint. It links the aspect ratio (contained in $\mathbf{\Lambda}$, cf. (7)), the principal point, and the parameters $\mu, \lambda, \beta, \alpha$ of the plane's metric structure (in $\mathbf{\Sigma}$ and $\mathbf{v}$). This constraint is the basis for our self-calibration approach, cf. §5.2.

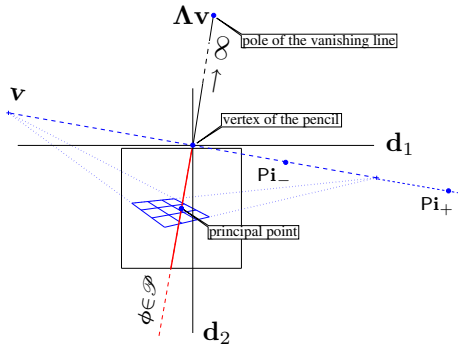


Figure 2: The pole of the vanishing line $\mathbf{v}$ w.r.t. $\mathbf{\Lambda}$, the vertex of the line pencil $\mathscr{P}$ (containing $\mathbf{d}_1, \mathbf{d}_2$) and the principal point are aligned; $\mathsf{Pi}_\pm$ denote the image of the CP $\mathbf{i}_\pm$ under P.

Further notes. Within the above geometrical framework based on the line pencil $\mathscr{P}$, we give the following propositions (proofs are straightforward). (1) From (13), it follows that the vertex of $\mathscr{P}$ is the pole of the image plane's line at infinity w.r.t. $\mathbf{\Sigma}$. (2) The *intersection point of the centre line and the line at infinity* is the pole of the vanishing line w.r.t. $\mathbf{\Lambda}$, i.e. $\boldsymbol{\phi} \times \mathbf{e}_3 = \mathbf{\Lambda}\mathbf{v}$.

The centre line constraint has never been used in image rectification. In [5, pp. 57-63], a "rectangle ambiguity problem" in the estimation of the image of the CDCP from the four angles of a rectangle is mentioned, that might be solved by adding (14), with u_0, v_0, τ supposedly known.

5. Plane-Based Self-Calibration

5.1. Existing Non-Linear Solutions

"Basic" constraints on the IAC and ICP. In [13], a solution is given for plane-based calibration in the case of constant internal parameters, which has been extended in [3, § 18.7, pp. 470-471] to the "varying focal length" case. Geometrically, this approach is based on two main ideas. First, the ICP (which encode the metric structure of the plane) are mapped from image to image via the inter-image homographies H_{ij} ; if we denote by $\mathbf{p}_{1\pm} = \mathsf{P}\mathbf{i}_\pm$ the ICP in some key image (say image 1), then we have $\mathbf{p}_{j\pm} = \mathsf{H}_{1j}\mathbf{p}_{1\pm}$. Second, the CPs lie on the AC (which encode the internal parameters of the camera), hence calibration can be seen as fitting (imaginary) conics $\boldsymbol{\omega}_j$ (the IACs) to all available ICP $\mathbf{p}_{j\pm}$. Two equations are provided by each image j:

$$(\mathsf{H}_{1j}\mathbf{p}_{1\pm})^\top \boldsymbol{\omega}_j (\mathsf{H}_{1j}\mathbf{p}_{1\pm}) = 0. \tag{15}$$

Given n inter-image homographies H_{1j} $(1 \le j \le n)$, the *self-calibration* problem is that of solving the system of n equations (15) for the $3+m$ d.o.f. in $\boldsymbol{\omega}$ and the 4 d.o.f. in $\mathbf{p}_{1\pm}$, where m is the number of unknown focal lengths. If $m = 1$, at least 4 inter-image homographies are required; if $m = n$, at least 7 are required.

This problem is non-linear and can be solved using iterative methodes. It requires initial values, in particular for the (possibly different) focal lengths. This critical issue, already mentioned in [13], motivated us to (1) seek a minimization criterion that would be irrespective of f (see §5.2); (2) find closed-form solutions for minimal cases (see §5.3).

5.2. A New Non-Linear Solution

The centre line constraint (14) in §4.2 was developed in terms of the world-to-image homography P, and is thus suitable when the world plane's structure is known. Here, the structure is only known indirectly (each image represents the plane's projective structure), so we have to adapt the constraint to the use of inter-image homographies instead of the world-to-image one. Consider some image as the key image; let H_i be the inter-image homography matrix from the key image to image i. Under H_i, the vanishing line $\mathbf{v}$ of the key image transforms into $\mathbf{v}_i = \mathsf{H}_i^{-\top}\mathbf{v}$; the imaged $\mathbf{\Sigma}$ of the key image transforms into $\Sigma_i = \mathsf{H}_i \mathbf{\Sigma} \mathsf{H}_i^\top$.

It follows that the CL constraint (14) in image i is

$$\det\left(\mathsf{H}_i\mathbf{\Sigma}\mathsf{H}_i^\top\mathbf{e}_3 \mid \mathbf{\Lambda}\mathsf{H}_i^{-\top}\mathbf{v} \mid \mathbf{p}_0\right) = 0. \tag{16}$$

Problem 4 Given n inter-image homography matrices H_i $(1 \le i \le n)$, the *self-calibration* problem of a camera *with a possibly varying focal length* is that of solving the system of n equations (16) for the 2 d.o.f. in $\mathbf{p}_0$, the single d.o.f. in $\mathbf{\Lambda}$ and the 4 d.o.f. in $\mathbf{\Sigma}$, under the condition $\mathrm{null}(\mathbf{\Sigma}) = \mathbf{v}$.

There is a *fixed* number of 7 unknowns ($\alpha, \beta, \lambda, \mu$ and u_0, v_0, τ) – *even in* the "varying focal length" case – so that at least 7 inter-image homographies are required. Once the constant internal parameters and Euclidean structure are recovered, the different focal lengths can be computed using linear algorithms described in §4.

Implementation. Referring to (16), let us denote by M the matrix $\left(\mathsf{H}\mathbf{\Sigma}\mathsf{H}^\top\mathbf{e}_3 \mid \mathbf{\Lambda}\mathsf{H}^{-\top}\mathsf{v} \mid \mathbf{p}_0\right)$ (we omit the index

for H). A solution to Problem 4 can be obtained by minimizing a cost function depending on $\det \mathsf{M}$. This is a non-linear problem but let us notice that since the condition $\mathrm{null}(\mathbf{\Sigma}) = \mathbf{v}$ is directly ensured from the definition of $\mathbf{\Sigma}$ in (6), no constrained optimization algorithm is required.

An interesting aspect of our formulation is that we can easily attach a geometric meaning to the algebraic quantity $\det \mathsf{M}$. Indeed, $\det \mathsf{M}$ is equal to the mixed triple product of its three column vectors: this means[3] that $\delta = \frac{1}{k}|\mathbf{m}_1^\top (\mathbf{m}_2 \times \mathbf{m}_3)|$, where $k = \sqrt{(\mathbf{m}_2 \times \mathbf{m}_3)^\top \mathbf{\Lambda} (\mathbf{m}_2 \times \mathbf{m}_3)}$, represents the distance from point $\mathbf{m}_1$ to (Centre) line $(\mathbf{m}_2 \times \mathbf{m}_3)$.

In our experiments, we used the non-linear least squares implementation (Levenberg-Marquardt algorithm) available in the MATLAB Optimization Toolbox [7].

The Jacobian information for the objective function can be easily supplied, using the following properties: (1) $\det \mathsf{M} = (H_{31} + \mu H_{33}) \det \mathsf{N}_1 + (H_{32} + \lambda H_{33}) \det \mathsf{N}_2$, where $\mathsf{N}_i = \begin{pmatrix} \mathsf{H}\boldsymbol{\sigma}_i \mid \mathbf{m}_2 \mid \mathbf{m}_3 \end{pmatrix}$; $\boldsymbol{\sigma}_i$ is the i-th column of $\mathbf{\Sigma}$. (2) $\frac{d \det \mathsf{M}}{dx} = \mathrm{trace}\left\{ \frac{d\mathsf{M}}{dx} \mathrm{adj}\, \mathsf{M} \right\}$, where $\mathrm{adj}\, \mathsf{M} = (\det \mathsf{M})\, \mathsf{M}^{-1}$ is the adjoint matrix of M.

5.3. Direct Solutions

We develop closed-form solutions, based on the assumption that one image was taken in fronto-parallel position relative to the world plane (*i.e.* image and world planes are parallel). As mentioned in §2, we immediately have the world plane's affine structure; equivalently, we now have $\lambda = \mu = 0$ for the representations described in §3.2 As mentioned in §2, we will use the assumption of fronto-parallelism *only for the algorithm initialization*. For the subsequent non-linear optimization, we drop this assumption.

Let us consider what we can say about the ICP in the fronto-parallel image: they lie on both, the IAC and plane's vanishing line (here, the image plane's line at infinity). Hence, according to (1), they are given as:

$$\left(\tau^{-1}, 0, 0\right)^\top \pm i\,(0, 1, 0)^\top \sim (1, \pm\tau i, 0)^\top . \qquad (17)$$

Consequently, using (5), we know that $\alpha = 0$ and thus, we can recover the world plane's Euclidean structure up to the single unknown $\beta = \tau^{-1}$.

We now sketch closed-form solutions for various scenarios, depending if the aspect ratio and/or principal point are known or not, and if the focal length is constant or varying. In the case of a *known aspect ratio*, the fronto-parallel image directly gives us the plane's metric structure, and self-calibration reduces to calibration [11, 15, 2]. So, in the following, we only consider an *unknown aspect ratio*.

As shown in (17), the ICP in the fronto-parallel image can be parameterized by the unknown τ. Using inter-image homographies, we also parameterize the ICP in the other images using τ. Let H be the homography, mapping the fronto-parallel to some other image. The basic calibration equations (8) then become:

$$\mathbf{h}_1^\top \boldsymbol{\omega} \mathbf{h}_1 - \tau^2 \mathbf{h}_2^\top \boldsymbol{\omega} \mathbf{h}_2 = 0, \qquad \mathbf{h}_1^\top \boldsymbol{\omega} \mathbf{h}_2 = 0. \qquad (18)$$

The second equation is linear and the same as in (8), hence with 5 or more inter-image homographies, the unknowns can be recovered linearly. As for the first equation, using the fact that $\tau^2 = \omega_{11}/\omega_{22}$, we may reformulate it as

$$\omega_{22} \mathbf{h}_1^\top \boldsymbol{\omega} \mathbf{h}_1 - \omega_{11} \mathbf{h}_2^\top \boldsymbol{\omega} \mathbf{h}_2 = 0.$$

This is quadratic in the set of coefficients of ω, with only ω_{11} and ω_{22} appearing squared. In the following, we describe several minimal cases, but due to lack of space, without much detail. Note that the focal length of the fronto-parallel image can not be recovered [11], so we ignore it.

In the case of a *known principal point*, two images, the fronto-parallel and another one, are sufficient for self-calibration. The only unknowns are the aspect ratio and the focal length of the second view (may be different from that of the fronto-parallel view). We suppose that the images are centered in the principal point, *i.e.* we have $\boldsymbol{\omega} = \mathsf{diag}(\tau^2, 1, \tau^2 f^2)$. Equations (18) thus become, after replacing the unknowns by $a = \tau^2$ and $b = \tau^2 f^2$:

$$H_{21}^2 + a(H_{11}^2 - H_{22}^2) + bH_{31}^2 - abH_{32}^2 - a^2 H_{12}^2 = 0$$
$$H_{21}H_{22} + aH_{11}H_{12} + bH_{31}H_{32} = 0.$$

The two equations can be reduced a single quadratic one in b. Writing down explicit closed-form solutions for τ and f in terms of H is trivial.

In case of an *unknown principal point* and *constant (resp. varying) focal length*, three (resp. four) images are sufficient and the problem can be written as a cubic (resp. quartic) polynomial in one variable. Hence, self-calibration has a closed-form solution.

6. Experiments

Synthetic data. Self-calibration using the CL constraint of §5.2 ("CL-NONL-SELF" has first been tested using synthetic data. We compare it with the results of an algorithm using the basic constraints, see §5.1 ("BAS-NONL-SELF"). For each experiment, the camera has constant internal parameters with nominal values $u_0 = 255 \pm 50$ pixels, $v_0 = 255 \pm 50$ pixels, $\tau = 1 \pm 0.1$ (with normal distribution). For each camera c in each experiment, the inclination angle between the world and the image plane is set to $30° \pm 10°$ (except for the first for which it is set to $\pm 10°$); the angles for azimuth and rotation around the optical axis are set to $0° \pm 90°$ (normal distribution); the (varying) focal length is set to $f_c = 700 \pm 700$ pixels (normal distribution). 100 points are randomly generated in the first image, then transferred to the others with a of perturbation ± 1 pixel (Gaussian noise). The inter-image homographies have been

[3] Indices are interchangeable.

estimated using the normalized DLT algorithm of [3], from the perturbed points. We conducted 200 independent trials for a number of cameras varying from 8 to 24, with a step of 2. In Fig. 3, we show the computed absolute errors for u_0, v_0, for world coordinates x, y (in mms), and relative errors for τ and the focal lengths (in percent). We also sought a threshold on the number of cameras for which the "CL-NONL-SELF" and "BAS-NONL-SELF" algorithms have similar accuracies. Regarding our tests, this threshold is about 15 views. Typically, the algorithm converges in 5 iterations. A "good" initialization of the parameters proved to be crucial. We used the direct solution given in §5.3, except in one case for "BAS-NONL-SELF": this case is plotted with the dashed line with marker '*' and corresponds to focal lengths initialized to 2000 pixels (which is quite realistic); one notes that the convergence is significantly affected.

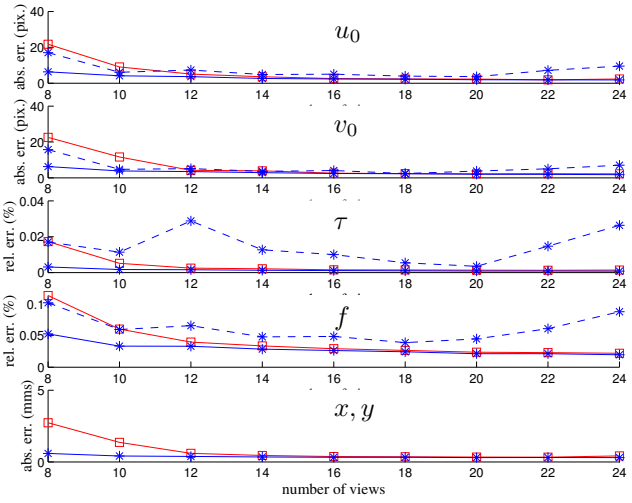


Figure 3: Self-calibration results. (1) our CL-NONL-SELF method ("square" marker). (2) BASIC-NONL-SELF ("star"); dashed line shows influence of a "bad" initialization of f.

Real Images. To evaluate the performance of our non-linear self-calibration algorithm ("CL-NONL-SELF"), we compared the results with those obtained by both basic linear *calibration* ("BAS-LIN-CAL") and basic non-linear *self-calibration* ("BAS-NONL-SELF") algorithms. We used 15 images of a calibration grid (with 80 points), taken from different positions (see top of Fig. 4) using a NIKON COOLPIX 800 at 640×480 resolution. The corners of the squares have been extracted in order to compute inter-image homographies. To avoid critical motions, we took care to apply significative rotations around the optical axis between successive shots. The metric structure of the calibration grid was *only* used by the calibration algorithm BAS-LIN-CAL. The principal point and aspect ratio were assumed to be constant; their estimated values are: $(308, 250, 1.0083)$ for BAS-LIN-CAL, $(325, 253, 0.999)$ for BAS-NONL-SELF and $(325, 260, 0.999)$ for our CL-NONL-SELF algorithm. Note that the relative error between the different aspect ratios is less than 1%.

There could be variations of the focal length owing to the camera's auto-focus, so we assumed f to be varying. The different focal lengths recovered by the algorithm BAS-NONL-SELF are: 1368, 1390, 1383, 1352, 1357, 1357, 1371, 1322, 1346, 1352, 1358, 1345, 1390, 1394, 1387, with mean 1364 and standard deviation around 20 (1%). Table 1 gives the the relative difference (percent) with BAS-LIN-CAL obtained by CL-NONL-SELF. In brackets, the relative errors with BAS-NONL-SELF are shown (a negative value means "closer to BAS-LIN-CAL's estimates"). The relative errors between the calibration and self-calibration algorithms are very small (in most cases less than 1%) for all focal lengths.

0.5 (−0.1)	0.4 (−0.1)	1.2 (−0.1)	0.9 (−0.1)	0.7 (0.1)
0.0 (−0.1)	0.3 (0.1)	1.6 (1.0)	0.2 (0.1)	0.3 (0.2)
0.6 (0.3)	0.4 (0.2)	0.9 (0.1)	1.3 (0.0)	0.1 (−0.1)

Table 1: Focal length self-calibration obtained by the CL-NONL-SELF algorithm from the 15 images of Fig. 4. Relative errors (percent) w.r.t. the BAS-LIN-CAL algorithm are shown. In brackets, the difference with the BAS-NONL-SELF algorithm.

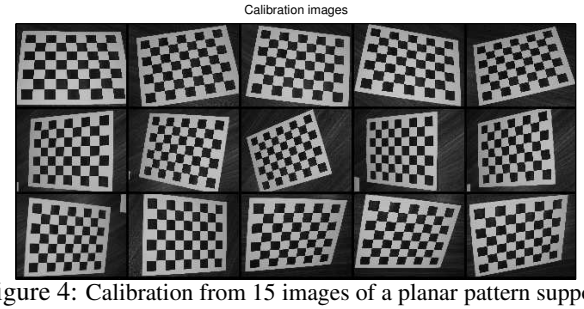


Figure 4: Calibration from 15 images of a planar pattern supposedly unknown.

Videos of a comic book. We acquired several videos of a comic book using a handheld digital camcorder (Panasonic NV-MX 300). Acquisitions were started in roughly fronto-parallel position. The videos were processed automatically to extract and track interest points and to compute inter-image homographies (using a RANSAC-based method). This and the remaining processing was done on 1 out of every 10 frames. Figure 5 shows 8 of the 20 frames used for one of the sequences. We used the closed-form solution of §5.3 corresponding to an unknown aspect ratio but known principal point (image center). This gave the focal length for every frame but the first, and one estimate of the aspect ratio per frame. A single value for the aspect ratio was computed using robust statistics, and used to obtain an initial solution of the world plane's metric structure from the fronto-parallel frame. Then, initial pose estimates for all frames were obtained [12].

These initial estimates were refined by a bundle adjustment (including position of points on the world plane and radial distortion). Bundle adjustment was implemented in

the usual sparse way, and converged in 2 to 3 iterations, each iteration taking a few seconds.

The results were compared with calibration values obtained by filming a 3D calibration grid. The aspect ratio was estimated with 0.2% error (1.0893, compared to a "ground truth" of 1.0919). The principal point was estimated about 10 pixels off. As for the focal length, the results for the first half of the frames were bad as expected (frames are too close to fronto-parallel) whereas for the second half, the mean value was 1366, which means an error of 3.5% (ground truth was 1319). Similar results were obtained for other sequences of the same object.

Figure 5 shows a rectified image of the world plane, obtained from the first frame of the last row in the figure. The structure is well recovered, considering that the page of the comic book was not perfectly flat towards its left.

Figure 5: Left: some of the input images. Right: rectified image of the world plane.

7. Conclusion

We addressed the problem of camera self-calibration, from inter-image homographies induced by a plane with *unknown* metric structure. A non-linear solution, based on properties of circular points and the absolute conic, was previously proposed in [13]. This "basic" algorithm proved to be efficient, but requires a good initial estimate of the focal length. We solved this issue by first proposing a practical approach, that simply requires to take one image in roughly fronto-parallel position. Closed-form solutions for various configurations were obtained. The assumption of fronto-parallelism is only used for initialization and dropped for non-linear optimization.

Another contribution is a new non-linear algorithm, so-called Centre Line-based, that is irrespective of the (possibly varying) focal length. The underlying constraint has already been used for plane-based calibration [2] but is extended here to self-calibration. This extension involves the conic dual to the circular points (and all intrinsic parameters except the focal length). The constraint has a nice geometric interpretation which gives information about the conditions under which the Centre Line-based algorithm is efficient (*i.e.* by applying rotations around the optical axis). From $n \geq 15$ images, the accuracy of the Centre Line-based algorithm is similar to the basic algorithm, while having a lower algorithmic complexity: $\mathcal{O}(n)$ instead of $\mathcal{O}(n^3)$. The need of $n \geq 15$ images can be explained by the intuition that the Centre Line-based algorithm has a higher number of critical configurations than the basic one. Anyway, thanks to the low algorithmic complexity and given that there exist "real-time" algorithms for estimating inter-image homographies [4], one can intend to carry out self-calibration using a large set of images in order to reach a high accuracy.

References

[1] M. Armstrong, A. Zisserman, P. Beardsley. Euclidean Structure from Uncalibrated Images. BMVC, pp. 509-518, 1994.

[2] P. Gurdjos and R. Payrissat. Plane-based Calibration of a Camera with Varying Focal Length: the Centre Line Constraint. BMVC, 623-632, 2001.

[3] R. Hartley and A. Zisserman. Multiple View Geometry in Computer Vision. Cambridge University Press, 2000.

[4] F. Jurie and M. Dhome. A simple and efficient template matching algorithm, ICCV, 2001.

[5] D. Liebowitz. Camera Calibration and Reconstruction of Geometry from Images. PhD Thesis, Oxford, 2001.

[6] E. Malis and R. Cipolla. Multi-view Constraints between Collineations: Application to Self-Calibration from Unknown Planar Structures. ECCV, vol. 2, pp. 610-624, 2000.

[7] Optimization Toolbox User's Guide. Version 2. The MathWorks, Inc. September 2000.

[8] M. Pollefeys and L. Van Gool and M. Proesmans. Euclidean 3D Reconstruction from Image Sequences with Variable Focal Lengths. ECCV, pp. 31-42, 1996.

[9] J.-V. Poncelet. Applications d'Analyse et de Géométrie - Traité des Propriétés Projectives des Figures. Tome I. Imprimerie de Mallet-Bachelier, Paris. 1862.

[10] G. Sparr and L. Nielsen. Shape and Mutual Cross-ratios with Applications to Exterior, Interior and Relative Orientation. ECCV, pp. 607-609, 1990.

[11] P. Sturm and S. Maybank. On Plane-Based Camera Calibration: a General Algorithm, Singularities, Applications. CVPR, pp. 432-437, 1999.

[12] P. Sturm. Algorithms for Plane-Based Pose Estimation. CVPR, pp. 1010-1017, 2000.

[13] B. Triggs. Autocalibration from Planar Scenes. ECCV, pp. 89-105, 1998.

[14] C. Zeller. Calibration projective, affine et euclidienne en vision par ordinateur et application à la perception tridimensionnelle. PhD Thesis, École Polytechnique, 1996.

[15] Z. Zhang. A Flexible New Technique for Camera Calibration. PAMI, vol. 22, no. 11, pp. 1330-1334. 2000.

[16] Z. Zhang. Camera Calibration with One-Dimensional Objects. ECCV, vol. 4, pp. 161-174, 2002.

IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 26, NO. 4, APRIL 2004 1

Nonlinear Estimation of the Fundamental Matrix with Minimal Parameters

Adrien Bartoli and
Peter Sturm, *Member*, *IEEE Computer Society*

Abstract—The purpose of this paper is to give a very simple method for nonlinearly estimating the fundamental matrix using the minimum number of seven parameters. Instead of minimally parameterizing it, we rather update what we call its orthonormal representation, which is based on its singular value decomposition. We show how this method can be used for efficient bundle adjustment of point features seen in two views. Experiments on simulated and real data show that this implementation performs better than others in terms of computational cost, i.e., convergence is faster, although methods based on minimal parameters are more likely to fall into local minima than methods based on redundant parameters.

Index Terms—Structure-from-motion, bundle adjustment, minimal parameterization, fundamental matrix.

1 Introduction

THE fundamental matrix has received a great interest in the computer vision community, see, e.g., [5], [6], [11], [12], [20], [23], [24]. It encapsulates the epipolar geometry or the projective motion between two uncalibrated perspective cameras and can be used for 3D reconstruction, motion segmentation, self-calibration, etc. Accurately estimating the fundamental matrix is therefore a major research issue. Most of the time, point correspondences between the two images are used. A linear solution is obtained using the 8-point algorithm [5], [11] optionally embedded in a robust estimation scheme [20], [23]. This estimate is then nonlinearly refined by minimizing a physically meaningful criterion that may involve reconstructed 3D point coordinates as well (in particular for bundle adjustment). However, nonlinearly estimating the fundamental matrix suffers from the lack of a simple technique to represent it efficiently. This paper, which is an extension of [2], provides such a technique in Section 3, based on the orthonormal representation of the fundamental matrix that we introduce. We show in Section 4 how this method can be used to refine the fundamental matrix by bundle adjustment of point features. We demonstrate experimentally in Sections 5.1 and 5.2 that the resulting algorithm performs better than existing ones in terms of computational cost.

2 Notations and Relation to Previous Work

The fundamental matrix denoted as F is a homogeneous (i.e., defined up to scale) (3×3) rank-2 matrix. It therefore has nine entries, but only 7 degrees of freedom.

There have been many attempts to minimally parameterize it, i.e., to represent it with seven parameters. Most of the previous works deal with directly parameterizing the epipolar geometry. The fundamental matrix F is decomposed into the epipoles $\mathbf{e}$ and $\mathbf{e}'$ and the epipolar transformation, which is a 1D projective transformation relating the epipolar pencils, represented by a homogeneous (2×2) matrix g [4], [12], [23].

• *The authors are with INRIA Rhone-Alpes, 655 avenue de l'Europe, 38334 Saint Ismier cedex, France. E-mail: {Adrien.Bartoli, Peter.Sturm}@inria.fr.*

Manuscript received 30 Apr. 2002; revised 26 Jan. 2003; accepted 19 July 2003.
Recommended for acceptance by M. Irani.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 116437.

Representing these entities with minimal parameters requires eliminating their arbitrary scale factors. This can be done by fixing, e.g., the 2-norm of homogeneous entities, but then the parameterization would not be minimal. Another solution is to freeze one entry of each homogeneous entity (in practice, the largest entry), which yields three possibilities for each epipole and four for the epipolar transformation, so $3 \cdot 3 \cdot 4 = 36$ possible parameterizations.

In [12], the authors propose to restrict the two-view configurations considered to the cases where both epipoles are finite and can therefore be expressed in affine coordinates. Consequently, this parameterization can be used only when both epipoles do not lie at infinity. Due to the homogeneity of the epipolar transformation, four distinct parameterizations are still necessary for g. A total of four parameterizations are then needed to represent this restricted set of fundamental matrices.

The method has been extended in [23] to the general case, i.e., when the epipoles can be either finite or infinite. In this case, it is shown that all 36 distinct parameterizations are necessary. This leads to a cumbersome and error-prone implementation of the optimization process.

Note that there are nine different possibilities to form the fundamental matrix—or any other 2D entity such as the extended epipolar transformation [4] or the canonic plane homography $\mathrm{H}^{\star}$ [13]—from $\mathbf{e}$, $\mathbf{e}'$, and g [23].

In [4], [24], the method has been revised so as to reduce the number of parameterizations using image transformations. In [4], the image transformations used are metric and the number of distinct parameterizations is restricted to three plus one bilinear constraint on the entries of g, while, in [24], the transformations used are projective, which allows one to reduce the number of parameterizations to one. The main drawback is that in the transformed image space, the original noise model on the image features is not preserved. A means to preserve it, up to first order approximation, has been proposed in [24] for the gradient-weighted criterion, which is not the one used for bundle adjustment.

Another solution is the point-based parameterization of [19]. The idea is to represent the fundamental matrix by a set of 7-point correspondences. Minimal optimization can then be conducted by varying one coordinate for each point correspondence. The fundamental matrix is obtained at each minimization step by computing the standard 7-point solution, which means that the null-space of a (7×9) matrix has to be computed and a cubic equation has to be solved. There may be up to three solutions. The one giving the lowest residual error is kept. The disadvantage of this parameterization is that it is costly to obtain the fundamental matrix given its parameters (i.e., the 7-point correspondences). Also, analytic differentiation is not possible.

3 Nonlinear Optimization with Seven Parameters

In contrast to the existing work, we do not try to represent the entire set of fundamental matrices using seven parameters. We rather locally update it with seven parameters. Before going further, we illustrate this idea by considering the case of the nonlinear estimation of 3D rotations, which is simpler and, as will be seen later, has similarities with the case of the fundamental matrix.

3.1 The Case of 3D Rotations

There exist many representations of 3D rotations, see, e.g., [18], including Euler angles, the Gibbs vector, Cayley-Klein parameters, Pauli spin matrices, axis-and-angle systems, $SO(3)$ matrices,[1] and

1. $SO(3)$ is the Lie group of (3×3) matrices R satisfying $\mathrm{R}^{\top}\mathrm{R} = \mathrm{I}$ and $\det \mathrm{R} = 1$.

unit quaternions. None of these representations is able to uniquely represent all 3D rotations with the minimum three parameters. For that reason, the following scheme is often used for their nonlinear estimation, see, e.g., [1], [7], [21]. The rotation is represented by an $SO(3)$ matrix R and is locally updated using three parameters by any well-behaved (locally nonsingular) representation, such as three Euler angles $\boldsymbol{\theta}^\top = (\theta_1\ \theta_2\ \theta_3)$ as:

$$\mathrm{R} \leftarrow \mathrm{R}\,\mathrm{R}(\boldsymbol{\theta}), \tag{1}$$

where $\mathrm{R}(\boldsymbol{\theta}) = \mathrm{R}_x(\theta_1)\,\mathrm{R}_y(\theta_2)\,\mathrm{R}_z(\theta_3)$ is the $SO(3)$ matrix representation of the 3D rotation corresponding to $\boldsymbol{\theta}$ with

$$\mathrm{R}_x(\theta_1) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_1 & -\sin\theta_1 \\ 0 & \sin\theta_1 & \cos\theta_1 \end{pmatrix},$$
$$\mathrm{R}_y(\theta_2) = \begin{pmatrix} \cos\theta_2 & 0 & \sin\theta_2 \\ 0 & 1 & 0 \\ -\sin\theta_2 & 0 & \cos\theta_2 \end{pmatrix},$$
$$\mathrm{R}_z(\theta_3) = \begin{pmatrix} \cos\theta_3 & -\sin\theta_3 & 0 \\ \sin\theta_3 & \cos\theta_3 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

At the end of each iteration, R is updated and $\boldsymbol{\theta}$ is reset to zero. Hence, at each iteration, the estimated Euler angles are small (initialized as zero), which makes this representation nonsingular.

3.2 Minimal Update

Following the example of 3D rotations, we propose the *orthonormal representation* of the fundamental matrix where more parameters than degrees of freedom are needed, but that can be easily updated using the minimum seven parameters.

Given an estimate of the fundamental matrix F obtained using, e.g., the 8-point algorithm, consider its singular value decomposition $\mathrm{F} \sim \mathrm{U}\Sigma\mathrm{V}^\top$, where U and V are $O(3)$ matrices[2] and Σ a diagonal one containing the singular values of F. Since F has rank 2, $\Sigma \sim \mathrm{diag}(\sigma_1, \sigma_2, 0)$, where $\sigma_1 \geq \sigma_2 > 0$ [22]. We can scale Σ such that $\mathrm{F} \sim \mathrm{U}\,\mathrm{diag}(1, \sigma, 0)\,\mathrm{V}^\top$, where $\sigma = \sigma_2/\sigma_1$ ($\sigma_1 \neq 0$ since $\mathrm{F} \neq 0$) and $1 \geq \sigma > 0$.

This decomposition shows that any fundamental matrix can be represented by $(\mathrm{U}, \mathrm{V}, \sigma)$, i.e., two $O(3)$ matrices and a scalar, which form what we call its orthonormal representation. Note that, in the case $\sigma = 1$, i.e., when the fundamental matrix is an essential matrix [8], the orthonormal representation is not unique (see below).

The orthonormal representation is consistent in that it yields $3 + 3 + 1 = 7$ degrees of freedom. The fundamental matrix can be recovered as:

$$\mathrm{F} \sim \mathbf{u}_1\mathbf{v}_1^\top + \sigma\mathbf{u}_2\mathbf{v}_2^\top, \tag{2}$$

where $\mathbf{u}_i$ and $\mathbf{v}_i$ are the columns of U and V, respectively.

This representation suggests the following update scheme. Each $O(3)$ matrix can be updated using an $SO(3)$ matrix, using (1) as in the case of 3D rotations, while σ can be included as such into the optimization:

$$\mathrm{U} \leftarrow \mathrm{U}\,\mathrm{R}(\mathbf{x}) \quad \mathrm{V} \leftarrow \mathrm{V}\,\mathrm{R}(\mathbf{y}) \quad \sigma \leftarrow \sigma + \delta_\sigma. \tag{3}$$

Here, $\mathbf{x}$ and $\mathbf{y}$ are 3-vectors of Euler angles. Intuitively, the orthonormal representation should be intrinsically well-conditioned since U and V are $O(3)$ matrices.

Completeness. A first question that immediately follows about the above-proposed method is whether all two-view configurations are covered. Clearly, any fundamental matrix can be decomposed into two $O(3)$ matrices and a scalar. The question arises from the fact that U and V are $O(3)$ matrices, which may have positive or negative determinants, and are updated using $SO(3)$ matrices, $\mathrm{R}(\mathbf{x})$ and $\mathrm{R}(\mathbf{y})$, respectively, which have positive determinants. Actually, this is not a problem since the signs of U and V can be freely switched, which accordingly switches the signs of their determinants, while leaving the corresponding F invariant: $\mathrm{F} \sim (\pm\mathrm{U})\,\Sigma(\pm\mathrm{V})^\top$.

Ensuring bounds on σ. A second remark is about the bounds on σ: $0 < \sigma \leq 1$. There are several possibilities to ensure them while leaving the corresponding F invariant. However, we have found during our experiments that, in practice, this does not affect the behavior of the underlying optimization process.

Essential matrices. As pointed out previously, in the case of $\sigma = 1$, where the fundamental matrix considered is an essential matrix, the proposed orthonormal representation is not unique: If U and V represent F, then also $\mathrm{U}\,\mathrm{R}_z(\alpha)$ and $\mathrm{V}\,\mathrm{R}_z(\alpha)$ for any α. This induces that the Jacobian matrix (6) has rank 6, as shown in Section 4.3. We propose two ways to deal with this singularity.

First, one can use a nonlinear optimization technique that handles singular parameterizations, e.g., damped Newton-type techniques. Using Levenberg-Marquardt, we found in our experiments that the singularity does not induce numerical instabilities.

Second, one can avoid singular configurations by properly normalizing the image points. Indeed, an essential matrix arises usually from a semicalibrated configuration where the origin of the coordinate frame in the image lies close to the principal point and where the image coordinates have been scaled by approximately the inverse focal length. In practice, the principal point position is unknown, but it is likely to be close to the image center. Hence, singular configurations can be avoided by translating the origin of the coordinate frame off the image center.

4 Bundle Adjustment

In this section, we show how the orthonormal representation can be used for bundle adjustment of point features $\mathbf{q}_i \leftrightarrow \mathbf{q}'_i$, $i \in 1 \ldots m$ seen in two views, through the minimization of the reprojection error. Similar results can be derived for other criteria, such as the minimization of the distances between points and epipolar lines or the gradient-weighted criterion [12], [23]. However, in order to obtain the maximum likelihood estimate of the fundamental matrix, one has also to estimate corrected point positions $\widehat{\mathbf{q}}_i \leftrightarrow \widehat{\mathbf{q}}'_i$, i.e., which satisfy exactly the epipolar geometry and, therefore, correspond to 3D points $\mathbf{Q}_i$.

Bundle adjustment consists in minimizing a cost function described in Section 4.1 over structure and motion parameters. In projective space, there are 15 inherent degrees of gauge freedom, due to the coordinate-frame ambiguity. In [9], a general framework consisting in incorporating gauge constraints up to first order in numerical estimation is introduced. The method of [15] falls in that category. Another technique is to let the gauge be free to drift, sometimes partially, while it is ensured that it does not move too far at each iteration. These methods are compared to ours in Section 5.

When the motion is represented by the fundamental matrix, the gauge is completely eliminated. We call any pair of camera matrices P and P′ a *realization*. In Section 4.2, we give analytical formulae to compute a realization from the orthonormal representation of F (as opposed to [12], [19], [23], [24]).

The algorithm is summarized in Table 1.

4.1 Cost Function

Bundle adjustment consists in solving the following optimization problem, see e.g., [15], [21], [23]: $\min_{\mathbf{a},\mathbf{b}} \sum_j r_j^2$, where $\mathbf{a}$ and $\mathbf{b}$ are respectively motion and structure parameters (or parameters used to update them), $\mathbf{r}$ is the $4m$-vector of residual errors defined by:

2. $O(3)$ is the Lie group of (3×3) matrices R satisfying $\mathrm{R}^\top\mathrm{R} = \mathrm{I}$.

TABLE 1
Implementing Our Minimal Estimator within the Bundle Adjustment Levenberg-Marquardt-Based Framework Given in [7, p. 574] (Algorithm A4.1)

Two-view projective bundle adjustment expressed within the framework of [7, p.574] (algorithm A4.1). The initial guess of the fundamental matrix is F_0.

Add the following steps:

(i') Initialize the orthonormal representation $(\mathsf{U}, \mathsf{V}, \sigma)$ by a scaled singular value decomposition of F_0: $\boxed{\mathsf{F}_0 \sim \mathsf{U}\,\mathrm{diag}(1, \sigma, 0)\,\mathsf{V}^\mathsf{T}}$.

(ii') Turn the full $(r \times 12)$ camera Jacobian matrix $\mathsf{A} = \bar{\mathsf{A}}$ into the minimal $(r \times 7)$ Jacobian matrix of the orthonormal representation: $\boxed{\mathsf{A} \leftarrow \mathsf{A}\,\mathsf{A}^\perp}$, where $\mathsf{A}^\perp$ is given by equations (6,7).

Change the parameter update step as:

(viii) Update the orthonormal representation as:

$$\boxed{\mathsf{U} \leftarrow \mathsf{U}\,\mathsf{R}(\mathbf{x})} \quad \boxed{\mathsf{V} \leftarrow \mathsf{V}\,\mathsf{R}(\mathbf{y})} \quad \boxed{\sigma \leftarrow \sigma + \delta_\sigma},$$

where $\boldsymbol{\delta}_a^\mathsf{T} = (\mathbf{x}^\mathsf{T}\ \mathbf{y}^\mathsf{T}\ \delta_\sigma)$ are the 7 motion update parameters, update the structure parameters by adding the incremental vector $\boldsymbol{\delta}_b$ and compute the new residual vector.

Note that r is the number of residuals and that the second projection matrix has to be extracted from the orthonormal representation using (5) (e.g., for computing the residual vector).

$$\mathbf{r}_{(4m\times 1)}^\top = \big(\ldots \quad \mathbf{q}_{i1} - \widehat{\mathbf{q}}_{i1} \quad \mathbf{q}_{i2} - \widehat{\mathbf{q}}_{i2} \quad \mathbf{q}'_{i1} - \widehat{\mathbf{q}}'_{i1} \quad \mathbf{q}'_{i2} - \widehat{\mathbf{q}}'_{i2} \quad \ldots\big),$$

where $\widehat{\mathbf{q}}_i \sim \mathrm{P}\mathbf{Q}_i$ and $\widehat{\mathbf{q}}'_i \sim \mathrm{P}'\mathbf{Q}_i$ are predicted image points.

4.2 Computing a Realization

Due to the projective frame ambiguity, there exists a 15-parameter family of realizations for a given fundamental matrix. A common choice is the canonic projection matrices given by [13]:

$$\mathrm{P} \sim (\mathrm{I}_{(3\times 3)}\ 0_{(3\times 1)}) \text{ and } \mathrm{P}' \sim (\mathrm{H}^\star\ \gamma\mathbf{e}'), \tag{4}$$

where $\mathbf{e}'$ is the second epipole, given by the left null-vector of F, $\mathrm{F}^\top\mathbf{e}' \sim 0_{(3\times 1)}$, and $\mathrm{H}^\star \sim [\mathbf{e}']_\times \mathrm{F}$ is the canonic plane homography [13]. The arbitrary scalar γ fixes the relative scale between $\mathrm{H}^\star$ and $\mathbf{e}'$. Without loss of generality, we assume that $\gamma = \|\mathbf{e}'\| = 1$. Any other realization can then be obtained by postmultiplying P and P′ by a nonsingular 3D homography.

Computing the canonic projection matrices (4) can be achieved directly from the orthonormal representation of F. The second epipole is the last column of U: $\mathbf{e}' \sim \mathbf{u}_3$ ($\|\mathbf{u}_3\| = 1$), so the canonic plane homography can be formulated as:

$$\mathrm{H}^\star \sim [\mathbf{e}']_\times \mathrm{F} \sim [\mathbf{u}_3]_\times \big(\mathbf{u}_1\mathbf{v}_1^\top + \sigma\mathbf{u}_2\mathbf{v}_2^\top\big).$$

Since U is an $O(3)$ matrix, $[\mathbf{u}_3]_\times \mathbf{u}_1 = \pm\mathbf{u}_2$ and $[\mathbf{u}_{3_\times}\mathbf{u}_2 = \mp\mathbf{u}_1$ which yields $\mathrm{H}^\star \sim \mathbf{u}_2\mathbf{v}_1^\top - \sigma\mathbf{u}_1\mathbf{v}_2^\top$ and, thus, the particularly simple and direct form of the second projection matrix:

$$\mathrm{P}' \sim \big(\mathbf{u}_2\mathbf{v}_1^\top - \sigma\mathbf{u}_1\mathbf{v}_2^\top \mid \mathbf{u}_3\big). \tag{5}$$

4.3 Analytical Differentiation

Many nonlinear optimization methods necessitate computing the Jacobian matrix $\mathrm{J} = (\mathrm{A} \mid \mathrm{B})$ of the residual vector r with respect to motion and structure parameters $\mathbf{a}$ and $\mathbf{b}$. While this can be achieved numerically using, e.g., finite differences [16], it may be better to use an analytical form for both computational efficiency and numerical accuracy. We focus on the computation of $\mathrm{A} = \frac{\partial \mathbf{r}}{\partial \mathbf{a}}$ since $\mathrm{B} = \frac{\partial \mathbf{r}}{\partial \mathbf{b}}$ only depends upon structure parameterization. Let $\mathbf{p}' = \mathrm{vect}(\mathrm{P}')$, where $\mathrm{vect}(.)$ is the row-wise vectorization. We decompose $\mathrm{A}_{(4m\times 7)} = \frac{\partial \mathbf{r}}{\partial \mathbf{p}'}\frac{\partial \mathbf{p}'}{\partial \mathbf{a}} = \bar{\mathrm{A}}_{(4m\times 12)}\,\mathrm{A}^\perp_{(12\times 7)}$. Only the 12 entries of P′ are considered since P is fixed in the canonic reconstruction basis (4). The matrix $\bar{\mathrm{A}} = \frac{\partial \mathbf{r}}{\partial \mathbf{p}'}$ depends on the chosen realization of the fundamental matrix, i.e., on the coordinate frame employed. We have chosen the canonic projection matrices (4). This Jacobian matrix is employed directly for the overparameterization proposed in [6]. Deriving its analytical form is straightforward. We therefore concentrate on deriving a closed-form expression for $\mathrm{A}^\perp$. One of the advantages of the update rule (3) is that there exists a simple closed-form expression for $\mathrm{A}^\perp$. Nonlinear least squares with analytical differentiation can be applied based on $\mathrm{A}^\perp$.

Let us consider the orthonormal representation $(\mathrm{U}, \mathrm{V}, \sigma)$. The motion update parameters are minimal and defined by $\mathbf{a}^\top = (x_1\ x_2\ x_3\ y_1\ y_2\ y_3\ \sigma)$, where $\mathbf{x}^\top = (x_1\ x_2\ x_3)$ and $\mathbf{y}^\top = (y_1\ y_2\ y_3)$ are used to update U and V, respectively. Since U and V are updated with respect to the current estimate, $\mathrm{A}^\perp$ is evaluated at $(\mathrm{U}, \mathrm{V}, \sigma)$, i.e., at $\mathbf{a}^\top = \mathbf{a}_0^\top = (0_{(6\times 1)}^\top\ \sigma)$. Equation (5) is used to derive a closed-form expression of the second canonic projection matrix after updating. By expanding, differentiating and evaluating this expression at a_0, we obtain:

$$\mathrm{A}^\perp = \frac{\partial \mathbf{p}'}{\partial \mathbf{a}} = \left(\left(\frac{\partial \mathbf{p}'}{\partial x_1}\right) \cdots \left(\frac{\partial \mathbf{p}'}{\partial y_3}\right)\ \left(\frac{\partial \mathbf{p}'}{\partial \sigma}\right)\right), \tag{6}$$

where:

$$\begin{array}{rcl}
\partial\mathbf{p}'/\partial x_1 &=& \mathrm{vect}(\mathbf{u}_3\mathbf{v}_1^\top \mid -\mathbf{u}_2) \\
\partial\mathbf{p}'/\partial x_2 &=& \mathrm{vect}(\sigma\mathbf{u}_3\mathbf{v}_2^\top \mid \mathbf{u}_1) \\
\partial\mathbf{p}'/\partial x_3 &=& \mathrm{vect}(-\mathbf{u}_1\mathbf{v}_1^\top - \sigma\mathbf{u}_2\mathbf{v}_2^\top \mid \mathbf{0}_{3\times 1}) \\
\partial\mathbf{p}'/\partial y_1 &=& \mathrm{vect}(-\sigma\mathbf{u}_1\mathbf{v}_3^\top \mid \mathbf{0}_{3\times 1}) \\
\partial\mathbf{p}'/\partial y_2 &=& \mathrm{vect}(-\mathbf{u}_2\mathbf{v}_3^\top \mid \mathbf{0}_{3\times 1}) \\
\partial\mathbf{p}'/\partial y_3 &=& \mathrm{vect}(\mathbf{u}_2\mathbf{v}_2^\top + \sigma\mathbf{u}_1\mathbf{v}_1^\top \mid \mathbf{0}_{3\times 1}) \\
\partial\mathbf{p}'/\partial \sigma &=& \mathrm{vect}(-\mathbf{u}_1\mathbf{v}_2^\top \mid \mathbf{0}_{3\times 1}).
\end{array} \tag{7}$$

In the general case, $\mathrm{rank}(\mathrm{A}^\perp) = 7$, but when $\sigma = 1$, $\mathrm{rank}(\mathrm{A}^\perp) = 6$ since $\partial\mathbf{p}'/\partial x_3 + \partial\mathbf{p}'/\partial y_3 = 0$.

If the minimal method of, e.g., [23] were used, 36 different Jacobian matrices, one for each parameterization, would have to be derived.

4.4 Particular Configurations

The epipolar geometry can be decomposed as a pair of epipoles and the 3-degrees of freedom epipolar transformation [12], [23]. If one or two of these components are a priori known, it may be convenient to leave them invariant during optimization of the fundamental matrix. Such features are easily added to our estimation method, as follows.

Leaving an epipole invariant. Consider, e.g., the second epipole encapsulated in the orthonormal representation as the third column of U. The update $\mathrm{U} \leftarrow \mathrm{U}\,\mathrm{R}(\mathbf{x})$ does not affect $\mathbf{u}_3$ if $x_1 = x_2 = 0$. Therefore, freezing the left or the right epipole can be done by removing x_1, x_2 or y_1, y_2, respectively, from the estimation and updating as $\mathrm{U} \leftarrow \mathrm{U}\,\mathrm{R}_z(x_3)$ or $\mathrm{V} \leftarrow \mathrm{V}\,\mathrm{R}_z(y_3)$, respectively.

Leaving the epipolar transformation invariant. The epipoles are encapsulated by the x_1, x_2 and the y_1, y_2 update parameters. Hence, the 3 degrees of freedom of the epipolar transformation are contained in the remaining update parameters: x_3, y_3, and σ. Removing them from the optimization freezes the underlying epipolar transformation.

5 Experimental Results

We compare an algorithm based on the orthonormal representation to other algorithms. We use simulated and real data in Sections 5.1 and 5.2, respectively. Below, we give details about the compared methods, the measured quantities, the computation of an initial suboptimal solution for structure and motion, and the nonlinear optimization scheme we use.

Compared methods. We compare the following motion parameterizations:

- FREE directly optimizes the 24 entries of the camera matrices. The gauge is left free to drift. The $24 - 7 = 17$ extra parameters are the homogeneous factors of each camera matrix and the 15-dimensional projective basis.
- NORMALIZED [15] is similar to FREE, but the gauge is fixed since a normalized coordinate frame is used. This is done by renormalizing the reconstruction before each step of the nonlinear minimization and by including first-order gauge constraints into the minimization. The reconstruction basis, as well as the homogeneous scale of the camera matrices are constrained.
- PARFREE [6] partially fixes the gauge by optimizing only the entries of the second camera matrix, while keeping $\mathrm{P} \sim (\mathrm{I}\,0)$. The $12 - 7 = 5$ extra parameters are the homogeneous scale of the second camera matrix, the global scene scale, and the position of the plane at infinity.
- MAPS [3], [23] is a minimal parameterization based on multiple maps.
- ORTHO uses the orthonormal representation proposed in this paper.

Measured quantities. We measure two quantities characterisic of a bundle adjustment process, computational cost, i.e., CPU time to convergence and the error at convergence.

Structure parameterization. We use the structure parameterization proposed in [7] which consists in scaling the reconstructed points such that their third element is unity. The three remaining free elements are then optimized. Note that this parameterization can be used only when a canonical basis enforcing $\mathrm{P} \sim (\mathrm{I}\,0)$ is used. Therefore, methods FREE and NORMALIZED have their own structure parameterization: They optimize the four elements of each point.

Initialization. We compute an initial solution for the motion using the normalized 8-point algorithm [5]. Image point coordinates are standardized such that they lie in $[-1 \dots 1]$. Each point is reconstructed by minimizing its reprojection error.

Nonlinear optimization. We use the Levenberg-Marquardt technique with analytic differentiation. This is a damped Gauss-Newton method. Let J be the Jacobian matrix and $\mathrm{H} = \mathrm{J}^\top \mathrm{J}$ the Gauss-Newton approximation of the Hessian matrix. The damping consists in augmenting the normal equations $\mathrm{H}\boldsymbol{\delta} = -\mathrm{J}^\top \mathbf{r}$ to be solved at each iteration: $\mathrm{H} \leftarrow \mathrm{H} + \mathrm{W}(\lambda)$. The parameter $\lambda \in \mathbb{R}$ is tuned heuristically, as described in [7], [21]. We try two approaches for the step control strategy, i.e., the choice of matrix $\mathrm{W}(\lambda)$. First, in [21], the authors recommend $\mathrm{W}(\lambda) = \lambda \mathrm{I}$. This is the original idea of the Levenberg-Marquardt algorithm [10], [14]. This will be referred to as LM. Second, in [7], the authors recommend $\mathrm{W}(\lambda) = (1 + \lambda)\,\mathrm{diag}(\mathrm{H})$, i.e., multiply the diagonal entries of H by $1 + \lambda$. This strategy is recommended in [16] and is due to [17]. This will be referred to as SEBER.

Note that gauge freedoms cause $\mathrm{H} = \mathrm{J}^\top \mathrm{J}$ to be rank-deficient, but that the damped matrix is guaranteed to have full-rank. Hence, Levenberg-Marquardt iterations change both the actual estimated geometry as well as the gauge.

We take advantage of the sparse structure of H and J to efficiently solve the augmented normal equations, as described in [7], [21]. More precisely, the sparseness of the structure parameters is exploited, and the complexity of the computation is $\mathcal{O}(mp^3)$, where m is the number of points and p is the number of motion parameters. Hence, we can expect the computational cost for an iteration to be similar for all parameterizations when the number of points is very large, and to be very different when the number of points is low.

We stop the estimation when the difference between two consecutive residual errors is lower than a threshold ξ, chosen typically in the range $10^{-8} \leq \xi \leq 10^{-4}$.

5.1 Simulated Data

5.1.1 Experimental Setup

We simulate points lying in a cube with one meter side length, observed by two cameras looking at the center of the cube. The standard configuration is the following: The focal length of the cameras is 1,000 (expressed in number of pixels). They are situated 10 meters away from the center of the cube and the baseline between them is one meter. The number of simulated points is 50. We add a centered Gaussian noise on true point positions with a 2-pixel variance. The normal equations are augmented using method LM. Each parameter of the above-described setup is independently varied to compare the parameterizations in different situations. The results are averaged over 100 trials. Computing the median gives similar results.

5.1.2 Results

Fig. 1 shows the results. We observe that all methods have roughly the same accuracy, i.e., they give the same reprojection errors, up to small discrepancies. Further comments on these discrepancies are given in the next paragraph.

On the other hand, there are quite large discrepancies between the computational costs of each method. The methods that have the highest computational costs are NORM and FREE, followed by PARFREE. The minimal methods MAPS and ORTHO have the lowest computational cost, roughly the same. These discrepancies are explained by the fact that redundant methods have more unknowns to estimate than minimal ones. Solving the normal equations is therefore more expensive (see below). These observations are valid for other experiments (not shown here) where the focal length of the cameras is varied from 500 to 2,000 pixels and where the baseline is varied from one to three meters. We also conduct the same experiments while augmenting the normal equation using SEBER. The same observations as above are valid. The results for all methods, compared to the LM augmentation, are worse in terms of both computational cost and reprojection error, while the discrepancies between the different methods for the reprojection error are reduced.

We observe that, in our C implementation, the computational cost of each iteration is dominated by the resolution of the normal equations, whose size is directly linked to the number of parameters. We measure the computational cost of an iteration for the different parameterizations. As said above, the complexity is linear in the number of points and cubic in the number of motion parameters. For different numbers of points, we obtain the results shown in Table 2.

These results show that the differences in computational costs are largely dominated by the number of motion parameters. The discrepancies become smaller when the number of points increases beyond 10,000, which is very large in the case of structure from motion for two views.

5.1.3 Convergence

As said above, there are small discrepancies in the reprojection errors achieved by the different methods, see in particular Fig. 1a.

TABLE 2
Computation Time (Seconds) of an Iteration for Different Parameterizations

	number of points			
parameterization	10	100	1,000	10,000
ORTHO	0.0045	0.0251	0.2152	2.0658
MAPS	0.0046	0.0251	0.2151	2.0658
PARFREE	0.0056	0.0307	0.2591	2.0753
FREE	0.0120	0.0589	0.5729	2.3231
NORM	0.0130	0.0664	0.6791	2.4148

These small discrepancies are due to the fact that each parameterization may lead to a different local minimum of the cost function. To better characterize this phenomenon, we measure the rate of successful estimations for the different methods against the distance from the scene to the cameras. An estimation is successful if it is not improved by any of the other compared method. More precisely, let M and M' designate two methods and $\mathcal{E}_{\mathrm{M}}(\mathrm{M}')$ be the error achieved by method M initialized by the result of method M'. We define the success of an estimation made with method M as:

$$\mathrm{Success}(\mathrm{M}) \equiv \left[\forall \mathrm{M}' \neq \mathrm{M},\ |\mathcal{E}_{\mathrm{M}}(\mathrm{INIT}) - \mathcal{E}_{\mathrm{M}'}(\mathrm{M})| \leq \xi\right],$$

where ξ is the threshold used to stop the iterations. We obtain the results as shown in Table 3.

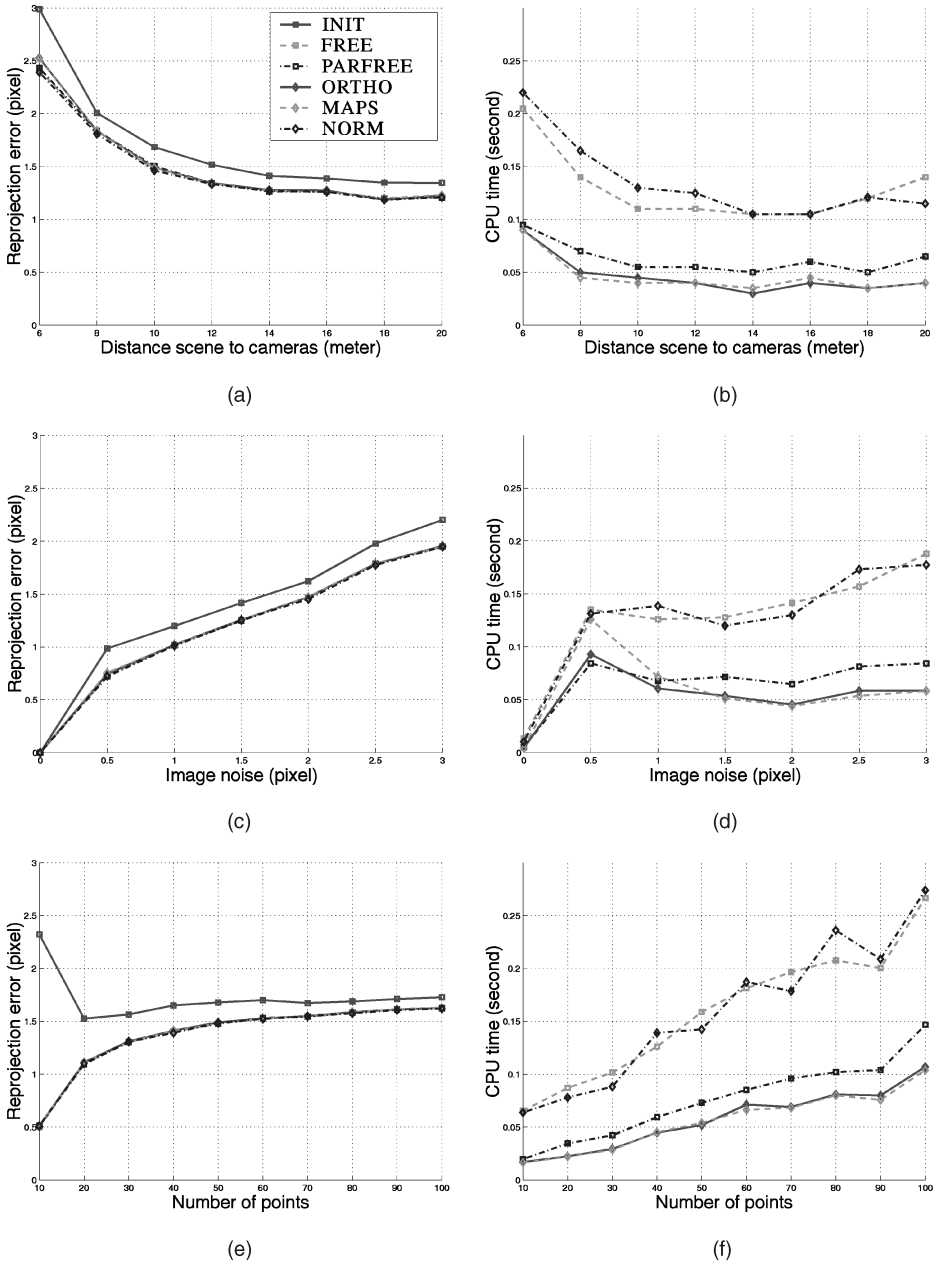


Fig. 1. Reprojection error (left column) and CPU time to convergence (right column) measured against different simulation parameters: distance scene to cameras (first row), image noise (second row), and number of points (third row). Concerning the reprojection error, the curves are almost always undistinguishable, apart from the initialization. For the CPU time, methods are divided into three groups: (from top to bottom) FREE and NORM, PARFREE, then MAPS and ORTHO.

TABLE 3
Convergence Results Shown as Success Rates in Percent

		distance scene to cameras (meter)							
	parameterization	6	8	10	12	14	16	18	20
minimal	ORTHO	83	97	97	100	100	98	100	99
	MAPS	88	95	96	100	100	99	98	99
redundent	PARFREE	100	100	94	100	100	100	100	100
	FREE & NORM	100	100	100	100	100	100	100	100

In the light of these results, we can say that methods using minimal parameters fall into local minima more often than methods based on redundant parameters. An explanation is that the minimal parameterizations are nonlinear, while the overparameterizations are linear, in the entries of the projection matrices. Hence, the local quadratic approximation of the cost fonction used in Levenberg-Marquardt is more accurate for overparameterizations.

5.1.4 Essential Matrix

As pointed out in Section 3.2, the orthonormal representation has a one-dimensional ambiguity when an essential matrix is considered. We want to check if, in the essential or near-essential cases, the orthonormal representation could induce numerical instabilities in the optimization process. For that purpose, we repeat the previous experiments, with the following two changes.

First, we map the fundamental matrix given by the 8-point algorithm to the closest essential matrix [8] and use this as an initial solution for the nonlinear optimization. Hence, the target epipolar geometry is a fundamental matrix, but the initial solution is an essential one.

Second, instead of using the coordinates of the points in the images, we use the coordinates of the points on the retina. Hence, the underlying true epipolar geometry is represented by an essential matrix. We run the experiments based on varying the geometry of the problem for both SEBER and LM.

We obtained results very similar to the previous experiments. This means that the orthonormal representation can be used for both fundamental and essential matrices, without inducing numerical instabilities, when an appropriate nonlinear optimizer is employed.

5.2 Real Data

We use different pairs of the images shown in Table 4, in order to cover all possibilities for the epipoles to be close to the images or at

TABLE 4
Reprojection Error at Convergence, $\mathcal{E}$, and CPU Time to Convergence, $\mathcal{T}$, Obtained When Combining Pairs of Images to Obtain Epipoles Close to the Images or Toward Infinity

(a) (b)

(c) (d)

epipoles		views	INIT		FREE		PARFREE		ORTHO		MAPS		NORM	
$\mathbf{e}$	$\mathbf{e}'$		$\mathcal{E}$	$\mathcal{T}$	$\mathcal{E}$	$\mathcal{T}$	$\mathcal{E}$	$\mathcal{T}$	$\mathcal{E}$	$\mathcal{T}$	$\mathcal{E}$	$\mathcal{T}$	$\mathcal{E}$	$\mathcal{T}$
~~∞~~	∞	$\mathcal{A}, \mathcal{B}$	0.49	-	0.47	0.99	0.47	0.54	0.47	0.64	0.47	0.65	0.47	1.10
		$\mathcal{A}, \mathcal{C}$	0.68	-	0.65	0.67	0.65	0.39	0.65	0.29	0.65	0.34	0.65	0.70
~~∞~~	~~∞~~	$\mathcal{A}, \mathcal{D}$	0.84	-	0.67	0.70	0.67	0.38	0.67	0.33	0.67	0.33	0.67	0.74
∞	∞	$\mathcal{B}, \mathcal{C}$	0.57	-	0.53	0.27	0.53	0.14	0.53	0.15	0.53	0.14	0.53	0.23
∞	~~∞~~	$\mathcal{B}, \mathcal{D}$	0.79	-	0.55	0.25	0.55	0.23	0.55	0.18	0.55	0.19	0.55	0.25
		$\mathcal{C}, \mathcal{B}$	0.57	-	0.53	0.30	0.53	0.10	0.53	0.12	0.53	0.21	0.53	0.20
average $\bar{\mathcal{E}}$ and $\bar{\mathcal{T}}$			0.66	-	0.57	0.53	0.57	0.30	0.57	0.28	0.57	0.31	0.57	0.54

infinity, with 60 point correspondences. The results are shown in Table 4. For each combination of images and each algorithm, we estimate the computational cost and the reprojection error. The last row of the table shows mean values for each algorithm over the set of image pairs. Note that, for any image pair, the reprojection error is the same for all algorithms. Methods ORTHO, PARFREE, and MAPS give the lowest computational costs, roughly twice as low as those of methods FREE and NORM. We obtain similar results using SEBER.

6 Conclusions

We studied the problem of estimating the fundamental matrix over a minimal set of seven parameters. We proposed the orthonormal representation which enables to easily update an estimate of the fundamental matrix using seven parameters. The canonic projection matrices can be directly extracted from the orthonormal representation. The method can be plugged into most of the (possibly sparse) nonlinear optimizers such as Levenberg-Marquardt. We gave a closed-form expression for the Jacobian matrix of the residuals with respect to the motion parameters for bundle adjustment purposes, necessary for Newton-type optimization techniques.

We conducted experiments on simulated and real data. Our conclusions are that the methods based on minimal parameter sets have lower computational cost, but may be more frequently trapped in local minima.

Acknowledgments

The authors would like to thank Bill Triggs for discussions and one of the anonymous reviewers for very useful comments.

References

[1] K.B. Atkinson, ed., *Close Range Photogrammetry and Machine Vision.* Whittles Publishing, 1996.
[2] A. Bartoli, "On the Non-Linear Optimization of Projective Motion Using Minimal Parameters," *Proc. Seventh European Conf. Computer Vision,* vol. 2, pp. 340-354, May 2002.
[3] A. Bartoli and P. Sturm, "Three New Algorithms for Projective Bundle Adjustment with Minimum Parameters," Research Report 4236, INRIA, Grenoble, France, Aug. 2001.
[4] A. Bartoli, P. Sturm, and R. Horaud, "Projective Structure and Motion from Two Views of a Piecewise Planar Scene," *Proc. Eighth Int'l Conf. Computer Vision,* vol. 1, pp. 593-598, July 2001.
[5] R. Hartley, "In Defence of the 8-Point Algorithm," *Proc. Fifth Int'l Conf. Computer Vision,* pp. 1064-1070, June 1995.
[6] R.I. Hartley, "Projective Reconstruction and Invariants from Multiple Images," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 16, no. 10, pp. 1036-1041, Oct. 1994.
[7] R.I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision.* Cambridge Univ. Press, June 2000.
[8] T.S. Huang and O.D. Faugeras, "Some Properties of the E Matrix in Two-View Motion Estimation," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 11, no. 12, pp. 1310-1312, Dec. 1989.
[9] K. Kanatani and D.D. Morris, "Gauges and Gauge Transformations for Uncertainty Description of Geometric Structure with Indeterminacy," *IEEE Trans. Information Theory,* vol. 47, no. 5, July 2001.
[10] K. Levenberg, "A Method for the Solution of Certain Non-Linear Problems in Least Squares," *Quarterly of Applied Math.,* pp. 164-168, 1944.
[11] H.C. Longuet-Higgins, "A Computer Program for Reconstructing a Scene from Two Projections," *Nature,* vol. 293, pp. 133-135, Sept. 1981.
[12] Q.T. Luong and O. Faugeras, "The Fundamental Matrix: Theory, Algorithms and Stability Analysis," *Int'l J. Computer Vision,* vol. 17, no. 1, pp. 43-76, 1996.
[13] Q.T. Luong and T. Vieville, "Canonic Representations for the Geometries of Multiple Projective Views," *Computer Vision and Image Understanding,* vol. 64, no. 2, pp. 193-229, 1996.
[14] D.W. Marquardt, "An Algorithm for Least-Squares Estimation of Nonlinear Parameters," *J. Soc. for Industrial and Applied Math.,* vol. 11, no. 2, pp. 431-441, June 1963.
[15] P.F. McLauchlan, "Gauge Invariance in Projective 3D Reconstruction," *Proc. Multi-View Workshop,* 1999.
[16] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery, *Numerical Recipes in C—The Art of Scientific Computing,* second ed. Cambridge Univ. Press, 1992.
[17] G.A.F. Seber and C.J. Wild, *Non-Linear Regression.* John Wiley & Sons, 1989.
[18] J. Stuelpnagel, "On the Parametrization of the Three-Dimensional Rotation Group," *SIAM Rev.,* vol. 6, no. 4, pp. 422-430, Oct. 1964.
[19] P. Torr and A. Zisserman, "MLESAC: A New Robust Estimator with Application to Estimating Image Geometry," *Computer Vision and Image Understanding,* vol. 78, no. 1, 2000.
[20] P.H.S. Torr, A. Zisserman, and S.J. Maybank, "Robust Detection of Degenerate Configurations While Estimating the Fundamental Matrix," *Computer Vision and Image Understanding,* vol. 71, no. 3, pp. 312-333, Sept. 1998.
[21] B. Triggs, P.F. McLauchlan, R.I. Hartley, and A. Fitzgibbon, "Bundle Ajustment—A Modern Synthesis," *Proc. Int'l Workshop Vision Algorithms: Theory and Practice,* B. Triggs, A. Zisserman, and R. Szeliski, eds., 2000.
[22] R.Y. Tsai and T.S. Huang, "Uniqueness and Estimation of Three-Dimensional Motion Parameters of Rigid Objects with Curved Surfaces," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 6, no. 1, pp. 13-27, Jan. 1984.
[23] Z. Zhang, "Determining the Epipolar Geometry and Its Uncertainty: A Review," *Int'l J. Computer Vision,* vol. 27, no. 2, pp. 161-195, Mar. 1998.
[24] Z. Zhang and C. Loop, "Estimating the Fundamental Matrix by Transforming Image Points in Projective Space," *Computer Vision and Image Understanding,* vol. 82, no. 2, pp. 174-180, May 2001.

Part III

Generic Camera Models and Unified Treatment of Structure from Motion

Chapter 6

Calibration

Paper 11 [32]: P. Sturm and S. Ramalingam. A generic concept for camera calibration. In T. Pajdla and J. Matas, editors, *Proceedings of the 8th European Conference on Computer Vision, Prague, Czech Republic*, volume 3022 of *Lecture Notes in Computer Science*, pages 1–13. Springer-Verlag, May 2004.

Paper 12 [17]: S. Ramalingam, P. Sturm, and S.K. Lodha. Towards complete generic camera calibration. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, San Diego, USA*, volume 1, pages 1093–1098, June 2005.

Paper 13 [16]: S. Ramalingam, P. Sturm, and S. Lodha. Theory and calibration algorithms for axial cameras. In *Proceedings of the Asian Conference on Computer Vision, Hyderabad, India*, volume I, pages 704–713, January 2006.

Paper 14 [36]: J.-P. Tardif and P. Sturm. Calibration of cameras with radially symmetric distortion. In *Proceedings of the 6th Workshop on Omnidirectional Vision, Camera Networks and Non-Classical Cameras, Beijing, China*, pages 44–51, October 2005.

A Generic Concept for Camera Calibration

Peter Sturm[1] and Srikumar Ramalingam[2]

[1] INRIA Rhône-Alpes, 38330 Montbonnot, France
Peter.Sturm@inrialpes.fr • http://www.inrialpes.fr/movi/people/Sturm/
[2] Dept. of Computer Science, University of California, Santa Cruz, CA 95064, USA

Abstract. We present a theory and algorithms for a generic calibration concept that is based on the following recently introduced general imaging model. An image is considered as a collection of pixels, and each pixel measures the light travelling along a (half-) ray in 3-space associated with that pixel. Calibration is the determination, in some common coordinate system, of the coordinates of all pixels' rays. This model encompasses most projection models used in computer vision or photogrammetry, including perspective and affine models, optical distortion models, stereo systems, or catadioptric systems – central (single viewpoint) as well as non-central ones. We propose a concept for calibrating this general imaging model, based on several views of objects with known structure, but which are acquired from unknown viewpoints. It allows in principle to calibrate cameras of any of the types contained in the general imaging model using one and the same algorithm. We first develop the theory and an algorithm for the most general case: a non-central camera that observes 3D calibration objects. This is then specialized to the case of central cameras and to the use of planar calibration objects. The validity of the concept is shown by experiments with synthetic and real data.

1 Introduction

We consider the camera calibration problem, i.e. the estimation of a camera's intrinsic parameters. A camera's intrinsic parameters (plus the associated projection model) give usually exactly the following information: for any point in the image, they allow to compute a ray in 3D along which light travels that falls onto that point (here, we neglect point spread).

Most existing camera models are parametric (i.e. defined by a few intrinsic parameters) and address imaging systems with a single effective viewpoint (all rays pass through one point). In addition, existing calibration procedures are taylor-made for specific camera models.

The aim of this work is to relax these constraints: we want to propose and develop a calibration method that should work for any type of camera model, and especially also for cameras without a single effective viewpoint. To do so, we first renounce on parametric models, and adopt the following very general model: a camera acquires images consisting of pixels; each pixel captures light that travels along a ray in 3D. The camera is fully described by:

- the coordinates of these rays (given in some local coordinate frame).
- the mapping between rays and pixels; this is basically a simple indexing.

This general imaging model allows to describe virtually any camera that captures light rays travelling along straight lines[3]. Examples (cf. figure 1):

- a camera with any type of optical distortion, such as radial or tangential.
- a camera looking at a reflective surface, e.g. as often used in surveillance, a camera looking at a spherical or otherwise curved mirror [10]. Such systems, as opposed to central catadioptric systems [3] composed of cameras and parabolic mirrors, do not in general have a single effective viewpoint.
- multi-camera stereo systems: put together the pixels of all image planes; they "catch" light rays that definitely do not travel along lines that all pass through a single point. Nevertheless, in the above general camera model, a stereo system (with rigidly linked cameras) is considered as a **single** camera.
- other acquisition systems, see e.g. [4, 14, 19], insect eyes, etc.

Relation to previous work. See [9, 17] for reviews and references on existing calibration methods and e.g. [6] for an example related to *central* catadioptric devices. A calibration method for certain types of *non-central* catadioptric cameras (e.g. due to misalignment of mirror), is given in [2].

The above imaging model has already been used, in more or less explicit form, in various works [8, 12–16, 19, 23–25], and is best described in [8], were also other issues than sensor geometry, e.g. radiometry, are discussed. There are conceptual links to other works: acquiring an image with a camera of our general model may be seen as sampling the plenoptic function [1], and a light field [11] or lumigraph [7] may be interpreted as a single image, acquired by a camera of an appropriate design.

To our knowledge, the only previously proposed calibration approaches for the general imaging model, are due to Swaminathan, Grossberg and Nayar [8, 22]. The approach in [8] requires the acquisition of two or more images of a calibration object with known structure, and knowledge of the camera or object motion between the acquisitions. In this work, we develop a completely general approach, that requires taking three or more images of calibration objects, from arbitrary and **unknown viewing positions**. The approach in [22] does not require calibration objects, but needs to know the camera motion. Calibration is formulated as a non-linear optimization problem. In this work, "closed-form" solutions are proposed (requiring to solve linear equation systems).

Other related works deal mostly with epipolar geometry estimation and modeling [13, 16, 24] and motion estimation for already calibrated cameras [12, 15].

Organization. In §2, we explain the camera model used and give some notations. For ease of explanation and understanding, the calibration concept is first introduced for 2D cameras, in §3. The general concept for 3D cameras is described in §4 and variants (central vs. non-central camera and planar vs. 3D calibration objects) are developed in §5. Some experimental results are shown in §6, followed by discussions and conclusions.

[3] However, it would not work for example with a camera looking from the air, into water: still, to each pixel is associated a refracted ray in the water. However, when the camera moves, the refraction effect causes the set of rays to move non-rigidly, hence the calibration would be different for each camera position.

2 Camera Model and Notations

We give the definition of the (purely geometrical) camera model used in this work. It is essentially the same as the model of [8] where in addition other issues such as point spread and radiometry are treated. We assume that a camera delivers images that consist of a set of pixels, where each pixel captures/measures the light travelling along some half-ray. In our calibration method, we do not model half-rays explicitly, but rather use their infinite extensions – **camera rays**. Camera rays corresponding to different pixels need not intersect – in this general case, we speak of **non-central cameras**, whereas if all camera rays intersect in a single point, we have a **central camera** with an **optical center**.

Furthermore, the physical location of the actual photosensitive elements that correspond to pixels, does in general not matter at all. On the one hand, this means that the camera ray corresponding to some pixel, needs not pass through that pixel, cf. figure 1. On the other hand, neighborship relations between pixels are in general not necessary to be taken into account: the set of a camera's photosensitive elements may lie on a single surface patch (image plane), but may also lie on a 3D curve, on several surface patches or even be placed at completely isolated positions. In practice however, we do use some continuity assumption, useful in the stage of 3D-2D matching, as explained in §6: we suppose that pixels are indexed by two integer coordinates like in traditional cameras and that camera rays of pixels with neighboring coordinates, are "close" to one another.

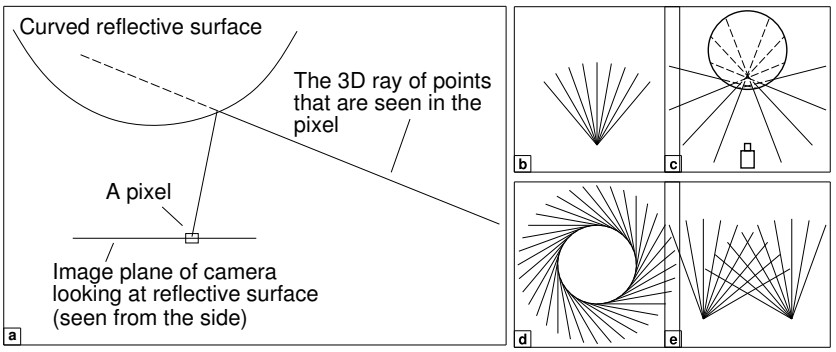


Fig. 1. Examples of imaging systems. (a) Catadioptric system. Note that camera rays do not pass through their associated pixels. (b) Central camera (e.g. perspective, with or without radial distortion). (c) Camera looking at reflective sphere. This is a non-central device (camera rays are not intersecting in a single point). (d) Omnivergent imaging system [14, 19]. (e) Stereo system (non-central) consisting of two central cameras.

3 The Calibration Concept for 2D Cameras

We consider here a camera and scene living in a 2D plane, i.e. camera rays are lines in that plane. Two images are acquired, while the imaged object undergoes some motion. Consider a single pixel and its camera ray, cf. figure 2. Figures 2 (b) and (c) show the two points on the object that are seen by that pixel in the two images. We suppose to be able to determine the coordinates of these two points, in some local coordinate frame attached to the object ("matching").

The case of known motion. If the object's motion between image acquisitions is known, then the two object points can be mapped to a single coordinate frame, e.g. the object's coordinate frame at its second position, as shown in figure 2 (d). Computing our pixel's camera ray is then simply done by joining the two points. This summarizes the calibration approach proposed by Grossberg and Nayar [8], applied here for the 2D case. Camera rays are thus initially expressed in a coordinate frame attached to the calibration object. This does not matter (all that counts are the relative positions of the rays), but for convenience, one would typically choose a better frame. For a central camera for example, one would choose the optical center as origin or for a non-central camera, the point that minimizes the sum of distances to the set of camera rays (if it exists).

Note that it is not required that the two images be taken of the same object; all that is needed is knowledge of point positions relative to coordinate frames of the objects, and the "motion" between the two coordinate frames.

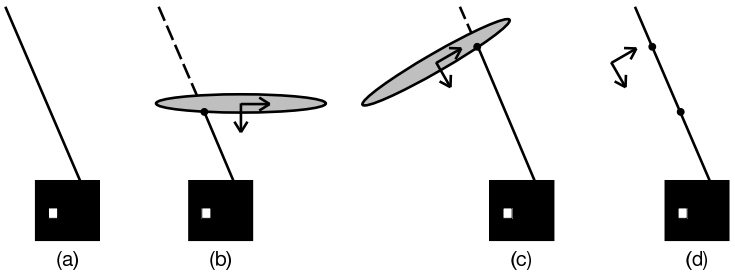


Fig. 2. (a) The camera as black box, with one pixel and the associated camera ray. (b) The pixel sees a point on a calibration object, whose coordinates are identified in a frame associated with the object. (c) Same as (b), for another position of the object. (d) Due to known motion, the two points on the calibration object can be placed in the same coordinate frame. The camera ray is then determined by joining them.

The case of unknown motion. This approach is no longer applicable and we need to estimate, implicitly or explicitly, the unknown motion. We show how to do this, given three images. Let $\mathbf{Q}, \mathbf{Q}'$ and $\mathbf{Q}''$ be the points on the calibration objects, that are seen in the same pixel. These are 3-vectors of homogeneous coordinates, expressed in the respective local coordinate frame. Without loss of generality, we choose the coordinate frame associated with the object's first position, as common frame. The unknown relative motions between the second and third frames and the first one, are given by 2×2 rotation matrices R' and R'' and translation vectors $\mathbf{t}'$ and $\mathbf{t}''$. Note that $R'_{11} = R'_{22}$ and $R'_{12} = -R'_{21}$ (same for R''). Mapping the calibration points to the common frame gives points

$$\mathbf{Q} \qquad \begin{pmatrix} \mathsf{R}' & \mathbf{t}' \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \mathbf{Q}' \qquad \begin{pmatrix} \mathsf{R}'' & \mathbf{t}'' \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \mathbf{Q}'' \quad .$$

They must lie on the pixel's camera ray, i.e. must be collinear. Hence, the determinant of the matrix composed of their coordinate vectors, must vanish:

$$\begin{vmatrix} Q_1 & R'_{11}Q'_1 + R'_{12}Q'_2 + t'_1Q'_3 & R''_{11}Q''_1 + R''_{12}Q''_2 + t''_1Q''_3 \\ Q_2 & R'_{21}Q'_1 + R'_{22}Q'_2 + t'_2Q'_3 & R''_{21}Q''_1 + R''_{22}Q''_2 + t''_2Q''_3 \\ Q_3 & Q'_3 & Q''_3 \end{vmatrix} = 0 \quad . \tag{1}$$

i	C_i	V_i
1	$Q_1Q_1'Q_3'' + Q_2Q_2'Q_3''$	R_{21}'
2	$Q_1Q_2'Q_3'' - Q_2Q_1'Q_3''$	R_{22}'
3	$Q_1Q_3'Q_1'' + Q_2Q_3'Q_2''$	$-R_{21}''$
4	$Q_1Q_3'Q_2'' - Q_2Q_3'Q_1''$	$-R_{22}''$
5	$Q_3Q_1'Q_1'' + Q_3Q_2'Q_2''$	$R_{11}'R_{21}'' - R_{11}''R_{21}'$
6	$Q_3Q_1'Q_2'' - Q_3Q_2'Q_1''$	$R_{11}'R_{22}'' - R_{12}''R_{21}'$

i	C_i	V_i
7	$Q_1Q_3'Q_3''$	$t_2' - t_2''$
8	$Q_2Q_3'Q_3''$	$-t_1' + t_1''$
9	$Q_3Q_1'Q_3''$	$R_{11}'t_2'' - R_{21}'t_1''$
10	$Q_3Q_2'Q_3''$	$R_{12}'t_2'' - R_{22}'t_1''$
11	$Q_3Q_3'Q_1''$	$R_{21}''t_1' - R_{11}''t_2'$
12	$Q_3Q_3'Q_2''$	$R_{22}''t_1' - R_{12}''t_2'$
13	$Q_3Q_3'Q_3''$	$t_1't_2'' - t_1''t_2'$

Table 1. Non-zero coefficients of the trifocal calibration tensor for a general 2D camera.

This equation is trilinear in the calibration point coordinates. The equation's coefficients may be interpreted as coefficients of a trilinear matching tensor; they depend on the unknown motions' coefficients, and are given in table 1. In the following, we sometimes call this the *calibration tensor*. It is somewhat related to the *homography tensor* derived in [18]. Among the $3 \cdot 3 \cdot 3 = 27$ coefficients of the calibration tensor, 8 are always zero and among the remaining 19, there are 6 pairs of identical ones. The columns of table 1 are interpreted as follows: the C_i are trilinear products of point coordinates and the V_i are the associated coefficients of the tensor. The following equation is thus equivalent to (1):

$$\sum_{i=1}^{13} C_i V_i = 0 \quad . \tag{2}$$

Given triplets of points $\mathbf{Q}, \mathbf{Q}'$ and $\mathbf{Q}''$ for at least 12 pixels, we may compute the trilinear tensor up to an unknown scale λ by solving a system of linear equations of type (2). Note that we have verified using simulated data, that we indeed can obtain a unique solution (up to scale) for the tensor. The main problem is then that of extractin the motion parameters from the calibration tensor. In [21] we give a simple algorithm for doing so[4]. Once the motions are determined, the approach described above can be readily applied to compute the camera rays and thus to finalize the calibration.

The special case of central cameras. It is worthwhile to specialize the calibration concept to the case of central cameras (but which are otherwise general, i.e. not perspective). A central camera can already be calibrated from two views. Let $\mathbf{Z}$ be the homogeneous coordinates of the optical center (in the frame associated with the object's first position). We have the following collinearity constraint:

$$\begin{vmatrix} Z_1 & Q_1 & R_{11}'Q_1' + R_{12}'Q_2' + t_1'Q_3' \\ Z_2 & Q_2 & R_{21}'Q_1' + R_{22}'Q_2' + t_2'Q_3' \\ Z_3 & Q_3 & Q_3' \end{vmatrix} = \mathbf{Q}'^{\mathsf{T}} \begin{pmatrix} R_{21}'Z_3 & -R_{22}'Z_3 & R_{22}'Z_2 - R_{21}'Z_1 \\ R_{22}'Z_3 & R_{21}'Z_3 & -R_{21}'Z_2 - R_{22}'Z_1 \\ Z_3t_2' - Z_2 & Z_1 - Z_3t_1' & Z_2t_1' - Z_1t_2' \end{pmatrix} \mathbf{Q} = 0$$

The bifocal calibration tensor in this equation is a 3×3 matrix and somewhat similar to a fundamental or essential matrix. It can be estimated linearly from calibration points associated with 8 pixels or more. It is of rank 2 and its right null vector is the optical center $\mathbf{Z}$, which is thus easy to compute. Once this is done, the camera ray for a pixel can be determined e.g. by joining $\mathbf{Z}$ and $\mathbf{Q}$.

[4] This is similar, though more complicated than extracting (ego-)motion of perspective cameras from the classical essential matrix [9].

The special case of a linear calibration object. This is equally worthwhile to investigate. We propose an algorithm in [21], which works but is more complicated than the algorithm for general calibration objects.

4 Generic Calibration Concept for 3D Cameras

This and the next section describe our main contributions. We extend the concept described in §3 to the case of cameras living in 3-space. We first deal with the most general case: non-central cameras and 3D calibration objects.

In case of **known motion**, two views are sufficient to calibrate, and the procedure is equivalent to that outlined in §3, cf. [8]. In the following, we consider the practical case of **unknown motion**. Input are now, for each pixel, three 3D points $\mathbf{Q}, \mathbf{Q}'$ and $\mathbf{Q}''$, given by 4-vectors of homogeneous coordinates, relative to the calibration object's local coordinate system. Again, we adopt the coordinate system associated with the first image as global coordinate frame. The object's motion for the other two images is given by 3×3 rotation matrices R' and R'' and translation vectors $\mathbf{t}'$ and $\mathbf{t}''$. With the correct motion estimates, the aligned points must be collinear. We stack their coordinates in the following 4×3 matrix:

$$\begin{pmatrix} Q_1 & R'_{11}Q'_1 + R'_{12}Q'_2 + R'_{13}Q'_3 + t'_1Q'_4 & R''_{11}Q''_1 + R''_{12}Q''_2 + R''_{13}Q''_3 + t''_1Q''_4 \\ Q_2 & R'_{21}Q'_1 + R'_{22}Q'_2 + R'_{23}Q'_3 + t'_2Q'_4 & R''_{21}Q''_1 + R''_{22}Q''_2 + R''_{23}Q''_3 + t''_2Q''_4 \\ Q_3 & R'_{31}Q'_1 + R'_{32}Q'_2 + R'_{33}Q'_3 + t'_3Q'_4 & R''_{31}Q''_1 + R''_{32}Q''_2 + R''_{33}Q''_3 + t''_3Q''_4 \\ Q_4 & Q'_4 & Q''_4 \end{pmatrix} . \quad (3)$$

The collinearity constraint means that this matrix must be of rank less than 3, which implies that all sub-determinants of size 3×3 vanish. There are 4 of them, obtained by leaving out one row at a time. Each of these corresponds to a trilinear equation in point coordinates and thus to a trifocal calibration tensor whose coefficients depend on the motion parameters.

Table 2 gives the coefficients of the first two calibration tensors (all 4 are given in the appendix of [21]). For both, 34 out of 64 coefficients are always zero. One may observe that the two tensors share some coefficients, e.g. $V_8 = W_1 = R'_{31}$.

The tensors can be estimated by solving linear equation system, and we verified using simulated random experiments that in general unique solutions (up to scale) are obtained, if 3D points for sufficiently many pixels (29 at least) are available. In the following, we give an algorithm for computing the motion parameters. Let $V'_i = \lambda V_i$ and $W'_i = \mu W_i, i = 1 \ldots 37$ be the estimated tensors (up to scale). The algorithm proceeds as follows.

1. Estimate scale factors: $\lambda = \sqrt{V'^2_8 + V'^2_9 + V'^2_{10}}$ and $\mu = \sqrt{W'^2_1 + W'^2_2 + W'^2_3}$.
2. Compute $V_i = \frac{V'_i}{\lambda}$ and $W_i = \frac{W'_i}{\mu}, i = 1 \ldots 37$
3. Compute R' and R'':

$$\mathsf{R}' = \begin{pmatrix} -W_{15} & -W_{16} & -W_{17} \\ -V_{15} & -V_{16} & -V_{17} \\ V_8 & V_9 & V_{10} \end{pmatrix} \qquad \mathsf{R}'' = \begin{pmatrix} W_{18} & W_{19} & W_{20} \\ V_{18} & V_{19} & V_{20} \\ -V_{11} & -V_{12} & -V_{13} \end{pmatrix} .$$

   They will not be orthonormal in general. We "correct" this as shown in [21].
4. Compute $\mathbf{t}'$ and $\mathbf{t}''$ by solving a straightforward linear least squares problem, which is guaranteed to have a unique solution, see [21] for details.

Using simulations, we verified that the algorithm gives a unique and correct solution in general.

i	C_i	V_i	W_i
1	$Q_1Q_1'Q_4''$	0	R_{31}'
2	$Q_1Q_2'Q_4''$	0	R_{32}'
3	$Q_1Q_3'Q_4''$	0	R_{33}'
4	$Q_1Q_4'Q_1''$	0	$-R_{31}''$
5	$Q_1Q_4'Q_2''$	0	$-R_{32}''$
6	$Q_1Q_4'Q_3''$	0	$-R_{33}''$
7	$Q_1Q_4'Q_4''$	0	$t_3' - t_3''$
8	$Q_2Q_1'Q_4''$	R_{31}'	0
9	$Q_2Q_2'Q_4''$	R_{32}'	0
10	$Q_2Q_3'Q_4''$	R_{33}'	0
11	$Q_2Q_4'Q_1''$	$-R_{31}''$	0
12	$Q_2Q_4'Q_2''$	$-R_{32}''$	0
13	$Q_2Q_4'Q_3''$	$-R_{33}''$	0
14	$Q_2Q_4'Q_4''$	$t_3' - t_3''$	0
15	$Q_3Q_1'Q_4''$	$-R_{21}'$	$-R_{11}'$
16	$Q_3Q_2'Q_4''$	$-R_{22}'$	$-R_{12}'$
17	$Q_3Q_3'Q_4''$	$-R_{23}'$	$-R_{13}'$
18	$Q_3Q_4'Q_1''$	R_{21}''	R_{11}''
19	$Q_3Q_4'Q_2''$	R_{22}''	R_{12}''

i	C_i	V_i	W_i
20	$Q_3Q_4'Q_3''$	R_{23}''	R_{13}''
21	$Q_3Q_4'Q_4''$	$t_2'' - t_2'$	$t_1'' - t_1'$
22	$Q_4Q_1'Q_1''$	$R_{21}'R_{31}'' - R_{21}''R_{31}'$	$R_{11}'R_{31}'' - R_{11}''R_{31}'$
23	$Q_4Q_1'Q_2''$	$R_{21}'R_{32}'' - R_{22}''R_{31}'$	$R_{11}'R_{32}'' - R_{12}''R_{31}'$
24	$Q_4Q_1'Q_3''$	$R_{21}'R_{33}'' - R_{23}''R_{31}'$	$R_{11}'R_{33}'' - R_{13}''R_{31}'$
25	$Q_4Q_1'Q_4''$	$R_{21}'t_3'' - R_{31}'t_2''$	$R_{11}'t_3'' - R_{31}'t_1''$
26	$Q_4Q_2'Q_1''$	$R_{22}'R_{31}'' - R_{21}''R_{32}'$	$R_{12}'R_{31}'' - R_{11}''R_{32}'$
27	$Q_4Q_2'Q_2''$	$R_{22}'R_{32}'' - R_{22}''R_{32}'$	$R_{12}'R_{32}'' - R_{12}''R_{32}'$
28	$Q_4Q_2'Q_3''$	$R_{22}'R_{33}'' - R_{23}''R_{32}'$	$R_{12}'R_{33}'' - R_{13}''R_{32}'$
29	$Q_4Q_2'Q_4''$	$R_{22}'t_3'' - R_{32}'t_2''$	$R_{12}'t_3'' - R_{32}t_1''$
30	$Q_4Q_3'Q_1''$	$R_{23}'R_{31}'' - R_{21}''R_{33}'$	$R_{13}'R_{31}'' - R_{11}''R_{33}'$
31	$Q_4Q_3'Q_2''$	$R_{23}'R_{32}'' - R_{22}''R_{33}'$	$R_{13}'R_{32}'' - R_{12}''R_{33}'$
32	$Q_4Q_3'Q_3''$	$R_{23}'R_{33}'' - R_{23}''R_{33}'$	$R_{13}'R_{33}'' - R_{13}''R_{33}'$
33	$Q_4Q_3'Q_4''$	$R_{23}'t_3'' - R_{33}'t_2''$	$R_{13}'t_3'' - R_{33}'t_1''$
34	$Q_4Q_4'Q_1''$	$R_{31}''t_2' - R_{21}''t_3'$	$R_{31}''t_1' - R_{11}''t_3'$
35	$Q_4Q_4'Q_2''$	$R_{32}''t_2' - R_{22}''t_3'$	$R_{32}''t_1' - R_{12}''t_3'$
36	$Q_4Q_4'Q_3''$	$R_{33}''t_2' - R_{23}''t_3'$	$R_{33}''t_1' - R_{13}''t_3'$
37	$Q_4Q_4'Q_4''$	$t_2't_3'' - t_3't_2''$	$t_1't_3'' - t_1''t_3'$

Table 2. Coefficients of two trifocal calibration tensors for a general 3D camera.

5 Variants of the Calibration Concept

Analogously to the case of 2D cameras, cf. §3, we developed important specializations of our calibration concept, for central cameras and planar calibration objects. We describe them very briefly; details are given in [21].

Central cameras. In this case, two images are sufficient. Let $\mathbf{Z}$ be the optical center (unknown). By proceeding as in §3, we obtain 4 bifocal calibration tensors of size 4×4 and rank 2, that are somewhat similar to fundamental matrices. One of them is shown here:

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ R_{31}'Z_4 & R_{32}'Z_4 & R_{33}'Z_4 & -Z_3 + Z_4t_3' \\ -R_{21}'Z_4 & -R_{22}'Z_4 & -R_{23}'Z_4 & Z_2 - Z_4t_2' \\ R_{21}'Z_3 - R_{31}'Z_2 & R_{22}'Z_3 - R_{32}'Z_2 & R_{23}'Z_3 - R_{33}'Z_2 & Z_3t_2' - Z_2t_3' \end{pmatrix} .$$

It is relatively straightforward to extract the motion parameters and the optical center from these tensors.

Non-central cameras and planar calibration objects. The algorithm for this case is rather more complicated and not shown here. Using simulations, we proved that we obtain a unique solution in general.

Central cameras and planar calibration objects. As with non-central cameras, we already obtain constraints on the motion parameters (and the optical center) from two views of the planar object. In this case however, the associated calibration tensors do not contain sufficient information in order to uniquely estimate the motion and optical center. This is not surprising: even in the very restricted case of **perspective** cameras with 5 intrinsic parameters, two views of a planar calibration object do not suffice for calibration [20, 26]. We thus developed an algorithm working with three views [21]. It is rather complicated, but was shown to provide unique solutions in general.

6 Experimental Evaluation

As mentioned previously, we verified each algorithm using simulated random experiments. This was first done using noiseless data. We also tested our methods using noisy data and obtained satisfying results. A detailled quantitative analysis remains yet to be carried out.

We did various experiments with real images, using a 3M-Pixel digital camera with moderate optical distortions, a camera with a fish-eye lens and "home-made" catadioptric systems consisting of a digital camera and various curved off-the-shelf mirrors. We used planar calibration objects consisting of black dots or squares on white paper. Figure 3 shows three views taken by the digital camera.

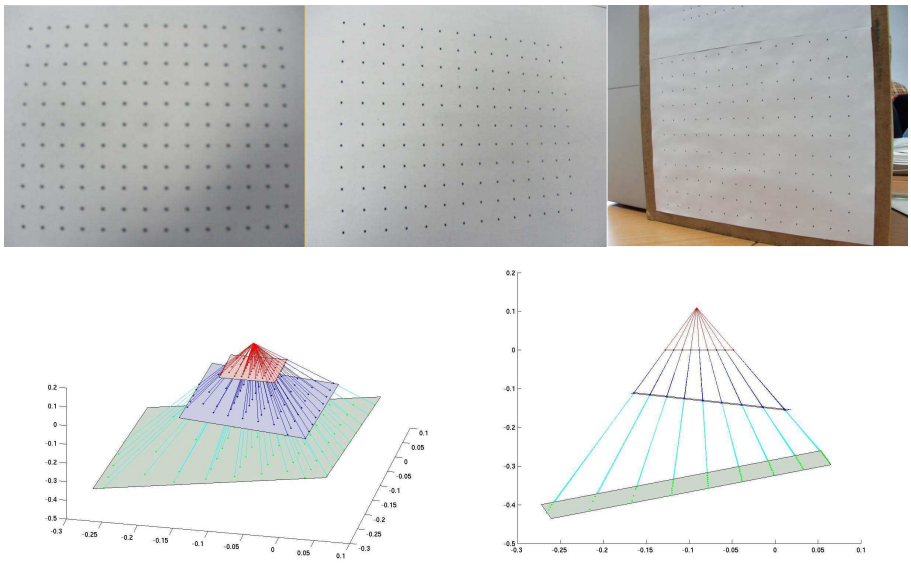


Fig. 3. Top: images of 3 boards of different sizes, captured by a digital camera. Bottom: two views of the calibrated camera rays and estimated pose of the calibration boards.

Dots/corners were extracted using the Harris detector. Matching of these image points to points on calibration objects was done semi-automatically. This gives calibration points for a sparse set of pixels per image, and in general there will be few, if any, pixels for which we get a calibration point in every view! We thus take into account the continuity assumption mentioned in §2. For every image, we compute the convex hull of the pixels for which calibration points were extracted. We then compute the intersection of the convex hulls over all three views, and henceforth only consider pixels inside that region. For every such pixel in the first image we estimate the calibration points for the second and third images using the following interpolation scheme: in each of these images, we determine the 4 closest extracted calibration points. We then compute the homography between these pixels and the associated calibration points on the planar object. The calibration point for the pixel of interest is then computed using that homography.

On applying the algorithm for central cameras (cf. §5), we obtained the results shown in figure 3. The bottom row shows the calibrated camera rays and the pose of the calibration objects, given by the estimated motion parameters. It is difficult to evaluate the calibration quantitatively, but we observe that for every pixel considered, the estimated motion parameters give rise to nearly perfectly collinear calibration points. Note also, cf. the bottom right figure, that radial distortion is correctly modeled: the camera rays are setwise coplanar, although the corresponding sets of pixels in the image are not perfectly collinear.

The same experiment was performed for a fish-eye lens, cf. figure 4. The result is slightly worse – aligned calibration points are not always perfectly collinear. This experiment is preliminary in that only the central image region has been calibrated (cf. figure 4), due to the difficulty of placing planar calibration objects that cover the whole field of view.

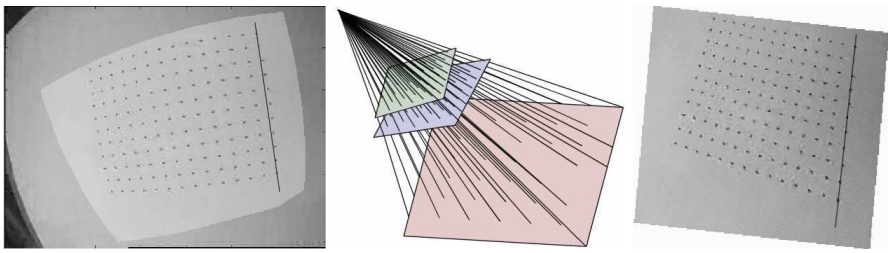


Fig. 4. Left: one of 3 images taken by the fish-eye lens (in white the area that was calibrated). Middle: calibrated camera rays and estimated pose of calibration objects. Right: image from the left after distortion correction, see text.

Using the calibration information, we carried out two sample applications, as described in the following. The first one consists in correcting non-perspective distortions: calibration of the central camera model gives us a bunch of rays passing through a single point. We may cut these rays by a plane; at each intersection with a camera ray, we "paint" the plane with the "color" observed by the pixel

associated with the ray in some input image. Using the same homography-based interpolation scheme as above, we can thus create a "densely" colored plane, which is nothing else than the image plane of a distortion-corrected perspective image. See figure 4 for an example. This model-free distortion correction scheme is somewhat similar to the method proposed in [5].

Another application concerns (ego-) motion and epipolar geometry estimation. Given calibration information, we can estimate relative camera pose (or motion), and thus epipolar geometry, from two or more views of an unknown object. We developed a motion estimation method similar to [15] and applied it to two views taken by the fish-eye lens. The epipolar geometry of the two views can be computed and visualized as follows: for a pixel in the first view, we consider its camera ray and determine all pixels of the second view whose rays (approximately) intersect the first ray. These pixels form the "epipolar curve" associated with the original pixel. An example is shown in figure 5. The estimated calibration and motion also allow of course to reconstruct objects in 3D (see [21] for examples).

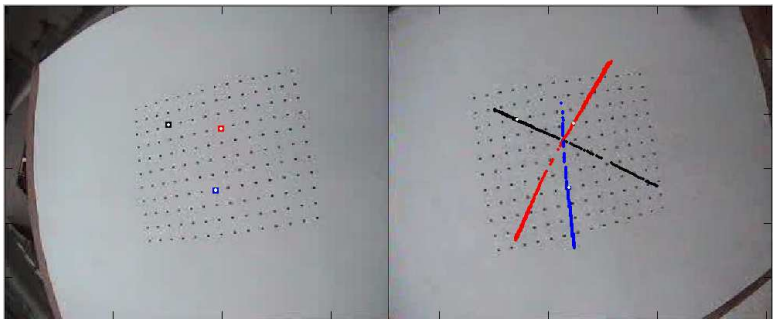


Fig. 5. Epipolar curves for three points. These are not straight lines, but intersect in a single point, since we here use the central camera model.

7 Discussion

The algorithm for central cameras seems to work fine, even with the minimum input of 3 views and a planar calibration object. Experiments with non-central catadioptric cameras however did so far not give satisfying results. One reason for poor stability of the non-central method is the way we currently obtain our input (homography-based interpolation of calibration points). We also think that the general algorithm, which is essentially based on solving linear equations, can only give stable results with minimum input (3 views) if the considered camera is clearly non-central. By this, we mean that there is not any point that is "close" to all camera rays; the general algorithm does not work for perspective cameras, but for multi-stereo systems consisting of sufficiently many cameras[5].

[5] Refer to the appendix of [21] on the feasibility of the general calibration method for stereo systems consisting of three or more central cameras.

We propose several ideas for overcoming these problems. Most importantly, we probably need to use several to many images for a stable calibration. We have developed bundle adjustment formulations for our calibration problem, which is not straightforward: the camera model is of discrete nature and does not directly allow to handle sub-pixel image coordinates, which are for example needed in derivatives of a reprojection error based cost function. For initialization of the non-central bundle adjustment, we may use the (stabler) calibration results for the central model. Model selection may be applied to determine if the central or non-central model is more appropriate for a given camera. Another way of stabilizing the calibration might be the possible inclusion of constraints on the set of camera rays, such as rotational or planar symmetry, if appropriate.

Although we have a single algorithm that works for nearly all existing camera types, different cameras will likely require different designs of calibration objects, e.g. panoramic cameras vs. ones with narrow field of view. We stress that a single calibration can use images of different calibration objects; in our experiments, we actually use planar calibration objects of different sizes for the different views, imaged from different distances, cf. figure 3. This way, we can place them such that they do not "intersect" in space, which would give less stable results, especially for camera rays passing close to the intersection region. We also plan to use different calibration objects for initialization and bundle adjustment: initialization, at least for the central model, can be performed using the type of calibration object used in this work. As for bundle adjustment, we might then switch to objects with a much denser "pattern" e.g. with a coating consisting of randomly distributed colored speckles. Another possibility is to use a flat screen to produce a dense set of calibration points [8].

One comment on the difference between calibration and motion estimation: here, with 3 views of a *known* scene, we solve simultaneously for motion and calibration (motion is determined explicitly, calibration implicitly). Whereas once a (general) camera is calibrated, (ego-)motion can already be estimated from 2 views of an *unknown* scene [15]. Hence, although our method estimates motion directly, we consider it a calibration method.

8 Conclusions

We have proposed a theory and algorithms for a highly general calibration concept. As for now, we consider this mainly as a conceptual contribution: we have shown how to calibrate nearly any camera, using one and the same algorithm.

We already propose specializations that may be important in practice: an algorithm for central, though otherwise unconstrained cameras, is presented, as well as an algorithm for the use of planar calibration objects. Results of preliminary experiments demonstrate that the approach allows to calibrate central cameras without using any parametric distortion model.

We believe in our concept's potential for calibrating cameras with "exotic" distortions – such as fish-eye lenses with hemispheric field of view or catadioptric cameras, especially non-central ones. We are working towards that goal, by developing bundle adjustment procedures to calibrate from multiple images,

and by designing better calibration objects. These issues could bring about the necessary stability to really calibrate cameras without any parametric model in practice. Other ongoing work concerns the extension of classical structure-from-motion tasks such as motion and pose estimation and triangulation, from the perspective to the general imaging model.

References

1. E.H. Adelson, J.R. Bergen. The Plenoptic Function and the Elements of Early Vision. *Computational Models of Visual Processing*, MIT Press, 1991.
2. D.G. Aliaga. Accurate Catadioptric Calibration for Real-time Pose Estimation in Room-size Environments. *ICCV*, 127-134, 2001.
3. S. Baker, S. Nayar. A Theory of Catadioptric Image Formation. *ICCV*, 1998.
4. H. Bakstein, T. Pajdla. An overview of non-central cameras. *Proceedings of Computer Vision Winter Workshop*, Ljubljana, Slovenia, 2001.
5. P. Brand. Reconstruction tridimensionnelle d'une scène à partir d'une caméra en mouvement.PhD Thesis, Université Claude Bernard, Lyon, October 1995.
6. C. Geyer, K. Daniilidis. Paracatadioptric Camera Calibration. *PAMI*, 2002.
7. S.J. Gortler et al.The Lumigraph. *SIGGRAPH*, 1996.
8. M.D. Grossberg, S.K. Nayar. A general imaging model and a method for finding its parameters. *ICCV*, 2001.
9. R.I. Hartley, A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.
10. R.A. Hicks, R. Bajcsy. Catadioptric Sensors that Approximate Wide-angle Perspective Projections. *CVPR*, pp. 545-551, 2000.
11. M. Levoy, P. Hanrahan. Light field rendering. *SIGGRAPH*, 1996.
12. J. Neumann, C. Fermüller, Y. Aloimonos. Polydioptric Camera Design and 3D Motion Estimation. *CVPR*, 2003.
13. T. Pajdla. Stereo with oblique cameras. *IJCV*, 47(1), 2002.
14. S. Peleg, M. Ben-Ezra, Y. Pritch. OmniStereo: Panoramic Stereo Imaging. *PAMI*, pp. 279-290, March 2001.
15. R. Pless. Using Many Cameras as One. *CVPR*, 2003.
16. S. Seitz. The space of all stereo images. *ICCV*, 2001.
17. C.C. Slama (editor). *Manual of Photogrammetry*. Fourth Edition, ASPRS, 1980.
18. A. Shashua, L. Wolf. Homography Tensors: On Algebraic Entities That Represent Three Views of Static or Moving Planar Points. *ECCV*, 2000.
19. H.-Y. Shum, A. Kalai, S.M. Seitz. Omnivergent Stereo. *ICCV*, 1999.
20. P. Sturm, S. Maybank. On Plane-Based Camera Calibration. *CVPR*, 1999.
21. P. Sturm, S. Ramalingam. A Generic Calibration Concept: Theory and Algorithms. Research Report 5058, INRIA, France, 2003.
22. R. Swaminathan, M.D. Grossberg, and S.K. Nayar. Caustics of Catadioptric Cameras. *ICCV*, 2001.
23. R. Swaminathan, M.D. Grossberg, S.K. Nayar. A perspective on distortions. *CVPR*, 2003.
24. Y. Wexler, A.W. Fitzgibbon, A. Zisserman. Learning epipolar geometry from image sequences. *CVPR*, 2003.
25. D. Wood et al.Multiperspective panoramas for cell animation. *SIGGRAPH*, 1997.
26. Z. Zhang. A flexible new technique for camera calibration. *PAMI*, 22(11), 2000.

Towards Complete Generic Camera Calibration

Srikumar Ramalingam[1], Peter Sturm[2], and Suresh K. Lodha[1]
[1] Dept. of Computer Science, University of California, Santa Cruz, CA 95064, USA
[2] INRIA Rhône-Alpes, GRAVIR-CNRS, 38330 Montbonnot, France

Abstract

We consider the problem of calibrating a highly generic imaging model, that consists of a non-parametric association of a projection ray in 3D to every pixel in an image. Previous calibration approaches for this model do not seem to be directly applicable for cameras with large fields of view and non-central cameras. In this paper, we describe a complete calibration approach that should in principle be able to handle any camera that can be described by the generic imaging model. Initial calibration is performed using multiple images of overlapping calibration grids simultaneously. This is then improved using pose estimation and bundle adjustment-type algorithms. The approach has been applied on a wide variety of central and non-central cameras including fisheye lens, catadioptric cameras with spherical and hyperbolic mirrors, and multi-camera setups. We also consider the question if non-central models are more appropriate for certain cameras than central models.

1. Introduction

This paper is about camera calibration. We adopt a general non-parametric imaging model that consists in associating one projection ray to each individual pixel. By projection ray we refer to the 3D (half-) line along which light travels that falls onto the pixel (here, we neglect point spread and the finite spatial extent of a pixel). Rays may be unconstrained, i.e. they may not intersect in a single point, in which case the camera is called *non-central*. This general model has been used in various works [7, 12, 14, 15, 16, 19, 20, 22, 23, 25, 26], and is best described in [7], where properties other than geometric ones are also considered.

By adopting this model, one may formulate "black-box calibration" and provide algorithms that allow to calibrate any camera (see figure 1 for examples), be it of pinhole type (with or without optical distortions), catadioptric [2, 10], pushbroom [8], or some other acquisition system [15, 20]. Such calibration algorithms have been proposed in [3, 6, 7, 22]. In this paper, we adopt the approach of [22] which allows to perform calibration from three images of a calibration grid, without having to know the motion between the images. To calibrate the complete image with only three images, one would need a calibration grid of appropriate dimensions and shape; especially for omnidirectional cameras (fisheye, catadioptric, etc), this will be cumbersome to produce and handle. In this paper, we propose a similar method, that uses multiple images to accurately and completely calibrate large fields of view.

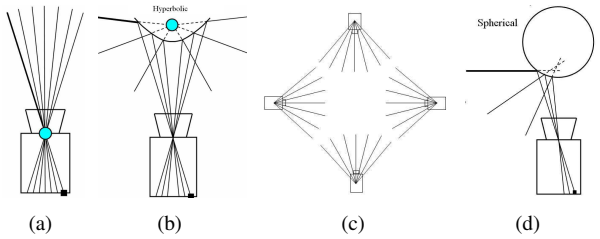


Figure 1. Examples of generic imaging model. (a) pinhole camera, (b) catadioptric with hyperbolic mirror (central), (c) multi-camera, (d) catadioptric with spherical mirror (non-central).

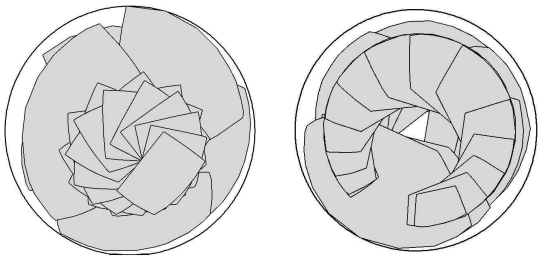


Figure 2. Examples of complete calibration. Left: 23 calibration grids, used in calibrating a fisheye. Right: 24 calibration grids used in calibrating a spherical catadioptric camera.

Our approach works as follows. An initial calibration is done with images of calibration grids that present sufficient overlap. We then recursively incorporate additional images: at each step, we select the image that has the largest overlap with the already calibrated image region. We show how to compute the pose of the associated calibration grid. Then, given the pose, one may compute projection rays for previously uncalibrated pixels, thus enlarging the calibrated image region. This is iterated until all images have been used. We also propose a bundle adjustment method that can be used at any stage of the procedure. This approach and the underlying methods are developed for both, non-central and central models, although the central case is described in more detail here. Besides developing algorithms, we are also interested in the question if for certain cameras it is worth going to a full non-central model, cf. also [1, 11].

This paper is organized as follows. The calibration approach is described in §2 and some variants are proposed in §3. Practical issues are discussed in §4. Experimental results are presented in §5, followed by conclusions in §6.

2. Complete Calibration

We first provide an overview of complete generic camera calibration. We take several images of a calibration grid such as to cover the entire image region. Then, matching between image pixels and points on the calibration grids is performed. From such matches, we then compute the pose of each of these grids in a common coordinate system. After this pose computation, a 3D projection ray is computed for each pixel, as follows. For all grid points matching a given pixel, we compute their 3D coordinates (via the pose of the grids). The pixel's projection ray is then simply computed by fitting a straight line to the associated grid points.

For a non-central camera, atleast two grid points per pixel are of course required. If the camera is (assumed to be) central however, a single grid point is enough: as will be seen later, the above stage of pose computation also comprises the estimation of the camera's optical center (in the same coordinate frame as the grids' pose). Thus, we compute projection rays by fitting lines to 3D points, but which are constrained to contain the optical center.

In the following, we describe different parts of our approach in more detail. In this section, we describe the case of central cameras. For conciseness, the non-central case is described more briefly in §3.1. First, we show how to use the images of multiple grids simultaneously, to compute grid pose and the optical center. It is then shown how to compute the pose of additional grids. Refinement of calibration after each step, through bundle adjustment, is then discussed in §2.3.

2.1 Calibration using Multiple Grids

Our goal is to obtain the poses of multiple calibration grids w.r.t. a common coordinate system. Let B_i denote the image region covered by the i_{th} calibration grid, for $i = 1 \cdots n$. Let $\cup$ and $\cap$ refer to union and intersection operations respectively. The calibration algorithm is applied to a partial region given by $\cup_{i=2}^{n}(B_1 \cap B_i)$. Once the poses are computed the calibration is extended to a larger region given by $\cup_{i=1}^{n} B_i$.

We now outline the theory behind calibration using multiple grids. Consider one pixel and its associated grid points, with homogeneous coordinates $Q^i = (Q_1^i, Q_2^i, Q_3^i, Q_4^i)^T$, for grids $i = 1 \cdots n$. In the following, we consider planar calibration grids, and thus suppose that $Q_3^i = 0$. Let the unknown grid poses be represented by rotation matrices R^i and translation vectors t^i, such that the point Q^i, given in local grid coordinates, is mapped to global coordinates via

$$\begin{pmatrix} R^i & t^i \\ 0^T & 1 \end{pmatrix} \begin{pmatrix} Q_1^i \\ Q_2^i \\ 0 \\ Q_4^i \end{pmatrix} \tag{1}$$

Furthermore, let $O = (O_1, O_2, O_3, 1)$ be the coordinates of the camera's optical center. As global coordinate system, we adopt, without loss of generality, the reference frame of the first grid, i.e. $R^1 = I$ and $t^1 = 0$.

We now show how to estimate the unknown grid poses and the optical center. This is based on the following collinearity constraint: with the correct poses, the grid points associated with one pixel, after mapping into the global coordinate system via (1), must be collinear, and in addition, collinear with the optical center. This is because all these points must lie on the pixel's projection ray, i.e. a straight line. Algebraically, this collinearity constraint can be formulated as follows. Consider the matrix containing the coordinates of the collinear points:

$$\begin{pmatrix} O_1 & Q_1^1 & R_{11}^2 Q_1^2 + R_{12}^2 Q_2^2 + t_1^2 Q_4^2 & \cdots \\ O_2 & Q_2^1 & R_{21}^2 Q_1^2 + R_{22}^2 Q_2^2 + t_2^2 Q_4^2 & \cdots \\ O_3 & 0 & R_{31}^2 Q_1^2 + R_{32}^2 Q_2^2 + t_3^2 Q_4^2 & \cdots \\ 1 & Q_4^1 & Q_4^2 & \cdots \end{pmatrix} \tag{2}$$

The collinearity of these points implies that this $4 \times (n+1)$ matrix must be of rank smaller than 3. Consequently, the determinants of all its 3×3 submatrices must vanish. This gives equations linking calibration point coordinates and the unknowns (camera poses and optical center). On using the first column (optical center) and two other columns with Q^j and Q^k to form a submatrix, we get bilinear equations in terms of calibration point coordinates Q^j and Q^k. Hence, we may write the equations in the form:

$$\left(Q^j\right)^T T_{3\times 3} Q^k = 0 \tag{3}$$

This matrix T (a bifocal matching tensor), depends on camera pose and optical center, in a way specific to which 3×3 submatrix of (2) is considered. Using (3), we estimate such tensors T from available correspondences. Since 3×3 submatrices can be obtained by removing one row and $n-2$ columns at a time, we have $4 \times \binom{n+1}{3}$ possible matching tensors T. However, using simulations we observed that not all of them can be estimated uniquely from point matches. Let $T_{ijk;i'j'k'}$ represent the tensor corresponding to the submatrix with rows (i, j, k) and columns (i', j', k'). In the following, we use $2 \times (n-1)$ constraints of the form $T_{x34;12y}, (x = 1, 2;\ y = 3 \cdots n)$ for calibration, i.e. constraints combining the optical center and the first grid, with the other grids. For these tensors, the equation (3) takes the following form: $\sum_{i=1}^{9} C_i^y V_i^y = 0$ and $\sum_{i=1}^{9} C_i^y W_i^y = 0$ for $T_{134;12y}$ and $T_{234;12y}$ respectively. Here, $C_i^y = Q_j^1 Q_k^y$, for appropriate indices j, as shown in Table 1.

V_i^y and W_i^y are computed up to scale using least squares. Note that they share some coefficients (e.g. $R_{3,1}^y$), hence they can be estimated up to the same scale factor, λ_y. We perform this step for $(n-1)$ constraints by choosing $y = 3 \cdots n$. We now combine all the coupled variables

Table 1. Tensors $T_{134;12y}$ and $T_{234;12y}$ for a central camera.

i	j	k	V_i^y	W_i^y
1	1	1	0	$R^y_{3,1}$
2	1	2	0	$R^y_{3,2}$
3	1	4	0	$-O_3 + t^y_3$
4	2	1	$R^y_{3,1}$	0
5	2	2	$R^y_{3,2}$	0
6	2	4	$-O_3 + t^y_3$	0
7	4	1	$-O_2 R^y_{3,1} + O_3 R^y_{2,1}$	$-O_1 R^y_{3,1} + O_3 R^y_{1,1}$
8	4	2	$-O_2 R^y_{3,2} + O_3 R^y_{2,2}$	$-O_1 R^y_{3,2} + O_3 R^y_{1,2}$
9	4	4	$-O_2 t^y_3 + O_3 t^y_2$	$-O_1 t^y_3 + O_3 t^y_1$

contained in the different tensors, to obtain the following system which links the pose variables of all the grids.

$$\begin{bmatrix} H^2_{6\times 2} & J_{6\times 6} & .. & 0_{6\times 6} \\ .. & .. & .. & .. \\ H^n_{6\times 2} & 0_{6\times 6} & .. & J_{6\times 6} \end{bmatrix} \begin{bmatrix} -O_1 \\ -O_2 \\ X^2_{6\times 1} \\ .. \\ X^n_{6\times 1} \end{bmatrix} = \begin{bmatrix} Y^2_{6\times 1} \\ .. \\ Y^n_{6\times 1} \end{bmatrix}, \quad (4)$$

$$H^i = \begin{bmatrix} 0 & V^i_4 \\ 0 & V^i_5 \\ 0 & V^i_6 \\ V^i_4 & 0 \\ V^i_5 & 0 \\ V^i_6 & 0 \end{bmatrix}, J = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$X^i = \begin{bmatrix} \lambda_i O_3 (t^i_1 - O_1) \\ \lambda_i O_3 (t^i_2 - O_2) \\ \lambda_i O_3 R^i_{1,1} \\ \lambda_i O_3 R^i_{1,2} \\ \lambda_i O_3 R^i_{2,1} \\ \lambda_i O_3 R^i_{2,2} \end{bmatrix}, Y^i = \begin{bmatrix} V^i_7 \\ V^i_8 \\ V^i_9 \\ W^i_7 \\ W^i_8 \\ W^i_9 \end{bmatrix}$$

We rewrite equation (4) as follows:

$$A_{6(n-1)\times(2+6(n-1))} Z_{2+6(n-1)} = Y_{6(n-1)}$$

Since A is of rank $6(n-1)$, we obtain the $(2+6(n-1))$ variables ($Z's$) up to a linear combination of three vectors. The coefficients of the linear combination are computed using orthogonality constraints on rotation matrices R^i. More details are given in [18].

Using the definition of Z, it is possible to compute the pose variables uniquely except for a sign ambiguity in n variables: there are two mirror solutions for each grid's pose (they can lie on either side of the optical center). In the case of a pinhole camera we can resolve this ambiguity by applying the constraint that the grids must lie on the same side of the optical center. However this constraint becomes difficult to apply for omnidirectional cameras where the grids essentially get distributed around the center. We apply the following technique. First we arbitrarily select one solution for the first grid's pose. Then we identify the correct location of each of the other grids by minimizing their distance with an already fixed grid, with which it has some overlap. This is easily achieved because we usually collect images in succession and not in a completely random order.

Having determined the pose of grids and the optical center, we now compute projection rays for all pixels that have at least one matching point in one of the grids used here.

2.2 Pose Estimation of Additional Grids

We suppose here that a partial calibration of the camera has been performed with the method of the previous section. The calibration is partial because only grids whose projection in the image had some overlap with one of the grids ("the first grid") were used. In order to make the calibration complete, we use the pose estimation technique, described in our earlier work [17], to include additional grids, which do not have any overlap with the first grid, but with some of the others. A 4th degree polynomial equation is solved to compute the pose and the correct solution is identified from the ambiguous ones as given in [9].

2.3 Bundle Adjustment

We use bundle adjustment [24] to refine the pose of all grids (except for the first one) and the projection rays. During bundle adjustment, we minimize the generic *ray-point* distance metric [17], i.e. the sum of distances between a grid point and the projection ray of a pixel that has seen that point. This can be applied at any stage of our approach; we apply it after the initial calibration using multiple grids (cf. §2.1), for refining the pose of each additional grid (cf. §2.2), as well as at the end of the whole calibration [18].

3. Variants

3.1. Non-Central Cameras

In the non-central case, collinearity constraints require 3 or more grid points per pixel, instead of 2 for central cameras (where the optical center, though unknown, is taken into account). We use the same notations as in §2.1. For a non-central camera, we apply the collinearity constraint on the region given by $\cup_{i=3}^{n}(B_1 \cap B_2 \cap B_i)$. Once the poses are computed the calibration is eventually extended to a larger region given by $\cup_{i,j=1\cdots n; i\neq j}(B_i \cap B_j)$.

We now summarize the calibration procedure, analogously to §2.1. We have no optical center here, so do consider the following $4 \times n$ matrix of collinear points:

$$\begin{pmatrix} Q^1_1 & R^2_{11}Q^2_1 + R^2_{12}Q^2_2 + t^2_1 Q^2_4 & \cdots \\ Q^1_2 & R^2_{21}Q^2_1 + R^2_{22}Q^2_2 + t^2_2 Q^2_4 & \cdots \\ 0 & R^2_{31}Q^2_1 + R^2_{32}Q^2_2 + t^2_3 Q^2_4 & \cdots \\ Q^1_4 & Q^2_4 & \cdots \end{pmatrix}$$

Similarly to the central case we can apply the collinearity constraint by equating the determinant of every 3×3 submatrix to zero. Using simulations we found, as in the central case, that not all of these provide unique solutions. In contrast to the central case, where we used the center and the first grid to build a system linking all the pose variables, we here use the first and second grid to build the system. Thus we have $3 \times (n-2)$ possible tensors, represented by $T_{3jk;12y}, (j,k \in \{1,2,4\},\ y = 3\cdots n)$. As in the central

case, we are able to use these tensors to estimate the poses of all the grids. More details are given in [18].

The next step of the calibration chain, pose estimation and computation of further projection rays, is also slightly different compared to central cameras (cf. §2.2). Here, the calibration region is extended to $C_{k+1} = \cup_{i,j=1\cdots n;i\neq j}(B_i \cap B_j)$, i.e. it contains all pixels that are matched to at least 2 grid points. As for the actual pose estimation, it can be formulated in the same way as for central cameras, but may lead to a set of 8 solutions that does not contain reflected pairs [4, 13, 17]. Disambiguation can be carried out using additional points besides the 3 used for the minimal pose routine.

3.2. Slightly Non-Central Cameras

For slightly non-central cameras like fisheye, spherical or hyperbolic catadioptric cameras, we start by running the central version of the generic calibration to obtain an initial partial calibration. Typically we use four or five images simultaneously to calibrate an image region and then use pose estimation to add other images and cover the rest of the image region. Next, we relax the central assumption; projection rays are first computed from grid points, without enforcing them to pass through an optical center. After this, a non-central bundle adjustment is performed [18].

3.3. Selecting the Best Camera Model

The non-central calibration algorithm of §3.1, can not be used as such to calibrate a central camera: data (pixel-to-grid correspondences) coming from a central camera, will lead to a higher rank-deficiency in the linear solution of the tensors, causing an incorrect calibration (although residuals will be lower). However, we may, by analyzing the rank of the underlying equation system, detect this problem and maybe even classify the camera as being central and then apply the appropriate calibration algorithm. More generally speaking, this is a model selection problem, and the rank-analysis or any other solution will allow to build a truly complete black box calibration system.

To this end, we have to take into account a few intermediate camera models that may be encountered in practice. One such case is the class of cameras for which there exists a single line that cuts all projection rays (we call them *axial cameras*). Examples are the classical two-camera stereo systems (the mentioned line is the baseline joining the two optical centers) and certain non-central catadioptric cameras, e.g. all catadioptric cameras with a spherical mirror. A yet more special class of cameras are so-called crossed-slits cameras [5], which encompass pushbroom cameras [8]. We are currently specializing our calibration approach to these additional general classes of camera types. Overall, it seems that these 4 classes (central, axial, crossed-slits, fully non-central) and their associated calibration algorithms, maybe with a few additional classes, should be sufficient to calibrate most cameras.

Besides considering these general camera types, we may also discuss the choice between parametric and non-parametric models for a given camera. Generic calibration not only allows to calibrate any camera system by treating it as a black box, it also provides the ability to easily obtain a parametric calibration once the model for the camera is known. Every parametric calibration will just be a model-fitting problem, which can be solved as a non-linear optimization problem starting with the good initial solution obtained using generic calibration.

4. Practical Issues

First, we found that grids with circular targets, using point spread function to compute the centers, provide stable calibration compared to checkerboard patterns.

Secondly, the usage of grids with very different orientations and positions is important for stable calibration. One way to easily achieve this is to use calibration grids of different sizes and to put them at different distances from the camera (together with sufficient orientation differences).

Thirdly, by using a combination of *local* 4-point homography based prediction, local collinearity and orthogonality constraints, we start from four features (circular targets or corners), located at the corners of a square, and incrementally extend the matching of image features to grid coordinates along all directions. This approach worked automatically for all pinhole images as well as for several fisheye and catadioptric images. However we also had to use manual input for some images.

The last issue is concerned with a required interpolation process: for every grid point in the first image we compute the interpolated points in the other grids' coordinate systems (since for other grids, the extracted targets or corners do not lie on the same pixels in general). To take care of the noise we impose collinearity constraints (globally for central cameras and locally for non-central cameras) during interpolation process for the originally collinear corners in the calibration grids [18]. This improved the numerical stability of the results significantly.

5. Experiments and Results

We have calibrated a wide variety of cameras (both central and non-central) as shown in Table 2. Results are first discussed for several "slightly non-central" cameras, and then for a multi-camera system.

Slightly non-central cameras: central vs. non-central models. For three cameras (a fisheye, a hyperbolic and a spherical catadioptric system, see sample images in Figure 3), we applied both, central calibration and the procedure explained in § 3.2, going from central to non-central.

Table 2. Bundle adjustment statistics for different cameras. (C) and (NC) refer to central and non-central calibration respectively, and RMS is the root-mean-square residual error of the bundle adjustment (ray-point distances). It is given in percent, relative to the overall size of the scene (largest pairwise distance between points on calibration grids).

Camera	Images	Rays	Points	RMS
Pinhole (C)	3	217	651	0.04
Fisheye (C)	23	508	2314	0.12
(NC)	23	342	1712	0.10
Sphere (C)	24	380	1441	2.94
(NC)	24	447	1726	0.37
Hyperbolic (C)	24	293	1020	0.40
(NC)	24	190	821	0.34
Multi-Cam (NC)	3	1156	3468	0.69
Eye+Pinhole (C)	3	29	57	0.98

Table 3. RMS error for circle fits to grid points, for turntable sequences (see text).

Camera	Grids	Central	Non-Central
Fisheye	14	0.64	0.49
Spherical	19	2.40	1.60
Hyperbolic	12	0.81	1.17

Table 2 shows that the bundle adjustment's residual errors for central and non-central calibration, are very close to one another for the fisheye and hyperbolic catadioptric cameras. This suggests that for the cameras used in the experiments, the central model is appropriate. As for the spherical catadioptric camera, the non-central model has a significantly lower residual, which may suggest that a non-central model is better here.

To further investigate this issue we performed another evaluation. A calibration grid was put on a turntable, and images were acquired for different turntable positions. We are thus able to quantitatively evaluate the calibration, by measuring how close the recovered grid pose corresponds to a turntable sequence. Individual grid points move on a circle in 3D; we thus compute a least squares circle fit to the 3D positions given by the estimated grid pose. At the bottom of Figure 3, recovered grid poses are shown, as well as a circle fit to the positions of one grid point. Table 3 shows the RMS errors of circle fits (again, relative to scene size, and given in percent). We note that the non-central model provides a significantly better reconstruction than the central one for the spherical catadioptric camera, which thus confirms the above observation. For the fisheye, the non-central calibration also performs better, but not as significantly. As for the hyperbolic catadioptric camera, the central model gives a better reconstruction though. This can probably be explained as follows. Inspite potential imprecisions in the camera setup, the camera seems to be sufficiently close to a central one, so that the non-central model leads to overfitting. Consequently, although the bundle adjustment's residual is lower than for the central model (which always has to be the case), it gives "predictions" (here, pose or motion estimation) which are unreliable.

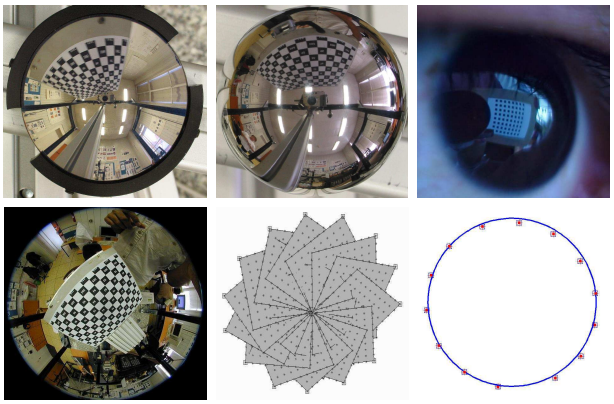


Figure 3. Top: sample images for hyperbolic (left), spherical (middle) and eye based catadioptric cameras (right). Bottom: fisyeye image (left), pose of calibration grids used to calibrate the fisheye (middle) and a least squares circle fit to the estimated positions of one grid point (right).

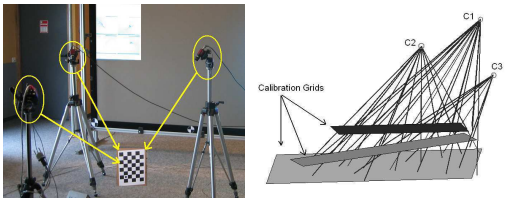


Figure 4. Multi-camera setup consisting of 3 cameras (left). Recovered projection rays and grid poses (right).

Calibration of a multi-camera system. A multi-camera network can be considered as a single generic imaging system. As shown in Figure 4 (left), we used a system of three (approximately pinhole) cameras to capture three images each of a calibration grid. We virtually concatenated the images from the individual cameras and computed all projection rays and the three grid poses in a single reference frame (see Figure 4 (right)), using the non-central algorithm described in § 3.1.

In order to evaluate the calibration, we compared results with those obtained by plane-based calibration [21, 27], that used the knowledge that the three cameras are pinholes. In both, our multi-camera calibration, and plane-based calibration, the first grid was used to fix the global coordinate system. We can thus compare the estimated poses of the other two grids for the two methods. This is done for both, the rotational and translational parts of the pose. As for rotation, we measure the angle (in radians) of the relative rotation between the rotation matrices given by the two methods, see columns R_i in Table 4). As for translation, we measure the

Table 4. Evaluation of non-central multi-camera calibration relative to plane-based calibration. See text for more details.

Camera	R_2	R_3	t_2	t_3	Center
1	0.0117	0.0359	0.56	3.04	2.78
2	0.0149	0.0085	0.44	2.80	2.17
3	0.0088	0.0249	0.53	2.59	1.16

distance between the estimated 3D positions of the grids' centers of gravity (columns t_i in Table 4) expressed in percent, relative to the scene size. Here, plane-based calibration is done separately for each camera, leading to the three rows of Table 4.

From the non-central multi-camera calibration, we also estimate the positions of the three optical centers, by clustering the projection rays and computing least squares point fits to them. The column "Center" of Table 4 shows the distances between optical centers (expressed in percent and relative to the scene size) computed using this approach and plane-based calibration. The discrepancies are low, suggesting that the non-central calibration of a multi-camera setup is indeed feasible.

Another experiment we carried out was to calibrate a small region of the exotic catadioptric system formed with an eye as mirror, cf. an image in Figure 3 and bundle adjustment statistics in Table 2.

6. Summary and Conclusions

We have proposed a non-parametric, generic calibration approach and shown its feasibility by calibrating a wide variety of cameras. One of the important issues is in the identification of appropriate models, central or non-central, for slightly non-central cameras. For understanding complex cameras or mirror surfaces, where mathematical modeling might be more demanding, generic calibration can be used as a black box tool to first obtain the projection rays. The nature of these projection rays can be experimented further to identify the right parametric model.

Acknowledgments. We want to thank Tomaš Pajdla, Branislav Mičušik, Bertrand Holveck and Thomas Bonfort for their help in the experiments. This work was partially supported by the Multi University Research Initiative (MURI) grant by Army Research Office under contract DAA19-00-1-0352 and the NSF grant ACI-0222900.

References

[1] D. Aliaga. Accurate Catadioptric Calibration for Real-size Pose Estimation of Room-size Environments. *ICCV*, 2001.

[2] S. Baker and S. Nayar. A Theory of Catadioptric Image Formation. *ICCV*, 1998.

[3] G. Champleboux, S. Lavallée, P. Sautot and P. Cinquin. Accurate Calibration of Cameras and Range Imaging Sensors: the NPBS Method. *ICRA*, 1992.

[4] C.-S. Chen and W.-Y. Chang. On Pose Recovery for Generalized Visual Sensors. *PAMI*, 2004.

[5] D. Feldman, T. Pajdla and D. Weinshall. On the Epipolar Geometry of the Crossed-Slits Projection. *ICCV*, 2003.

[6] K.D. Gremban, C.E. Thorpe, and T. Kanade. Geometric Camera Calibration using Systems of Linear Equations. *ICRA*, 1988.

[7] M.D. Grossberg and S.K. Nayar. A General Imaging Model and a Method for Finding its Parameters. *ICCV*, 2001.

[8] R. Gupta and R.I. Hartley. Linear Pushbroom Cameras. *PAMI*, 1997.

[9] R.M. Haralick, C.N. Lee, K. Ottenberg, and M. Nolle. Review and Analysis of Solutions of the Three Point Perspective Pose Estimation Problem. *IJCV*, 1994.

[10] R.A. Hicks and R. Bajcsy. Catadioptric Sensors that Approximate Wide-angle Perspective Projections. *CVPR*, 2000.

[11] B. Mičušik and T. Pajdla. Autocalibration and 3D Reconstruction with Non-central Catadioptric Cameras. *CVPR*, 2004.

[12] J. Neumann, C. Fermüller, and Y. Aloimonos. Polydioptric Camera Design and 3D Motion Estimation. *CVPR*, 2003.

[13] D. Nistér. A Minimal Solution to the Generalized 3-Point Pose Problem. *CVPR*, 2004.

[14] T. Pajdla. Stereo with Oblique Cameras. *IJCV*, 2002.

[15] S. Peleg, M. Ben-Ezra, and Y. Pritch. Omnistereo: Panoramic Stereo Imaging. *PAMI*, 2001.

[16] R. Pless. Using Many Cameras as One. *CVPR*, 2003.

[17] S. Ramalingam, S.K. Lodha, and P. Sturm. A Generic Cross-Camera Structure-from-Motion Analysis. *Omnivis*, 2004.

[18] S. Ramalingam, P. Sturm, and S.K. Lodha. Theory and Experiments towards Complete Generic Calibration. *Research Report*, INRIA, April, 2005.

[19] S. Seitz and J. Kim. The Space of All Stereo Images. *IJCV*, 2002.

[20] H.Y. Shum and L.W. He. Rendering with Concentric Mosaics. *SIGGRAPH*, 1999.

[21] P. Sturm and S. Maybank. On Plane-Based Camera Calibration: A General Algorithm, Singularities, Applications. *CVPR*, 1999.

[22] P. Sturm and S. Ramalingam. A Generic Concept for Camera Calibration. *ECCV*, 2004.

[23] R. Swaminathan, M.D. Grossberg, and S.K. Nayar. A Perspective on Distortions. *CVPR*, 2003.

[24] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle Adjustment: A Modern Synthesis. *Workshop on Vision Algorithms: Theory and Practice*, 2000.

[25] Y. Wexler, A.W. Fitzgibbon, and A. Zisserman. Learning Epipolar Geometry from Image Sequences. *CVPR*, 2003.

[26] D.N. Wood, A. Finkelstein, J.F. Hughes, C.E. Thayer, and D.H. Salesin. Multiperspective Panoramas for Cel Animation. *SIGGRAPH*, 1997.

[27] Z. Zhang. A Flexible New Technique for Camera Calibration. *PAMI*, 2000.

Theory and Calibration for Axial Cameras

Srikumar Ramalingam[1&2], Peter Sturm[1], and Suresh K. Lodha[2]

[1] INRIA Rhône-Alpes, Montbonnot St Martin, France
{Srikumar.Ramalingam, Peter.Sturm}@inrialpes.fr
[2] University of California, Santa Cruz, USA
{lodha}@soe.ucsc.edu

Abstract. Although most works in computer vision use perspective or other central cameras, the interest in non-central camera models has increased lately, especially with respect to omnidirectional vision. Calibration and structure-from-motion algorithms exist for both, central and non-central cameras. An intermediate class of cameras, although encountered rather frequently, has received less attention. So-called *axial cameras* are non-central but their projection rays are constrained by the existence of a line that cuts all of them. This is the case for stereo systems, many non-central catadioptric cameras and pushbroom cameras for example. In this paper, we study the geometry of axial cameras and propose a calibration approach for them. We also describe the various axial catadioptric configurations which are more common and less restrictive than central catadioptric ones. Finally we used simulations and real experiments to prove the validity of our theory.

1 Introduction

Many camera models have been considered in computer vision and related fields and even more taylor-made calibration methods have been developed. Most of those are designed for central cameras, but approaches and studies for non-central or general ones also exist [5–9, 16, 12, 13, 3]. An intermediate class of cameras, lying between central and fully non-central ones, is that of so-called *axial cameras*: their projection rays are constrained by the existence of a line that cuts all of them, the **camera axis**, but they may not go through a single optical center.

The axial model is a rather useful one (cf. figure 1(a) and (b)). Many misaligned catadioptric configurations fall under this model. Such configurations, which are slightly non-central, are usually classified as a non-central camera and calibrated using an iterative nonlinear algorithm [10, 2, 14]. For example, whenever the mirror is a surface of revolution and the central camera looking at the mirror lies anywhere on the revolution axis, the system is of axial type. Furthermore, two-camera stereo systems or systems consisting of three or more aligned cameras, are axial. Pushbroom cameras [15] are another example, although they are of a more restricted class (there exist two camera axes [4]).

In this paper, we propose a generic calibration approach for axial cameras, the first to our knowledge. It uses images of planar calibration grids, put in unknown positions. We show the existence of multi-view tensors that can be estimated linearly and from which the pose of the calibration grids as well as the position of the camera axis, can be recovered. The actual calibration is then performed by computing projection rays for all individual pixels of a camera, constrained to cut the camera axis.

The paper is organized as follows. The problem is formalized in section 2. In section 3, we show what can be done with two images of calibration grids. Complete calibration using three images, is described in section 4, followed by a bundle adjustment algorithm in section 5. Various types of axial catadioptric cameras are listed in section 6. Experimental results and conclusions are given in sections 7 and 8.

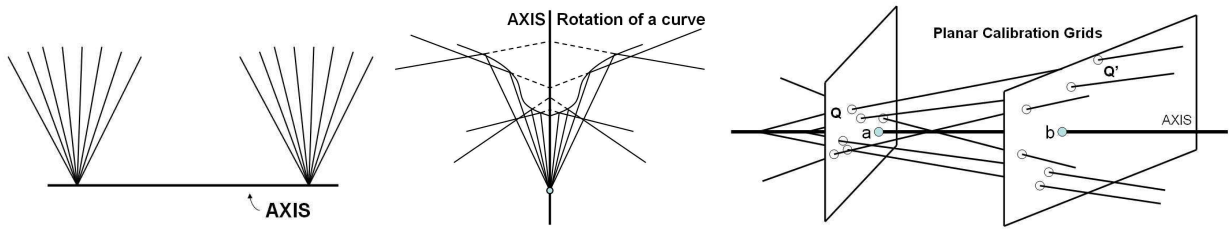
2 Problem Formulation

In the following, we will call **camera axis** the line cutting all projection rays. It will be represented by a 6-vector $\mathbf{L}$ and the associated 4×4 skew-symmetric Plücker matrix $[\mathbf{L}]_\times$:

$$[\mathbf{L}]_\times = \begin{pmatrix} 0 & -L_4 & L_6 & -L_2 \\ L_4 & 0 & -L_5 & -L_3 \\ -L_6 & L_5 & 0 & -L_1 \\ L_2 & L_3 & L_1 & 0 \end{pmatrix}$$

The product $[\mathbf{L}]_\times \mathbf{Q}$ gives the plane spanned by the line $\mathbf{L}$ and the point $\mathbf{Q}$. Consider further the two 3-vectors:

$$\mathbf{A} = \begin{pmatrix} L_5 \\ L_6 \\ L_4 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} L_2 \\ L_3 \\ L_1 \end{pmatrix}$$



(a) (b) (c)

Fig. 1. Examples of axial imaging models (a) stereo camera (b) a mirror formed by rotating a planar curve about an axis containing the optical center of the perspective camera.(c) Calibration of axial cameras using calibration grids: The projection rays, camera axis and two grids are shown. The axis intersects at **a** and **b** on the first and the second calibration grids respectively.

for which the Plücker constraint holds: $\mathbf{B}^\mathsf{T}\mathbf{A} = 0$. $\mathbf{A}$ represents the point at infinity of the line. The Plücker matrix can be written as:

$$[\mathbf{L}]_\times = \begin{pmatrix} 0 & -L_4 & L_6 & -L_2 \\ L_4 & 0 & -L_5 & -L_3 \\ -L_6 & L_5 & 0 & -L_1 \\ L_2 & L_3 & L_1 & 0 \end{pmatrix} = \begin{pmatrix} [\mathbf{A}]_\times & -\mathbf{B} \\ \mathbf{B}^\mathsf{T} & 0 \end{pmatrix}$$

The calibration problem considered in this paper is to compute projection rays for all pixels of a camera, from images of planar calibration grids in unknown positions. We assume that dense point correspondences are given, i.e. for (many) pixels, we are able to determine the points on the calibration grids that are seen in that pixel. Computed projection rays will be constrained to cut the camera axis. The coordinate system in which calibration will be expressed, is that of the first calibration grid. Calibration thus consists in computing the position of the camera axis and of the projection rays, in that coordinate system. The proposed approach proceeds by first estimating the camera axis and the pose of all grids but the first one.

3 What can be Done with Two Views of Calibration Grids?

Consider some pixel and let $\mathbf{Q}$ and $\mathbf{Q}'$ be the corresponding points on the two calibration grids, given as 3D points in the grids' local coordinate systems. Since we consider planar grids, we impose $Q_3 = Q'_3 = 0$.

We have the following constraint on the pose of the second grid $(\mathsf{R}', \mathbf{t}')$ as well as the unknown camera axis $\mathbf{L}$: the line spanned by $\mathbf{Q}$ and $\mathbf{Q}'$ cuts $\mathbf{L}$, hence is coplanar with it. Hence, for the correct pose and camera axis, we must have:

$$\mathbf{Q}^\mathsf{T}[\mathbf{L}]_\times \begin{pmatrix} \mathsf{R}' & \mathbf{t}' \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \mathbf{Q}' = 0$$

Hence:

$$\begin{pmatrix} Q_1 \\ Q_2 \\ Q_4 \end{pmatrix}^\mathsf{T} \begin{pmatrix} 0 & -L_4 & L_6 & -L_2 \\ L_4 & 0 & -L_5 & -L_3 \\ L_2 & L_3 & L_1 & 0 \end{pmatrix} \begin{pmatrix} \bar{\mathsf{R}}' & \mathbf{t}' \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \begin{pmatrix} Q'_1 \\ Q'_2 \\ Q'_4 \end{pmatrix} = 0$$

where $\bar{\mathsf{R}}'$ refers to the 3×2 submatrix of R' containing only the first and the second rows. We thus have the following 3×3 tensor that can be estimated linearly from point correspondences:

$$\mathsf{F} \sim \begin{pmatrix} 0 & -L_4 & L_6 & -L_2 \\ L_4 & 0 & -L_5 & -L_3 \\ L_2 & L_3 & L_1 & 0 \end{pmatrix} \begin{pmatrix} \bar{\mathsf{R}}' & \mathbf{t}' \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \tag{1}$$

It has only 7 degrees of freedom (9 - 1 for scale, -1 for rank-deficiency) so the 10 unknowns (4 for the camera axis, 3 for R' and 3 for $\mathbf{t}'$) can not be recovered from it.

We now look at what can actually be recovered from F. Let us first notice that its left null-vector is $(L_3, -L_2, L_4)^\mathsf{T}$ (it truly is the null-vector, as can be easily verified when taking into account the Plücker constraint). We thus can recover 2 of the 4 parameters of the camera axis. That null-vector contains actually the

coordinates of the camera axis' intersection with the first grid (in plane coordinates). Its 3D coordinates are given by $(L_3, -L_2, 0, L_4)^{\mathsf{T}}$. Similarly, the right null-vector of F gives the plane coordinates of the axis' intersection with the second grid. Besides this F also gives constraints on R' and $\mathbf{t}'$. For example R' can be extracted up to 2 to 4 solutions. We will later observe that once we locally shift the intersection points, between the camera axis and calibration grids, to the origins of the respective grids the vector $\mathbf{t}'$ will lie on the camera axis. Inspite of all these additional constraints, arising from axial geometry, two views of calibration grids are not sufficient to uniquely extract R' and $\mathbf{t}'$. Thus we use three calibration grids as described below.

4 Full Calibration using Three Views of Calibration Grids

Let $\mathbf{Q}$, $\mathbf{Q}'$, $\mathbf{Q}''$ refer to the grid points corresponding to a single pixel in the three grids. The poses of the grids are $(\mathsf{I}, \mathbf{0})$, $(\mathsf{R}', \mathbf{t}')$ and $(\mathsf{R}'', \mathbf{t}'')$ respectively. Since the three points $\mathbf{Q}$, $\mathbf{Q}'$ and $\mathbf{Q}''$ are collinear we use this constraint to extract the poses of the calibration grids [12]. Every 3×3 submatrix of the following 4×3 matrix has zero subdeterminant.

$$\begin{pmatrix} \mathbf{Q} & \begin{pmatrix} \mathsf{R}' & \mathbf{t}' \\ \mathbf{0}^{\mathsf{T}} & 1 \end{pmatrix} \mathbf{Q}' & \begin{pmatrix} \mathsf{R}'' & \mathbf{t}'' \\ \mathbf{0}^{\mathsf{T}} & 1 \end{pmatrix} \mathbf{Q}'' \end{pmatrix}$$

The submatrices constructed by removing the first and the second rows lead to the constraints $\sum C_i T1_i = 0$ and $\sum C_i T2_i = 0$ respectively (as described in Table 1). These are nothing but homogeneous linear systems of the form $AX = 0$. The unknown vector X is formed from the 14 variables (C_i). Each of these variables are coupled coefficients of the poses of the grids. The matrix A is constructed by stacking the trilinear tensors $T1$ and $T2$, which can be computed from the coordinates of $\mathbf{Q}$, $\mathbf{Q}'$ and $\mathbf{Q}''$. In future when we refer to the rank of a linear system $AX = 0$, we refer to the rank of the matrix A. The rank has to be one less than the number of variables to estimate them uniquely upto a scale. For example, each of the above linear systems must have a rank of 13 to estimate the coefficients (C_i) uniquely. These systems were used to calibrate completely non-central cameras [10]. However in the case of axial cameras, these systems were found to have a rank of 12. This implies that the solution can not be obtained uniquely. In order to resolve this ambiguity we will need more constraints.

i	Motion (C_i)	$T1_i$	$T2_i$	i	Motion (C_i)	$T1_i$	$T2_i$
1	R'_{31}	$Q_2Q'_1Q''_4$	$Q_1Q'_1Q''_4$	13	$R'_{22}R''_{32} - R'_{32}R''_{22}$	$Q_4Q'_2Q''_2$	0
2	R'_{32}	$Q_2Q'_2Q''_4$	$Q_1Q'_2Q''_4$	14	$R'_{11}t''_3 - R'_{31}t''_1$	0	$Q_4Q'_1Q''_4$
3	R''_{31}	$-Q_2Q'_4Q''_1$	$-Q_1Q'_4Q''_1$	15	$R'_{12}t''_3 - R'_{32}t''_1$	0	$Q_4Q'_2Q''_4$
4	R''_{32}	$-Q_2Q'_4Q''_2$	$-Q_1Q'_4Q''_2$	16	$R'_{21}t''_3 - R'_{31}t''_2$	$Q_4Q'_1Q''_4$	0
5	$t'_3 - t''_3$	$Q_2Q'_4Q''_4$	$Q_1Q'_4Q''_4$	17	$R'_{22}t''_3 - R'_{32}t''_2$	$Q_4Q'_2Q''_4$	0
6	$R'_{11}R''_{31} - R'_{31}R''_{11}$	0	$Q_4Q'_1Q''_1$	18	$R''_{11}t'_3 - R''_{31}t'_1$	0	$-Q_4Q'_4Q''_1$
7	$R'_{11}R''_{32} - R'_{31}R''_{12}$	0	$Q_4Q'_1Q''_2$	19	$R''_{12}t'_3 - R''_{32}t'_1$	0	$-Q_4Q'_4Q''_2$
8	$R'_{12}R''_{31} - R'_{32}R''_{11}$	0	$Q_4Q'_2Q''_1$	20	$R''_{21}t'_3 - R''_{31}t'_2$	$-Q_4Q'_4Q''_1$	0
9	$R'_{12}R''_{32} - R'_{32}R''_{12}$	0	$Q_4Q'_2Q''_2$	21	$R''_{22}t'_3 - R''_{32}t'_2$	$-Q_4Q'_4Q''_2$	0
10	$R'_{21}R''_{31} - R'_{31}R''_{21}$	$Q_4Q'_1Q''_1$	0	22	$t'_1t''_3 - t'_3t''_1$	0	$Q_4Q'_4Q''_4$
11	$R'_{21}R''_{32} - R'_{31}R''_{22}$	$Q_4Q'_1Q''_2$	0	23	$t'_2t''_3 - t'_3t''_2$	$Q_4Q'_4Q''_4$	0
12	$R'_{22}R''_{31} - R'_{32}R''_{21}$	$Q_4Q'_2Q''_1$	0				

Table 1. Trifocal tensor in the generic calibration of completely non-central cameras.

4.1 Intersection of axis and calibration grids

Using the technique described earlier we compute the intersection of the camera axis with the three grids at $\mathbf{a}$,$\mathbf{b}$ and $\mathbf{c}$ respectively. We translate the local grid coordinates such that these intersection points become their respective origins. Without loss of generality we continue to use the same notations after the transformations.

$$\mathbf{Q} \longleftarrow \mathbf{Q} - \mathbf{a},\ \mathbf{Q}' \longleftarrow \mathbf{Q}' - \mathbf{b},\ \mathbf{Q}'' \longleftarrow \mathbf{Q}'' - \mathbf{c},$$

We can obtain a collinearity constraint by putting these origins in the same coordinate system. Every 3×3 subdeterminant of the following 4×3 matrix vanishes.

$$\begin{pmatrix} \begin{pmatrix} \mathbf{0} \\ 1 \end{pmatrix} & \begin{pmatrix} \mathsf{R}' & \mathbf{t}' \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \begin{pmatrix} \mathbf{0} \\ 1 \end{pmatrix} & \begin{pmatrix} \mathsf{R}'' & \mathbf{t}'' \\ \mathbf{0}^\mathsf{T} & 0 \end{pmatrix} \begin{pmatrix} \mathbf{0} \\ 1 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} 0 & t'_1 & t''_1 \\ 0 & t'_2 & t''_2 \\ 0 & t'_3 & t''_3 \\ 1 & 1 & 1 \end{pmatrix}$$

The camera axis passes through O, $\mathbf{t}'$ and $\mathbf{t}''$. This enables us to express $\mathbf{t}''$ as a multiple of $\mathbf{t}'$ using some scalar Δ: $\mathbf{t}'' = \Delta\mathbf{t}'$. As a result, the variables C_{22} and C_{23} from Table 1 disappear.

$$C_{22} = t'_1 t''_3 - t'_3 t''_1 = t'_1 \Delta t'_3 - t'_3 \Delta t'_1 = 0$$

$$C_{23} = t'_2 t''_3 - t'_3 t''_2 = t'_2 \Delta t'_3 - t'_3 \Delta t'_2 = 0$$

On disappearing, C_{22} and C_{23} reduce the size of the linear systems $\sum C_i T1_i = 0$ and $\sum C_i T2_i = 0$ each by one. Inspite of this reduction there still exists a rank deficiency of 2 in both these systems. The rank of each of these systems is 11 with 13 nonzero coefficients to be estimated. In the next section we provide the details of the usage of a coplanarity constraint, which exists in axial cameras, to remove the degeneracy problems.

4.2 Coplanarity constraints in axial cameras

The camera axis cuts all the projection rays. As observed earlier both $\mathbf{O}$ and $\mathbf{t}'$ lie on the camera axis. Along with these two points, we consider two grid points $\mathbf{Q}'$ and $\mathbf{Q}''$ lying on a single projection ray. Since these four points are coplanar, the determinant of the following 4×4 matrix disappears.

$$\begin{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} & \begin{pmatrix} t'_1 \\ t'_2 \\ t'_3 \\ 1 \end{pmatrix} & \begin{pmatrix} \mathsf{R}' & \mathbf{t}' \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \mathbf{Q}' & \begin{pmatrix} \mathsf{R}'' & \mathbf{\Delta} t' \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \mathbf{Q}'' \end{pmatrix}$$

The corresponding constraint is a linear system $\sum \alpha_{ij} Q'_i Q''_j = 0$ (see table 2). Note that Q'_4 and Q''_4 are not present because of the three zeros in the first column. We can solve this linear system to computer the solutions for α_{ij}. We expand the above linear system and do some algebraic manipulation.

$$\alpha_{11} Q'_1 Q''_1 + \alpha_{12} Q'_1 Q''_2 + \alpha_{21} Q'_2 Q''_1 + \alpha_{22} Q'_2 Q''_2 = 0$$

$$Q_4(\alpha_{11} Q'_1 Q''_1 + \alpha_{12} Q'_1 Q''_2 + \alpha_{21} Q'_2 Q''_1 + \alpha_{22} Q'_2 Q''_2) = 0$$

$$Q_4 Q'_2 Q''_2 = -\frac{\alpha_{11}}{\alpha_{22}} Q_4 Q'_1 Q''_1 - \frac{\alpha_{12}}{\alpha_{22}} Q_4 Q'_1 Q''_2 - \frac{\alpha_{21}}{\alpha_{22}} Q_4 Q'_2 Q''_1$$

This will enable us to represent both $T2_9$ and $T1_{13}$, from the earlier systems, in terms of other variables in the tensors $T1$ and $T2$ respectively.

$$T2_9 = -\frac{\alpha_{11}}{\alpha_{22}} T2_6 - \frac{\alpha_{12}}{\alpha_{22}} T2_7 - \frac{\alpha_{21}}{\alpha_{22}} T2_8$$

$$T1_{13} = -\frac{\alpha_{11}}{\alpha_{22}} T1_{10} - \frac{\alpha_{12}}{\alpha_{22}} T1_{11} - \frac{\alpha_{21}}{\alpha_{22}} T1_{12}$$

Using the above relation we obtain two new constraints given by $\sum A_i A1_i = 0$ and $\sum A_i A2_i = 0$. Note that each of these constraints are linear systems with 12 nonzero coefficients each. Both of them have a rank of 11 and thereby producing unique solutions for their coefficients (A_i). The individual elements in the poses of the grids are extracted from these coupled coefficients using orthonormality constraints of the rotation matrix [12].

i	j	α_{ij}
1	1	$t'_1(R'_{2,1}R''_{3,1} - R''_{2,1}R'_{3,1}) - t'_2(R'_{1,1}R''_{3,1} - R''_{1,1}R'_{3,1}) + t'_3(R'_{1,1}R''_{2,1} - R''_{1,1}R'_{2,1})$
1	2	$t'_1(R'_{2,1}R''_{3,2} - R''_{2,2}R'_{3,1}) - t'_2(R'_{1,1}R''_{3,2} - R''_{1,2}R'_{3,1}) + t'_3(R'_{1,1}R''_{2,2} - R''_{1,2}R'_{2,1})$
2	1	$t'_1(R'_{2,2}R''_{3,1} - R''_{2,1}R'_{3,2}) - t'_2(R'_{1,2}R''_{3,1} - R''_{1,1}R'_{3,2}) + t'_3(R'_{1,2}R''_{2,1} - R''_{1,1}R'_{2,2})$
2	2	$t'_1(R'_{2,2}R''_{3,2} - R''_{2,2}R'_{3,2}) - t'_2(R'_{1,2}R''_{3,2} - R''_{1,2}R'_{3,2}) + t'_3(R'_{1,2}R''_{2,2} - R''_{1,2}R'_{2,2})$

Table 2. Bifocal tensor from the coplanarity constraint on $\mathbf{O}$, $\mathbf{t}'$, $\mathbf{Q}'$ and $\mathbf{Q}''$.

i	Motion (A_i)	$A1_i$	$A2_i$	i	Motion (A_i)	$A1_i$	$A2_i$
1	R'_{31}	$Q_2Q'_1Q''_4$	$Q_1Q'_1Q''_4$	11	$C_{12} - \frac{\alpha_{21}}{\alpha_{22}}C_{13}$	$Q_4Q'_2Q''_1$	0
2	R'_{32}	$Q_2Q'_2Q''_4$	$Q_1Q'_2Q''_4$	12	$\Delta(R'_{11}t'_3 - R'_{31}t'_1)$	0	$Q_4Q'_1Q''_4$
3	R''_{31}	$-Q_2Q'_4Q''_1$	$-Q_1Q'_4Q''_1$	13	$\Delta(R'_{12}t'_3 - R'_{32}t'_1)$	0	$Q_4Q'_2Q''_4$
4	R''_{32}	$-Q_2Q'_4Q''_2$	$-Q_1Q'_4Q''_2$	14	$\Delta(R'_{21}t'_3 - R'_{31}t'_2)$	$Q_4Q'_1Q''_4$	0
5	$t'_3 - t''_3$	$Q_2Q'_4Q''_4$	$Q_1Q'_4Q''_4$	15	$\Delta(R'_{22}t'_3 - R'_{32}t'_2)$	$Q_4Q'_2Q''_4$	0
6	$C_6 - \frac{\alpha_{11}}{\alpha_{22}}C_9$	0	$Q_4Q'_1Q''_1$	16	$R''_{11}t'_3 - R''_{31}t'_1$	0	$-Q_4Q'_4Q''_1$
7	$C_7 - \frac{\alpha_{12}}{\alpha_{22}}C_9$	0	$Q_4Q'_1Q''_2$	17	$R''_{12}t'_3 - R''_{32}t'_1$	0	$-Q_4Q'_4Q''_2$
8	$C_8 - \frac{\alpha_{21}}{\alpha_{22}}C_9$	0	$Q_4Q'_2Q''_1$	18	$R''_{21}t'_3 - R''_{31}t'_2$	$-Q_4Q'_4Q''_1$	0
9	$C_{10} - \frac{\alpha_{11}}{\alpha_{22}}C_{13}$	$Q_4Q'_1Q''_1$	0	19	$R''_{22}t'_3 - R''_{32}t'_2$	$-Q_4Q'_4Q''_2$	0
10	$C_{11} - \frac{\alpha_{12}}{\alpha_{22}}C_{13}$	$Q_4Q'_1Q''_2$	0				

Table 3. Trifocal tensor for the generic calibration of axial cameras.

5 Bundle Adjustment Formulation

We give the details of a bundle adjustment which refines the estimated camera axis and poses of the calibration grids. This is similar to our earlier method [10], except that we have an additional constraint coming from the camera axis. The bundle adjustment is done by minimizing the distance between the grid points and the corresponding projection rays. The cost function is given below.

$$Cost = \sum_{i=1}^{n}\sum_{j=1}(\mathbf{A} + \lambda_i\mathbf{D} + \mu_{ji}\mathbf{D}_i - [\mathsf{R}_j\mathbf{T}_j]\mathbf{Q}_{ji})$$

- $(\mathbf{A}, \mathbf{D})$ - represents the axis (point, direction)
- $\mathbf{D}_i$ - unit direction vector of the i_{th} projection ray
- λ_i - parameter selecting the intersection of the i_{th} ray and the axis
- $\mathbf{Q}_{ji}$ - grid point on the j_{th} grid lying the i_{th} ray
- μ_{ji} - parameter selecting the point on the i_{th} ray closest to Q_j
- $(\mathsf{R}_j, \mathbf{T}_j)$ - pose of the calibration grid

6 Axial Catadioptric Configurations

Our formulation can classify a given camera into either axial or not. For example on applying our method on axial data we obtain unique solutions. On the other hand, a completely non-central camera will lead to an inconsistent (no solution), whereas a central camera will produce a rank deficient system (ambiguous solutions). Thus our technique produces unique solutions only for axial configurations. This can be used as a simple test in simulations to study the nature of complex catadioptric arrangements (as shown in Figure 2(a)). Since axial cameras are less restrictive than central cameras, they can be easily constructed using various combinations of mirrors and lenses. For example there are very few central configurations [1] (also see Table 4). Furthermore these configurations are difficult to build and maintain. For example, in a central catadioptric camera with hyperbolic mirror and perspective camera, the optical center has to be placed precisely on one of the mirror's focal points. On the other hand, the optical center can be anywhere on the mirror axis to have an axial geometry.

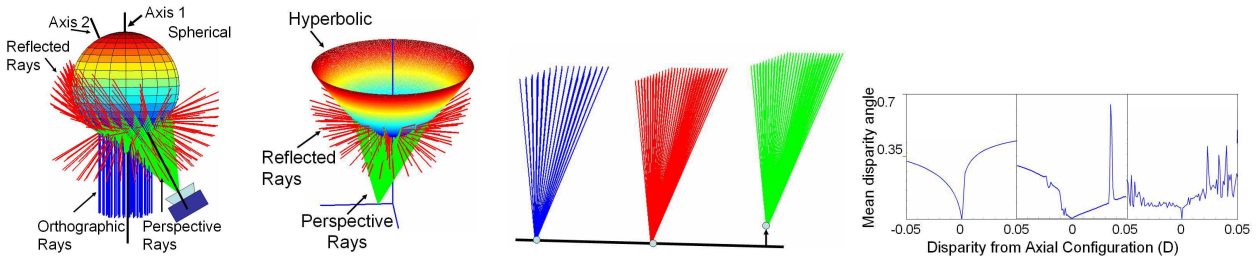
mirror	ctrl (pers)	axial (pers)	nctrl (pers)	ctral (ortho)	axial (ortho)	nctrl (ortho)
hyperbolic	o=f	o $\in$ MA	o $\notin$ MA	-	OA $\parallel$ MA	OA $\nparallel$ MA
spherical	-	always	-	-	always	-
parabolic	-	o $\in$ MA	o $\notin$ MA	OA $\parallel$ MA	-	OA $\nparallel$ MA
elliptic	o = f	o $\in$ MA	o $\notin$ MA	-	OA $\parallel$ MA	OA $\nparallel$ MA
cone	-	o $\in$ MA	o $\notin$ MA	-	OA $\parallel$ MA	OA $\nparallel$ MA
planar	always	-	planar	-	-	-
mir-rot	-	always	-	-	always	-

Table 4. Catadioptric configurations. Notations: ctrl (pers) - central configuration with perspective camera, nctrl (ortho) - non-central configuration with orthographic camera, mir-rot - mirror obtained by rotating a planar curve about the optical axis, o - optical center of the perspective camera, f - focus of the mirror, MA - major axis of mirror, OA - optical axis of the camera, = refers to same location, $\in$-lies on, $\parallel$-parallel, $\nparallel$-not parallel.

7 Experiments

7.1 Simulation

We started with perfect axial configurations for three scenarios (as shown in Figures 2(a), (b) and (c)) and gradually change the configurations to make them non-central. We quantify this change from the perfect axial configuration as disparity. For example, in Figure 2(a), the disparity represents the distance between the optical center of the perspective camera and the orthographic camera axis passing through the center of the sphere. This optical center is initially at a distance of 3 units from the center of the sphere (which is of radius 1 unit). In Figure 2(b), the disparity represents the distance between the optical center of the perspective camera and the major axis of the hyperboloid. Initially the optical center is at a distance of 5 units from the tip of the hyperboloid, whose two radii are 5 and 10 units. In Figure 2(c), the disparity represents the distance between the optical center of the third camera and the line joining the first two cameras. The distance between two consecutive centers of the cameras is 40 units. We calibrate these systems in the presence of disparities. We compute the mean angular error between the original and the reconstructed projection rays in Figure 2(d). Note that the the mean angular error (given in radians) reaches zero only at the precise axial configuration.



(a) (b) (c) (d)

Fig. 2. Test for axial configuration. (a) Catadioptric (spherical mirror+pers.camera+ortho.camera): becomes non-central when the two optical centers and the sphere center are not collinear (as shown).(b) Catadioptric (Hyperbolic mirror+pers.camera): becomes non-central if the optical center is not on the axis of the hyperbolic mirror (as shown). (c) Tristereo when one of the cameras is axially misplaced (as shown). (d) shows the mean angular error between the original and reconstructed projection rays w.r.t disparity. The graphs shown in left, middle and right correspond to scenarios in (a), (b) and (c) respectively (see text for more details).

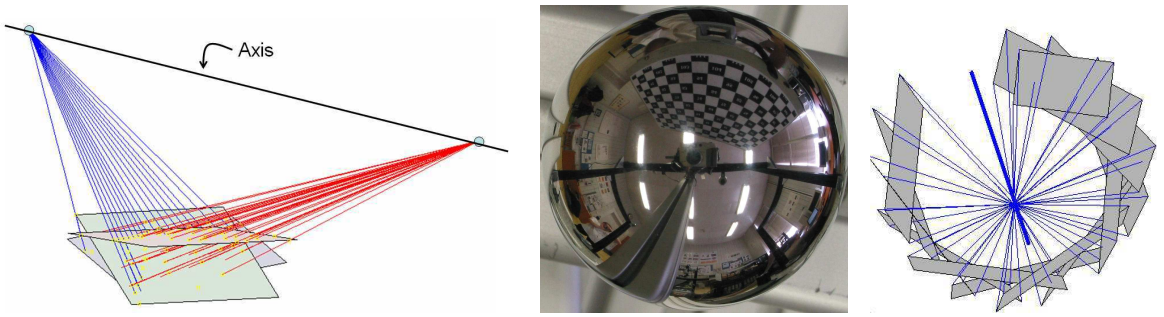
7.2 Stereo camera

We captured three images of a calibration grid using two different cameras. The goal is to reconstruct the projection rays of both the cameras in the same generic framework using our axial calibration algorithm. Here

the camera axis is the line joining the two optical centers (see Figure 3(a)). The image of the combined system is formed by concatenating the images from the two cameras. Figure 2(d) shows that our algorithm is very sensitive to noise. However using RANSAC, it is possible to obtain a good calibration. Once we compute the pose of the grids we can compute the rays corresponding to individual cameras in the stereo system. These rays can also be made to intersect separately and parameterized using a pinhole model. The RMS bundle adjustment error, based on the distance between the projection rays and grid points on the calibration grids, is of the order of 0.29% w.r.t overall size of the scene. The estimated camera parameters are close to the correct results. The reconstructed projection rays and grids are shown in Figure 3(a).

7.3 Spherical catadioptric cameras

We calibrated a real spherical catadioptric camera and extracted the camera axis. We start with an initial calibration using three grids using the above axial algorithm. This enables us to obtain an initial estimate for the axis and the projection rays. Using this partial calibration, we use pose estimation to incrementally compute the pose of newer grids. We followed our earlier method to obtain complete calibration [10]. The calibration grid captured by a spherical catadioptric camera is shown in Figure 3(b). We estimated the pose of several grids on a turntable sequence using the calibration. The grid positions and the axis are shown in Figure 3(c). For more details about results and other experimental issues please refer to [11].



(a) (b) (c)

Fig. 3. Axial calibration: (a) Calibration of a stereo system (b) Image captured by a catadioptric system with a spherical mirror and a perspective camera. (b) Estimated poses of several grids along with the camera axis.

8 Conclusions

We studied the theory and proposed a linear calibration algorithm for an intermediate class of cameras called axial cameras. Further line of investigation needs to be carried out to test the accuracy of this approach with respect to parametric and completely non-central approaches.

Acknowledgments: We thank Tomáš Pajdla, Branislav Mičušík and Diana Mateus for the data.

References

1. S. Baker and S. Nayar. A theory of catadioptric image formation. *ICCV*, 1998.
2. D. Aliaga. Accurate Catadioptric Calibration for Real-size Pose Estimation of Room-size Environments, *ICCV*, 2001.
3. H. Bakstein and T. Pajdla. An overview of non-central cameras. *Computer Vision Winter Workshop*, Ljubljana, Slovenia, 2001.
4. Doron Feldman, Tomas Pajdla and Daphna Weinshall. On the Epipolar Geometry of the Crossed-Slits Projection. *ICCV*, 2003.
5. M.D. Grossberg and S.K. Nayar. A general imaging model and a method for finding its parameters. *ICCV*, 2001.
6. J. Neumann, C. Fermüller, and Y. Aloimonos. Polydioptric Camera Design and 3D Motion Estimation. *CVPR*, 2003.
7. T. Pajdla. Stereo with oblique cameras. *IJCV*, 2002.

8. S. Peleg, M. Ben-Ezra, and Y. Pritch. Omnistereo: Panoramic Stereo Imaging. *PAMI*, 2001.
9. R. Pless. Using Many Cameras as One. In *CVPR*, 2003.
10. S. Ramalingam, P. Sturm and S.K. Lodha. Towards Complete Generic Camera Calibration. *CVPR*, 2005.
11. S. Ramalingam, P. Sturm and S.K. Lodha. Generic calibration of axial cameras. *INRIA Research Report*, France, December 2005.
12. P. Sturm and S. Ramalingam. A generic concept for camera calibration. *ECCV*, 2004.
13. R. Swaminathan, M.D. Grossberg, and S.K. Nayar. A perspective on distortions. *CVPR*, 2003.
14. B. Micusik and T. Pajdla. Autocalibration and 3D Reconstruction with Non-central Catadioptric Cameras. *CVPR*, 2004.
15. R. Gupta and R.I. Hartley. Linear Pushbroom Cameras. *PAMI* 1997.
16. S. Seitz and J. Kim. The Space of All Stereo Images. *IJCV*, 2002.

Calibration of Cameras with Radially Symmetric Distortion

Jean-Philippe Tardif *
DIRO, Université de Montréal
Canada

Peter Sturm †
INRIA Rhône-Alpes
38330 Montbonnot St Martin, France

Abstract

We present a theory and algorithms for plane-based calibration and pose recovery of general radially distorted cameras. By this we understand cameras that have a distortion center and an optical axis such that the projection rays of pixels lying on a circle centered on the distortion center, form a right cone centered on the optical axis. The camera is said to have a singular viewpoint (SVP) if all such view cones have the same vertex (the optical center), otherwise we speak of non-SVP, and each view cone may have its own optical center on the optical axis. This model encompasses the classical radial distortion model, fisheyes, most central or non-central catadioptric cameras, but also cameras with radially symmetric caustics.
Calibration consists in the estimation of the distortion center, the opening angles of all view cones and their optical center. We present two approaches of computing a full calibration from dense correspondences of a single or multiple planes with known euclidean structure. The first one is based on a geometric constraint linking view cones and associated ellipses in the calibration plane; calibration of the view cones can be solved by determining the closest point to a set of hyperbolas. The second approach uses existing plane-based calibration methods to directly calibrate individual view cones. A simple distortion correction algorithm for calibrated SVP images is given. Preliminary experiments show convincing results.

1. Introduction

In the last few years, we have seen an increasing interest in non-conventional cameras and projection models, going beyond affine or perspective projection. There exists a large diversity of camera models; many of them specific to certain types of projections [1, 13] or families of cameras such as central catadioptric systems [2, 8, 3, 6]. All these models are described by a few intrinsic parameters, much like the classical pinhole model, possibly enhanced with radial or decentering distortion coefficients. Calibration methods exist for all these models, and they are usually tailor-made for them, i.e. can not be used for any other projection model. Several works address the calibration problem from an opposite point of view, by adopting a very generic imaging model that incorporates most commonly used cameras [9, 5, 17, 10, 15]. In the most general case, cameras are modeled by attributing an individual projection ray to each individual pixel. Such a model is highly expressive, but it is difficult to obtain a stable calibration of cameras with it.

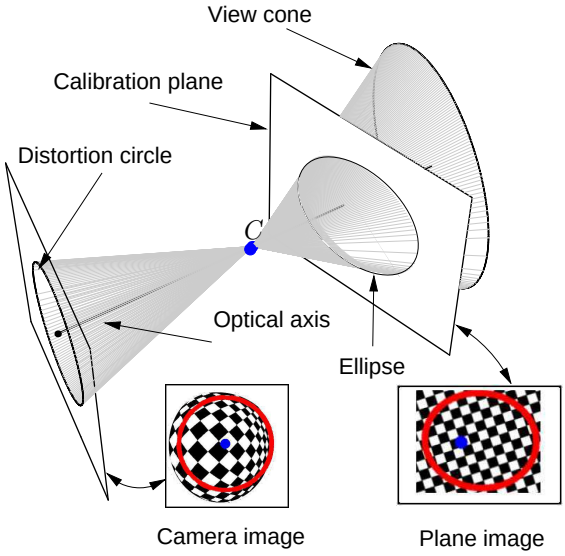


Figure 1: Our camera model (see text for explanations). The inlayed illustrations show the distortion center (in blue) and a distortion circle for a true image taken with a fisheye, and the corresponding calibration ellipse overlaid on a pattern shown on the calibration plane.

In this paper, we propose a simple camera model (and associated calibration methods) that hopefully offers a good compromise: it is sufficiently general to model many common camera types, but has much fewer parameters than the above generic model, thus making calibration much easier and more stable. We model cameras using the notions of a **distortion center** in the image and an **optical axis** in 3D. For cameras with **radially symmetric distortion**, the projection rays associated with pixels lying on a same circle centered on the distortion center, lie on a right **viewing cone** centered on the optical axis (cf. fig. 1). This encompasses many common camera models, such as pinhole (modulo aspect ratio and skew), the classical polynomial radial dis-

*tardifj@iro.umontreal.ca
†peter.sturm@inrialpes.fr

tortion model, fisheyes, or any catadioptric system whose mirror is a surface of revolution, and for which the optical axis of the perspective (or affine) camera looking at it is aligned with the mirror's revolution axis. Our model comprises **central** cameras (SVP), where all viewing cones have the same vertex (the **optical center**), but also **non-central** ones (NSVP), for which the viewing cones' vertices lie anywhere on the optical axis. In the latter case, we may speak of one optical center per viewing cone.

Problem statement. We want to calibrate cameras based on the above model, from one or several images of a calibration plane in unknown positions. The input to the calibration algorithms is a dense matching between the plane(s) and the camera image, and the euclidean structure of the plane(s). From this, we compute, for all viewing cones, their focal length (equivalent to the opening angle). Our algorithms assume a known position of the distortion center, but we also show how to estimate it, using repeated calibrations for different candidates. Calibration also comprises a pose estimation problem: estimating the orientation of the optical axis (relative to a calibration plane) and the location of each viewing cone's vertex on it.

Organization. A geometric study of our model is presented in §2, together with our first calibration approach. The second approach, based on the standard plane-based calibration method, is described in §3. In §4, we give an algorithm for performing perspective image rectification based on calibration results. Several practical issues and experimental results are presented in §5 and §6, respectively.

2. Geometry

2.1. One Distortion Circle

Let us consider one distortion circle in the image plane. We suppose that we have determined the ellipse on the calibration plane that is mapped to that circle via the camera's projection function (see §5). If we knew the position of the camera's optical center relative to the calibration plane, then we could compute the cone that has the optical center as vertex and that contains the above calibration ellipse. That cone has several interesting properties: its axis is the camera's optical axis and it is a right cone, i.e. rotationally symmetric with respect to its axis. From the cone, the focal length of the considered distortion circle can be easily computed (the cone's opening angle equals the field of view).

In practice, we do not know the optical center's position relative to the calibration plane. In the following, we show geometrical relations between the calibration ellipse, the optical center and the optical axis of the camera. When talking about optical center, we mean the optical center per distortion circle; they all lie on the optical axis and in the SVP case, they are identical.

Without loss of generality, we assume that the calibration plane is the plane $Z = 0$, and that the calibration ellipse is given by the matrix $\mathrm{diag}(a, b, -1)$, with $b \geq a > 0$, i.e. the X-axis is the ellipse's major axis. Our aim is to provide constraints on the position of the optical center, as well as on the orientation of the optical axis, from this ellipse.

Let us first state a well-known result. Consider a right cone (whose vertex is a point with real coordinates) and its intersection with a plane. For now, we only consider the case where the intersection is an ellipse (the case of the hyperbola will be discussed later). It is easy to prove that the orthogonal projection of the cone's vertex onto the plane, lies on the ellipse's major axis (cf. fig. 2a and §5). This implies that the cone's vertex lies in the plane that is orthogonal to the ellipse's supporting plane and that contains its major axis.

For our problem, this means that the optical center must lie in the plane $Y = 0$ (since the ellipse lies in plane $Z = 0$ and has the X-axis as major axis). We may further constrain its position $\mathbf{C} = (X, 0, Z, 1)^{\mathsf{T}}$, as follows [4]. The cone with $\mathbf{C}$ as vertex and that contains the calibration ellipse, is given by ($\propto$ represents equality up to scale):

$$\Lambda \propto \begin{pmatrix} a\,Z^2 & 0 & -a\,X\,Z & 0 \\ 0 & b\,Z^2 & 0 & 0 \\ -a\,X\,Z & 0 & a\,X^2 - 1 & Z \\ 0 & 0 & Z & -Z^2 \end{pmatrix}.$$

For this cone to be a right one, its upper left 3×3 matrix $\bar{\Lambda}$ must have a double eigenvalue. The three eigenvalues are:

$$b\,Z^2\,,\ \frac{a\,X^2 + a\,Z^2 - 1 \pm \sqrt{4a\,Z^2 + \left(-a\,X^2 - a\,Z^2 + 1\right)^2}}{2}$$

The second and third eigenvalues can not be equal for real values of X and Z (besides in the trivial case $X = Z = 0$). The first eigenvalue is equal to the third one if $Z = 0$ and to the second one if:

$$a\,b\,X^2 + b(a-b)\,Z^2 + (a-b) = 0. \qquad (1)$$

This equation tells us that the optical center lies on a conic in the plane $Y = 0$, given by the following matrix:

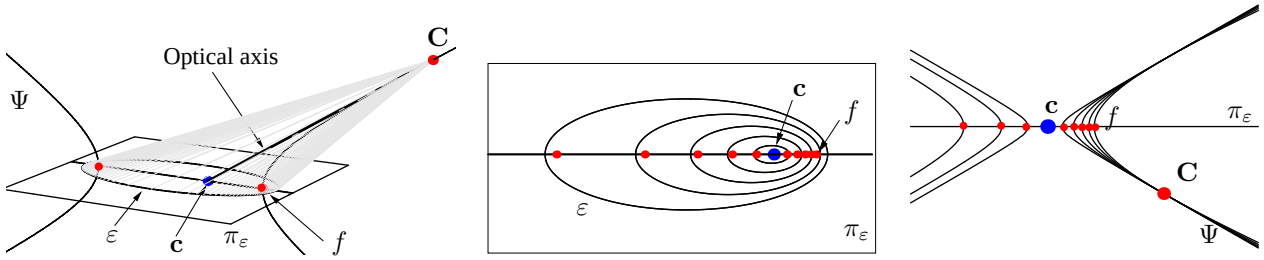
$$\Psi = \begin{pmatrix} a\,b & & \\ & b(a-b) & \\ & & a-b \end{pmatrix}.$$

This is a hyperbola, since $(a - b) < 0$. Furthermore, its asymptotes correspond to the direction of the two cylinders that contain the calibration ellipse.

Let us now consider the orientation of the optical axis. Due to (1), let us consider an optical center with:

$$Z = \pm\sqrt{\frac{a\,b\,X^2 + a - b}{b(b-a)}}.$$

In proceedings of *The 6th the Sixth Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras, (OMNIVIS'2005)*, in conjunction with ICCV'2005, october 2005.



(a) (b) (c)

Figure 2: Illustrations of the geometry of viewing cones, calibration ellipses and location of optical center. cf. text. **a**) Complete illustration for one viewing cone. **b**) View of the calibration plane, showing many cones' calibration ellipses. Note that their major axes are collinear. **c**) Side view of the hyperbolas associated with many calibration ellipses.

Here, we exclude the case $a = b$, which would correspond to the camera looking straight at the calibration plane.
The direction of the cone's axis is given by the eigenvector associated with the single eigenvalue of $\bar{\Lambda}$, augmented with a homogeneous coordinate 0:

$$\begin{pmatrix} \pm\sqrt{b(b-a)(a\,b\,X^2+a-b)} \\ 0 \\ a\,b\,X \\ 0 \end{pmatrix}. \qquad (2)$$

We now show that the cone's axis is identical with the tangent of the hyperbola Ψ in the optical center $\mathbf{C}$, which is given by (in the plane $Y = 0$):

$$\Psi \begin{pmatrix} X \\ Z \\ 1 \end{pmatrix} = \begin{pmatrix} a\,b\,X \\ \mp\sqrt{b(b-a)(a\,b\,X^2+a-b)} \\ a-b \end{pmatrix}.$$

Its point at infinity is (still in the plane $Y = 0$):

$$\begin{pmatrix} \pm\sqrt{b(b-a)(a\,b\,X^2+a-b)} \\ a\,b\,X \\ 0 \end{pmatrix}$$

i.e. it is identical with the point given in (2). Hence, for an optical center on Ψ, the optical axis is directly given by the associated tangent.
The case where the intersection between a cone and the calibration plane yields a hyperbola, given by $\mathrm{diag}(a, -b, -1)$, can be dealt with in a similar fashion. This typically occurs with very wide angle cameras. Once again, the calibration hyperbolas have their major axes aligned together. We can show that the possible viewpoints lie on an ellipse given by $\mathrm{diag}(a\,b, b(a+b), -a-b)$ and that the optical axis is tangent to it. For simplicity's sake, the rest of the article concentrates on the elliptic case; nevertheless, everything holds when some intersections are hyperbolas.

2.2. Multiple Distortion Circles

So far, we have shown that for an individual distortion circle, the associated viewing cone can be determined from the associated calibration ellipse, up to 1 degree of freedom (location on the hyperbola Ψ). We now show how to get a unique solution, when considering several distortion circles simultaneously. Let us first note that calibration ellipses corresponding to different distortion circles, are not independent: their major axes are collinear (cf. fig. 2b) [1]. Their centers are not identical however (unless they are all circles, i.e. if the camera looks straight at the plane).
Let Ψ_d be the hyperbolas for different distortion circles, given in the same coordinate frame. In the case of a single viewpoint camera, the optical center must lie on all these hyperbolas. Furthermore, the optical axis is tangent to all of them. This implies that all hyperbolas touch each other (have a double intersection point) in the optical center. This is illustrated in figure 2c. A naïve algorithm would compute the hyperbolas for all ellipses and seek their single intersection/contact point. The drawback of this situation is that very little noise can cause two hyperbolas to have no real intersection point at all, instead of a double one.
Consider now the NSVP case: to each distortion circle and viewing cone, corresponds a separate optical center. Hence, the hyperbolas won't have a single contact point anymore. However, the optical axis is shared by all viewing cones. Hence, it is the single (in general) line that is tangent to all hyperbolas. Furthermore, the individual optical centers are its contact points with the associated hyperbolas.

2.3. Calibration and Pose Estimation

A simple calibration method consists in computing the 3D point which is closest in average to all hyperbolas (see next paragraph). This gives the camera's optical center (relative to the calibration plane). Then, viewing cones can be spanned with individual calibration ellipses, and the focal

[1] This constraint is non-linear, but can be enforced when fitting the ellipses in cases where the correspondences with the calibration plane have large errors, or not uniformly distributed around the curve. It is not shown here due to lack of space.

lengths for all distortion circles are computed based on their opening angles. In the NSVP case, we would first compute the optical axis. A plausible criterion would be to find the line $\mathbf{L}$ that minimizes the sum of squared distances between itself and the closest tangent line to each hyperbola parallel to $\mathbf{L}$. In the following, this calibration approach is referred as the Right Cone Constraint method (RCC).

Closest point to the hyperbolas. Computing the orthogonal distance of a point to a general conic requires solving fourth degree polynomial [19]. Using this to compute the closest point to our set of hyperbolas is not very practical. Instead, we minimize a simpler cost function: the closest point $\mathbf{q}$ is found by solving:

$$\min_{\mathbf{q},\mathbf{q}_d} \sum_d dist(\mathbf{q},\mathbf{q}_d)^2, \text{ subject to } \mathbf{q}_d^\mathsf{T}\,\Psi_d\,\mathbf{q}_d = 0,$$

i.e. we also estimate one point per hyperbola Ψ_d that will, after convergence, be the orthogonal projection of $\mathbf{q}$ on Ψ_d. The problem is solved using the *Minimize* function of *Mathematica*. Since the function and constraints are polynomial, it uses a cylindrical algebraic decomposition that guarantees a global minimum [18].

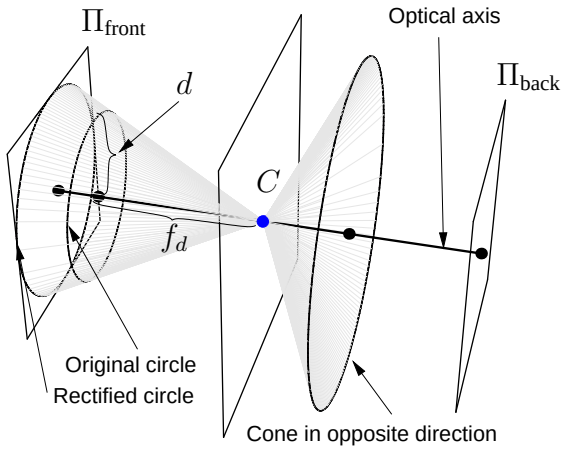


Figure 3: Viewing cones can also be seen as individual perspective cameras with different focal length. A rectified image can be obtained by projecting the distortion circles (which lie in different planes) on one plane Π_{front} (or Π_{back} for a field of view larger than $180°$).

3. Calibration with the IAC

The RCC approach relied on pose recovery from the image of one plane to calibrate. In practice, if many calibration planes are available, one would want to use them to increase robustness. We present another approach that first computes the calibration (from one or many images of planes) and then recovers the pose. The approach uses well-known results on plane-based calibration for perspective cameras [20, 16]. Indeed, it is possible to see the viewing cones in terms of many *perspective cameras*, with different focal lengths but identical principal point. In the SVP case, their extrinsic parameters are also identical, whereas an NSVP camera can be modeled by adding a translation along the optical axis per viewing cone.

Calibration. Let us consider the distortion circle of radius d and one image of a calibration plane. From point correspondences between pixels on this circle and points on the calibration plane (on the calibration ellipse), we can compute a plane-to-image homography H_d. For simplicity, let us assume that image coordinates have been translated to put the distortion center at the origin. The homography can then be decomposed such that:

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \propto \mathsf{H}_d \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \mathsf{K}_d\,\mathsf{R} \begin{pmatrix} 1 & 0 & \\ 0 & 1 & -\mathbf{t} + t_d\mathbf{r}_3 \\ 0 & 0 & \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{3}$$

where (x, y) is a calibration point, (u, v) a pixel on the distortion circle, and R and $\mathbf{t}$ a rotation matrix and translation vector representing camera pose (same for all d). The scalar t_d allows to model translational displacement of individual viewing cones along the optical axis (given by $\mathbf{r}_3^\mathsf{T}$, the third row of R), which is needed for NSVP cameras. For SVP cameras, we set $t_d = 0$ for all d. As for K_d, it is a calibration matrix[2] $\text{diag}(f_d, f_d, 1)$, where f_d is the focal length associated with the considered distortion circle. We may interpret the relation between d and f_d as a distortion function applied to a perspective camera whose undistorted image plane is π_{front} (cf. fig. 3).

Note that this parameterization only accounts for viewing cones with field of view smaller than $180°$. Larger fields of view can be modeled by adding a reflection to the rotational component of the pose, $\mathsf{R}' = \text{diag}(1, 1, -1)\mathsf{R}$, and a corresponding image plane π_{back}.

From H_d, we first compute K_d, using the approach of [20, 16]. Of course, this can be done using the homographies given for multiple images of the calibration plane.

Once the calibration is known for each viewing cone, the pose R and $\mathbf{t}$ can be computed from the homography of any distortion circle, using [14]. In the SVP case, the pose is the same for all d, and we may "average" the different estimates or better, non-linearly optimize the pose and calibrations simultaneously for all d. In the NSVP case, we first compute R, which is the same for all d. As for the position of optical centers ($\mathbf{t}$ and the t_d), we must fix one t_d, e.g. $t_0 = 0$.

[2] As mentioned in the introduction, this model does not include a skew between pixel axes or an aspect ratio different from 1. Also, it assumes that the distortion center is at the principal point. Generalizing this would be straightforward though.

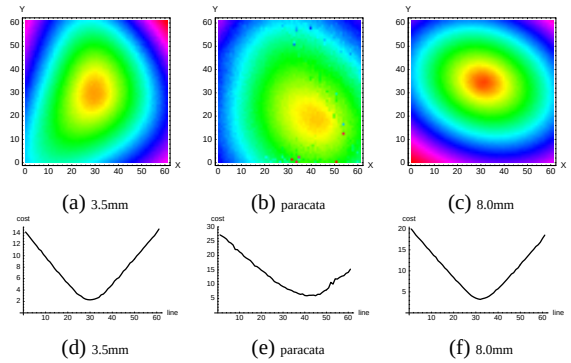
Then, from H_0, we can compute $\mathbf{t}$ and finally from the H_d, the scalars t_d. This time, non-linear optimization is recommended and straightforward to perform.

3.1. Computing the Distortion Center

Until now, we have assumed, for both algorithms, that the distortion center was known; this information was used to select the distortion circles. Tests with noiseless simulated data showed that the calibration may be quite sensitive to a bad choice of distortion center; as for real cameras, using the image center as an approximation was not satisfying in general. Hence, the distortion center should be estimated as part of the calibration process. The sensitivity of calibration we observed in simulation suggests that it should be possible to estimate the distortion center rather reliably, which was confirmed in practice.

We used the following heuristics to define an optimization criterion for the distortion center. Let us apply the IAC approach of §3 with several images as input. The plane-based calibration for each distortion circle is then capable of estimating a principal point, besides the focal length f_d. It seems plausible that the better the assumed distortion center was, the closer the estimated principal points will be to it. Since plane-based calibration is applied on images centered on the assumed distortion center, we can consider the average norm of the estimated principal points (on per distortion circle) as a measure for the goodness of the center.

Figure 4 shows the values of this measure, computed for distortion center positions on a 60×60 grid around the image center, for real cameras. The shape of the cost surface indicates that we can find the optimum distortion center using a simple steepest descent type method. We implemented such an approach that accurately finds the distortion center within a couple of minutes of computation. Note that the second row of figure 4b shows that, although the principal points used to plot it were computed individually per distortion circle, they are very densely clustered (average distance to assumed distortion center of less than 3 pixels). This suggests a high stability of the calibration.



(a) 3.5mm (b) paracata (c) 8.0mm

(d) 3.5mm (e) paracata (f) 8.0mm

Figure 4: Plots of the goodness measure for the distortion center, obtained for three tested lens (cf. §6). **a,b,c)** 60×60 grid around the image center (yellow meaning smaller). **d,e,f)** One slice of the grid, through the best position.

4. Image Rectification

Once the calibration of an SVP camera is known, an image can be perspectively rectified. Then, straight lines in the scene appear straight in the image.

Rectification is done by placing a virtual perspective camera at the actual camera's optical center. Let K^v and $\mathsf{R}^v_{3\times3}$ represent the virtual camera's calibration and orientation. A naïve approach for image creation is to render each pixel of the original (distorted) image into the virtual (distortion-free) image, and then fill out the holes by interpolation (cf. fig. 3). It is well known that an inverse approach is better. We achieve this by inverting the relation between d and the view angle $\theta(d)$. As seen in figure 11, this function is generally simple (close to linear), so easily invertable (see §4). Let $D(\theta)$ denote this inverse function. One pixel $\mathbf{q}^v$ in the rectified image is backprojected in space with $q^w = (\mathsf{K}^v\,\mathsf{R}^v)^{-1} q^v$. Then, we compute the angle θ between this pixel and the Z-axis (the optical axis of the original camera). Finally, the corresponding position in the original image is given by $\mathsf{K}_{D(\theta)}\,\mathbf{q}^w$.

5. Practical Issues

5.1. Dense Camera–Plane Correspondences

The easiest approach we found to get dense correspondences between the calibration plane and the camera is to use a flat screen as plane. We used a simple coded structured light algorithm [12], which consists in successively displaying patterns of horizontal and vertical black and white stripes on the screen to encode the position of each screen pixel (cf. fig. 5). Then, for each camera pixel, we identify the corresponding position on the calibration plane by decoding the observed intensities in each pattern. When performed in a controlled environment (low-constant ambient lighting, screen of high contrast and resolution), the accuracy of such a method is reasonably good (around ± 2 pixel of error on average). Since the points located on the distortion circles are given in floating point coordinates, we compute their correspondences by a weighted sum of the correspondences recovered for the four closest image pixels.

5.2. Omnidirectional Cameras

There are several issues worth mentioning for omnidirectional cameras. If the field of view is larger than $180°$, then some distortion circles will have viewing cones that actually approach planes. Their pose can not be estimated the same way as for true cones. These can be detected as the ones

(a) (b) (c)

Figure 5: Projected patterns for correspondences are horizontal and vertical black and white stripes. Images taken with **a**) the Goyo 3.5 mm, **b**) catadioptric, and **c**) paracatadioptric camera.

whose correspondences on the calibration plane are close to collinear. They can then be discarded from the actual calibration procedure and we may attribute them an unknown focal length. In the case of the IAC, rank deficient homographies are discarded resulting once again in an unknown focal length. Their actual value can be determine afterward by interpolation (see §6)

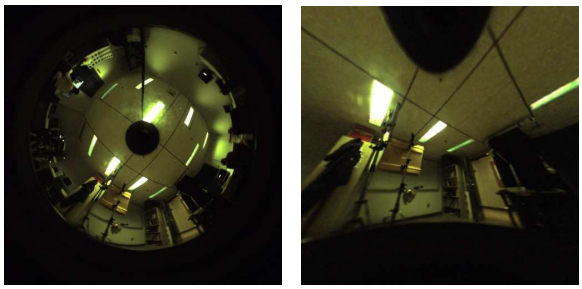
In the NSVP case, we still need to compute the offset t_d on the optical axis of such (almost) "viewing planes", since it may differ from that of other viewing cones. This is simple once the optical axis has been computed using other distortion circles and their exact opening angle has been determined: the cones' offset can be computed such that they go through the extracted correspondences in the calibration plane.

6. Experiments

We used a wide-angle Goyo 3.5 mm lens combined with a CCTV A201bc Basler camera, a Cosmicar 8.0 mm with little distortion, a paracatadioptric camera built with a Cosmicar 12.5 mm (referred to as "paracata" in the text), and also a homemade catadioptric device built from a Fujinon 12.5 mm lens, pointed at a roughly spherical mirror (cf. fig. 7). The calibration plane of known euclidean structure was a 21 inch CRT screen in all cases, except for the paracatadioptric camera where a multimedia projector was carefully placed in a fronto-parallel position w.r.t. to a wall. Even though the alignment was not perfect and the camera self-occluded, it did not affecte the solution significantly. The only non-linear optimizations that were performed to obtain the following results are in the hyperbola intersection algorithm and the pose estimation for the IAC approach.

Figure 10 gives the computed focal length of the 3.5 mm, 8.0 mm and paracatadioptric lenses, w.r.t. the distance d to the distortion center, using both methods. The wide-angle camera could already be calibrated from a single image of the screen with both approaches (cf. fig. 9a,b for the RCC), although better results were obtained using five images and the IAC approach. The paracatadioptric camera was calibrated with the two approaches with very similar results (cf. fig. 9c); however, the RCC algorithm gave the best results.

Figure 7: Catadioptric camera built from a Basler A201bc camera with a Fujinon 12.5 mm lens pointed at a roughly spherical mirror.

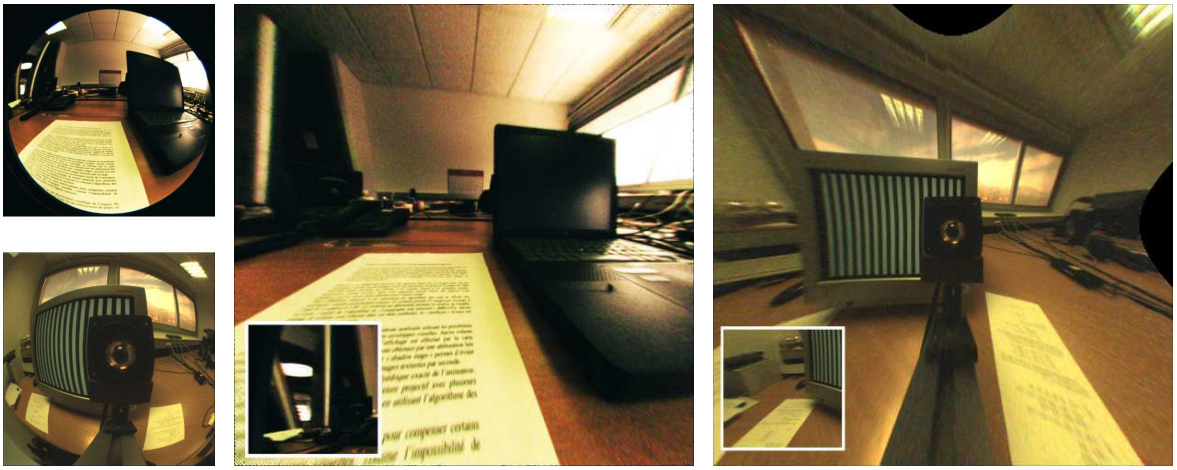


(a) (b)

Figure 8: Image rectification of the paracatadioptric camera. **a**) Original image. **b**) Rectified image for a rotated camera.

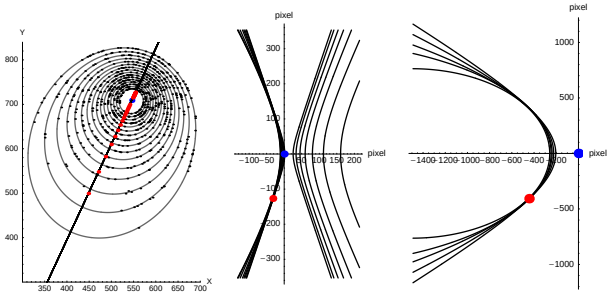
Indeed the radius for which the focal length is 0 corresponds better to the measurement of the correspondences' colinearity. The radial configuration of the catadioptric camera was not perfect. Nevertheless, the distortion center could be found and a satisfying calibration could be obtained with both methods. The IAC approach gave the best results because it could take advantage of up to eight images, which is more robust to the imperfect configuration of the camera. We also observed that the calibration is more stable for the lens with the wider field of view. Indeed, when there is very little distortion in the image, the hyperbolas' curvatures are similar, which induces more instability for the recovered camera pose. We also calibrated the 8.0 mm with the OpenCV library [11], and found the recovered focal lengths to be inside the result's uncertainty interval. Image rectification also yielded almost identical results.

In practice, only a subset of distortion circles are used for calibration; others can then be extrapolated or interpolated from a polynomial fitting of the data. Let us define this polynomial p; from the camera model, it is best to ensure that its derivative at 0 (corresponding to the distortion center) is 0. This constraint is due to the symmetry of the distortion model, and can be enforced by using a polynomial of the form $p(d) = a_0 + a_1 d^2 + ... + a_{n-1} d^n$. In practice, polynomials of degree 3 appeared to be sufficient. To handle the case of omnidirectional cameras more appropriately, the interpolation is carried out with the view angle instead of the focal length. In this case, a polynomial passing through 0 can also be fitted (see fig. 11).



(a) (b) (c)

Figure 6: Image rectification. **a**) Original images. **b**) Rectified image for the Goyo 3.5 mm. **c**) Rectified image for the home-made catadioptric camera. Small inset images show rectification of the border regions.



(a) (b) (c)

Figure 9: Calibration with the RCC approach. **a**) Fitted ellipses for the Goyo lens and **b**) corresponding hyperbolas and computed intersection. **c**) For the paracatadioptric camera, the intersection between the calibration plane and the cones yielded ellipses and hyperbolas, constraining the viewpoint to lie respectively on hyperbolas and ellipses.

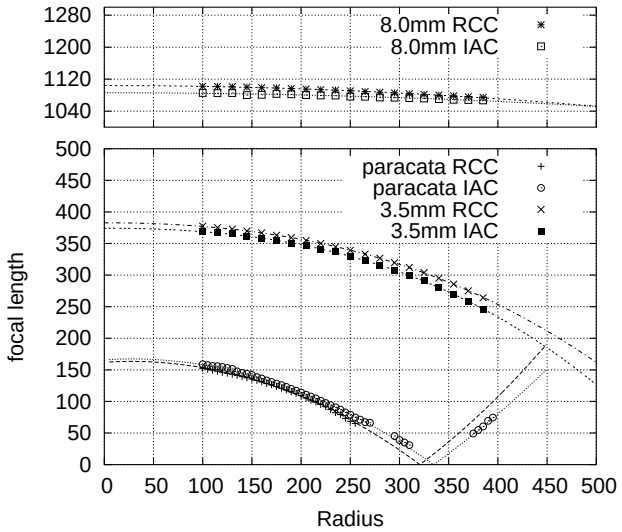


Figure 10: Recovered focal length (in pixel) from the two algorithms and extrapolated values from polynomial fitting of the data for the tested cameras.

Evaluating the results based on the reprojection error can lead to biased conclusions in the case of a generic model. Indeed, the model offers more freedom which allows to fit the data better. This analysis goes together with the comparison between SVP and NSVP constraints and the displacement t_d of the viewpoints on the optical axis. This topic is to be explored more thoroughly, but the preliminary results obtained with the IAC approach indicate that our model is useful (see table 2). They show that the paracatadioptric and to a lesser extent the 3.5 mm are probably NSVP. The displacement along the optical axis confirms this observation; the shape of the curves also leads us to believe that it is not a result of overfitted data (see fig. 12).

More meaningful quantitative results were obtained for the

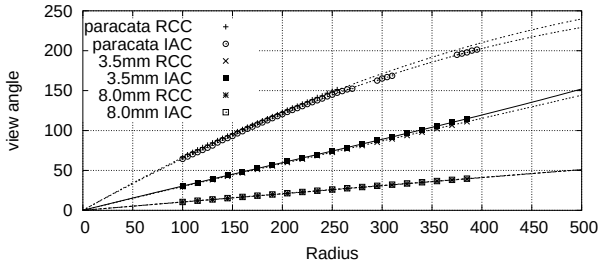


Figure 11: View angle in degrees for the tested cameras.

	Position			Angle		
Algorithms	p_{01}	p_{12}	p_{02}	a_{01}	a_{12}	a_{02}
Ground truth	5	5	10	0°	0°	0°
RCC	4.99	5.09	10.08	2.25°	0.89°	2.91°
IAC	4.89	5.13	9.99	0.6°	0.4°	1.1°

Table 1: Result for pose estimation. The camera was moved to three positions with known relative motion. Coefficients p_{ij} and a_{ij} denote the distance (in centimeters) and relative angle (in degrees) between camera positions i and j.

Camera	Constraint	
Algorithms	SVP	NSVP
paracata	9.10	1.01
3.5 mm	5.18	2.23
8.0 mm	2.20	1.35

Table 2: Comparison of the average reprojection error for different constraint of the viewpoint.

Goyo lens, using a pose estimation procedure. Using a translation stage, the camera was moved to three positions with known relative motion (no rotation, known translation). Using the calibration information (obtained using other images), the pose of the camera relative to the calibration plane was computed for all three positions (the NSVP configuration being very similar in all three cases). From this, the relative motions were computed and compared to the ground truth. The results presented in table 1 show a good stability for both methods.

Finally, images from the three panoramic cameras were rectified based on the calibration results (cf. fig. 6 and 8). Especially for the wide-angle Goyo lens (with little NSVP), the rectification seems to be very good, even towards the image borders (cf. the inset image in fig. 6b). As for our paracatadioptric camera, the rectification is very good, although not perfect, a likely result of its NSVP. Finally, the rectification of our home-made catadioptric device is also surprisingly good for a large part of the image, especially around the borders. The remaining distortions in the center were found to be caused by a small bump on the "mirror's" surface.

7. Summary and Conclusion

We have proposed new calibration approaches for a camera model that may be a good compromise between flexibility and stability for many camera types, especially wide-angle ones. The RCC approach is theoretically very interesting but its practical usability remains limited. This is due to the fact that only one calibration plane can be used directly. We also intend to perform a better analysis of its stability in the future.

The IAC algorithm, especially when used with many images, showed greater stability. It is also the basis for the distortion center estimation which is an important issue of the calibration.

Our approach may be especially suitable for unknown configurations (SVP/NSVP, mirror equation) or slightly misaligned catadioptric systems.

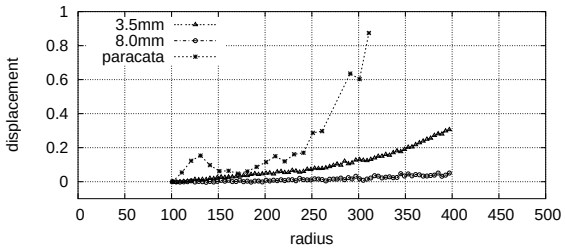


Figure 12: Displacement (in mm) along the optical axis for the 8.0 mm, the 3.5 mm and the paracatadioptric. The general curves' form of the two last leads us to think that the NSVP optimization is not a result of overfitting.

References

[1] R. Bajcsy, S.-S. Lin. True single view point cone mirror omnidirectional catadioptric system. ICCV 2001. 1

[2] S. Baker, S.K. Nayar. A Theory of Single-Viewpoint Catadioptric Image Formation. IJCV, 35(2), 1–22, 1999. 1

[3] J.P. Barreto, K. Daniilidis. Unifying image plane liftings for central catadioptric and dioptric cameras. OMNIVIS 2004. 1

[4] W. Boehm, H. Prautzsch. *Geometric Concepts for Geometric Design*. A.K. Peters, 1994. 2

[5] G. Champleboux, S. Lavallée, P. Sautot, P. Cinquin. Accurate Calibration of Cameras and Range Imaging Sensors: the NPBS Method. ICRA 1992. 1

[6] D. Claus, A.W. Fitzgibbon. A Rational Function Model for Fish-eye Lens Distortion. CVPR 2005. 1

[7] W. Gander, G.H. Golub, R. Strebel. Fitting of Circles and Ellipses. *BIT*, 34, 556–577, 1994.

[8] C. Geyer, K. Daniilidis. A Unifying Theory for Central Panoramic Systems and Practical Applications. ECCV 2000. 1

[9] K.D. Gremban, C.E. Thorpe, T. Kanade. Geometric Camera Calibration using Systems of Linear Equations. ICRA 1988. 1

[10] M. Grossberg, S. Nayar. A general imaging model and a method for fi nding its parameters. ICCV 2001. 1

[11] Intel Open Source Computer Vision Library. URL http://www.intel.com/research/mrl/research/opencv/. 6

[12] J. Salvi, J. Pagès, J. Batlle. Pattern codifi cation strategies in structured light systems. *Pattern Recognition*, 37(4), 827–849, 2004. 5

[13] D.E. Stevenson, M.M. Fleck. Nonparametric correction of distortion. TR 95–07, University of Iowa, 1995. 1

[14] P. Sturm. Algorithms for plane-based pose estimation. CVPR 2000. 4

[15] P. Sturm, S. Ramalingam. A generic concept for camera calibration. ECCV 2004. 1

[16] P. Sturm, S. Maybank. On Plane-Based Camera Calibration. CVPR 1999. 4

[17] R. Swaminathan, M. Grossberg, S. Nayar. Caustics of catadioptric cameras. ICCV 2001. 1

[18] E.W. Weisstein et al. Cylindrical algebraic decomposition. http://mathworld.wolfram.com/CylindricalAlgebraicDecomposition.html. 4

[19] Z. Zhang. Parameter estimation techniques: A tutorial with application to conic fi tting. TR 2676, INRIA, 1995. 4

[20] Z. Zhang. A Flexible New Technique for Camera Calibration. PAMI, 22(11), 1330–1334, 2000.

4

Chapter 7

Self-Calibration

Paper 15 [18]: S. Ramalingam, P. Sturm, and S.K. Lodha. Towards generic self-calibration of central cameras. In *Proceedings of the 6th Workshop on Omnidirectional Vision, Camera Networks and Non-Classical Cameras, Beijing, China*, pages 20–27, October 2005.

Paper 16 [37]: J.-P. Tardif, P. Sturm, and S. Roy. Self-calibration of a general radially symmetric distortion model. In H. Bischof and A. Leonardis, editors, *Proceedings of the 9th European Conference on Computer Vision, Graz, Austria*, Lecture Notes in Computer Science, May 2006.

Towards Generic Self-Calibration of Central Cameras

Srikumar Ramalingam[1&2], Peter Sturm[1], and Suresh K. Lodha[2]
[1] INRIA Rhône-Alpes, GRAVIR-CNRS, 38330 Montbonnot, France
[2] Dept. of Computer Science, University of California, Santa Cruz, CA 95064, USA
{Srikumar.Ramalingam, Peter.Sturm}@inrialpes.fr, lodha@soe.ucsc.edu

Abstract

We consider the self-calibration problem for the generic imaging model that assigns projection rays to pixels without a parametric mapping. In this paper, we consider the central *variant of this model, which encompasses all camera models with a single effective viewpoint. Self-calibration refers to calibrating a camera's projection rays, purely from matches between images, i.e. without knowledge about the scene such as using a calibration grid. This paper presents our first steps towards generic self-calibration; we consider specific camera motions, concretely, pure translations and rotations, although without knowing rotation angles etc. Knowledge of the type of motion, together with image matches, gives geometric constraints on the projection rays. These constraints are formulated and we show for example that with translational motions alone, self-calibration can already be performed, but only up to an affine transformation of the set of projection rays. We then propose a practical algorithm for full metric self-calibration, that uses rotational and translational motions.*

1. Introduction

Many different types of cameras have been used in computer vision. Existing calibration and self-calibration procedures are often taylor-made for specific camera models, mostly for pinhole cameras (possibly including radial or decentering distortion), fisheyes, specific types of catadioptric cameras etc. see examples e.g. in [1, 2, 7, 4, 9, 11].

A few works have proposed calibration methods for a highly generic camera model that encompasses the above mentioned models and others [5, 3, 6, 15, 14]: a camera acquires images consisting of pixels; each pixel captures light that travels along a projection ray in 3D. Projection rays may in principle be positioned arbitrarily, i.e. no functional relationship between projection rays and pixels, governed by a few intrinsic parameters, is assumed. Calibration is thus described by:

- the coordinates of these rays (given in some local coordinate frame).
- the mapping between rays and pixels; this is basically a simple indexing.

One motivation of the cited works is to provide flexible calibration methods that should work for many different camera types. More importantly these calibration works also provide the flexibility to build newer cameras for special applications and still calibrate them with existing techniques. The proposed methods rely on the use of a calibration grid and some of them on equipment to carry out precisely known motions.

The work presented in this paper aims at further flexibility, by addressing the problem of self-calibration for the above generic camera model. The fundamental questions are: can one calibrate the generic imaging model, without any information other than image correspondences, and how? This work presents a first step in this direction, by presenting principles and methods for self-calibration using specific camera motions. Concretely, we consider how pure rotations and pure translations may enable generic self-calibration.

Further we consider the *central* variant of the imaging model, i.e. the existence of an optical center through which all projection rays pass, is assumed. Besides this assumption, projection rays are unconstrained, although we do need some continuity (neighboring pixels should have "neighboring" projection rays), in order to match images.

2. Problem Formulation

We want to calibrate a central camera with n pixels. To do so, we have to recover the directions of the associated projection rays, in some common coordinate frame. Rays need only be recovered up to a euclidean transformation, i.e. ray *directions* need only be computed up to rotation. Let us denote by $\mathbf{D}_p$ the 3-vector describing the direction of the ray associated with the pixel p.

Input for computing ray directions are pixel correspondences between images and the knowledge that the motion between images is a pure rotation or a pure translation (with unknown angle or length). For simplicity of presentation, we assume that we have dense matches over space and time, i.e. we assume that for any pixel p, we have determined all pixels that match p at some stage during the rotational or translational motion. Let us call a complete such set of matching pixels, a *flow curve*. Flow curves can be obtained

from multiple images undergoing the same motion (rotations about same axis but not necessarily by the same angle; translation in same direction but not necessarily with constant speed) or from just a pair of images I and I'.

In Figure 1 we show flow curves obtained from a single image pair each for a pure translation and a pure rotation about an axis passing through the optical center. Let p and p' refer to two matching pixels, i.e. pixels observing the same 3D point in I and I'. Let p'' refer to the pixel that in I' matches to pixel p' in I. Similarly let p''' be the pixel that in I' matches to pixel p'' in I, and so forth. The sequence of pixels $p, p', p'', p''', \ldots$ gives a subset of a flow curve. A dense flow curve can be obtained in several ways: by interpolation or fusion of such subsets of matching pixels or by fusing the matches obtained from multiple images for the same motion (constant rotation axis or translation direction, but varying speed).

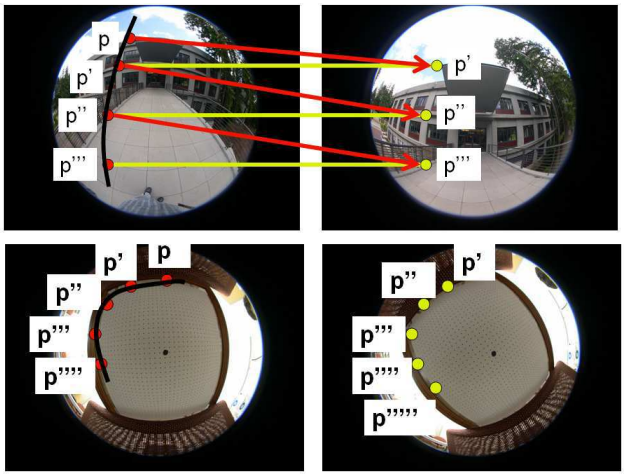


Figure 1: Illustration of flow curves: translational motion (top) and rotational motion (bottom).

3. Constraints From Specific Camera Motions

We explain constraints on self-calibration of projection ray directions that are obtained from flow curves due to specific camera motions: one translational or one rotational motion.

3.1. One Translational Motion

Consider two matching pixels p and q, i.e. the scene point seen in pixel p in image 1, is seen in image 2 in pixel q. Due to the motion being purely translational, this implies that the projection rays of these two pixels, and the motion line, the ray along which the center of the camera moves while undergoing pure translation, are *coplanar* (they indeed form an epipolar plane, although we won't use this notation in the following).

It is obvious that this statement extends to all pixels in a flow curve: their projection rays are all coplanar (and that they are coplanar with the motion line). We conclude that the ray *directions* of the pixels in a flow curve, lie on one line at infinity. That line at infinity also contains the direction of motion.

When considering all flow curves for one translational motion, we thus conclude that the ray directions of pixels are grouped into a pencil of lines at infinity, whose vertex is the direction of motion. Clearly, these collinearity constraints tell us something about the camera's calibration.

When counting degrees of freedom, we observe the following: at the outset, the directions for our n pixels, have $2n$ degrees of freedom (minus the 3 for rotation R). Due to the translational motion, this is reduced to:

- 2 dof for the motion direction
- 1 dof per flow curve (for the line at infinity, that is constrained to contain the motion direction)
- 1 dof per pixel (the position of its ray along the line at infinity of its flow curve).
- minus 3 dof for R.

3.2. One Rotational Motion

Let $\mathbf{L}$ be the rotation axis (going through the optical center). Consider two matching pixels p and q. Clearly, the associated rays lie on a right cone with $\mathbf{L}$ as axis and the optical center as vertex, i.e. the angles the two rays form with the rotation axis $\mathbf{L}$, are equal. Naturally, the rays of all pixels in a flow curve, lie on that cone. Each flow curve is associated with one such cone.

When counting degrees of freedom, we so far observe the following. Due to the rotational motion, the following dof remain:

- 2 dof for the direction of the rotation axis
- 1 dof per flow curve (for the opening angle of the associated cone).
- 1 dof per pixel (the "position" of its ray along the associated cone).
- minus 3 dof for R.

We have not yet exploited all information that is provided by the rotational motion. Besides the knowledge of rays lying on the same cone, we have more information, as follows. Let Θ be the (unknown) angle of rotation. Then, the angular separation between any two rays whose pixels match in the two images, is equal to Θ. Hence, the rays for each set of pixels that are transitive 2-view matches, can be parameterized by a single parameter (an "offset" angle). We remain with:

- 2 dof for the direction of the rotation axis
- 1 dof for the rotation angle Θ
- 1 dof per flow curve (for the opening angle of the associated cone).
- 1 dof per set of matching pixels (the "offset" of its ray along the associated cone).
- minus 3 dof for R.

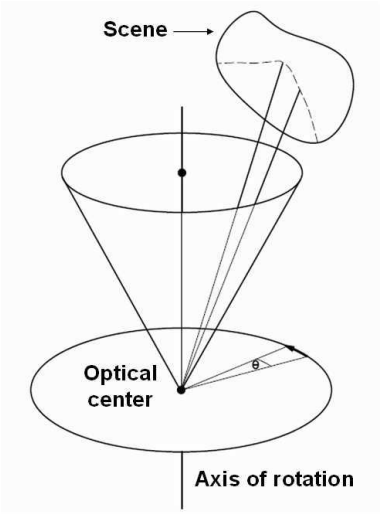


Figure 2: The rays of the pixels in the rotation flow curve form a cone.

3.2.1. Closed Flow Curves

Let us consider what we can do in addition, if the rotation axis "pierces" the image, i.e. if there is a pixel whose ray is collinear with the rotation axis. Then, in the vicinity of that pixel, *closed* flow curves can be obtained. For example, for a pinhole camera with square pixels and no skew, a rotation about its optical axis produces flow curves in the form of circles centered in the principal point, covering a large part of the image.

What does a closed flow curve give us? Let us "start" with some pixel p on a closed flow curve, and let us "hop" from one matching pixel to another, as explained in Figure 1. We count the number of pixels until we get back to p. Then, the rotation angle Θ can be computed by dividing 360° by that number. Of course, pixel hopping may not always lead us exactly to the pixel we started with, but by interpolation, we can get a good approximation for Θ. Furthermore, this can be done by starting from every single pixel on every closed flow curve, and we may hope to get a good average estimation of Θ.

4. Multiple Translational Motions

In this section, we explain that multiple translational motions allow to recover camera calibration up to an affine transformation. First, it is easy to explain that no more than an affine "reconstruction" of projection rays is possible here. Let us consider one valid solution for all ray directions $\mathbf{D}_i$, i.e. ray directions that satisfy all collinearity constraints associated with flow curves (cf. section 3.1). Let us transform all ray directions by an affine transformation of 3-space

$$\begin{pmatrix} \mathsf{A} & \mathbf{b} \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix}$$

i.e. we apply the 3×3 homography A to the $\mathbf{D}_i$. This may be seen as a projective transformation inside the plane at infinity, although we prefer to avoid any possible confusion by such an interpretation, and simply think of the mapping as an affine one. Clearly, the $\mathbf{D}'_i = \mathsf{A}\mathbf{D}_i$ also satisfy all collinearity constraints (collinearity is preserved by affine and projective transformations).

This situation is very similar to what has been observed for perspective cameras: a completely uncalibrated perspective camera can be seen as one whose rays are known up to an affine transformation of 3-space: the role of A is played by the product KR of calibration and rotation matrix; since calibration is only required up to rotation, only K matters. So, the rays of a perspective camera are always (at least) "affinely" calibrated (not to confuse with the concept of affine calibration of a stereo system). Even with uncalibrated perspective cameras, 3D reconstruction is possible, but only up to projective transformations. Now, when moving a camera by pure translations, no further information on calibration can be gained, although a projective reconstruction may be upgraded to affine [12].

Coming back to our generic camera model, it is thus obvious that from pure translations, we can not reach farther than recovering the rays up to an affine transformation (the situation would be different for example if multiple translations were considered with the knowledge that speed is *constant*).

We now provide a simple constructive approach to recover actual affine self-calibration. Let us consider 4 translational motions, in different directions such that no 3 directions are collinear. Let us carry out the translations such that the FOE (focus of expansion) is inside the image, i.e. such that there exists a pixel for each motion whose ray is parallel to the motion line. Let these 4 pixels be pixels 1 to 4. Since we can recover ray directions up to a 3×3 homography only, we may, without loss of generality, attribute arbitrary coordinates to the directions $\mathbf{D}_1 \cdots \mathbf{D}_4$ (such that no 3 of them are collinear). We now alternate between the following two steps:

1. Compute the line at infinity of ray directions for all

flow curves for which two ray directions have already been determined.

2. Compute ray directions of pixels who lie on two flow curves whose line at infinity has already been determined.

Repeat this until convergence, i.e. until no more directions or lines at infinity can be computed.

In the first iteration, 6 lines at infinity can be computed, for the flow curves that link pairs of our 4 basis pixels. After this, 3 new ray directions can be recovered.

In the second iteration, 3 new lines at infinity are computed. From then on, the number of computable ray directions and lines at infinity increases exponentially in general (although pixels and flow curves will be more and more often "re-visited" towards convergence).

This algorithm is deterministic, hence the computed ray directions will necessarily be an "affine reconstruction" of the true ones.

There are a few issues with this "proof":

- the construction does not state sufficient condition in order to calibrate all ray directions of a camera; it just says that the ray directions we do calibrate (i.e. that are attained by the construction scheme), are indeed up to the same global affine transformation equal to the true ones.
- a practical implementation of the above algorithm will have to deal with noise: for example, computed flows curves are not exact and the lines at infinity computed for flow curves that contain the same pixel, will not usually intersect in a single point.
- strictly speaking, the above scheme for self-calibration is not valid for cameras with finitely many rays. To explain what we mean, let us consider a camera with finitely many rays, in two positions. In general, i.e. for an arbitrary translation between the two positions, a ray in the second camera position, will have zero probability of cutting any ray in the first camera positions! Hence, the concept of matching pixels has to be handled with care. However, if we consider a camera with infinitely many rays (that completely fill some closed volume of space), a ray in one position will always have matching rays in the other position (unless it is outside the other position's field of view). Hence, our constructive proof given in this section, is valid for cameras with infinitely many rays. In future work we will clarify this issue more properly.

5. Self-Calibration Algorithm

We put together constraints derived in section 3 in order to propose a self-calibration algorithm that requires rotational and translational motions.

5.1. Two Rotational Motions

From a single rotation we obtain the projection rays in several cones corresponding to flow curves. The local offsets and the opening angles are unknown in each of the cones. In the presence of another rotation we obtain a new set of cones around a different axis. It is possible to compute the projection rays without any ambiguity using these two motions. However we propose a simple and practical algorithm for computing the projection rays with two rotations and an additional translation in the next subsection.

5.2. Two Rotations and One Translation

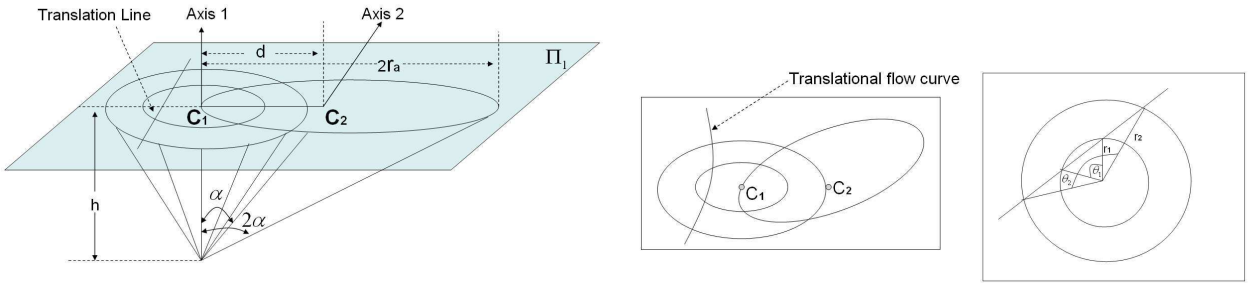
By combining our observations so far, we are able to formulate a self-calibration algorithm that does not require any initialization. It requires 2 rotational and 1 translational motions with at least one closed flow curve.

The translational motion only serves here to fix the offset angles of all cones arising from the two rotational motions. Let p_1 be the center pixel of the first rotation and p_2 that of the second one. Consider the translational flow curve that contains p_1. All pixels on one side of the flow curve starting from p_1 will have the same ϕ_1. Similarly let ϕ_2 refer to the offset angle for pixels lying on the flow curve passing through p_2. The same holds for the second rotation.

Without loss of generality, we set the first rotation axis as the Z-axis, and set $\phi_1 = 0$ for p_2, and $\phi_2 = 0$ for p_1. Hence, the ray associated with p_2 is determined up to the angle α between the two rotation axes. Below, we explain how to compute this angle. If we already knew it, we could immediately compute all ray directions: for every pixel p, we know a line going through $\mathbf{D}_1$ (associated with its ϕ_1) and similarly for $\mathbf{D}_2$. The pixel's ray is simply computed by intersecting the two lines.

What about pixels whose rays are coplanar with the two rotation axes? This is not a problem because every computed ray direction gives the angle of the associated cone. Hence, all pixels on that cone can directly by reconstructed, by intersecting the line issuing from $\mathbf{D}_1$ or $\mathbf{D}_2$ with its cone.

This reasoning is also the basis for the computation of α. However in general the flow curves are not always closed. Thus we present a more detailed approach which can work with several open flow curves. In order to understand the algorithm let us first visualize a setup as shown in Figure 3(a). Consider a plane π_1 orthogonal to the first rotation axis. The intersection of the cones associated with the first rotation axis and the plane π_1 will form concentric circles $C_1, C_2, ..C_n$ with radii $r_1, r_2, ..r_n$. Let h be the distance of the camera center from π_1. Thus the opening angle of the i_{th} cone can be computed if we know the r_i and h. Now



(a) (b) (c)

Figure 3: a) We show two rotation axes and a plane orthogonal π_1 to the first axis. We compute the rotation axis, radii of the concentric circles around the first rotation axis, the distance between C_1 and C_2 and eventually the angle α between the two axis. See text for more details. b) Two rotation flow curves and one translation flow curve on the image. c) Concentric circles from rotation and a line translation on π_1.

let us consider the intersection of the cones from the second rotation with the plane π_1. These intersections are ellipses.

As we observed earlier translational flow curves consists of pixels whose corresponding projection rays are coplanar. The intersection of these coplanar rays and the plane π_1 is a line. We use this information to compute the relation between r_i and later the offset angles.

Here we briefly describe the technique used in computing r_i. Let θ_1 and θ_2 be the two angles subtended by a single translational curve with C_1 and C_2. We can compute the angle subtended by two consecutive pixels in a rotation flow curve. Thus it is possible to obtain the angle subtended by any two pixels on the flow curve.We assume r_1 to be unity. Thus r_2 can be computed as below.

$$r_2 = \frac{cos(\frac{\theta_1}{2})}{cos(\frac{\theta_2}{2})}$$

Similarly we can compute the radii of the circles of all other cones.

The distance between C_1 and C_2, the distance d between the two axes on π_1, can be computed by constructing a flow curve passing through the center pixel (pixel corresponding to the axis) of the second rotation and estimating its radius. Finally we need to compute the value of h to compute α. In order to compute h let us consider the flow curve of the second rotation passing through the center pixel of the first rotation. The corresponding cone intersects π_1 as an ellipse. We intersect this flow curves with the flow curves about the first axis to obtain some 3D points on π_1. These points can be used to parameterize the ellipse. Once we know the major radius r_a of the ellipse we can compute h and α as shown below.

$$tan(2\alpha) = \frac{2tan(\alpha)}{1 - tan^2(\alpha)},\ \frac{2r_a}{h} = \frac{\frac{d}{h}}{1 - (\frac{d}{h})^2},\ \alpha = tan^{-1}(\frac{d}{h})$$

The algorithm does not require all flow curves to be closed. For example in Figure 5 we show the scenario where we calibrate a fisheye camera with only few closed flow curves.

5.3. Many Rotations and many Translations

For example, once we know the projection rays for a part of the image and the inter-axis angle α, we can compute the projection rays for pixels in the corners of the image using flow curves from two different translational motions or alternatively, from a single rotational motion.

6. Experiments

We tested the algorithm of section 5.2 using simulated and real cameras. For the real cameras, ground truth is difficult to obtain, so we visualize the self-calibration result by performing perspective distortion correction.

6.1. Dense Matching

It is relatively easy to acquire images in favorable conditions. For pure translations, we use a translation stage. As for pure rotations, one could use a tripod for example, but another possibility is to point the camera at a far away scene and perform hand-held rotations. To make the image matching problem simpler we used planar surfaces. We considered two scenarios. The first approach uses simple coded structured light algorithm [16], which involves in successively displaying patterns of horizontal and vertical black

and white stripes on the screen to encode the position of each screen pixel. In the second scenario we consider a planar scene with black dots. In both these cases we do not know the physical coordinates of the scene. We used OpenCV library to perform dense matching [13]. Neighborhood matches were used to check the consistency in matching and to remove false matches. Planar scene was used to simplify the matching process. However our calibration algorithm is independent of the nature of the scene. We tested our algorithm with simulations and real data. In simulations we tested a pinhole camera with and without radial distortions. The virtual pinhole camera, constructed using an arbitrary camera matrix, is made to capture a random surface. We obtained matches in the case of pure translation and pure rotations. The flow curves and calibrated 3D rays are shown in Figure 4. We used ellipse parametrization for fitting the flow curves. It is easy to realize that the flow curve in the case of rotation is an ellipse for perspective cameras. The ellipse fitting was reasonably accurate for fisheye cameras as well. In the case of central catadioptric cameras the flow curves will not be ellipses. In such scenarios we may need to use nonparametric approaches. As expected we obtained accurate results in simulations and it confirmed the validity of our algorithm.

Secondly we tested our algorithm on Nikon coolpix fisheye lens, FC-E8, with a field of view of 183 degrees. In Figure 5 we show the translation and rotation flow curves. We fitted ellipses for both the rotational and translational flow curves.

6.2. Distortion Correction

Once a camera is calibrated, one can perform distortion correction in a straightforward manner. We do this by placing a virtual perspective camera at the optical center of the calibrated real camera, and apply the following simple rendering scheme. One has to specify a field of view and the image resolution of the virtual camera, i.e. a focal length and the size of the distortion-corrected image. Further, one needs to specify the virtual camera's orientation. By default, we choose a rotation such that the center pixel of the virtual camera and of the real camera, have collinear projection rays.

The distortion-corrected image is rendered as follows. For every pixel of the image to be rendered, we compute its projection ray, using the specified focal length and camera orientation. We then determine the k closest (in terms of angle) projection ray(s) of the real camera. We look up the RGB values of the associated pixels in the original, distorted image, and interpolate them to determine the RGB value of the pixel to be rendered. Different interpolation schemes are possible, i.e. nearest neighbor interpolation for $k = 1$ or a weighted average (weights depending on angle between real and virtual projection ray) for $k > 1$.

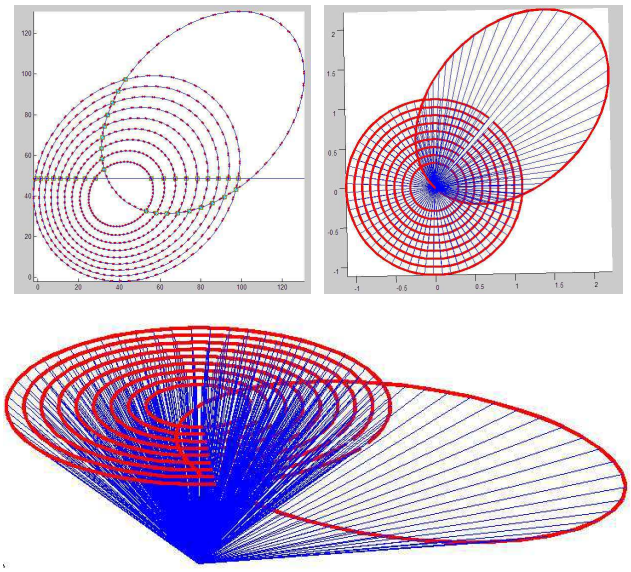


Figure 4: Top left: flow curves associated with a single rotation on a perspective image. We also fitted ellipses on the flow curves to analytically compute the intersections with other flow curves. Top right and bottom: projection rays after calibration in two different views.

For example we show the perspectively synthesized images in Figure 6. The minor artifacts could be due to the imprecision in the experimental data during rotation. Nevertheless, the strong distortions of the camera have been corrected to a large extent.

7. Conclusions

We have studied the generic self-calibration problem and calibrated general central cameras using different combinations of pure translations and pure rotations. Our initial simulations and experimental results are promising and show that self-calibration may indeed be feasible in practice. As for future work, we are interested in relaxing the constraints on the camera model and the motion scenarios.

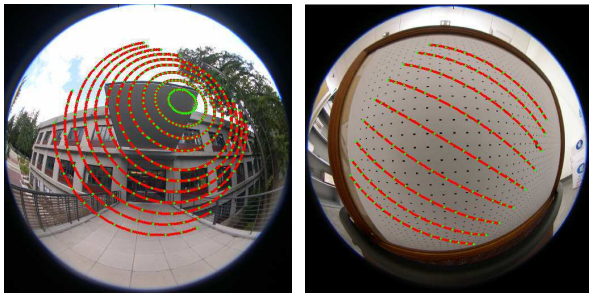
References

[1] J.P. Barreto and H. Araujo. Paracatadioptric camera calibration using lines. *ICCV*, 1359–1365, 2003.

[2] D.C. Brown. Close-Range Camera Calibration. Photogrammetric Engineering, 37(8), 855–866, 1971.

[3] G. Champleboux, S. Lavallée, P. Sautot, and P. Cinquin. Accurate calibration of cameras and range imaging sensors: the NPBS method. *ICRA*, 1552–1558, 1992.

[4] C. Geyer and K. Daniilidis. Paracatadioptric camera calibration. *PAMI*, 24(5):687–695, 2002.



(a) (b)

Figure 5: a) Flow curves of pure rotation on a fisheye image. b) Translational flow curves on a fisheye image.

[5] K.D. Gremban, C.E. Thorpe, and T. Kanade. Geometric camera calibration using systems of linear equations. *ICRA*, 562–567, 1988.

[6] M.D. Grossberg and S.K. Nayar. A general imaging model and a method for finding its parameters. *ICCV*, 2:108–115, 2001.

[7] R.I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[8] R.A. Hicks and R. Bajcsy. Catadioptric sensors that approximate wide-angle perspective projections. *CVPR*, 545–551, 2000.

[9] S.B. Kang. Catadioptric self-calibration. *CVPR*, 201–207, 2000.

[10] H.C. Longuet-Higgins. A computer program for reconstructing a scene from two projections. *Nature*, 293:133–135, 1981.

[11] B. Micusik and T. Pajdla. Autocalibration and 3D Reconstruction with Non-central Catadioptric Cameras. *CVPR*, 2004.

[12] T. Moons, L. Van Gool, M. van Diest, and E. Pauwels. Affine Reconstruction from Perspective Image Pairs. DARPA–ESPRIT *Workshop on Applications of Invariants in Computer Vision, Azores*, 249–266, 1993.

[13] OpenCV (Open Source Computer Vision Library), Intel, `www.intel.com/research/mrl/research/opencv/`

[14] P. Sturm and S. Ramalingam. A generic concept for camera calibration. *ECCV*, 1–13. 2004.

[15] R. Swaminathan, M.D. Grossberg, and S.K. Nayar. Caustics of Catadioptric Cameras *ICCV*, 2001.

[16] J. Salvi, J. Pages and J. Batlle. Pattern codification strategies in structured light systems. *Pattern Recognition*, 34(7), 827-849, 2004.

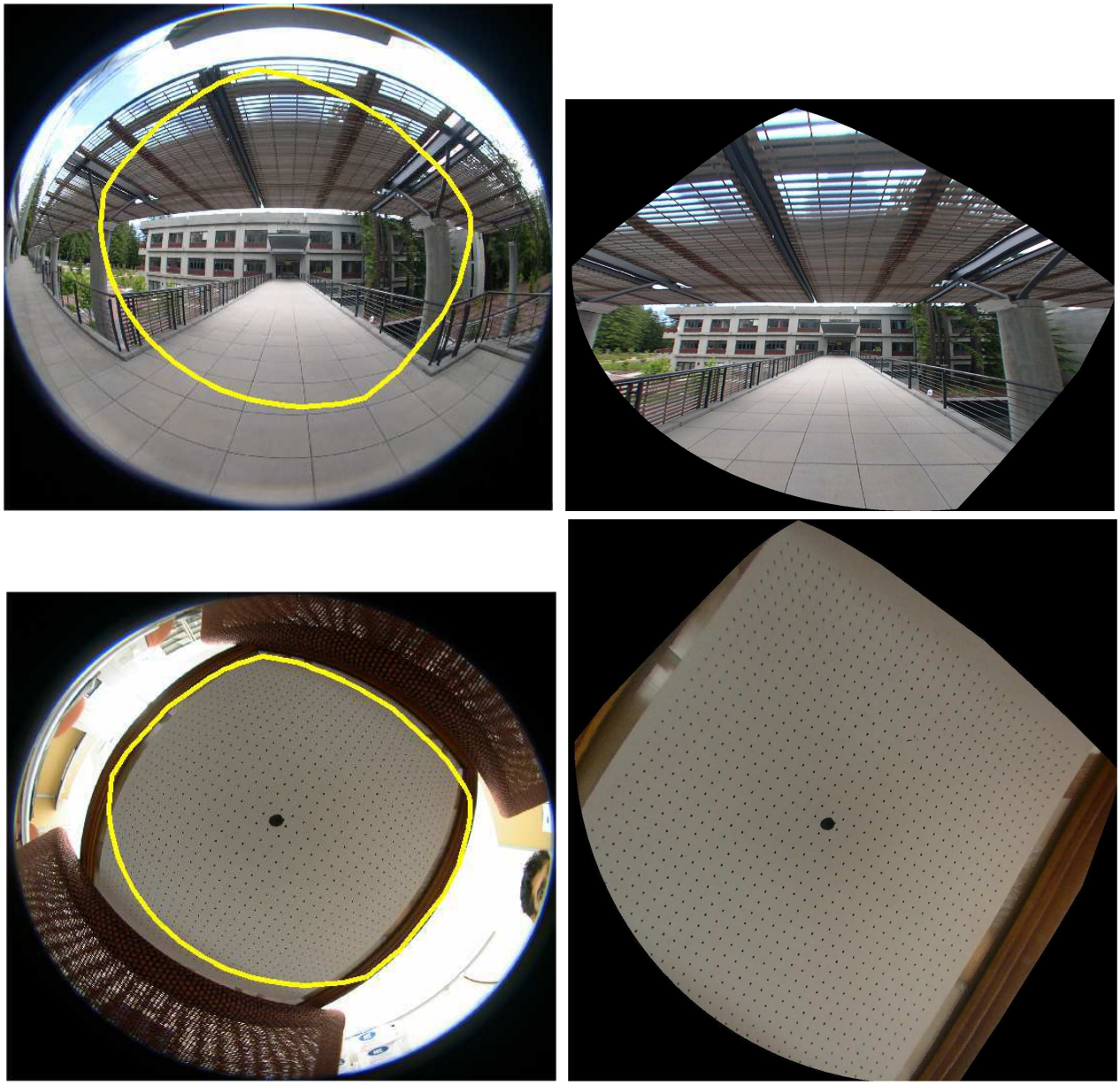


Figure 6: Top: original images with the boundaries showing the calibration region. Middle and bottom: generated perspective images.

Self-calibration of a general radially symmetric distortion model

Jean-Philippe Tardif[1], Peter Sturm[2], and Sébastien Roy[1]

[1] DIRO, Université de Montréal,Canada.
{tardifj, roys}@iro.umontreal.ca
[2] INRIA Rhône-Alpes. 38330 Montbonnot St Martin, France.
{Peter.Sturm}@inrialpes.fr

Abstract. We present a new approach for self-calibrating the distortion function and the distortion center of cameras with general radially symmetric distortion. In contrast to most current models, we propose a model encompassing fisheye lenses as well as catadioptric cameras with a view angle larger than 180°.
Rather than representing distortion as an image displacement, we model it as a varying focal length, which is a function of the distance to the distortion center. This function can be discretized, acting as a general model, or represented with e.g. a polynomial expression.
We present two flexible approaches for calibrating the distortion function. The first one is a plumbline-type method; images of line patterns are used to formulate linear constraints on the distortion function parameters. This linear system can be solved up to an unknown scale factor (a global focal length), which is sufficient for image rectification. The second approach is based on the first one and performs self-calibration from images of a textured planar object of unknown structure. We also show that by restricting the camera motion, self-calibration is possible from images of a completely unknown, non-planar scene.
The analysis of rectified images, obtained using the computed distortion functions, shows very good results compared to other approaches and models, even those relying on non-linear optimization.

1. Introduction

Most theoretical advances in geometric computer vision make use of the pin-hole camera model. One benefit of such a model is the linearity of the projection which simplifies multi-view constraints and other structure-from-motion computations. Unfortunately in many cases, this model is a poor representation of how the camera samples the world, especially when dealing with wide angle cameras where radial distortion usually occurs. In addition to these cameras, catadioptric devices (i.e. cameras pointed at a mirror) also admit a very large field of view. Their image distortion can also be seen as a type of radial distortion, although, in general, it cannot be modeled with traditional models. This is because the view angle of these cameras can be larger than 180°, which is not compatible

2

with the usual *image-displacement* approach. The effect of radial distortion is that straight lines in the scene are not in general projected onto straight lines in the image, contrary to pin-hole cameras. Many calibration algorithms can deal with distortion, but they are usually tailor-made for specific distortion models and involve non-linear optimization.

In this paper, we introduce a general distortion model, whose main feature is to consider radially symmetric distortion. More precisely, we make the following assumptions on the camera projection function:

- the aspect ratio is 1,
- the distortion center is aligned with the principal point[3],
- the projection function is radially symmetric (around the distortion center),
- the projection is central, i.e. projection rays pass through a single (effective) optical center.

Given the quality of camera hardware manufacturing, it is common practice to assume an aspect ratio of 1. As for the second and third assumptions, they are made to ensure our model is consistent with both catadioptric devices and regular fisheye cameras. Finally, a central projection is assumed for simplicity even for very large field of view cameras [1, 22] in which a non-single viewpoint might be induced by the lens [3], or by a misaligned mirror [18].

Our full camera model consists therefore of the position of the distortion center and the actual distortion function that maps distance from the distortion center to focal length. This model, together with the above assumptions, fully represents a camera projection function. It is a good compromise between traditional low-parametric camera models and fully general ones, modeling one projection ray per pixel [10, 17], in terms of modeling power and ease and stability of calibration. The model is indeed general enough to represent cameras of different types and with very different view angles.

Problem statement. In this paper, we intend to solve the proposed model relying on images of collinear points in space. Our algorithm makes no assumption on the distortion function and on the distortion center position. Only a rough initial value of the latter is needed.

Organization. A short review of the most popular distortion models is presented in the first section. The model we adopt is presented in §3. In §4 we propose a plumbline method for calibrating our model using images of collinear points. Based on this, we propose a plane-based self-calibration approach, in §5. Finally, the performance of our methods is analyzed and compared to another similar approach [6].

2. Related Work

As the field of view of a camera lens increases, the distortion occurring in the captured images becomes more and more important. Traditionally, researchers

[3] We will see that this constraint may be dropped in some cases.

have sought new models with more degrees of freedom and complexity. These models include the traditional polynomial model [11] (which can be combined with a field of view model (FOV) [6]), division [7] and rational [5]. Most of the time the models are calibrated using non-linear optimization of either a full projection model from points located on a calibration object [23] or a homography mapping from a planar grid [5]. Recent papers have also shown that radial distortion models can be calibrated linearly from a calibration grid [12] of by feature point matching between images [7, 5, 19, 20].

Other approaches focus only on calibrating the distortion function by imposing either that a straight line in space should appear straight in the image [4, 6] or that spherical objects should appear circular [16].

The aforementioned models all apply to cameras with a field of view smaller than 180° since the distortion is *image-based*. They fail to handle data captured by a camera with a view angle larger than 180°, typical for catadioptric devices. Different models and algorithm have been specifically designed to address these cases [9, 14] and their parameters have an explicit geometric interpretation rather than expressing distortion directly.

Finally, only few attempts were made to find models able to deal with dioptric systems (including radial distortion) and catadioptric ones [22, 2]. The model we propose fits in this category with the benefit that its distortion function can be general.

3. Camera Model

We describe the camera model that corresponds to the assumptions explained in the introduction. Consider a camera with canonical orientation, i.e. the optical axis is aligned with the Z-axis and image x and y-axes are parallel to world X and Y-axes respectively. Our camera model is then fully described by the position of a distortion center $(c_x, c_y)^\top$ and a distortion "function" $f : \mathcal{R} \to \mathcal{R}$, such that an image point $(x, y)^\top$ is back-projected to a 3D line spanned by the optical center and the point at infinity with coordinates:

$$\left[x - c_x,\, y - c_y,\, f(r),\, 0\right]^\top,\ r = \sqrt{(x - c_x)^2 + (y - c_y)^2}$$

The distortion function (it should actually be called "undistortion function", but we did not find this very elegant) can for example be chosen as a polynomial with even powers of r, in which case we have the division model, as used in [7, 19]. The model also subsumes fisheye models [8, 15] and cameras of the 'unified central catadioptric model' [9].

In this paper, we use two representations for the distortion function. The first one is a polynomial of a degree d to be fixed, like in the division model, however including odd powers:

$$f(r) = \sum_{i=0}^{d} \lambda_i r^i. \tag{1}$$

4

The second one is a discrete representation, consisting of a lookup table of the distortion function values at a set of discrete values for r (in practice, we use one sample per step of one pixel). We denote these values as:

$$f(r) = f_r. \tag{2}$$

Note that a constant function f allows the representation of a pinhole camera with f's value as focal length. From the above back-projection equation, it is easy to deduce equations for distortion correction, also called rectification in the sequel. This can for example be done by re-projecting the points at infinity of projection rays into a pinhole camera with the same optical center and orientation as the original camera. As for the intrinsic parameters of the (virtual) pinhole camera, we usually also adopt an aspect ratio of 1 and zero skew; if the distortion center is to have the same coordinates in the rectified image as in the original one, and if g denotes the rectified image's focal length, then the homogeneous coordinates of the rectified point are:

$$\begin{bmatrix} g & 0 & c_x \\ 0 & g & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x - c_x \\ y - c_y \\ f(r) \end{bmatrix}.$$

In the following, we introduce a few geometric notions that will be used in this paper. A **distortion circle** is a circle in the image, centered in the distortion center. Projection rays of points lying on a distortion circle span an associated **viewing cone** in space. In our model, all cones have the same axis (the optical axis) and vertex (the optical center).

Each cone can actually be understood as an individual pinhole camera, with $f(r)$ as focal length (r being the distortion circle's radius). Geometrically, this is equivalent to virtually moving the image plane along the optical axis, according to the distortion function. This situation is depicted in fig. 1. In the case of a camera with a view angle larger than 180°, the focal length becomes equal or smaller than zero. In the zero case, the cone is actually the **principal plane**, i.e. the plane containing the optical center and that is perpendicular to the optical axis. Let us call the associated distortion circle **principal distortion circle**. A negative $f(r)$ is equivalent to a camera with positive focal length, looking backward and whose image is mirrored in x and y. Typical situations for rectification are depicted in fig. 2.

Rectification for cameras with a view angle larger than 180° cannot be done as usual: the above rectification operation is no longer a bijection (two points in the original image may be mapped to the same location in the rectified one) and points on the principal distortion circle are mapped to points at infinity (fig. 2b). It is still possible to rectify individual parts of the image correctly, by giving the virtual pinhole camera a limited field of view and allowing it to rotate relative to the true camera.

4. Plumbline Calibration

In this section, we show that the distortion function f and the distortion center can be recovered linearly from the images of lines (straight edges) or points that

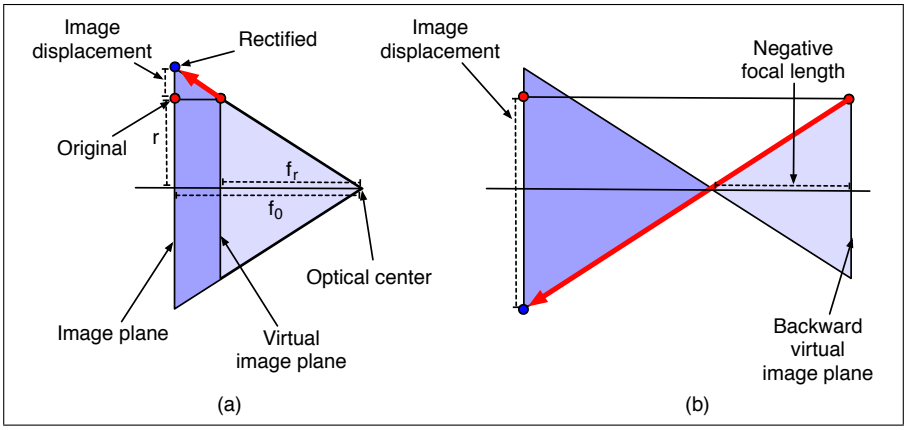


Fig. 1. Distortion circles are associated with cones in space. Theoretically, any point of the image can be projected into a single plane. **a)** Pixel from a cone looking forward, **b)** one from a cone looking backward.

are collinear in space. This is thus akin to the classical plumbline calibration technique [4, 6].

4.1. Calibration of Distortion Function

We obtain linear constraints on the distortion function as follows. Consider the images of three collinear points, $\mathbf{p}_i = (x_i, y_i)^\top$. For now, let us assume that the distortion center is known and that the image coordinate system is centered in this point. Hence, $r_i = ||(x_i, y_i)||$ is the distance of a point from the distortion center. Provided that these points should be collinear once rectified, we know that:

$$\begin{vmatrix} x_0 & x_1 & x_2 \\ y_0 & y_1 & y_2 \\ f(\,r_0\,) & f(\,r_1\,) & f(\,r_2\,) \end{vmatrix} = 0 \tag{3}$$

which can be written explicitly as a linear constraint on the $f(r_i)$'s:

$$f(\,r_0\,)\begin{vmatrix} x_1 & x_2 \\ y_1 & y_2 \end{vmatrix} + f(\,r_1\,)\begin{vmatrix} x_2 & x_0 \\ y_2 & y_0 \end{vmatrix} + f(\,r_2\,)\begin{vmatrix} x_0 & x_1 \\ y_0 & y_1 \end{vmatrix} = 0. \tag{4}$$

If f is of the form (1) or (2), then this equation gives a linear constraint on its parameters λ_i respectively f_r.

Constraints can be accumulated from all possible triplets of points that are projections of collinear points in space. We thus obtain a linear equation system of the form $\mathsf{A}\,\mathbf{x} = \mathbf{0}$, where $\mathbf{x}$ contains the parameters of f (the λ_i's or the f_r's). Note that constraints from triplets where two or all three image points lie close to one another are not very useful and hence can be neglected in order to reduce the number of equations. Solving this system to least squares yields parameters that maximize the collinearity of the rectified points[4]. Note that the equation

[4] However, it is not optimal in terms of geometric distance.

6

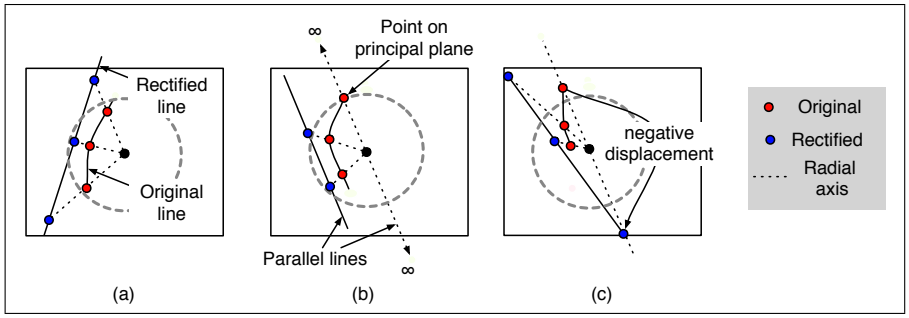


Fig. 2. Situations where three points are rectified into collinear positions. **a)** Three points corresponding to forward cones. **b)** One point located on principal distortion circle, i.e. scene point on principal plane. **c)** Two points on forward cones and one on a backward cone.

system is homogeneous, i.e. the distortion parameters are only estimated up to scale. This is natural, as explained below; a unique solution can be guaranteed by setting $\lambda_0 = 1$ as is usually done for the division model, or by setting one f_r to a fixed value.

4.2. Calibration of Distortion Center

So far, we have assumed a known distortion center. In this section, we show how it can be estimated as well, in addition to the actual distortion function. A first idea is to sample likely positions of the distortion center, e.g. consider a regular grid of points in a circular region in the image center, and compute the distortion function for each of them using the above method. We then keep the point yielding the smallest residual of the linear equation system as the estimated distortion center. This approach is simple and not very elegant, but is fully justified and works well in practice. Its downside is that the computation time is proportional to the number of sampled points.

Therefore, we investigate a local optimization procedure, as opposed to the above brute force one. Let (c_x, c_y) be the unknown distortion center. Equation (3) now becomes:

$$\begin{vmatrix} x_0 - c_x & x_1 - c_x & x_2 - c_x \\ y_0 - c_y & y_1 - c_y & y_2 - c_y \\ f(\left\| \begin{bmatrix} x_0 - c_x \\ y_0 - c_y \end{bmatrix} \right\|) & f(\left\| \begin{bmatrix} x_1 - c_x \\ y_1 - c_y \end{bmatrix} \right\|) & f(\left\| \begin{bmatrix} x_2 - c_x \\ y_2 - c_y \end{bmatrix} \right\|) \end{vmatrix} = 0. \qquad (5)$$

First, this constraint cannot be used directly for the discretized version of the distortion function. Second, if we use the polynomial model, the constraint is highly non-linear in the coordinates of the distortion center.

We thus consider an approximation of (5): we assume that a current estimate of the distortion center is not too far away from the true position ($||(c_x, c_y)||$ is

small), so that f can be approximated with $(c_x, c_y) = \mathbf{0}$ and

$$f(\left\| \begin{bmatrix} x \\ y \end{bmatrix} \right\|) \approx f(\left\| \begin{bmatrix} x - c_x \\ y - c_y \end{bmatrix} \right\|).$$

Equation (5) thus simplifies to:

$$\begin{vmatrix} x_0 - c_x & x_1 - c_x & x_2 - c_x \\ y_0 - c_y & y_1 - c_y & y_2 - c_y \\ f(\left\| \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \right\|) & f(\left\| \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} \right\|) & f(\left\| \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} \right\|) \end{vmatrix} = 0 \tag{6}$$

which is linear in c_x and c_y. Once again, combining many constraints leads to an over-determined linear equation system. The recovered distortion center may not be optimal because the points are expressed relative to the approximate center and because of the simplification of (5). Hoping that the previous assumptions are applicable, this new center should nevertheless improve our rectification. This estimation is used in a local optimization scheme of alternation type:

0. Initialize the distortion center with e.g. the center of the image.
1. Fix the distortion center and compute the distortion function (§4.1).
2. Fix the distortion function and update the distortion center (§4.2).
3. Go to step 1, unless convergence is observed.

Instead of using the least-squares cost function based on the algebraic distance (3), we also consider a more geometric cost function to judge convergence in step 3. Consider a set of image points belonging to a line image. From the current values of distortion center and function, we compute their projection rays and fit a plane as follows: determine the plane that contains the optical center and that minimizes the sum of (squared) angles with projection rays. The residual squared angles, summed over all line images, give the alternative cost function.

4.3. Discussion

The estimation of distortion center and function is based on an algebraic distance expressing collinearity of rectified image points. Better would be of course to use a geometric distance in the original images; this is possible but rather involved and is left for future work.

We briefly describe what the calibration of the distortion function amounts to, in terms of full metric calibration. First, recall that the distortion function can be computed up to scale only from our input (see §4.1). This is natural: if we have a distortion function that satisfies all collinearity constraints, then multiplying it by a scale factor results in a distortion function that satisfies them as well. This ambiguity means that once the distortion function is computed (up to scale) and the image rectified, the camera can be considered as equivalent to a pinhole camera with unknown focal length, with the difference that the field of view is potentially larger than 180°. Any existing focal length calibration or self-calibration algorithm designed for pinhole cameras can be applied to obtain

8

a full metric calibration. A direct application of such algorithms can probably use only features that lie inside the principal distortion circle, but it should be possible to adapt them so as to use even fields of view larger than 180°. At this step, the second assumption of §1 can also be relaxed if desired: a full pinhole model, i.e. not only focal length, can in principle be estimated from rectified images.

5. Self-Calibration

We now develop a plane-based self-calibration approach that is based on the plumbline technique of the previous section. Consider that the camera acquires two images of a textured plane with otherwise unknown structure. We suppose that we can match the two images densely; the matching does not actually need to be perfectly dense, but assuming it simplifies the following explanations. This is discussed below in more details.

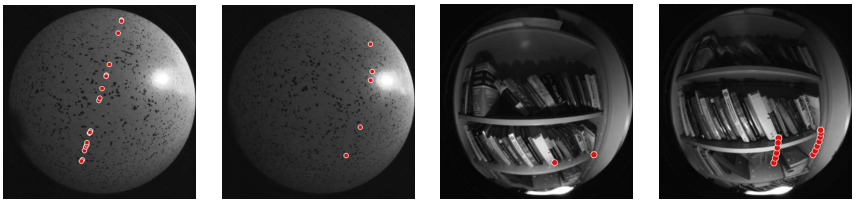
We now describe how dense matches between two images of a planar scene allow the generation of line images and hence to apply the plumbline technique. Consider any radial line (line going through the distortion center) in the first image; the projection rays associated with the points on that line are necessarily coplanar according to our camera model. Therefore, the scene points that are observed along that radial line must be collinear: they lie on the intersection of the plane of projection rays, with the scene plane. Due to the dense matching, we know the projections of these collinear scene points in the second image. By considering dense matches of points along n radial lines in one image, we thus obtain n line images in the other image, and vice versa. In addition, these line images usually extend across a large part of the image, bringing about strong constraints.

We now simply stack all plumbline constraints (4) for all pairs of images, and solve for the distortion parameters as in §4. Here, we have assumed the knowledge of the distortion center (in order to define radial lines); the distortion center can of course also be estimated, using e.g. the exhaustive approach of §4.2. Moreover, the input, once rectified, can be given to a classical plane-based self-calibration algorithm to obtain a full metric calibration, using e.g. [21].

Dense Matching. Dense matching can be achieved rather straightforwardly. If the camera acquires a continuous image sequence, most existing optical flow algorithms can be applied for successive frames and their results propagated in order to obtain a dense matching between two images with a substantial motion between them. In addition, the fact that a planar scene is observed eliminates the occlusion problem. If the scene is not sufficiently textured, but only allows to extract and track sparse interest points, then we proceed as follows. We extract dominant lines in each image using a Hough transform of the extracted interest points, and only keep the lines passing near the current distortion center estimate. These are almost radial lines. An example is shown in fig. 3a,b. The rest of the self-calibration is as above.

Constrained Camera Motions. Another way to obtain line images without the need for linear features in the scene is to acquire images under constrained

camera motions. A first possibility is to carry out pure rotations about the optical center, as suggested also by [19]. The scene can then be assimilated to a plane, and the above self-calibration method can be directly applied. A second possibility is to perform pure translations (with e.g. a tripod) and to track image points across several images. In this case, any point track constitutes a line image (an example is shown in fig. 3c,d).



(a) (b) (c) (d)

Fig. 3. (a)+(b) Two images of a planar scene. a) shows interest points lying on a radial line in the first image and b) corresponding points in the second image. **(c)+(d)** Two images of a general scene, taken with pure translation. c) shows two interest points in the first image and d) their paths, accumulated in the last image.

6. Results and Analysis

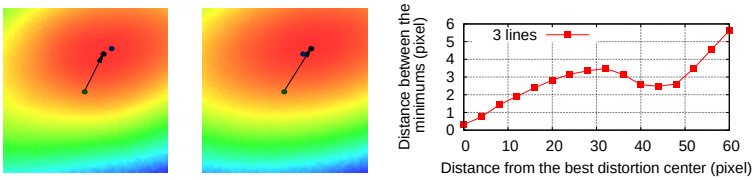
We tested our algorithm with data acquired from real and simulated cameras. An 8.0 mm lens, a 3.5mm fisheye lens and a para-catadioptric camera were used. We also simulated ten cameras featuring distortions from small to very large.

6.1. Convergence Analysis of the Distortion Center Detection

Two aspects of convergence of the plumbline method were evaluated. First, evaluating if the minimization of the constraints given by (6) instead of (5) leads to similar results. This is not critical though, as the path of the optimizer needs not be the same to ensure convergence. On the other hand, if the paths are similar, it suggests that the convergence pace is not penalized too much with the simplified cost function. We proceeded as follows. For samples of distortion center positions in a box around the initial position, we computed the two cost functions and found their minima (fig. 4a,b). We see that the functions' general shapes are almost identical, as well the positions of their respective minima. Another evaluation consists in initializing the distortion center randomly around the optimal one and finding the minima of the two cost functions. Figure 4c shows the average distance between these minima, as a function of the distance of the given distortion center from the optimal one. It is generally small, suggesting that both cost functions may lead to similar optimization paths.

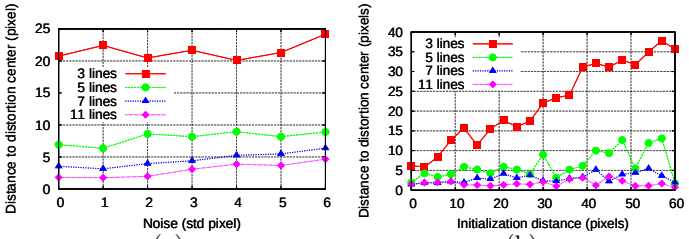
Secondly, the overall convergence was tested with simulated and real data. In the first case, three criteria were considered: the number of line images given

10



(a) (b) (c)

Fig. 4. Plots of cost functions and optimization paths associated with **(a)** eq. (5) and **(b)** eq. (6). **(c)** Distance between minima of these two cost functions, with respect to distance of current estimate of distortion center from optimal one. Data from the 3.5mm fisheye lens.



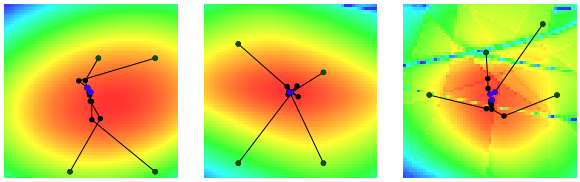
(a) (b)

Fig. 5. Precision of the recovered distortion center on simulated data w.r.t. **a)** noise and number of lines, **b)** number of lines and initialization distance.

as input, the amount of noise added to the data and the distance of the given initial distortion center from the true one. For each simulated camera, up to 11 line segments were generated randomly, Gaussian noise of standard deviation 0 to 6 pixels was added to image point coordinates and these were then quantized to pixel precision. For every camera, 50 initial values for the distortion center were randomly chosen in a circle of 60 pixels radius around the true position (for images of size 1000×1000) and given as input to the algorithm. This a realistic test considering that for our real cameras, we found that the estimated distortion center converged to around 30 pixels from the initial value (image center) in the worst case. The results in fig. 5 show that the number of lines has a much larger impact on the quality of the recovered distortion center than the noise and the initialization distance. This is especially true when the number of line is larger than 7.

6.2. Plumbline Calibration

We acquired images of lines with our real cameras, calibrated the distortion and then performed rectification. Once again, we tested the convergence and also the quality of the rectification by checking the collinearity of rectified line images. Convergence was never found to be an issue, especially for the two dioptric lenses (fig. 6). Even with a really bad initialization of the distortion center, resulting in a poor initial estimate of the distortion function, the algorithm converged



(a) (b) (c)

Fig. 6. Convergence examples of the algorithm for **a)** the 8.0 mm, **b)** the 3.5 mm fisheye, **c)** the para-catadioptric. The density plots show the value of the cost function explained at the end of §4.2, with f computed using distortion center positions (c_x, c_y) in a box of 60×60 pixels around the final distortion centers. In dark-green, different initializations of the algorithm; in black, the centers at each step of the algorithm; in purple, the final centers.

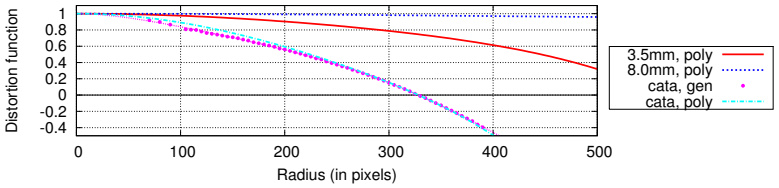


Fig. 7. Calibrated distortion functions for our real cameras. *poly* refers to (1) and *gen* to (2). For the 8.0 and 3.5mm, both representations lead to virtually identical results (details at table 1).

surprisingly fast (fig. 8). The distortion functions for our real cameras are shown in fig. 7 as well as rectified images in fig. 9 (images not used for the calibration). We compared our approach with the one presented in [6], run on the same data. Since that approach performs non-linear optimization, it can easily incorporate different distortion models. Results for different models are shown in table 1; we initialized the distortion centers with the one that was estimated with our approach and the distortion function as a constant.

Details are given in fig. 10 for the catadioptric cameras. We observe that a polynomial function did not give satisfying results. Using higher degrees (up to 10) and changing the distortion function did not give much better results. On the other hand, we see that a division function is very well suited to model the distortion in the image.

6.3. Self-Calibration from Real Sequences

Two sequences were tested. In the first one, points were tracked from a flat surface (our laboratory floor) with a hand-held camera. In the second case, a tripod was used and the camera was translated in constant direction. Overall, the results were satisfying although not as precise as with the direct plumbline technique using images of actual linear features. Results are summarized in table 2; values shown were computed like explained in table 1 and using images of

12

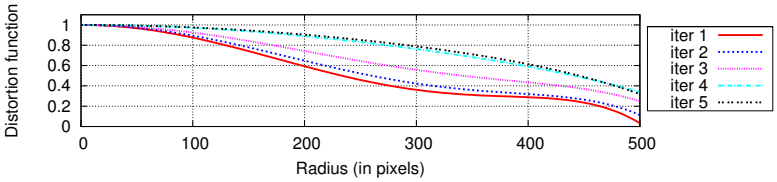
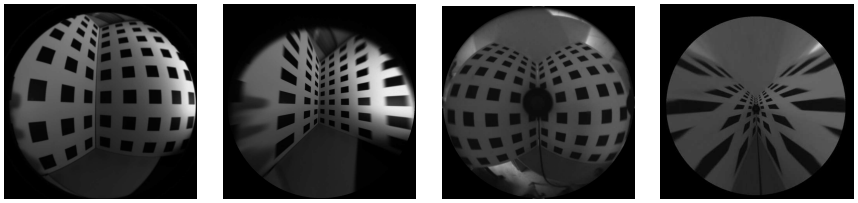


Fig. 8. The distortion function of the fisheye lens, at different iterations of the calibration algorithm for an initial center very far from the true position (200,400). The final estimate of (512,523) was found in only 5 iterations (image of size 1000×1000 pixels). Subsequent steps were only minor improvements.



(a) (b) (c) (d)

Fig. 9. Rectification examples. **a,b)** A 3.5mm fisheye original and rectified images. **c,d)** a catadioptric image. The radius of the principal distortion circle was estimated as 329 pixels, so circles of radius 0 to 320 pixels were rectified.

actual lines. The distortion center detection was also not as precise. The algorithm converged as usual, but not exactly to the best distortion center. In fact, it was much closer to the image center. This is explained by the fact that towards the image border, features are much more difficult to track: they are smaller and blurry. In this case, they are usually dropped by the tracking algorithm resulting in less data for large radiuses, where the distortion is the worst. Consequently, the distortion is a little bit under-evaluated and the distortion center less well constrained.

7. Conclusion

We presented flexible calibration methods for a general model for radial distortion, one plumbline type method and one for plane-based self-calibration. The methods were applied for simulated and real images of different cameras (fisheye and catadioptric). Results are satisfying, in terms of convergence basin and speed, precision as well as accuracy.

The most closely related works are [19, 20]. There, elegant though rather more involved procedures are proposed. These start with an even more general camera model than here, that does not enforce radial symmetry; only after computing and exploiting multi-view relations for that model, radial symmetry is enforced in order to compute distortion parameters. Our methods are much simpler to implement, use radial symmetry directly and can work with fewer images (two for plane-based self-calibration). Future work will mainly concern improving the

Table 1. Results using our models and algorithm (first two rows) and other models and the non-linear algorithm of [6]. Shown values refer to residual distances for fitting lines to rectified points (average and worst case). The rectified images were scaled to have the same size as the original. For the catadioptric camera, our approach used all the points, whereas the others used only the points corresponding to forward viewing cones (they failed otherwise). "—" means the algorithm did not converge without careful initialization or gave very bad results.

Models and rectifying equations	8mm		3.5mm		catadioptric	
Discrete model of (2)	0.16	1.03	**0.35**	**3.7**	**0.51**	**7.6**
Model of (1) with $d = 6$	0.16	1.12	0.35	5.5	**0.47**	**6.3**
6^{th} order polynomial $\mathbf{p}(1 + \lambda_1\|\|\mathbf{p}\|\| + ... + \lambda_6\|\|\mathbf{p}\|\|^6)$	0.16	1.08	0.42	7.0	1.5	14.4
6^{th} order division (non-linear)	0.16	1.08	0.36	5.6	—	—
FOV-model [6]: $\mathbf{p}\frac{\tan(\omega\|\|\mathbf{p}\|\|)}{2\tan\left(\frac{\omega}{2}\right)\|\|\mathbf{p}\|\|}$	0.23	4.86	0.54	7.9	—	—
FOV-model + 2^{nd} order polynomial	0.16	1.06	0.37	6.1	—	—

Table 2. Results for the 3.5mm fisheye with data from real sequences (fig. 3).

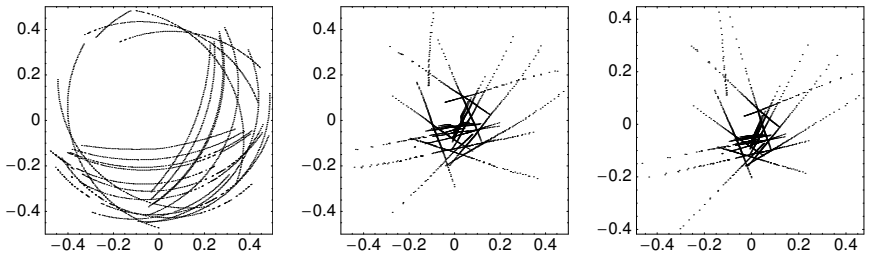
Models	plane		translation	
Discrete model of (2)	0.68	8.05	0.55	7.0
Model of (1) with $d = 6$	0.58	9.7	0.85	14.6

tracking for the self-calibration method and investigating the optimization of reprojection based cost functions.

References

1. S. Baker, S.K. Nayar. A Theory of Single-Viewpoint Catadioptric Image Formation. IJCV, 35(2), 1–22, 1999.
2. J.P. Barreto, K. Daniilidis. Unifying image plane liftings for central catadioptric and dioptric cameras. OMNIVIS 2004.
3. M. Born and E. Wolf. Principles of Optics, Pergamon Press, 1965.
4. D.C. Brown. Close-Range Camera Calibration. Photogrammetric Engineering, 37(8), 855-866, 1971.
5. D. Claus, A.W. Fitzgibbon. Rational Function Model for Fish-eye Lens Distortion CVPR 2005.
6. F. Devernay, O. Faugeras. Straight lines have to be straight: Automatic calibration and removal of distortion from scenes of structured environments. MVA 2001.
7. A.W. Fitzgibbon. Simultaneous linear estimation of multiple view geometry and lens distortion. CVPR 2001.
8. M.M. Fleck. Perspective Projection: The Wrong Imaging Model. TR 95–01, University of Iowa, 1995.
9. C. Geyer, K. Daniilidis. Catadioptric Camera Calibration. ICCV 1999.

14



(a) (b) (c)

Fig. 10. Line images used as input of the algorithms and rectification results for the catadioptric camera (with principal distortion circle found at 329 pixels). **a)** Input (only shown for radius smaller than 315 pixels), **b)** Rectification with a traditional model of degree 6 (model as in third row of table 1), **c)** with the polynomial distortion function (1) and $d = 6$ (the discrete model of (2) gave almost identical results).

10. M.D. Grossberg, S.K. Nayar. A general imaging model and a method for finding its parameters. ICCV 2001.
11. R. Hartley, A. Zisserman. Multiple View Geometry in Computer Vision Cambridge University Press 2000.
12. R. I. Hartley, S. B. Kang. Parameter-free Radial Distortion Correction with Centre of Distortion Estimation. ICCV 2005.
13. Intel Open Source Computer Vision Library. http://www.intel.com/research/mrl/research/opencv/
14. B.Micusik, T.Pajdla. Autocalibration & 3D Reconstruction with Non-central Catadioptric Cameras. CVPR 2004.
15. S. Shah, J.K. Aggarwal. Intrinsic Parameter Calibration Procedure for A (High-Distortion) Fish-Eye Lens Camera with Distortion Model and Accuracy Estimation. Pattern Recognition, 29(11), 1775-1788, 1996.
16. D.E. Stevenson, M.M. Fleck. Nonparametric correction of distortion. TR 95–07, University of Iowa, 1995.
17. P. Sturm, S. Ramalingam. A Generic Concept for Camera Calibration. ECCV 2004.
18. R. Swaminathan, M. Grossberg, S. Nayar. Caustics of catadioptric cameras. ICCV 2001.
19. S. Thirthala, M. Pollefeys. The Radial Trifocal Tensor. A tool for calibrating the radial distortion of wide-angle cameras. CVPR 2005.
20. S. Thirthala, M. Pollefeys. Multi-View Geometry of 1D Radial Cameras and its Application to Omnidirectional Camera Calibration. to appear, ICCV 2005.
21. B. Triggs. Autocalibration from Planar Scenes. ECCV 1998.
22. X. Ying, Z. Hu Can We Consider Central Catadioptric Cameras and Fisheye Cameras within a Unified Imaging Model. ECCV 2004.
23. Z. Zhang. A Flexible New Technique for Camera Calibration. PAMI, 22(11), 1330-1334, 2000.

Chapter 8

Structure from Motion

Paper 17 [34]: P. Sturm, S. Ramalingam, and S.K. Lodha. On calibration, structure-from-motion and multi-view geometry for general camera models. In R. Reulke and U. Knauer, editors, *Proceedings of the 2nd ISPRS Panoramic Photogrammetry Workshop, Berlin, Germany*. International Society for Photogrammetry and Remote Sensing, February 2005. Published in the Int. Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. XXXVI-5/W8.

Paper 18 [35]: P. Sturm, S. Ramalingam, and S.K. Lodha. On calibration, structure from motion and multi-view geometry for generic camera models. In K. Daniilidis, R. Klette, and A. Leonardis, editors, *Imaging Beyound the Pinhole Camera*. Kluwer Academic Publishers, 2006.

ON CALIBRATION, STRUCTURE-FROM-MOTION AND MULTI-VIEW GEOMETRY FOR PANORAMIC CAMERA MODELS

Peter Sturm[a], Srikumar Ramalingam[b], Suresh K. Lodha[b]

[a] INRIA Rhône-Alpes, 655 Avenue de l'Europe, 38330 Montbonnot, France – Peter.Sturm@inrialpes.fr
[b] Dept. of Computer Science, University of California, Santa Cruz, USA – {srikumar,lodha}@cse.ucsc.edu

Commission V, WG V/1 and 5

KEY WORDS: Panoramic camera, Non-central camera, Multi-view geometry, Calibration, 3D Reconstruction, Motion estimation

ABSTRACT

We consider calibration and structure-from-motion tasks for a previously introduced, highly general imaging model, where cameras are modeled as possibly unconstrained sets of projection rays. This allows to describe most existing camera types (at least for those operating in the visible domain), including pinhole cameras, sensors with radial or more general distortions, and especially panoramic cameras (central or non-central). Generic algorithms for calibration and structure-from-motion tasks (absolute and relative orientation, 3D point triangulation) are outlined. The foundation for a multi-view geometry of non-central cameras is given, leading to the formulation of multi-view matching tensors, analogous to the essential matrix, trifocal and quadrifocal tensors of perspective cameras. Besides this, we also introduce a natural hierarchy of camera models: the most general model has unconstrained projection rays whereas the most constrained model dealt with here is the central one, where all rays pass through a single point.

1 INTRODUCTION

Many different types of cameras including pinhole, stereo, catadioptric, omnidirectional and non-central cameras have been used in computer vision and photogrammetry. Most existing camera models are parametric (i.e. defined by a few intrinsic parameters) and address imaging systems with a single effective viewpoint (all rays pass through one point). In addition, existing calibration or structure-from-motion procedures are often taylor-made for specific camera models, see examples e.g. in (Barreto & Araujo, 2003; Gruen & Huang, 2001; Hartley & Zisserman, 2000; Geyer & Daniilidis, 2002).

The aim of this work is to relax these constraints: we want to propose and develop calibration and structure-from-motion methods that should work for any type of camera model, and especially also for cameras without a single effective viewpoint. To do so, we first renounce on parametric models, and adopt the following very general model: a camera acquires images consisting of pixels; each pixel captures light that travels along a ray in 3D. The camera is fully described by (Grossberg & Nayar, 2001):

- the coordinates of these rays (in a local coordinate frame).
- the mapping between rays and pixels; this is basically a simple indexing.

This is of course an idealistic model; other aspects, e.g. photometry and point-spread function are described in (Grossberg & Nayar, 2001). This general imaging model allows to describe virtually any camera that captures light rays travelling along straight lines. Examples are (cf. figure 1):

- a camera with any type of optical distortion, e.g. radial or tangential.
- a camera looking at a reflective surface, e.g. as often used in surveillance, a camera looking at a spherical or otherwise curved mirror (Hicks & Bajcsy, 2000). Such systems, as opposed to central catadioptric devices using parabolic or hyperbolic mirrors (Baker & Nayar, 1999; Geyer & Daniilidis, 2000), do not usually have a single effective viewpoint.
- multi-camera stereo systems: put together the pixels of all image planes; they "catch" light rays that do not travel along lines that all pass through a single point. Nevertheless, in the above general camera model, a stereo system (with rigidly linked cameras) is considered as a **single** camera.

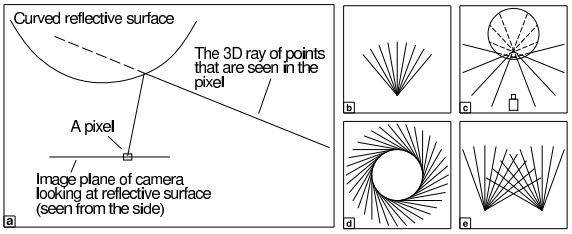


Figure 1: Examples of imaging systems. (a) Catadioptric system. Note that camera rays do not pass through their associated pixels. (b) Central camera (e.g. perspective, with or without radial distortion). (c) Camera looking at reflective sphere. This is a non-central device (camera rays are not intersecting in a single point). (d) Omnivergent imaging system (Peleg 2001; Shum 1999). (e) Stereo system (non-central) consisting of two central cameras.

- other acquisition systems, many of them being non-central, see e.g. (Bakstein, 2001; Bakstein & Pajdla, 2001; Neuman et al., 2003; Pajdla, 2002b; Peleg et al., 2001; Shum et al., 1999; Swaminathan et al., 2003; Yu & McMillan, 2004), insect eyes, etc.

In this article, we first review some recent work on calibration and structure-from-motion for this general camera model. Concretely, we outline basics for calibration, pose and motion estimation, as well as 3D point triangulation. We then describe the foundations for a mult-view geometry of the general, non-central camera model, leading to the formulation of multi-view matching tensors, analogous to the fundamental matrices, trifocal and quadrifocal tensors of perspective cameras. Besides this, we also introduce a natural hierarchy of camera models: the most general model has unconstrained projection rays whereas the most constrained model dealt with here is the central model, where all rays pass through a single point. An intermediate model is what we term *axial cameras*: cameras for which there exists a 3D line that cuts all projection rays. This encompasses for example x-slit projections, linear pushbroom cameras and some non-central catadioptric systems. Hints will be given how to adopt the multi-view geometry proposed for the general imaging model, to such axial cameras.

The paper is organized as follows. A hierarchy of camera models is proposed in section 2. Sections 3 to 5 deal with calibration, pose estimation, motion estimation, as well as 3D point triangulation. The multi-view geometry for the general camera model is given in section 6. A few experimental results are shown in section 7.

2 CAMERA MODELS

A **non-central camera** may have completely unconstrained projection rays, whereas for a **central camera**, there exists a point – the **optical center** – that lies on all projection rays. An intermediate case is what we call **axial cameras**, where there exists a line that cuts all projection rays – the **camera axis** (not to be confounded with optical axis). Examples of cameras falling into this class are:

- x-slit cameras (Pajdla, 2002a; Zomet et al., 2003) (also called two-slit or crossed-slits cameras), and their special case of linear pushbroom cameras (Hartley & Gupta, 1994). Note that these form a sub-class of axial cameras, see below.
- stereo systems consisting of 2 central cameras or 3 or more central cameras with collinear optical centers.
- non-central catadioptric cameras of the following construction: the mirror is any surface of revolution and the optical center of the central camera (can be any central camera, i.e. not necessarily a pinhole) looking at the mirror lies on its axis of revolution. It is easy to verify that in this case, all projection rays cut the mirror's axis of revolution, i.e. the camera is an axial camera, with the mirror's axis of revolution as camera axis. Note that catadioptric cameras with a spherical mirror and a central camera looking at it, are always non-central, and are actually always axial cameras.

These three classes of camera models may also be defined as: existence of a linear space of d dimensions that has an intersection with all projection rays. In this sense, $d = 0$ defines central cameras, $d = 1$ axial cameras and $d = 2$ general non-central cameras.

Intermediate classes do exist. X-slit cameras are a special case of axial cameras: there actually exist 2 lines in space that both cut all projection rays. Similarly, central 1D cameras (cameras with a single row of pixels) can be defined by a point and a line in 3D. Camera models, some of which do not have much practical importance, are summarized in table 1. A similar way of defining camera types was suggested in (Pajdla, 2002a).

It is worthwhile to consider different classes due to the following observation: the usual calibration and motion estimation algorithms proceed by first estimating a matrix or tensor by solving linear equation systems (e.g. the calibration tensors in (Sturm & Ramalingam, 2004) or the essential matrix (Pless, 2003)). Then, the parameters that are searched for (usually, motion parameters), are extracted from these. However, when estimating for example the 6×6 essential matrix of *non-central* cameras based on image correspondences obtained from *central* or *axial* cameras, then the associated linear equation system does not give a unique solution. Consequently, the algorithms for extracting the actual motion parameters, can not be applied without modification.

Points/lines cutting rays	Description
None	Non-central camera
1 point	Central camera
2 points	Camera with a single projection ray
1 line	Axial camera
1 point, 1 line	Central 1D camera
2 skew lines	X-slit camera
2 coplanar lines	Union of a non-central 1D camera and a central camera
3 coplanar lines without a common point	Non-central 1D camera

Table 1: Camera models, defined by 3D points and lines that have an intersection with all projection rays of a camera.

3 CALIBRATION

3.1 Basic Approach

We briefly review a generic calibration approach developed in (Sturm & Ramalingam, 2004), an extension of (Champleboux et al., 1992; Gremban et al, 1988; Grossberg & Nayar, 2001), to calibrate different camera systems. As mentioned, calibration consists in determining, for every pixel, the 3D projection ray associated with it. In (Grossberg & Nayar, 2001), this is done as follows: two images of a calibration object with known structure are taken. We suppose that for every pixel, we can determine the point on the calibration object, that is seen by that pixel[1]. For each pixel in the image, we thus obtain two 3D points. Their coordinates are usually only known in a coordinate frame attached to the calibration object; however, if one knows the motion between the two object positions, one can align the coordinate frames. Then, every pixel's projection ray can be computed by simply joining the two observed 3D points.

In (Sturm & Ramalingam, 2004), we propose a more general approach, that does not require knowledge of the calibration object's displacement. In that case, three images need to be taken at least. The fact that all 3D points observed by a pixel in different views, are on a line in 3D, gives a constraint that allows to recover both the motion and the camera's calibration. The constraint is formulated via a set of trifocal tensors, that can be estimated linearly, and from which motion, and then calibration, can be extracted. In (Sturm & Ramalingam, 2004), this approach is first formulated for the use of 3D calibration objects, and for the general imaging model, i.e. for non-central cameras. We also propose variants of the approach, that may be important in practice: first, due to the usefulness of planar calibration patterns, we specialized the approach appropriately. Second, we propose a variant that works specifically for central cameras (pinhole, central catadioptric, or any other central camera). More details are given in (Sturm & Ramalingam, 2003).

This basic approach only handles the minimum number of images (two respectively three, for central respectively non-central cameras). Also, it only allows to calibrate the pixels that are matched to the calibration object in all images. Especially for panoramic cameras, complete calibration with this approach is thus very hard (unless an "omnidirectional" calibration object is available). Recently, we have thus developed an approach that deals with these drawbacks; it handles any number of images and also allows to calibrate image regions that are not covered by the calibration object in all images. This approach is described in the next paragraph.

3.2 General Approach

We propose two ideas to overcome the above mentioned limitations of our basic calibration approach. First, we have recently developed a method along the lines of (Sturm & Ramalingam, 2004) that can use more than the minimum number of images. This method can not be described in full detail here; it will be given in a future publication. This method nevertheless has the drawback of only allowing to calibrate image regions that are covered by the calibration object in all images used.

Our second idea is relatively straightforward. We first perform

[1] This can be achieved for example by using a flat screen as calibration "grid" and taking images of several black & white patterns that together uniquely encode the position of pixels on the screen.

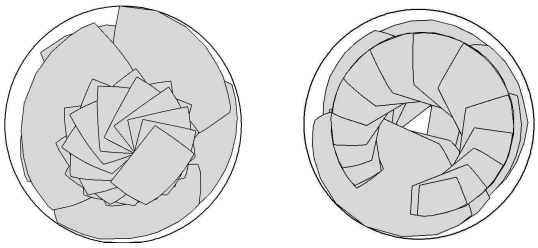


Figure 2: Examples of image regions corresponding to different images of calibration objects. Left: 23 images of calibration objects with a fisheye camera. Right: 24 images with a spherical catadioptric camera.

an initial calibration using our basic approach. This only allows to calibrate an image region that is covered by the calibration object in all images used. We then extend the calibration to the rest of the image, as follows. For each image in which the calibration object covers a sufficiently large already calibrated region, we can compute the object's pose relative to the camera (see section 4.1). Then, for each as yet uncalibrated pixel, we check if it is matched to the calibration object in sufficiently many images (one for central cameras, two for non-central ones); if so, we can compute the coordinates of its projection ray. For a non-central camera, we simply fit a straight line to the matching 3D points on the calibration object for different positions/images. As for the central model, we compute a straight line that is constrained to pass through the optical center.

These two procedures – computation of pose and projection rays – are repeated in alternation, until all available images have been used. Figure 2 gives examples of image regions covered by calibration objects in different images, for panoramic cameras that have been calibrated using our approach.

We also have developed a bundle adjustment that can be used between iterations, or only at the end of the above process, to refine calibration and pose. Our bundle adjustment minimizes *ray–point* distance, i.e. the distance in 3D, between projection rays and matching points on calibration objects. This is not the optimal measure, but reprojection-based bundle adjustment is not trivial to formulate for the generic imaging model (some ideas on this are given in (Ramalingam et al., 2004)). The minimization is done for the optical center position (only for central cameras), the pose of calibration objects, and of course the coordinates of projection rays. The ray–point distance is computed as

$$E = \sum_{i=1}^{r} \sum_{j=1}^{n} \|\mathbf{C}_i + \lambda_{ij}\mathbf{D}_i - \mathsf{R}_j\mathbf{P}_{ij} - \mathbf{t}_j\|^2$$

with:

- n is the number of calibration objects and r the number of rays.
- $\mathbf{C}_i$ is a point on the ith ray (in the non-central case) or the optical center (in a central model).
- $\mathbf{D}_i$ is the direction of the ith ray.
- λ_{ij} parameterizes the point on the ith ray that should correspond to its intersection with the jth calibration object.
- $\mathbf{P}_{ij}$ is the point on the jth calibration object that is matched to the pixel associated with the ith ray.
- R_j and $\mathbf{t}_j$ represent the pose of the jth calibration object.

4 ORIENTATION

4.1 Pose Estimation

Pose estimation is the problem of computing the relative position and orientation between an object of *known* structure, and a calibrated camera. A literature review on algorithms for pinhole cameras is given in (Haralick et al., 1994). Here, we briefly show how the minimal case can be solved for general cameras. For pinhole cameras, pose can be estimated, up to a finite number of solutions, from 3 point correspondences (3D-2D) already. The same holds for general cameras. Consider 3 image points and the associated projection rays, computed using the calibration information. We parameterize generic points on the rays as follows: $\mathbf{A}_i + \lambda_i\mathbf{B}_i$.

We know the structure of the observed object, meaning that we know the mutual distances d_{ij} between the 3D points. We can thus write equations on the unknowns λ_i, that parameterize the object's pose:

$$\|\mathbf{A}_i + \lambda_i\mathbf{B}_i - \mathbf{A}_j - \lambda_j\mathbf{B}_j\|^2 = d_{ij}^2$$
$$\text{for } (i,j) = (1,2),(1,3),(2,3)$$

This gives a total of 3 equations that are quadratic in 3 unknowns. Many methods exist for solving this problem, e.g. symbolic computation packages such as MAPLE allow to compute a resultant polynomial of degree 8 in a single unknown, that can be numerically solved using any root finding method.

Like for pinhole cameras, there are up to 8 theoretical solutions. For pinhole cameras, at least 4 of them can be eliminated because they would correspond to points lying behind the camera (Haralick et al., 1994). As for general cameras, determining the maximum number of feasible solutions requires further investigation. In any case, a unique solution can be obtained using one or two additional points (Haralick et al., 1994). More details on pose estimation for non-central cameras are given in (Chen & Chang, 2004; Nistér, 2004).

4.2 Motion Estimation

We outline how ego-motion, or, more generally, relative position and orientation of two calibrated general cameras, can be estimated. This is done via a generalization of the classical motion estimation problem for pinhole cameras and its associated centerpiece, the essential matrix (Longuet-Higgins, 1981). We briefly summarize how the classical problem is usually solved (Hartley & Zisserman, 2000). Let R be the rotation matrix and $\mathbf{t}$ the translation vector describing the motion. The essential matrix is defined as $\mathsf{E} = -[\mathbf{t}]_\times\mathsf{R}$. It can be estimated using point correspondences $(\mathbf{x}_1, \mathbf{x}_2)$ across two views, using the epipolar constraint $\mathbf{x}_2^\mathsf{T}\mathsf{E}\mathbf{x}_1 = 0$. This can be done linearly using 8 correspondences or more. In the minimal case of 5 correspondences, an efficient non-linear minimal algorithm, which gives exactly the theoretical maximum of 10 feasible solutions, was only recently introduced (Nistér, 2003). Once the essential matrix is estimated, the motion parameters R and $\mathbf{t}$ can be extracted relatively straightforwardly (Nistér, 2003).

In the case of our general imaging model, motion estimation is performed similarly, using pixel correspondences $(\mathbf{x}_1, \mathbf{x}_2)$. Using the calibration information, the associated projection rays can be computed. Let them be represented by their Plücker coordinates (see section 6), i.e. 6-vectors $\mathbf{L}_1$ and $\mathbf{L}_2$. The epipolar constraint extends naturally to rays, and manifests itself by a 6×6 essential matrix (Pless, 2003):

$$\mathsf{E} = \begin{pmatrix} -[\mathbf{t}]_\times\mathsf{R} & \mathsf{R} \\ \mathsf{R} & 0 \end{pmatrix}$$

The epipolar constraint then writes: $\mathbf{L}_2^\mathsf{T}\mathsf{E}\mathbf{L}_1 = 0$ (Pless, 2003). Once E is estimated, motion can again be extracted straightforwardly (e.g., R can simply be read off E). Linear estimation of E requires 17 correspondences.

There is an important difference between motion estimation for central and non-central cameras: with central cameras, the translation component can only be recovered up to scale. Non-central

cameras however, allow to determine even the translation's scale. This is because a single calibrated non-central camera already carries scale information (via the distance between mutually skew projection rays). One consequence is that the theoretical minimum number of required correspondences is 6 instead of 5. It might be possible, though very involved, to derive a minimal 6-point method along the lines of (Nistér, 2003).

More details on motion estimation for non-central cameras and intermediate camera models, will be given in a forthcoming publication.

5 3D RECONSTRUCTION

We now describe an algorithm for 3D reconstruction from two or more calibrated images with known relative position. Let $\mathbf{C} = (X, Y, Z)^\mathsf{T}$ be a 3D point that is to be reconstructed, based on its projections in n images. Using calibration information, we can compute the n associated projection rays. Here, we represent the ith ray using a starting point $\mathbf{A}_i$ and the direction, represented by a unit vector $\mathbf{B}_i$. We apply the mid-point method (Hartley & Sturm, 1997; Pless, 2003), i.e. determine $\mathbf{C}$ that is closest in average to the n rays. Let us represent generic points on rays using position parameters λ_i, as in the previous section. Then, $\mathbf{C}$ is determined by minimizing the following expression over $\mathbf{C}^\mathsf{T} = (X, Y, Z)$ and the λ_i: $\sum_{i=1}^{n} \|\mathbf{A}_i + \lambda_i \mathbf{B}_i - \mathbf{C}\|^2$.

This is a linear least squares problem, which can be solved e.g. via the Pseudo-Inverse, leading to the following explicit equation (derivations omitted):

$$\begin{pmatrix} \mathbf{C} \\ \lambda_1 \\ \vdots \\ \lambda_n \end{pmatrix} = \mathsf{M}^{-1} \begin{pmatrix} \mathtt{I}_3 & \cdots & \mathtt{I}_3 \\ -\mathbf{B}_1^\mathsf{T} & & \\ & \ddots & \\ & & -\mathbf{B}_n^\mathsf{T} \end{pmatrix} \begin{pmatrix} \mathbf{A}_1 \\ \vdots \\ \mathbf{A}_n \end{pmatrix}$$

with

$$\mathsf{M} = \begin{pmatrix} n\mathtt{I}_3 & -\mathbf{B}_1 & \cdots & -\mathbf{B}_n \\ -\mathbf{B}_1^\mathsf{T} & 1 & & \\ \vdots & & \ddots & \\ -\mathbf{B}_n^\mathsf{T} & & & 1 \end{pmatrix}$$

where $\mathtt{I}_3$ is the identity matrix of size 3×3. Due to its sparse structure, the inversion of M can actually be performed in closed-form. Overall, the triangulation of a 3D point using n rays, can by carried out very efficiently, using only matrix multiplications and the inversion of a symmetric 3×3 matrix.

6 MULTI-VIEW GEOMETRY

We establish the foundations of a multi-view geometry for general (non-central) cameras. Its cornerstones are, as with perspective cameras, matching tensors. We show how to establish them, analogously to the perspective case.

Here, we only talk about the calibrated case; the uncalibrated case is nicely treated for perspective cameras, since calibrated and uncalibrated cameras are linked by projective transformations. For non-central cameras however, there is no such link: in the most general case, every pair (pixel, camera ray) may be completely independent of other pairs.

6.1 Reminder on Multi-View Geometry for Perspective Cameras

We briefly review how to derive multi-view matching relations for perspective cameras (Faugeras & Mourrain, 1995). Let P_i be projection matrices and $\mathbf{q}_i$ image points. A set of image points are matching, if there exists a 3D point $\mathbf{Q}$ and scale factors λ_i such that:

$$\lambda_i \mathbf{q}_i = \mathsf{P}_i \mathbf{Q}$$

This may be formulated as the following matrix equation:

$$\underbrace{\begin{pmatrix} \mathsf{P}_1 & \mathbf{q}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathsf{P}_2 & \mathbf{0} & \mathbf{q}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathsf{P}_n & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{q}_n \end{pmatrix}}_{\mathsf{M}} \begin{pmatrix} \mathbf{Q} \\ -\lambda_1 \\ -\lambda_2 \\ \vdots \\ -\lambda_n \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

The matrix M, of size $3n \times (4+n)$ has thus a null-vector, meaning that its rank is less than $4+n$. Hence, the determinants of all its submatrices of size $(4+n) \times (4+n)$ must vanish. These determinants are multi-linear expressions in terms of the coordinates of image points $\mathbf{q}_i$.

They have to be considered for every possible submatrix. Only submatrices with 2 or more rows per view, give rise to constraints linking all projection matrices. Hence, constraints can be obtained for up to n views with $2n \leq 4 + n$, meaning that only for up to 4 views, matching constraints linking all views can be obtained.

The constraints for n views take the form:

$$\sum_{i_1=1}^{3} \sum_{i_2=1}^{3} \cdots \sum_{i_n=1}^{3} q_{1,i_1} q_{2,i_2} \cdots q_{n,i_n} T_{i_1,i_2,\cdots,i_n} = 0 \qquad (1)$$

where the multi-view matching tensor T of dimension $3 \times \cdots \times 3$ depends on and partially encodes the cameras' projection matrices P_i. Note that as soon as cameras are calibrated, this theory applies to any central camera: for a camera with radial distortion for example, the above formulation holds for distortion-corrected image points.

6.2 Multi-View Geometry for Non-Central Cameras

Here, instead of projection matrices (depending on calibration and pose), we deal with pose matrices:

$$\mathsf{P}_i = \begin{pmatrix} \mathsf{R}_i & \mathbf{t}_i \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix}$$

These express the similarity transformations that map a point from some global reference frame, into the cameras' local coordinate frames (since no optical center and no camera axis exist, no assumptions about the local coordinate frames are made). As for image points, they are now replaced by camera rays. Let the ith ray be represented by two 3D points $\mathbf{A}_i$ and $\mathbf{B}_i$. Eventually, we will to obtain expressions in terms of the rays' Plücker coordinates. Plücker coordinates can be defined in various ways; the definition we use is as follows. The line can be represented by the skew-symmetric 4×4 so-called Plücker matrix

$$\mathsf{L} = \mathbf{A}\mathbf{B}^\mathsf{T} - \mathbf{B}\mathbf{A}^\mathsf{T}$$

Note that the Plücker matrix is independent (up to scale) of which pair of points on the line are chosen to represent it. An alternative representation for the line is its Plücker coordinate vector of length 6:

$$\mathbf{L} = \begin{pmatrix} A_4B_1 - A_1B_4 \\ A_4B_2 - A_2B_4 \\ A_4B_3 - A_3B_4 \\ A_3B_2 - A_2B_3 \\ A_1B_3 - A_3B_1 \\ A_2B_1 - A_1B_2 \end{pmatrix} \qquad (2)$$

Our goal is to obtain matching tensors T and matching constraints of the form (1), with the difference that tensors will have size $6 \times \cdots \times 6$ and act on Plücker line coordinates:

$$\sum_{i_1=1}^{6} \sum_{i_2=1}^{6} \cdots \sum_{i_n=1}^{6} L_{1,i_1} L_{2,i_2} \cdots L_{n,i_n} T_{i_1,i_2,\cdots,i_n} = 0 \qquad (3)$$

In the following, we explain how to derive such matching constraints. Consider a set of n camera rays and let them be defined by two points $\mathbf{A}_i$ and $\mathbf{B}_i$ each; the choice of points to represent a ray is not important, since later we will fall back onto the ray's Plücker coordinates.

Now, a set of n camera rays are matching, if there exist a 3D point $\mathbf{Q}$ and scale factors λ_i and μ_i associated with each ray such that:

$$\lambda_i \mathbf{A}_i + \mu_i \mathbf{B}_i = \mathsf{P}_i \mathbf{Q}$$

i.e. if the point $\mathsf{P}_i\mathbf{Q}$ lies on the line spanned by $\mathbf{A}_i$ and $\mathbf{B}_i$. As for perspective cameras, we group these equations in matrix form:

$$\mathsf{M} \begin{pmatrix} \mathbf{Q} \\ -\lambda_1 \\ -\mu_1 \\ -\lambda_2 \\ -\mu_2 \\ \vdots \\ -\lambda_n \\ -\mu_n \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{pmatrix}$$

with:

$$\mathsf{M} = \begin{pmatrix} \mathsf{P}_1 & \mathbf{A}_1 & \mathbf{B}_1 & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ \mathsf{P}_2 & \mathbf{0} & \mathbf{0} & \mathbf{A}_2 & \mathbf{B}_2 & \cdots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathsf{P}_n & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{A}_n & \mathbf{B}_n \end{pmatrix}$$

As above, this equation shows that M must be rank-deficient. However, the situation is different here since the P_i are of size 4×4 now, and M of size $4n\times(4+2n)$. We thus have to consider submatrices of M of size $(4+2n)\times(4+2n)$. Furthermore, in the following we show that only submatrices with 3 rows or more per view, give rise to constraints on all pose matrices. Hence, $3n \leq 4+2n$, and again, $n \leq 4$, i.e. multi-view constraints are only obtained for up to 4 views.

Let us first see what happens for a submatrix of M where some view contributes only a single row. The two columns corresponding to its base points $\mathbf{A}$ and $\mathbf{B}$, are multiples of one another since they consist of zeroes only, besides a single non-zero coefficient, in the single row associated with the considered view. Hence, the determinant of the considered submatrix of M is always zero, and no constraint is available.

In the following, we exclude this case, i.e. we only consider submatrices of M where each view contributes at least 2 rows. Let N be such a matrix. Without loss of generality, we start to develop its determinant with the columns containing $\mathbf{A}_1$ and $\mathbf{B}_1$. The determinant is then given as a sum of terms of the form:

$$(A_{1,j}B_{1,k} - A_{1,k}B_{1,j}) \det \bar{\mathsf{N}}_{jk}$$

where $j,k \in \{1..4\}$, $j \neq k$, and $\bar{\mathsf{N}}_{jk}$ is obtained from N by dropping the columns containing $\mathbf{A}_1$ and $\mathbf{B}_1$ as well as the rows containing $A_{1,j}$ etc.

We observe several things:

- The term $(A_{1,j}B_{1,k} - A_{1,k}B_{1,j})$ is nothing else than one of the Plücker coordinates of the ray of camera 1 (cf. (2)). By continuing with the development of the determinant of $\bar{\mathsf{N}}_{jk}$, it becomes clear that the total determinant of N can be written in the form:

  $$\sum_{i_1=1}^{6}\sum_{i_2=1}^{6}\cdots\sum_{i_n=1}^{6} L_{1,i_1}L_{2,i_2}\cdots L_{n,i_n}T_{i_1,i_2,\cdots,i_n} = 0$$

  i.e. the coefficients of the $\mathbf{A}_i$ and $\mathbf{B}_i$ are "folded together" into the Plücker coordinates of camera rays and T is a matching tensor between the n cameras. Its coefficients depend exactly on the cameras' pose matrices.

	central		non-central	
# views	M	useful	M	useful
2	6×6	3-3	8×8	4-4
3	9×7	3-2-2	12×10	4-3-3
4	12×8	2-2-2-2	16×12	3-3-3-3

Table 2: Cases of multi-view matching constraints for central and non-central cameras. The columns entitled "useful" contain entries of the form $x-y-z$ etc. that correspond to sub-matrices of M that give rise to matching constraints linking *all* views: $x-y-z$ etc. refers to submatrices of M containing x rows from one camera, y from another etc.

- If camera 1 contributes only two rows to N, then the determinant of N becomes of the form:

  $$L_{1,x}\left(\sum_{i_2=1}^{6}\cdots\sum_{i_n=1}^{6} L_{2,i_2}\cdots L_{n,i_n}T_{i_2,\cdots,i_n}\right) = 0$$

  i.e. it only contains a single coordinate of the ray of camera 1, and the tensor T does not depend at all on the pose of that camera. Hence, to obtain constraints between all cameras, every camera has to contribute at least three rows to the considered submatrix.

We are now ready to establish the different cases that lead to useful multi-view constraints. As mentioned above, for more than 4 cameras, no constraints linking all of them are available: submatrices of size at least $3n\times3n$ would be needed, but M only has $4+2n$ columns. So, only for $n\leq4$, such submatrices exist.

Table 2 gives all useful cases, both for central and non-central cameras. These lead to two-view, three-view and four-view matching constraints, encoded by essential matrices, trifocal and quadrifocal tensors. Deriving their forms is now mainly a mechanical task.

6.3 Multi-View Geometry for Intermediate Camera Models

This multi-view geometry can be specialized to some of the intermediate camera models described in section 2. We have derived this for the axial and x-slit camera models. This will be reported elsewhere in detail.

7 EXPERIMENTAL RESULTS

We have calibrated a wide variety of cameras (both central and non-central) as shown in Table 3. Results are first discussed for several "slightly non-central" cameras and for a multi-camera system. We then report results for structure-from-motion algorithms, applied to setups combining cameras of different types (pinhole and panoramic).

Slightly non-central cameras: central vs. non-central models. For three cameras (a fisheye, a hyperbolic and a spherical catadioptric system, see sample images in Figure 3), we applied our calibration approach with both, a central and a non-central model assumption. Table 3 shows that the bundle adjustment's residual errors for central and non-central calibration, are very close to one another for the fisheye and hyperbolic catadioptric cameras. This suggests that for the cameras used in the experiments, the central model is appropriate. As for the spherical catadioptric camera, the non-central model has a significantly lower residual, which may suggest that a non-central model is better here.

To further investigate this issue we performed another evaluation. A calibration grid was put on a turntable, and images were acquired for different turntable positions. We are thus able to quantitatively evaluate the calibration, by measuring how close the recovered grid pose corresponds to a turntable sequence. Individual grid points move on a circle in 3D; we thus compute a least squares circle fit to the 3D positions given by the estimated grid

Camera	Images	Rays	Points	RMS
Pinhole (C)	3	217	651	0.04
Fisheye (C)	23	508	2314	0.12
(NC)	23	342	1712	0.10
Sphere (C)	24	380	1441	2.94
(NC)	24	447	1726	0.37
Hyperbolic (C)	24	293	1020	0.40
(NC)	24	190	821	0.34
Multi-Cam (NC)	3	1156	3468	0.69
Eye+Pinhole (C)	3	29	57	0.98

Table 3: Bundle adjustment statistics for different cameras. (C) and (NC) refer to central and non-central calibration respectively, and RMS is the root-mean-square residual error of the bundle adjustment (ray-point distances). It is given in percent, relative to the overall size of the scene (largest pairwise distance between points on calibration grids).

Camera	Grids	Central	Non-Central
Fisheye	14	0.64	0.49
Spherical	19	2.40	1.60
Hyperbolic	12	0.81	1.17

Table 4: RMS error for circle fits to grid points, for turntable sequences (see text).

pose. At the bottom of Figure 3, recovered grid poses are shown, as well as a circle fit to the positions of one grid point. Table 4 shows the RMS errors of circle fits (again, relative to scene size, and given in percent). We note that the non-central model provides a significantly better reconstruction than the central one for the spherical catadioptric camera, which thus confirms the above observation. For the fisheye, the non-central calibration also performs better, but not as significantly. As for the hyperbolic catadioptric camera, the central model gives a better reconstruction though. This can probably be explained as follows. Inspite potential imprecisions in the camera setup, the camera seems to be sufficiently close to a central one, so that the non-central model leads to overfitting. Consequently, although the bundle adjustment's residual is lower than for the central model (which always has to be the case), it gives "predictions" (here, pose or motion estimation) which are unreliable.

Calibration of a multi-camera system. A multi-camera network can be considered as a single generic imaging system. As shown in Figure 4 (left), we used a system of three (approximately pinhole) cameras to capture three images each of a calibration grid. We virtually concatenated the images from the individual cameras and computed all projection rays and the three grid poses in a single reference frame (see Figure 4 (right)), using the algorithm outlined in section 3.

In order to evaluate the calibration, we compared results with those obtained by plane-based calibration (Sturm & Maybank, 1999; Zhang, 2000), that used the knowledge that the three cameras are pinholes. In both, our multi-camera calibration, and plane-based calibration, the first grid was used to fix the global coordinate system. We can thus compare the estimated poses of the other two grids for the two methods. This is done for both, the rotational and translational parts of the pose. As for rotation, we measure the angle (in radians) of the relative rotation between the rotation matrices given by the two methods, see columns R_i in Table 5). As for translation, we measure the distance between the estimated 3D positions of the grids' centers of gravity (columns t_i in Table 5) expressed in percent, relative to the scene size. Here, plane-based calibration is done separately for each camera, leading to the three rows of Table 5.

From the non-central multi-camera calibration, we also estimate the positions of the three optical centers, by clustering the projection rays and computing least squares point fits to them. The column "Center" of Table 5 shows the distances between optical centers (expressed in percent and relative to the scene size) computed using this approach and plane-based calibration. The discrepancies are low, suggesting that the non-central calibration of a multi-camera setup is indeed feasible.

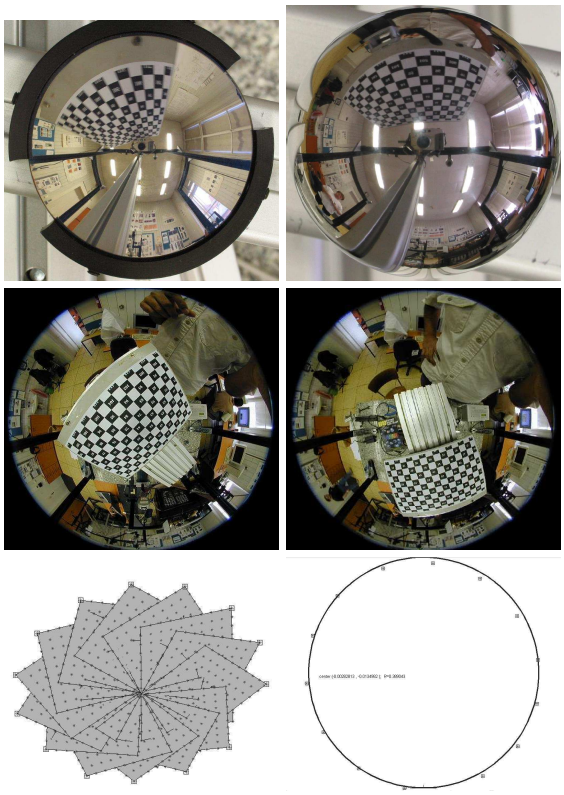


Figure 3: Top: sample images for hyperbolic and spherical catadioptric cameras. Middle: two images taken with a fisheye. Bottom: pose of calibration grids used to calibrate the fisheye (left) and a least squares circle fit to the estimated positions of one grid point (right).

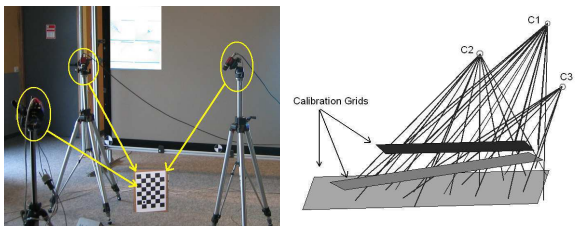


Figure 4: Multi-camera setup consisting of 3 cameras (left). Recovered projection rays and grid poses (right).

Camera	R_2	R_3	t_2	t_3	Center
1	0.0117	0.0359	0.56	3.04	2.78
2	0.0149	0.0085	0.44	2.80	2.17
3	0.0088	0.0249	0.53	2.59	1.16

Table 5: Evaluation of non-central multi-camera calibration relative to plane-based calibration. See text for more details.

Structure-from-motion with hybrid camera setups. We created hybrid camera setups by taking images with both, pinhole and fisheye cameras. Each camera was first calibrated individually using our approach of section 3. We then estimated the relative pose of two cameras (or, motion), using the approach

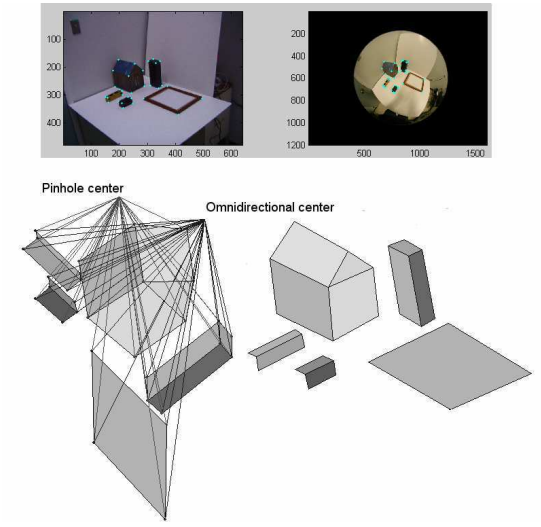


Figure 5: Combination of a pinhole and a fisheye camera. Top: input images and matching points. Bottom: estimated relative pose and 3D model.

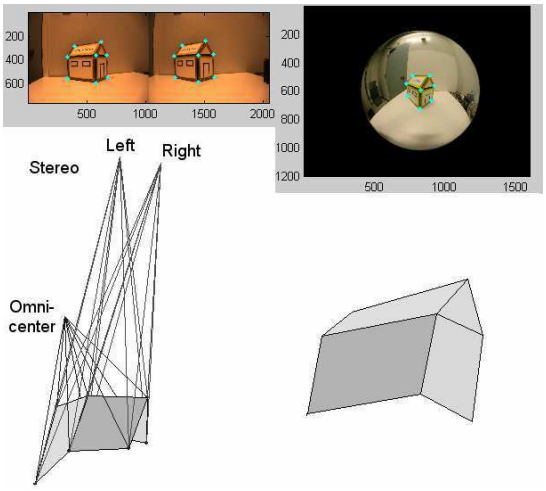


Figure 6: Combination of a stereo system and a fisheye camera. Top: input images and matching points. Bottom: estimated relative pose and 3D model.

outlined in section 4.2 and manually defined matches. Then, 3D structure was computed by reconstructing 3D points associated with the given matches.

Figure 5 shows this for a combination of a pinhole and a fisheye camera, and figure 6 for a combination of a stereo system and a fisheye. Here, the stereo system is handled as a single, non-central camera. Note that the same scene point usually appears more than once in the stereo camera. Therefore in the ray-intersection approach of section 5, we intersect three rays to find one 3D point here.

These results are preliminary: at the time we obtained them, we had not developed our full calibration approach of section 3.2, hence only the central region of the fisheye camera was calibrated and used. Nevertheless, the qualitatively correct results demonstrate that our generic structure-from-motion algorithms work, and actually are applicable to different cameras, or combinations thereof.

8 CONCLUSIONS

We have reviewed calibration and structure-from-motion tasks for the general non-central camera model. We also proposed a multi-view geometry for non-central cameras. A natural hierarchy of camera models has been introduced, grouping cameras into classes depending on, loosely speaking, the spatial distribution of their projection rays. We hope that the theoretical work presented here allows to define some common ground for recent efforts in characterizing the geometry of non-classical cameras.

The feasibility of our generic calibration and structure-from-motion approaches has been demonstrated on several examples. Of course, more investigations are required to evaluate the potential of these methods and the underlying models.

Among ongoing and future works, there is the adaptation of our calibration approach to axial and other camera models as well as first ideas on self-calibration for the general imaging model. We also continue our work on bundle adjustment for the general imaging model, cf. (Ramalingam et al. 2004), and the exploration of hybrid systems, combining cameras of different types (Sturm, 2002; Ramalingam et al. 2004).

Acknowledgements. This work was partially supported by the NSF grant ACI-0222900 and by the Multidisciplinary Research Initiative (MURI) grant by Army Research Office under contract DAA19-00-1-0352.

REFERENCES

References from Journals:

Baker, S. and Nayar, S.K., 1999. A Theory of Single-Viewpoint Catadioptric Image Formation. *IJCV*, 35(2), pp. 1-22.

Chen, C.-S. and Chang, W.-Y., 2004. On Pose Recovery for Generalized Visual Sensors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(7), pp. 848-861.

Geyer, C. and Daniilidis, K., 2002. Paracatadioptric camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5), pp. 687-695.

Haralick, R.M., Lee, C.N., Ottenberg, K. and Nolle, M., 1994. Review and analysis of solutions of the three point perspective pose estimation problem. *International Journal of Computer Vision*, 13(3), pp. 331-356.

Hartley, R.I. and Sturm, P., 1997. Triangulation. *Computer Vision and Image Understanding*, 68(2), pp. 146-157.

Longuet-Higgins, H.C., 1981. A Computer Program for Reconstructing a Scene from Two Projections. *Nature*, 293, pp. 133-135.

Pajdla, T., 2002b. Stereo with oblique cameras. *International Journal of Computer Vision*, 47(1), pp. 161-170.

Peleg, S., Ben-Ezra, M. and Pritch, Y., 2001. OmniStereo: Panoramic Stereo Imaging. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3), pp. 279-290.

Zhang, Z., 2000. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11), pp. 1330-1334.

Zomet, A., Feldman, D., Peleg, S. and Weinshall, D., 2003. Mosaicing New Views: The Crossed-Slit Projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(6), pp. 741-754.

References from Books:

Gruen, A. and Huang, T.S. (editors), 2001. *Calibration and Orientation of Cameras in Computer Vision*, Springer-Verlag.

Hartley, R.I. and Zisserman, A., 2000. *Multiple view geometry in computer vision*. Cambridge University Press.

References from Other Literature:

Non-central cameras for 3D reconstruction. Technical Report CTU-CMP-2001-21, Center for Machine Perception, Czech Technical University, Prague.

Bakstein, H. and Pajdla, T., 2001. An overview of non-central cameras. *Computer Vision Winter Workshop*, Ljubljana, Slovenia, pp. 223-233.

Barreto, J. and Araujo, H., 2003. Paracatadioptric Camera Calibration Using Lines. *International Conference on Computer Vision*, Nice France, pp. 1359-1365.

Champleboux, G., Lavallée, S., Sautot, P. and Cinquin, P., 1992. Accurate Calibration of Cameras and Range Imaging Sensors: the NPBS Method. *International Conference on Robotics and Automation*, Nice, France, pp. 1552-1558.

Faugeras, O. and Mourrain, B., 1995. On the Geometry and Algebra of the Point and Line Correspondences Between N Images. *International Conference on Computer Vision*, Cambridge, MA, USA, pp. 951-956.

Geyer, C. and Daniilidis, K., 2000. A unifying theory of central panoramic systems and practical applications. *European Conference on Computer Vision*, Dublin, Ireland, Vol. II, pp. 445-461.

Gremban, K.D., Thorpe, C.E. and Kanade, T., 1988. Geometric Camera Calibration using Systems of Linear Equations. *International Conference on Robotics and Automation*, Philadelphia, USA, pp. 562-567.

Grossberg, M.D. and Nayar, S.K., 2001. A general imaging model and a method for finding its parameters. *International Conference on Computer Vision*, Vancouver, Canada, Vol. 2, pp. 108-115.

Hartley, R.I. and Gupta, R., 1994. Linear Pushbroom Cameras. *European Conference on Computer Vision*, Stockholm, Sweden, pp. 555-566.

Hicks, R.A. and Bajcsy, R., 2000. Catadioptric Sensors that Approximate Wide-angle Perspective Projections. *Conference on Computer Vision and Pattern Recognition*, Hilton Head Island, USA, pp. 545-551.

Neumann, J., Fermüller, C. and Aloimonos, Y., 2003. Polydioptric Camera Design and 3D Motion Estimation. *Conference on Computer Vision and Pattern Recognition*, Madison, WI, USA, Vol. II, pp. 294-301.

Nistér, D., 2003. An Efficient Solution to the Five-Point Relative Pose Problem. *Conference on Computer Vision and Pattern Recognition*, Madison, WI, USA, Vol. II, pp. 195-202.

Nistér, D., 2004. A Minimal Solution to the Generalized 3-Point Pose Problem. *Conference on Computer Vision and Pattern Recognition*, Washington DC, USA, Vol. 1, pp. 560-567.

Pajdla, T., 2002a. Geometry of Two-Slit Camera. Technical Report CTU-CMP-2002-02, Center for Machine Perception, Czech Technical University, Prague.

Pless, R., 2003. Using Many Cameras as One. *Conference on Computer Vision and Pattern Recognition*, Madison, WI, USA, Vol. II, pp. 587-593.

Ramalingam, S., Lodha, S. and Sturm, P., 2004. A Generic Structure-from-Motion Algorithm for Cross-Camera Scenarios. *5th Workshop on Omnidirectional Vision, Camera Networks and Non-Classical Cameras*, Prague, Czech Republic, pp. 175-186.

Shum, H.-Y., Kalai, A. and Seitz, S.M., 1999. Omnivergent Stereo. *International Conference on Computer Vision*, Kerkyra, Greece, pp. 22-29.

Sturm, P., 2002. Mixing catadioptric and perspective cameras. *Workshop on Omnidirectional Vision*, Copenhagen, Denmark, pp. 60-67.

Sturm, P. and Maybank, S., 1999. On Plane-Based Camera Calibration: A General Algorithm, Singularities, Applications. *Conference on Computer Vision and Pattern Recognition*, Fort Collins, CO, USA, pp. 432-437.

Sturm, P. and Ramalingam, S., 2003. A Generic Calibration Concept – Theory and Algorithms. Research Report 5058, INRIA.

Sturm, P. and Ramalingam, S., 2004. A generic concept for camera calibration. *European Conference on Computer Vision*, Prague, Czech Republic, pp. 1-13.

Swaminathan, R., Grossberg, M.D. and Nayar, S.K., 2003. A perspective on distortions. *Conference on Computer Vision and Pattern Recognition*, Madison, WI, USA, Vol. II, pp. 594-601.

Yu, J. and McMillan, L., 2004. General Linear Cameras. *European Conference on Computer Vision*, Prague, Czech Republic, pp. 14-27.

ON CALIBRATION, STRUCTURE FROM MOTION AND MULTI-VIEW GEOMETRY FOR GENERIC CAMERA MODELS

Peter Sturm[1], Srikumar Ramalingam[2], and Suresh Lodha[2]
[1] *INRIA Rhône-Alpes,* [2]*University of California, Santa Cruz*

Abstract We consider calibration and structure from motion tasks for a previously introduced, highly general imaging model, where cameras are modeled as possibly unconstrained sets of projection rays. This allows to describe most existing camera types (at least for those operating in the visible domain), including pinhole cameras, sensors with radial or more general distortions, catadioptric cameras (central or non-central), etc. Generic algorithms for calibration and structure from motion tasks (pose and motion estimation and 3D point triangulation) are outlined. The foundation for a multi-view geometry of non-central cameras is given, leading to the formulation of multi-view matching tensors, analogous to the fundamental matrices, trifocal and quadrifocal tensors of perspective cameras. Besides this, we also introduce a natural hierarchy of camera models: the most general model has unconstrained projection rays whereas the most constrained model dealt with here is the central model, where all rays pass through a single point.

Keywords: Calibration, motion estimation, 3D reconstruction, camera models, non-central cameras.

1. Introduction

Many different types of cameras including pinhole, stereo, catadioptric, omnidirectional and non-central cameras have been used in computer vision. Most existing camera models are parametric (i.e. defined by a few intrinsic parameters) and address imaging systems with a single effective viewpoint (all rays pass through one point). In addition, existing calibration or structure from motion procedures are often taylor-made for specific camera models, see examples e.g. in [4, 15, 9].

The aim of this work is to relax these constraints: we want to propose and develop calibration and structure from motion methods that should work for any type of camera model, and especially also for cameras without a single effective viewpoint. To do so, we first renounce on parametric models, and adopt the following very general model: a camera acquires images consisting of pixels; each pixel captures light that travels along a ray in 3D. The camera is fully described by [11]:

- the coordinates of these rays (given in some local coordinate frame).
- the mapping between rays and pixels; this is basically a simple indexing.

This general imaging model allows to describe virtually any camera that captures light rays travelling along straight lines. Examples are (cf. figure 1):

- a camera with any type of optical distortion, such as radial or tangential.
- a camera looking at a reflective surface, e.g. as often used in surveillance, a camera looking at a spherical or otherwise curved mirror [16]. Such systems, as opposed to central catadioptric systems [1, 8] composed of cameras and parabolic mirrors, do not in general have a single effective viewpoint.
- multi-camera stereo systems: put together the pixels of all image planes; they "catch" light rays that definitely do not travel along lines that all pass through a single point. Nevertheless, in the above general camera model, a stereo system (with rigidly linked cameras) is considered as a **single** camera.
- other acquisition systems, many of them being non-central, see e.g. [2, 3, 19, 23, 24, 27, 31, 32], insect eyes, etc.

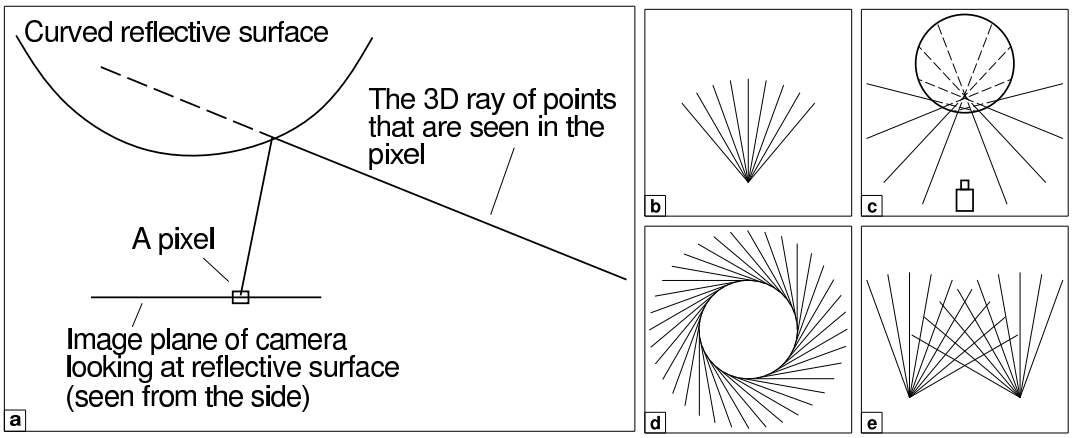


Figure 1. Examples of imaging systems. (a) Catadioptric system. Note that camera rays do not pass through their associated pixels. (b) Central camera (e.g. perspective, with or without radial distortion). (c) Camera looking at reflective sphere. This is a non-central device (camera rays are not intersecting in a single point). (d) Omnivergent imaging system [24, 27]. (e) Stereo system (non-central) consisting of two central cameras.

In this article, we first review some recent work on calibration and structure from motion for this general camera model. Concretely, we outline basics for calibration, pose and motion estimation, as well as 3D point triangulation. We then describe the foundations for a mult-view geometry of the general, non-central camera model, leading to the formulation of multi-view matching tensors, analogous to the fundamental matrices, trifocal and quadrifocal tensors of perspective cameras. Besides this, we also introduce a natural hierarchy of camera models: the most general model has unconstrained projection rays whereas the most constrained model dealt with here is the central model, where all rays pass through a single point. An intermediate model is what we term *axial cameras*: cameras for which there exists a 3D line that cuts all projection rays. This encompasses for example x-slit projections, linear pushbroom cameras and some non-central catadioptric systems. Hints will be given how to adopt the multi-view geometry proposed for the general imaging model, to such axial cameras.

The paper is organized as follows. Section 2 explains some background on Plücker coordinates for 3D lines, which are used to parameterize camera rays in this work. A hierarchy of camera models is proposed in section 3. Sections 4 to 7 deal with calibration, pose estimation, motion estimation, as well as 3D point triangulation. The multi-view geometry for the general camera model is given in section 8. A few experimental results on calibration, motion estimation and 3D reconstruction are shown in section 9.

2. Plücker Coordinates

We represent projection rays as 3D lines, via Plücker coordinates. There exist different definitions for them, the one we use is explained in the following.

Let $\mathbf{A}$ and $\mathbf{B}$ be two 3D points given by homogeneous coordinates, defining a line in 3D. The line can be represented by the skew-symmetric 4×4 Plücker matrix

$$
\begin{aligned}
\mathsf{L} &= \mathbf{A}\mathbf{B}^{\mathsf{T}} - \mathbf{B}\mathbf{A}^{\mathsf{T}} \\
&= \begin{pmatrix}
0 & A_1B_2 - A_2B_1 & A_1B_3 - A_3B_1 & A_1B_4 - A_4B_1 \\
A_2B_1 - A_1B_2 & 0 & A_2B_3 - A_3B_2 & A_2B_4 - A_4B_2 \\
A_3B_1 - A_1B_3 & A_3B_2 - A_2B_3 & 0 & A_3B_4 - A_4B_3 \\
A_4B_1 - A_1B_4 & A_4B_2 - A_2B_4 & A_4B_3 - A_3B_4 & 0
\end{pmatrix}
\end{aligned}
$$

Note that the Plücker matrix is independent (up to scale) of which pair of points on the line are chosen to represent it.

An alternative representation for the line is by its Plücker coordinate vector of length 6:

$$\mathbf{L} = \begin{pmatrix} A_4B_1 - A_1B_4 \\ A_4B_2 - A_2B_4 \\ A_4B_3 - A_3B_4 \\ A_3B_2 - A_2B_3 \\ A_1B_3 - A_3B_1 \\ A_2B_1 - A_1B_2 \end{pmatrix} \tag{1}$$

The Plücker coordinate vector can be split in two 3-vectors $\mathbf{a}$ and $\mathbf{b}$ as follows:

$$\mathbf{a} = \begin{pmatrix} L_1 \\ L_2 \\ L_3 \end{pmatrix} \quad \mathbf{b} = \begin{pmatrix} L_4 \\ L_5 \\ L_6 \end{pmatrix}$$

They satisfy the so-called Plücker constraint: $\mathbf{a}^\mathsf{T}\mathbf{b} = 0$. Furthermore, the Plücker matrix can now be conveniently written as

$$\mathsf{L} = \begin{pmatrix} [\mathbf{b}]_\times & -\mathbf{a} \\ \mathbf{a}^\mathsf{T} & 0 \end{pmatrix}$$

where $[\mathbf{b}]_\times$ is the 3×3 skew-symmetric matrix associated with the cross-product and defined by: $\mathbf{b}\times\mathbf{y} = [\mathbf{b}]_\times\mathbf{y}$.

Consider a metric transformation defined by a rotation matrix R and a translation vector $\mathbf{t}$, acting on points via:

$$\mathbf{C} \to \begin{pmatrix} \mathsf{R} & \mathbf{t} \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \mathbf{C}$$

Plücker coordinates are then transformed according to

$$\begin{pmatrix} \mathbf{a} \\ \mathbf{b} \end{pmatrix} \to \begin{pmatrix} \mathsf{R} & 0 \\ -[\mathbf{t}]_\times\mathsf{R} & \mathsf{R} \end{pmatrix} \begin{pmatrix} \mathbf{a} \\ \mathbf{b} \end{pmatrix}$$

3. A Natural Hierarchy of Camera Models

A **non-central camera** may have completely unconstrained projection rays, whereas for a **central camera**, there exists a point – the **optical center** – that lies on all projection rays. An intermediate case is what we call **axial cameras**, where there exists a line that cuts all projection rays – the **camera axis** (not to be confounded with optical axis). Examples of cameras falling into this class are pushbroom cameras (if motion is translational) [13], x-slit cameras [22, 33], and non-central catadioptric cameras of the following construction: the mirror is any surface of revolution and the optical center of the central camera (can be any central camera, i.e. not necessarily a pinhole) looking at the mirror lies on its axis of revolution. It is easy to verify that in this case, all projection rays cut the mirror's axis of revolution, i.e. the camera is an axial camera, with the mirror's axis of revolution as camera axis.

These three classes of camera models may also be defined as: existence of a linear space of d dimensions that has an intersection with all projection rays. In this sense, $d = 0$ defines central cameras, $d = 1$ axial cameras and $d = 2$ general non-central cameras.

Intermediate classes do exist. X-slit cameras are a special case of axial cameras: there actually exist 2 lines in space that both cut all projection rays. Similarly, central 1D cameras (cameras with a single row of pixels) can be defined by a point and a line in 3D. Camera models, some of which do not have much practical importance, are summarized in table 1.

It is worthwhile to consider different classes due to the following observation: the usual calibration and motion estimation algorithms proceed by first estimating a matrix or tensor by solving linear equation systems (e.g. the calibration tensors in [30] or the essential matrix [25]). Then, the parameters that are searched for (usually, motion parameters), are extracted from these. However, when estimating for example the 6×6 essential matrix of *non-central* cameras based on image correspondences obtained from *central* or *axial* cameras, then the associated linear equation system does not give a unique solution. Consequently, the algorithms for extracting

Points/lines cutting the rays	Description
None	Non-central camera
1 point	Central camera
2 points	Camera with a single projection ray
1 line	Axial camera
1 point, 1 line	Central 1D camera
2 skew lines	X-slit camera
2 coplanar lines	Union of a non-central 1D camera and a central camera
3 coplanar lines without a common point	Non-central 1D camera

Table 1. Camera models, defined by 3D points and lines that have an intersection with all projection rays of a camera.

the actual motion parameters, can not be applied without modification. This is the reason why in [29, 30] we already introduced generic calibration algorithms for both, central and non-central cameras.

In the following, we only deal with central, axial and non-central cameras. Structure from motion computations and multi-view geometry, will be formulated in terms of the Plücker coordinates of camera rays. As for *central cameras*, all rays go through a single point, the optical center. Choosing a local coordinate system with the optical center at the origin, leads to projection rays whose Plücker sub-vector $\mathbf{b}$ is zero, i.e. the projection rays are of the form:

$$\mathbf{L} = \begin{pmatrix} \mathbf{a} \\ \mathbf{0} \end{pmatrix}$$

This is one reason why the multi-linear matching tensors, e.g. the fundamental matrix, have a "base size" of 3.

As for *axial cameras*, all rays touch a line, the camera axis. Again, by choosing local coordinate systems appropriately, the formulation of the multi-view relations may be simplified, as shown in the following. Assume that the camera axis is the Z-axis. Then, all projection rays have Plücker coordinates with $L_6 = b_3 = 0$:

$$\mathbf{L} = \begin{pmatrix} \mathbf{a} \\ b_1 \\ b_2 \\ 0 \end{pmatrix}$$

Multi-view relations can thus be formulated via tensors of "base size" 5, i.e. the essential matrix for axial cameras will be of size 5×5 (see in later sections).

As for *general non-central cameras*, no such simplification occurs, and multi-view tensors will have "base size" 6.

4. Calibration

We briefly review a generic calibration approach developed in [30], an extension of [5, 10, 11], to calibrate different camera systems. As mentioned, calibration consists in determining, for every pixel, the 3D projection ray associated with it. In [11], this is done as follows: two images of a calibration object with known structure are taken. We suppose that for every pixel, we can determine the point on the calibration object, that is seen by that pixel. For each pixel in the image, we thus obtain two 3D points. Their coordinates are usually only known in a coordinate frame attached to the calibration object; however, if one knows the motion between the two object positions, one can align the coordinate frames. Then, every pixel's projection ray can be computed by simply joining the two observed 3D points.

In [30], we propose a more general approach, that does not require knowledge of the calibration object's displacement. In that case, three images need to be taken at least. The fact that all 3D points observed by a pixel in different views, are on a line in 3D, gives a constraint that allows to recover both the motion and the camera's calibration. The constraint is formulated via a set of trifocal tensors, that can be estimated linearly, and from which motion, and then calibration, can be extracted. In [30], this approach is first formulated for the use of 3D calibration objects, and for the general imaging model, i.e. for non-central cameras. We also propose variants

of the approach, that may be important in practice: first, due to the usefulness of planar calibration patterns, we specialized the approach appropriately. Second, we propose a variant that works specifically for central cameras (pinhole, central catadioptric, or any other central camera). More details are given in [29].

5. Pose Estimation

Pose estimation is the problem of computing the relative position and orientation between an object of *known* structure, and a calibrated camera. A literature review on algorithms for pinhole cameras is given in [12]. Here, we briefly show how the minimal case can be solved for general cameras. For pinhole cameras, pose can be estimated, up to a finite number of solutions, from 3 point correspondences (3D-2D) already. The same holds for general cameras. Consider 3 image points and the associated projection rays, computed using the calibration information. We parameterize generic points on the rays as follows: $\mathbf{A}_i + \lambda_i \mathbf{B}_i$.

We know the structure of the observed object, meaning that we know the mutual distances d_{ij} between the 3D points. We can thus write equations on the unknowns λ_i, that parameterize the object's pose:

$$\|\mathbf{A}_i + \lambda_i \mathbf{B}_i - \mathbf{A}_j - \lambda_j \mathbf{B}_j\|^2 = d_{ij}^2 \quad \text{for } (i,j) = (1,2),(1,3),(2,3)$$

This gives a total of 3 equations that are quadratic in 3 unknowns. Many methods exist for solving this problem, e.g. symbolic computation packages such as MAPLE allow to compute a resultant polynomial of degree 8 in a single unknown, that can be numerically solved using any root finding method.

Like for pinhole cameras, there are up to 8 theoretical solutions. For pinhole cameras, at least 4 of them can be eliminated because they would correspond to points lying behind the camera [12]. As for general cameras, determining the maximum number of feasible solutions requires further investigation. In any case, a unique solution can be obtained using one or two additional points [12]. More details on pose estimation for non-central cameras are given in [6, 21].

6. Motion Estimation

We describe how to estimate ego-motion, or, more generally, relative position and orientation of two calibrated general cameras. This is done via a generalization of the classical motion estimation problem for pinhole cameras and its associated centerpiece, the essential matrix [17]. We briefly summarize how the classical problem is usually solved [15]. Let R be the rotation matrix and $\mathbf{t}$ the translation vector describing the motion. The essential matrix is defined as $\mathsf{E} = -[\mathbf{t}]_\times \mathsf{R}$. It can be estimated using point correspondences $(\mathbf{x}_1, \mathbf{x}_2)$ across two views, using the epipolar constraint $\mathbf{x}_2^\mathsf{T} \mathsf{E} \mathbf{x}_1 = 0$. This can be done linearly using 8 correspondences or more. In the minimal case of 5 correspondences, an efficient non-linear minimal algorithm, which gives exactly the theoretical maximum of 10 feasible solutions, was only recently introduced [20]. Once the essential matrix is estimated, the motion parameters R and $\mathbf{t}$ can be extracted relatively straightforwardly [20].

In the case of our general imaging model, motion estimation is performed similarly, using pixel correspondences $(\mathbf{x}_1, \mathbf{x}_2)$. Using the calibration information, the associated projection rays can be computed. Let them be represented by their Plücker coordinates, i.e. 6-vectors $\mathbf{L}_1$ and $\mathbf{L}_2$. The epipolar constraint extends naturally to rays, and manifests itself by a 6×6 essential matrix, cf. [25] and section 8.3:

$$\mathsf{E} = \begin{pmatrix} -[\mathbf{t}]_\times \mathsf{R} & \mathsf{R} \\ \mathsf{R} & 0 \end{pmatrix}$$

The epipolar constraint then writes: $\mathbf{L}_2^\mathsf{T} \mathsf{E} \mathbf{L}_1 = 0$ [25]. Once E is estimated, motion can again be extracted straightforwardly (e.g., R can simply be read off E). Linear estimation of E requires 17 correspondences.

There is an important difference between motion estimation for central and non-central cameras: with central cameras, the translation component can only be recovered up to scale. Non-central cameras however, allow to determine even the translation's scale. This is because a single calibrated non-central camera already carries scale information (via the distance between mutually skew projection rays). One consequence is that the theoretical minimum number of required correspondences is 6 instead of 5. It might be possible, though very involved, to derive a minimal 6-point method along the lines of [20].

7. 3D Point Triangulation

We now describe an algorithm for 3D reconstruction from two or more calibrated images with known relative position. Let $\mathbf{C} = (X, Y, Z)^\mathsf{T}$ be a 3D point that is to be reconstructed, based on its projections in n images. Using calibration information, we can compute the n associated projection rays. Here, we represent the ith ray using a starting point $\mathbf{A}_i$ and the direction, represented by a unit vector $\mathbf{B}_i$. We apply the mid-point method [14, 25], i.e. determine $\mathbf{C}$ that is closest in average to the n rays. Let us represent generic points on rays using position parameters λ_i. Then, $\mathbf{C}$ is determined by minimizing the following expression over X, Y, Z and the λ_i: $\sum_{i=1}^{n} \|\mathbf{A}_i + \lambda_i \mathbf{B}_i - \mathbf{C}\|^2$.

This is a linear least squares problem, which can be solved e.g. via the Pseudo-Inverse, leading to the following explicit equation (derivations omitted):

$$\begin{pmatrix} \mathbf{C} \\ \lambda_1 \\ \vdots \\ \lambda_n \end{pmatrix} = \underbrace{\begin{pmatrix} n\mathtt{I}_3 & -\mathbf{B}_1 & \cdots & -\mathbf{B}_n \\ -\mathbf{B}_1^\mathsf{T} & 1 & & \\ \vdots & & \ddots & \\ -\mathbf{B}_n^\mathsf{T} & & & 1 \end{pmatrix}}_{\mathsf{M}}{}^{-1} \begin{pmatrix} \mathtt{I}_3 & \cdots & \mathtt{I}_3 \\ -\mathbf{B}_1^\mathsf{T} & & \\ & \ddots & \\ & & -\mathbf{B}_n^\mathsf{T} \end{pmatrix} \begin{pmatrix} \mathbf{A}_1 \\ \vdots \\ \mathbf{A}_n \end{pmatrix}$$

where $\mathtt{I}_3$ is the identity matrix of size 3×3. Due to its sparse structure, the inversion of the matrix M in this equation, can actually be performed in closed-form. Overall, the triangulation of a 3D point using n rays, can by carried out very efficiently, using only matrix multiplications and the inversion of a symmetric 3×3 matrix (details omitted).

8. Multi-View Geometry

We establish the basics of a multi-view geometry for general (non-central) cameras. Its cornerstones are, as with perspective cameras, matching tensors. We show how to establish them, analogously to the perspective case.

Here, we only talk about the calibrated case; the uncalibrated case is nicely treated for perspective cameras, since calibrated and uncalibrated cameras are linked by projective transformations. For non-central cameras however, there is no such link: in the most general case, every pair (pixel, camera ray) may be completely independent of other pairs.

8.1 Reminder on Multi-View Geometry for Perspective Cameras

We briefly review how to derive multi-view matching relations for perspective cameras [7]. Let P_i be projection matrices and $\mathbf{q}_i$ image points. A set of image points are matching, if there exists a 3D point $\mathbf{Q}$ and scale factors λ_i such that:

$$\lambda_i \mathbf{q}_i = \mathsf{P}_i \mathbf{Q}$$

This may be formulated as the following matrix equation:

$$\underbrace{\begin{pmatrix} \mathsf{P}_1 & \mathbf{q}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathsf{P}_2 & \mathbf{0} & \mathbf{q}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathsf{P}_n & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{q}_n \end{pmatrix}}_{\mathsf{M}} \begin{pmatrix} \mathbf{Q} \\ -\lambda_1 \\ -\lambda_2 \\ \vdots \\ -\lambda_n \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

The matrix M, of size $3n \times (4+n)$ has thus a null-vector, meaning that its rank is less than $4+n$. Hence, the determinants of all its submatrices of size $(4+n) \times (4+n)$ must vanish. These determinants are multi-linear expressions in terms of the coordinates of image points $\mathbf{q}_i$.

They have to be expressed for any possible submatrix. Only submatrices with 2 or more rows per view, give rise to constraints linking all projection matrices. Hence, constraints can be obtained up to n views with $2n \leq 4+n$, meaning that only for up to 4 views, matching constraints linking all views can be obtained.

The constraints for n views take the form:

$$\sum_{i_1=1}^{3}\sum_{i_2=1}^{3}\cdots\sum_{i_n=1}^{3} q_{1,i_1}q_{2,i_2}\cdots q_{n,i_n}T_{i_1,i_2,\cdots,i_n} = 0 \tag{2}$$

where the multi-view matching tensor T of dimension $3\times\cdots\times 3$ depends on and partially encodes the cameras' projection matrices P_i.

Note that as soon as cameras are calibrated, this theory applies to any central camera: for a camera with radial distortion for example, the above formulation holds for distortion-corrected image points.

8.2 Multi-View Geometry for Non-Central Cameras

Here, instead of projection matrices (depending on calibration and pose), we deal with pose matrices:

$$\mathsf{P}_i = \begin{pmatrix} \mathsf{R}_i & \mathbf{t}_i \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix}$$

These express the similarity transformations that map a point from some global reference frame, into the camera's local coordinate frames (note that since no optical center and no camera axis exist, no assumptions about the local coordinate frames are made). As for image points, they are now replaced by camera rays. Let the ith ray be represented by two 3D points $\mathbf{A}_i$ and $\mathbf{B}_i$.

Eventually, we will to obtain expressions in terms of the rays' Plücker coordinates, i.e. we will end up with matching tensors T and matching constraints of the form (2), with the difference that tensors will have size $6\times\cdots\times 6$ and act on Plücker line coordinates:

$$\sum_{i_1=1}^{6}\sum_{i_2=1}^{6}\cdots\sum_{i_n=1}^{6} L_{1,i_1}L_{2,i_2}\cdots L_{n,i_n}T_{i_1,i_2,\cdots,i_n} = 0 \tag{3}$$

In the following, we explain how to derive such matching constraints.

Consider a set of n camera rays and let them be defined by two points $\mathbf{A}_i$ and $\mathbf{B}_i$ each; the choice of points to represent a ray is not important, since later we will fall back onto the ray's Plücker coordinates.

Now, a set of n camera rays are matching, if there exist a 3D point $\mathbf{Q}$ and scale factors λ_i and μ_i associated with each ray such that:

$$\lambda_i\mathbf{A}_i + \mu_i\mathbf{B}_i = \mathsf{P}_i\mathbf{Q}$$

i.e. if the point $\mathsf{P}_i\mathbf{Q}$ lies on the line spanned by $\mathbf{A}_i$ and $\mathbf{B}_i$.

Like for perspective cameras, we group these equations in matrix form:

$$\underbrace{\begin{pmatrix} \mathsf{P}_1 & \mathbf{A}_1 & \mathbf{B}_1 & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ \mathsf{P}_2 & \mathbf{0} & \mathbf{0} & \mathbf{A}_2 & \mathbf{B}_2 & \cdots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathsf{P}_n & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{A}_n & \mathbf{B}_n \end{pmatrix}}_{\mathsf{M}} \begin{pmatrix} \mathbf{Q} \\ -\lambda_1 \\ -\mu_1 \\ -\lambda_2 \\ -\mu_2 \\ \vdots \\ -\lambda_n \\ -\mu_n \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{pmatrix}$$

As above, this equation shows that M must be rank-deficient. However, the situation is different here since the P_i are of size 4×4 now, and M of size $4n\times(4+2n)$. We thus have to consider submatrices of M of size $(4+2n)\times(4+2n)$. Furthermore, in the following we show that only submatrices with 3 rows or more per view, give rise to constraints on all pose matrices. Hence, $3n \le 4+2n$, and again, $n \le 4$, i.e. multi-view constraints are only obtained for up to 4 views.

Let us first see what happens for a submatrix of M where some view contributes only a single row. The two columns corresponding to its base points $\mathbf{A}$ and $\mathbf{B}$, are multiples of one another since they consist of zeroes

# cameras	central M	central useful submatrices	non-central M	non-central useful submatrices
2	6×6	3-3	8×8	4-4
3	9×7	3-2-2	12×10	4-3-3
4	12×8	2-2-2-2	16×12	3-3-3-3

Table 2. Cases of multi-view matching constraints for central and non-central cameras. The second columns of "central" and "non-central" contain entries of the form $x - y - z$ etc. This refers to submatrices of M containing x rows from one camera, y from another etc., whose determinant being equal zero, constitutes a matching constraint between all cameras.

only, besides a single non-zero coefficient, in the single row associated with the considered view. Hence, the determinant of the considered submatrix of M is always zero, and no constraint is available.

In the following, we exclude this case, i.e. we only consider submatrices of M where each view contributes at least two rows. Let N be such a matrix. Without loss of generality, we start to develop its determinant with the columns containing $\mathbf{A}_1$ and $\mathbf{B}_1$. The determinant is then given as a sum of terms of the following form:

$$(A_{1,j}B_{1,k} - A_{1,k}B_{1,j}) \det \bar{\mathsf{N}}_{jk}$$

where $j, k \in \{1..4\}$, $j \neq k$, and $\bar{\mathsf{N}}_{jk}$ is obtained from N by dropping the columns containing $\mathbf{A}_1$ and $\mathbf{B}_1$ as well as the rows containing $A_{1,j}$ etc.

We observe several things:

- The term $(A_{1,j}B_{1,k} - A_{1,k}B_{1,j})$ is nothing else than one of the Plücker coordinates of the ray of camera 1 (cf. section 2). By continuing with the development of the determinant of $\bar{\mathsf{N}}_{jk}$, it becomes clear that the total determinant of N can be written in the form:

$$\sum_{i_1=1}^{6} \sum_{i_2=1}^{6} \cdots \sum_{i_n=1}^{6} L_{1,i_1} L_{2,i_2} \cdots L_{n,i_n} T_{i_1,i_2,\cdots,i_n} = 0$$

  i.e. the coefficients of the $\mathbf{A}_i$ and $\mathbf{B}_i$ are "folded together" into the Plücker coordinates of camera rays and T is a matching tensor between the n cameras. Its coefficients depend exactly on the cameras' pose matrices.

- If camera 1 contributes only two rows to N, then the determinant of N becomes of the form:

$$L_{1,x} \left(\sum_{i_2=1}^{6} \cdots \sum_{i_n=1}^{6} L_{2,i_2} \cdots L_{n,i_n} T_{i_2,\cdots,i_n} \right) = 0$$

  i.e. it only contains a single coordinate of the ray of camera 1, and the tensor T does not depend at all on the pose of that camera. Hence, to obtain constraints between all cameras, every camera has to contribute at least three rows to the considered submatrix.

We are now ready to establish the different cases that lead to useful multi-view constraints. As mentioned above, for more than 4 cameras, no constraints linking all of them are available: submatrices of size at least $3n \times 3n$ would be needed, but M only has $4 + 2n$ columns. So, only for $n \leq 4$, such submatrices exist.

Table 2 gives all useful cases, both for central and non-central cameras. These lead to two-view, three-view and four-view matching constraints, encoded by essential matrices, trifocal and quadrifocal tensors.

8.3 The Case of Two Views

We have so far explained how to formulate bifocal, trifocal and quadrifocal matching constraints between non-central cameras, expressed via matching tensors of dimension 6×6 to $6 \times 6 \times 6 \times 6$. To make things more concrete, we explore the two-view case in some more detail in the following. We show how the bifocal matching tensor, or essential matrix, can be expressed in terms of the motion/pose parameters. This is then specialized from non-central to axial cameras.

8.3.1 Non-Central Cameras. For simplicity, we assume here that the global coordinate system coincides with the first camera's local coordinate system, i.e. the first camera's pose matrix is the identity. As for the pose of the second camera, we drop indices, i.e. we express it via a rotation matrix R and a translation vector $\mathbf{t}$. The matrix M is thus given as:

$$\mathsf{M} = \begin{pmatrix} 1 & 0 & 0 & 0 & A_{1,1} & B_{1,1} & 0 & 0 \\ 0 & 1 & 0 & 0 & A_{1,2} & B_{1,2} & 0 & 0 \\ 0 & 0 & 1 & 0 & A_{1,3} & B_{1,3} & 0 & 0 \\ 0 & 0 & 0 & 1 & A_{1,4} & B_{1,4} & 0 & 0 \\ R_{11} & R_{12} & R_{13} & t_1 & 0 & 0 & A_{2,1} & B_{2,1} \\ R_{21} & R_{22} & R_{23} & t_2 & 0 & 0 & A_{2,2} & B_{2,2} \\ R_{31} & R_{32} & R_{33} & t_3 & 0 & 0 & A_{2,3} & B_{2,3} \\ 0 & 0 & 0 & 1 & 0 & 0 & A_{2,4} & B_{2,4} \end{pmatrix}$$

For a matching pair of lines, M must be rank-deficient. In this two-view case, this implies that its determinant is equal to zero. As for the determinant, it can be developed to the following expression, where the Plücker coordinates $\mathbf{L}_1$ and $\mathbf{L}_2$ are defined as in equation (1):

$$\mathbf{L}_2^{\mathsf{T}} \begin{pmatrix} -[\mathbf{t}]_\times \mathsf{R} & \mathsf{R} \\ \mathsf{R} & \mathsf{0} \end{pmatrix} \mathbf{L}_1 = 0 \tag{4}$$

We find the essential matrix E and the epipolar constraint that were already mentioned in section 6.

8.3.2 Axial Cameras. As mentioned in section 3, we adopt local coordinate systems where camera rays have $L_6 = 0$. Hence, the epipolar constraint (4) can be expressed by a reduced essential matrix of size 5×5:

$$\begin{pmatrix} L_{2,1} & \cdots & L_{2,5} \end{pmatrix} \begin{pmatrix} -[\mathbf{t}]_\times \mathsf{R} & \begin{pmatrix} R_{11} & R_{12} \\ R_{21} & R_{22} \\ R_{31} & R_{32} \end{pmatrix} \\ \begin{pmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \end{pmatrix} & \mathsf{0}_{2\times 2} \end{pmatrix} \begin{pmatrix} L_{1,1} \\ \vdots \\ L_{1,5} \end{pmatrix} = 0$$

Note that this essential matrix is in general of full rank (rank 5), but may be rank-deficient. It can be shown that it is rank-deficient exactly if the two camera axes cut each other. In that case, the left and right null-vectors of E represent the camera axes of one view in the local coordinate system of the other one (one gets the Plücker vectors when adding a zero between second and third coordinates).

8.3.3 Central Cameras. As mentioned in section 3, we here deal with camera rays of the form $(L_1, L_2, L_3, 0, 0, 0)^{\mathsf{T}}$. Hence, the epipolar constraint (4) can be expressed by a reduced essential matrix of size 3×3:

$$\begin{pmatrix} L_{2,1} & L_{2,2} & L_{2,3} \end{pmatrix} \left(-[\mathbf{t}]_\times \mathsf{R} \right) \begin{pmatrix} L_{1,1} \\ L_{1,2} \\ L_{1,3} \end{pmatrix} = 0$$

We actually find here the "classical" 3×3 essential matrix $-[\mathbf{t}]_\times \mathsf{R}$ [15, 17].

9. Experimental Results

We describe a few experiments on calibration, motion estimation and 3D reconstruction, on the following three indoor scenarios:

- A house scene, captured by an omnidirectional camera and a stereo system.
- A house scene, captured by an omnidirectional and a pinhole camera.
- A scene consisting of a set of objects placed in random positions as shown in Figure 3(b), captured by an omnidirectional and a pinhole camera.

9.1 Calibration

We calibrate three types of cameras here: pinhole, stereo, and omni-directional systems.

Pinhole Camera: Figure 2(a) shows the calibration of a pinhole camera using the single center assumption [30].

Stereo camera: Here we calibrate the left and right cameras separately as two individual pinhole cameras. In the second step we capture an image of a same scene from left and right cameras and compute the motion between them using the technique described in section 6. Finally using the computed motion we obtain both the rays of left camera and the right camera in the same coordinate system, which essentially provides the required calibration information.

Omni-directional camera: Our omni-directional camera is a Nikon Coolpix-5400 camera with an E-8 Fish-Eye lens. Its field of view is 360×183. In theory, this is just another pinhole camera with large distortions. The calibration results are shown in Figure 2. Note that we have calibrated only a part of the image because three images are insufficient to capture the whole image in an omnidirectional camera. By using more than three boards it is possible to cover the whole image.

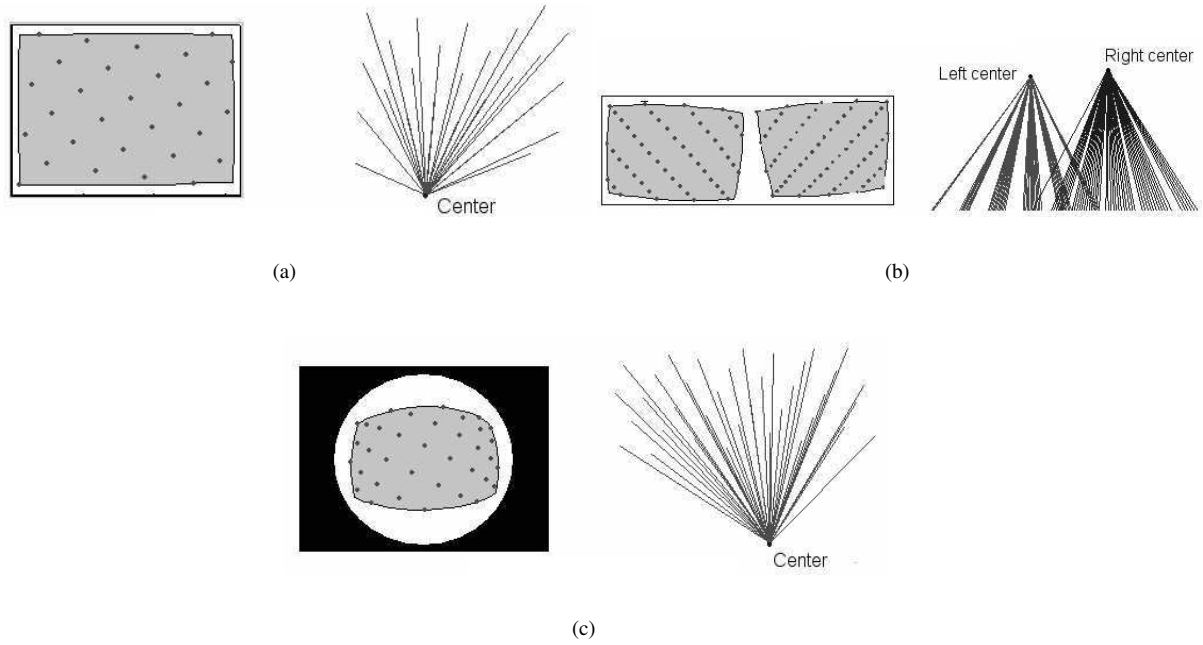


Figure 2. (a) Pinhole. (b) Stereo. (c) Omni-directional (fish-eye). The shading shows the calibrated region and the 3D rays on the right correspond to marked image pixels.

9.2 Motion and Structure Recovery

Pinhole and Omni-directional: Pinhole and omni-directional cameras are both central. Since the omni-directional camera has a very large field of view and consequently lower resolution compared to pinhole camera, the images taken from close viewpoints from these two cameras have different resolutions as shown in Figure 3. This poses a problem in finding correspondences between keypoints. Operators like SIFT [18], which are scale invariant, are not camera invariant. Direct application of SIFT failed to provide good results in our scenario. Thus we had to manually give the correspondences. One interesting research direction would be to work on the automatic matching of feature points in these images.

Stereo system and Omni-directional: A stereo system can be considered as a non-central camera with two centers. The image of a stereo system is a concatenated version of left and right camera images. Therefore the same scene point appears more than once in the image. While finding image correspondences one keypoint in the

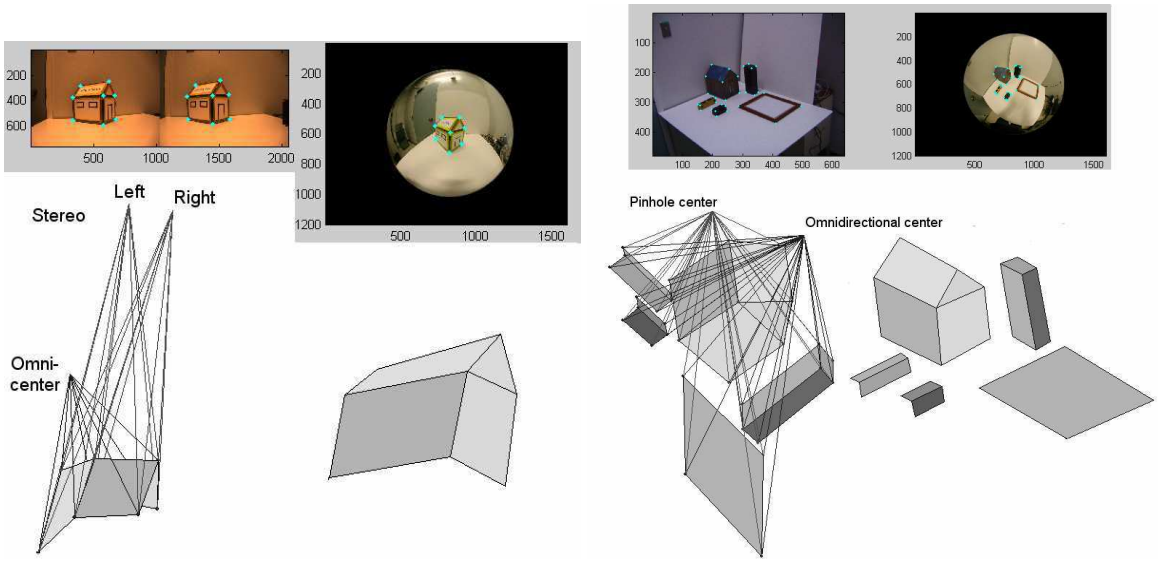


Figure 3. (a) Stereo and omni-directional. (b) Pinhole and omni-directional. We intersect the rays corresponding to the matching pixels in the images to compute the 3D points.

omni-directional image may correspond to 2 keypoints in the stereo system as shown in Figure 3(a). Therefore in the ray-intersection we intersect three rays to find one 3D point.

10. Conclusion

We have reviewed calibration and structure from motion tasks for the general non-central camera model. We also proposed a multi-view geometry for non-central cameras. A natural hierarchy of camera models has been introduced, grouping cameras into classes depending on, loosely speaking, the spatial distribution of their projection rays.

Among ongoing and future works, there is the adaptation of our calibration approach to axial and other camera models. We also continue our work on bundle adjustment for the general imaging model, cf. [26], and the exploration of hybrid systems, combining cameras of different types [28, 26].

Acknowledgements. This work was partially supported by the NSF grant ACI-0222900 and by the Multidisciplinary Research Initiative (MURI) grant by Army Research Office under contract DAA19-00-1-0352.

References

[1] S. Baker and S.K. Nayar. A Theory of Single-Viewpoint Catadioptric Image Formation. *IJCV*, 35(2), pp. 1-22, 1999.

[2] H. Bakstein. Non-central cameras for 3D reconstruction. Technical Report CTU-CMP-2001-21, Center for Machine Perception, Czech Technical University, Prague, 2001.

[3] H. Bakstein and T. Pajdla. An overview of non-central cameras. *Computer Vision Winter Workshop*, Ljubljana, Slovenia, pp. 223-233, 2001.

[4] J. Barreto and H. Araujo. Paracatadioptric Camera Calibration Using Lines. *International Conference on Computer Vision*, Nice France, pp. 1359-1365, 2003.

[5] G. Champleboux, S. Lavallée, P. Sautot and P. Cinquin. Accurate Calibration of Cameras and Range Imaging Sensors: the NPBS Method. *International Conference on Robotics and Automation*, Nice, France, pp. 1552-1558, 1992.

[6] C.-S. Chen and W.-Y. Chang. On Pose Recovery for Generalized Visual Sensors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(7), pp. 848-861, 2004.

[7] O. Faugeras and B. Mourrain. On the Geometry and Algebra of the Point and Line Correspondences Between N Images. *International Conference on Computer Vision*, Cambridge, MA, USA, pp. 951-956, 1995.

[8] C. Geyer and K. Daniilidis. A unifying theory of central panoramic systems and practical applications. *European Conference on Computer Vision*, Dublin, Ireland, Vol. II, pp. 445-461, 2000.

[9] C. Geyer and K. Daniilidis. Paracatadioptric camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5), pp. 687-695, 2002.

[10] K.D. Gremban, C.E. Thorpe and T. Kanade. Geometric Camera Calibration using Systems of Linear Equations. *International Conference on Robotics and Automation*, Philadelphia, USA, pp. 562-567, 1988.

[11] M.D. Grossberg and S.K. Nayar. A general imaging model and a method for finding its parameters. *International Conference on Computer Vision*, Vancouver, Canada, Vol. 2, pp. 108-115, 2001.

[12] R.M. Haralick, C.N. Lee, K. Ottenberg, and M. Nolle. Review and analysis of solutions of the three point perspective pose estimation problem. *International Journal of Computer Vision*, 13(3), pp. 331-356, 1994.

[13] R.I. Hartley and R. Gupta. Linear Pushbroom Cameras. *European Conference on Computer Vision*, Stockholm, Sweden, pp. 555-566, 1994.

[14] R.I. Hartley and P. Sturm. Triangulation. *Computer Vision and Image Understanding*, 68(2), pp. 146-157, 1997.

[15] R.I. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, 2000.

[16] R.A. Hicks and R. Bajcsy. Catadioptric Sensors that Approximate Wide-angle Perspective Projections. *Conference on Computer Vision and Pattern Recognition*, Hilton Head Island, USA, pp. 545-551, 2000.

[17] H.C. Longuet-Higgins. A Computer Program for Reconstructing a Scene from Two Projections. *Nature*, 293, pp. 133-135, 1981.

[18] D.G. Lowe. Object recognition from local scale-invariant features. *International Conference on Computer Vision*, Kerkyra, Greece, pp. 1150-1157, 1999.

[19] J. Neumann, C. Fermüller, and Y. Aloimonos. Polydioptric Camera Design and 3D Motion Estimation. *Conference on Computer Vision and Pattern Recognition*, Madison, WI, USA, Vol. II, pp. 294-301, 2003.

[20] D. Nistér. An Efficient Solution to the Five-Point Relative Pose Problem. *Conference on Computer Vision and Pattern Recognition*, Madison, WI, USA, Vol. II, pp. 195-202, 2003.

[21] D. Nistér. A Minimal Solution to the Generalized 3-Point Pose Problem. *Conference on Computer Vision and Pattern Recognition*, Washington DC, USA, Vol. 1, pp. 560-567, 2004.

[22] T. Pajdla. Geometry of Two-Slit Camera. Technical Report CTU-CMP-2002-02, Center for Machine Perception, Czech Technical University, Prague, 2002.

[23] T. Pajdla. Stereo with oblique cameras. *International Journal of Computer Vision*, 47(1), pp. 161-170, 2002.

[24] S. Peleg, M. Ben-Ezra, Y. Pritch. OmniStereo: Panoramic Stereo Imaging. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3), pp. 279-290, 2001.

[25] R. Pless. Using Many Cameras as One. *Conference on Computer Vision and Pattern Recognition*, Madison, WI, USA, Vol. II, pp. 587-593, 2003.

[26] S. Ramalingam, S. Lodha and P. Sturm. A Generic Structure-from-Motion Algorithm for Cross-Camera Scenarios. *5th Workshop on Omnidirectional Vision, Camera Networks and Non-Classical Cameras*, Prague, Czech Republic, pp. 175-186, 2004.

[27] H.-Y. Shum, A. Kalai, S.M. Seitz. Omnivergent Stereo. *International Conference on Computer Vision*, Kerkyra, Greece, pp. 22-29, 1999.

[28] P. Sturm. Mixing catadioptric and perspective cameras. *Workshop on Omnidirectional Vision*, Copenhagen, Denmark, pp. 60-67, 2002.

[29] P. Sturm and S. Ramalingam. A generic calibration concept-theory and algorithms. Research Report 5058, INRIA, 2003.

[30] P. Sturm and S. Ramalingam. A generic concept for camera calibration. *European Conference on Computer Vision*, Prague, Czech Republic, pp. 1-13, 2004.

[31] R. Swaminathan, M.D. Grossberg, and S.K. Nayar. A perspective on distortions. *Conference on Computer Vision and Pattern Recognition*, Madison, WI, USA, Vol. II, pp. 594-601, 2003.

[32] J. Yu and L. McMillan. General Linear Cameras. *European Conference on Computer Vision*, Prague, Czech Republic, pp. 14-27, 2004.

[33] A. Zomet, D. Feldman, S. Peleg and D. Weinshall. Mosaicing New Views: The Crossed-Slit Projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(6), pp. 741-754, 2003.

Chapter 9

Multi-View Geometry

Paper 19 [26]: P. Sturm. Multi-view geometry for general camera models. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, San Diego, USA*, volume 1, pages 206–212, June 2005.

Paper 20 [33]: P. Sturm and S. Ramalingam. Géométrie d'images multiples pour des modèles de caméra généraux. *Traitement du Signal*, 22(5), October 2005.

Paper 21 [24]: P. Sturm. Mixing catadioptric and perspective cameras. In *Proceedings of the Workshop on Omnidirectional Vision, Copenhagen, Denmark*, pages 37–44, June 2002.

Multi-View Geometry for General Camera Models

Peter Sturm
INRIA Rhône-Alpes, GRAVIR-CNRS, 38330 Montbonnot, France

Abstract

We consider the structure-from-motion problem for a highly general imaging model, where cameras are modeled as possibly unconstrained sets of projection rays. This allows to describe most existing camera types, including pinhole cameras, sensors with radial or more general distortions, catadioptric cameras (central or non-central), etc. We introduce a hierarchy of general camera models: the most general model has unconstrained projection rays whereas the most constrained model dealt with here is the central model, where all rays pass through a single point. Intermediate models are what we call axial cameras (all rays touch a single line), and x-slit cameras (rays touch two lines). The foundations for a multi-view geometry of completely non-central cameras are given, leading to the formulation of multi-view matching tensors, analogous to the fundamental/essential matrices, trifocal and quadrifocal tensors of perspective cameras. This framework is then specialized explicitly for the two-view case, for the intermediate camera types mentioned above.

1. Introduction

Many types of cameras including pinhole, stereo, catadioptric, omnidirectional and non-central cameras have been used in computer vision. Most existing camera models are parametric (i.e. defined by a few intrinsic parameters) and address imaging systems with a single effective viewpoint (all rays pass through one point). In addition, existing calibration or structure-from-motion procedures are often taylor-made for specific camera models, see e.g. [3, 11, 7].

The aim of this work is to relax these constraints: we want to propose and develop calibration and structure-from-motion methods that work for any type of camera model, including cameras without a single effective viewpoint. To do so, we first renounce on parametric models, and adopt a very general imaging model: a camera acquires images consisting of pixels; each pixel captures light traveling along a ray in 3D. The camera is fully described by [9]:

- the coordinates of these rays (given in some local coordinate frame).
- the mapping between rays and pixels; this is basically a simple indexing.

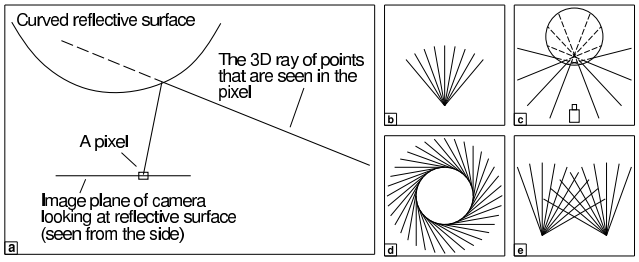


Figure 1. Examples of imaging systems; (c)–(e) are non-central devices. (a) Catadioptric camera. (b) Central camera (e.g. perspective, with or without radial distortion). (c) Camera looking at reflective sphere. (d) Omnivergent system [18, 21]. (e) Stereo.

This general imaging model allows to describe virtually any camera that captures light rays travelling along straight lines. Examples are (cf. figure 1):

- a camera with any type of optical distortion, such as radial or decentering.
- a camera looking at a reflective surface, e.g. as often used in surveillance, a camera looking at a spherical or otherwise curved mirror [12]. Such systems, as opposed to central catadioptric systems [1, 6] using e.g. parabolic mirrors, do not in general have a single effective viewpoint.
- multi-camera stereo systems: put together the pixels of all image planes; they "catch" light rays that definitely do not travel along lines that all pass through a single point. Nevertheless, in the above general camera model, a stereo system (with rigidly linked cameras) can be considered as a **single** camera.
- other acquisition systems, many of them non-central, see e.g. [2, 14, 17, 18, 21, 25, 26], insect eyes, etc.

In this paper, we propose the foundations for a multi-view geometry of the general, non-central camera model, leading to the formulation of multi-view matching tensors, analogous to the fundamental or essential matrices, trifocal and quadrifocal tensors of perspective cameras. The multi-view geometry will be formulated for calibrated cameras, i.e. we do not directly work with image point correspondences, but rather with correspondences between associated camera rays in 3D.

We also introduce a natural hierarchy of camera models: the most general model has unconstrained projection rays

whereas the most constrained model dealt with here is the central model, where all rays pass through a single point. Intermediate models considered in this paper are axial and x-slit cameras. The two-view geometry, first established for non-central cameras, is specialized for these intermediate camera types in this paper. Several works exist on epipolar geometry for omnidirectional cameras, central and non-central ones [5, 8, 15, 19, 22, 24]. Most of them aimed at obtaining matching constraints between *uncalibrated* images, whereas in this paper, we deal with *calibrated* cameras and give a rather complete treatment of the problem.

The paper is organized as follows. §2 gives some background on Plücker coordinates for 3D lines, used to parameterize projection rays. A hierarchy of camera models is proposed in §3. §4 gives parameterizations of projection rays, for the different camera models. The multi-view geometry for the general camera model, as well as two-view geometry for intermediate models, is given in §5.

2. Plücker Coordinates

We represent projection rays as 3D lines, via Plücker coordinates. Several definitions exist for them; we use the following. Let $\mathbf{A}$ and $\mathbf{B}$ be the homogeneous coordinates of 3D points defining a line. The line can be represented by the skew-symmetric 4×4 Plücker matrix $\mathsf{L} = \mathbf{A}\mathbf{B}^\mathsf{T} - \mathbf{B}\mathbf{A}^\mathsf{T}$. It is independent (up to scale) of the points used to represent the line. An alternative representation for the line is its Plücker coordinate vector of length 6:

$$\mathbf{L} = \begin{pmatrix} A_4B_1 - A_1B_4 \\ A_4B_2 - A_2B_4 \\ A_4B_3 - A_3B_4 \\ A_3B_2 - A_2B_3 \\ A_1B_3 - A_3B_1 \\ A_2B_1 - A_1B_2 \end{pmatrix} \tag{1}$$

We sometimes split it in two 3-vectors $\mathbf{a}$ and $\mathbf{b}$,

$$\mathbf{a}^\mathsf{T} = \begin{pmatrix} L_1 & L_2 & L_3 \end{pmatrix} \quad \mathbf{b}^\mathsf{T} = \begin{pmatrix} L_4 & L_5 & L_6 \end{pmatrix}$$

which satisfy the so-called Plücker constraint: $\mathbf{a}^\mathsf{T}\mathbf{b} = 0$.

Consider a metric transformation defined by a rotation matrix R and a translation vector $\mathbf{t}$, acting on points via:

$$\mathbf{Q} \rightarrow \begin{pmatrix} \mathsf{R} & \mathbf{t} \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \mathbf{Q}$$

Plücker coordinates are then transformed according to

$$\begin{pmatrix} \mathbf{a} \\ \mathbf{b} \end{pmatrix} \rightarrow \begin{pmatrix} \mathsf{R} & 0 \\ -[\mathbf{t}]_\times \mathsf{R} & \mathsf{R} \end{pmatrix} \begin{pmatrix} \mathbf{a} \\ \mathbf{b} \end{pmatrix}$$

Two lines intersect if the following relation holds:

$$\mathbf{L}_2^\mathsf{T} \begin{pmatrix} 0 & \mathsf{I} \\ \mathsf{I} & 0 \end{pmatrix} \mathbf{L}_1 = \mathbf{a}_2^\mathsf{T}\mathbf{b}_1 + \mathbf{b}_2^\mathsf{T}\mathbf{a}_1 = 0 \tag{2}$$

Table 1. Camera models, defined by 3D points and lines that have a non-empty intersection with all projection rays of a camera.

Points/lines cutting rays	Description
None	Non-central camera
1 point	Central camera
2 points	Camera with a single ray
1 line	Axial camera
1 point, 1 line	Central 1D camera
2 skew lines	X-slit camera
2 coplanar lines	Union of a non-central 1D camera and a central camera
3 coplanar lines without a common point	Non-central 1D camera

3. A Hierarchy of Camera Models

A **non-central camera** may have completely unconstrained projection rays, whereas for a **central camera**, there exists a point – the **optical center** – that lies on all projection rays. An intermediate case is what we call **axial cameras**, where there exists a line that cuts all projection rays – the **camera axis** (not to be confounded with optical axis). Examples of cameras falling into this class are:

- x-slit cameras [16, 27] (also called two-slit or crossed-slits cameras), and their special case of linear push-broom cameras [10]. Note that these form a sub-class of axial cameras, as explained below.
- stereo systems consisting of 2 central cameras or 3 or more central cameras with collinear optical centers.
- non-central catadioptric cameras of the following type: the mirror is any surface of revolution and the optical center of the central camera looking at it (can be any central camera, not only pinhole), lies on its axis of revolution. It is easy to verify that in this case, all projection rays cut the mirror's axis of revolution, i.e. the camera is an axial camera, with the mirror's axis of revolution as camera axis. Note that catadioptric cameras with a spherical mirror and a central camera looking at it, are always axial cameras.

These three classes of camera models may also be defined as: existence of a linear space of d dimensions that has an intersection with all projection rays: $d = 0$ defines central, $d = 1$ axial and $d = 2$ general non-central cameras.

Intermediate classes exist. X-slit cameras are a special case of axial cameras: two 3D lines exist that both cut all projection rays. Similarly, central 1D cameras (cameras with a single row of pixels) can be defined by a point and a line in 3D. Camera models, some of which without much practical importance, are summarized in table 1. A similar way of defining camera types was suggested in [16].

It is worthwhile to consider different classes due to the following observation: the usual calibration and motion estimation algorithms proceed by first estimating a matrix or

Table 2. Parameterization of projection rays for different camera models (see text).

Camera model	Central		Axial		X-slit	
	finite	infinite	finite	infinite	finite+finite	finite+infinite
Parameterization of projection rays	$\begin{pmatrix}\mathbf{a}\\ \mathbf{0}\end{pmatrix}$	$\begin{pmatrix}0\\0\\a_3\\b_1\\b_2\\0\end{pmatrix}$	$\begin{pmatrix}\mathbf{a}\\ b_1\\ b_2\\ 0\end{pmatrix}$	$\begin{pmatrix}0\\ a_2\\ a_3\\ \mathbf{b}\end{pmatrix}$	$\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\W&0&-Y&0\\0&0&0&1\\0&0&0&0\end{pmatrix}\begin{pmatrix}a_1\\a_2\\a_3\\b_2\end{pmatrix}$	$\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&W&0&0\\0&0&1&0\\0&0&0&1\\0&0&0&0\end{pmatrix}\begin{pmatrix}a_1\\a_3\\b_1\\b_2\end{pmatrix}$

tensor by solving linear equation systems (e.g. the calibration tensors in [23] or the essential matrix [19]). Then, the parameters that are searched for (usually, motion parameters), are extracted from these. However, when estimating for example the 6×6 essential matrix of *non-central* cameras based on image correspondences obtained from *central* or *axial* cameras, then the associated linear equation system does not give a unique solution (much like when estimating a fundamental matrix from correspondences coming from *coplanar* 3D points). Consequently, the algorithms for extracting the actual motion parameters, can not be applied without modification.

In the following, we deal with central, axial, x-slit and fully non-central cameras.

4. Parameterizations

Multi-view geometry will be formulated in terms of the Plücker coordinates of projection rays. For other models than the fully non-central one, projection rays belong to constrained sets, as explained in the previous section. We may thus choose the cameras' local coordinate systems such as to obtain "simpler" coordinate vectors for projection rays, and in turn simpler matching constraints. Since we deal with calibrated cameras, rays are given in metric coordinate systems, and we may apply rotations and translations to fix local coordinate systems. Appropriate parameterizations for different models are explained in the following.

4.1. Central Cameras

All rays go through a single point, the optical center. We distinguish the cases of a finite and infinite optical center.

Finite optical center. We choose a local coordinate system with the optical center as origin. This leads to projection rays whose Plücker sub-vector $\mathbf{b}$ is zero, cf. table 2. This is one reason why the multi-focal tensors, e.g. the fundamental matrix, can be written with a "base size" of 3.

Infinite optical center (e.g. affine camera). We can not adopt the optical center as origin, thus choose a coordinate system where it has coordinates $(0,0,1,0)^\mathsf{T}$. Projection rays are then of the form given in the 3rd column of table 2.

4.2. Axial Cameras

All rays touch a line, the camera axis. Again, by choosing local coordinate systems appropriately, the formulation of the multi-view relations may be simplified. We distinguish the cases of a finite and an infinite camera axis.

Finite axis. Assume that the camera axis is the Z-axis. Then, all projection rays have Plücker coordinates with $L_6 = b_3 = 0$, cf. the 4th column of table 2.

Infinite axis. We choose a local coordinate system where the axis is the line at infinity with coordinates $(1,0,0)^\mathsf{T}$ (line coordinates on plane at infinity). The camera axis' Plücker coordinates are then given by $(0,0,0,1,0,0)^\mathsf{T}$. Projection rays thus have coefficients with $L_1 = a_1 = 0$ (this is obtained using equation (2)), cf. the 5th column of table 2.

Multi-view relations for axial cameras, with finite or infinite axis, can thus be formulated via tensors of "base size" 5, e.g. the essential matrix will be of size 5×5 (see §5.3.2).

4.3. X-Slit Cameras

As mentioned above, x-slit cameras are defined as follows: there exist two lines – *camera axes* – that cut all projection rays. The case of the two axes cutting one another, i.e. being coplanar, is not of much interest, so we consider two mutually skew axes. Two cases are thus possible: (i) both axes are finite lines or (ii) one of the two axes is a line at infinity. In any case, one axis at least is a finite line; we adopt a local coordinate system as said above for axial cameras (finite axis is Z-axis). As for the second axis, we have to distinguish the two cases.

Two finite axes. Having fixed the first axis, we still have the freedom to rotate about it and translate along it. Since the two axes are skew, we may thus obtain a local coordinate system, where the second axis goes through a point on the Y-axis, and is parallel to the XZ-plane. Hence, it will be defined by two points as follows:

$$\mathbf{A}^\mathsf{T} = \begin{pmatrix}0 & Y & 0 & 1\end{pmatrix} \qquad \mathbf{B}^\mathsf{T} = \begin{pmatrix}X & 0 & Z & 0\end{pmatrix}$$

The second axis' Plücker coordinates are thus given by:

$$\mathbf{C}_2^\mathsf{T} = \begin{pmatrix}X & 0 & Z & -YZ & 0 & YZ\end{pmatrix}$$

Projection rays cut the two axes, so must be of the form:

$$\mathbf{L}^\mathsf{T} = \begin{pmatrix}a_1 & a_2 & a_3 & \left(\frac{YZ}{X}a_1 - Ya_3\right) & b_2 & 0\end{pmatrix}$$

The division by X is no problem since $X \neq 0$ (otherwise the second axis would be parallel to the first one, and thus coplanar, which is excluded here). Let us replace $\frac{YZ}{X}$ by W. Then, each projection ray can be parameterized by 4 coefficients (defined up to scale), as given in the 6th column of table 2. W and Y may be seen as intrinsic parameters, since they define the relative position of the two camera axes.

One finite and one infinite axis. Having fixed the first axis, we still have the freedom to rotate about it and translate along it. Translation has no effect on the infinite second axis, but we may rotate about the first axis, such that the second one has coordinates $(0, \cos\Theta, \sin\Theta)^\mathsf{T}$ (homogeneous coordinates of a line at infinity) for some Θ. The second axis' Plücker coordinates are thus:

$$\mathbf{C}_2^\mathsf{T} = \begin{pmatrix} 0 & 0 & 0 & 0 & \cos\Theta & \sin\Theta \end{pmatrix}$$

Projection rays cut the two axes, so must be of the form:

$$\mathbf{L}^\mathsf{T} = \begin{pmatrix} a_1 & -a_3\tan\Theta & a_3 & b_1 & b_2 & 0 \end{pmatrix}$$

For ease of notation, let us define $W = -\tan\Theta$. Then, each projection ray can be parameterized by 4 coefficients (defined up to scale), as given in the last column of table 2.

4.4. General Non-Central Cameras

No such simplification occurs, and multi-view tensors will have "base size" 6.

5. Multi-View Geometry

We establish the foundations of a multi-view geometry for general (non-central) cameras. Its cornerstones are, as with perspective cameras, matching tensors. We show how to establish them, analogously to the perspective case.

Here, we only deal with the calibrated case; the uncalibrated case is nicely treated for perspective cameras, since calibrated and uncalibrated images are linked by projective transformations. For non-central cameras, there is no such link: in the most general case, every pair pixel+projection ray may be completely independent of other pairs.

5.1. Reminder on Perspective Multi-View Geometry

We briefly review how to derive multi-view matching relations for perspective cameras [4]. Let P_i be projection matrices of n images. Image points $\mathbf{q}_i$ are matching, if there exist a 3D point $\mathbf{Q}$ and scale factors λ_i with:

$$\lambda_i \mathbf{q}_i = \mathsf{P}_i \mathbf{Q}, \;\; \forall i = 1 \cdots n$$

This may be formulated as the following matrix equation:

$$\underbrace{\begin{pmatrix} \mathsf{P}_1 & \mathbf{q}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathsf{P}_2 & \mathbf{0} & \mathbf{q}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathsf{P}_n & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{q}_n \end{pmatrix}}_{\mathsf{M}} \begin{pmatrix} \mathbf{Q} \\ -\lambda_1 \\ -\lambda_2 \\ \vdots \\ -\lambda_n \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

The matrix M, of size $3n \times (4+n)$ has thus a null-vector, meaning that its rank is less than $4+n$. Hence, the determinants of all submatrices of size $(4+n) \times (4+n)$ must vanish. These determinants are multi-linear expressions in terms of the coordinates of image points $\mathbf{q}_i$. Every possible submatrix should be considered, but only those with 2 or more rows per view, give rise to constraints linking all projection matrices. Hence, constraints can be obtained for up to n views with $2n \leq 4+n$, meaning that only for up to 4 views, matching constraints linking all views can be obtained.

The constraints for n views take the form:

$$\sum_{i_1=1}^{3}\sum_{i_2=1}^{3}\cdots\sum_{i_n=1}^{3} q_{1,i_1} q_{2,i_2} \cdots q_{n,i_n} T_{i_1,i_2,\cdots,i_n} = 0 \quad (3)$$

where the multi-view matching tensor T of dimension $3 \times \cdots \times 3$ depends on and partially encodes the cameras' projection matrices P_i.

Note that as soon as cameras are calibrated, this theory applies to any *central* camera: for a camera with radial distortion for example, the above formulation holds for distortion-corrected image points.

5.2. Multi-View Geometry of Non-Central Cameras

Here, instead of projection matrices (depending on calibration and pose), we deal with pose matrices:

$$\mathsf{P}_i = \begin{pmatrix} \mathsf{R}_i & \mathbf{t}_i \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \quad (4)$$

These are the similarity transformations that map a point from some global reference frame, into the camera's local coordinate frames (note that since no optical center and no camera axis exist, no assumptions about the local coordinate frames are made). As for image points, they are now replaced by projection rays. We will obtain expressions in terms of the rays' Plücker coordinates, i.e. we will end up with matching tensors T and matching constraints of the form (3), with the difference that tensors will have size $6 \times \cdots \times 6$ and act on Plücker line coordinates:

$$\sum_{i_1=1}^{6}\sum_{i_2=1}^{6}\cdots\sum_{i_n=1}^{6} L_{1,i_1} L_{2,i_2} \cdots L_{n,i_n} T_{i_1,i_2,\cdots,i_n} = 0 \quad (5)$$

In the following, we explain how to derive such matching constraints. Consider a set of n projection rays and let them be defined by two points $\mathbf{A}_i$ and $\mathbf{B}_i$ each; the choice of points to represent a ray is not important, since later we will fall back onto the ray's Plücker coordinates.

Now, a set of n projection rays are matching, if there exist a 3D point $\mathbf{Q}$ and scale factors λ_i and μ_i with:

$$\lambda_i \mathbf{A}_i + \mu_i \mathbf{B}_i = \mathsf{P}_i \mathbf{Q}, \;\; \forall i = 1 \cdots n$$

i.e. if the point $\mathsf{P}_i\mathbf{Q}$ lies on the line spanned by $\mathbf{A}_i$ and $\mathbf{B}_i$.

Like for perspective cameras, we group these equations in matrix form:

$$\underbrace{\begin{pmatrix} \mathsf{P}_1 & \mathbf{A}_1 & \mathbf{B}_1 & \cdots & \mathbf{0} & \mathbf{0} \\ \mathsf{P}_2 & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathsf{P}_n & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{A}_n & \mathbf{B}_n \end{pmatrix}}_{\mathsf{M}} \begin{pmatrix} \mathbf{Q} \\ -\lambda_1 \\ -\mu_1 \\ \vdots \\ -\lambda_n \\ -\mu_n \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{pmatrix}$$

As above, this equation shows that M must be rank-deficient. However, the situation is different here since the P_i are of size 4×4, and M of size $4n \times (4 + 2n)$. We thus consider submatrices of size $(4+2n) \times (4+2n)$. In the following we show that only submatrices with 3 rows or more per view, give rise to constraints linking all pose matrices. Thus, $3n \leq 4 + 2n$, and $n \leq 4$, i.e. multi-view constraints are again only obtained for up to 4 views.

Let us first see what happens for a submatrix of M where some view contributes a single row. The two columns corresponding to its base points $\mathbf{A}$ and $\mathbf{B}$, are then multiples of one another: they contain only zeroes, besides a single non-zero coefficient, in the single row associated with the considered view. Hence, the determinant of the submatrix of M is *always* zero, and no constraint is available.

In the following, we exclude this case, i.e. we only consider submatrices of M where each view contributes at least two rows. Let N be such a matrix. Without loss of generality, we start to develop its determinant with the columns containing $\mathbf{A}_1$ and $\mathbf{B}_1$. The determinant is then given as a sum of terms of the following form:

$$(A_{1,j}B_{1,k} - A_{1,k}B_{1,j}) \det \bar{\mathsf{N}}_{jk}$$

where $j, k \in \{1..4\}$, $j \neq k$, and $\bar{\mathsf{N}}_{jk}$ is obtained from N by dropping the columns containing $\mathbf{A}_1$ and $\mathbf{B}_1$ as well as the rows containing $A_{1,j}$ and $A_{1,k}$. We observe several things:

- The term $(A_{1,j}B_{1,k} - A_{1,k}B_{1,j})$ is nothing else than a Plücker coordinate of the ray of camera 1 (cf. §2). By continuing with the development of the determinant of $\bar{\mathsf{N}}_{jk}$, it becomes clear that the total determinant of N can be written in the form:

  $$\sum_{i_1=1}^{6} \sum_{i_2=1}^{6} \cdots \sum_{i_n=1}^{6} L_{1,i_1} L_{2,i_2} \cdots L_{n,i_n} T_{i_1,i_2,\cdots,i_n} = 0$$

  i.e. the coefficients of the $\mathbf{A}_i$ and $\mathbf{B}_i$ are "folded together" into Plücker coordinates of projection rays and T is a matching tensor relating the n cameras. Its coefficients depend exactly on the cameras' pose matrices.

- If camera 1 contributes only two rows to N, then the determinant of N will have the form:

  $$L_{1,x} \left(\sum_{i_2=1}^{6} \cdots \sum_{i_n=1}^{6} L_{2,i_2} \cdots L_{n,i_n} T_{i_2,\cdots,i_n} \right) = 0$$

  i.e. it only contains a single coordinate $L_{1,x}$ of the ray of camera 1, and the tensor T does not depend at all on the pose of that camera. Hence, to obtain constraints relating *all* cameras, each camera has to contribute at least three rows to the considered submatrix of M.

Table 3. Cases of multi-view matching constraints for central and non-central cameras. Columns named "useful" contain entries of the form x-y-z etc. that correspond to sub-matrices of M that give rise to matching constraints linking *all* views: x-y-z refers to sub-matrices containing x rows from one camera, y from another etc.

	central		non-central	
# views	M	useful	M	useful
2	6×6	3-3	8×8	4-4
3	9×7	3-2-2	12×10	4-3-3
4	12×8	2-2-2-2	16×12	3-3-3-3

We are now ready to establish the different cases that lead to useful multi-view constraints. As mentioned above, for more than 4 cameras, no constraints linking all of them are available: submatrices of size at least $3n \times 3n$ would be needed, but M only has $4+2n$ columns. So, only for $n \leq 4$, such constraints exist.

Table 3 gives all useful cases, both for central and non-central cameras. These lead to two-view, three-view and four-view matching constraints, encoded by essential matrices, trifocal and quadrifocal tensors. Deriving their forms is now mainly a mechanical task.

5.3. The Case of Two Views

We have so far explained how to formulate bifocal, trifocal and quadrifocal matching constraints between non-central cameras, expressed via matching tensors of dimension 6×6 to $6 \times 6 \times 6 \times 6$. To make things more concrete, we explore the two-view case in some more detail in the following. We show how the bifocal matching tensor, or essential matrix, can be expressed in terms of the pose (or, motion) parameters. This is then specialized from non-central to axial, x-slit and central cameras. The essential matrices for these cases are summarized in table 4. That table also gives the minimum numbers of correspondences required for estimating them using linear equations. These are not explained in detail due to lack of space, but can be derived easily by considering coefficients in essential matrices, that are zero or appear twice.

5.3.1. Non-Central Cameras

For simplicity, we assume here that the global coordinate system coincides with the first camera's local coordinate system, i.e. the first camera's pose matrix is the identity. As for the pose of the second camera, we drop indices, i.e. we express it via a pose matrix P, composed of a rotation

Table 4. Essential matrices for different camera models. The last column gives the minimum number of correspondences between projection rays required for computing essential matrices using linear equations.

Camera model	Essential matrix	Size	# corr.
Non-central	$\mathsf{E}_n = \begin{pmatrix} -[\mathbf{t}]_\times \mathsf{R} & \mathsf{R} \\ \mathsf{R} & \mathsf{0}_{3\times 3} \end{pmatrix}$	6×6	17
Axial with finite axis	$\mathsf{E}_{af} = \begin{pmatrix} & \begin{pmatrix} R_{11} & R_{12} \\ R_{21} & R_{22} \\ R_{31} & R_{32} \end{pmatrix} \\ -[\mathbf{t}]_\times \mathsf{R} & \\ \begin{pmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \end{pmatrix} & \mathsf{0}_{2\times 2} \end{pmatrix}$	5×5	16
Axial with infinite axis	$\mathsf{E}_{ai} = \begin{pmatrix} t_1R_{32} - t_3R_{12} & t_1R_{33} - t_3R_{13} & R_{21} & R_{22} & R_{23} \\ t_2R_{12} - t_1R_{22} & t_2R_{13} - t_1R_{23} & R_{31} & R_{32} & R_{33} \\ R_{12} & R_{13} & 0 & 0 & 0 \\ R_{22} & R_{23} & 0 & 0 & 0 \\ R_{32} & R_{33} & 0 & 0 & 0 \end{pmatrix}$	5×5	11
X-slit with two finite axes	$\mathsf{E}_{xff} = \begin{pmatrix} 1 & 0 & 0 & W_2 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -Y_2 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \mathsf{E}_{af} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ W_1 & 0 & -Y_1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$	4×4	13
X-slit with one finite and one infinite axis	$\mathsf{E}_{xfi} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & W_2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \mathsf{E}_{af} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & W_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$	4×4	10
Central with finite optical center	$\mathsf{E}_{cf} = -[\mathbf{t}]_\times \mathsf{R}$	3×3	8
Central with infinite optical center	$\mathsf{E}_{ci} = \begin{pmatrix} t_2R_{13} - t_1R_{23} & R_{31} & R_{32} \\ R_{13} & 0 & 0 \\ R_{23} & 0 & 0 \end{pmatrix}$	3×3	4

matrix R and a translation vector $\mathbf{t}$, according to (4). The matrix M is thus given as:

$$\mathsf{M}_{8\times 8} = \begin{pmatrix} \mathtt{I}_{4\times 4} & \mathbf{A}_1 & \mathbf{B}_1 & \mathbf{0} & \mathbf{0} \\ \mathsf{P} & \mathbf{0} & \mathbf{0} & \mathbf{A}_2 & \mathbf{B}_2 \end{pmatrix}$$

For a matching pair of rays, M must be rank-deficient. Here, this implies that its determinant is equal to zero. It can be developed to the following expression, where the Plücker coordinates $\mathbf{L}_1$ and $\mathbf{L}_2$ are defined as in equation (1):

$$\mathbf{L}_2^\mathsf{T} \underbrace{\begin{pmatrix} -[\mathbf{t}]_\times \mathsf{R} & \mathsf{R} \\ \mathsf{R} & \mathsf{0} \end{pmatrix}}_{\mathsf{E}_n} \mathbf{L}_1 = 0 \qquad (6)$$

We find the essential matrix E_n, as was done in [19].

5.3.2. Axial Cameras

Finite axis. As mentioned in §3, we adopt local coordinate systems where projection rays have $L_6 = 0$. Hence, the epipolar constraint (6) can be expressed by a reduced essential matrix of size 5×5, which acts on reduced Plücker vectors, consisting of the first five Plücker coordinates. This essential matrix is obtained from the non-central one E_n (6), by dropping its sixth row and column, leading to E_{af}, as given in table 4.

Note that E_{af} is in general of full rank (rank 5), but may be rank-deficient. It can be shown that it is rank-deficient exactly if the axes of the two cameras cut each other. In that case, the left and right null-vectors of E_{af} represent the camera axes of one view in the local coordinate system of the other one (one gets their Plücker vectors when adding a zero as 6th coordinate to the length-5 null-vectors).

Infinite axis. The epipolar constraint (6) can be expressed by a reduced essential matrix E_{ai} (cf. table 4) of size 5×5, acting on reduced Plücker vectors, consisting of the last five Plücker coordinates (cf. table 2). It is always rank-deficient (the two camera axes are lines at infinity, thus always cut each other, cf. the discussion in the previous paragraph).

5.3.3. X-Slit Cameras

Two finite axes. We get a reduced essential matrix E_{xff} (cf. table 4) of size 4×4, acting on reduced Plücker vectors of the form $(a_1, a_2, a_3, b_2)^\mathsf{T}$ (cf. §4.3).

Contrary to previous cases, the essential matrix now not only encodes motion, but also "intrinsic parameters" (the coefficients W_i and Y_i of the two cameras' second axes).

One finite and one infinite axis. We get a reduced essential matrix E_{xfi} (cf. table 4) of size 4×4, acting on reduced Plücker vectors of the form $(a_1, a_3, b_1, b_2)^\mathsf{T}$ (cf. §4.3). Again, it not only encodes motion, but also "intrinsic parameters" (the coefficients W_i of the two cameras' infinite axes).

5.3.4. Central Cameras

Finite optical center. As mentioned in §3, we here deal with projection rays of the form $(L_1, L_2, L_3, 0, 0, 0)^\mathsf{T}$. Hence, the epipolar constraint (6) can be expressed by a 3×3 essential matrix. We actually find here the "classical" 3×3 essential matrix $\mathsf{E}_{cf} = -[\mathbf{t}]_\times \mathsf{R}$ [11, 13].

Infinite optical center. The essential matrix in this case is E_{ci}, cf. table 4. This resembles the affine fundamental matrix [20], but is not exactly the same: here, the essential matrix acts on 3D lines, not on image points. For example, the right null-vector of E_{ci} is $(0, R_{32}, -R_{31})^\mathsf{T}$, which represents the 3D line with Plücker coordinates $(0, 0, 0, R_{32}, -R_{31}, 0)^\mathsf{T}$. This is the line spanned by the two optical centers, i.e. the baseline (expressed in the first camera's coordinate system).

6. Conclusion

We have proposed a multi-view geometry for non-central cameras, the first to our knowledge. A natural hierarchy of camera models has been introduced, grouping cameras into classes depending on, loosely speaking, the spatial distribution of their projection rays. Two-view geometry was specialized in detail to different camera models. We hope that this theoretical work allows to define some common ground for recent efforts in characterizing the geometry of non-classical cameras.

Concerning possibilites for further work, geometrical relations between cameras of *different* types would be straightforward to derive along the lines used here, and all expressions can of course be transcribed in tensor notation. In this paper, we concentrated on the theory and did not address the issue of actually estimating the matching tensors and extracting motion parameters from them. It is relatively straightforward though to extract the motion parameters from the various essential matrices, due to their forms given in table 4. Experiments with the essential matrix for non-central cameras were successful, as also reported in [19], and experiments with intermediate camera types are ongoing.

Finally, we would like to note that, although motivated by the generic imaging model associating rays to pixels, the multi-view relations derived here hold naturally for any camera model that allows to attribute projection rays to image points with sub-pixel precision.

References

[1] S. Baker, S.K. Nayar. A theory of single-viewpoint catadioptric image formation. *IJCV*, 35(2), 1999.

[2] H. Bakstein, T. Pajdla. An overview of non-central cameras. *Computer Vision Winter Workshop, Ljubljana*, 2001.

[3] J.P. Barreto, H. Araujo. Paracatadioptric camera calibration using lines. *ICCV*, 2003.

[4] O. Faugeras, B. Mourrain. On the geometry and algebra of the point and line correspondences between n images. *ICCV*, 1995.

[5] D. Feldman, T. Pajdla, D. Weinshall. On the epipolar geometry of the crossed-slits projection. *ICCV*, 2003.

[6] C. Geyer, K. Daniilidis. A unifying theory of central panoramic systems and practical applications. *ECCV*, 2000.

[7] C. Geyer, K. Daniilidis. Paracatadioptric camera calibration. *PAMI*, 24(5), 2002.

[8] C. Geyer, K. Daniilidis. Mirrors in Motion: Epipolar geometry and motion estimation. *ICCV*, 2003.

[9] M.D. Grossberg, S.K. Nayar. A general imaging model and a method for finding its parameters. *ICCV*, 2001.

[10] R.I. Hartley, R. Gupta. Linear pushbroom cameras. *ECCV*, 1994.

[11] R.I. Hartley, A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[12] R.A. Hicks, R. Bajcsy. Catadioptric sensors that approximate wide-angle perspective projections. *CVPR*, 2000.

[13] H.C. Longuet-Higgins. A computer program for reconstructing a scene from two projections. *Nature*, 293, 1981.

[14] J. Neumann, C. Fermüller, Y. Aloimonos. Polydioptric camera design and 3D motion estimation. *CVPR*, 2003.

[15] T. Pajdla. Epipolar Geometry of Some Non-classical Cameras. *Computer Vision Winter Workshop, Bled*, 2001.

[16] T. Pajdla. Geometry of two-slit camera. Report CTU-CMP-2002-02, Czech Technical University, Prague, 2002.

[17] T. Pajdla. Stereo with oblique cameras. *IJCV*, 47(1-3), 2002.

[18] S. Peleg, M. Ben-Ezra, Y. Pritch. Omnistereo: Panoramic stereo imaging. *PAMI*, 23(3), 2001.

[19] R. Pless. Using many cameras as one. *CVPR*, 2003.

[20] L.S. Shapiro, A. Zisserman, M. Brady. 3D Motion Recovery via Affine Epipolar Geometry. *IJCV*, 16(2), 1995.

[21] H.-Y. Shum, A. Kalai, S.M. Seitz. Omnivergent stereo. *ICCV*, 1999.

[22] P. Sturm. Mixing catadioptric and perspective cameras. *Workshop on Omnidirectional Vision, Copenhagen*, 2002.

[23] P. Sturm, S. Ramalingam. A generic concept for camera calibration. *ECCV*, 2004,

[24] T. Svoboda. *Central Panoramic Cameras: Design, Geometry, Egomotion*. PhD Thesis, Czech Technical University, Prague, 1999.

[25] R. Swaminathan, M.D. Grossberg, S.K. Nayar. A perspective on distortions. *CVPR*, 2003.

[26] J. Yu, L. McMillan. General linear cameras. *ECCV*, 2004.

[27] A. Zomet, D. Feldman, S. Peleg, D. Weinshall. Mosaicing new views: The crossed-slit projection. *PAMI*, 25(6), 2003.

Géométrie d'images multiples pour des modèles de caméras généraux
Multi-view geometry for general camera models

Peter Sturm
INRIA Rhône-Alpes
655 Avenue de l'Europe
38330 Montbonnot
France
Peter.Sturm@inrialpes.fr

Srikumar Ramalingam
University of California
Dept. of Computer Science
Santa Cruz
USA
srikumar@cse.ucsc.edu

Résumé. Nous considérons le problème de l'estimation de la structure et du mouvement pour un modèle de caméras hautement général, qui représente une caméra par un ensemble de rayons de projection. Ceci permet de décrire la plupart des types de caméras existants (du moins celles qui opèrent dans le domaine visible), y inclus les caméras sténopé, les caméras avec des distorsions radiales ou plus générales, les caméras catadioptriques (à point de vue unique ou non), etc. Nous introduisons une hiérarchie de modèles de caméras généraux : le modèle le plus général peut posséder des rayons de projection quelconques tandis que le modèle le plus contraint que nous considérons ici est le modèle à point de vue unique (tous les rayons passent par un même point). Parmi les modèles intermédiaires, nous identifions ce que nous appelons les caméras axiales (tous les rayons touchent une même ligne) et les caméras connues sous le nom de « cross-slit » (les rayons touchent deux lignes). Les fondements d'une géométrie d'images multiples pour le modèle de caméras le plus général sont donnés. Ils se manifestent par la formulation de tenseurs d'appariement multi-vues, qui sont l'analogue des matrices fondamentales/essentielles, tenseurs trifocaux ou quadrifocaux des caméras perspectives. Ce cadre théorique général est ensuite spécialisé pour les modèles de caméras intermédiaires mentionnés, pour le cas de deux images.

Mots clés. Modèle de caméras, caméra non centrale, caméra omnidirectionnelle, tenseur d'appariement, géométrie épipolaire, géométrie d'images multiples.

Abstract. We consider the structure from motion problem for a previously introduced, highly general imaging model, where cameras are modeled as possibly unconstrained sets of projection rays. This allows to describe most existing camera types (at least for those operating in the visible domain), including pinhole cameras, sensors with radial or more general distortions, catadioptric cameras (central or non-central), etc. We introduce a hierarchy of general camera models : the most general model has unconstrained projection rays whereas the most constrained model dealt with here is the central model, where all rays pass through a single point. Intermediate models are what we call axial cameras (all rays touch a single line), and x-slit cameras (rays touch two lines). The foundations for a multi-view geometry of completely non-central cameras are given, leading to the formulation of multi-view matching tensors, analogous to the fundamental/essential matrices, trifocal and quadrifocal tensors of perspective cameras. This framework is then specialized explicitly for the two-view case, for the intermediate camera types mentioned above.

Keywords. Camera model, non-central camera, omnidirectional camera, matching tensor, epipolar geometry, multi-view geometry.

1 Introduction

Beaucoup de différents capteurs sont utilisés en vision par ordinateur, dont les caméras perspectives, les systèmes stéréo, les caméras omnidirectionnelles (par exemple, celles catadioptriques), etc. La plupart des modèles utilisés pour ces caméras sont paramétriques et définis par quelques paramètres intrinsèques (distance focale, coefficients de distorsion, etc.) et considèrent surtout des caméras à point de vue unique. De plus, les algorithmes existants de calibrage, de reconstruction 3-D ou d'estimation du mouvement, sont le plus souvent conçus pour un seul modèle de caméras à la fois (voir par exemple [5, 15, 11]).

Le but de notre travail est de relâcher ces contraintes : nous voulons proposer et développer des approches de calibrage, de reconstruction 3-D etc. qui puissent être appliquées quels que soient les types des caméras utilisées, notamment les caméras omnidirectionnelles et/ou n'ayant pas de point de vue unique. Pour ce faire, nous renonçons aux modèles paramétriques classiques et adoptons un modèle très général [13] : une caméra acquiert des images qui consistent en un ensemble de pixels ; chaque pixel capte la lumière qui se propage le long d'un rayon (rayon de projection). Une caméra est alors complètement modélisée par :

- les coordonnées de ces rayons (en 3-D, données par rapport à un repère local de la caméra) ;
- la correspondance entre pixels et rayons.

Ce modèle général permet de décrire la plupart des types de caméra, par exemple (cf. la figure 1) :

- des caméras avec des distorsions optiques quelconques, telles les distorsions radiales ou tangentielles ;
- les caméras catadioptriques, c'est-à-dire des caméras qui perçoivent la scène au travers d'une réflexion dans un miroir, typiquement de forme convexe. De tels systèmes peuvent avoir un point de vue unique [2, 10], mais uniquement si le miroir ainsi que la position relative miroir–caméra sont bien choisis. Si un miroir sphérique est utilisé, ou un miroir dont la surface ne correspond pas à une quadrique [16], le système catadioptrique n'aura pas de point de vue unique ;
- des systèmes stéréo (deux caméras ou plus) : conceptuellement, on peut considérer un système stéréo comme un seul capteur qui consiste de l'ensemble des pixels des caméras et des rayons associés. Il s'agit bien évidemment d'un capteur qui n'a pas de point de vue unique ;
- d'autres systèmes d'acquisition, dont beaucoup n'ont pas de point de vue unique et/ou sont de type omnidirectionnel [3, 4, 14, 6, 7, 18, 23, 24, 28, 33, 34] ;
- un exemple où le modèle énoncé ci-dessus ne s'appliquerait pas est celui d'une caméra qui regarde une scène à travers une interface entre deux matières. Considérons par exemple une caméra qui regarde dans l'eau mais n'y est pas plongée : les rayons de projection sont réfractés et si la caméra se déplace, l'ensemble des rayons ne se déplacera pas de manière rigide.

Bien évidemment, le modèle de caméras que nous utilisons n'est qu'une approximation : en réalité, un pixel capte non pas un seul rayon de lumière, mais plutôt de la lumière qui se propage dans un certain volume. Cette remarque s'applique pourtant à la majorité des modèles existants. D'autres aspects importants, par exemple ceux liés à la photométrie, sont très bien décrits dans [13].

Dans cet article, nous introduisons les fondements pour une géométrie d'images multiples pour le modèle de caméras générique décrit ci-dessus. Ils s'expriment au travers de tenseurs d'appariement, similairement aux matrices fondamentales ou essentielles et aux tenseurs trifocaux ou quadrifocaux des caméra perspectives. Nous rappelons ici simplement que les tenseurs d'appariement servent

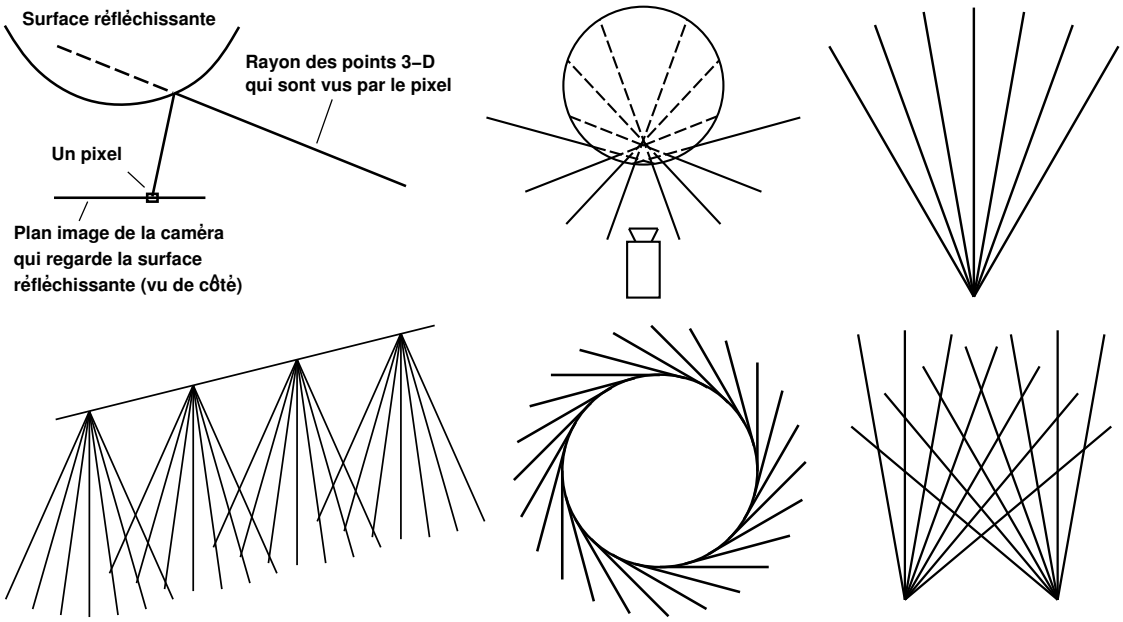


FIG. 1 – Exemples de types de caméras. Première ligne : (i) Système catadioptrique (notons que les rayons ne passent pas par les pixels associés). (ii) Système catadioptrique basé sur un miroir sphérique (ce système n'a pas de point de vue unique – les rayons ne se coupent pas en un seul point). (iii) Caméra à point de vue unique (par exemple, caméra perspective, avec ou sans distorsion radiale ou autre). Deuxième ligne : (i) Caméra de type « push-broom ». (ii) Système d'acquisition dit « omni-vergent » [24, 28]. (iii) Système stéréo.

à donner des contraintes pour l'appariement de primitives géométriques entre images. Ceci sera mieux expliqué dans la suite.

Nous formulons la géométrie d'images multiples pour des caméras calibrées, c'est-à-dire pour lesquelles la relation pixels–rayons est connue. Ainsi, les correspondances entre pixels d'images différentes se traduiront directement en correspondances de rayons de projection en 3-D. Les tenseurs d'appariement que nous allons dériver, agissent alors sur les coordonnées des rayons.

Nous introduisons également une hiérarchie naturelle de modèles de caméras : le modèle le plus général consiste d'un ensemble non contraint de rayons de projection tandis que le modèle le plus contraint considéré ici est celui des caméras à point de vue unique. Dans la suite, nous utilisons l'expression concise de *caméra centrale* pour désigner les caméras à point de vue unique. Des caméras sans point de vue unique sont appelées *caméras non centrales*. Un modèle intermédiaire est ce que nous appelons une *caméra axiale* : une caméra telle qu'il existe une droite en 3-D qui touche tous les rayons de projection. Ce modèle comprend les caméras de type « push-broom » [14] et certaines caméras catadioptriques. Une sous-classe est celle des caméras dites de type « cross-slit » : il existe deux droites en 3-D qui touchent tous les rayons (les caméras de type « push-broom linéaire » sont en effet de ce type).

La géométrie d'images multiples, formulée d'abord pour des caméras non centrales générales, est ensuite spécialisée à ces modèles intermédiaires, pour le cas de base de deux vues (géométrie épipolaire). Il existe plusieurs travaux sur la géométrie épipolaire de caméras omnidirectionnelles, centrales ou non [9, 12, 20, 21, 25, 30, 32]. Le but de la plupart de ces travaux est d'obtenir des contraintes d'appariement pour des caméras non calibrées (ce qui est difficile même pour certaines caméras catadioptriques centrales). Dans cet article, nous abordons le cas de caméras calibrées et

donnons un traitement assez complet du problème.

Cet article est structuré comme suit. Dans la section 2, nous rappelons la définition et des propriétés des coordonnées de Plücker pour les droites en 3-D, qui sont utilisées pour paramétrer les rayons de projection. Une hiérarchie de modèles de caméras est proposée dans la section 3. Des paramétrisations de rayons de projection pour différents modèles de caméras, sont proposées en section 4. La géométrie d'images multiples pour le modèle de caméras général est développée dans la section 5. Cette géométrie est ensuite explorée en détail pour le cas de deux vues et différents modèles de caméras, en section 6.

Notations utilisées : les matrices sont notées en sans empattement ($\mathsf{L}, \mathsf{R}, \ldots$), les vecteurs en caractères gras ($\mathbf{a}, \mathbf{b}, \ldots$) et les scalaires en caractères italiques ($u, v, \ldots$). Les coefficients de tenseurs, matrices ou vecteurs sont des scalaires, donc notés en italique ($T_{i,j,k}, L_{i,j}, \ldots$). Le produit vectoriel de deux vecteurs de longueur 3 est écrit $\mathbf{u} \times \mathbf{v}$. La notation $[\mathbf{u}]_\times$ désigne la matrice anti-symétrique de dimension 3×3 définie par le produit vectoriel : $[\mathbf{u}]_\times \mathbf{v} = \mathbf{u} \times \mathbf{v}$. La transposée d'une matrice est notée par L^T. Les vecteurs sont parfois interprétés comme des matrices à une colonne ; la transposée $\mathbf{a}^\mathsf{T}$ d'un vecteur désigne donc une matrice à une ligne.

2 Coordonnées de Plücker

Nous représentons les rayons de projection par des droites en 3-D, en utilisant leurs coordonnées de Plücker. Nous en utilisons la définition suivante.

Soient $\mathbf{A}$ et $\mathbf{B}$ deux points 3-D, donnés en coordonnées homogènes. La droite définie par ces points peut être représentée par la matrice 4×4 anti-symétrique L, dite matrice de Plücker :

$$\begin{aligned} \mathsf{L} &= \mathbf{A}\mathbf{B}^\mathsf{T} - \mathbf{B}\mathbf{A}^\mathsf{T} \\ &= \begin{pmatrix} 0 & A_1B_2 - A_2B_1 & A_1B_3 - A_3B_1 & A_1B_4 - A_4B_1 \\ A_2B_1 - A_1B_2 & 0 & A_2B_3 - A_3B_2 & A_2B_4 - A_4B_2 \\ A_3B_1 - A_1B_3 & A_3B_2 - A_2B_3 & 0 & A_3B_4 - A_4B_3 \\ A_4B_1 - A_1B_4 & A_4B_2 - A_2B_4 & A_4B_3 - A_3B_4 & 0 \end{pmatrix} \end{aligned}$$

Notons que la matrice de Plücker d'une droite est indépendante (à l'échelle près) de la paire des points sur cette droite ayant servie à son calcul.

Une représentation alternative de la droite est le vecteur des coordonnées de Plücker, de longueur 6 :

$$\mathbf{L} = \begin{pmatrix} A_4B_1 - A_1B_4 \\ A_4B_2 - A_2B_4 \\ A_4B_3 - A_3B_4 \\ A_3B_2 - A_2B_3 \\ A_1B_3 - A_3B_1 \\ A_2B_1 - A_1B_2 \end{pmatrix} \tag{1}$$

Nous identifions deux sous-vecteurs de longueur 3, $\mathbf{a}$ et $\mathbf{b}$:

$$\mathbf{a} = \begin{pmatrix} L_1 \\ L_2 \\ L_3 \end{pmatrix} \quad \mathbf{b} = \begin{pmatrix} L_4 \\ L_5 \\ L_6 \end{pmatrix}$$

Ces deux vecteurs satisfont la contrainte dite de Plücker : $\mathbf{a}^\mathsf{T}\mathbf{b} = 0$. Un vecteur de longueur 6 correspond à des coordonnées de Plücker d'une droite si et seulement si il vérifie cette contrainte. Avec cette définition de $\mathbf{a}$ et $\mathbf{b}$, la matrice de Plücker peut s'écrire :

$$\mathsf{L} = \begin{pmatrix} [\mathbf{b}]_\times & -\mathbf{a} \\ \mathbf{a}^\mathsf{T} & 0 \end{pmatrix}$$

Considérons maintenant comment les droites sont transformées par des changements de repère. Soit une transformation euclidienne, définie par une matrice de rotation R et un vecteur de translation $\mathbf{t}$, qui agit sur les points 3-D comme suit :

$$\mathbf{C} \to \begin{pmatrix} \mathsf{R} & \mathbf{t} \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \mathbf{C}$$

Les coordonnées de Plücker sont alors transformées ainsi :

$$\begin{pmatrix} \mathbf{a} \\ \mathbf{b} \end{pmatrix} \to \begin{pmatrix} \mathsf{R} & 0 \\ -[\mathbf{t}]_\times \mathsf{R} & \mathsf{R} \end{pmatrix} \begin{pmatrix} \mathbf{a} \\ \mathbf{b} \end{pmatrix}$$

Notons finalement que deux droites $\mathbf{L}_1$ et $\mathbf{L}_2$ se coupent exactement si la relation suivante est satisfaite :

$$\mathbf{L}_2^\mathsf{T} \begin{pmatrix} 0 & \mathsf{I} \\ \mathsf{I} & 0 \end{pmatrix} \mathbf{L}_1 = \mathbf{a}_2^\mathsf{T}\mathbf{b}_1 + \mathbf{b}_2^\mathsf{T}\mathbf{a}_1 = 0 \tag{2}$$

Interprétation euclidienne des coordonnées de Plücker. Si les points $\mathbf{A}$ et $\mathbf{B}$ sont donnés en coordonnées affines ($A_4 = B_4 = 1$), les vecteurs $\mathbf{a}$ et $\mathbf{b}$ peuvent s'interpréter comme suit. Notons d'abord $\bar{\mathbf{A}}^\mathsf{T} = (A_1, A_2, A_3)$ et $\bar{\mathbf{B}}^\mathsf{T} = (B_1, B_2, B_3)$. Nous avons alors $\mathbf{a} = \bar{\mathbf{B}} - \bar{\mathbf{A}}$ et $\mathbf{b} = \bar{\mathbf{B}} \times \bar{\mathbf{A}}$. Le vecteur $\mathbf{a}$ est donc le vecteur directeur de la droite. Quant à $\mathbf{b}$, il est orthogonal au plan engendré par l'origine et la droite. Finalement, la distance carrée de la droite de l'origine, c'est-à-dire la distance carrée du point sur la droite le plus proche de l'origine, est donnée par :

$$d^2 = \frac{\mathbf{b}^\mathsf{T}\mathbf{b}}{\mathbf{a}^\mathsf{T}\mathbf{a}}$$

3 Une hiérarchie de modèles de caméras

Une **caméra non centrale** peut avoir des rayons de projections quelconques, tandis que pour une **caméra centrale**, il existe un point – le **centre optique** – qui se trouve sur tous les rayons de projection. Un cas intermédiaire est ce que nous appelons ici celui des **caméras axiales** : il existe une droite qui touche tous les rayons de projection. Nous l'appelons l'**axe de la caméra** (à ne pas confondre avec l'axe optique du modèle perspectif). Des exemples de caméras qui se trouvent dans cette classe sont :

- les caméras dites de type cross-slit [22, 35] (ou bien, *x-slit* ou *two-slit*), et le cas particulier des caméra dites push-broom linéaire [14]. Ces caméras forment en effet une sous-classe des caméras axiales, comme il l'est expliqué plus bas ;
- des systèmes stéréo consistant de deux caméras centrales ou de plusieurs caméras centrales avec des centres optiques collinéaires ;

TAB. 1 – Classes de caméras, définies par des configurations de points ou droites en 3-D qui touchent tous les rayons de projection.

Points/droites touchant les rayons	Description de la classe
Aucun	Caméra non centrale
1 point	Caméra centrale
2 points	Caméra ayant un seul rayon de projection
1 droite	Caméra axiale
1 point, 1 droite	Caméra 1-D centrale
2 droites qui ne se coupent pas	Caméra de type cross-slit
2 droites coplanaires	Union d'une caméra centrale et d'une caméra 1-D non centrale
3 droites coplanaires sans point d'intersection commun	Caméra 1-D non centrale

– certaines caméras catadioptriques non centrales : si le miroir est une surface de révolution est si le centre optique de la caméra centrale qui le regarde (pas nécessairement une caméra perspective) se trouve sur l'axe de révolution, il s'agit d'une caméra axiale. Il est facile de vérifier que dans ce cas, tous les rayons de projection réfléchis par le miroir, coupent l'axe de révolution, qui joue donc le rôle de l'axe de la caméra axiale. Remarquons qu'une caméra catadioptrique avec un miroir sphérique est obligatoirement non centrale, et de type axial. Un autre exemple est celui des systèmes basés sur un miroir conique (avec centre optique sur l'axe du cône) [7]. Le modèle axial peut aussi servir à modéliser des systèmes catadioptriques qui, à cause d'un mauvais alignement entre miroir et caméra, ne produisent pas une projection centrale.

Ces trois classes de modèles de caméras peuvent aussi être définies ainsi : existence d'un espace linéaire de dimension d qui a une intersection non vide avec chacun des rayons de projection. Avec cette définition, $d = 0$ correspond aux caméras centrales, $d = 1$ aux caméras axiales et $d = 2$ aux caméras complètement non centrales.

Des classes intermédiaires existent. Les caméras de type cross-slit, déjà mentionnées, sont un cas spécial des caméras axiales : il existe deux droites qui toutes deux touchent tous les rayons de la caméra. Similairement, on peut définir des caméras 1-D centrales (caméras ayant une seule ligne de pixels) par un point et une droite en 3-D qui touchent chacun des rayons. Le tableau 1 résume des modèles de caméras définis de cette manière, dont certains n'ont bien sur aucun intérêt pratique. Une approche similaire pour la définition de classes de caméras a été explorée dans [22] ; le but de cette approche était de trouver *une seule* primitive géométrique qui touche tous les rayons de projection et qui puisse donc servir à la définition d'une classe de caméras. Cette approche est moins générale et moins intuitive que celle adoptée ici.

Il est intéressant de considérer certaines de ces classes de caméras plus en détail, grâce à l'observation suivante. Beaucoup d'algorithmes existants de calibrage ou d'estimation du mouvement procèdent typiquement en deux étapes : (1) estimation d'une matrice ou d'un tenseur en résolvant des systèmes d'équations linéaires (par exemple, la matrice de projection [1], la matrice essentielle [25], les tenseur trifocaux [15], les tenseur de calibrage [31], etc.) ; (2) ensuite, les paramètres recherchés (paramètres intrinsèques, matrice de rotation, etc.) sont extraits de ces matrices ou tenseurs. Il y a deux problèmes intrinsèques :

– si un algorithme qui a été développé pour une classe de caméras est appliqué à une classe de

caméras différente, l'étape (1) ne donnera pas de solution. Par exemple, l'estimation d'une matrice fondamentale perspective (matrice 3×3) à partir de correspondances de points issues de deux caméras catadioptriques, ne donnera évidemment aucun résultat exploitable ;

– ce premier problème est évident ; un problème plus subtil est le suivant. Si un algorithme qui a été développé pour une classe de caméras est appliqué à une caméra appartenant à une sous-classe, la résolution du système linéaire dans l'étape (1) n'aura pas de solution unique ; il y aura en effet une infinité de solutions. L'étape (2), si effectuée avec une de ces solutions choisie au hasard, donnera un résultat incorrect. Un exemple simple pour illustrer ce problème est l'estimation de la matrice fondamentale perspective, à partir de correspondances de points d'une scène qui ne contient qu'un plan : l'estimation est sous-contrainte. Un exemple plus proche de nos préoccupations est le suivant : si l'on tente d'estimer la matrice essentielle du modèle de caméras *non central* (une matrice 6×6 [25]) à partir de correspondances obtenues de caméras *centrales* ou *axiales*, alors le système d'équations linéaires associé n'aura pas de solution unique.

Ce deuxième problème nous amène à définir la géométrie d'images multiples pour le modèle le plus général d'abord, puis à la spécialiser à des sous-classes. Dans cet article, nous traitons les caméras centrales, axiales, de type cross-slit et le modèle complètement non central.

4 Paramétrisations

La géométrie d'images multiples sera formulée en utilisant les coordonnées de Plücker des rayons de projection. Les rayons de projection de toutes les classes de caméras sauf de la plus générale, appartiennent à des sous-ensembles particuliers des droites en 3-D, cf. la section précédente. Nous pouvons alors tenter de choisir un système de coordonnées local à une caméra tel que les vecteurs de coordonnées de ses rayons aient une forme particulière, menant à des tenseurs d'appariement de forme simplifiée. Comme nous considérons des caméras calibrées, les rayons sont donnés dans un repère métrique et nous pouvons alors appliquer des rotations et translations pour en choisir un de « sympathique ». Les transformations et paramétrisations de rayons de projection appropriées pour différents modèles de caméras sont expliquées dans la suite.

4.1 Caméras centrales

Tous les rayons passent par un même point, le centre optique. Nous distinguons les cas où le centre optique est un point fini ou à l'infini.

Centre optique fini. Nous choisissons un repère local avec comme origine le centre optique. Ainsi, pour tous les rayons de projection le vecteur $\mathbf{b}$ (cf. la section 2) est nul, c'est-à-dire que les rayons ont des coordonnées de la forme :

$$\mathbf{L} = \begin{pmatrix} \mathbf{a} \\ \mathbf{0} \end{pmatrix}$$

Notons que ceci est en accord avec le fait que les tenseurs d'appariement des caméras perspectives ont une « taille de base » de 3 : la matrice fondamentale par exemple, est une matrice 3×3.

Centre optique infini. Dans ce cas (par exemple, une caméra orthographique), nous ne pouvons pas choisir le centre optique comme origine. A la place, nous orientons le repère tel que le centre optique ait les coordonnées $(0,0,1,0)^\mathsf{T}$. Les rayons de projection sont alors de la forme :

$$\mathbf{L} = \begin{pmatrix} 0 & 0 & a_3 & b_1 & b_2 & 0 \end{pmatrix}$$

4.2 Caméras axiales

Tous les rayons touchent une droite particulière, l'axe de la caméra. Nous distinguons les cas d'un axe étant une droite à l'infini ou non.

Axe fini. Nous choisissons un repère où l'axe de la caméra coïncide avec l'axe des Z. Pour les rayons de projection nous avons alors $L_6 = b_3 = 0$:

$$\mathbf{L} = \begin{pmatrix} \mathbf{a} \\ b_1 \\ b_2 \\ 0 \end{pmatrix}$$

Axe infini. Choisissons un repère où l'axe est la droite à l'infini avec les coordonnées $(1,0,0)^\mathsf{T}$ (coordonnées d'une droite sur le plan à l'infini). Ceci correspond aux coordonnées de Plücker $(0,0,0,1,0,0)^\mathsf{T}$. Les rayons de projection ont donc des coordonnées avec $L_1 = a_1 = 0$ (ceci est basé sur l'équation (2)).

$$\mathbf{L} = \begin{pmatrix} 0 \\ a_2 \\ a_3 \\ \mathbf{b} \end{pmatrix}$$

Pour les deux cas, nous voyons que les rayons de projection d'une caméra axiale peuvent être représentés par 5 coordonnées de Plücker. Les tenseurs d'appariement auront alors une taille de base de 5 ; par exemple, la matrice essentielle pour des caméras axiales sera de dimension 5×5 (voir plus loin).

4.3 Caméras de type cross-slit

Comme il a été mentionné, les caméras de type cross-slit sont définies par l'existence de deux droites qui coupent tous les rayons de projection. Le cas où ces deux axes se coupent, c'est-à-dire sont coplanaires, n'a pas d'intérêt ici (voir le tableau 1). Deux cas sont alors possibles : (i) tous les deux axes sont des droites finies ou (ii) exactement un des deux axes est une droite à l'infini. Il y a forcément au moins un axe fini ; nous adoptons un repère comme il a été décrit ci-dessus pour les caméras axiales avec axe fini. Ceci nous laisse encore des degrés de liberté dans le choix du repère, ce qui sera exploité pour obtenir des coordonnées simples pour le deuxième axe.

Deux axes finis. Ayant fixé le premier axe, nous avons toujours la possibilité d'effectuer des rotations autour, ou des translations le long de cet axe. Nous pouvons alors choisir un repère où

le deuxième axe de la caméra coupe l'axe des Y et est parallèle au plan $X-Z$. Ainsi, il contient deux points avec des coordonnées de la forme :

$$\mathbf{A} = \begin{pmatrix} 0 \\ Y \\ 0 \\ 1 \end{pmatrix} \qquad \mathbf{B} = \begin{pmatrix} X \\ 0 \\ Z \\ 0 \end{pmatrix}$$

Ses coordonnées de Plücker sont alors données par :

$$\mathbf{L}_2^\mathsf{T} = \begin{pmatrix} X & 0 & Z & -YZ & 0 & YZ \end{pmatrix}$$

Les rayons de projection coupent les deux axes et doivent alors être de la forme :

$$\mathbf{L}^\mathsf{T} = \begin{pmatrix} a_1 & a_2 & a_3 & \left(\frac{YZ}{X}a_1 - Ya_3\right) & b_2 & 0 \end{pmatrix}$$

Nous divisons par X, ce qui est permis ici puisque $X \neq 0$ (sinon, le deuxième axe serait parallèle au premier, donc coplanaire, ce qui est exclu ici). Remplaçons ensuite $\frac{YZ}{X}$ par W. Les rayons de projection peuvent alors être paramétrés par les 4 coefficients a_1, a_2, a_3, b_2 (qui sont définis à une échelle près) :

$$\mathbf{L} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ W & 0 & -Y & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ b_2 \end{pmatrix}$$

Les coefficients W et Y sont connus et identiques pour tous les rayons de projection (ils représentent la position relative des deux axes de caméra, qui est connue puisque la caméra est supposée être calibrée).

Un axe fini et un axe infini. Comme il a été dit ci-dessus, nous fixons d'abord l'axe fini comme pour les caméras axiales, puis pouvons encore effectuer des rotations autour ou des translations le long de cet axe pour obtenir des coordonnées particulières pour l'axe infini. Les translations n'ont pas d'effet sur les coordonnées de l'axe infini ; quant à la rotation, nous pouvons la choisir telle que l'axe infini ait les coordonnées $(0, \cos\Theta, \sin\Theta)^\mathsf{T}$ (coordonnées homogènes d'une droite à l'infini), pour un certain Θ. Ceci correspond aux coordonnées de Plücker :

$$\mathbf{L}_2^\mathsf{T} = \begin{pmatrix} 0 & 0 & 0 & 0 & \cos\Theta & \sin\Theta \end{pmatrix}$$

Les rayons de projection doivent couper les deux axes, donc sont de la forme :

$$\mathbf{L}^\mathsf{T} = \begin{pmatrix} a_1 & -a_3\tan\Theta & a_3 & b_1 & b_2 & 0 \end{pmatrix}$$

Définissons $W = -\tan\Theta$. Les rayons de projection peuvent alors être représentés par 4 coefficients a_1, a_3, b_1, b_2 (définis à l'échelle près) :

$$\mathbf{L} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & W & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} a_1 \\ a_3 \\ b_1 \\ b_2 \end{pmatrix}$$

Les tenseurs d'appariement pour des caméras de type cross-slit auront alors une taille de base de 4.

4.4 Caméras non centrales générales

Aucune simplification des coordonnées des rayons de projection n'est possible ; ce seront alors des coordonnées de Plücker générales. Les tenseurs d'appariement auront une taille de base de 6.

5 Géométrie d'images multiples

Nous établissons les fondements d'une géométrie d'images multiples pour des caméras générales (non centrales). Elle sera incarnée, comme avec les caméras perspectives, par des tenseurs d'appariement. Nous montrons comment les obtenir, de manière analogue au cas perspectif.

Ici, nous ne traitons que du cas calibré ; le cas non calibré n'est bien géré que pour les caméras perspectives, puisque ces caméras, calibrées ou non, sont liées par des transformations projectives. Pourtant, quant aux caméras non centrales, un tel lien n'existe pas en général : dans le cas le plus général, chaque paire pixel+rayon de projection peut être complètement indépendante de toutes les autres paires.

Dans la suite, nous rappelons d'abord la notion de tenseur d'appariement, puis les principes d'une approche de géométrie d'image multiples pour les caméras perspectives. Cette approche travaille avec des coordonnées de points dans les images. Nous appliquons ensuite les mêmes idées, tout en travaillant avec des coordonnées de droites en 3-D, pour dériver notre géométrie des caméras non centrales.

5.1 Tenseurs d'appariement

Il n'existe pas de véritable définition de la notion de tenseur d'appariement dans la littérature. Néanmoins, il est entendu qu'il s'agit, pour les caméras perspectives, de tenseurs dont les coefficients dépendent des matrices de projection d'un ensemble de vues considérées et qui permettent de formuler des contraintes d'appariement pour des primitives géométriques dans ces vues. Il est généralement sous-entendu que ces contraintes sont de forme multi-linéaire, c'est-à-dire que les contraintes sont linéaires en les coordonnées de chaque primitive. L'exemple le plus connu est le tenseur d'appariement bi-focal, ou bien la matrice fondamentale F, qui donne la contrainte d'appariement bi-linéaire classique :

$$\mathbf{q}_2^\mathsf{T} \mathsf{F} \mathbf{q}_1 = 0$$

Plus généralement, pour n vues et des points $\mathbf{q}_1 \cdots \mathbf{q}_n$ dans ces vues, on espère trouver des tenseurs d'appariement T de dimension $3 \times \cdots \times 3$ qui permettent de formuler des contraintes

d'appariement multi-linéaires :

$$\sum_{i_1=1}^{3}\sum_{i_2=1}^{3}\cdots\sum_{i_n=1}^{3} q_{1,i_1}q_{2,i_2}\cdots q_{n,i_n}T_{i_1,i_2,\cdots,i_n} = 0 \tag{3}$$

Ceci est décrit plus en détail dans la section suivante.

Ici, nous ne traitons que des tenseurs d'appariement pour n points, mais il en existe également pour l'appariement de droites ou d'un mélange de droites et de points [15].

Notons aussi que les tenseurs d'appariement ont deux applications principales :

1. ils permettent d'établir des contraintes d'appariement, voir ci-dessus ;
2. réciproquement, des appariements entre images permettent de calculer les tenseurs. Puisqu'ils dépendent des matrices de projection, on peut espérer de remonter à ces dernières, donc faire de l'estimation du mouvement de caméra, de l'auto-calibrage, de la reconstruction 3-D etc. Il existe un éventail assez large de telles applications et nous renvoyons à [15] pour un tour d'horizon assez complet.

Le but principal de cet article est de montrer l'existence et la forme de tenseurs d'appariement pour nos modèles de caméras généraux. Plus concrètement, nous recherchons des contraintes d'appariement qui seraient de la forme :

$$\sum_{i_1=1}^{6}\sum_{i_2=1}^{6}\cdots\sum_{i_n=1}^{6} L_{1,i_1}L_{2,i_2}\cdots L_{n,i_n}T_{i_1,i_2,\cdots,i_n} = 0 \tag{4}$$

où les $\mathbf{L}$ sont les coordonnées de Plücker de rayons de projection, ou bien des vecteurs de coordonnées tels que définis section 4. Dans le cas le plus général, les tenseurs d'appariement seraient alors de dimension $6\times\cdots\times 6$. La contrainte (4) exprimerais que les rayons de projection considérés peuvent se correspondre, c'est-à-dire que les pixels associés à ces rayons peuvent être un appariement potentiel. Ces concepts seront introduits dans la section 5.3.

5.2 Rappels sur la géométrie d'images multiples des caméras perspectives

Nous rappelons des principes de l'une des approches pour dériver les relations entre images multiples, développée pour les caméras perspectives [8]. Soient P_i des matrices de projection et $\mathbf{q}_i$ des points image. Un ensemble de points image peut constituer une correspondance uniquement s'il existe un point 3-D $\mathbf{Q}$ et des facteurs d'échelle scalaires λ_i tels que, pour tout i :

$$\lambda_i\mathbf{q}_i = \mathsf{P}_i\mathbf{Q}$$

Ceci peut être écrit sous la forme d'une équation matricielle :

$$\underbrace{\begin{pmatrix} \mathsf{P}_1 & \mathbf{q}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathsf{P}_2 & \mathbf{0} & \mathbf{q}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathsf{P}_n & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{q}_n \end{pmatrix}}_{\mathsf{M}} \begin{pmatrix} \mathbf{Q} \\ -\lambda_1 \\ -\lambda_2 \\ \vdots \\ -\lambda_n \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

Cette équation exprime que la matrice M, de dimension $3n \times (4+n)$, possède un vecteur nul, c'est-à-dire qu'elle est de rang inférieur à $4+n$. Par conséquent, les déterminants de toutes ses sous-matrices de dimension $(4+n) \times (4+n)$ valent zéro. Ces déterminants sont des expressions multi-linéaires en termes des coordonnées des points image $\mathbf{q}_i$.

Il faut les développer pour toute sous-matrice possible de la bonne dimension. Notons que seule une sous-matrice contenant deux lignes ou plus associées à chacune des vues, peut donner une contrainte qui lie toutes les matrices de projection. Ainsi, des contraintes peuvent être obtenues pour n vues avec $2n \leq 4+n$, ce qui implique que des contraintes multi-vues (et multi-linéaires) n'existent que jusqu'à 4 vues.

Les contraintes pour n vues sont de la forme (3), où T désigne donc un tenseur de dimension $3 \times \cdots \times 3$, appelé tenseur d'appariement. Les tenseurs dépendent uniquement des matrices de projection P_i et constituent en effet une représentation compacte de celles-ci (représentation qui permet d'extraire les P_i à une transformation projective près).

Notons que dès que l'on considère des caméras calibrées, cette théorie peut s'appliquer à n'importe quelle caméra *centrale* (en plus des caméras perspectives) : des contraintes comme ci-dessus peuvent par exemple être écrites pour des caméras avec distorsion radiale, en termes des coordonnées de points image corrigés.

5.3 Géométrie d'images multiples pour les caméras non centrales

Ici, nous traitons avec des matrices de pose au lieu de matrices de projection (qui, elles, dépendent et de la pose et du calibrage) :

$$\mathsf{P}_i = \begin{pmatrix} \mathsf{R}_i & \mathbf{t}_i \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix}$$

Ces transformations euclidiennes représentent des déplacements d'une caméra, ou bien le changement de repère d'un repère global vers le repère local de la caméra. Les points image du paragraphe précédent sont maintenant remplacés par des rayons de projection. Soit le i^e rayon représenté par deux points 3-D $\mathbf{A}_i$ et $\mathbf{B}_i$. Ultérieurement, nous voulons aboutir à des expressions en termes des coordonnées de Plücker des rayons, c'est-à-dire des tenseurs T et des contraintes d'appariement de la même forme que (3), mais avec des tenseurs de dimension $6 \times \cdots \times 6$, qui agissent sur des coordonnées de Plücker, voir l'équation (4) ci-dessus. Dans la suite, nous expliquons comment obtenir ces contraintes.

Considérons un ensemble de n rayons de projection, un pour chaque vue, qui chacun est représenté par deux points $\mathbf{A}_i$ et $\mathbf{B}_i$. Le choix des points sur une droite n'a aucune importance, comme plus tard nous arriverons à des expressions en termes des coordonnées de Plücker.

Les n rayons peuvent être une correspondance s'il existe un point 3-D $\mathbf{Q}$ et des facteurs d'échelle λ_i et μ_i tels que, pour chaque vue $i = 1 \ldots n$:

$$\lambda_i \mathbf{A}_i + \mu_i \mathbf{B}_i = \mathsf{P}_i \mathbf{Q}$$

Cette équation exprime en effet que le point $\mathbf{Q}$ se trouve sur le i^e rayon de projection (le tout exprimé dans le repère local de la i^e caméra).

Comme pour les caméras perspectives, nous regroupons les équations de toutes les vues en une équation matricielle :

$$\underbrace{\begin{pmatrix} \mathsf{P}_1 & \mathbf{A}_1 & \mathbf{B}_1 & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ \mathsf{P}_2 & \mathbf{0} & \mathbf{0} & \mathbf{A}_2 & \mathbf{B}_2 & \cdots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathsf{P}_n & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{A}_n & \mathbf{B}_n \end{pmatrix}}_{\mathsf{M}} \begin{pmatrix} \mathbf{Q} \\ -\lambda_1 \\ -\mu_1 \\ -\lambda_2 \\ -\mu_2 \\ \vdots \\ -\lambda_n \\ -\mu_n \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{pmatrix}$$

Cette équation implique que M, de dimension $4n \times (4+2n)$, n'est pas de rang plein. Par conséquent, toutes ses sous-matrices de dimension $(4+2n) \times (4+2n)$ doivent être singulières, c'est-à-dire avoir un déterminant nul. Dans la suite, nous montrons que seules les sous-matrices qui contiennent au moins trois lignes associées à chacune des vues, mènent à des contraintes entre toutes les matrices de pose P_i. Ceci veut dire que nous aurons des contraintes pour n vues si $3n \leq 4+2n$, donc $n \leq 4$ comme pour les caméras perspectives.

Regardons pour commencer le cas d'une sous-matrice de M qui contient, pour une des vues, une seule ligne associée. Sans perte de généralité, supposons que ce soit le cas pour la première vue et que seule la première ligne associée à cette vue soit présente dans la sous-matrice de M. Cette sous-matrice sera donc de la forme :

$$\mathsf{N} = \begin{pmatrix} \mathbf{P}_{11}^{\mathsf{T}} & A_{11} & B_{11} & 0 & 0 & \cdots & 0 & 0 \\ \mathsf{P}_2 & \mathbf{0} & \mathbf{0} & \mathbf{A}_2 & \mathbf{B}_2 & \cdots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathsf{P}_n & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{A}_n & \mathbf{B}_n \end{pmatrix}$$

Ici, $\mathbf{P}_{11}^{\mathsf{T}}$ désigne la première ligne de la matrice P_1. Les deux colonnes contenant les scalaires A_{11} et B_{11} ne contiennent sinon que des zéros. L'une est donc le multiple de l'autre, ce qui implique que le déterminant de N vaut *toujours* zéro. Il n'y a donc pas de contrainte exploitable.

Dans la suite, nous excluons ce cas, c'est-à-dire que nous ne considérons que des sous-matrices de M avec au moins deux lignes associées à chacune des vues. Soit N une telle matrice. Sans perte de généralité, nous commençons le développement de son déterminant avec la colonne contenant $\mathbf{A}_1$ et $\mathbf{B}_1$. Le déterminant sera alors donné comme une somme de termes de la forme :

$$(A_{1,j}B_{1,k} - A_{1,k}B_{1,j}) \det \bar{\mathsf{N}}_{jk}$$

où $j,k \in \{1..4\}$, $j \neq k$ et $\bar{\mathsf{N}}_{jk}$ est obtenue de N en omettant les colonnes contenant $\mathbf{A}_1$ et $\mathbf{B}_1$ ainsi que les lignes contenant $A_{1,j}$ et $A_{1,k}$.

Nous observons différentes choses :

– Le terme $(A_{1,j}B_{1,k} - A_{1,k}B_{1,j})$ représente en effet une des coordonnées de Plücker du rayon de la première vue, engendré par $\mathbf{A}_1$ et $\mathbf{B}_1$ (cf. la section 2). En continuant avec le développement du déterminant de $\bar{\mathsf{N}}_{jk}$ et ainsi de suite, le déterminant de N pourra s'écrire sous la forme :

$$\sum_{i_1=1}^{6} \sum_{i_2=1}^{6} \cdots \sum_{i_n=1}^{6} L_{1,i_1} L_{2,i_2} \cdots L_{n,i_n} T_{i_1,i_2,\cdots,i_n} = 0$$

où les $L_{i,j}$ sont des coordonnées de Plücker des rayons considérés, obtenues à partir des $\mathbf{A}_i$

TAB. 2 – Les cas utiles de contraintes d'appariement pour des caméras centrales et non centrales. Les colonnes sur-titrées par « cas utiles » contiennent des entrées de la forme $x - y - z$ etc. Ceci désigne la constitution des sous-matrices de M qui donnent lieu à des contraintes entre toutes les vues : $x - y - z$ par exemple veut dire que les sous-matrices contiennent x lignes associées à une vue, y à une autre, etc.

	central		non-central	
nombre de vues	M	cas utiles	M	cas utiles
2	6×6	3-3	8×8	4-4
3	9×7	3-2-2	12×10	4-3-3
4	12×8	2-2-2-2	16×12	3-3-3-3

et $\mathbf{B}_i$. Les coefficients des matrices de pose P_i sont, eux, regroupés dans un tenseur T, de dimension $6 \times \cdots \times 6$.

– Si N ne contient que deux lignes associées à la première vue, alors son déterminant sera de la forme :

$$L_{1,x} \left(\sum_{i_2=1}^{6} \cdots \sum_{i_n=1}^{6} L_{2,i_2} \cdots L_{n,i_n} T_{i_2,\cdots,i_n} \right) = 0$$

c'est-à-dire qu'une seule coordonnée du premier rayon sera présente dans l'expression. Cette contrainte ne lie pas toutes les vues entre elles : elle est vérifiée si $L_{1,x} = 0$, ce qui est une condition indépendante des autres vues ou bien si l'expression entre parenthèses vaut zéro, ce qui ne dépend que des vues 2 à n.

Ceci explique ce que nous avons constaté plus haut : pour obtenir des contraintes entre toutes les vues, seules des sous-matrices contenant au moins trois lignes pour chacune des vues sont utiles.

Nous sommes maintenant prêts à établir les différents cas qui mènent à des contraintes entre images. Comme il a été dit, aucune contrainte (multi-linéaire) n'existe qui lierait plus de quatre vues à la fois. Nous résumons alors les cas utiles, de deux à quatre vues, dans le tableau 2, pour les caméras centrales (basé sur la théorie développée pour les caméras perspectives) et non centrales. Le tableau donne les dimensions des sous-matrices de M, dont le déterminant donne des contraintes d'appariement, représentées par des matrices essentielles (pour deux vues), des tenseurs trifocaux ou quadrifocaux. L'écriture détaillée des tenseurs devient alors une tache plutôt « mécanique ».

6 Le cas de deux vues

Nous avons jusqu'alors expliqué comment formuler des contraintes d'appariement entre deux, trois ou quatre caméras non centrales, représentées par des tenseurs de dimension 6×6 à $6 \times 6 \times 6 \times 6$. Afin de rendre ces résultats plus concrets, nous explorons maintenant en détails le cas de deux vues. Nous montrons comment le tenseur bifocal, ou matrice essentielle, dépend des paramètres de pose (ou bien, du mouvement). Ceci est d'abord fait pour les caméras non centrales générales, puis spécialisé aux caméras axiales, de type cross-slit et finalement aux caméras centrales. A la fin de cette section, nous donnons quelques commentaires sur l'estimation des matrices essentielles, introduites dans la suite.

6.1 Caméras non centrales

Pour plus de simplicité, et sans perte de généralité, nous supposons ici que le repère global coïncide avec le repère de la première caméra. La matrice de pose P_1 est donc l'identité. Quant à la pose de la deuxième caméra, nous omettons alors les indices ; elle sera donc représentée par une matrice de rotation R et un vecteur de translation $\mathbf{t}$. La matrice M devient alors :

$$\mathsf{M} = \begin{pmatrix} 1 & 0 & 0 & 0 & A_{1,1} & B_{1,1} & 0 & 0 \\ 0 & 1 & 0 & 0 & A_{1,2} & B_{1,2} & 0 & 0 \\ 0 & 0 & 1 & 0 & A_{1,3} & B_{1,3} & 0 & 0 \\ 0 & 0 & 0 & 1 & A_{1,4} & B_{1,4} & 0 & 0 \\ R_{11} & R_{12} & R_{13} & t_1 & 0 & 0 & A_{2,1} & B_{2,1} \\ R_{21} & R_{22} & R_{23} & t_2 & 0 & 0 & A_{2,2} & B_{2,2} \\ R_{31} & R_{32} & R_{33} & t_3 & 0 & 0 & A_{2,3} & B_{2,3} \\ 0 & 0 & 0 & 1 & 0 & 0 & A_{2,4} & B_{2,4} \end{pmatrix}$$

Pour toute paire de rayons correspondants, M doit être singulière. Comme dans le cas présent de deux vues M est carrée, ceci implique que son déterminant vaut zéro. En le développant d'après les indications de la section précédente, nous obtenons la contrainte :

$$\mathbf{L}_2^{\mathsf{T}} \begin{pmatrix} -[\mathbf{t}]_\times \mathsf{R} & \mathsf{R} \\ \mathsf{R} & \mathsf{0} \end{pmatrix} \mathbf{L}_1 = 0 \tag{5}$$

où les coordonnées de Plücker $\mathbf{L}_1$ et $\mathbf{L}_2$ sont définies selon (1).

Nous pouvons identifier la matrice essentielle, donnée dans [25] :

$$\mathsf{E}_n = \begin{pmatrix} -[\mathbf{t}]_\times \mathsf{R} & \mathsf{R} \\ \mathsf{R} & \mathsf{0} \end{pmatrix} \tag{6}$$

L'équation (5) représente en effet une contrainte épipolaire, tout à fait similaire à celle des caméras perspectives.

6.2 Caméras axiales

Axe fini. Comme il a été expliqué dans la section 3, nous adoptons des repères locaux pour nos caméras où les rayons ont une coordonnée $L_6 = 0$. Ainsi, la contrainte épipolaire (5) peut être exprimée par une matrice essentielle amputée, de dimension 5×5, qui agit sur des vecteurs de coordonnées de Plücker amputés de leur sixième coordonnée :

$$\mathsf{E}_{af} = \begin{pmatrix} -[\mathbf{t}]_\times \mathsf{R} & \begin{pmatrix} R_{11} & R_{12} \\ R_{21} & R_{22} \\ R_{31} & R_{32} \end{pmatrix} \\ \begin{pmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \end{pmatrix} & \mathsf{0}_{2\times 2} \end{pmatrix} \tag{7}$$

Cette matrice essentielle est obtenue à partir de E_n (cf. (6)) en omettant les sixièmes ligne et colonne.

Remarquons que E_{af} est en général de rang plein, c'est-à-dire de rang 5. Elle est singulière exactement si les axes des deux caméras se coupent. Dans ce cas-ci, les vecteurs nuls gauche et

droit de E_{af} donnent les coordonnées des deux axes, exprimées dans le repère de l'autre caméra respectivement (pour obtenir les coordonnées de Plücker, il faut juste rajouter un zéro comme sixième coefficient).

Axe infini. Comme ci-dessus, la contrainte épipolaire (5) se simplifie, et est représentée par une matrice essentielle de dimension 5×5 :

$$\begin{pmatrix} \begin{pmatrix} -t_3 & 0 & t_1 \\ t_2 & -t_1 & 0 \end{pmatrix} \begin{pmatrix} R_{12} & R_{13} \\ R_{22} & R_{23} \\ R_{32} & R_{33} \end{pmatrix} & \begin{pmatrix} R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{pmatrix} \\ \begin{pmatrix} R_{12} & R_{13} \\ R_{22} & R_{23} \\ R_{32} & R_{33} \end{pmatrix} & \mathsf{0} \end{pmatrix}$$

En détail, la matrice essentielle s'écrit :

$$\mathsf{E}_{ai} = \begin{pmatrix} t_1R_{32} - t_3R_{12} & t_1R_{33} - t_3R_{13} & R_{21} & R_{22} & R_{23} \\ t_2R_{12} - t_1R_{22} & t_2R_{13} - t_1R_{23} & R_{31} & R_{32} & R_{33} \\ R_{12} & R_{13} & 0 & 0 & 0 \\ R_{22} & R_{23} & 0 & 0 & 0 \\ R_{32} & R_{33} & 0 & 0 & 0 \end{pmatrix}$$

Elle est toujours singulière (les axes des deux caméras sont des droites à l'infini, donc se coupent forcément). Son vecteur nul de droite est $(0, 0, R_{11}, R_{12}, R_{13})^\mathsf{T}$. Il représente l'axe de la deuxième caméra, exprimé par rapport au repère de la première caméra. Réciproquement, le vecteur nul de gauche est $(0, 0, R_{11}, R_{21}, R_{31})^\mathsf{T}$, ce qui représente l'axe de la première caméra, dans le repère de la deuxième.

6.3 Caméras de type cross-slit

Nous considérons les deux cas expliqués dans la section 4.3.

Deux axes finis. La contrainte épipolaire (5) se simplifie et peut s'écrire basée sur une matrice essentielle de dimension 4×4, qui agit sur des vecteurs de coordonnées de Plücker amputés de la forme $(a_1, a_2, a_3, b_2)^\mathsf{T}$ (cf. la section 4.3) :

$$\mathsf{E}_{xff} = \begin{pmatrix} 1 & 0 & 0 & W_2 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -Y_2 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \mathsf{E}_{af} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ W_1 & 0 & -Y_1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

où E_{af} est donnée dans (7). Contrairement aux cas précédents, la matrice essentielle contient maintenant non seulement des paramètres de pose, mais aussi des « paramètres intrinsèques » (les coefficients W_i et Y_i des deuxièmes axes des caméras). En détail, elle peut s'écrire :

$$\mathsf{E}_{xff} = \begin{pmatrix} -[\mathbf{t}]_\times \mathsf{R} & \mathbf{0} \\ \mathbf{0}^\mathsf{T} & 0 \end{pmatrix} + \begin{pmatrix} R_{11}(W_1+W_2) & R_{12}W_2 & R_{13}W_2 - R_{11}Y_1 & R_{12} \\ R_{21}W_1 & 0 & -R_{21}Y_1 & R_{22} \\ R_{31}W_1 - R_{11}Y_2 & -R_{12}Y_2 & -R_{13}Y_2 - R_{31}Y_1 & R_{32} \\ R_{21} & R_{22} & R_{23} & 0 \end{pmatrix}$$

Un axe fini et un axe infini. Similairement, la contrainte épipolaire se simplifie et nous obtenons une matrice essentielle de dimension 4×4 :

$$\mathsf{E}_{xfi} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & W_2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \mathsf{E}_{af} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & W_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

6.4 Caméras centrales

Centre optique fini. Les rayons de projection sont ici de la forme $(L_1, L_2, L_3, 0, 0, 0)^\mathsf{T}$ (voir la section 3). La contrainte épipolaire (5) se réduit donc à l'expression :

$$\begin{pmatrix} L_{2,1} & L_{2,2} & L_{2,3} \end{pmatrix} \left(-[\mathbf{t}]_\times \mathsf{R}\right) \begin{pmatrix} L_{1,1} \\ L_{1,2} \\ L_{1,3} \end{pmatrix} = 0$$

où nous retrouvons la matrice essentielle « classique » de dimension 3×3 : $\mathsf{E}_{cf} = -[\mathbf{t}]_\times \mathsf{R}$ [15, 17].

Centre optique infini. Dans ce cas, la matrice essentielle est donnée par :

$$\mathsf{E}_{ci} = \begin{pmatrix} t_2R_{13} - t_1R_{23} & R_{31} & R_{32} \\ R_{13} & 0 & 0 \\ R_{23} & 0 & 0 \end{pmatrix}$$

Ceci ressemble à la matrice fondamentale affine [27], mais ne lui correspond pas exactement : ici, la matrice essentielle agit sur des droites en 3-D, tandis que la matrice fondamentale agit sur des points image. Par exemple, le vecteur nul de droite de E_{ci} est $(0, R_{32}, -R_{31})^\mathsf{T}$, ce qui représente la droite en 3-D avec les coordonnées de Plücker $(0, 0, 0, R_{32}, -R_{31}, 0)^\mathsf{T}$. C'est la droite engendrée par les centres optiques des deux caméras (une droite à l'infini), exprimée par rapport au repère de la première caméra.

6.5 Estimation des matrices essentielles

Cet article a une vocation théorique, mais nous voulons néanmoins donner quelques éléments utiles pour l'estimation numérique des matrices essentielles introduites ainsi que pour l'extraction des paramètres de mouvement à partir de celles-ci. Pour les caméras perspectives, deux types majeurs d'algorithmes pour l'estimation de la matrice essentielle et du mouvement ont été développés. La première approche consiste à utiliser le moins de correspondances possible. Pour la matrice essentielle perspective, il s'agit de 5 correspondances de points, et seulement très récemment, un

algorithme vraiment minimal a été trouvé [19]. Ce type d'algorithme correspond à la résolution d'équations non linéaires, donnant un nombre fini de solutions.

L'autre type d'approche consiste en un premier lieu à ignorer la structure de la matrice essentielle (c'est-à-dire comment elle est construite à partir de R et $\mathbf{t}$) et de la traiter comme une matrice de dimension 3×3 quelconque. La contrainte épipolaire permet alors de l'estimer en résolvant un système d'équations linéaires cette fois-ci [17]. Les prix à payer sont un nombre plus important de correspondances requises (8 au minimum) et que la matrice ainsi estimée ne correspond pas en général (en présence de bruit) à une matrice essentielle exacte.

Il existe aussi des algorithmes intermédiaires, mais ceci n'est pas important pour notre propos. Les deux types d'approche extrayent finalement les paramètres de mouvement R et $\mathbf{t}$ de la matrice essentielle estimée. Pour la deuxième approche, ceci requiert une solution approximative, puisque la matrice estimée ne respecte pas la structure d'une vraie matrice essentielle.

Dans la suite, nous esquissons la deuxième approche pour deux exemples de matrices essentielles introduites dans cette section ; le développement d'approches minimales (par exemple pour la matrice essentielle 6×6 des caméras non centrales) est un problème ouvert et probablement assez difficile.

Caméras non centrales. La matrice essentielle a 36 coefficients, mais 9 d'entre eux sont toujours zéro et 9 autres apparaissent en double (les coefficients de R). L'estimation linéaire doit alors porter sur 18 coefficients, et comme elle se fait à partir d'équation linéaires et homogènes, 17 correspondances de rayons de projection sont suffisantes.

Comme il a été mentionné ci-dessus, l'estimation linéaire ne donnera pas une matrice essentielle parfaite. Pour extraire les paramètres de mouvement R et $\mathbf{t}$, nous devons alors en tenir compte. Nous donnons l'esquisse d'un algorithme :

1. Soit A la sous-matrice 3×3 en bas à gauche de E_n. Sans bruit, elle serait égale (à un facteur d'échelle près) à la matrice de rotation R. En présence de bruit, nous pouvons obtenir une matrice de rotation R via la SVD (décomposition en valeurs singulières [26]) de A : si $\mathsf{A} = \mathsf{U}\Sigma\mathsf{V}$ est la SVD de A, alors $\mathsf{R} = \mathsf{UV}$ est la matrice de rotation qui est la plus proche de A (au sens de la norme de Frobenius) [29]. Si le déterminant de R ainsi calculée vaut -1, il faut encore multiplier la matrice avec -1.
2. Déterminer le scalaire λ qui minimise
$$\|\lambda\mathsf{A} - \mathsf{R}\|_F$$
où $\|\cdot\|_F$ désigne la norme de Frobenius d'une matrice. C'est un problème de moindres carrés assez simple. Multiplier ensuite E_n avec λ.
3. Soit B la sous-matrice 3×3 en haut à gauche de E_n. Calculer $\mathbf{t}$ en minimisant
$$\|\mathsf{B} + [\mathbf{t}]_\times\mathsf{R}\|_F$$
ce qui revient à la résolution d'un problème de moindres carrés.

D'autres algorithmes sont bien sur possibles.

Caméras axiales avec axe fini. La matrice essentielle E_{af} contient 17 coefficients différents et peut alors être estimée en résolvant un système linéaire, à partir de 16 correspondances. L'algorithme d'extraction de R et $\mathbf{t}$ est assez similaire au cas précédent. Soit A la sous-matrice 3×2 en haut à droite de E_{af}. Nous pouvons estimer R à partir de A basé sur sa SVD (voir les détails dans [29]). Le reste de l'algorithme est analogue à celui des caméras non centrales.

TAB. 3 – Résumé des matrices essentielles pour différents modèles de caméras. La dernière colonne donne le nombre minimum de correspondances de rayons de projections qui sont requises pour une estimation linéaire des matrices essentielles.

Modèle de caméras	Matrice essentielle	Nombre
Caméra non centrale	$\mathsf{E}_n = \begin{pmatrix} -[\mathbf{t}]_\times \mathsf{R} & \mathsf{R} \\ \mathsf{R} & \mathsf{0}_{3\times 3} \end{pmatrix}$	17
Caméra axiale avec axe fini	$\mathsf{E}_{af} = \begin{pmatrix} -[\mathbf{t}]_\times \mathsf{R} & \begin{pmatrix} R_{11} & R_{12} \\ R_{21} & R_{22} \\ R_{31} & R_{32} \end{pmatrix} \\ \begin{pmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \end{pmatrix} & \mathsf{0}_{2\times 2} \end{pmatrix}$	16
Caméra axiale avec axe infini	$\mathsf{E}_{ai} = \begin{pmatrix} t_1R_{32} - t_3R_{12} & t_1R_{33} - t_3R_{13} & R_{21} & R_{22} & R_{23} \\ t_2R_{12} - t_1R_{22} & t_2R_{13} - t_1R_{23} & R_{31} & R_{32} & R_{33} \\ R_{12} & R_{13} & 0 & 0 & 0 \\ R_{22} & R_{23} & 0 & 0 & 0 \\ R_{32} & R_{33} & 0 & 0 & 0 \end{pmatrix}$	11
Caméra de type cross-slit avec deux axes finis	$\mathsf{E}_{xff} = \begin{pmatrix} 1 & 0 & 0 & W_2 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -Y_2 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \mathsf{E}_{af} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ W_1 & 0 & -Y_1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$	13
Caméra de type cross-slit avec un axe fini et un axe infini	$\mathsf{E}_{xfi} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & W_2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \mathsf{E}_{af} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & W_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$	10
Caméra centrale avec centre optique fini	$\mathsf{E}_{cf} = -[\mathbf{t}]_\times \mathsf{R}$	8
Caméra centrale avec centre optique infini	$\mathsf{E}_{ci} = \begin{pmatrix} t_2R_{13} - t_1R_{23} & R_{31} & R_{32} \\ R_{13} & 0 & 0 \\ R_{23} & 0 & 0 \end{pmatrix}$	4

Autres cas. La plupart des autres cas (à l'exception de la caméra centrale avec centre optique infini) peuvent être résolus de manière assez similaire et nous en omettons les détails. La forme des matrices essentielles des différents modèles de caméras ainsi que le nombre minimum de correspondances requises pour une estimation linéaire, sont donnés dans le tableau 3.

7 Conclusions

Nous avons proposé une géométrie d'images multiples pour des caméras non centrales générales, la première d'après notre connaissance. Une hiérarchie naturelle de modèles de caméras a été introduite, regroupant les caméras en classes selon la répartition spatiale de leurs rayons de projection. La géométrie de deux vues a été spécialisée et décrite en détail pour différents modèles de caméras. Nous espérons que ce travail théorique permet de définir un terrain commun pour de récents efforts dans la caractérisation de la géométrie de caméras non classiques.

Nous donnons quelques perspectives de travail assez immédiates. Les relations géométriques entre des caméras de différents types devraient être simples à dériver selon le schéma utilisé dans ce travail. Par exemple, la matrice essentielle entre une caméra centrale et une caméra axiale sera de taille 3×4. Aussi, une traduction de nos résultats en notation tensorielle est aisément envisageable. Comme c'est le cas pour les caméras perspectives, des contraintes d'appariement pour des images de droites au lieu de points peuvent probablement être développées.

Dans cet article, nous avons principalement considéré la théorie de la géométrie d'images multiples ; quant à l'estimation numérique des tenseurs et l'extraction des paramètres de mouvement, nous avons esquissé des méthodes pour le cas de deux vues, mais un traitement complet nécessite plus de travail. Des expériences pratiques avec l'estimation de la matrice essentielle pour des caméras non centrales ont pourtant été achevées avec succès, comme ça l'a été le cas pour d'autres auteurs [25]. Sinon, nous travaillons actuellement sur le développement de méthodes de calibrage spécifiques aux caméras axiales et de type cross-slit, dans l'esprit de [31].

Références

[1] Y.I. ABDEL-AZIZ et H.M. KARARA, "Direct Linear Transformation from Comparator Coordinates into Object Space Coordinates in Close-Range Photogrammetry", *ASP/UI Symposium on Close-Range Photogrammetry, Urbana, Illinois*, 1–18, 1971.

[2] S. BAKER et S.K. NAYAR, "A theory of single-viewpoint catadioptric image formation", *International Journal of Computer Vision*, 35(2) :1–22, 1999.

[3] H. BAKSTEIN, "Non-central cameras for 3D reconstruction", Rapport Technique CTU-CMP-2001-21, Center for Machine Perception, Czech Technical University, Prague, 2001.

[4] H. BAKSTEIN et T. PAJDLA, "An overview of non-central cameras", *Computer Vision Winter Workshop, Ljubljana, Slovenie*, 223–233, 2001.

[5] J.P. BARRETO et H. ARAUJO, "Paracatadioptric camera calibration using lines", *ICCV*, 1359–1365, 2003.

[6] R. BENOSMAN et S.B. KANG (éditeurs), *Panoramic Vision Sensors, Theory, and Applications*. Springer Verlag, 2001.

[7] E. BRASSART, L. DELAHOCHE, C. CAUCHOIS, C. DROCOURT, C. PEGARD et E. MOUADDIB, "Experimental Results Got With the Omnirectional Vision Sensor : SYCLOP", *IEEE Workshop on Omnidirectional Vision, Hilton Head, Caroline du Sud*, 145–152, 2000.

[8] O. FAUGERAS et B. MOURRAIN, "On the geometry and algebra of the point and line correspondences between n images", *ICCV*, 951–956, 1995.

[9] D. FELDMAN, T. PAJDLA et D. WEINSHALL, "On the epipolar geometry of the crossed-slits projection", *ICCV*, 988–995, 2003.

[10] C. GEYER et K. DANIILIDIS, "A unifying theory of central panoramic systems and practical applications", *ECCV*, 445–461, 2000.

[11] C. GEYER et K. DANIILIDIS, "Paracatadioptric camera calibration", *PAMI*, 24(5) :687–695, 2002.

[12] C. GEYER et K. DANIILIDIS, "Mirrors in Motion : Epipolar geometry and motion estimation", *ICCV*, 2 :766–773, 2003.

[13] M.D. GROSSBERG et S.K. NAYAR, "A general imaging model and a method for finding its parameters", *ICCV*, 2 :108–115, 2001.

[14] R.I. HARTLEY et R. GUPTA, "Linear pushbroom cameras", *ECCV*, 555–566, 1994.

[15] R.I. HARTLEY et A. ZISSERMAN, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[16] R.A. HICKS et R. BAJCSY, "Catadioptric sensors that approximate wide-angle perspective projections", *CVPR*, 545–551, 2000.

[17] H.C. LONGUET-HIGGINS, "A computer program for reconstructing a scene from two projections", *Nature*, 293 :133–135, 1981.

[18] J. NEUMANN, C. FERMÜLLER et Y. ALOIMONOS, "Polydioptric camera design and 3D motion estimation", *CVPR*, II :294–301, 2003.

[19] D. NISTÉR, "An efficient solution to the five-point relative pose problem", *CVPR*, II :195–202, 2003.

[20] T. PAJDLA, "Epipolar Geometry of Some Non-classical Cameras", *Computer Vision Winter Workshop, Bled, Slovenie*, 159–180, 2001.

[21] T. PAJDLA, T. SVOBODA et V. HLAVAC, "Epipolar Geometry of Central Panoramic Cameras", Dans *Panoramic Vision : Sensors, Theory, and Applications*, R. Benosman et S.B. Kang (éditeurs), 85–114, Springer Verlag, 2001.

[22] T. PAJDLA. "Geometry of two-slit camera", Rapport Technique CTU-CMP-2002-02, Center for Machine Perception, Czech Technical University, Prague, 2002.

[23] T. PAJDLA. "Stereo with oblique cameras", *IJCV*, 47(1-3) :161–170, 2002.

[24] S. PELEG, M. BEN-EZRA et Y. PRITCH, "Omnistereo : Panoramic stereo imaging", *PAMI*, 23(3) :279–290, 2001.

[25] R. PLESS, "Using many cameras as one", *CVPR*, II :587–593, 2003.

[26] W.H. PRESS, S.A. TEUKOLSKY, W.T. VETTERLING et B.P. FLANNERY, *Numerical Recipes in C - The Art of Scientific Computing*. Cambridge University Press, 1992.

[27] L.S. SHAPIRO, A. ZISSERMAN et M. BRADY, "3D Motion Recovery via Affine Epipolar Geometry", *IJCV*, 16(2) :147–182, 1995.

[28] H.-Y. SHUM, A. KALAI et S.M. SEITZ, "Omnivergent stereo", *ICCV*, 22–29, 1999.

[29] P. STURM, "Algorithms for Plane-Based Pose Estimation", *CVPR*, 1010–1017, 2000.

[30] P. STURM, "Mixing catadioptric and perspective cameras", *Workshop on Omnidirectional Vision, Copenhagen*, 37–44, 2002.

[31] P. STURM et S. RAMALINGAM, "A generic concept for camera calibration", *ECCV*, 1–13. 2004.

[32] T. SVOBODA, *Central Panoramic Cameras : Design, Geometry, Egomotion*. Thèse de doctorat, Faculty of Electrical Engineering, Czech Technical University, Prague, 1999.

[33] R. SWAMINATHAN, M.D. GROSSBERG et S.K. NAYAR, "A perspective on distortions", *CVPR*, II :594–601, 2003.

[34] J. YU et L. McMILLAN, "General linear cameras", *ECCV*, 14–27, 2004.

[35] A. ZOMET, D. FELDMAN, S. PELEG et D. WEINSHALL, "Mosaicing new views : The crossed-slit projection", *PAMI*, 25(6) :741–754, 2003.

Mixing Catadioptric and Perspective Cameras

Peter Sturm
INRIA Rhône-Alpes
655 Avenue de l'Europe, 38330 Montbonnot, France
Peter.Sturm@inrialpes.fr ⊙ http://www.inrialpes.fr/movi/people/Sturm

Abstract

We analyze relations that exist between multiple views of a static scene, where the views can be taken by any mixture of para-catadioptric, perspective or affine cameras. Concretely, we introduce the notion of fundamental matrix, trifocal and quadrifocal tensors for the different possible combinations of these camera types. We also introduce the notion of plane homography for mixed image pairs. Generally speaking, these novel multi-view relations may form the basis for the typical geometric computations like motion estimation, 3D reconstruction or (self-) calibration. A few novel algorithms illustrating some of these aspects, are described, especially concerning what we call calibration transfer*, using fundamental matrices, and self-calibration from plane homographies.*

1. Introduction

This work has been motivated by the increasing interest of vision researchers and practitioners in the theory and use of omnidirectional cameras [12, 13, 3]. Our main goal is to contribute to a unified theory encompassing omnidirectional and traditional (perspective) cameras. We are especially interested in the study of geometrical and algebraic multi-view relations and their use in various geometrical computations like 3D reconstruction, self-calibration or motion estimation.

During the last decade and until today, multi-view relations between perspective views have been extensively studied [9, 5]. Among the most important concepts, one might cite the multi-linear matching constraints (fundamental matrix and trifocal tensors) that enable robust matching of images and are useful in motion estimation; self-calibration and the notion of uncalibrated 3D vision; multi-view reconstruction using factorization etc. We would like to derive analogous concepts for omnidirectional cameras. Some of these concepts are already known, e.g. the fundamental and essential matrices for para-catadioptric cameras [7, 17], epipolar geometry for general central catadioptric cameras [17], calibration [2, 6] and self-calibation [7, 10] of para-catadioptric cameras.

In this paper, we generalize some previous results and introduce several new concepts. Very important, in our opinion, is to study multi-view relations that hold between omnidirectional and perspective cameras, and their applications. An important potential application of omnidirectional cameras, especially in video-surveillance, is to locate a visual event, and to "guide" a perspective camera that might fixate and zoom in on the event, to take close-ups. A perspective camera with a large zoom is usually better modeled as an affine camera (typically, an orthographic one). So, we study the multi-view relations that hold between any combination of omnidirectional, perspective and affine cameras. Concretely, we will introduce the different types of fundamental matrices, and show the existence of trifocal and quadrifocal tensors, as well as plane homographies between pairs of views. We then briefly discuss their use for (self-) calibration, by giving novel algorithms for calibration transfer and self-calibration from planes.

Concerning the types of omnidirectional camera, our eventual goal is to treat the various types of *central catadioptric cameras* [1]. In this paper, we nearly exclusively consider *para-catadioptric* cameras, e.g. systems consisting of a parabolic mirror and an affine camera. Currently, we are not able to generalize several of our results to the other types of central catadioptric cameras (especially, those based on hyperbolic mirrors), the problem being that the multi-view relations are not multi-linear in general.

Organization. In §2, linear backprojection equations are explained, that allow to derive multi-linear matching constraints in §3. Self-calibration and calibration transfer using fundamental matrices and plane homographies for mixed types of cameras, is shown in §4. Experimental results illustrating these concepts are given in these sections. §5 concludes and describes perspectives.

Notations. We denote matrices in sans serif (e.g. R), vectors in bold (e.g. $\mathbf{t}$), zero vectors as $\mathbf{0}$. The symbol $\sim$ means equality of vectors or matrices up to scale, accounting for homogeneous coordinates. The 3×3 identity matrix is denoted as $\mathtt{I}$. The skew-symmetric matrix associated with the cross-product is represented by $[\mathbf{v}]_\times$: $[\mathbf{v}]_\times \mathbf{w} = \mathbf{v} \times \mathbf{w}$.

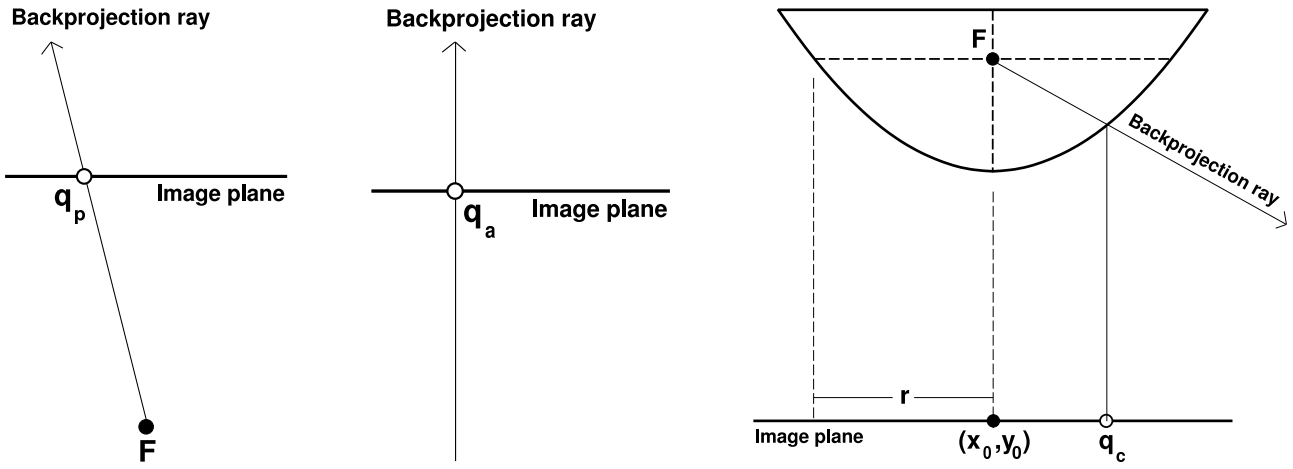


Figure 1. Camera models used in this paper. (a) Perspective projection: the optical center $\mathbf{F}$ is at position $\mathbf{t}_p$ (see text). (b) Affine projection: the (back-) projection rays are all parallel, and their direction is $\mathbf{r}_{a,3}$. (c) Para-catadioptric projection: the effective single viewpoint $\mathbf{F}$ is at position $\mathbf{t}_c$. The effective intrinsic parameters r, x_0 and y_0 are measured in pixels.

2. Camera Models

In this section, we explain the models we use for the camera types considered (see also figure 1). Since we are interested in deriving *multi-linear* constraints among views, we are keen to find linear projection equations. For perspective and affine cameras, 3×4 projection matrices *linearly* map homogeneous 3D point coordinates to homogeneous image point coordinates. As for catadioptric cameras, such linear projection equations do not seem to exist. What we will use instead are *backprojection matrices*, that map image point coordinates to the *direction* of the (back-) projection ray between the original 3D point and the (effective) optical center. It is possible to derive such mappings, that are linear, although not in standard image point coordinates, but in "lifted" ones, which shall be explained below. The backprojection equations derived in this section, will be used in section 3 to derive multi-view matching constraints.

2.1. Perspective Cameras

Let the projection matrix of a perspective camera be $\mathsf{P}_p \sim \mathsf{K}_p\mathsf{R}_p\begin{pmatrix}\mathtt{I} & -\mathbf{t}_p\end{pmatrix}$, where K_p is a calibration matrix (upper triangular 3×3), R_p a 3×3 rotation matrix and $\mathbf{t}_p$ the 3-vector of the optical center's coordinates.

All (finite) 3D points projecting onto a given image point $\mathbf{q}_p$ can be parameterized by a scale factor λ_p via:

$$\mathbf{Q} = \mathbf{t}_p + \lambda_p \mathbf{D}_p \ , \tag{1}$$

where the direction $\mathbf{D}_p$ of the projection ray is given by:

$$\mathbf{D}_p = (\mathsf{K}_p\mathsf{R}_p)^{-1}\mathbf{q}_p = \mathsf{R}_p^{\mathsf{T}}\mathsf{K}_p^{-1}\mathbf{q}_p \ . \tag{2}$$

2.2. Affine Cameras

Let the projection matrix of an affine camera be:

$$\mathsf{P}_a = \begin{pmatrix}\mathsf{K}_a\bar{\mathsf{R}}_a & \mathbf{t}_a \\ \mathbf{0}^{\mathsf{T}} & d_a\end{pmatrix} \ ,$$

with a 2×2 calibration matrix K_a, a 2-vector $\mathbf{t}_a$ and a 2×3 "amputated" rotation matrix $\bar{\mathsf{R}}_a$:

$$\bar{\mathsf{R}}_a = \begin{pmatrix}\mathbf{r}_{a,1}^{\mathsf{T}} \\ \mathbf{r}_{a,2}^{\mathsf{T}}\end{pmatrix} \ .$$

The missing third row gives the direction of the projection rays (they are all parallel). It is obtained (up to sign) as the cross-product of the other two rows: $\mathbf{r}_{a,3} = \mathbf{r}_{a,1} \wedge \mathbf{r}_{a,2}$.

All (finite) 3D points projecting onto an image point $\mathbf{q}_a$ (3-vector of homogeneous coordinates) can be parameterized by a scale factor λ'_a as follows:

$$\mathbf{Q} = \frac{1}{q_{a,3}}\mathsf{R}_a^{\mathsf{T}}\mathsf{K}_a^{-1}\begin{pmatrix}d_a & 0 & -t_{a,1} \\ 0 & d_a & -t_{a,2}\end{pmatrix}\mathbf{q}_a + \lambda'_a\mathbf{r}_{a,3} \ .$$

We will later use the following equation, obtained by multiplying the previous one by $q_{a,3}$:

$$q_{a,3}\mathbf{Q} = \underbrace{\mathsf{R}_a^{\mathsf{T}}\mathsf{K}_a^{-1}\begin{pmatrix}d_a & 0 & -t_{a,1} \\ 0 & d_a & -t_{a,2}\end{pmatrix}}_{\mathbf{B}_a}\mathbf{q}_a + \lambda_a\mathbf{r}_{a,3} \ , \tag{3}$$

with $\lambda_a = q_{a,3}\lambda'_a$ as free scale factor.

2.3. Para-Catadioptric Cameras

In this paper, we consider catadioptric systems consisting of a parabolic mirror and an affine camera. Concretely, the mirror is radially symmetric, and its surface may be represented by the quadric with the following matrix, for some scalar m defining the mirror's "curvature":

$$\Omega \sim \begin{pmatrix} 4m^2 & 0 & 0 & 0 \\ 0 & 4m^2 & 0 & 0 \\ 0 & 0 & 0 & -2m \\ 0 & 0 & -2m & -1 \end{pmatrix} .$$

Its two real focal points are the origin and the point at infinity of the Z-axis. Let the origin be the effective single viewpoint of the para-catadioptric system – we will sometimes also call it the *first focus*, whereas the point at infinity will be the *second focus*. Let P_c be the projection matrix of an affine camera, whose optical center is the second focus:

$$\mathsf{P}_c = \begin{pmatrix} \mathsf{K}_c & \mathbf{0} & \mathbf{t}_c \\ 0 \;\; 0 & 0 & d_c \end{pmatrix} ,$$

with a 2×2 calibration matrix K_c and a 2-vector $\mathbf{t}_c$. The calibration matrix allows to represent all types of affine camera: para-perspective, weak perspective or orthographic. For easier reading, we present in the following only formulas for orthographic projection, but note that all derivations have also been done for the general affine camera. For the orthographic camera, we have:

$$\mathsf{P}_c = \begin{pmatrix} k & 0 & 0 & t_{c,1} \\ 0 & k & 0 & t_{c,2} \\ 0 & 0 & 0 & d_c \end{pmatrix} .$$

Let $\mathbf{q}_c$ be the 3-vector of homogeneous coordinates of a point in the orthographic image. The direction $\mathbf{D}'_c$ of the effective (back-) projection ray (the line linking the effective viewpoint and the original 3D point $\mathbf{Q}$), can be computed as follows (we omit the subscripts c for clarity):

$$\mathbf{D}' = \begin{pmatrix} 4mkq_3 \left(q_1 d - q_3 t_1\right) \\ 4mkq_3 \left(q_2 d - q_3 t_2\right) \\ 4m^2 \left(\left(q_1 d - q_3 t_1\right)^2 + \left(q_2 d - q_3 t_2\right)^2\right) - k^2 q_3^2 \end{pmatrix} .$$

This is not linear in the image coordinates, however, by "lifting" them from the 3-vector $\mathbf{q}_c$ to the 4-vector[1]

$$\hat{\mathbf{q}}_c = \begin{pmatrix} q_{c,1}^2 + q_{c,2}^2 \\ q_{c,1} q_{c,3} \\ q_{c,2} q_{c,3} \\ q_{c,3}^2 \end{pmatrix} , \tag{4}$$

we obtain the following *linear* backprojection equation:

$$\mathbf{D}'_c = \mathsf{B}_c \hat{\mathbf{q}}_c ,$$

[1] This is similar to the lifted coordinates in [7], although here they are obtained in a purely algebraic manner.

where (we again omit the subscripts c):

$$\mathsf{B} = \begin{pmatrix} 0 & 4mkd & 0 & -4mkt_1 \\ 0 & 0 & 4mkd & -4mkt_2 \\ 4m^2 d^2 & -8m^2 dt_1 & -8m^2 dt_2 & 4m^2\left(t_1^2 + t_2^2\right) - k^2 \end{pmatrix} .$$

The parameters m, k, d, t_1 and t_2 are not independent, and we replace them by three *effective intrinsic parameters*: $r = \frac{k}{2md}, x_0 = \frac{t_1}{d}$ and $y_0 = \frac{t_2}{d}$ (cf. figure 1 (c)). With these, the backprojection matrix takes the form:

$$\mathsf{B}_c \sim \begin{pmatrix} 0 & 2r & 0 & -2rx_0 \\ 0 & 0 & 2r & -2ry_0 \\ 1 & -2x_0 & -2y_0 & x_0^2 + y_0^2 - r^2 \end{pmatrix} . \tag{5}$$

All (finite) 3D points projecting onto a given image point $\mathbf{q}_c$ can now be parameterized by a scale factor λ_c via:

$$\mathbf{Q} = \mathbf{t}_c + \lambda_c \underbrace{\mathsf{R}_c^{\mathsf{T}} \mathsf{B}_c \hat{\mathbf{q}}_c}_{\mathbf{D}_c} , \tag{6}$$

where R_c and $\mathbf{t}_c$ represent the extrinsic parameters of the para-catadioptric system.

3. Multi-Linear Multifocal Matching Constraints

We use the backprojection equations laid out in the previous section for perspective, affine and para-catadioptric cameras, to obtain multifocal matching constraints. We proceed similarly to what has been done in the pure perspective case to derive multi-linear matching constraints [4, 19]. In the first paragraph, we describe the general scheme, and in the following ones, we concentrate on special cases.

3.1. General Scheme

Consider projections of a 3D point $\mathbf{Q}$ (non-homogeneous coordinates) in a set of views. Consider the general case of u perspective, v affine and w para-catadioptric views, with image points $\mathbf{q}_p^1, \ldots, \mathbf{q}_p^u$ in the perspective views, and analogously for the other camera types. In the following, superscripts are associated to different images. The backprojection equations (1), (3) and (6) may be grouped in a single equation system as shown in equation (7) on top of the following page. The vectors $\mathbf{D}_p^i, \mathbf{B}_a^j$ and $\mathbf{D}_c^k$ respectively depend linearly on the (lifted) image points, and are defined in equations (2), (3) and (6) respectively.

The matrix M of this equation system, in the following also called *joint matrix*, has $3(u+v+w)$ rows and $(u+v+w+4)$ columns. Relations among the different projections of $\mathbf{Q}$ arise due to the fact that this matrix has a non-trivial null-vector (the vector containing the λ's and the coordinates of the 3D point $\mathbf{Q}$). Hence, M can not be of full column rank, i.e. its rank must be lower than $(u+v+w+4)$. This is equivalent to the statement that the determinants of all minors of size $(u+v+w+4)$ vanish. It is these determinants that give the multi-linear relations between matching image points in different views.

$$\underbrace{\left[\begin{array}{c|ccc|c|c} \mathbf{t}_p^1 & \mathbf{D}_p^1 & & & & \mathtt{I} \\ \vdots & & \ddots & & 0 & \vdots \\ \mathbf{t}_p^u & & & \mathbf{D}_p^u & & \mathtt{I} \\ \hline \mathbf{B}_a^1 & & & \mathbf{r}_{a,3}^1 & & q_{a,3}^1\mathtt{I} \\ \vdots & & 0 & & \ddots & \vdots \\ \mathbf{B}_a^v & & & & \mathbf{r}_{a,3}^v & q_{a,3}^v\mathtt{I} \\ \hline \mathbf{t}_c^1 & & & & \mathbf{D}_c^1 & \mathtt{I} \\ \vdots & & 0 & & \ddots & \vdots \\ \mathbf{t}_c^w & & & & & \mathbf{D}_c^w \quad \mathtt{I} \end{array}\right]}_{\mathsf{M}} \left[\begin{array}{c} 1 \\ \hline \lambda_p^1 \\ \vdots \\ \lambda_p^u \\ \hline \lambda_a^1 \\ \vdots \\ \lambda_a^v \\ \hline \lambda_c^1 \\ \vdots \\ \lambda_c^w \\ \hline -\mathbf{Q} \end{array}\right] = \left[\begin{array}{c} \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \hline \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \hline \mathbf{0} \\ \vdots \\ \mathbf{0} \end{array}\right] \tag{7}$$

In the following, we make these relations explicit. We do this especially for the various two-view cases, which gives rise to different types of fundamental matrix. We then show that, like in the purely perspective case, trifocal and quadrifocal tensors exist, but no higher-order matching tensors.

3.2. Bifocal Constraints – Fundamental Matrices

With two views, of any mixture of camera types, the joint matrix M is of size 6×6. Consider for example the joint matrix for a perspective and a para-catadioptric view, shown here in detail:

$$\underbrace{\left[\begin{array}{ccc|ccc} & & 0 & 1 & 0 & 0 \\ \mathbf{t}_p & \mathsf{R}_p^\mathsf{T}\mathsf{K}_p^{-1}\mathbf{q}_p & 0 & 0 & 1 & 0 \\ & & 0 & 0 & 0 & 1 \\ \hline & 0 & & 1 & 0 & 0 \\ \mathbf{t}_c & 0 & \mathsf{R}_c^\mathsf{T}\mathsf{B}_c\hat{\mathbf{q}}_c & 0 & 1 & 0 \\ & 0 & & 0 & 0 & 1 \end{array}\right]}_{\mathsf{M}} \left[\begin{array}{c} 1 \\ \hline \lambda_p \\ \lambda_c \\ \hline -Q_1 \\ -Q_2 \\ -Q_3 \end{array}\right] = \mathbf{0} \ .$$

This equation means that the 6×6 matrix M has a non-trivial null-vector, and hence must be of rank lower than 6. This in turn implies that all minors (submatrices) of size 6 are singular. The only minor of size 6 of M is the matrix itself. Hence, we obtain the bifocal matching constraint (the epipolar constraint) by developing its determinant. By doing so, one obtains an equation that is bilinear in the elements of $\mathbf{q}_p$ and $\hat{\mathbf{q}}_c$. This equation may thus be written in the following form:

$$\mathbf{q}_p^\mathsf{T} \ \mathsf{F}_{pc} \ \hat{\mathbf{q}}_c = 0 \ ,$$

where the matrix F_{pc} is of size 3×4 and its coefficients depend entirely on the entities defining the projections, i.e. the extrinsic parameters $\mathsf{R}_p, \mathbf{t}_p, \mathsf{R}_c, \mathbf{t}_c$ and the intrinsic parameters K_p and B_c.

One may recognize without difficulty in F_{pc} a fundamental matrix, which however relates here two views acquired with *different* camera types, and which does not have the usual dimensions, i.e. it is not even square as the fundamental matrix between two perspective views or between two para-catadioptric views [7].

This example concerned a perspective view, combined with a para-catadioptric one. The same findings hold for any mixture of the camera types considered in this paper:

- for two perspective views, the "traditional" fundamental matrix [11] is obtained. Any 3×3 matrix of rank 2 is a valid fundamental matrix.
- two affine views give a 3×3 affine fundamental matrix [14]. Affine fundamental matrices have a special form (upper left 2×2 submatrix is a null matrix).
- for two para-catadioptric views, a 4×4 fundamental matrix of rank 2 is obtained [7].
- mixtures of camera types lead to fundamental matrices of size 3×3 (perspective-affine) or 3×4 (perspective-catadioptric or affine-catadioptric). They can all be shown to be of rank 2.

A short comment is at order concerning affine cameras. In equation (7), image coordinates of affine views appear both in the first column (via the vectors $\mathbf{B}_a^j$) and in the last three columns (the identity matrices are multiplied by coordinates $q_{a,3}^j$). Thus, it is not obvious that a development of M's minors will lead to equations that are linear in the coordinates of each affine image point. Happily, it turns out that the equations can be factored such as to lead indeed to (multi-) linear equations.

In the following, we examine some properties of fundamental matrices of mixtures of a para-catadioptric with a perspective or an affine view.

3.3. Fundamental Matrices and Plane Homographies for Mixed View Pairs

These fundamental matrices are of size 3×4 (or 4×3, if we consider the transpose, which gives the "other direc-

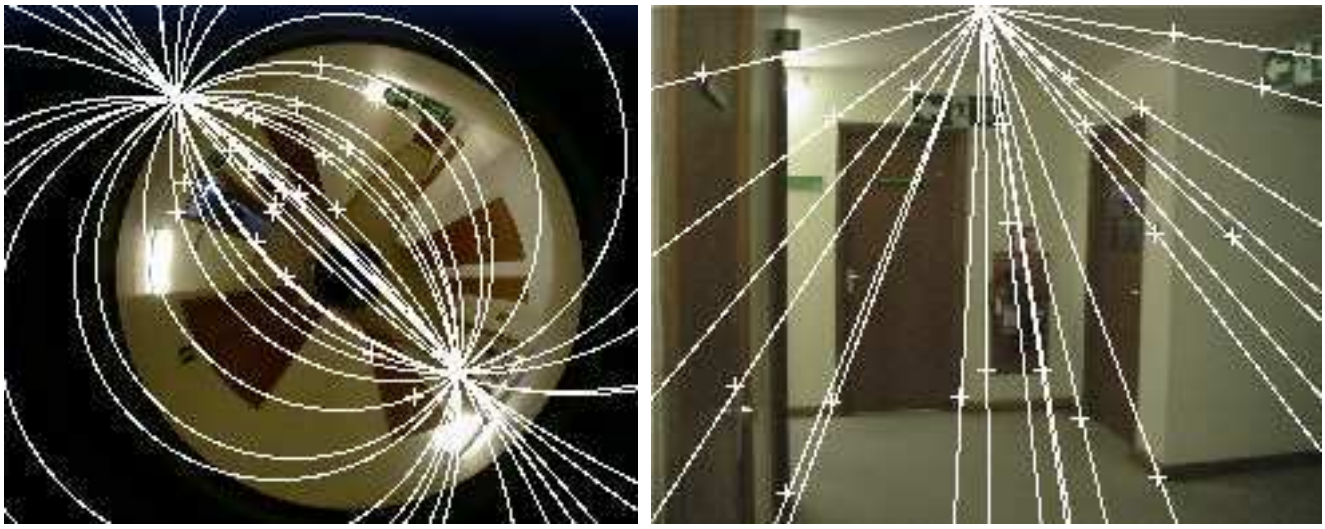


Figure 2. Estimated epipolar geometry for the stereo pair shown in figure 3. Points used to estimate F are shown by white crosses. For all 20 points, the epipolar circles in the catadioptric view and the epipolar lines in the perspective view, are shown. The two intersection points of the epipolar *circles* are the two epipoles of the catadioptric view mentioned in the text, whereas the single intersection point of the epipolar *lines* is the single epipole of the perspective view.

tion" of the epipolar constraint) and are of rank 2. The one-dimensional left null-space represents the epipole of the perspective (or affine) view, i.e. the image of the catadioptric view's effective viewpoint. The right null-space is two-dimensional. However, the fundamental matrix is only "valid" for 4-vectors of lifted coordinates, as defined in (4). There are exactly two right null-vectors of F (up to scale) that correspond to lifted coordinates. These are the two epipoles of the catadioptric view, i.e. the two projections of the perspective or affine camera's optical center (cf. [7]).

Products $\mathsf{F}\hat{\mathbf{q}}_c$ are 3-vectors, representing the usual epipolar lines in the perspective (or affine) view. As for products $\mathsf{F}^\mathsf{T}\mathbf{q}_p$, these are 4-vectors, representing the epipolar conics of catadioptric views. Let $\mathbf{x} = \mathsf{F}^\mathsf{T}\mathbf{q}_p$. The usual symmetric matrix of the associated epipolar conic is then given by:

$$\begin{pmatrix} 2x_1 & 0 & x_2 \\ 0 & 2x_1 & x_3 \\ x_2 & x_3 & 2x_4 \end{pmatrix} \ ,$$

which is a circle (the upper left 2×2 submatrix is a multiple of the identity matrix), which is in accordance with the known fact that epipolar conics of para-catadioptric systems are circles [17] (although this is only true for systems whose camera is perfectly orthographic).

Figure 2 shows the epipolar geometry (fundamental matrix), estimated by the analogon of the linear 8-point algorithm for the purely perspective case. Twenty manually selected points were used for the estimation. The estimated fundamental was also used for calibration transfer, see §4.3.

Figure 3. Stereo pair used in experiments.

Analogously to the purely perspective case, we may decompose the fundamental matrix to obtain the sometimes convenient epipole-homography form:

$$\mathsf{F} \sim [\mathbf{e}_p]_\times \mathsf{H} \ , \tag{8}$$

where $\mathbf{e}_p$ is the epipole in the perspective (or affine) view, and H a 3×4 *plane homography* matrix representing the mapping between the projections of points on some 3D scene plane. For example, the analogon to the *infinity homography* between two perspective views [9], for the case of a perspective and a para-catadioptric view, is given by the following 3×4 matrix:

$$\mathsf{H}_\infty = \mathsf{K}_p \mathsf{R}_p \mathsf{R}_c^\mathsf{T} \mathsf{B}_c \ , \tag{9}$$

with B_c defined as in equation (5). Using H_∞, we may derive the following expression for the fundamental matrix:

$$\mathsf{F} \sim \mathsf{K}_p^{-\mathsf{T}} [\mathsf{R}_p(\mathbf{t}_c - \mathbf{t}_p)]_\times \mathsf{K}_p \mathsf{R}_p \mathsf{R}_c^\mathsf{T} \mathsf{B}_c \ .$$

Concerning the above *plane homographies* H, they can be derived for all 3D scene planes Π: let $\hat{\mathbf{q}}_c$ be the projection of any point on Π, then $\mathbf{q}_p \sim \mathsf{H}\hat{\mathbf{q}}_c$ is the projection of the same point in the perspective view, where H is a 3×4 matrix. Note however that there is an important difference to the purely perspective case. A plane homography, as given above, is only defined in one direction: the mapping of an image point in the para-catadioptric view, via the scene plane, and then onto the perspective view, is unique, whereas the reverse direction isn't. Indeed, the mapping of an image point in the perspective view, onto a scene plane, *is* unique, however the projection into the catadioptric camera, leads to two (theoretically possible) image points. It is possible to exclude the image point that is physically not possible, but the projection equation is still not linear in general, which prevents forming an homography matrix as for the other direction.

In section 4, we examine further properties of fundamental matrices and plane homographies and show their application for calibration.

3.4. Multifocal Constraints

Three views. Let us first consider the case of three views, with any mixture of camera types. The joint matrix M is of size 9×7 in this case. Its rank-deficiency implies that the determinants of all minors of size 7 vanish. In other words, the determinant of a submatrix of M, obtained by choosing any 7 rows, must be equal to zero. Since to each of the three views, three rows of M are associated, only the following two possibilities of choosing 7 rows exist:

$$\begin{array}{ll}(a) & 3-3-1\\ (b) & 3-2-2\end{array}$$

where the figures refer to the number of rows chosen per one view. In case (a), it can be shown that the coordinates of the point in the view with a single contributed row, can be factored out from the resulting equation, and that we simply obtain the above bifocal relation for the two views with three contributed rows.

As for case (b), this gives rise to trilinear equations, which can be encoded via trifocal tensors. We identify tensors of size $4 \times 4 \times 4$ for the case of three para-catadioptric views, of size $4 \times 3 \times 3$ for a combination of one para-catadioptric and two perspective views, and so forth. Studying the properties of these tensors in more detail is beyond the scope of this paper though. As for trifocal tensors between triplets of cameras of the same type, the perspective case has been treated e.g. in [15] and the affine case in [18]. To our knowledge, no existing publication deals with the trifocal tensor for three para-catadioptric views or for the mixed configurations considered here.

Four views. In this case, the joint matrix is of size 12×8. Its rank-deficiency implies that the determinants of all minors of size 8 vanish. Analogously to the three-view case, we consider the different possibilities of choosing 8 rows of the joint matrix and their distribution among the four views:

$$\begin{array}{ll}(a) & 3-3-2-0\\ (b) & 3-3-1-1\\ (c) & 3-2-2-1\\ (d) & 2-2-2-2\end{array}$$

Case (a) leads to trivial equations (always zero). Cases (b) and (c) lead to bifocal and trifocal relations respectively, whereas case (d) gives quadrifocal relations. Quadrifocal tensors for perspective views are dealt with e.g. in [8, 16].

More than four views. With five views, the joint matrix is of size 15×9. Obviously, there is no minor of size 9 that contains at least two rows per image. Hence, there are no multi-linear matching constraints between five views (or more), that can not be represented using bifocal, trifocal or quadrifocal ones. The same holds for the purely perspective case of course.

4. Calibration using Fundamental Matrices and Plane Homographies

4.1. Self-Calibration from Plane Homographies

Let H be the 3×4 homography between a catadioptric and a perspective view, associated with a 3D plane. It can be shown (proof omitted due to lack of space) that the null-vector of any such plane homography is:

$$\begin{pmatrix} r^2 + x_0^2 + y_0^2 \\ x_0 \\ y_0 \\ 1 \end{pmatrix} . \tag{10}$$

Hence, self-calibration of the para-catadioptric camera is possible from a single plane homography, defined with respect to a perspective camera, by computing its null-vector and extracting the three intrinsic parameters r, x_0 and y_0 from it in a straightforward manner.

This might also be explained intuitively as follows. A para-catadioptric camera can be calibrated by identifying line images (circles in the image plane, that constitute images of 3D lines). If we know a plane homography with respect to a perspective view, we may virtually create all possible lines images, by mapping all lines of the perspective view via the homography, to the catadioptric view. Calibration could then be done as e.g. shown in [6], or, better, via the above solution using H's null-vector.

This self-calibration approach was tested using the image pair of section 3. Seven manually selected points lying on the wall in the background of the right hand part of figure 3, were used to estimate the associated plane homography

H, using a straightforward linear algorithm. The catadioptric view's intrinsic parameters, extracted from H, were 2% (x_0), 0.6% (y_0) respectively 5% (r) off the ground truth values, obtained as the center of the circle circumscribing the image (x_0, y_0) or via constructor-provided values (r).

4.2. Self-Calibration from Fundamental Matrices

It has been shown in [7], that the vector given in (10) is a null-vector of any fundamental matrix that a para-catadioptric camera shares with another camera of the same type. Hence, fundamental matrices between catadioptric cameras are useful for self-calibration [7, 10].

This observation can be generalized to self-calibration from fundamental matrices between a para-catadioptric view and e.g. a perspective one: the above vector can actually be identified as the single null-vector (up to scale) of the 3×4 backprojection matrix B_c defined in equation (5). Since $\mathsf{F} \sim [\mathbf{e}_p]_\times \mathsf{K}_p \mathsf{R}_p \mathsf{R}_c^\mathsf{T} \mathsf{B}_c$ (cf. equations (8) and (9)), it follows that the null-vector of B_c is also in the null-space of any fundamental matrix F. Hence, given several fundamental matrices, the null-vector of B_c can be found by "intersecting" all their right null-spaces, and intrinsic parameters can then be extracted from it.

4.3. Calibration Transfer by Fundamental Matrices

Consider the surveillance scenario sketched in the introduction. A typical configuration might consist of one static catadioptric camera, which in addition can usually be assumed to be pre-calibrated, and one or several traditional cameras, perspective or affine. It might be useful to estimate the position of a perspective camera, relative to the catadioptric one. Another task might be to calibrate the perspective camera (e.g. after zooming or focusing), using the fundamental matrix and the available calibration of the catadioptric camera, which is what we call *calibration transfer*.

The analogous task for two perspective cameras has been developed in [21]. The development for the mixed perspective-catadioptric case, is similar. Concretely, given a fully calibrated catadioptric view, a perspective view that is calibrated besides the unknown focal length, and the fundamental matrix between the two, a closed-form solution for the focal length, in terms of the SVD (singular value decomposition) of the fundamental matrix, is possible. We very briefly outline the algorithm (derivations are based on an analogon to the classical Kruppa equations for perspective cameras [22]).

Let F be the 3×4 fundamental matrix between a catadioptric camera and a perspective one. We assume that the catadioptric camera is calibrated, so we know e.g. its backprojection matrix B_c. As for the perspective camera, we know all its intrinsic parameters, besides the focal length. Let its calibration matrix K_p be decomposed in its known part K_k and a diagonal matrix with the unknown focal length:

$$\mathsf{K}_p = \mathsf{K}_k \operatorname{diag}(f, f, 1) \ .$$

1. Compute a "semi-calibrated" fundamental matrix:

   $$\mathsf{G} \sim \mathsf{K}_k^\mathsf{T} \, \mathsf{F} \, \mathsf{B}_c^+ \ ,$$

   where B_c^+ is the Moore-Penrose pseudo-inverse. It can be shown that G is of the form:

   $$\mathsf{G} \sim \operatorname{diag}(1, 1, f) \, [\mathbf{t}]_\times \mathsf{R} \ ,$$

   for a rotation matrix R. From this form, the following steps can be derived (cf. [21]).

2. Compute the SVD of G (remember that it is of rank 2):

   $$\mathsf{G} = \mathsf{U} \operatorname{diag}(r, s, 0) \, \mathsf{V}^\mathsf{T} \ .$$

3. The focal length can be computed by the following closed-form solution:

   $$f = \sqrt{\frac{s^2 u_{32}^2 - r^2 u_{31}^2}{r^2(u_{11}^2 + u_{21}^2) - s^2(u_{12}^2 + u_{22}^2)}} \ .$$

The algorithm was applied using the fundamental matrix estimated for the stereo pair shown in §3. The estimated focal length for the perspective camera was about 8% off the ground truth, which is reasonable, considered that no non-linear optimization was performed and that the points were specified with an accuracy of probably less than a pixel.

5. Conclusion and Perspectives

We have shown that it is possible to obtain multi-linear matching constraints, especially fundamental matrices and trifocal tensors, for any mixed configuration of perspective, affine or para-catadioptric cameras. Our approach unifies the development of the previously known multifocal tensors for pairs or triplets of cameras of the same type, and substantially generalizes the concept in that it allows a transparent combination of cameras of different types.

We are only partly satisfied, since our basic goal is to get a complete generalization that encompasses *all* central catadioptric systems. We have already established (not shown here) the existence of a 3×6 fundamental matrix between a perspective or affine view, and a general central catadioptric view, which however only "works in one direction" (there is a linear mapping from points in the perspective view to the corresponding epipolar conic in the catadioptric view; the reverse however is not available yet). Thus, we currently do not know if a complete generalization of our approach (in the multi-linear framework), is possible.

In this paper, we have also outlined the possibility of self-calibration and calibration transfer using "mixed fundamental matrices" and "mixed plane homographies".

Throughout the paper we have, for the sake of clarity, only presented formulas for para-catadioptric systems whose camera is an *orthographic* one. Note however, that all formulas have an analogon for the general case of affine cameras, the difference being that lifted image coordinates are 6-vectors, resulting e.g. in 6×6 fundamental matrices between two such catadioptric systems and similarly the dimension 4 is replaced by 6 for the other concepts.

As for our future work, we have several perspectives, some of which should be relatively straightforward to realize, others maybe not. Motion estimation for mixed camera configurations should be straightforward, but has to be developed and tested. In this paper, we have introduced plane homographies only for one direction: from catadioptric to perspective views. We want to clarify if and how the mapping in the inverse direction can be represented linearly. It should be relatively straightforward to develop trifocal tensors for line images, again for mixed camera configurations. A complete study of matching relations for mixed configurations should also list in detail the different types of essential matrices. A detailed study of algebraic properties of such essential matrices and trifocal tensors is possible, but is not central to our interests.

Besides the above mentioned generalization of our approach to general central catadioptric cameras, we are interested in the possibility of factorization-based methods for 3D reconstruction from multiple catadioptric views. For practical applications, it might for example be fruitful to develop methods similar to "reconstruction from N views having one view in common" [20], for the case of several perspective views, overlapping with a single catadioptric camera.

Acknowledgements. I wish to thank João Barreto for helpful discussions.

References

[1] S. Baker, S. Nayar, "A Theory of Catadioptric Image Formation," *Proceedings of the International Conference on Computer Vision, Bombay, India*, pp. 35–42, 1998.

[2] J.P. Barreto, H. Araujo, "Issues on the Geometry of Central Catadioptric Image Formation," *Proceedings of the* IEEE *Conference on Computer Vision and Pattern Recognition, Kauai, USA*, pp. 422–427, Vol. II, 2001.

[3] R. Benosman, S.B. Kang, (Editors), *Panoramic Vision: Sensors, Theory, and Applications*, Springer Verlag, 2001.

[4] O. Faugeras, B. Mourrain, "On the Geometry and Algebra of the Point and Line Correspondences Between N Images," *Proceedings of the International Conference on Computer Vision, Boston, USA*, pp. 951–956, 1995.

[5] O. Faugeras, Q.-T. Luong, T. Papadopoulo, *The Geometry of Multiple Images*, MIT Press, 2001.

[6] C. Geyer, K. Daniilidis, "Catadioptric Camera Calibration," *Proceedings of the International Conference on Computer Vision, Kerkyra, Greece*, pp. 398–404, 1999.

[7] C. Geyer, K. Daniilidis, "Structure and Motion from Uncalibrated Catadioptric Views," *Proceedings of the* IEEE *Conference on Computer Vision and Pattern Recognition, Kauai, USA*, pp. 279–286, Vol. I, 2001.

[8] R.I. Hartley, "Computation of the quadrifocal tensor," *Proceedings of the European Conference on Computer Vision, Freiburg, Germany*, pp. 20–35, Vol. I, 1998.

[9] R.I. Hartley, A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, 2000.

[10] S.B. Kang, "Catadioptric Self-Calibration," *Proceedings of the* IEEE *Conference on Computer Vision and Pattern Recognition, Hilton Head Island, USA*, pp. 201–207, Vol. I, 2000.

[11] Q.-T. Luong, O. Faugeras, "The Fundamental Matrix: Theory, Algorithms and Stability Analysis," *International Journal of Computer Vision*, pp. 43–76, Vol. 17, No. 1, 1996.

[12] *Proceedings of the* IEEE *Workshop on Omnidirectional Vision, Hilton Head Island, USA*, IEEE Computer Society Press, 2000.

[13] *Proceedings of the* ICAR *Workshop on Omnidirectional Vision Applied to Robotic Orientation and Nondestructive Testing (NDT), Budapest, Hungary*, 2001.

[14] L.S. Shapiro, A. Zisserman, M. Brady, "3D Motion Recovery via Affine Epipolar Geometry," *International Journal of Computer Vision*, pp. 147–182, Vol. 16, No. 2, 1995.

[15] A. Shashua, "Algebraic Functions for Recognition," IEEE *Transactions on Pattern Analysis and Machine Intelligence*, pp. 779–789, Vol. 17, No. 8, 1995.

[16] A. Shashua, L. Wolf, "On the Structure and Properties of the Quadrifocal Tensor," *Proceedings of the European Conference on Computer Vision, Dublin, Ireland*, pp. 710–724, Vol. I, 2000.

[17] T. Svoboda, T. Pajdla, V. Hlavác, "Epipolar Geometry for Panoramic Cameras," *Proceedings of the European Conference on Computer Vision, Freiburg, Germany*, pp. 218–232, Vol. I, 1998.

[18] T. Thorhallsson, D. Murray, "The Tensors of Three Affine Views," *Proceedings of the* IEEE *Conference on Computer Vision and Pattern Recognition, Fort Collins, USA*, pp. 450–456, 1999.

[19] B. Triggs, "Matching Constraints and the Joint Image," *Proceedings of the International Conference on Computer Vision, Boston, USA*, pp. 338–343, 1995.

[20] M. Urban, T. Pajdla, V. Hlavac, "Projective reconstruction from N views having one view in common," *Proceedings of the ICCV Workshop on Vision Algorithms: Theory and Practice*, pp. 116-131, Springer Verlag, 2000.

[21] M. Urbanek, R. Horaud, P. Sturm, "Combining Off- and Online Calibration of a Digital Camera," *Proceedings of the Third International Conference on 3D Digital Imaging and Modeling, Québec City, Canada*, pp. 99–106, 2001.

[22] C. Zeller, O. Faugeras, "Camera Self-Calibration from Video Sequences: the Kruppa Equations Revisited," Rapport de Recherche 2793, INRIA, France, 1996.

Part IV

3D Reconstruction

Chapter 10

Using Geometric Constraints for 3D Vision

10.1 Piecewise Planar Scenes

Paper 22 [1]: A. Bartoli and P. Sturm. Constrained structure and motion from multiple uncalibrated views of a piecewise planar scene. *International Journal of Computer Vision*, 52(1):45–64, 2003.

Paper 23 [8]: O. Chum, T. Pajdla, and P. Sturm. The geometric error for homographies. *Computer Vision and Image Understanding*, 97(1):86–102, January 2005.

Paper 24 [9]: D. Cobzas and P. Sturm. 3D SSD tracking with estimated 3D planes. In *Proceedings of the Second Canadian Conference on Computer and Robot Vision, Victoria, Canada*, May 2005.

10.2 Structure from Motion for Lines

Paper 25 [2]: A. Bartoli and P. Sturm. The 3D line motion matrix and alignement of line reconstructions. *International Journal of Computer Vision*, 57(3):159–178, 2004.

Paper 26 [4]: A. Bartoli and P. Sturm. Structure from motion using lines: Representation, triangulation and bundle adjustment. *Computer Vision and Image Understanding*, 100(3):416–441, December 2005.

10.3 Geometric Constraints

Paper 27 [30]: P. Sturm and S.J. Maybank. A method for interactive 3D reconstruction of piecewise planar objects from single images. In T. Pridmore and D. Elliman, editors, *Proceedings of the 10th British Machine Vision Conference, Nottingham, England*, pages 265–274. British Machine Vision Association, September 1999.

Paper 28 [22]: P. Sturm. A method for 3D reconstruction of piecewise planar objects from single panoramic images. In *Proceedings of the* IEEE *Workshop on Omnidirectional Vision, Hilton Head Island, South Carolina*, pages 119–126. IEEE, June 2000.

Paper 29 [39]: M. Wilczkowiak, P. Sturm, and E. Boyer. Using geometric constraints through parallelepipeds for calibration and 3D modelling. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 27(2):194–207, February 2005.

International Journal of Computer Vision 52(1), 45–64, 2003

Constrained Structure and Motion From Multiple Uncalibrated Views of a Piecewise Planar Scene

ADRIEN BARTOLI AND PETER STURM
INRIA Rhône-Alpes, 655, avenue de l'Europe, 38334 Saint Ismier cedex, France
Adrien.Bartoli@inria.fr
Peter.Sturm@inria.fr

Received March 14, 2001; Revised September 13, 2002; Accepted September 13, 2002

Abstract. This paper is about multi-view modeling of a rigid scene. We merge the traditional approaches of reconstructing image-extractable features and of modeling via user-provided geometry. We use features to obtain a first guess for structure and motion, fit geometric primitives, correct the structure so that reconstructed features lie exactly on geometric primitives and optimize both structure and motion in a bundle adjustment manner while enforcing the underlying constraints. We specialize this general scheme to the point features and the plane geometric primitives. The underlying geometric relationships are described by multi-coplanarity constraints. We propose a minimal parameterization of the structure enforcing these constraints and use it to devise the corresponding maximum likelihood estimator. The recovered primitives are then textured from the input images. The result is an accurate and photorealistic model.

Experimental results using simulated data confirm that the accuracy of the model using the constrained methods is of clearly superior quality compared to that of traditional methods and that our approach performs better than existing ones, for various scene configurations. In addition, we observe that the method still performs better in a number of configurations when the observed surfaces are not exactly planar. We also validate our method using real images.

Keywords: 3D reconstruction, piecewise planar scene, constrained structure and motion, maximum likelihood estimator

1. Introduction

The general problem of scene modeling is, given a sequence of images without a priori information, to recover a model of the scene as well as (relative) pose and calibration. Performing this task accurately is one of the key goals in computer vision.

In this paper, we focus on the geometric scene modeling, i.e. we do not address aspects of lighting and surface appearance recovery besides perspective correction of texture maps. We aim at devising a framework for the recovery of photorealistic and accurate models from a sparse set of images.

Existing works fall into two categories: the *feature-* and the *primitive-based* approaches. By *features*, we designate two- or lower-dimensional geometric entities that can be extracted from individual images (e.g. points, lines, conics). By *primitives*, we mean other entities, e.g. planes or higher-dimensional ones (cubes, spheres). Let us examine these two approaches in more detail. First, the *primitive-based* approach, see e.g. Debevec et al. (1996), Lang and Förstner (1996), and Streilein and Hirschberg (1995), in which the user typically provides parametric primitives through a modeling program. Parameters are determined using 3D-2D or 2D-2D matches and possibly refined using photometric criteria, such as maximization of the gradient for wireframe models, to optimize their reprojection. If necessary, camera calibration is performed and texture maps are extracted for each primitive to produce

a renderable model. This approach has proven to give convincing results, notably in terms of producing photorealistic rendering.

The *feature-based* approach, see e.g. Beardsley et al. (1996), relies on the existence of extractable image features. These features are matched accross the different views, typically using photometric and geometric criteria or by hand. From these, structure and motion are recovered. If necessary, camera calibration is performed and parameters refined in a bundle adjustment manner. This approach has proven to provide accurate reconstruction results, due to the high (in general) number of features considered. The problem is that modeling a scene with features alone does not allow to produce photorealistic rendering. Several works consider this issue, by using as features all the pixels, via dense matching (Pollefeys et al., 2000), space-carving (Kutulakos and Seitz, 1999; Seitz and Dyer, 1997), or plenoptic modeling (Gortler et al., 1996; Levoy and Hanrahan, 1996). The main limitation of at least the latter approach is that a high number of images is necessary to achieve accurate reconstruction. Other approaches relying on an a priori known environment (e.g. using turn-table sequences (Niem, 1994; Szeliski, 1993) or apparent contours (Cross and Zisserman, 2000)) can produce high quality rendering but do not work in the general case.

Actually, there exists a continuum between the two extreme feature- and primitive-based categories, made of *hybrid* approaches using both features and primitives.[1] These approaches are made to draw on the strength of both feature- and primitive-based categories: the high (in general) number of features might allow to obtain an accurate model recovery (even more accurate than for feature-based approaches) while the primitives contribute to form a photorealistic model. In hybrid approaches, the features and the primitives are linked by geometric constraints.

We study such an hybrid approach based on the point feature and the plane primitive. The geometric constraints used are incidence of points with none, one or several modeled planes and are called *multi-coplanarity constraints*. The corresponding constrained structure and motion recovery process is then called *piecewise planar structure and motion*.

These choices are motivated as follow. The point is a standard, widespread feature that may be easily extracted from the images. Most existing sparse structure and motion recovery systems deal with point features. The plane is a primitive sufficiently general to model a large number of real scenes, especially in man-made environments. Moreover, there are several works dealing with planes, that might be useful for an integrated modeling system: plane detection (Baillard and Zisserman, 1999; Berthilsson and Heyden, 1998; Dick et al., 2001; Faugeras and Lustman, 1988; Fornland and Schnörr, 1997; Sinclair and Blake, 1996; Tarel and Vézien, 1995), plane-guided point matching (Alon and Sclaroff, 2000; Faugeras and Lustman, 1988; Fornland and Schnörr, 1997; Sinclair and Blake, 1996; Viéville et al., 1995), and self-calibration using the knowledge of planes (Alon and Sclaroff, 2000; Malis and Cipolla, 2000; Triggs, 1998a; Viéville et al., 1995; Xu et al., 2000).

Concretely, we propose methods to parameterize points and planes under multi-coplanarity constraints. This parameterization is consistent in the sense that its number of parameters is the same as the number of degrees of freedom of the scene. It is employed to derive maximum likelihood estimators. Scene structure and camera motion are consistently estimated at once. A projective as well as a Euclidean version of the estimator are derived. The recovered structure perfectly satisfies the geometric constraints and is optimal in this respect, where optimal means maximum likelihood under a geometric error model.

In the following two sections, we present the piecewise planar structure and motion process and review existing work.

1.1. Piecewise Planar Structure and Motion

Given point correspondences between images, traditional *unconstrained structure and motion* reconstruct the points without using geometric constraints. First, suboptimal methods, see e.g. Beardsley et al. (1996) and Sturm and Triggs (1996), are used to compute an initial solution. The result is then refined using bundle adjustment (Slama, 1980; Triggs et al., 2000). If camera calibration is not available, the result is a projective reconstruction. In this case, the calibration information can be recovered online using several techniques (Heyden and Åström, 1998; Maybank and Faugeras, 1992; Pollefeys et al., 1998; Triggs, 1997). The uncalibrated reconstruction is then upgraded to metric and bundle adjustment is used to compute an optimal metric structure and motion.

In the projective case, when only points are used as features, then the scene has $11n - 15 + 3m$ essential degrees of freedom, where n is the number of views

and m the number of points. Each view has 11 degrees of freedom; 15 degrees of freedom for the choice of the projective basis are deduced.

Assume now, that not only point correspondences are available, but also their plane memberships. The goal is to compute an optimal structure and motion including the geometric constraints underlying to the special multi-coplanar structure of the points. Ideally, this process is a maximum likelihood estimator optimizing features, primitive positions, and viewing parameters while enforcing the underlying geometric constraints. Consequently, there is a need for a new formulation of structure and motion, that models both features and primitives, and that preserves the relationships between them, in our case, that models points and enforces multi-coplanarity constraints. The use of such a constrained estimator has a strong impact on the structure and motion process. Compared to the unconstrained case, the use of primitives constituting an important geometric constraint on both structure and motion, we can expect better reconstruction results. It might also be faster, as the number of parameters is usually lower.

Intra-primitive constraints, such as a priori known angles or parallelism of the modeled planes could be used. One problem with these constraints is that, generally speaking, they can not be used in a projective framework. Many other kinds of constraints could be modeled, such as the collinearity of points.

Choosing the constraints to model is difficult. Indeed, this is a trade-off between accuracy (the more constraints are used, the more accurate the reconstruction will be) and the complexity of the algebraic modeling. If too many kinds of constraints are used, then we end up with a network of constraints, that may be viewed as a graph linking features and primitives, and that might be redundent in the sense of cycles in this graph. Another issue is the automatization of an integrated modeling system. High-order constraints, such as the arrangement of planes in e.g. cuboïds, are more difficult to detect than the coplanarity of a set of points. A comprehensive treatment of the possible geometric constraints is out of the scope of this paper.

As said before, the incorporation of multi-coplanarity constraints has an impact on the number of essential degrees of freedom of the scene, e.g. a point on one plane has 2 degrees of freedom instead of 3 in the unconstrained case. Consequently, the number of degrees of freedom of such a scene becomes equal to $11n - 15 + 3p + 3m - \sum_j jm_j$ where the notation m_j designates the number of points lying on j of a total of p modeled planes.

Let us review existing piecewise planar structure and motion estimators.

1.2. Previous Work

Most of the existing works yield only a sub-optimal estimation of the geometry. Actually, they fall into two categories:

- The recovered structure is only approximately piecewise planar so clearly the results can not be optimal (Faugeras and Lustman, 1988; McGlone, 1996; Szeliski and Torr, 1998; Tarel and Vézien, 1995; Xu et al., 2000);
- The recovered structure is piecewise planar but the method is not optimal because it can not be turned into a maximum likelihood estimator or only the single-coplanarity constraint is modeled (Alon and Sclaroff, 2000; Baillard and Zisserman, 1999; Bartoli et al., 2001).

If we want our estimator to be optimal with respect to piecewise planarity, it has to fall into the second category, i.e. the recovered model has to be exactly piecewise planar. The constrained structure and motion is a maximum likelihood estimator that incorporates points, planes and multi-coplanarity constraints in a bundle adjustment manner. The cost function is non-linear (Slama, 1980; Triggs et al., 2000) and subject to constraints. There are several ways to conduct such an optimization process, in particular, we could use constrained optimization techniques such as sequential quadratic programming or a specific structure and motion parameterization enforcing the multi-coplanarity constraints (Bartoli and Sturm, 2001; Bartoli et al., 2001).

Ideally, these two possibilities give the same results because they are both consistent (i.e. the number of algebraic degrees of freedom is the same as that of essential degrees of freedom of the scene) and the cost function being optimized is the same. However, in practice, the convergence of the optimization process is determined by the number of parameters used which directly influences numerical stability. This determines which method to use in which case.

In our case, the number of parameters is high and so we have to reduce it to or close to the minimum, i.e. the number of essential degrees of freedom, if we

want to ensure a stable optimization process. The first possibility consists in systematically adding parameters to the system to model constraints and is consequently unadapted. The second possibility is less systematic, so needs more algebraic manipulations to be derived. However, the number of parameters is so reduced that the convergence might be faster and more reliable. Another issue that is important to be dealt with for both estimation cost and stability is that of analytic differentiation for the non-linear minimizer, which implies that the parameterization has a closed-form expression.

We addressed the case of two views and points lying on one plane (i.e. the single-coplanarity constraint) in Bartoli et al. (2001) and extended it to multi-view and multi-coplanarity constraints in Bartoli and Sturm (2001) where we derived the maximum likelihood estimator but without the possibility of analytic differentiation. In this paper, we present an estimator and the corresponding parameterization which is minimal for the structure and quasi-minimal for the motion, for n views and a quasi-general set of multi-coplanarity constraints and which allows analytic differentiation.

As real world surfaces are only approximately planar, we experimentally evaluate the performance of the constrained method compared to an unconstrained one with respect to different degrees of deviation from planarity and different scene configurations.

Since our approach needs to upgrade an uncalibrated reconstruction to metric, we perform self-calibration. To initialize a bundle adjustment procedure, we use the linear method of Pollefeys (1999), inspired by Triggs (1997), for the estimation of variable focal length. In practice, we encountered a singular situation, that is likely to occur in modeling applications: the optical axes of all images meet in a single 3D point (which will usually be the center of the modeled object). We adapt the basic method to this case and validate the approach on real images.

In Section 2 we give our notations. We then present our parameterization and the corresponding maximum likelihood estimator for a projective framework in Section 3, followed by an equivalent scheme in the Euclidean case in Section 4 where we also present self-calibration. We report on experiments on simulated data for constrained structure and motion in Section 5. Finally, Section 6 shows experimental results obtained using real images taken with an uncalibrated camera which validate both the reconstruction and the self-calibration processes, followed by our conclusions.

2. Notations

Physical entities (points, planes, etc.) are typeset using italic fonts (X, π, etc.) and their corresponding homogeneous coordinate vectors using the same letters in bold fonts ($\mathbf{X}$, $\boldsymbol{\pi}$, etc.). Matrices are designated by sans-serif fonts such as H. Vector and matrix elements are typeset using italic fonts and indices, e.g. $\mathbf{X} \sim (X_1, X_2, X_3, X_4)^\top$ where $^\top$ is the transposition and $\sim$ the equality up to a non-zero scale factor.

The notation $\mathbf{X}_{/j}$ is used to designate the vector formed with the elements of $\mathbf{X}$ with index different from j. Similarly, $\mathbf{X}_{j \leftarrow \alpha}$ represents the vector $\mathbf{X}$ with the value α inserted at the j-th position. The cross product is written $\times$ and the associated 3×3 skew-symmetric matrix $[\cdot]_\times$.

We model cameras using perspective projection, described by a 3×4 homogeneous matrix P. Non-linear optimization processes are conducted using the Levenberg-Marquardt algorithm (Gill et al., 1981).

3. Constrained Projective Structure and Motion

In this section, we describe how to minimally parameterize the structure and quasi-minimally the motion in the projective case. We then derive the maximum likelihood estimator corresponding to the constrained structure and motion.

As shown in the expression for the number of essential degrees of freedom of the scene, we have to take into account 15 degrees of gauge freedom left by the arbitrary choice for the projective basis of the reconstruction. Gauge freedom is defined as the internal freedom of choice for a coordinate system (Triggs, 1998b). It can be fixed using a particular formulation for the structure (Heyden and Åström, 1995) or for the camera matrices (Beardsley et al., 1996). Due to the complexity of structure parameterization, we have chosen to absorb the gauge freedom into the parameterization of motion.

In the next two sections, we describe respectively our structure and motion parameterizations.

3.1. Structure Parameterization

As said in the introduction, we have to parameterize both planes and points and in addition enforce the underlying multi-coplanarity constraints. The parameterization consists in passing from the usual homogeneous 4-vectors that represent points and planes in

3D projective space, to a minimal set of parameters representing the structure while enforcing the multi-coplanarity constraints. We first give an homogeneous and consistent parameterization for planes and points and then remove the homogeneity to reach a minimal parameterization. This last step is achieved using what we call *mapped coordinates* that allow to locally remove homogeneity. This is also used in the parameterization of the motion and in the Euclidean case.

3.1.1. Multi-coplanarity Constraints. A multi-coplanarity constraint is a geometric constraint between a point and a set of planes. Before parameterizing the structure, we have to decide where, in the parameterization of planes, of points or both, these constraints have to be incorporated. Actually, it seems inevitable to incorporate them in the point parameterization. Let us investigate the case of plane parameterization. Indeed, consider the case of a point lying on more than three planes. Such a point does not have, in general, any degree of freedom, and can be determined using three of the planes it lies on.[2] Once this point has been determined, it constrains the position of the other planes. Consequently, plane parameterization is dependent on multi-coplanarity constraints provided they contain a point lying on more than three planes.

If we do not model points lying on more than three planes (or take into account only three of the planes they lie on), it is possible to parameterize each plane independently while the multi-coplanarity constraints up to three planes are to be taken into account only for point parameterization. Considering that points lying on four or more planes are rare, we make such an assumption (an algebraic solution will just be sketched). Let us see the corresponding parameterization.

3.1.2. Planes. As said above, planes do not incorporate multi-coplanarity constraints and each one has therefore 3 degrees of freedom. An homogeneous 4-vector is then a consistent parameterization.

3.1.3. Points Under Multi-coplanarity Constraints. We describe point parameterization performed under a local incorporation of multi-coplanarity constraints. Let us examine different possible means. We then present our solution for the different multi-coplanarity cases.

To simplify the reading, we consider the case of a 2D point x constrained to lie on a 2D line l, which is similar to the 3D single-coplanarity case. Such a constraint is expressed as $\mathbf{l}^\top \mathbf{x} = 0$ and is satisfied for any point expressed in the nullspace of $\mathbf{l}^\top \sim (l_1, l_2, l_3)$.

The approach that naturally comes to mind is to compute a basis for the nullspace of $\mathbf{l}^\top$ and to express the coordinates of point x in this basis. We examine two ways to compute this nullspace basis and show that each of them is not appropriate to our problem.

A basis for the nullspace of $\mathbf{l}^\top$ is given by the skew-symmetric 3×3 cross-product matrix associated to $\mathbf{l}$ (there are other possible bases):

$$\mathsf{L} \sim [\mathbf{l}]_\times \sim \begin{pmatrix} 0 & -l_3 & l_2 \\ l_3 & 0 & -l_1 \\ -l_2 & l_1 & 0 \end{pmatrix}.$$

One can easily check that, as required, $\mathbf{l}^\top \mathsf{L} = \mathbf{0}_3^\top$ and $\operatorname{rank} \mathsf{L} = 2$. Any point on l can be represented by a linear combination of the 3 columns of L, thereby involving 3 homogeneous coefficients. This is not consistent since a point on a line has only 1 degree of freedom. On the other hand, one could think of using only 2 columns of L as a basis for the nullspace, say drop the leading column $\mathbf{l}_1$. In this case, the representation is consistent, but it is no more complete: the point with coordinate $\mathbf{l}_1$ lying on l can not be represented as a linear combination of the two last columns of L.

Another possibility is to compute an orthonormal basis for the nullspace of $\mathbf{l}^\top$ through e.g. its singular value decomposition:

$$\mathbf{l}^\top \sim \mathbf{l}^\top \operatorname{diag}(1, 0, 0)(\mathbf{l}_{3\times 1} \mid \bar{\mathsf{V}}_{3\times 2}).$$

In this case, the basis given by the two columns of $\bar{\mathsf{V}}$ is minimal and the corresponding parameterization would be consistent. However, since the entries of $\bar{\mathsf{V}}$ do not depend directly on the coefficients of $\mathbf{l}$, analytic differenciation would not be possible in the underlying optimization process.

Consistency and analytic differenciation are the main reasons for our specific parameterization to be used. We successively deal with points lying on none, one, two and three planes.

3.1.3.1. Unconstrained Points. Such a point does not lie on any modeled plane and being therefore not subject to any modeled geometric constraint, it has 3 degrees of freedom. An homogeneous 4-vector is then a consistent parameterization.

3.1.3.2. Single-Coplanar Points. Let X be a point constrained to lie on a plane π. Such a point has 2 degrees of freedom and our goal is then to express it via an homogeneous 3-vector—instead of the general homogeneous 4-vector—by incorporating the single-coplanarity constraint.

Algebraically, this constraint is written as $\boldsymbol{\pi}^\top \mathbf{X} = 0$. Let us find a change of projective basis where each point lying on π has an element fixed to a constant value, so that this element can be ignored in the parameterization of X. For that purpose, we define the class of homographies H_π^j by the identity matrix of size 4×4 where the jth row ($j \in \{1, \ldots, 4\}$) has been replaced by the 4-vector $\boldsymbol{\pi}^\top$ (e.g. $\mathsf{H}_\pi^1 \sim \begin{pmatrix} \boldsymbol{\pi}^\top \\ \mathbf{0}_3 \quad \mathrm{I}_{3\times 3} \end{pmatrix}$). Let $\Xi \sim \mathsf{H}_\pi^j \mathbf{X}$ be the representation of X in this new basis. By definition of H_π^j ,we have $\Xi_j = 0$ and the point X can therefore be parameterized by $\Xi_{/j}$, the homogeneous 3-vector formed from the 3 elements of Ξ with index different from j, $\mathbf{X}$ being further recovered using $\mathbf{X} \sim (\mathsf{H}_\pi^j)^{-1} \Xi$.

There are 4 possibilities for the choice of j. Since $(\mathsf{H}_\pi^j)^{-1}$ is necessary to recover $\mathbf{X}$ from Ξ, we choose j as the index that maximizes (in magnitude) the determinant of H_π^j: $j = \mathrm{argmax}_i \, |\det \mathsf{H}_\pi^i|$ which in fact leads to $j = \mathrm{argmax}_i \, |\pi_i|$. Such a choice ensures H_π^j to be a bijective transformation since $\det \mathsf{H}_\pi^j = \pi_j$ that, by construction, is always non-zero. Indeed, $\boldsymbol{\pi}$ is an homogeneous vector and has therefore at least one non-zero element.

Table 1 shows the practical algorithm for parameterizing/unparameterizing $X \in \pi$ derived from the above reasoning. In the unparameterization, we divide by π_j that, as said above, is always non-zero.

The dropped coordinate depends on the current estimate of π. Therefore, it might change between two steps of the optimization process. However, this does not create discontinuities since after each optimization step, the structure is unparameterized and standard homogeneous coordinates are recovered. The structure is then reparameterized for the next iteration, and the index of the dropped coordinate may change. The parameterization is therefore used in a local manner, which is important in order to keep smooth the cost function.

Table 1. Parameterization/unparameterization of a single-coplanar point.

Let X be a point subject to a single-coplanarity constraint with plane π. The homogeneous 4-vector $\mathbf{X}$ represents X in the current projective basis while the homogeneous 3-vector $\tilde{\mathbf{X}}$ is a parameterization of X incorporating the single-coplanarity constraint.

Parameterization ($\mathbf{X} \to \tilde{\mathbf{X}}$):

- Choose j such that $j = \arg\max_i |\pi_i|$ subject to $j \in \{1, \ldots, 4\}$ in the projective case and $j \in \{1, \ldots, 3\}$ in the Euclidean case;
- $\tilde{\mathbf{X}} \sim \mathbf{X}_{/j}$.

Unparameterization ($\tilde{\mathbf{X}} \to \mathbf{X}$):

- Compute $\alpha = -\frac{\sum_{i \neq j} \pi_i X_i}{\pi_j}$;
- $\mathbf{X} \sim \tilde{\mathbf{X}}_{j \leftarrow \alpha}$.

3.1.3.3. Multi-coplanar Points, Two Planes. Let X be a point constrained to lie on planes π and π'. Such a point has 1 degree of freedom provided that $\pi \neq \pi'$ and our goal is then to express it via an homogeneous 2-vector—instead of the general homogeneous 4-vector—by incorporating the multi-coplanarity constraint.

We follow the same reasoning as for the previous case. We define the class of homographies $\mathsf{H}_{\pi,\pi'}^{j,j'}$ by the matrix H_π^j where the j'-th row has been replaced by the 4-vector $\boldsymbol{\pi}'^\top$ (e.g. $\mathsf{H}_{\pi,\pi'}^{1,2} \sim \begin{pmatrix} \boldsymbol{\pi}^\top \\ \boldsymbol{\pi}'^\top \\ \mathbf{0}_{2\times 2} \quad \mathrm{I}_{2\times 2} \end{pmatrix}$). Let us consider $\Xi \sim \mathsf{H}_{\pi,\pi'}^{j,j'} \mathbf{X}$. By definition of $\mathsf{H}_{\pi,\pi'}^{j,j'}$, we have $\Xi_j = \Xi_{j'} = 0$ and point X can therefore be parameterized by $\Xi_{/j,j'}$, the homogeneous 2-vector formed from the 2 elements of Ξ with index different from j and j', $\mathbf{X}$ being further recovered using $\mathbf{X} \sim (\mathsf{H}_{\pi,\pi'}^{j,j'})^{-1} \Xi$.

Since j and j' must be different, this leaves $4 \times 3 = 12$ different choices for them. As $(\mathsf{H}_{\pi,\pi'}^{j,j'})^{-1}$ is needed, we choose j and j' such that the determinant of $\mathsf{H}_{\pi,\pi'}^{j,j'}$ is maximized (in magnitude). Subsequently deriving a practical algorithm as in the single-coplanarity case is then straightforward.

3.1.3.4. Multi-coplanar Points, Three Planes. Let X be a point constrained to lie on planes π, π' and π''. As already mentioned previously, it is straightforward to see that a point lying on three planes does not have, in general, any degree of freedom.[2] Such points are therefore not represented in the parameterization and have to be recovered from the three plane equations. There are two ways to do that. One can either choose a scheme similar to the one given previously or use the generalized cross-product, which gives a closed-form expression for the point (each point coordinate is given by the determinant of a 3×3 matrix of plane coefficients).

3.1.3.5. Multi-coplanar Points, More Than Three Planes. As said previously, this case is rare. Dealing with it properly would add a great complexity to the system, in the sense that constraints would then be

expressed not only on points but also on planes, thereby creating a graph of constraints with possible redundancies and cycles. Let us sketch, however, how the case of a point X lying on 4 planes π, π', π'' and π''' could be handled algebraically. Other higher order cases, though more complicated, could then be handled in a similar manner. The constraints are express as:

$$\mathsf{B}^\top \mathbf{X} = \mathbf{0}_4 \quad \text{where } \mathsf{B}_{4\times 4} \sim (\boldsymbol{\pi} \;\; \boldsymbol{\pi}' \;\; \boldsymbol{\pi}'' \;\; \boldsymbol{\pi}''').$$

This equation means that the matrix B has a (at least) 1-dimensional nullspace, i.e. $\det \mathsf{B} = 0$, which yields a 4-linear constraint on the coefficients of the plane equations. If one chooses to constrain e.g. plane π, then one of its coordinates may be dropped by considering the above-derived equation, and by applying a scheme similar to that described in Table 1, for the single-coplanarity case.

3.1.3.6. Modeling Intra-primitive Metric Constraints. In this paragraph, we give some hints on the algebraic modeling of intra-primitive constraints, and in particular on the perpendicularity and the orthogonality of planes. As explained in the introduction, a comprehensive treatment of all these constraints is out of the scope of this paper.

Firstly, consider the perpendicularity of two planes π and π'. This constraint can be algebraically expressed by considering that the dot product between the normal vectors of two such planes must vanish:

$$\pi_1\pi'_1 + \pi_2\pi'_2 + \pi_3\pi'_3 = 0.$$

This bilinear constraint can be enforced by the elimination of one parameter to contrain one of the two planes to be perpendicular to the other one. We end up with the same problem as that of modeling the single-coplanarity constraint described above.

Secondly, consider the modeling of the parallelism of two planes π and π'. The normal vectors of two such planes must be equal, up to scale, which is equivalent to nullifying there cross-product:

$$\begin{cases} \pi_2\pi'_3 - \pi_3\pi'_2 = 0 \\ \pi_3\pi'_1 - \pi_1\pi'_3 = 0 \\ \pi_1\pi'_3 - \pi_3\pi'_1 = 0. \end{cases}$$

Among these 3 equations, only 2 are independent, but one can not choose 2 of them a priori. Therefore, depending of which plane is to be contrained and on which axes, 2 equations are used to eliminate 2 of its parameters. Since these equations are bilinear, we end up with the same problem as that of modeling the multi-coplanarity constraint with 2 planes, described previously.

3.1.4. Mapped Coordinates. Homogeneous algebraic entities have an internal gauge freedom as they are only defined up to a non-zero scale factor. Consequently, they are not minimal in the sense that they are overparameterized. We define a tool called *mapped coordinates* that locally removes the homogeneity, in other words produces a minimal version of an homogeneous entity. Let us consider the case of homogeneous vectors of $\mathbb{P}^\nu$, which is not a restriction, the method being valid for any homogeneous entity (matrix, tensor). In more detail, a $(\nu+1)$-vector $\mathbf{v}$, can be decomposed into a ν-vector $\tilde{\mathbf{v}}$ and a map $\mu \in \{1, \ldots, \nu+1\}$, the index of a coefficient to be fixed. An important property is that slightly changing $\mathbf{v}$ does not, in general, affect μ but only $\tilde{\mathbf{v}}$, and if μ is affected, it will usually not create numerical instability (in the sense that the maximum coefficient of $\mathbf{v}$ will not tend towards zero during e.g. optimization).

The map μ is chosen as the index of the entry of $\mathbf{v}$ that has the largest absolute value. This choice can be justified as follows. If we assume that all entries of $\mathbf{v}$ have the same probability to become zero during an optimization step, our choice minimizes the probability that the selected entry (i.e. the one corresponding to the map μ) vanishes.

Consequently, this system is adapted to non-linear optimizers such as Levenberg-Marquardt (Gill et al., 1981), where the map can be re-estimated at each step of the optimization process. A practical algorithm for using mapped coordinates is given in Table 2.

Table 2. Mapped coordinates for homogeneous entities. Only $\tilde{\mathbf{v}}$ has to be included in optimization processes.

Let $\mathbf{v}$ be an homogeneous $(\nu+1)$-vector. Any other homogeneous entity (matrix, tensor) can be brought back to this case by stacking its elements into a single vector. The inhomogeneous ν-vector $\tilde{\mathbf{v}}$ represents the mapped coordinates of $\mathbf{v}$ whereas the integer μ represents its map.
Mapping $(\mathbf{v} \to (\tilde{\mathbf{v}}, \mu))$:
• Choose μ such that $\mu = \arg\max_i \lvert\mathbf{v}_i\rvert$;
• $\tilde{\mathbf{v}} = \frac{\mathbf{v}/\mu}{v_\mu}$.
Unmapping $((\tilde{\mathbf{v}}, \mu) \to \mathbf{v})$:
• $\mathbf{v} \sim \tilde{\mathbf{v}}_{\mu \leftarrow 1}$.

3.1.5. Summary of Structure Parameterization. We have given algorithms to exploit multi-coplanarity constraints for up to three planes per point. These constraints are enforced in an homogeneous manner while reducing the number of parameters for each point, see Section 3.1.3, and the homogeneity is removed using mapped coordinates, as indicated in Table 2, to obtain a minimal parameterization.

3.2. Motion Parameterization

In this section, we first parameterize camera projection matrices in an homogeneous manner and then remove the homogeneity using mapped coordinates to obtain a quasi-minimal parameterization.

We have chosen previously to fix the projective reconstruction basis via the camera parameterization. It has then to express $11n - 15$ degrees of freedom but actually has $10 + 11(n - 2)$ parameters (see below), i.e. is overparameterized by 3. This is not a problem for the optimization process since this number does not depend neither on the number of views nor on the number of points.

The number of parameters is obtained as follows. Each of the n views is represented by 11 parameters from its camera matrix, except for 2 of them, related by the epipolar geometry (or equivalently, one special-form projection matrix), that we represent using 10 parameters. More details are given below, where we describe the geometry of one, two, three or more views. Note that the motion parameterization is independent from the structure, and in particular, does not depend on the fact that the structure is constrained or not.

One View. The projective reconstruction basis can not be uniquely fixed. However the camera matrix P can be arbitrarily set, e.g. we use here $\mathsf{P} \sim (\mathrm{I} \mid \mathbf{0})$.

Two Views. If we suppose that the first camera matrix has been fixed, the second one has 7 degrees of freedom. Indeed, the geometry of such a system is described by the epipolar geometry contained in the rank deficient fundamental matrix F. Provided P has the form given above, the second camera matrix can be extracted from F as $\mathsf{P}' \sim ([\mathbf{e}']_\times \mathsf{F} \mid \mathbf{e}')$ where $\mathbf{e}'$ is the second epipole defined by $\mathsf{F}^\top \mathbf{e}' = 0$.

Minimally parameterizing the rank-2-ness of the fundamental matrix requires the use of several maps (Bartoli et al., 2001; Zhang, 1998) which is complicated from an implementation point of view. Alternatively, it is possible to overparameterize rank-2-ness by using a plane homography H and the second epipole e'. The second camera matrix is then formed as $\mathsf{P}' \sim ([\mathbf{e}']_\times^2 \mathsf{H} \mid \mathbf{e}')$ where $[\mathbf{e}']_\times^2 \mathsf{H}$ is the *canonical plane homography* which is the only plane homography satisfying $\mathsf{H}^\top \mathbf{e}' = 0$ (Bartoli and Sturm, 2001) (it is thus singular).

In this paper, we use this second possibility. The problem is parameterized by the 8 mapped coordinates of H and the 2 mapped coordinates of $\boldsymbol{e}'$, which yield 10 parameters. Consequently, it is overparameterized by $10 - 7 = 3$ parameters, since the two-view motion has only 7 degrees of freedom.

Three or More Views. Two or more views completely fix the projective basis. Consequently, each additional view adds 11 degrees of freedom to the system and in the general case their camera matrices do not have any special form and have to be entirely parameterized. We use mapped coordinates for that purpose.

The motion parameterization is summarized in Table 3.

3.3. Maximum Likelihood Estimator

We describe the maximum likelihood estimator for constrained structure and motion using the previously described parameterization. We first analyze which kinds of points are reconstructable and under which conditions, notably if they have to be included in the constrained optimization process. We then show how to initialize the parameterization from a general structure

Table 3. Motion parameterization. Notations $\tilde{\mathsf{H}}$, $\tilde{\mathbf{e}}'$ and $\tilde{\mathsf{P}}_k$ respectively designate the mapped coordinates (see Table 2) of the canonic plane homography (see text), of the second epipole (i.e. the projection of the first camera's center of projection onto the image plane of the second camera) and of other camera matrices. dof stands for degrees of freedom.

No. of views	No. of dof	No. of parameterization	Parameters	Gauge constraints
$n = 2$	7	10	$\tilde{\mathsf{H}}, \tilde{\mathbf{e}}'$	$\mathsf{H}^\top \mathbf{e}' = \mathbf{0}$
$n \geq 3$	$7 + 11(n - 2)$	$10 + 11(n - 2)$	$\tilde{\mathsf{H}}, \tilde{\mathbf{e}}', \tilde{\mathsf{P}}_{k \geq 3}$	$\mathsf{H}^\top \mathbf{e}' = \mathbf{0}$

and motion (when multi-coplanarity constraints are not enforced), in the case of motion and then structure. Finally, we give the cost function and details on the maximum likelihood estimator.

3.3.1. Initialization. At this step, we suppose to have a first guess of structure and motion as well as a clustering of points into multi-coplanar groups, see Section 6.

Feature Reconstructability. Planes are reconstructable provided that at least three points that they contain can be themselves reconstructed without geometric constraints. Once planes are reconstructed, new point reconstructions can be obtained. Table 4 gives which points, in terms of the number of views they are seen in and number of planes they lie on, can be reconstructed and if they have to be incorporated in the optimization process (i.e. if they add redundancy useful for optimization).

Motion. We have to change the projective basis such that the first camera matrix becomes $(\mathrm{I} \mid \mathbf{0})$. This is done by post-multiplying all camera matrices by an appropriately chosen 3D homography and pre-multiplying the structure by the inverse of this homography.

Constrained Structure. The initialization of points depending on that of planes, we first estimate plane equations and then points.

A plane is fitted to the points of each coplanar group. If X is a point lying on the plane π, the constraint $\mathbf{X}^\top \boldsymbol{\pi} = 0$ holds. By stacking the equations for all points lying on the plane, we obtain a linear system for $\boldsymbol{\pi}$ which can be solved using e.g. singular value decomposition. Another possibility is to estimate a plane homography between two images of the plane and to further extract the plane equation.

The unconstrained points and the multi-coplanar points lying on three or more planes are easy to initialize. Indeed, the former are not subject to any modeled geometric constraint and are taken directly from the initial structure, and the latter do not have any degree of freedom and so do not need initial values.

On the other hand, single-coplanar and multi-coplanar points lying on two planes need a special initialization. As we work in projective space, we can not consider any metric in space (such as orthogonal projection) and have to do measurements in the images.

For a single-coplanar point X lying on a plane π, we consider one of its projections and reconstruct the 3D point by intersecting the associated viewing ray with the plane π. We measure the reprojection error in all images where X is visible. We iterate over the set of images where X is visible and select the one that minimizes the total reprojection error.

For a multi-coplanar point X lying on planes π and π', we adopt the same method. However, to ensure that the reconstructed point lies on the two planes, we orthogonally project one of its image points onto the projection of the intersection line of π and π' and then reconstruct as above. Which plane π or π' is used to reconstruct does not matter. Details for this initialization are given in Bartoli and Sturm (2001).

Table 4. Summary of which points are reconstructable under which condition. "unconstrained" indicates a reconstruction when planes are not yet modeled, "optimization" indicates a reconstruction possible using planes and for points that add redundancy useful for optimization and "constrained" indicates a reconstruction possible only after the maximum likelihood estimation.

No. of views	No. of planes	Unconstrained	Optimization	Constrained
0	0	No	No	No
	1	No	No	No
	2	No	No	No
	≥ 3	No	No	Yes
1	0	No	No	No
	1	No	No	Yes
	≥ 2	No	Yes	No
≥ 2	Any	Yes	Yes	No

3.3.2. Optimization. Our goal is to derive an optimal estimator, in the sense of the maximum likelihood, for points and planes under multi-coplanarity constraints. This result is obtained by enforcing exactly the constraints, as is does by our parameterization. The cost function to minimize is the root mean square or, equivalently, the sum of square of the reprojection residuals (Slama, 1980; Triggs et al., 2000). In fact, this gives the maximum likelihood estimator under the assumption that errors in image point positions are identically and independently distributed according to a centered Gaussian, or normal, law.

We also include camera motion parameters into the non-linear optimization since an independent computation of the maximum likelihood estimate for the structure only is not possible: both structure and motion have to be estimated at once.

The cost function, denoted by $\mathcal{C}$, depends on measured image points x_{ij} and on reprojected points $\hat{x}_{ij}$ predicted by structure and motion parameters $\mathcal{S}$. It is defined by:

$$\mathcal{C}(\mathcal{S}) = \sum_i \sum_j w_{ij}\, d^2(\mathbf{x}_{ij}, \hat{\mathbf{x}}_{ij}).$$

Indices i and j respectively represent the different images and the different structure points and $d(.,.)$ is the Euclidean distance. We set $w_{ij} = 1$ if and only if the j-th point appears in the i-th image and 0 otherwise. The optimal structure and motion parameters $\hat{\mathcal{S}}$ are then given by the minimization of $\mathcal{C}$ over $\mathcal{S}$:

$$\hat{\mathcal{S}} = \arg\min_{\mathcal{S}} \mathcal{C}(\mathcal{S}).$$

This is done in practice using the Levenberg-Marquardt algorithm with analytic differentiation.

Let us investigate how to upgrade the obtained structure and motion to a metric frame.

4. Constrained Euclidean Structure and Motion

In this section, we describe how to upgrade the previously recovered projective structure and motion to metric and how to parameterize them in order to obtain a constrained maximum likelihood estimator.

4.1. Upgrade to Metric

There exist several possibilities to upgrade a projective reconstruction to metric, without a full prior calibration, e.g. by providing constraints on scene structure, camera motion, or calibration. In this work, we perform self-calibration. A Euclidean bundle adjustment is initialized using the linear method of Pollefeys (1999), inspired by Triggs (1997), that assumes known intrinsic parameters, besides the variable focal length. The method is rather straightforward, but we describe it here since the basic method is subject to a degenerate situation we encountered in practice, and that is likely to occur quite often in modeling applications for e.g. built environments. We give a variant of the method that does not degenerate in this case.

Suppose that P_i are the projection matrices associated with the projective reconstruction obtained so far. We suppose that all the intrinsic parameters are given, besides the focal lengths, f_i, for the individual images. In practice, we assume the principal points (u_i, v_i) to lie in the center of the respective image, and we know the cameras' aspect ratios τ_i (in fact, they could easily be included in the linear self-calibration routine). The skew factor is neglected, i.e. we assume pixels to be rectangular (in the linear method; skew is estimated during bundle adjustment).

Self-calibration is based on estimating a projective transformation T such that the transformed projection matrices can be decomposed into extrinsic and intrinsic parameters, such that the latter have the known values, i.e.:

$$\exists f_i, \mathsf{R}_i, \mathbf{t}_i : \mathsf{P}_i\mathsf{T} \sim \begin{pmatrix} \tau_i & 0 & u_i \\ 0 & 1 & v_i \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f_i & 0 & 0 \\ 0 & f_i & 0 \\ 0 & 0 & 1 \end{pmatrix} (\mathsf{R}_i \mid \mathbf{t}_i),$$

where the R_i are orthonormal matrices and the $\mathbf{t}_i$ 3-vectors. Considering only the leading 3×3 submatrix of the equation, and multiplying it by its transpose, we get:

$$\begin{aligned} &\mathsf{P}_i \bar{\mathsf{T}} \bar{\mathsf{T}}^\top \mathsf{P}_i^\top \\ &\quad \sim \begin{pmatrix} \tau_i & 0 & u_i \\ 0 & 1 & v_i \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f_i^2 & 0 & 0 \\ 0 & f_i^2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \tau_i & 0 & u_i \\ 0 & 1 & v_i \\ 0 & 0 & 1 \end{pmatrix}^\top, \end{aligned}$$

where $\bar{\mathsf{T}}$ is the 4×3 matrix consisting of the first three columns of T. Let

$$\begin{aligned} \mathsf{X} &= \bar{\mathsf{T}} \bar{\mathsf{T}}^\top \\ \mathsf{M}_i &= \begin{pmatrix} \tau_i & 0 & u_i \\ 0 & 1 & v_i \\ 0 & 0 & 1 \end{pmatrix}^{-1} \mathsf{P}_i. \end{aligned}$$

Then, the above equation becomes:

$$\mathsf{M}_i \mathsf{X} \mathsf{M}_i^\top \sim \begin{pmatrix} f_i^2 & 0 & 0 \\ 0 & f_i^2 & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{1}$$

The matrix X represents the "absolute quadric" (Triggs, 1997), in the space of the projective reconstruction. It is 4×4, symmetric and of rank 3.

Let $\mathbf{m}_{ij}^\top$ be the vector representing the j-th row of M_i. From Eq. (1), the following linear equations on X can be obtained:

$$\begin{aligned} \mathbf{m}_{i1}^\top \mathsf{X} \mathbf{m}_{i2} &= 0 \\ \mathbf{m}_{i1}^\top \mathsf{X} \mathbf{m}_{i3} &= 0 \\ \mathbf{m}_{i2}^\top \mathsf{X} \mathbf{m}_{i3} &= 0 \\ \mathbf{m}_{i1}^\top \mathsf{X} \mathbf{m}_{i1} - \mathbf{m}_{i2}^\top \mathsf{X} \mathbf{m}_{i2} &= 0. \end{aligned}$$

The rank-3 constraint on X can not be imposed via linear equations, which implies that there exist singularities for the linear method, that are not singular for the generic case (Sturm, 2000). The generic singularities (critical motions) for self-calibration of varying focal length (with other intrinsic parameters known), are described in Kahl et al. (2000) and Sturm (1999). An imaging configuration that is singular for the linear approach, but not in general, is the case where the optical axes of all views pass through one 3D point. Image sequences taken for modeling objects will very often be singular in this respect (e.g. the sequence shown in Fig. 3).

Due to this singularity, the system of the above linear equations will have a one-dimensional family of solutions:

$$\mathsf{X} \sim \mathsf{X}_1 + \mu \mathsf{X}_2.$$

The rank-3-constraint allows to solve for μ via the equation $\det \mathsf{X} = 0$. This is a degree-4-polynomial in μ. We solve it numerically, thus obtaining a maximum of 4 solutions for X. To find a unique solution, we compute the focal lengths that each solution gives rise to, and choose the solution, where these respect practical bounds (they have to lie in an interval of the order [300, 5000], depending on the camera used). In practice, we always found a single solution satisfying these constraints, the others being far off.

Focal lengths are extracted by computing $\omega_i \sim \mathsf{M}_i \mathsf{X} \mathsf{M}_i^\top$ and then

$$f_i = \sqrt{\frac{\frac{1}{2}(\omega_{i,11} + \omega_{i,22})}{\omega_{i,33}}}.$$

From the estimated X, we extract a projective transformation that upgrades projection matrices and point coordinates to metric. There is no unique solution for this, so in practice we choose one that has roughly equal singular values. Let $\mathsf{X} = \pm \mathsf{U} \Sigma \mathsf{U}^\top$ be the singular value decomposition of X. Since X is of rank 3, the 4-th singular value is zero. Let Σ' be obtained by replacing that zero with e.g. the largest singular value in Σ, we obtain the projective upgrade transformation needed:

$$\mathsf{T} = \mathsf{U} \sqrt{\Sigma'}.$$

Extracting extrinsic parameters from the upgraded projection matrices is then straightforward—it basically requires fitting of orthonormal matrices to general 3×3 matrices (Horn et al., 1988). More details are given in Section 4.4. The result is optimized via bundle adjustment. An alternative to the described approach would be to use the coplanarity information already available, like Alon and Sclaroff (2000), Malis and Cipolla (2000), Triggs (1998a), Viéville et al. (1995), and Xu et al. (2000).

In the following paragraph, we just give a few numerical details. In order to improve the condition of the linear equation system, we transform the matrices M_i as follows. First, we assume that images are normalized using e.g. Hartley (1995). Second, we make use of the free choice for the basis of the projective reconstruction, by computing a projective transformation, that hopefully leads to better conditioning. A simple method to do that is as follows. We stack the M_i in a matrix M of size $3n \times 4$, and compute its singular value decomposition:

$$\mathsf{M} = \mathsf{A} \Gamma \mathsf{B}^\top.$$

From A, we extract sub-matrices replacing the M_i in the linear equations: A is orthonormal, so the linear equations are more likely to be well conditioned. The product $\Gamma \mathsf{B}^\top$ represents the projective transformation corresponding to the mapping between the original and the transformed M_i (naturally, the 3D points have to be transformed accordingly). Using this normalization, we obtained much more accurate initial values and actually prevented the bundle adjustment to fall in a local minimum it got trapped in otherwise, in one case.

4.2. *Structure Parameterization*

In this section, we adapt the projective structure parameterization of Section 3.1 to the Euclidean case. In this case, planes are modeled as homogeneous 4-vectors, whereas points can be written as inhomogeneous 3-vectors.

The plane parameterization has been described in Section 3.1.2 and mapped coordinates (cf. Section 3.1.4) were used to reach the minimality. The point parameterization under multi-coplanarity constraints of Section 3.1.3 for the projective case can be used either directly or adapted to take full advantage of the Euclidean structure. We successively specialize the different cases.

Unconstrained Points. As said above, points can be parameterized using inhomogeneous 3-vectors, which is minimal in this case.

Single-Coplanar Points. Let X be a point lying on a plane π. As for the projective case, we want to change the reconstruction basis such as to fix an element of $\mathbf{X}$ to a constant value. In the Euclidean case, we have $\mathbf{X}^\top \sim (\bar{\mathbf{X}}^\top \mid 1)$ in the homogeneous form, so that the 4-th element is already fixed. Consequently, we must choose a transformation that preserves this element while fixing another one. This class of transformation is H_π^j where $j \in \{1 \dots 3\}$. The practical algorithm for parameterizing/unparameterizing such a point in the Euclidean case is similar to that of Table 1 but using the constraint $j \in \{1 \dots 3\}$ for the choice of j.

Multi-coplanar Points. We follow the same reasoning as in the previous case. A point lying on two planes is then parameterized by a scalar and does not have parameters in the three planes case. The practical algorithms are then identical to the projective case, provided a choice for the indices j and j' in $\{1 \dots 3\}$ for the two planes case.

4.3. Motion Parameterization

For motion parameterization in the Euclidean case, we suppose that each camera has z unknown intrinsic parameters, where $z \in \{1 \dots 5\}$.

One View. We choose the reconstruction basis such that $\mathsf{P} \sim \mathsf{K}\,(\mathsf{I} \mid \mathbf{0})$ where K is the calibration matrix, containing the intrinsic parameters. We have therefore z degrees of freedom for this first camera.

Two or More Views. The Euclidean basis has been fixed by the first view up to a global scale factor. We then have to completely parameterize the other camera matrices. Such an additional camera is written as $\mathsf{P}' \sim \mathsf{K}'\,(\mathsf{R} \mid \mathbf{t})$. Making the same assumption on the intrinsic parameters than for the first view, this leaves $z + 6$ degrees of freedom for each view, its internal parameters and the 6 parameters for the rotation R and the translation $\mathbf{t}$. These entities are minimally parameterized, as described in e.g. Atkinson (1996).

4.4. Maximum Likelihood Estimator

The maximum likelihood estimator in the metric case is very similar to that of the projective case as the cost function is the same. The intrinsic parameters for each camera have been recovered previously, see Section 4.1. In order to initialize our parameterization, we still need to extract the relative pose of each camera, i.e. factorize each projection matrix $\mathsf{P} \sim (\bar{\mathsf{P}} \mid \mathbf{p})$ under the form $\mathsf{P} = \frac{1}{\lambda}\mathsf{K}(\mathsf{R} \mid \mathbf{t})$ where λ is an unknown scale factor. Let us define $\mathsf{S} = \mathsf{K}^{-1}\bar{\mathsf{P}}$. We first estimate the scale factor as $\lambda = \sqrt[3]{\det \mathsf{S}}$. The translation can then be obtained by $\mathbf{t} = \lambda\mathsf{K}^{-1}\mathbf{p}$. In the noise-free case, $\lambda\mathsf{S}$ is an orthonormal matrix, but in practice it is not and we choose the closest rotation matrix in the sense of the Frobenius norm. This can be done using a singular value decomposition of $\lambda\mathsf{S}$ and a recomposition where the matrix of singular values Σ is omitted: $\mathsf{R} = \mathsf{U}\mathsf{V}^\top$ where $\lambda\mathsf{S} = \mathsf{U}\Sigma\mathsf{V}^\top$. Once this initialization has been done, non-linear optimization of the cost function $\mathcal{C}$ (cf. Section 3.3) can be launched using the Levenberg-Marquardt algorithm (Gill et al., 1981) with analytic differentiation.

5. Experimental Results Using Simulated Data

In this section, we compare our method to existing ones, notably to that consisting in individually reconstructing each point and to that using approximate multi-coplanarity constraints. We perform this comparison for the structure results, then for the motion results.

The test bench consists of a cube of one meter side length observed by a set of cameras. Points are generated on the cube, possibly offset from their planes in order to simulate non-perfect coplanarity and projected onto the images, where centered Gaussian noise is added. The default parameters of this simulation are the following. Up to 50, 10 and 1 points are generated on respectively each face, edge and vertex of the cube. Two cameras with a focal length of 1000 (expressed in number of pixels) and a 1 meter baseline are situated at a distance of 10 meters from the cube. The standard deviation of image noise is up to 3.0 pixels. The intrinsic parameters are not supposed to be known which yields projective reconstructions.

In the sequel, we vary independently each of these parameters to compare the different approaches under various conditions, especially we want to determine how the constrained methods behave when the observed surfaces are only approximately planar.

We measure the quality of reconstructions using the 3D residual of its Euclidean distance to the ground truth scene structure $\underline{X}$: $E = \sqrt{\frac{1}{m}\sum_{j=1}^{m} d^2(\mathsf{H}\mathbf{X}_j, \underline{\mathbf{X}}_j)}$, where H is a 3D homography (mapping the projective to the Euclidean structure) estimated using non-linear minimization of E. We measure the median value over 100 trials.

The estimators compared are:

- *Po-ML*: Optimal structure and motion obtained in a bundle adjustment manner (Triggs et al., 2000) without geometric constraints;
- *Pl-wt*: (*wt* stands for weights) similar to *Po-ML* but uses heavily weighted ($2^{60} \approx 10^{20}$) additional equations to approximate multi-coplanarity (McGlone, 1996; Szeliski and Torr, 1998);
- *Pl-ML*: Uses the parameterization described in this paper to explicitly model multi-coplanarity;
- *Pl-h*: (*h* stands for homography) uses method *Po-ML* described above with as input point correspondences corrected by maximum likelihood estimation of homographies. This method is described in more detail below. Note that it works only with two images and with the single-coplanarity constraint.

The last method evaluated relies on a simple homography-based point correction. A plane observed by two cameras induces an homography. This homography relates the projections of the points lying on the plane. The family of such homographies is 3-dimensional, provided that the epipolar geometry is known (this is linked to the fact that a plane has 3 degrees of freedom). In the calibrated case, they depend upon the relative pose between the two cameras and on their intrinsic parameters. If all these consistency constraints are ignored, and if the piecewise planar structure and motion problem is considered only for two views and with single-coplanarity constraints, one can devise a simple process to incorporate the knownledge of coplanarity in a standard unconstrained reconstruction method. Indeed, one can estimate independently each homography corresponding to each coplanar group of points and correct them so that they perfectly correspond through the homography. A standard structure and motion algorithm can then be launched with as input the corrected points. This is what *Pl-h* does. Obviously, this process is suboptimal since most consistency constraints have been ignored, and since the final reconstruction is only approximately planar. Extending the idea to multi-view and multi-coplanarity constraints, by enforcing all the underlying consistency constraints would yield the same result as our estimator, up to the convergence of the underlying non-linear optimizers. However, the algebraic structure would be more complicated since more consistency constraints have to be imposed in the images than in the 3D space.

Let us describe the different experimental situations when varying a scene parameter and the simulation results we have obtained.

Added Image Noise (Fig. 1(a)): The standard deviation of added image noise is varied from 0 to 3 pixels;

Baseline (Fig. 1(b)): The baseline is varied between 0.1 and 1 meter;

Number of Points (Fig. 1(c)): The number of points is respectively equal to 50α, 10α and 1 for each face, edge and vertex of the cube, where α varies from 0.1 to 1;

Number of Views (Fig. 1(d)): The number of views varies from 2 to 10. The different cameras are situated such that the baseline between two consecutive ones is 1 meter;

Distance Scene/Cameras (Fig. 1(e)): The distance between the cube and the cameras is varied between 10 and 20 meters.

In all these cases, the method *Po-ML* based only on individual point reconstruction gives results of a quality lower than methods *Pl*-modeling also planes (the residual is at least twice as low). The method *Pl-ML* performs slightly better than *Pl-wt* in all cases. Finally, method *Pl-h* gives results slightly worse than *Pl-wt*, but much better than *Po-ML*.

One aspect not shown on the graphs of Fig. 1, due to the use of a median value over a large number of trials, is that methods *Po-ML* and *Pl-wt* have a percentage of convergence lower than *Pl-ML* and *Pl-h*, especially for unstable configurations (large image noise, small baseline, high distance scene/cameras etc.). For example, the percentage of convergent estimations over 1000 trials is 95.2%, 89.1%, 97.5% and 97.3% for *Po-ML*, *Pl-wt*, *Pl-ML* and *Pl-h* respectively, for a distance scene/cameras of 20 meters and a 0.1 meter baseline.

Plane Unflatness (Fig. 1(f)): 3D points are offset from the planes they lie on by distances drawn from a normal distribution with standard deviation between 0 and 0.1 meters.

We observe that there is a threshold on the plane unflatness where methods *Pl-* using the knowledge of planes begin to perform worse than method *Po-ML*. It is interesting to define the *breakdown ratio*, denoted by ε, as the ratio between the extent of 3D noise and plane surface area, assuming that the scene is seen completely in all views. In the case of Fig. 1(f),

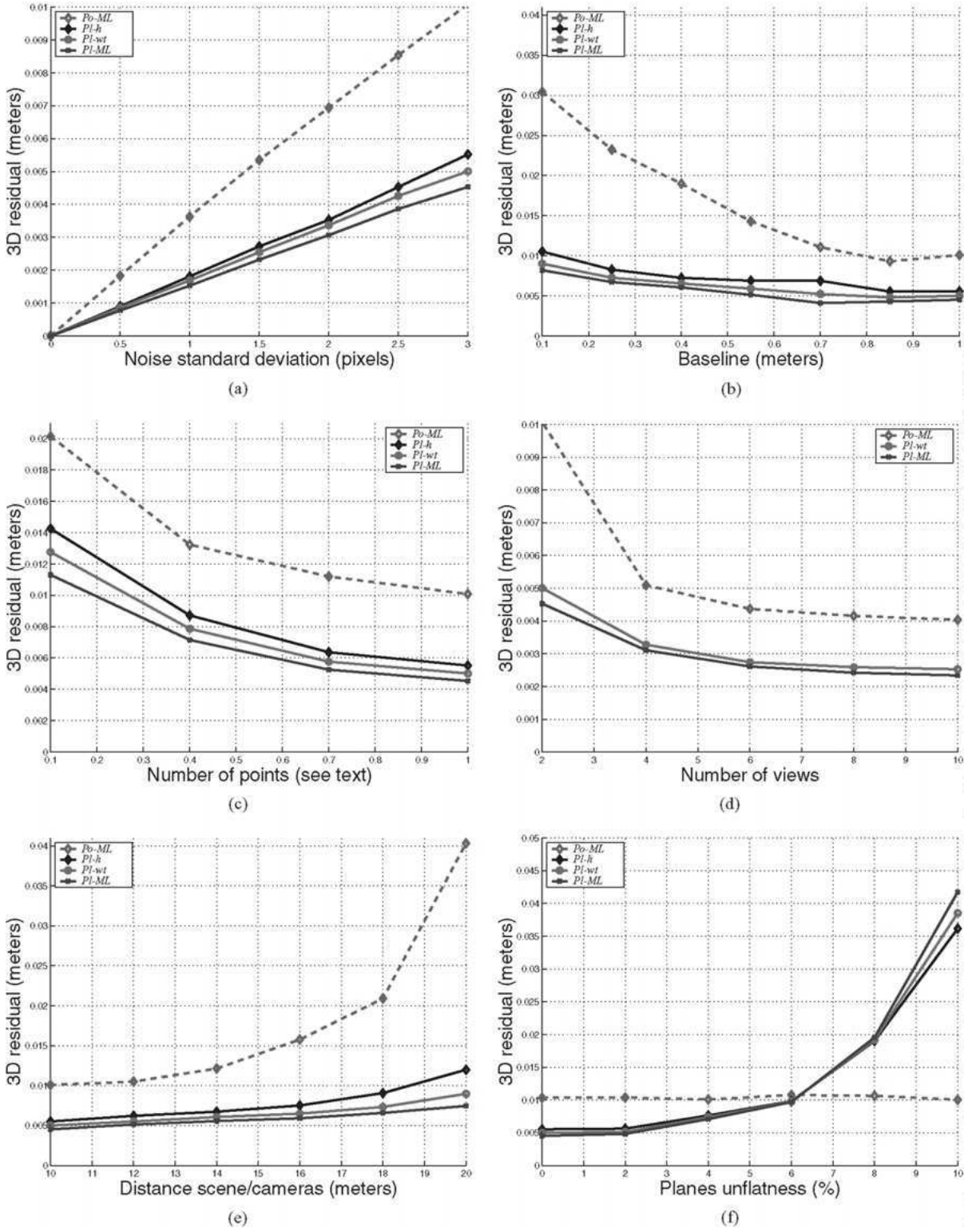


Figure 1. Comparison of the 3D residuals for different approaches versus different scene parameters. Note that method *Pl-h* is not visible on (d) since it works with two views only.

Table 5. Breakdown ratio ε for different scene configurations (image noise, number of views, distance scene/cameras).

	3 m (%)	10 m (%)	20 m (%)
$n = 2$			
1 pixel	0.5	2	4
3 pixels	2	6	9
$n = 10$			
1 pixel	0.3	1.2	3
3 pixels	1.3	4	8

$\varepsilon = 6\%$, recalling that each plane of the cube is 1 square meter. The value of ε depends on all scene parameters.

Table 5 shows values of ε established experimentally for various scene parameters. We observe that the less stable the configuration is the higher is ε, i.e. the more important is the incorporation of multi-coplanarity constraints, even if the scene is not perfectly piecewise planar.

The values of one or several percent in Table 5 represent relatively large variations which are superior to those of a great majority of approximately planar real surfaces. Consequently, we can say that there are many cases when a method using piecewise planarity will perform better than any method based on individual point reconstruction.

Similar results with other point- and plane-based methods have been obtained in Bartoli and Sturm (2001). We have also performed similar experiments in the calibrated case, i.e. the reconstructions obtained are Euclidean, and we observed that this does not change the results significantly. This can be explained by the fact that the optimization criterion is image-based, and so invariant to projective transformation (such as the upgrade from projective to metric space).

Comparing the Motion Estimates. We compare the results on the motion parameters provided by the different methods. We use the same experimental setup as previously. The quality of the estimated motion is measured as follow. We extract the n projection centers C_i of the estimated camera matrices and compute the 3D residual of their Euclidean distances to the ground-truth projection centers $\underline{C}_i$: $E_{\text{motion}} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} d^2(\mathsf{H}C_i, \underline{C}_i)}$. The 3D homography H is estimated as in the previous case, using non-linear minimization of E, i.e. using estimated to ground-truth point correspondences (estimating it with corresponding centers of projection would be highly sensitive to noise, due to the low number of data). We measure the median value of E_{motion} over 100 trials.

Let us describe the different experimental situations and results obtained.

Added Image Noise (Fig. 2(a)): The standard deviation of added image noise is varied from 0 to 3 pixels;

Number of Views (Fig. 2(b)): The number of views is varied from 2 to 10, a 3 pixels standard deviation noise is added.

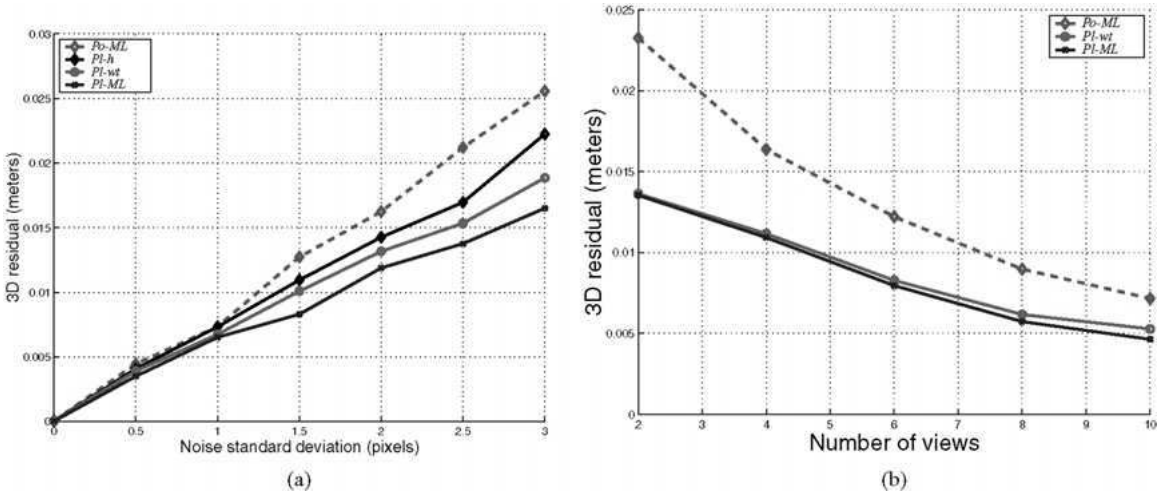


Figure 2. Comparison of the 3D residuals for the motion for different approaches versus different scene parameters. Note that method *Pl-h* is not visible on (b) since it works with two views only.

As already observed for the results on the structure, the method *Po-ML* that do not use coplanarity information performs worse than the others. The method *Pl-ML* performs better than *Pl-wt* and the method *Pl-h* performs worse than *Pl-wt*. We observe that the gap between plane-based methods *Pl-* and the point-based method *Po-ML* is reduced compared to the error estimated on the structure. In all cases, we also observe that the error measure obtained is worse than for the structure. This is due to the fact that the homography mapping the reconstruction result to the ground-truth data is estimated by minimizing the criterion E, based on the structure only.

6. Results Using Real Images

In this section, we present the reconstruction results obtained using the images shown in Fig. 3. Similar results have been obtained with other images (see Bartoli et al., 2001). We describe the different steps followed to perform a complete reconstruction, from the images to the 3D textured model. Table 6 shows the reprojection errors obtained at various stages of the process.

Structure and Motion Initialization. This has been obtained using image point matches given manually. We perform a partial reconstruction from two images using the method (Hartley, 1995; Hartley and Sturm, 1997) and incrementally add the other images to obtain the complete structure and motion. We then run a bundle adjustment to minimize the reprojection error and to obtain the maximum likelihood estimate for an unconstrained structure.

Multi-coplanarity Constraints. These relationships are established semi-automatically using plane homographies. The user provides three image points matched in at least one other view to obtain a first guess for the plane. The other points lying on this plane are then automatically detected. The user may interact to correct badly clustered points and add points visible in only one view.

Table 6. Reprojection errors (pixel) and number of iterations of non-linear optimizers at various stages of the reconstruction process. MLE stands for Maximum Likelihood Estimator.

Space	Approach	Step	Rep. error (pixels)	No. of iterations
Projective	Unconstrained	Init.	3.86	–
		MLE	1.07	7
	Constrained	Init.	1.90	–
		MLE	1.20	3
Metric	Unconstrained	MLE	2.69	6
	Constrained	Init.	3.86	–
		MLE	1.43	9

Constrained Refinement of Structure and Motion. From the previous data, the structure is parameterized as described in Section 3 and the maximum likelihood estimate for constrained structure and motion of Section 3.3 is computed. According to Table 4, points visible in only one view and constrained to lie on two or more planes are reconstructed and involved in the optimization process.

Structure Completion. Points appearing in only one view and lying on one plane are then reconstructed. The structure is complete in the sense that no more points will be further added. Figure 5 shows structure reprojection on an original image.

Calibration. The metric structure is obtained via the self-calibration process described in Section 4.1 and the reprojection error is minimized while enforcing the multi-coplanarity constraints as indicated in Section 4.4. Figure 4 shows different views of the recovered structure and the positioning of the cameras and Fig. 5 the reprojection of the model in two original images. For the intrinsic parameters of each camera, only the focal lengths are involved. It appears that also including principal points does not change significantly the results.

Texture Maps. The texture mapping requires the user to provide a polygonal delineation for each planar facet

Figure 3. Images used to validate the method.

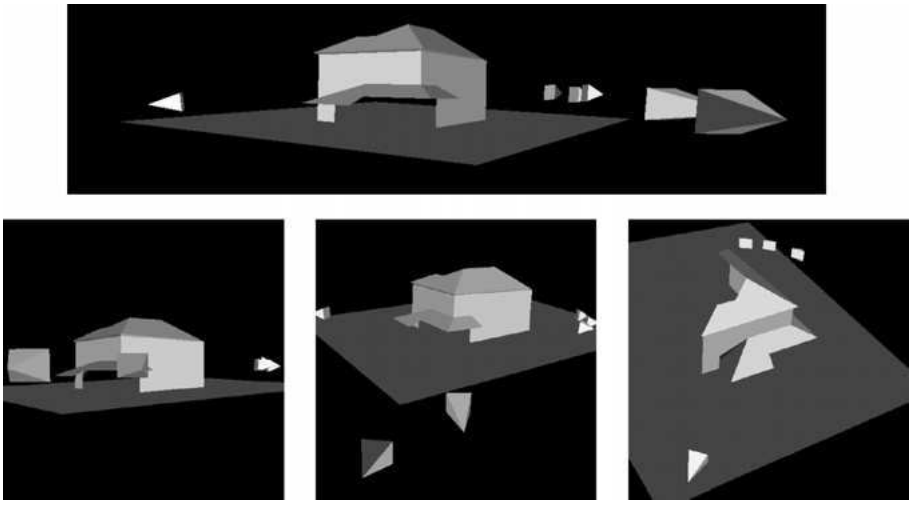


Figure 4. Recovered metric structure and motion. The structure is shown as a set of planar polygons while the different cameras (the motion) are represented by pyramids. The height of a pyramid is proportional to the recovered focal length of the camera. The bottom-right image shows a rendering from above the point of view of the right image of Fig. 5.

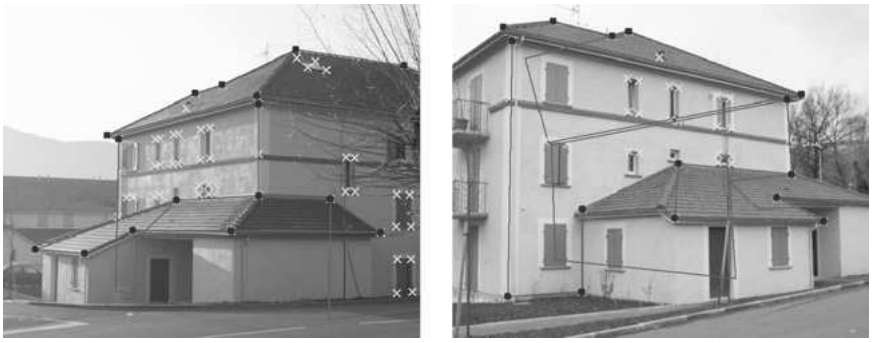


Figure 5. Reprojection of the recovered model onto the original images. The yellow crosses indicate the position of the reprojected point features.

in one of the images. The texture maps are then extracted and perspectively corrected using calibrated projection matrices and bicubic interpolation. Figure 6 shows different views of the recovered textured model.

Quality Assessment. We have performed several measures on the metric reconstruction "before" and "after" the constrained optimization process (i.e. reflecting the changes when using the method described in this paper). Two kinds of quantity are significant: length ratios and angles. Table 7 shows measures of such quantities. In this table, σ_1 and σ_2 are the variances of the length of respectively the height and width of the largest windows on the two walls, whereas μ is the mean of $1 - 2\alpha_i/\pi$ where α_i are the measures of right angles. The values given in Table 7 show that the metric reconstruction obtained with our method is clearly of superior quality than the unconstrained one.

Table 7. Metric measures on the Euclidean reconstruction "before" and "after" the constrained optimization. The lower λ_1, λ_2 and μ (see text) are, the better the reconstruction is.

	σ_1	σ_2	μ
Before	0.0489	0.0254	0.1032
After	0.0102	0.0168	0.0720

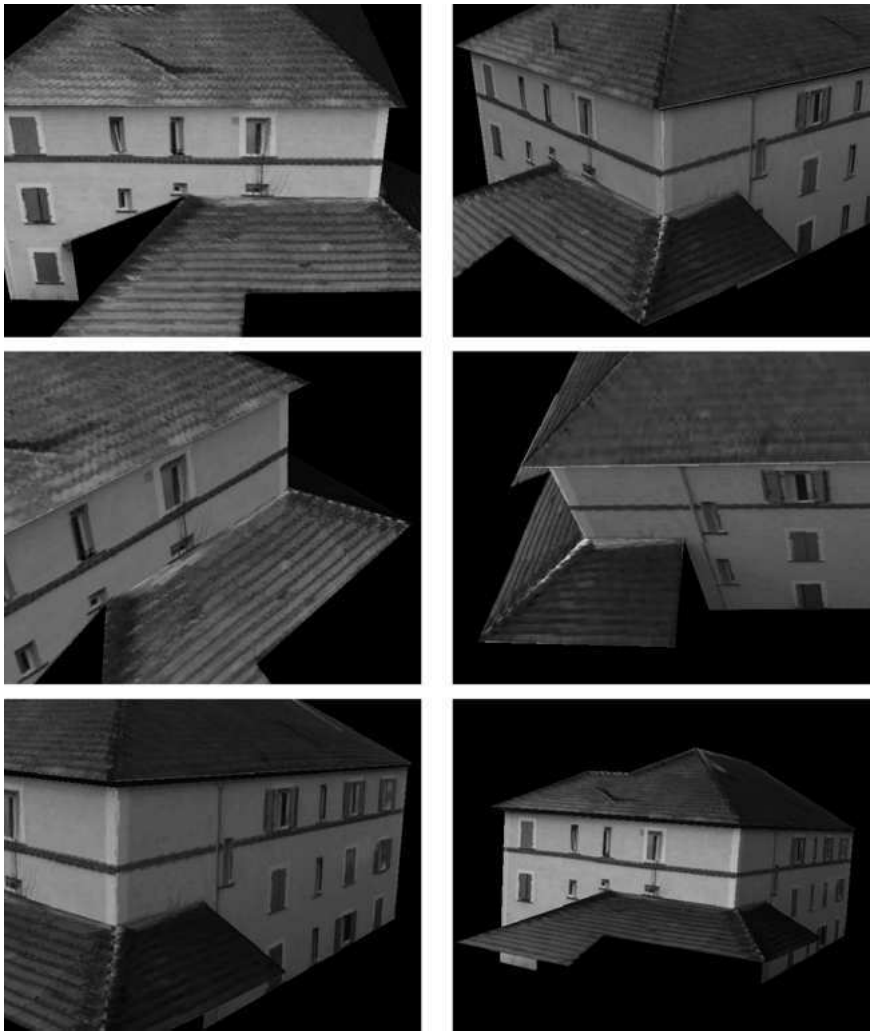


Figure 6. Different views of the textured model. Note that artefacts may be induced by possibly unmodeled non-planar parts of the surfaces, e.g. the pole bulging out of the roof in the top-right image is wrongly mapped to the roof plane, and is therefore distorted in other views, e.g. the top-left one.

7. Conclusions

We have presented an hybrid approach that draws on the strengths of both the traditional feature- and primitive-based approaches, i.e. the reconstruction is accurate and the recovered model allows to produce photorealistic rendering. More precisely, we focus on the case of points and planes related by multi-coplanarity constraints and on the design of a *constrained structure and motion* maximum likelihood estimator in both the projective and the metric cases. This maximum likelihood estimator uses a *minimal* parameterization of scene structure, enforcing underlying geometric constraints and a quasi-minimal parameterization of motion.

Experimental results on simulated data show that the quality of the reconstruction obtained with our method is clearly superior to those of traditional feature-based methods, in a large variety of experimental configurations, and for both structure and motion. We also consider the case when surfaces are only approximately planar and experimentally determined breakpoints of plane unflatness above which the incorporation of multi-coplanarity constraints makes the estimation less reliable.

The method is validated using real images. The results are convincing, in terms of both rendering quality and accuracy of metric values compared to a feature-based method.

The implementation of our methods comprises modules for unconstrained projective reconstruction ("linear" ones and bundle adjustment), constrained projective reconstruction (initialization and optimization), self-calibration ("linear" method and optimization), as well as constrained Euclidean reconstruction (initialization and bundle adjustment).

Notes

1. Note that this is very different from the hybrid approach of Debevec et al. (1996) which is actually primitive-based.
2. This is not true if the planes form a pencil.

References

Alon, J. and Sclaroff, S. 2000. Recursive estimation of motion and planar structure. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, Hilton Head Island, South Carolina, USA, pp. 550–556.

Atkinson, K.B. (Ed.). 1996. *Close Range Photogrammetry and Machine Vision*. Whittles Publishing.

Baillard, C. and Zisserman, A. 1999. Automatic reconstruction of piecewise planar models from multiple views. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, Fort Collins, Colorado, USA, IEEE Computer Society Press: Los Alamitos, CA, pp. 559–565.

Bartoli, A. and Sturm, P. 2001. Constrained structure and motion from *N* views of a piecewise planar scene. In *Proceedings of the First International Symposium on Virtual and Augmented Architecture (VAA'01)*, Dublin, Ireland, pp. 195–206.

Bartoli, A., Sturm, P., and Horaud, R. 2001. Projective structure and motion from two views of a piecewise planar scene. In *Proceedings of the 8th International Conference on Computer Vision*, Vancouver, Canada, Vol. 1, pp. 593–598.

Beardsley, P., Torr, P., and Zisserman, A. 1996. 3D model acquisition from extended image sequences. In *Proceedings of the 4th European Conference on Computer Vision*, Cambridge, England, B. Buxton and R. Cipolla (Eds.), Vol. 1065 of Lecture Notes in Computer Science. Springer-Verlag: Berlin, pp. 683–695.

Berthilsson, R. and Heyden, A. 1998. Recognition of planar point configurations using the density of affine shape. In *Proceedings of the 6th International Conference on Computer Vision*, Bombay, India, pp. 72–88.

Cross, G. and Zisserman, A. 2000. Surface reconstruction from multiple views using apparent contours and surface texture. In *NATO Advanced Research Workshop on Confluence of Computer Vision and Computer Graphics*, Ljubljana, Slovenia, A. Leonardis, F. Solina, and R. Bajcsy (Eds.), pp. 25–47.

Debevec, P.E., Taylor, C.J., and Malik, J. 1996. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In *SIGGRAPH'96*, New Orleans.

Dick, A.R., Torr, P.H.S., Ruffle, S.F., and Cipolla, R. 2001. Combining single view recognition and multiple view stereo for architectural scenes. In *Proceedings of the 8th International Conference on Computer Vision*, Vancouver, Canada.

Faugeras, O. and Lustman, F. 1988. Motion and structure from motion in a piecewise planar environment. *International Journal of Pattern Recognition and Artificial Intelligence*, 2(3):485–508.

Fornland, P. and Schnörr, C. 1997. A robust and convergent iterative approach for determining the dominant plane from two views without correspondence and calibration. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, Puerto Rico, USA, IEEE (Ed.), pp. 508–513.

Gill, P.E., Murray, W., and Wright, M.H. 1981. *Practical Optimization*. Academic Press: New York.

Gortler, S.J., Grzeszczuk, R., Szeliski, R., and Cohen, M. 1996. The lumigraph. In *Proceedings of SIGGRAPH*, New Orleans, LA, pp. 43–54.

Hartley, R. 1995. In defence of the 8-point algorithm. In *Proceedings of the 5th International Conference on Computer Vision*, Cambridge, MA, USA, pp. 1064–1070.

Hartley, R. and Sturm, P. 1997. Triangulation. *Computer Vision and Image Understanding*, 68(2):146–157.

Heyden, A. and Åström, K. 1995. A canonical framework for sequences of images. In *Workshop on Representation of Visual Scenes*, Cambridge, MA, USA, pp. 45–52.

Heyden, A. and Åström, K. 1998. Minimal conditions on intrinsic parameters for euclidean reconstruction. In *Proceedings of the Third Asian Conference on Computer Vision*, Hong Kong, Vol. II, pp. 169–176.

Horn, B.K.P., Hilden, H.M., and Negahdaripour, S. 1988. Closed-form solution of absolute orientation using orthonormal matrices. *Journal of the Optical Society of America A*, 5(7):1127–1135.

Kahl, F., Triggs, B., and Åström, K. 2000. Critical motions for auto-calibration when some intrinsic parameters can vary. *Journal of Mathematical Imaging and Vision*, 13(2):131–146.

Kutulakos, K.N. and Seitz, S.M. 1999. A theory of shape by space carving. In *Proceedings of the 7th International Conference on Computer Vision*, Kerkyra, Greece, Vol. 1, pp. 307–314.

Lang, F. and Förstner, W. 1996. 3D-city modeling with a digital one-eye stereo system. In *Proceedings of the XVIII ISPRS-Congress*, Vienna, Austria.

Levoy, M. and Hanrahan, P. 1996. Light field rendering. In *Proceedings of SIGGRAPH*, New Orleans, LA, pp. 31–42.

Malis, E. and Cipolla, R. 2000. Multi-view constraints between collineations: Application to self-calibration from unknown planar structures. In *Proceedings of the 6th European Conference on Computer Vision*, Dublin, Ireland, D. Vernon (Ed.), Vol. 1843 of Lecture Notes in Computer Science. Springer-Verlag: Berlin, pp. 610–624.

Maybank, S.J. and Faugeras, O.D. 1992. A theory of self calibration of a moving camera. *International Journal of Computer Vision*, 8(2):123–151.

McGlone, C. 1996. Bundle adjustment with geometric constraints for hypothesis evaluation. In *Proceedings of the XVIII ISPRS-Congress*, Vienna, Austria, pp. 529–534.

Niem, W. 1994. Robust and fast modelling of 3D natural objects from multiple views. In *Proceedings of the SPIE Conference on*

Image and Video Processing II, San Jose, USA, Vol. 2182, pp. 388–397.

Pollefeys, M. 1999. Self-calibration and metric 3D reconstruction from uncalibrated image sequences. Ph.D. Thesis, Katholieke Universiteit Leuven, Belgium, Faculteit Toegepaste Wetenschappen, Arenbergkasteel, B-3001 Heverlee, Belgium.

Pollefeys, M., Koch, R., and Van Gool, L. 1998. Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. In *Proceedings of the 6th International Conference on Computer Vision*, Bombay, India, pp. 90–95.

Pollefeys, M., Vergauwen, M., and Van Gool, L. 2000. Automatic 3D modeling from image sequences. In *Proceedings of the XIX ISPRS-Congress*, Amsterdam, the Netherlands, Vol. B5, pp. 619–626.

Seitz, S.M. and Dyer, C.R. 1997. Photorealistic scene reconstruction by voxel coloring. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, Puerto Rico, USA, IEEE Computer Society Press: Los Alamitos, CA, pp. 1067–1073.

Sinclair, D. and Blake, A. 1996. Quantitative planar region detection. *International Journal of Computer Vision*, 18(1):77–91.

Slama, C.C. (Ed.). 1980. *Manual of Photogrammetry*, Fourth edn. American Society of Photogrammetry and Remote Sensing: Falls Church, Virginia, USA.

Streilein, A. and Hirschberg, U. 1995. Integration of digital photogrammetry and CAAD: Constraint-based modeling and semi-automatic measurement. In *Proceedings of the International CAAD Futures Conference*, Singapore.

Sturm, P. 1999. Critical motion sequences for the self-calibration of cameras and stereo systems with variable focal length. In *Proceedings of the Tenth British Machine Vision Conference*, Nottingham, England, T. Pridmore and D. Elliman (Eds.), British Machine Vision Association, pp. 63–72.

Sturm, P. 2000. A case against Kruppas equations for camera self-calibration. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 22(10):1199–1204.

Sturm, P. and Triggs, B. 1996. A factorization based algorithm for multi-image projective structure and motion. In *Proceedings of the 4th European Conference on Computer Vision*, Cambridge, England, B. Buxton and R. Cipolla (Eds.), Vol. 1065 of Lecture Notes in Computer Science. Springer-Verlag: Berlin, pp. 709–720.

Szeliski, R. 1993. Rapid octree construction from image sequences. *Computer Vision, Graphics and Image Processing*, 58(1):23–32.

Szeliski, R. and Torr, P.H.S. 1998. Geometrically constrained structure from motion: Points on planes. In *3D Structure from Multiple Images of Large-Scale Environments (SMILE'98)*. Springer Verlag: Berlin.

Tarel, J.-P. and Vézien, J.-M. 1995. A generic approach for planar patches stereo reconstruction. In *Proceedings of the Scandinavian Conference on Image Analysis*, Uppsala, Sweden, pp. 1061–1070.

Triggs, B. 1997. Autocalibration and the absolute quadric. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, Puerto Rico, USA, IEEE Computer Society Press: Los Alamitos, CA, pp. 609–614.

Triggs, B. 1998a. Autocalibration from planar scenes. In *Proceedings of the 5th European Conference on Computer Vision*, Freiburg, Germany.

Triggs, B. 1998b. Optimal estimation of matching constraints. In *3D Structure from Multiple Images of Large-Scale Environments (SMILE'98)*, Springer Verlag: Berlin.

Triggs, B., McLauchlan, P.F., Hartley, R.I., and Fitzgibbon, A. 2000. Bundle ajustment—A modern synthesis. In *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*, Corfu, Greece, B. Triggs, A. Zisserman, and R. Szeliski (Eds.), Vol. 1883 of Lecture Notes in Computer Science. Springer-Verlag: Berlin, pp. 298–372.

Viéville, T., Zeller, C., and Robert, L. 1995. Using collineations to compute motion and structure in an uncalibrated image sequence. *International Journal of Computer Vision*, 20(3):213–242.

Xu, G., Terai, J.-I., and Shum, H.-Y. 2000. A linear algorithm for camera self-calibration, motion and structure recovery for multi-planar scenes from two perspective images. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, Hilton Head Island, South Carolina, USA.

Zhang, Z. 1998. Determining the epipolar geometry and its uncertainty: A review. *International Journal of Computer Vision*, 27(2):161–195.

Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computer Vision and Image Understanding
ELSEVIER
Computer Vision and Image Understanding 97 (2005) 86–102
www.elsevier.com/locate/cviu

The geometric error for homographies☆

Ondřej Chum,[a,*] Tomáš Pajdla,[a] and Peter Sturm[b]

[a] *Center for Machine Perception, Department of Cybernetics, Czech Technical University in Prague, Faculty of Electrical Engineering, Karlovo náměstí 13, 121 35 Praha 2, Czech Republic*
[b] *INRIA Rhône-Alpes, 655 Avenue de l'Europe, Montbonnot 38330, France*

Received 23 July 2003; accepted 10 March 2004
Available online 22 April 2004

Abstract

We address the problem of finding optimal point correspondences between images related by a homography: given a homography and a pair of matching points, determine a pair of points that are exactly consistent with the homography and that minimize the geometric distance to the given points. This problem is tightly linked to the triangulation problem, i.e., the optimal 3D reconstruction of points from image pairs. Our problem is non-linear and iterative optimization methods may fall into local minima. In this paper, we show how the problem can be reduced to the solution of a polynomial of degree eight in a single variable, which can be computed numerically. Local minima are thus explicitly modeled and can be avoided. An application where this method significantly improves reconstruction accuracy is discussed. Besides the general case of homographies, we also examine the case of affine transformations, and closely study the relationships between the geometric error and the commonly used Sampson's error, its first order approximation. Experimental results comparing the geometric error with its approximation by Sampson's error are presented.

Keywords: Homography; Geometric error; Sampson's error; Triangulation

☆ The authors were supported by Grants GACR 102/01/0971, FP5 EU IST-2001-39184, Aktion 34p24, MSM 212300013, CTU0306013, and STINT Dur IG2003-2 062. We thank Mirko Navara for helping us with the proofs.
* Corresponding author.
E-mail addresses: chum@cmp.felk.cvut.cz (O. Chum), pajdla@cmp.felk.cvut.cz (T. Pajdla), Peter.Sturm@inrialpes.fr (P. Sturm).

1077-3142/$ - see front matter
doi:10.1016/j.cviu.2004.03.004

1. Introduction

Homographies are used in many applications, e.g., in mosaicing [1] or wide baseline stereo matching [2,3]. In many applications we also need to compute the error (or the distance) of a point correspondence with respect to a given homography $\mathtt{H}$. This is necessary for instance in RANSAC [4], a commonly used robust estimation algorithm. Some applications may require not only to compute the distance of a given point correspondence to the model of homography but actually need to determine points, which are consistent with the given homography and are in a small neighborhood of the measured, thus noisy, given points.

This work addresses the problem of finding optimal point correspondences between images related by an homography: given a known homography and a pair of matching noisy points, determine a pair of points that are exactly consistent with the homography and that minimize the geometric distance to the given noisy points. There are two approaches to achieve such a goal [5]: (1) non-linear optimization using iterative methods and (2) parametric approach, where the solution is parametrized so that it automatically satisfies the given constraint. The paper concentrates on the latter strategy.

A similar problem, based on the geometric error for the epipolar geometry, has been addressed by Hartley and Sturm [6]. The geometric error for homographies was introduced by Sturm [7, Appendix B], and independently derived by Chum and Pajdla in [8,9]. In this paper, previous results are reviewed from a common perspective, the derivation of the geometric error for homographies is described and a mathematical proof of its correctness given. Furthermore, we discuss the commonly used approximation of the geometric error, Sampson's error. Links between the two are studied in detail, for the general case of homographies, as well as the case of affine transformations between images.

The rest of the paper is structured as follows. Basic concepts are introduced in Section 2. Section 3 contains the derivation of the formulae for the geometric error. In Section 4, Sampson's approximation is derived and studied. Geometric properties of both error measures are studied in Section 5. Experiments are presented in Section 6. An application of the proposed method is described in Section 7 and conclusions are given in Section 8.

2. Basic concepts

We assume that a planar homography $\mathtt{H}$ [10] and a noisy correspondence $\mathbf{x} \leftrightarrow \mathbf{x}'$ measured in the images are given. Let the homogeneous coordinates of the corresponding points be $\mathbf{x} = (x, y, 1)^{\mathrm{T}}$ and $\mathbf{x}' = (x', y', 1)^{\mathrm{T}}$ and the homography be represented by the (regular) matrix

$$\mathtt{H} = \begin{pmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{pmatrix}.$$

There are several possible ways to measure the "error" of that point correspondence with respect to the homography. We will mention the geometric error and Sampson's approximation of it.

Supposing the Gaussian noise model for perturbations of image coordinates, the maximum likelihood estimation of the position of the noise-free correspondence $\hat{\mathbf{x}} \leftrightarrow \mathtt{H}\hat{\mathbf{x}}$ is obtained by minimizing the *geometric error* $d_\perp^2 = d(\mathbf{x}, \hat{\mathbf{x}})^2 + d(\mathbf{x}', \mathtt{H}\hat{\mathbf{x}})^2$ over all $\hat{\mathbf{x}}$ (Fig. 1). This error measure could be thought of as the Euclidean distance of point $\mathbf{X} = (x, y, x', y') \in \mathbb{R}^4$, representing the given point correspondence, to the two-dimensional variety $\mathcal{V}_\mathtt{H}$ (Fig. 2) defined as

$$\mathcal{V}_\mathtt{H} = \{\mathbf{Y} \in \mathbb{R}^4 \,|\, \mathbf{t}(\mathbf{Y}) = \mathbf{0}\}, \tag{1}$$

where $\mathbf{t} = (t_x, t_y)^\mathtt{T}$ and

$$t_x = Y_1 h_1 + Y_2 h_2 + h_3 - Y_1 Y_3 h_7 - Y_2 Y_3 h_8 - Y_3 h_9, \tag{2}$$

$$t_y = Y_1 h_4 + Y_2 h_5 + h_6 - Y_1 Y_4 h_7 - Y_2 Y_4 h_8 - Y_4 h_9, \tag{3}$$

i.e., such $\mathbf{Y}$ represent point correspondences that are consistent with $\mathtt{H}$.

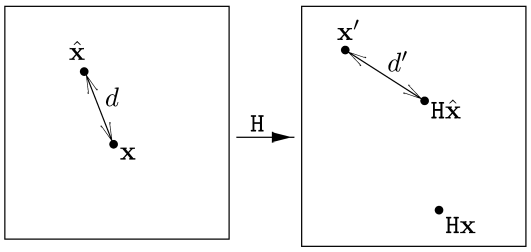


Fig. 1. Two images linked by homography $\mathtt{H}$. Points $\mathbf{x}$ and $\mathbf{x}'$ are measured points, $\hat{\mathbf{x}}$ is the point minimizing $d^2 + d'^2$ where d and d' are the distances $\mathbf{x}$ to $\hat{\mathbf{x}}$ and $\mathbf{x}'$ to $\mathtt{H}\hat{\mathbf{x}}$.

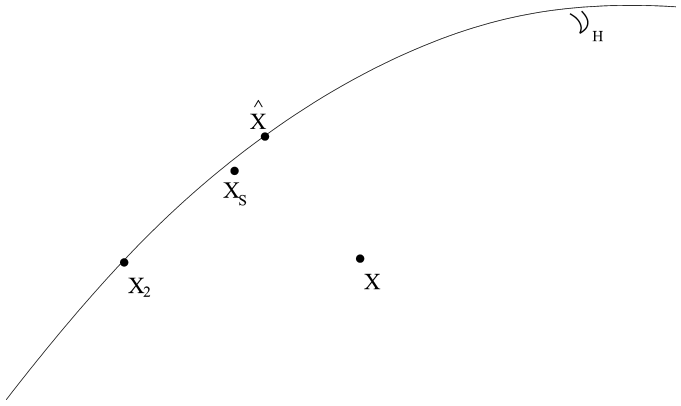


Fig. 2. The variety $\mathcal{V}_\mathtt{H}$ and points where different error measures of the measured noisy point correspondence $\mathbf{X}$ with respect to homography $\mathtt{H}$ are minimized. The geometric error is minimized at $\hat{\mathbf{X}}$, Sampson's error at $\mathbf{X}_\mathrm{S}$, and the error in the second image at $\mathbf{X}_2$.

The first-order approximation of this error measure, called *Sampson's error*, was first used by Sampson [11] for conics. The derivation of Sampson's error for homographies is described in Section 4.

The exact computation of the geometric error is equivalent to finding the point $\hat{\mathbf{X}} \in \mathbb{R}^4$ on the variety $\mathcal{V}_{\mathtt{H}}$, that minimizes the Euclidean distance to the measured point $\mathbf{X}$. We show that the geometric error can be exactly determined by solving a polynomial of degree eight.

3. The geometric error

In this section the problem of computing the geometric error is transformed, so that it reduces to finding roots of a polynomial of degree eight.

The distance of points lying on the variety $\mathcal{V}_{\mathtt{H}}$ to the measured point correspondence $\mathbf{X}$ can be written as a function of the matrix $\mathtt{H}$, the measured image points $\mathbf{x}$, $\mathbf{x}'$, and a point $\hat{\mathbf{x}}$ in the first image. If we expand the matrix multiplication, we have

$$e(\hat{\mathbf{x}}) = (x - \hat{x})^2 + (y - \hat{y})^2 + (x' - \hat{x}')^2 + (y' - \hat{y}')^2, \tag{4}$$

where

$$\hat{x}' = \frac{h_1\hat{x} + h_2\hat{y} + h_3}{h_7\hat{x} + h_8\hat{y} + h_9} \tag{5}$$

and

$$\hat{y}' = \frac{h_4\hat{x} + h_5\hat{y} + h_6}{h_7\hat{x} + h_8\hat{y} + h_9}. \tag{6}$$

Directly solving the equation $(\partial e/\partial \hat{y}) = 0$ leads to a polynomial in two variables of order four in $\hat{x}$ and order five in $\hat{y}$. The same happens for the partial derivative of e with respect to $\hat{x}$. Therefore, we first transform the images such as to lower the degree of the polynomial. We use Euclidean transformations, which do not change distances, and thus the solution of the transformed problem will be the transformed solution of the original problem.

At first we shift the points $\mathbf{x}$ and $\mathbf{x}'$ to the origin of the first and the second image, respectively. This is achieved by applying the following translations:

$$\mathtt{L} = \begin{pmatrix} 1 & 0 & -x \\ 0 & 1 & -y \\ 0 & 0 & 1 \end{pmatrix}, \quad \mathtt{L}' = \begin{pmatrix} 1 & 0 & -x' \\ 0 & 1 & -y' \\ 0 & 0 & 1 \end{pmatrix}.$$

After translating the images we have

$$\mathtt{L}'\hat{\mathbf{x}}' \sim \mathtt{L}'\mathtt{H}\mathtt{L}^{-1}\mathtt{L}\hat{\mathbf{x}}. \tag{7}$$

In this equation, $\sim$ stands for "equal up to a nonzero scale." Let $\mathtt{B} = \mathtt{L}'\mathtt{H}\mathtt{L}^{-1}$ be the homography between the transformed images and $\bar{\mathbf{x}} = \mathtt{L}\hat{\mathbf{x}}$. We can easily verify that

the first two entries in the third row of the matrix B equal the corresponding entries in $\mathtt{H}$ after applying the translations, and so

$$\mathtt{B} = \begin{pmatrix} b_1 & b_2 & b_3 \\ b_4 & b_5 & b_6 \\ h_7 & h_8 & b_9 \end{pmatrix}. \tag{8}$$

We can now rewrite (re-parameterize) the error term e as follows:[1]

$$e(\hat{\mathbf{x}}) = e(\bar{\mathbf{x}}) = \bar{x}^2 + \bar{y}^2 + \left(\frac{b_1\bar{x} + b_2\bar{y} + b_3}{h_7\bar{x} + h_8\bar{y} + b_9}\right)^2 + \left(\frac{b_4\bar{x} + b_5\bar{y} + b_6}{h_7\bar{x} + h_8\bar{y} + b_9}\right)^2. \tag{9}$$

From (9) we can observe, that solving for the minimum of e would be simple if h_8 were equal to 0, as $\partial e/\partial\bar{y}$ would be linear in $\bar{y}$ (e would be quadratic in $\bar{x}$). To achieve this situation, we simply rotate the first image appropriately: we design a rotation matrix $\mathtt{R}$ so that the homography between the rotated first image and the second image, i.e., $\mathtt{Q} = \mathtt{BR}^{-1}$, satisfies $q_8 = 0$. With

$$\mathtt{R} = \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

the rotation angle α for which $q_8 = h_7\sin(\alpha) + h_8\cos(\alpha) = 0$ is obtained as

$$\alpha = \arctan\left(-\frac{h_8}{h_7}\right). \tag{10}$$

Now we can rewrite the term e as follows:

$$e(\tilde{\mathbf{x}}) = \tilde{x}^2 + \tilde{y}^2 + \left(\frac{q_1\tilde{x} + q_2\tilde{y} + q_3}{q_7\tilde{x} + q_9}\right)^2 + \left(\frac{q_4\tilde{x} + q_5\tilde{y} + q_6}{q_7\tilde{x} + q_9}\right)^2, \tag{11}$$

where $\tilde{\mathbf{x}} = \mathtt{R}\bar{\mathbf{x}} = \mathtt{RL}\hat{\mathbf{x}}$. The partial derivative $\partial e/\partial\tilde{y}$ is linear in $\tilde{y}$. The minimum is reached in $\partial e/\partial\tilde{y} = 0$, so

$$\tilde{y} = -\frac{q_2q_3 + q_5q_6 + q_1q_2\tilde{x} + q_4q_5\tilde{x}}{q_2^2 + q_5^2 + q_9^2 + 2q_7q_9\tilde{x} + q_7^2\tilde{x}^2}. \tag{12}$$

Now we can simply substitute (12) into (11) and find the minimum of e. Solving

$$\frac{\partial e}{\partial\tilde{x}}(\tilde{x}, \tilde{y}) = 0$$

gives a polynomial of degree eight which is completely given in Appendix B. The proof of correctness of the procedure described above, i.e., the proof that a global minimum of the function e *exists* and that the partial derivatives are *defined* at it, can be found in Appendix A.

[1] The function e is parametrized not only by $\hat{\mathbf{x}}$, but also by $\mathbf{x}$, $\mathbf{x}'$, and $\mathtt{H}$. The term $e(\hat{\mathbf{x}})$ actually stands for $e(\hat{\mathbf{x}}, \mathbf{x}, \mathbf{x}', \mathtt{H})$, whereas $e(\bar{\mathbf{x}})$ stands for $e(\bar{\mathbf{x}}, \mathtt{L}\mathbf{x}, \mathtt{L}'\mathbf{x}', \mathtt{B})$.

3.1. The affine case

Eqs. (2) and (3) are linear in the entries of the mapping matrix and bilinear in the entries of $\mathbf{Y}$ when a full homography matrix is sought.

Proposition 1. *Eqs.* (2) *and* (3) *are linear in the entries of* $\mathbf{Y}$ *if and only if the mapping is affine.*

Proof. If (and only if) both $h_7 = 0$ and $h_8 = 0$, then (2) and (3) are linear in the entries of $\mathbf{Y}$. This exactly corresponds to affine mappings. □

For an affine transformation, $\mathbf{x}' = \mathtt{A}\mathbf{x}$, where

$$\mathtt{A} = \begin{pmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ 0 & 0 & 1 \end{pmatrix},$$

the geometric error can be easily obtained in closed form. As the error function

$$e = (\hat{x} - x)^2 + (\hat{y} - y)^2 + (a_1\hat{x} + a_2\hat{y} + a_3 - y')^2 + (a_4\hat{x} + a_5\hat{y} + a_6 - y')^2$$

is order of two in both $\hat{x}$ and $\hat{y}$, partial derivatives $\partial e/\partial x$ and $\partial e/\partial\hat{y}$ are linear in both $\hat{x}$ and $\hat{y}$. Solving the system $\partial e/\partial\hat{x} = 0$ and $\partial e/\partial\hat{y} = 0$ using, e.g. [12] yields (in the general case) a unique solution

$$\begin{aligned} N &= (a_5^2 + 1)(a_1^2 + 1) - 2a_1a_2a_4a_5 + a_2^2 + a_2^2a_4^2 + a_4^2, \\ \hat{x} &= \frac{1}{N}\big(a_1(a_2a_5a_6 - a_3 - a_3a_5^2) + a_2a_3a_4a_5 - (a_4a_6)(a_2^2 + 1) + x(a_5^2 + a_2^2 + 1) \\ &\quad - y(a_1a_2 + a_4a_5) + x'(a_1 + a_1a_5^2 - a_2a_4a_5) + y'(a_4 - a_1a_2a_5 + a_2^2a_4)\big), \\ \hat{y} &= \frac{1}{N}\big(a_1(a_3a_4a_5 + a_2a_4a_6) - (a_5a_6)(a_1^2 + 1) - (a_2a_3)(a_4^2 + 1) - x(a_1a_2 + a_4a_5) \\ &\quad + y(a_1^2 + a_4^2 + 1) + x'(a_2 - a_1a_4a_5 + a_2a_4^2) + y'(a_5 - a_1a_2a_4 + a_1^2a_5)\big). \end{aligned}$$

4. Sampson's error

To find the closest point on the variety $\mathcal{V}_{\mathtt{H}}$ to our measured correspondence $\mathbf{x} \leftrightarrow \mathbf{x}'$, or $\mathbf{X} \in \mathbb{R}^4$, requires solving a polynomial of degree eight, which is computationally expensive. Another possibility is to compute an approximation of the geometric error.

We can use the first-order Taylor approximation of t_x and t_y by their tangent hyperplanes in the measured point $\mathbf{X}$. Let $\mathbf{J}$ be the Jacobian matrix

$$\mathbf{J}_{\mathtt{H},\mathbf{x},\mathbf{x}'} = \begin{pmatrix} \frac{\partial t_x(\mathbf{X})}{\partial x} & \frac{\partial t_x(\mathbf{X})}{\partial y} & \frac{\partial t_x(\mathbf{X})}{\partial x'} & \frac{\partial t_x(\mathbf{X})}{\partial y'} \\ \frac{\partial t_y(\mathbf{X})}{\partial x} & \frac{\partial t_y(\mathbf{X})}{\partial y} & \frac{\partial t_y(\mathbf{X})}{\partial x'} & \frac{\partial t_y(\mathbf{X})}{\partial y'} \end{pmatrix}.$$

To simplify $\mathbf{J}$ for further use, we apply the observations that $\frac{\partial t_x(\mathbf{X})}{\partial y'} = 0$, $\frac{\partial t_y(\mathbf{X})}{\partial x'} = 0$, and $\frac{\partial t_x(\mathbf{X})}{\partial x'} = \frac{\partial t_y(\mathbf{X})}{\partial y'}$. We obtain $\mathbf{J}$ in the following form:

$$\mathbf{J}_{\mathrm{H},\mathbf{x},\mathbf{x}'} = \mathbf{J} = \begin{pmatrix} j_1 & j_2 & j_3 & 0 \\ j_4 & j_5 & 0 & j_3 \end{pmatrix},$$

where $j_1 = h_1 - h_7x'$, $j_2 = h_2 - h_8x'$, $j_3 = -h_7x - h_8y - h_9$, $j_4 = h_4 - h_7y'$, and $j_5 = h_5 - h_8y'$ are the respective partial derivatives. Then, the first order Taylor approximation of $\mathbf{t}$ is

$$\tilde{\mathbf{t}}(\tilde{\mathbf{X}}) = \mathbf{t}(\mathbf{X}) + \mathbf{J}(\tilde{\mathbf{X}} - \mathbf{X}). \tag{13}$$

The approximate solution (Sampson's error) might be found as the closest point to $\mathbf{X}$ on the two-dimensional variety $\mathcal{V}_S$ defined as follows:

$$\mathcal{V}_S = \{\tilde{\mathbf{X}} \in \mathbb{R}^4 \mid \tilde{\mathbf{t}}(\tilde{\mathbf{X}}) = \mathbf{0}\}.$$

As $\mathcal{V}_S$ is linear, the solution is given by

$$\mathbf{X}_S = \mathbf{J}^+\mathbf{t}(\mathbf{X}) + \mathbf{X},$$

where $\mathbf{J}^+ = \mathbf{J}^{\mathrm{T}}(\mathbf{J}\mathbf{J}^{\mathrm{T}})^{-1}$ is the pseudo-inverse of the Jacobian $\mathbf{J}$.

If we have a closer look at the function $\tilde{\mathbf{t}}$ (Eq. (13)), we can observe that it is similar to the function $\mathbf{t}$, but linear in the entries of $\tilde{\mathbf{X}}$. If we examine it in more detail, we find that Sampson's error is in fact the geometric error for the affine transformation, that locally approximates the homography $\mathtt{H}$. The affine approximation $\mathbf{x}' = A_{\mathtt{H}}\mathbf{x}$ of the homography $\mathtt{H}$ in the measured points $\mathbf{x} \leftrightarrow \mathbf{x}'$ is as follows:

$$A_{\mathtt{H}} = \frac{1}{j_3}\begin{pmatrix} j_1 & j_2 & -j_1x - j_2y - j_3x' + t_x(\mathbf{X}) \\ j_4 & j_5 & -j_4x - j_5y - j_3y' + t_y(\mathbf{X}) \\ 0 & 0 & j_3 \end{pmatrix}.$$

The affine transformation $A_{\mathtt{H}}$ has the same partial derivatives as $\mathtt{H}$ in the measured point $\mathbf{X}$. Let us give a geometric interpretation of $A_{\mathtt{H}}$. The construction of $A_{\mathtt{H}}$ based on the points that are mapped identically by both, the homography $\mathtt{H}$ and its affine approximation $A_{\mathtt{H}}$, gives an illustrative explanation. These points are given as the fixed points of $A_{\mathtt{H}}^{-1}\mathtt{H}$. The eigenvectors and eigenvalues of $A_{\mathtt{H}}^{-1}\mathtt{H}$ are in the general case (using [12]) $\mathbf{v_1} = (h_8, -h_7, 0)^{\mathrm{T}}$, $\lambda_1 = 1$, $\mathbf{v_2} = (0, h_7x + h_8y, h_8)^{\mathrm{T}}$, $\lambda_2 = 1$, and $\mathbf{v_3} = \mathtt{H}^{-1}\mathbf{x}'$, $\lambda_3 \neq 1$. Hence, there is a line of fixed points, passing through the points $\mathbf{v_1}$ and $\mathbf{v_2}$ (including $\mathbf{x} = x\mathbf{v_1} + \mathbf{v_2}$) and a fixed point $\mathtt{H}^{-1}\mathbf{x}'$. The point $\mathbf{v_1}$ is the only point (in general) that is mapped by the non-affine homography from the line at infinity to the line at infinity, i.e., $\mathtt{H}^{-1}(\mathtt{H}^{-\mathrm{T}}(0,0,1)^{\mathrm{T}} \times (0,0,1)^{\mathrm{T}})$, satisfying $(h_7, h_8, h_9)\mathbf{v_1} = 0$.

5. Geometric properties

An important property of an error measure is its independence of the choice of the Cartesian coordinate system in the images. In this section, we study the behavior of

the discussed error measures under the application of rigid transformations to the images. The originally formulated relation $\mathbf{x}' \sim \mathtt{H}\mathbf{x}$ changes to

$$\mathtt{T}'\mathbf{x}' \sim (\mathtt{T}'\mathtt{H}\mathtt{T}^{-1})\mathtt{T}\mathbf{x},$$

where $\mathtt{T}$ and $\mathtt{T}'$ represent the rigid image transformations of the first and the second image, respectively.

As distances in the images are not affected by rigid transformations and the new homography links the transformed points, the geometric error remains the same. We already used this property in Section 3.

Proposition 2. *The Jacobian* $\mathbf{J}$ *is covariant to any affine transformation of the images. Let* $\mathtt{T}$ *and* $\mathtt{T}'$ *be affine transformations of the first and second image respectively, and* $\mathtt{H}_1$, $\mathtt{H}_2$ *be homographies satisfying* $\mathbf{J}_{\mathtt{H}_1,\mathbf{x},\mathbf{x}'} \sim \mathbf{J}_{\mathtt{H}_2,\mathbf{x},\mathbf{x}'}$. *Then* $\mathbf{J}_{\mathtt{T}'\mathtt{H}_1\mathtt{T}^{-1},\mathtt{T}\mathbf{x},\mathtt{T}'\mathbf{x}'} \sim \mathbf{J}_{\mathtt{T}'\mathtt{H}_2\mathtt{T}^{-1},\mathtt{T}\mathbf{x},\mathtt{T}'\mathbf{x}'}$.

Proof. Denote the affine transformations as

$$\mathtt{T} = \begin{pmatrix} t_1 & t_2 & t_3 \\ t_4 & t_5 & t_6 \\ 0 & 0 & 1 \end{pmatrix} \quad \text{and} \quad \mathtt{T}' = \begin{pmatrix} t'_1 & t'_2 & t'_3 \\ t'_4 & t'_5 & t'_6 \\ 0 & 0 & 1 \end{pmatrix}.$$

The Jacobian $\mathbf{J}$ after transforming the first and the second image, respectively can be expressed in terms of the transformations $\mathtt{T}$ and $\mathtt{T}'$ and the original Jacobian $\mathbf{J}$ as follows:

$$\mathbf{J}_{\mathtt{T}'\mathtt{H},\mathbf{x},\mathtt{T}'\mathbf{x}'} = \begin{pmatrix} j_1 t'_1 + j_4 t'_2 & j_2 t'_1 + j_5 t'_2 & j_3 & 0 \\ j_1 t'_4 + j_4 t'_5 & j_2 t'_4 + j_5 t'_5 & 0 & j_3 \end{pmatrix} \quad \text{and} \tag{14}$$

$$\mathbf{J}_{\mathtt{H}\mathtt{T}^{-1},\mathtt{T}\mathbf{x},\mathbf{x}'} = \begin{pmatrix} \frac{-j_2 t_4 + j_1 t_5}{t_1 t_5 - t_4 t_2} & \frac{j_2 t_1 - j_1 t_2}{t_1 t_5 - t_4 t_2} & j_3 & 0 \\ \frac{-j_5 t_4 + j_4 t_5}{t_1 t_5 - t_4 t_2} & \frac{j_5 t_1 - j_4 t_2}{t_1 t_5 - t_4 t_2} & 0 & j_3 \end{pmatrix}. \tag{15}$$

From Eqs. (14) and (15) it follows, that the transformed Jacobian can be expressed as a function of the original Jacobian $\mathbf{J}$ and the affine transformations $\mathtt{T}$ and $\mathtt{T}'$. The proposition is a straightforward consequence of this fact. □

Proposition 3. *Sampson's error measure is invariant to the choice of Cartesian coordinate system, i.e., any rotation or translation of the images does not affect it.*

Proof. From its definition, the affine approximation $A_{\mathtt{H}}$ of $\mathtt{H}$ has the same Jacobian in the measured point $\mathbf{X}$ as $\mathtt{H}$. From Proposition 2 the Jacobians $\mathbf{J}_{\mathtt{T}'\mathtt{H}\mathtt{T}^{-1},\mathtt{T}\mathbf{x},\mathtt{T}'\mathbf{x}'} \sim \mathbf{J}_{\mathtt{T}'\mathbf{A}_{\mathtt{H}}\mathtt{T}^{-1},\mathtt{T}\mathbf{x},\mathtt{T}'\mathbf{x}'}$ for any affine transformations $\mathtt{T}$ and $\mathtt{T}'$. The composition of affine transformations $\mathtt{T}'A_{\mathtt{H}}\mathtt{T}^{-1}$ is an affine transformation, hence

$$A_{\mathtt{T}'\mathtt{H}\mathtt{T}^{-1}} = \mathtt{T}'A_{\mathtt{H}}\mathtt{T}^{-1}. \tag{16}$$

Both rotation and translation fall into the family of affine transformations and so the Eq. (16) holds for any choice of Cartesian coordinates. Sampson's error is then a geometric error for $A_{\mathtt{T}'\mathtt{H}\mathtt{T}^{-1}}$, which we already know is invariant to the choice of Cartesian coordinate system. □

Note that a general affine transformation does preserve neither the geometric nor Sampson's error, since it does not preserve perpendicularity and distances. However, Sampson's error changes in the same manner as the geometric error does, because it is the geometric error of the affine approximation $A_{\mathtt{H}}$ of $\mathtt{H}$ that is covariant to affine transformations of images.

6. Experiments

From Section 5, we already know, that Sampson's and the geometric error are equivalent for pure affine transformations. The aim of this experiment is to show their behavior with respect to the influence of the non-affine part of a general homography $\mathtt{H}$.

Consider now the decomposition $\mathtt{H} = \mathtt{PA}$, where $\mathtt{A}$ is an affine transformation and $\mathtt{P}$ has the form

$$\mathtt{P} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ p_7 & p_8 & 1 \end{pmatrix}.$$

Let the decomposition $\mathtt{H} = \mathtt{PA}$ exist. Let also $\mathtt{P}' = \mathtt{PT}$ be of the same form as $\mathtt{P}$ and let $\mathtt{A}' = \mathtt{T}^{-1}\mathtt{A}$ be an affine transformation. Then it can be easily shown that if $\mathtt{H} = \mathtt{P}'\mathtt{A}'$ then $\mathtt{T}$ must be the identity to keep both $\mathtt{A}'$ affine and $\mathtt{P}'$ in the desired form. Hence, if such a decomposition exists then it is unique. Let $\mathtt{G} = \mathtt{H}^{-1}$. Then from $\mathtt{H}^{-1}\mathtt{P} = \mathtt{GP} = \mathtt{A}^{-1}$ we get the equations for p_7 and p_8 as

$$p_7 = -\frac{g_7}{g_9} \quad \text{and} \quad p_8 = -\frac{g_8}{g_9}.$$

Thus, the decomposition exists iff $g_9 \neq 0$. The geometric meaning of this condition is that the origin of the coordinate system of the second image does not lie on the image of the line at infinity of the first image, i.e., $(\mathtt{H}^{-\mathtt{T}} l_\infty)^{\mathtt{T}}(0,0,1)^{\mathtt{T}} \neq 0$.

To acquire real data, we shot two images of a checkerboard, see Fig. 3. We manually selected four corresponding points in each image. The four points form rectangles R and R' that are depicted in solid line in Figs. 3A and B, respectively. From these four point-to-point correspondences the homography $\mathtt{H}$ was calculated. The origin of the coordinate system was chosen to coincide with one of the corners and is depicted by the 'X' marker. The homography $\mathtt{H}$ was decomposed into $\mathtt{H} = \mathtt{PA}$. The dashed rectangle in Fig. 3B is the rectangle R mapped from the first image to the second by the affine part $\mathtt{A}$ of $\mathtt{H}$. The dashed line in Fig. 3A arose as a mapping of the dashed rectangle in the second image back to the first by $\mathtt{H}^{-1}$, i.e., the image of R by $\mathtt{H}^{-1}\mathtt{A}$ within the first image.

In this experiment, we will use the following notation: $\mathbf{x}$ and $\mathbf{x}'$ stand for noise-free points, i.e., $\mathbf{x}' \sim \mathtt{H}\mathbf{x}$; $\mathbf{x}_0$ and $\mathbf{x}'_0$ denote the noisy points. The points, where the geometric error is minimized are $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}'$, and points where Sampson's error is minimized are $\mathbf{x}_{\mathrm{S}}$ and $\mathbf{x}'_{\mathrm{S}}$. Note that $\hat{\mathbf{x}}' \sim \mathtt{H}\hat{\mathbf{x}}$, but $\mathbf{x}'_{\mathrm{S}} \nsim \mathtt{H}\mathbf{x}_{\mathrm{S}}$ in general. In our experiment, we measured different errors: $d_\perp$ and d_{S} are the geometric and Sampson's errors

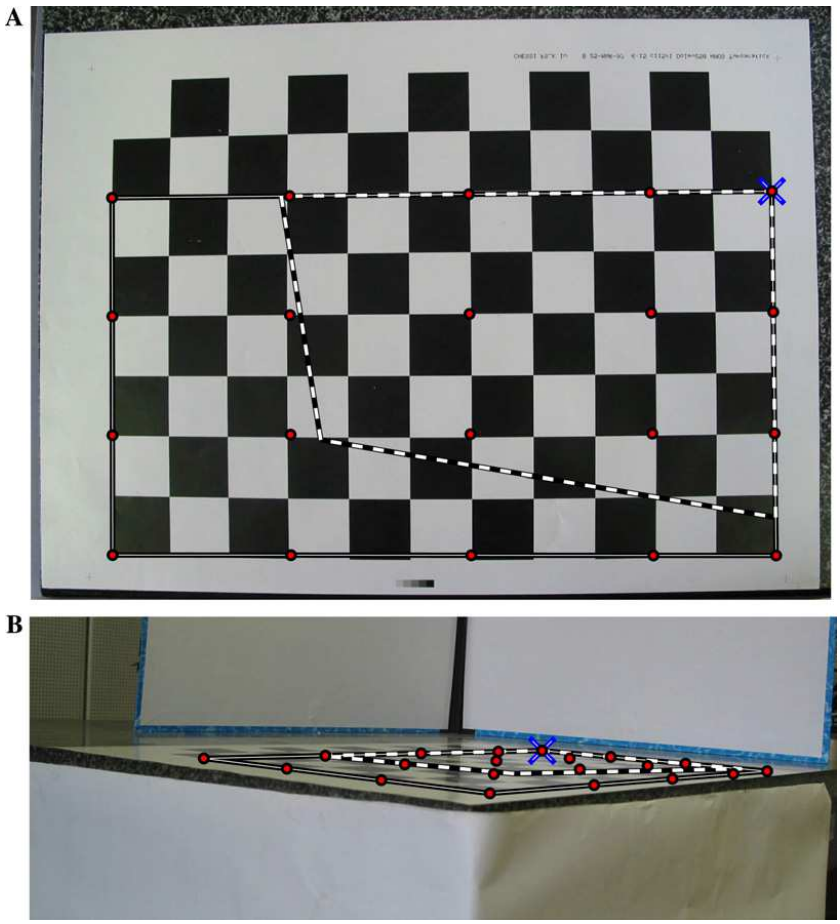


Fig. 3. Experimental setup. Two images of a checkerboard taken by a digital camera. The homography H was estimated from four point-to-point correspondences, shown as corners of the solid-line rectangles. The dashed rectangles show the effect of the affine part of the decomposed H. The origin of the decomposition is depicted by the 'X' marker.

respectively, $d_{\perp}^{*} = \sqrt{d^2(\mathbf{x}, \tilde{\mathbf{x}}) + d^2(\mathbf{x}', \tilde{\mathbf{x}}')}$, similarly $d_{\mathrm{S}}^{*} = \sqrt{d^2(\mathbf{x}, \mathbf{x}_{\mathrm{S}}) + d^2(\mathbf{x}', \mathbf{x}'_{\mathrm{S}})}$. The displacement of points $\mathbf{x}_{\mathrm{S}}$ and $\mathbf{x}'_{\mathrm{S}}$ is measured either as the distance in the second image $d_2(\mathbf{x}_{\mathrm{S}}, \mathbf{x}'_{\mathrm{S}}) = d(\mathtt{H}\mathbf{x}_{\mathrm{S}}, \mathbf{x}'_{\mathrm{S}})$ or by using the geometric error $d_{\perp}(\mathbf{x}_{\mathrm{S}}, \mathbf{x}'_{\mathrm{S}})$. The errors were measured at points depicted in Figs. 3A and B. A Gaussian noise with $\sigma = 0.3$ was added to each coordinate of a noise-free point correspondence. All values were obtained as averages over all points over 1010 realizations of noise and are shown in Fig. 4.

The graphs in Fig. 4 show that the geometric (A,B) and Sampson's (C,D) error provide very similar results independently of the value of the non-affine part of the planar homography. The same graphs show that the realization of the noise has a much stronger influence than the values of p_7 and p_8 on both types of the error. The value of σ of the Gaussian noise was set to $\sigma = 0.3$ in this experiment. We observed the same behavior for $\sigma \in \langle 10^{-4}, 10 \rangle$. Graphs E and F show that the displace-

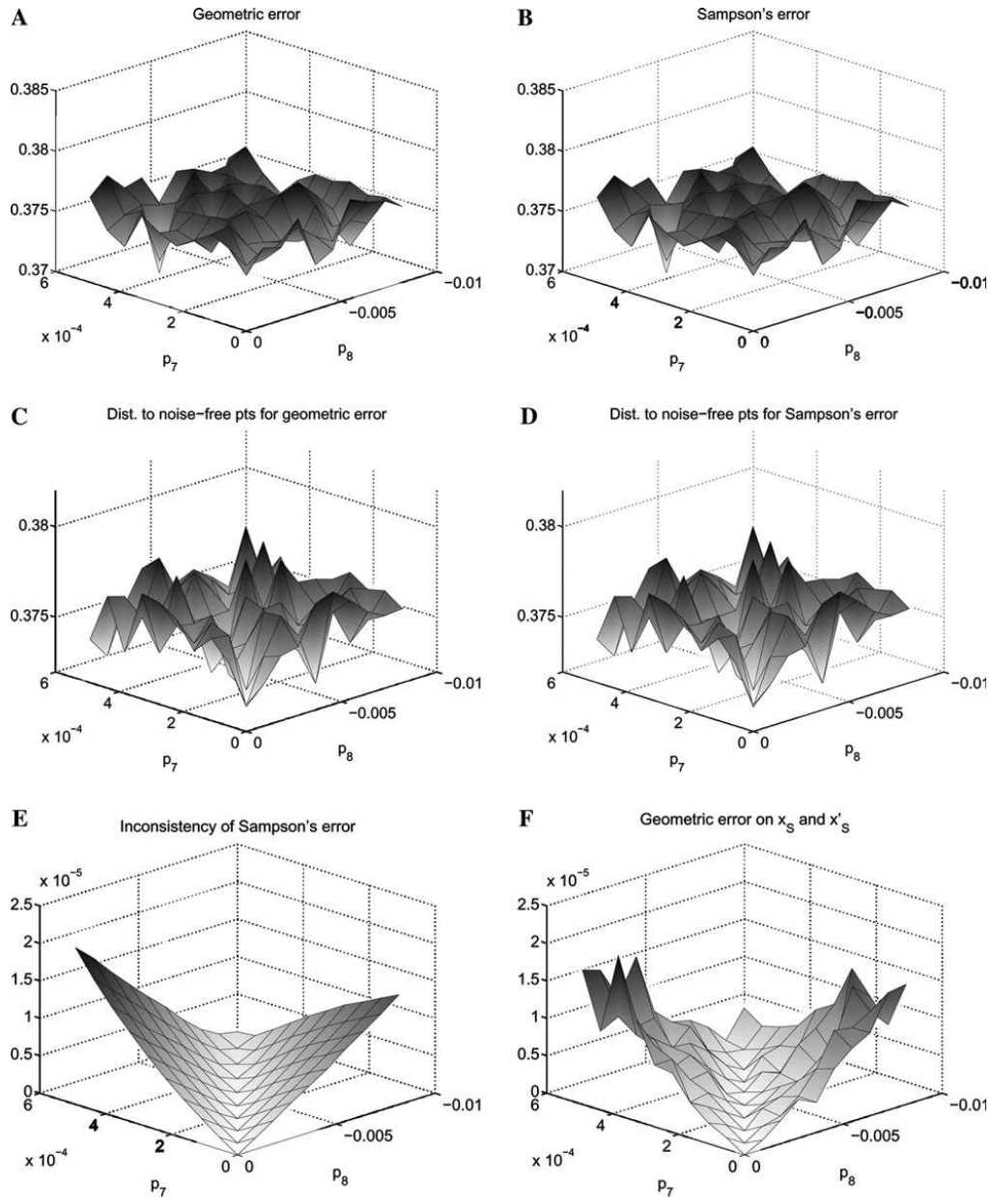


Fig. 4. Dependency of: (A) $d_{\perp}$, (B) d_{S}, (C) $d_{\perp}^{*}$, (D) d_{S}^{*}, (E) $d(\mathtt{H}\mathbf{x}_{\mathrm{S}}, \mathbf{x}'_{\mathrm{S}})$, and (F) $d_{\perp}(\mathbf{x}_{\mathrm{S}}, \mathbf{x}'_{\mathrm{S}})$ on the non-affine part of the homography $\mathtt{H}$.

ment of the points where the Sampson's error is minimized, i.e., $\mathbf{x}_{\mathrm{S}}$ and $\mathbf{x}'_{\mathrm{S}}$, depends on the value of p_7 and p_8. The more the homography "squeezes" the image, the more displaced the points are. On the other hand, the displacement is in four orders of magnitude smaller than the error itself.

The main conclusion of the experiment conducted in this section is that Sampson's error gives sufficiently good results in 2D that are comparable with the geometric error. The displacement of points $\mathbf{x}_S$ and $\mathbf{x}'_S$ is small, but still significantly higher than machine precision and can cause problems while reconstructing 3D points (see Section 7).

7. Triangulation

The method presented in this paper would be useful in applications where a high accuracy is desired. In this section, we will mention one problem where we can use the geometric error for homographies to improve the accuracy of the reconstruction of planes in the scene. We call it *planar triangulation*. It is an extension of the triangulation problem [6].

The two rays in space, the first one from camera center $\mathbf{C}$ through image point $\mathbf{x}$ in the first image and the other one from $\mathbf{C}'$ through $\mathbf{x}'$, will intersect only if $\mathbf{x}'^{\mathrm{T}}\mathbb{F}\mathbf{x} = 0$. If noise is attached to the image coordinates, then the rays may not meet.

In the triangulation problem [6], it is assumed that the fundamental matrix $\mathbb{F}$ is known exactly. For this fundamental matrix, the points $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{x}}'$ are found, so that $\tilde{\mathbf{x}}'^{\mathrm{T}}\mathbb{F}\tilde{\mathbf{x}} = 0$ and the sum of the square distances $d(\mathbf{x}, \tilde{\mathbf{x}})^2 + d(\mathbf{x}', \tilde{\mathbf{x}}')^2$ is minimal.

Assume there is a (dominant) plane in the scene and $\mathtt{H}$ is the homography induced by this plane. When the triangulation method [6] is used, the additional constraint of the planarity is omitted and the reconstructed points will in general not lie in a single plane. The homography $\mathtt{H}$ is compatible [10, Section 12] with the fundamental matrix if, and only if for all $\hat{\mathbf{x}}$

$$(\mathtt{H}\hat{\mathbf{x}})^{\mathrm{T}}\mathbb{F}\hat{\mathbf{x}} = 0.$$

This means all the correspondences satisfying $\hat{\mathbf{x}}' \sim \mathtt{H}\hat{\mathbf{x}}$ will automatically satisfy the epipolar geometry $\hat{\mathbf{x}}'^{\mathrm{T}}\mathbb{F}\hat{\mathbf{x}} = 0$ and hence the two rays in space, passing through $\mathbf{x}$ and $\mathbf{x}'$, respectively, will intersect. Moreover all these intersections in space given by correspondences satisfying homography $\mathtt{H}$ lie on the plane inducing $\mathtt{H}$.

7.1. Experiment

We have made synthetic experiments with the planar triangulation using images of an artificial scene (Figs. 5A and B). From noise-free images we obtained the fundamental matrix $\mathbb{F}$ and the homography $\mathtt{H}$. For testing purposes we used only the points on the front face of the building. Then, we added Gaussian noise with standard deviation σ to the image coordinates. From these noisy points we calculated corrected points using the standard and the planar triangulation. Fig. 5 gives the comparison of the distance of corrected points to the original noise-free points, denoted as 2D error (in pixels), and its standard deviation—graphs C and D. We then computed a 3D reconstruction using the corrected points (both from the standard and the planar triangulation). The distance of reconstructed 3D points to the

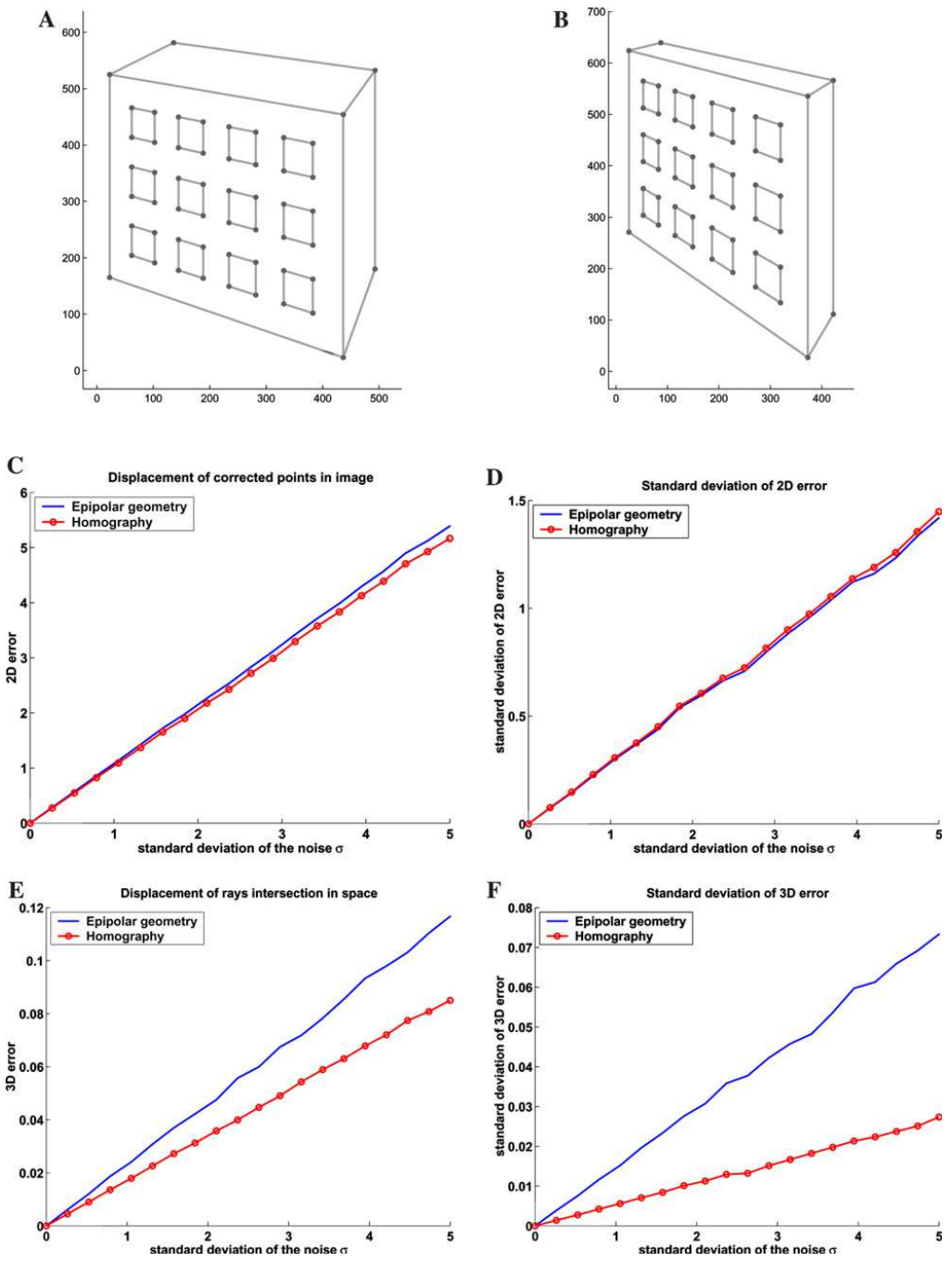


Fig. 5. Synthetic experiment with images (A) and (B). The graphs compare errors in triangulation using the fundamental matrix F (standard) and the homography H (planar) in images (C) and (D) and in 3D space (E) and (F). For testing, only points lying in the plane of the frontal side of the building were used. The dimensions of the building are $9 \times 7 \times 1$ units.

original 3D points is denoted as 3D-error (in units, the building dimensions are $9 \times 7 \times 1$)—graphs E and F.

The result of this experiment shows that the decrease in the 2D error is not significant. On the other hand, the 3D error is considerably decreased by the planar triangulation.

When we tried to use Sampson's approximation followed by the standard triangulation (it consists of computing pseudo-inversion and solving a polynomial of degree six), we got similar results to those when using the planar triangulation.

The experiment shows that the accuracy of the reconstruction of a planar scene could be improved by using the planar triangulation instead of the standard one. Using Sampson's approximation together with the standard triangulation gives very similar results as the planar triangulation but it is computationally more expensive and the planarity of the reconstructed scene is not guaranteed.

8. Conclusions

In this paper, a new method for computing the geometric error for homography was introduced. The main contribution of the paper is the derivation of the formula for computing the error. This formula has not been known before. It is interesting to see that the error is obtained as a solution of a degree eight polynomial. We have also proved that there indeed exist a corrected correspondence that minimizes the geometric distance to the measured correspondence, and that the proposed method finds it correctly.

We tested two different methods of measuring correspondence error with respect to given homography $\mathtt{H}$, the geometric error by the Sampson's error. Our experiments had shown that the Sampson's error is sufficiently precise for a wide range of applications including RANSAC. We also discovered (and proved) nice properties of the Sampson's error with respect to affine transformations of images. The applications where the use of the geometric error could bring higher accuracy were shown. This statement is encouraged with experiments with the planar triangulation.

Appendix A. Proof of correctness

Proposition 4. *Let* $\mathtt{H}$ *be a regular matrix of the following form*:

$$\mathtt{H} = \begin{pmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & 0 & h_9 \end{pmatrix}.$$

Then the function

$$e = x^2 + y^2 + \left(\frac{x'}{w'}\right)^2 + \left(\frac{y'}{w'}\right)^2, \tag{A.1}$$

where

$$x' = h_1 x + h_2 y + h_3, \tag{A.2}$$

$$y' = h_4 x + h_5 y + h_6, \tag{A.3}$$

$$w' = h_7 x + h_9 \tag{A.4}$$

has a global minimum. In this minimum the partial derivatives of e are defined and equal to zero.

Proof. First of all we introduce the notation used throughout the proof. Let us write e as a sum of three functions $e_1 = x^2 + y^2$, $e_2 = (x'/w')^2$, and $e_3 = (y'/w')^2$, i.e., $e = e_1 + e_2 + e_3$. Since all e_i, $i \in \{1, 2, 3\}$, are nonnegative, we have $e \geqslant e_i$. We can also define three lines, $\ell_{\mathbf{x}'}$, $\ell_{\mathbf{y}'}$, and $\ell_{\mathbf{w}'}$ in $\mathbb{R}^2$ letting x', y', and w' equal zero in (A.2), (A.3), and (A.4), respectively. Let **A** be the point of intersection of $\ell_{\mathbf{w}'}$ with $\ell_{\mathbf{x}'}$ and **B** be the point where $\ell_{\mathbf{w}'}$ intersects $\ell_{\mathbf{y}'}$. Since $\mathtt{H}$ is regular, there does not exist any $\mathbf{x} = (x, y, 1)^{\mathrm{T}}$, so that $\mathtt{H}\mathbf{x} = \mathbf{0}$. Thus **A** and **B** are two different points. The situation is depicted in the Fig. 6.

The function e is continuous and even differentiable throughout the region where the denominator $h_7 x + h_9$ is nonzero and finite, i.e., in $\mathbb{R}^2 \setminus \ell_{\mathbf{w}'}$. The term e_1 tends to plus infinity in all points of $\ell_{\mathbf{w}'}$ except for **A** where it is guaranteed to be nonnegative. Analogously, the term e_2 tends to plus infinity in all points of $\ell_{\mathbf{w}'}$ except for **B** where it is guaranteed to be nonnegative. The sum of e_1 and e_2, and thus e, tends to plus infinity in all points of $\ell_{\mathbf{w}'}$.

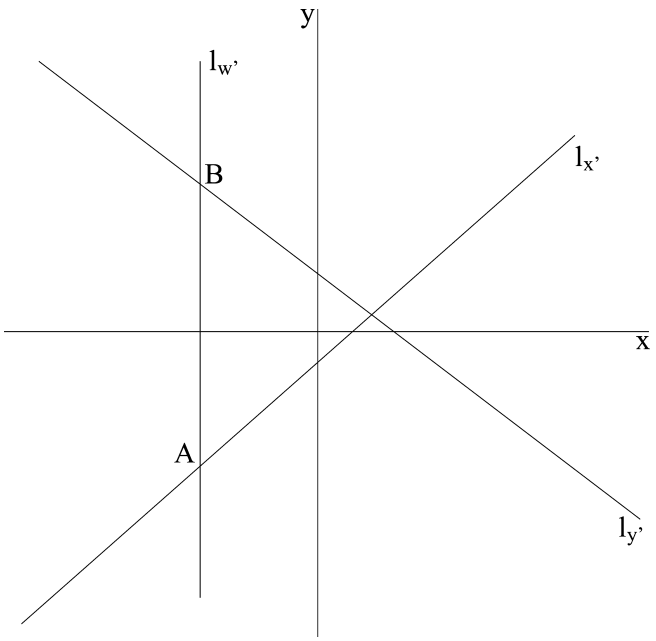


Fig. 6. Lines $\ell_{\mathbf{x}'}$, $\ell_{\mathbf{y}'}$, and $\ell_{\mathbf{w}'}$ are sets of points, where $x' = 0$, $y' = 0$, and $w' = 0$, respectively.

We choose a point in $\mathbb{R}^2 \setminus \ell_{\mathbf{w}'}$ and take the value of e in it for a constant K. The set

$$I = \{(x,y) \in \mathbb{R}^2 \setminus \ell_{\mathbf{w}'} \mid e(x,y) \leqslant K\}$$

is nonempty and closed. It is also bounded because it is a subset of the circle

$$\{(x,y) \in \mathbb{R}^2 \mid x^2 + y^2 \leqslant K\}.$$

Therefore I is a compact set and so it contains all global minima of e. At least one global minimum of e exists because the values of e on I are images of a compact set under a continuous mapping, thus they form a compact subset of $\mathbb{R}$. □

Appendix B. Coefficients of the polynomial

In Section 3, we derived the formula for computing the geometric error. Here we focus on the implementation.

First of all we can see that the image rotation matrix $\mathtt{R}$ depends only on h_7 and h_8 (10). From (8) we know that $\mathtt{R}$ stays unchanged by the translations $\mathtt{L}$ and $\mathtt{L}'$. So the matrix $\mathtt{R}$ could be computed directly from $\mathtt{H}$. Matrix $\mathtt{R}$ is the same for all the correspondences.

Coefficients of the resulting polynomial of degree eight are sums of products of entries of the matrix $\mathtt{Q}$, which are quite complicated. We can apply image rotation matrix $\mathtt{R}'$ to the second image. We have

$$\mathtt{R}'\tilde{\mathbf{x}}' = \mathtt{R}'\mathtt{Q}\tilde{\mathbf{x}},$$

and $\mathtt{Q}' = \mathtt{R}'\mathtt{Q}$. To decrease the number of summands, we can design this rotation in the same way as the matrix $\mathtt{R}$ to make $q'_4 = 0$. Note, that q'_8 stays unchanged by the rotation $\mathtt{R}'$, so $q'_8 = q_8 = 0$. Matrix $\mathtt{R}'$ differs for each correspondence.

After applying the rotations on both images, we have homography $\bar{Q}$ in the form

$$\bar{Q} = \begin{pmatrix} \bar{q}_1 & \bar{q}_2 & \bar{q}_3 \\ 0 & \bar{q}_5 & \bar{q}_6 \\ \bar{q}_7 & 0 & \bar{q}_9 \end{pmatrix}.$$

The resulting polynomial is in the following form:

$$\sum_{i=0}^{8} \bar{p}_i \tilde{x}^i.$$

Here is the list of the coefficients p_i expressed in entries $\bar{q}$ of the matrix $\bar{Q}$. We use the following substitutions:

$$t = \bar{q}_3\bar{q}_5 - \bar{q}_2\bar{q}_6,$$
$$r = \bar{q}_2^2 + \bar{q}_5^2 + \bar{q}_9^2.$$

The polynomial coefficients are:

$$\bar{p}_0 = \bar{q}_9^3((-\bar{q}_3^2 - \bar{q}_6^2)\bar{q}_7\bar{q}_9 + \bar{q}_1\bar{q}_3 r) + \bar{q}_1\bar{q}_5\bar{q}_9 rt - \bar{q}_7(\bar{q}_9^2 + r)t^2,$$

$$\begin{aligned}\bar{p}_1 = & -4\bar{q}_3^2\bar{q}_7^2\bar{q}_9^3 - 4\bar{q}_6^2\bar{q}_7^2\bar{q}_9^3 + 3\bar{q}_1\bar{q}_3\bar{q}_7\bar{q}_9^2 r + \bar{q}_9 r(\bar{q}_1^2(\bar{q}_5^2 + \bar{q}_9^2) + \bar{q}_9^2 r) \\ & - \bar{q}_1\bar{q}_5\bar{q}_7 rt - 4\bar{q}_7^2\bar{q}_9 t^2,\end{aligned}$$

$$\begin{aligned}\bar{p}_2 = & \bar{q}_7(\bar{q}_9(-6(\bar{q}_3^2 + \bar{q}_6^2)\bar{q}_7^2\bar{q}_9 - \bar{q}_1\bar{q}_3\bar{q}_7(\bar{q}_9^2 - 3r) + 4\bar{q}_9^3 r + 3\bar{q}_9 r^2 \\ & + \bar{q}_1^2\bar{q}_9(\bar{q}_5^2 + \bar{q}_9^2 + 3r)) - 5\bar{q}_1\bar{q}_5\bar{q}_7\bar{q}_9 t - 2\bar{q}_7^2 t^2),\end{aligned}$$

$$\begin{aligned}\bar{p}_3 = & \bar{q}_7^2(\bar{q}_9(-4(\bar{q}_3^2 + \bar{q}_6^2)\bar{q}_7^2 + 4\bar{q}_9^4 + 14\bar{q}_9^2 r + 3r^2) + \bar{q}_1^2(-(\bar{q}_5^2\bar{q}_9) + 3\bar{q}_9(\bar{q}_9^2 + r)) \\ & + \bar{q}_1\bar{q}_7(\bar{q}_3(-3\bar{q}_9^2 + r) - 3\bar{q}_5 t)),\end{aligned}$$

$$\bar{p}_4 = \bar{q}_7^3((-\bar{q}_3^2 - \bar{q}_6^2)\bar{q}_7^2 - 3\bar{q}_1\bar{q}_3\bar{q}_7\bar{q}_9 + 16\bar{q}_9^4 + 18\bar{q}_9^2 r + r^2 + \bar{q}_1^2(-\bar{q}_5^2 + 3\bar{q}_9^2 + r)),$$

$$\bar{p}_5 = \bar{q}_7^4(-(\bar{q}_1\bar{q}_3\bar{q}_7) + \bar{q}_1^2\bar{q}_9 + 25\bar{q}_9^3 + 10\bar{q}_9 r),$$

$$\bar{p}_6 = \bar{q}_7^5(19\bar{q}_9^2 + 2r),$$

$$\bar{p}_7 = 7\bar{q}_7^6\bar{q}_9,$$

$$\bar{p}_8 = \bar{q}_7^7.$$

References

[1] Y. Kanazawa, K. Kanatani, Stabilizing image mosaicing by model selection, in: Proc. Second Workshop on 3D Structure from Multiple Images of Large-scale Environments and Applications to Virtual and Augmented Reality, 2000, pp. 10–17.

[2] P. Pritchett, A. Zisserman, Wide baseline stereo matching, in: Proc. Internat. Conf. Computer Vision, 1998, pp. 754–760.

[3] J. Matas, O. Chum, M. Urban, T. Pajdla, Robust wide baseline stereo from maximally stable extremal regions, in: Proc. British Machine Vision Conf., vol. 1, BMVA, London, UK, 2002, pp. 384–393.

[4] M.A. Fischler, R.C. Bolles, Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, Commun. ACM 24 (6) (1981) 381–395.

[5] K. Kanatani, Statistical Optimization for Geometric Computation: Theory and Practice, vol. 18 of Machine Intelligence and Pattern Recognition, Elsevier, Amsterdam, 1996.

[6] R.I. Hartley, P. Sturm, Triangulation, Comput. Vision Image Und. 2 (68) (1997) 146–157.

[7] P. Sturm, Vision 3D non calibrée—contributions à la reconstruction projective et étude des mouvements critiques pour l'auto-calibrage, Ph.D. Thesis, INPG, Grenoble, France, 1997.

[8] O. Chum, The reconstruction of 3D scene from the correspondences in images, Master's Thesis, MFF UK, Prague, Czech Republic, January 2001.

[9] O. Chum, T. Pajdla, Evaluating error of homography, in: Proc. Computer Vision Winter Workshop, Wien, Austria, 2002, pp. 315–324.

[10] R.I. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, Cambridge University Press, Cambridge, UK, 2000.

[11] P.D. Sampson, Fitting conic sections to very scattered data: An iterative refinement of the Bookstein algorithm, Comput Graph. Image Process. 18 (1982) 97–108.

[12] Waterloo Maple Inc., Maple V, http://www.maplesoft.com.

3D SSD Tracking with Estimated 3D Planes

Dana Cobzas* Peter Sturm
INRIA Rhone-Alpes
655 Av. de l'Europe, 38330 Montbonnot, France
{cobzas,sturm}@inrialpes.fr

Abstract

We present a tracking method where full camera position and orientation is tracked from intensity differences in a video sequence. The camera pose is calculated based on plane equations, and hence does not depend on point correspondences. The plane based formulation also allows additional constraints to be naturally added, e.g. perpendicularity between walls, floor and ceiling surfaces, co-planarity of wall surfaces etc. A particular feature of our method is that the full 3D pose change is directly computed from temporal image differences without making a commitment to a particular intermediate (e.g. 2D feature) representation. We experimentally compared our method with regular 2D SSD tracking and found it more robust and stable. This is due to 3D consistency being enforced even in the low level registration of image regions. This yields better results than first computing (and hence committing to) 2D image features and then from these compute 3D pose.
Keywords: *visual tracking, structure estimation*

1. Introduction

In visual tracking the pose of an object or the camera motion is estimated over time based on image motion information. Some applications such as video surveillance only require that the target object is tracked in image space. For other applications such as augmented reality and robotics full 3D camera motion is needed. In this paper we concentrate on tracking full 3D pose.

One way to classify tracking methods is into feature-based and registration based. In feature-based approaches features in a (usually apriori) 3D model are matched with features in the current image. Commonly a feature detector is used to detect either special markers or natural image features. Pose estimation techniques can then be used to compute the camera position from the 2D-3D correspondences. Many approaches use image contours (edges or curves) that are matched with an apriori CAD model of the object [11, 14, 6]. Most systems compute pose parameters by linearizing with respect to object motion. A characteristic of these algorithms is that the feature detection is relatively decoupled from the pose computation, but sometimes past pose is used to limit search ranges, and the global model can be used to exclude feature mismatches [11, 2].

In registration based tracking the pose computation is based on directly aligning a reference intensity patch with the current image to match each pixel intensity as closely as possible. These methods assume that the change in location and appearance of the target in consecutive frames is small. Image constancy can be exploited to derive efficient gradient based schemes using normalized correlation, or a sum-of-squared differences (e.g. L_2 norm) criterion, giving the technique its popular name SSD tracking. Unlike the two previous approaches which build the definition of what is to be tracked into the low level routine (e.g. a line feature tracker tracks just lines), in registration based tracking any distinct pattern of intensity variation can be tracked. The first such methods required spatial image derivatives to be recomputed for each frame when "forward" warping the reference patch to fit the current image [12], while more recently, efficient "inverse" algorithms have been developed, which allow the real time tracking for the 6D affine [7] and 8D projective warp [3]. A more complicated appearance model can be used to compensate changes in intensity [7] or can be learned as a mixture of stable image structure and motion information [10].

In this paper we extend the registration-based techniques by constraining the tracked regions to 3D planes. This will allow tracking full 3D camera position like in the model-based approaches but eliminates the need for explicit feature matching. The update is based on the same SSD criterion as the classical registration-based methods with the difference that the update is done directly on the 3D parameters and not on the 2D warp parameters. The approach is thus different from previous approaches that first estimate the homography warp from salient points and then the 3D motion parameters from the homography [15]. The 3D plane parameters are estimated and optimized in a training phase (typically ≈ 100 frames) using structure-from-motion techniques. The algorithm does not require complete scene de-

* Acknowledgments to NSERC Canada for supporting this work.

composition in planar facets, but works with few planar patches identified in the scene. Man-made environments usually contain planar structures (e.g. walls, doors). Some advantages of using a global 3D model and local surface patches are that only surfaces with salient intensity variations need to be processed, while the 3D model connects these together in a physically correct way. We show experimentally that this approach yields more stable and robust tracking than previous approaches, where each surface patch motion is computed individually.

Related work of incorporating a 3D model into registration based tracking involve a full 3D model (3D patches defined by estimated 3D points) of the regions that are tracked [5]. Another similar approach is presented by Baker et al. [16] where the 3D model is calculated from a 2D active appearance model (AMM) and used to improve the tracking. In the proposed technique we loosen this constraint and require only the plane parameters to be estimated. Any regions on these planes can then be tracked.

The rest of the paper is organized as follows: The next section describes the tracking algorithm, then Section 3 presents the method for estimating plane equations from images. The complete tracking system is presented in Section 4 and its qualitative and quantitative evaluation in Section 5 followed by conclusions and a discussion in Section 6.

2. Tracking 3D planes

We consider the problem of tracking the motion of a camera looking at a rigid structure using image registration. The structure is represented by a set of 3D planes that are estimated a-priori as described later in Section 3. Full 3D camera motion is tracked by registering image regions on corresponding planes through the induced homography.

2.1. Homography induced by a plane

It is well known that images of points on a plane in two views are related by a homography [8]. For planes in general position this homography is uniquely determined by the plane equation. A 3D plane is represented as $\pi = [\mathbf{n}^T, d]$, where $\mathbf{n}$ is the unit normal and d is the signed distance from the origin to the plane. For points $\mathbf{X}$ on the plane $\mathbf{n}^T\mathbf{X} + d = 0$. If the world coordinate system is aligned with the first camera coordinate system, the calibrated projection matrices have the form:

$$P_0 = K[I|\mathbf{0}] \quad P_t = K[R|\mathbf{t}] \tag{1}$$

where K is the camera matrix (internal parameters) and $R, \mathbf{t}$ represents the 3D motion of the second camera with respect to the first one. Now, the homography induced by the plane π has the form:

$$H = K(R - \mathbf{t}\mathbf{n}^T/d)K^{-1} \tag{2}$$

Image points in the two views I_1, I_2 are then related by $\mathbf{u}_2 = H\mathbf{u}_1$. If the image points are normalized with respect to camera internal parameters $\mathbf{x} = K^{-1}\mathbf{u} = [R|\mathbf{t}]X$ the homography becomes:

$$H = R - \mathbf{t}\mathbf{n}^T/d \tag{3}$$

In the tracking problem formulation the goal is to directly estimate camera motion $R, \mathbf{t}$ that corresponds to the homography that best aligns the image points in two views assuming that the plane parameters are known.

2.2. Region-based tracking for planes

Assume we have estimated parameters in the plane equations for several planar regions in the scene. Let $\mathbf{x}_k = \{\mathbf{x}_1, \mathbf{x}_2, \dots \mathbf{x}_{K_k}\}$ denote all the (interior) normalized image pixels that define the projection of the planar region $\pi_k = [\mathbf{n}_k^T, d_k]$ in image I. We refer to $I_0 = T$ as the *reference image* and to the union of the projections of the planar regions in T, $\cup_k T(\mathbf{x}_k)$ as the *reference template*. The goal of the tracking algorithm is to find the (camera) motion $P_t = [R_t, \mathbf{t}_t]$ that best aligns the reference template with the current image I_t. The problem is formulated as finding an incremental motion update $\Delta\mathbf{p}$ from frame I_{t-1} to I_t that is added to the current motion. The model is defined so it is aligned with the first frame (template). A more precise formulation follows next (refer to Figure 1).

As described in the previous section the image motion in image t for each individual planar region k can be perfectly modeled by a homography warp $H(\mathbf{x}_k; P_t, \pi_k) = R_t - \mathbf{t}_t\mathbf{n}_k^T/d_k$. In the following we denote the homography warp by $H(\mathbf{x}_k; \mathbf{p}_t)$ where $\mathbf{p} = [\alpha_x, \alpha_y, \alpha_z, t_x, t_y, t_z]^T$ are column vectors of the 3D motion parameters that define the camera motion (Euler angles and translation). The main difference from the previous approaches in registration based tracking [3] is that we directly compute 3D motion parameters unified over the whole scene as opposed to 2D warp parameters for each individual patch.

Under the common image constancy assumption (e.g. no illumination variation, no occlusion) used in motion detection and tracking [9] the tracking problem can be formulated as finding $\mathbf{p}_t$ such as:

$$\cup_k T(\mathbf{x}_k) = \cup_k I_t(H(\mathbf{x}_k; \mathbf{p}_t)) \tag{4}$$

$\mathbf{p}_t = \mathbf{p}_{t-1} \circ \Delta\mathbf{p}$ (where '$\circ$' denotes the composition operation) can be obtained by minimizing the following objective function with respect to $\Delta\mathbf{p}$:

$$\sum_k \sum_{\mathbf{x}} [T(\mathbf{x}_k) - I_t(H(\mathbf{x}_k; \mathbf{p}_{t-1} \circ \Delta\mathbf{p}))]^2 \tag{5}$$

The update in position $\Delta\mathbf{p}$ is based on the image difference between the template image and the current image warped in the space of the template, the update in position being place on the side of the current image. As a consequence,

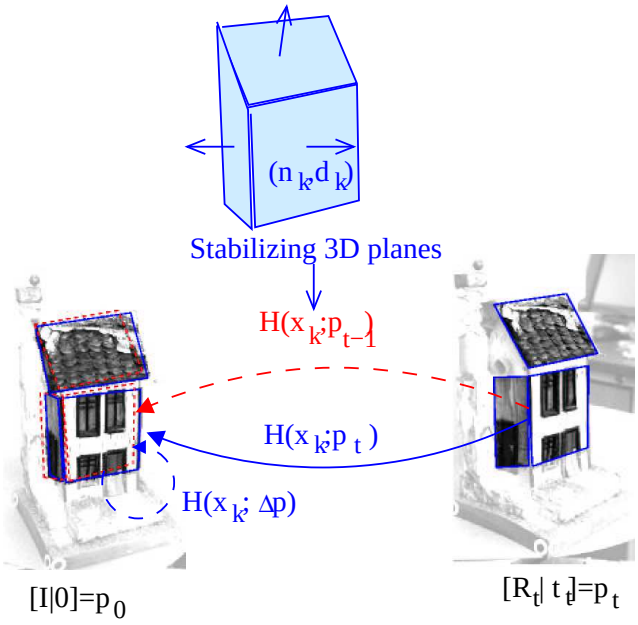


Figure 1. Overview of the 3D plane based tracking system. In standard SSD tracking 2D surface patches are related through a homography H between frames. In our system a 3D planes are estimated (from video alone), and global 3D pose change $\Delta\mathbf{p}$ is computed, and used to enforce a consistent update of all the surface warps.

the computations are performed in the space of the current image.

For efficiency, we solve the problem by an inverse compositional algorithm [3] that minimizes the error between the template image and the current image warped in the space of the template image, with the update on the template image (see Equation 7). As shown below, working in the space of the template image allow more computations to be done only once at the initialization speeding up the tracking. H becomes the homography from the image t to the template image (inverse of Equation 3). The goal is to find $\Delta\mathbf{p}$ that minimizes

$$\sum_k \sum_{\mathbf{x}} [T(H(\mathbf{x}_k;\Delta\mathbf{p})) - I_t(H(\mathbf{x}_k;\mathbf{p}_{t-1}))]^2 \qquad (6)$$

where in this case the 3D motion parameters are updated as:

$$P_t = P_{t-1} \circ \mathrm{inv}(\Delta P) \qquad (7)$$

where $\mathrm{inv}(P) = [R^T | -R^T\mathbf{t}]$ for $P = [R|\mathbf{t}]$. As a consequence the homography warp update is:

$$H(\mathbf{x}_k;\mathbf{p}_t) = H(\mathbf{x}_k;\Delta\mathbf{p})^{-1} \circ H(\mathbf{x}_k;\mathbf{p}_{t-1}) \qquad (8)$$

Performing a Taylor expansion of Equation 6 gives:

$$\sum_k \sum_x [T(H(\mathbf{x}_k;\mathbf{0})) + \nabla T \frac{\partial H}{\partial \mathbf{p}}(\mathbf{x}_k;\mathbf{0})\Delta\mathbf{p} - I_t(H(\mathbf{x}_k;\mathbf{p}_t))] \qquad (9)$$

As the motion of the template image is zero (the model is aligned with the template frame) $T = T(H(\mathbf{x}_k;\mathbf{0}))$. Denoting the image derivatives by M

$$M = \sum_k \sum_x \nabla T \frac{\partial H}{\partial \mathbf{p}} \qquad (10)$$

equation 9 can be rewritten as:

$$M\Delta\mathbf{p} \simeq \mathbf{e}_t \qquad (11)$$

where $\mathbf{e}_t$ represents the image difference between the template regions and warped image regions, and the motion $\Delta\mathbf{p}$ is computed as the least squares solution to Equation 11.

The image derivatives M are evaluated at the reference pose $\mathbf{p} = \mathbf{0}$ and they are constant across iterations and can be precomputed, resulting in an efficient tracking algorithm that can run in real time (see Section 4).

3. Estimating planes equations from images

The tracking algorithm described in Section 2.2 requires knowledge of the plane parameters for each planar region that is tracked. The plane equations are estimated from images in a bootstrapping phase. Salient feature points on each plane are tracked using standard (2D image-plane) SSD trackers as in [3, 7]. The grouping of the points depending on the plane can be easily solved by having the user mark planar regions in the first frame.

We first present the algorithm that computes a plane equation from images of points on the plane in two images. It is a special case of the structure from motion problem where the camera is internally calibrated and the feature points belong to a physical plane. The homography induced by the plane H is robustly computed using RANSAC from 4 or more corresponding points. Knowing that it is of the form $H = R - \mathbf{t}\mathbf{n}^T/d$, the motion and structure parameters $\{R, \frac{1}{d}\mathbf{t}, \mathbf{n}\}$ can be computed [13]. There are in general four solutions but only at most two are physically valid by imposing the positive depth constraint (model points are in front of the camera).

In a more general case, when multiple planes are viewed in several images, a reference view is chosen and the corresponding plane homographies that relate the reference view with additional views are computed. The motion for each frame is averaged over the motions estimated from each plane homography and the plane parameters are averaged over the ones computed from several views. Assuming a smooth motion between adjacent views only the solution that corresponds to the motion closest to the motion of the previous frame is chosen. For the first pair one of the

two physically valid solutions is chosen. The scale of the scene is also disambiguated by fixing the distance to one plane. At the end a nonlinear optimization using Levenberg-Marquardt algorithm over all the frames is performed. The error that we optimize is the symmetric transfer error for points related through a homography:

$$\{R_2, \mathbf{t}_2, \dots R_m, \mathbf{t}_m; \mathbf{n}_1, d_1, \dots, \mathbf{n}_k, d_k\} = \\ \operatorname{argmin} \sum_t \sum_k \sum_{\mathbf{x}_{tk}} d^2(\mathbf{x}_{tk}, H_{tk}\mathbf{x}_{1k}) + d^2(\mathbf{x}_{1k}, H_{tk}^{-1}\mathbf{x}_{tk}) \tag{12}$$

This is not exactly the maximum likelihood estimator under Gaussian noise but is more practical in our case as it will give the best motion and plane structure without explicitly computing the 3D points coordinates.

3.1. Incorporating constraints between planes

Known constraints between planes such as perpendicularity or parallelism of walls can potentially stabilize the tracking. We impose constraints by a minimum parametrization of the plane parameters as in [4].

Consider two planes $\pi_1 = [\mathbf{n}_1^T, d_1]$,$\pi_2 = [\mathbf{n}_2^T, d_2]$. A perpendicularity constraint can be algebraically expressed by a vanishing dot product between the plane normals:

$$n_{11}n_{21} + n_{12}n_{22} + n_{13}n_{23} = 0 \tag{13}$$

This bilinear constraint can be enforced by eliminating one plane parameter. We chose to eliminate the parameter n_{ik} such that the absolute value of the corresponding parameter on the second plane n_{jk} is maximal over all the parameters.

For the other type of constraint when the planes are parallel we impose that the normals of the two planes are the same. This eliminates all parameters that represent the unit normal of one plane.

$$n_{1k} = n_{2k}, k = 1, 2, 3 \tag{14}$$

The resulting plane parameters and the originally recovered motions are then optimized using the same Equation 12. A full parametrization of the planes is recovered for every plane from Equations 13,14. A potentially somewhat more accurate approach would involve obtaining a minimal parameterization of 3D points on constrained planes and estimating the structure of those points and the camera motion from feature correspondences. This would allow defining a maximum likelihood estimator under Gaussian image noise. The plane parameters are then computed from the estimated 3D points.

4. Tracking system overview

We incorporated the proposed plane tracking algorithm into a system that first initializes plane equations from 2D image tracking over a limited motion and then switches to track points on the estimated 3D planes.

Bootstrapping phase

1. The user marks planar regions in the first frame and specifies plane constraints (parallelism, perpendicularity) as applicable. Feature points inside these regions are tracked using standard SSD 2D trackers.
2. Plane parameters are first initialized by averaging close form solutions from homographies and then a minimal parametrization is optimized together with the estimated motion over all the training frames as described in Section 3.
3. The 3D planes are related to the current frame using the 2D tracked points. This will align the origin of the world coordinate system with the current frame. Then the plane based tracking is initialized by computing the derivative images M (Equation 10).

Tracking phase

The tracking now continues with 2D surface patches integrated in the 3D model of the planes that enforces a globally consistent motion for all surface patches as described in Section 2.2.

1. An incremental position update $\Delta\mathbf{p}$ is computed based on image differences between the regions in the reference template and the warped regions from the current image (Equation 11).
2. The global camera position is updated based on Equation 7.

5. Experiments

Two important properties of tracking methods are convergence and accuracy. Tracking algorithms based on optimization and spatio-temporal derivatives (Equation 9) can fail to converge because the image difference between consecutive frames I_{t-1}, I_t is too large (more than just few pixels), and the first order Taylor expansion (Equation 9) around $\mathbf{p}_{t-1}$is no longer valid, or some disturbance causes the image constancy assumption to be invalid.

In the numerical optimization the pose update $\Delta\mathbf{p}$ is computed by solving an overdetermined equation system, Equation 11. Each pixel in a tracking patch provides one equation and each model freedom (DOF) one variable. The condition number of the linearized motion model M affects how measurement errors propagate into $\Delta\mathbf{p}$, and ultimately if the computation converges or not. In general, it is more difficult to track many DOF. In particular, the homography warp (that incorporates scaling and out-of-plane rotations) causes less apparent image change compared to a 2D translational warp. By tracking a connected 3D model, the tracking convergence is no longer solely dependent on one surface patch alone, and the combination of differently located and oriented patches can give an accurate 3D pose estimate even when each patch would be difficult to track individually.

In the first experiment we compared the tracking stability for the plane based tracker and the traditional homography based tracker. The results are shown in Figure 2 (above). When three regions are individually tracked using an 8DOF homography by the algorithm from [3] (top images) the first region is lost already after 70 frames. The condition numbers for M vary between $4 * 10^6$ and $1 * 10^7$, indicating a numerically ill conditioned situation. When instead the regions are related by the global model, pose is successfully tracked through the whole sequence of 512 frames (middle, bottom of Figure 2). The condition number of the 6DOF (3 rot, 3 trans) model is 1000, which is significantly better than for the 8DOF homography. Imposing constraints on the estimated planes (e.g. roof planes perpendicular to front plane) further stabilizes the trackers (last row of Figure 2) . One of the trackers (the window on the tall house) starts drifting at about frame 250 when using the unconstrained model (middle row of Figure 2). The experiment is illustrated also in `video1` [1] where the red trackers use 8DOF homography the green trackers use general 3D planes and the blue ones constrained 3D planes. The planes that become occluded are eliminated using a Z-buffer algorithm.

One of the main advantages of the proposed approach over traditional SSD tracking is that actual 3D camera pose can be tracked. This is useful for example in robotics or augmented reality applications. In the next experiment we evaluate the accuracy of tracking in an indoor lab scene tracked by a moving camera. Ground truth was obtained by measuring the camera path and performing a Euclidean calibration of the model. Figure 3 shows two tracked frames, and the sequence can be seen in `video2` [1].

Figure 3. Tracking 3D planes. Pose accuracy experiment. `video2` **[1]**

The first test trajectory is a straight line in the horizontal plane of 1m. Figure 4 (left) illustrates the recovered trajectory. To measure the accuracy of the tracking algorithm we calibrated the 3D model for the planes assuming given real dimensions (distance from camera to one plane) so we could get the translation in meters. Here the parallelism constraints imposed between planes (e.g. back wall and Syncrude sign) had a very small influence on the pose accuracy. We found that the trajectory had 0.41 cm mean deviation from a straight line and 3.15 cm mean deviation from the horizontal plane. The recovered line length was 1.10 m, that result in an error of 10% with respect to the measured ground truth. The camera was not rotated along the first trajectory, that corresponds to the measured rotation (error was less than 1.4 degree on average).

We tracked the second trajectory along two perpendicular lines in the horizontal plane. In this experiment, the physical motion was not particularly smooth and the

recorded data therefore also somewhat jumpy. We measured the angle between the two lines fitted to the recovered positions (see Figure 4) as 76°. Hence it had a considerable angular error with respect to the ground truth. The MATLAB implementation of the plane tracking runs at about 3Hz.

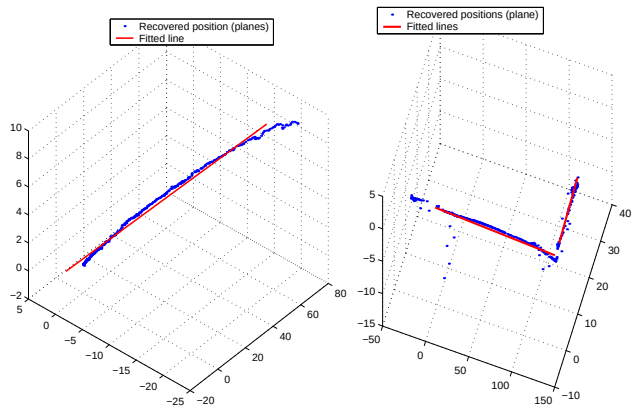


Figure 4. Recovered positions (in 3D space) for the straight line trajectory (left) and the 2 perpendicular lines trajectory (left). The red line are the fitted 3D lines to each line segment.

6. Discussion

We have presented a tracking algorithm that extends the existing SSD homography tracking by computing a global 3D position based on precomputed plane equations. The parameters of the 3D planes are estimated from an initial sequence (about 100 frames) where feature points on the planes are tracked using regular SSD translational trackers. Constraints between planes are also incorporated using a minimal parametrization of the planes. We showed that the proposed tracking algorithm is more stable due to the reduced DOF compared to tracking individual homographies and can handle a large range of motion.

A main advantage of the method is that it tracks full 3D camera position that might be required in applications like robotics or augmented reality. The pose is computed directly from image derivatives with respect to pose parameters that guarantes the best 3D pose update from the linearized model. This is unlike the other model based approaches where 3D pose is estimated from tracked 2D image correspondences.

The present version of the algorithm does not handle partial occlusions and illumination variation. This problem can be solved by using a robust norm like in [7].

References

[1] On-line mpeg movies of the experiments are available. See `videoX` at http://www.cs.ualberta.ca/∼dana/CRV05.

[2] M. Armstrong and A. Zisserman. Robust object tracking. In *Second Asian Conference on Computer Vision*, pages 58–62, 1995.

[3] S. Baker and I. Matthews. *Lucas-Kanade 20 Years On: A Unifying Framework*. Technical Report CMU-RITR02-16, 2002.

[4] A. Bartoli and P. Sturm. Constrained structure and motion from multiple uncalibrated views of a piecewise planar scene. *IJCV - International Journal of Computer Vision*, 52(1):45–64, 2003.

[5] D. Cobzas and M. Jagersand. 3d ssd tracking from uncalibrated video. In *ECCV 2004 Workshop on Spatial Coherence for Visual Motion Analysis (SCVMA)*, 2004.

[6] T. Drummond and R. Cipolla. Real-time visual tracking of complex structures. *PAMI*, 24(7):932–946, July 2002.

[7] G.D. Hager and P.N. Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *PAMI*, 20(10):1025–1039, October 1998.

[8] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[9] B.K.P. Horn. *Computer Vision*. MIT Press, Cambridge, Mass., 1986.

[10] Allan D. Jepson, David J. Fleet, and Thomas F. El-Maraghi. Robust online appearance models for visual tracking. *PAMI*, 25(10):1296–1311, 2003.

[11] D.G. Lowe. Fitting parameterized three-dimensional models to images. *PAMI*, 13(5):441–450, May 1991.

[12] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Int. Joint Conf. on Artificial Intelligence*, 1981.

[13] Y. Ma, S. Soatto, J. Kosecka, and S Sastry. *An Invitation to 3D Vision*. Springer, 2004.

[14] E. Marchand, P. Bouthemy, and F. Chaumette. A 2d-3d model-based approach to real-time visual tracking. *IVC*, 19(13):941–955, November 2001.

[15] Gilles Simon, Andrew W. Fitzgibbon, and Andrew Zisserman. Markerless tracking using planar structures in the scene. In *IEEE and ACM International Symposium on Augmented Reality (ISAR)*, 2000.

[16] Jing Xiao, Simon Baker, Iain Matthews, and Takeo Kanade. Real-time combined 2d+3d active appearance models. In *Proc. of International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

International Journal of Computer Vision 57(3), 159–178, 2004

The 3D Line Motion Matrix and Alignment of Line Reconstructions*

ADRIEN BARTOLI AND PETER STURM
INRIA Rhône-Alpes, 655, av. de l'Europe, 38334 St. Ismier cedex, France
Adrien.Bartoli@inria.fr

Received April 12, 2002; Revised February 20, 2003; Accepted July 3, 2003

Abstract. We study the problem of aligning two 3D line reconstructions in projective, affine, metric or Euclidean space.

We introduce the 6×6 3D line motion matrix that acts on Plücker coordinates. We characterize its algebraic properties and its relation to the usual 4×4 point motion matrix, and propose various methods for estimating 3D motion from line correspondences, based on cost functions defined in images or 3D space. We assess the quality of the different estimation methods using simulated data and real images.

Keywords: lines, reconstruction, motion

1. Introduction

The goal of this paper is to align two reconstructions of a set of 3D lines (Fig. 1). The recovered motions can be used in many areas of computer vision (Demirdjian and Horaud, 2000; Devernay and Faugeras, 1996; Horaud et al., 1997).

Lines are widely used for tracking (Hager and Toyama, 1998; Zhang and Faugeras, 1990), for visual servoing (Andreff et al., 2000) or for pose estimation (Liu et al., 1990) and their reconstruction has been well studied (see e.g. Canny (1986) for image detection, (Schmid and Zisserman, 1997) for matching and Hartley (1997), Spetsakis and Aloimonos (1990), Taylor and Kriegman (1995) and Zhang (1994) for structure and motion).

There are three intrinsic difficulties to motion estimation from 3D line correspondences, even in Euclidean space. Firstly, there is no global linear and minimal parameterization for lines representing their 4 degrees of freedom by 4 global parameters. Secondly, there is no universally agreed error metric for comparing lines. Thirdly, depending on the representation, it may be non trivial to transfer a line between two different bases.

In this paper, which is an extended version of Bartoli and Sturm (2001), we address the problem of motion computation using line reconstructions from images. We assume that two sets of images are given and independently registered. We also assume that correspondences of lines between these two sets are known. Reconstructing lines from each of these image sets provides the two 3D line sets to be aligned.

In projective, affine, metric and Euclidean space, motion is usually represented by 4×4 matrices (homography, affinity, similarity or rigid displacement), with different numbers of parameters, see Hartley and Zisserman (2000) for more details. This representation is well-suited to points and planes represented using homogeneous coordinates. We call it the *usual motion matrix*.

One way to represent 3D lines is to use Plücker coordinates. They are consistent in that they do not depend on particular points or planes used to define a line. On the other hand, transferring a line between bases is not direct (one must either recover two points lying on it, transfer them and form their Plücker coordinates

*A previous conference version appeared in *Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR'01)*, Kauai, Hawaii, USA, vol. I, pp. 287–289, December 2001.

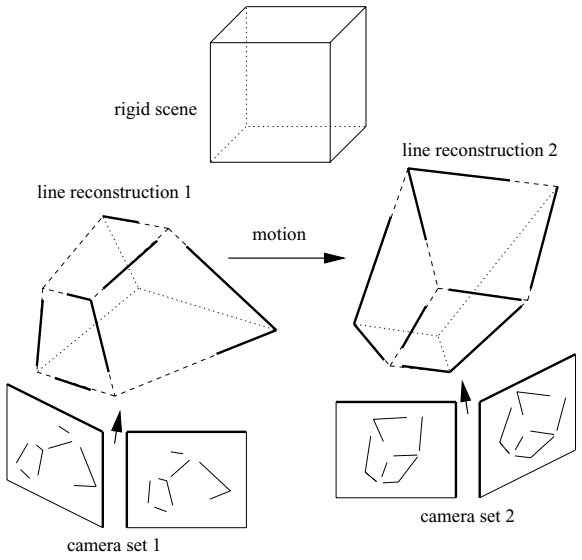


Figure 1. Our problem is to align two corresponding line reconstructions or, equivalently, to estimate the motion between the camera sets.

or transform the 4×4 skew-symmetric Plücker matrix representating the line). The problem with the Plücker matrix representation is that applying the motion is quadratic in the entries of the usual motion matrix which therefore can not be estimated linearly from line matches.

To overcome this, we derive a motion representation that is well-adapted to Plücker coordinates in that it transfers them linearly between bases. The transformation is represented by a 6×6 matrix that we call the *3D line motion matrix*. We characterize its algebraic properties in terms of the usual motion matrix. The expressions obtained were previously known in the Euclidean case (Navab and Faugeras, 1997). We also deal with projective and affine spaces. We give a means of extracting the usual motion matrix from the 3D line motion matrix. A given general 6×6 matrix can then be corrected so that it exactly represents a motion[1].

Using this representation, we derive several estimators for 3D motion from line reconstructions. The motion allows lines to be transferred from the first reconstruction into the second one. Cost functions can therefore be expressed either directly in the second reconstruction basis using 3D entities or in image-related quantities, in terms of the observed and reprojected lines in the second set of images.

Our first method is based on the direct comparison of 3D Plücker coordinates. Two other methods are based on algebraic distances between reprojected lines and either observed lines or points lying on them (such as their end-points). A 6×6 matrix is recovered linearly, then corrected so that it exactly represents a motion. A fourth method uses a more physically meaningful cost function based on orthogonal distances between reprojected lines and points lying on measured lines. This requires non-linear optimization techniques that need an initialization provided by e.g. one of the proposed linear methods. We also propose a means to quasi-linearly optimize this cost function, which does not require a separate initialization method.

Section 2 gives some preliminaries and our notations. We introduce the 3D line motion matrix in Section 3 and show in Section 4 how to extract the usual motion matrix from it. Section 5 shows how these techniques can be used to estimate the motion between two reconstructions of 3D lines. We validate our methods on both simulated data and real images in Sections 6 and 7 respectively, and give our conclusions in Section 8.

2. Preliminaries and Notations

We make no formal distinction between coordinate vectors and physical entities. Equality up to a non-null scale factor is denoted by $\sim$, transposition and transposed inverse by $^\mathsf{T}$ and $^{-\mathsf{T}}$, and the skew-symmetric 3×3 matrix associated with the cross product by $[\cdot]_\times$, i.e. $[\mathbf{v}]_\times \mathbf{q} = \mathbf{v} \times \mathbf{q}$. Vectors are typeset using bold fonts ($\mathbf{L}$, $\mathbf{l}$), matrices using sans-serif fonts (H, A, D) and scalars in italics. Everything is represented in homogeneous coordinates. Bars represent inhomogeneous leading parts of vectors or matrices, e.g. $\mathbf{M}^\mathsf{T} \sim (\bar{\mathbf{M}}^\mathsf{T}\ m)$. We use $\|\mathbf{v}\|$ to designate the $\mathcal{L}_2$-norm of vector $\mathbf{v}$.

Plücker Line Coordinates. Given two 3D points $\mathbf{M}^\mathsf{T} \sim (\bar{\mathbf{M}}^\mathsf{T}\ m)$ and $\mathbf{N}^\mathsf{T} \sim (\bar{\mathbf{N}}^\mathsf{T}\ n)$, one can form the Plücker coordinates of the line joining them as a 6-vector $\mathbf{L}^\mathsf{T} \sim (\mathbf{a}^\mathsf{T}\ \mathbf{b}^\mathsf{T})$ defined up to scale (Hartley and Zisserman, 2000):

$$\begin{cases} \mathbf{a} = \bar{\mathbf{M}} \times \bar{\mathbf{N}} \\ \mathbf{b} = m\bar{\mathbf{N}} - n\bar{\mathbf{M}}. \end{cases} \tag{1}$$

Note that other choices of constructing 6-vectors of Plücker coordinates are possible. Every choice goes with a bilinear constraint that 6-vectors have to satisfy in order to represent valid line coordinates. For our definition, the constraint is $\mathbf{a}^\mathsf{T}\mathbf{b} = 0$. An alternative representation is the *Plücker matrix* L, which is related

to $\mathbf{L}$ via:

$$\mathsf{L} \sim \begin{pmatrix} [\mathbf{a}]_\times & -\mathbf{b} \\ \mathbf{b}^\mathsf{T} & 0 \end{pmatrix},$$

and to point coordinates by:

$$\mathsf{L} \sim \mathbf{M}\mathbf{N}^\mathsf{T} - \mathbf{N}\mathbf{M}^\mathsf{T}.$$

This is a skew-symmetric rank-2 4×4 matrix.

Usual Motion Representation. Transformations in projective, affine, metric and Euclidean spaces are usually represented by 4×4 matrices. In the general projective case, the matrices are unconstrained, while in the affine, metric and Euclidean cases they have the following forms, where R is a 3×3 rotation matrix and the other blocks do not have any special form:

Projective: homography H	Affine: affinity A	Metric: similarity S	Euclidean: displacement D
$\begin{pmatrix} \bar{\mathsf{H}} & \mathbf{h}_1 \\ \mathbf{h}_2^\mathsf{T} & h \end{pmatrix}$	$\begin{pmatrix} \bar{\mathsf{A}} & \mathbf{t} \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix}$	$\begin{pmatrix} s\mathsf{R} & \mathbf{t} \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix}$	$\begin{pmatrix} \mathsf{R} & \mathbf{t} \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix}$

This block partitioning of H, A, S and D will be used to define the corresponding 3D line motion matrices in Section 3.

Camera Settings. We consider two sets of independently registered cameras. In the projective space, without loss of generality, we can express each set in a canonical reconstruction basis (Luong and Vieville, 1996), and in particular, we can set reference cameras, e.g. the first ones to:

$$\mathsf{P} \sim \mathsf{P}' \sim (\mathrm{I}\ \mathbf{0}),$$

where P and P' are the reference cameras of the first and the second set respectively, that we call the *first* and the *second reference cameras*. Let H be the 4×4 usual homography matrix mapping points from the first basis to the second one and $\mathsf{P}'' \sim (\mathsf{P}''\ \mathbf{p}'')$ the reference camera of the second set expressed in the first basis. These notations are illustrated on Fig. 2. We denote by $\boldsymbol{\pi}_\infty$ and $\boldsymbol{\pi}'_\infty$ the planes with coordinates $(0\ 0\ 0\ 1)^\mathsf{T}$ of the two reconstructions. In affine, metric or Euclidean reconstructions, these are the plane at infinity, while in projective reconstructions, they will in general correspond to two finite planes. Let $\boldsymbol{\pi}''_\infty$ be the plane in the first reconstruction corresponding to $\boldsymbol{\pi}'_\infty$. We also make use of the lines $\mathbf{L}_\infty$ and $\mathbf{L}'_\infty$ lying on $\boldsymbol{\pi}_\infty$ and $\boldsymbol{\pi}'_\infty$ respectively and related by H. In particular, $\mathbf{L}_\infty = \boldsymbol{\pi}_\infty \cap \boldsymbol{\pi}''_\infty = \{\mathbf{Q} \sim (\bar{\mathbf{Q}}^\mathsf{T}\ q)^\mathsf{T} \mid \bar{\mathbf{Q}}^\mathsf{T}\mathbf{h}_2 = q = 0\}$.

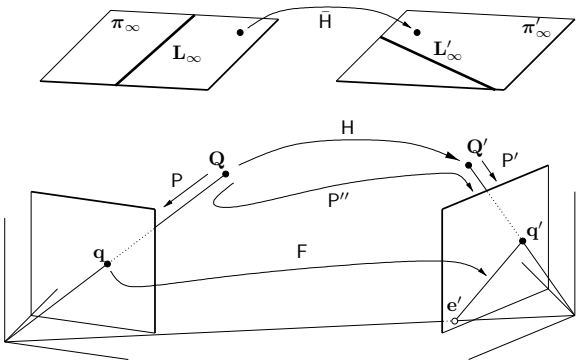


Figure 2. Camera settings. P is the perspective projection matrix from the first basis to the first reference camera and P' from the second basis to the second reference camera. P'' is the projection matrix of the second reference camera expressed in the first basis, i.e. it projects points expressed in the first basis in the second reference camera. P and P' are assumed known while P'', which depends on the motion parameters, is unknown. These settings can be easily transposed to the affine, metric and Euclidean cases.

Let us give a geometrical interpretation of the components of H. This will be useful to subsequently investigate the corresponding properties of the 3D line motion matrices in Section 3. The fundamental matrix (Luong and Faugeras, 1996) between the reference views is given by:

$$\mathsf{F} \sim [\mathbf{p}'']_\times \bar{\mathsf{P}}'' \sim [\mathbf{h}_1]_\times \bar{\mathsf{H}}.$$

Indeed, we have $(\bar{\mathsf{H}}\ \mathbf{h}_1) \sim (\bar{\mathsf{P}}''\ \mathbf{p}'')$ since:

$$\begin{aligned} \mathsf{P}'' &\sim \mathsf{P}'\mathsf{H} \\ (\bar{\mathsf{P}}''\ \mathbf{p}'') &\sim (\mathrm{I}\ \mathbf{0}) \begin{pmatrix} \bar{\mathsf{H}} & \mathbf{h}_1 \\ \mathbf{h}_2^\mathsf{T} & h \end{pmatrix} \\ (\bar{\mathsf{P}}''\ \mathbf{p}'') &\sim (\bar{\mathsf{H}}\ \mathbf{h}_1). \end{aligned}$$

These results may be easily specialized to the affine, metric and Euclidean spaces. The other parts of H can be interpreted as follows:

- $\bar{\mathsf{H}}$ is the 2D plane homography for points between the reference views induced by the plane $\boldsymbol{\pi}_\infty$. Indeed, let us consider $\mathbf{Q} \in \boldsymbol{\pi}_\infty$. One can easily check that $\mathbf{Q}^\mathsf{T} \sim (\mathbf{q}^\mathsf{T}\ 0)$ where $\mathbf{q}$ is the corresponding point in the first reference image and $\mathsf{P}'\mathsf{H}\mathbf{Q} \sim \bar{\mathsf{H}}\mathbf{q}$. Hence, if we deal

with affine, metric or Euclidean reconstructions, then $\bar{\mathsf{H}}$ is the infinite homography (Hartley and Zisserman, 2000) between the two reference views.

- $\mathbf{h}_1$ contains the coordinates of the second epipole since:

$$\mathsf{F}^\mathsf{T}\mathbf{h}_1 \sim \bar{\mathsf{P}''}^\mathsf{T}[\mathbf{p}'']_\times{}^\mathsf{T}\mathbf{h}_1 \sim \bar{\mathsf{H}}^\mathsf{T}[\mathbf{h}_1]_\times{}^\mathsf{T}\mathbf{h}_1 \sim \mathbf{0}.$$

- $\boldsymbol{\pi}''_\infty \sim (\mathbf{h}_2^\mathsf{T}\ h)^\mathsf{T}$ contains the coordinates of the plane at infinity of the second basis expressed in the first basis. Indeed, $\boldsymbol{\pi}''_\infty \sim \mathsf{H}^\mathsf{T}\boldsymbol{\pi}'_\infty \sim (\mathbf{h}_2^\mathsf{T}\ h)^\mathsf{T}$.

Perspective Projection Matrix for Lines. With our choice of Plücker coordinates (1), the image projection of a line (Faugeras and Mourrain, 1995; Hartley and Zisserman, 2000) becomes the 3×6 linear transformation $\tilde{\mathsf{P}}$:

$$\tilde{\mathsf{P}} \sim (\det(\bar{\mathsf{P}})\bar{\mathsf{P}}^{-\mathsf{T}}\ [\mathbf{p}]_\times\bar{\mathsf{P}}), \tag{2}$$

where $\mathsf{P} \sim (\bar{\mathsf{P}}\ \mathbf{p})$ is the 3×4 perspective camera matrix. This result can be easily demonstrated by finding the image line joining the projections of two points on the 3D line. An explicit proof is given in the appendix. Specific forms for affine cameras and calibrated perspective cameras are straightforward to derive.

3. The 3D Line Motion Matrix

In this section, we define and examine the properties of the *3D line motion matrix* for the projective space first and then specialize it to the affine, metric and Euclidean spaces respectively.

3.1. The 3D Line Homography Matrix

The Plücker coordinates of a line, expressed in two different bases, are linearly linked. The 6×6 *matrix* $\tilde{\mathsf{H}}$ *that we call the* 3D line homography matrix *describes the transformation in the projective case and can be parameterized as:*

$$\tilde{\mathsf{H}} \sim \begin{pmatrix} \det(\bar{\mathsf{H}})\bar{\mathsf{H}}^{-\mathsf{T}} & [\mathbf{h}_1]_\times\bar{\mathsf{H}} \\ -\bar{\mathsf{H}}[\mathbf{h}_2]_\times & h\bar{\mathsf{H}} - \mathbf{h}_1\mathbf{h}_2^\mathsf{T} \end{pmatrix}, \tag{3}$$

where H *is the usual* 4×4 *homography matrix for points. If* $\mathbf{L}^\mathsf{T} \sim (\mathbf{a}^\mathsf{T}\ \mathbf{b}^\mathsf{T})$ *are Plücker line coordinates* (*i.e.* $\mathbf{a}^\mathsf{T}\mathbf{b} = 0$), *then* $\tilde{\mathsf{H}}\mathbf{L}$ *are the Plücker coordinates of the transformed line* (*i.e.* $\tilde{\mathsf{H}}\mathbf{L}$ *satisfies the bilinear Plücker constraint*).

The proof of this result is provided in the appendix. Note that $\tilde{\mathsf{H}}$ is a 6×6 matrix defined up to scale and subject to 20 non-linear constraints since the projective motion has 15 degrees of freedom. Other algebraic properties directly follow from Eq. (3). Firstly, the determinant of $\tilde{\mathsf{H}}$ can be expressed in terms of that of H as:

$$\det(\tilde{\mathsf{H}}) = (\det(\mathsf{H}))^3,$$

which means that if H is full-rank, then $\tilde{\mathsf{H}}$ is also full-rank. Secondly, let $\tilde{\mathsf{G}}$ denote the 3D line motion matrix corresponding to the usual motion matrix G. Then:

$$\mathsf{G} \sim \mathsf{H}^\mathsf{T} \Leftrightarrow \tilde{\mathsf{G}} \sim \tilde{\mathsf{H}}^\mathsf{T}$$
$$\mathsf{G} \sim \mathsf{H}^{-1} \Leftrightarrow \tilde{\mathsf{G}} \sim \tilde{\mathsf{H}}^{-1}.$$

These properties can be easily verified using any linear algebra symbolic manipulation software such as MAPLE.

Consistency Constraints on $\tilde{\mathsf{H}}$. Let $\tilde{\mathsf{H}}$ be subdivided in 3×3 blocks as:

$$\tilde{\mathsf{H}} \sim \begin{pmatrix} \tilde{\mathsf{H}}_{11} & \tilde{\mathsf{H}}_{12} \\ \tilde{\mathsf{H}}_{21} & \tilde{\mathsf{H}}_{22} \end{pmatrix}.$$

By $\mathbf{r}_{ij,k}$, we denote the k-th row of matrix $\tilde{\mathsf{H}}_{ij}$ and by $\mathbf{c}_{ij,l}$ its l-th column. We express the 20 non-linear consistency constraints that must hold on $\tilde{\mathsf{H}}$ as follows:

$$\begin{aligned}
(\mathbf{r}_{11,k}^\mathsf{T})(\mathbf{r}_{12,k}) &= 0, \quad k = 1\ldots3 \\
(\mathbf{r}_{21,k}^\mathsf{T})(\mathbf{r}_{22,k}) &= 0, \quad k = 1\ldots3 \\
(\mathbf{c}_{11,l}^\mathsf{T})(\mathbf{c}_{21,l}) &= 0, \quad l = 1\ldots3 \\
(\mathbf{c}_{12,l}^\mathsf{T})(\mathbf{c}_{22,l}) &= 0, \quad l = 1\ldots3 \\
(\mathbf{r}_{11,k}^\mathsf{T})(\mathbf{r}_{22,k'}) + (\mathbf{r}_{12,k}^\mathsf{T})(\mathbf{r}_{21,k'}) &= 0, \quad k = 1\ldots3, \\
&\qquad k' = 1\ldots3, k \neq k' \\
(\mathbf{r}_{11,1}^\mathsf{T})(\mathbf{r}_{22,1}) + (\mathbf{r}_{12,1}^\mathsf{T})(\mathbf{r}_{21,1}) &= (\mathbf{r}_{11,2}^\mathsf{T})(\mathbf{r}_{22,2}) \\
&\quad + (\mathbf{r}_{12,2}^\mathsf{T})(\mathbf{r}_{21,2}) \\
&= (\mathbf{r}_{11,3}^\mathsf{T})(\mathbf{r}_{22,3}) \\
&\quad + (\mathbf{r}_{12,3}^\mathsf{T})(\mathbf{r}_{21,3}).
\end{aligned}$$

A detailled derivation is given in the appendix. Note that these contraints are bilinear in the entries of $\tilde{\mathsf{H}}$. These constraints are important in that they characterize the algebraic structure of a 3D line homography matrix.

Geometric Interpretation of $\tilde{\mathsf{H}}$

- *the upper-left* 3×3 *block* $\tilde{\mathsf{H}}_{11} \sim \bar{\mathsf{H}}^{-\mathsf{T}}$ *is the 2D plane homography for lines, between the reference views, induced by* $\boldsymbol{\pi}_\infty$. This follows from the observation made in Section 2 that $\tilde{\mathsf{H}}_{11}^{-\mathsf{T}} \sim \bar{\mathsf{H}}$ is the corresponding plane homography acting on points.
- *the upper-right* 3×3 *block* $\tilde{\mathsf{H}}_{12}$ *is the fundamental matrix between the reference views*, i.e. $\tilde{\mathsf{H}}_{12} = \mathsf{F}$. Indeed, $\tilde{\mathsf{H}}_{12} = [\mathbf{h}_1]_\times \bar{\mathsf{H}} \sim [\mathbf{p}'']_\times \bar{\mathsf{P}}'' \sim \mathsf{F}$ (cf Section 2).
- *the upper* 3×6 *block* $(\tilde{\mathsf{H}}_{11}\ \tilde{\mathsf{H}}_{12})$ *is the perspective projection matrix for Plücker line coordinates from the first basis to the second reference view*. Indeed,

$$\begin{aligned}(\tilde{\mathsf{H}}_{11}\ \tilde{\mathsf{H}}_{12}) &= (\det(\bar{\mathsf{H}})\bar{\mathsf{H}}^{-\mathsf{T}}\ [\mathbf{h}_1]_\times \bar{\mathsf{H}}) \\ &\sim (\det(\bar{\mathsf{P}}'')\bar{\mathsf{P}}''\ [\mathbf{p}]_\times \bar{\mathsf{P}}) \sim \tilde{\mathsf{P}},\end{aligned}$$

  where $\tilde{\mathsf{P}}$ corresponds to the perspective projection matrix (2) for Plücker line coordinates (1) and P'' is the perspective projection matrix from the first basis to the second reference view (see Fig. 2).
- *the lower-left* 3×3 *block* $\tilde{\mathsf{H}}_{21} \sim \bar{\mathsf{H}}[\mathbf{h}_2]_\times$ *is a degenerate line-to-point homography between the reference views*, which can be interpreted as follows. Let $\mathbf{l}$ be an image line in the first reference view. The intersection of $\mathbf{l}$ and the line of equation $\mathbf{h}_2$ is a point $\mathbf{q} \sim [\mathbf{h}_2]_\times \mathbf{l}$. The backprojection of $\mathbf{q}$ onto $\boldsymbol{\pi}_\infty$ lies on $\mathbf{L}_\infty$ and is given by $\mathbf{Q}_\infty^{\mathsf{T}} \sim (\mathbf{q}^{\mathsf{T}}\ 0)$. Its corresponding point in the second reconstruction lies therefore on $\mathbf{L}'_\infty$ and is given by $\mathbf{Q}'^{\mathsf{T}}_\infty \sim (\mathbf{q}'^{\mathsf{T}}\ 0)$ with $\mathbf{q}' \sim \bar{\mathsf{H}}[\mathbf{h}_2]_\times \mathbf{l}$. Projecting $\mathbf{Q}'_\infty$ into the second reference view gives finally $\mathbf{q}' \sim \bar{\mathsf{H}}[\mathbf{h}_2]_\times \mathbf{l}$.

  So, $\tilde{\mathsf{H}}_{21}$ is somehow reciprocal to a fundamental matrix that maps points to lines. Whereas a fundamental matrix gives matching constraints for image points, $\tilde{\mathsf{H}}_{21}$ does not give any matching constraints for general lines (there are none between two views).
- *the lower-right* 3×3 *block* $\tilde{\mathsf{H}}_{22}$ *is the 2D plane homography for points, induced by* $\boldsymbol{\pi}_\infty$ *(expressed in the second basis), between the reference views*. Indeed, we have shown that $\bar{\mathsf{H}}$ is a plane homography and $\mathbf{h}_1$ the second epipole corresponding to the pair of reference views. Using the formulation of Luong and Vieville (1996), $\tilde{\mathsf{H}}_{22} = h\bar{\mathsf{H}} - \mathbf{h}_1\mathbf{h}_2^{\mathsf{T}}$ is the 2D plane homography induced by the plane $(\mathbf{h}_2^{\mathsf{T}}\ h)^{\mathsf{T}}$, which are the coordinates of the plane at infinity of the second basis expressed in the first one.
- *the lower* 3×6 *block* $(\tilde{\mathsf{H}}_{21}\ \tilde{\mathsf{H}}_{22})$ *transfers a line* $\mathbf{L}$ *from the first basis to the second one and projects its intersection point with* $\boldsymbol{\pi}'_\infty$ *into the second reference view*. This can be seen as follows. $\mathbf{Q}'^{\mathsf{T}}_\infty \sim (\mathbf{q}'^{\mathsf{T}}\ 0)$ with $\mathbf{q}' \sim (\tilde{\mathsf{H}}_{21}\ \tilde{\mathsf{H}}_{22})\mathbf{L}$ is the point at infinity of $\tilde{\mathsf{H}}\mathbf{L}$ which projects to $\mathbf{q}'$ in the second reference view.

3.2. The 3D Line Affinity Matrix

For affine reconstructions, the 3D line motion matrix has the following form and we call it the 3D line affinity matrix:

$$\tilde{\mathsf{A}} \sim \begin{pmatrix} \det(\bar{\mathsf{A}})\bar{\mathsf{A}}^{-\mathsf{T}} & [\mathbf{t}]_\times \bar{\mathsf{A}} \\ 0 & \bar{\mathsf{A}} \end{pmatrix}. \tag{4}$$

This result is obtained by specializing the 3D line homography matrix (3). The geometric interpretation of this matrix is very similar to the projective case. In particular, $\bar{\mathsf{A}}$ is the homography at infinity between the two reference views. Note that a 6×6 matrix defined up to scale representing an affinity is subject to 23 constraints, many of them being linear or bilinear.

3.3. The 3D Line Similarity Matrix

In metric space, the 3D line motion matrix has the following form and we call it the 3D line similarity matrix*:*

$$\tilde{\mathsf{S}} \sim \begin{pmatrix} s\mathsf{R} & [\mathbf{t}]_\times \mathsf{R} \\ 0 & \mathsf{R} \end{pmatrix}. \tag{5}$$

This result is obtained by specializing the 3D line homography matrix (3). The geometric interpretation of this matrix is straightforward. The 3×3 upper-right block $[\mathbf{t}]_\times \mathsf{R}$ is the essential matrix between the reference views while the other two non-zero 3×3 blocks give the rotation matrix between the reference cameras, as well as the relative scale of the two reconstructions. Note that a 6×6 matrix defined up to scale representing a similarity is subject to 28 constraints.

3.4. The 3D Line Displacement Matrix

In Euclidean space, the 3D line motion matrix has the following form and we call it the 3D line displacement matrix*:*

$$\tilde{\mathsf{D}} \sim \begin{pmatrix} \mathsf{R} & [\mathbf{t}]_\times \mathsf{R} \\ 0 & \mathsf{R} \end{pmatrix}. \tag{6}$$

This result is obtained by specializing the 3D line homography matrix (3). It coincides with that obtained in Navab and Faugeras (1997). The geometric interpretation of this matrix is very similar to the metric case. Note that an homogeneous 6×6 matrix representing a rigid displacement is subject to 29 constraints.

4. Extracting the Usual Motion Matrix from the 3D Line Motion Matrix

Given a 6×6 3D line motion matrix, one can extract the corresponding motion parameters, i.e. the usual 4×4 motion matrix. In this section, we show how to extract a usual motion matrix from a 3D line motion matrix in projective, affine, metric and Euclidean spaces. We also give solutions for the cases where the 6×6 matrix considered is corrupted by noise and therefore does not exactly correspond to a motion, i.e. it differs from the forms (3), (4), (5) or (6).

4.1. Projective Space

We give an algorithm for the noise-free case in Table 1. Its simple proof is given in the appendix. In the presence of noise, $\tilde{\mathsf{H}}$ does not exactly satisfy the constraints (3) and steps 2–4 have to be achieved in a least squares sense (see below). From there, one can further improve the result, e.g. by non-linear minimization of the Frobenius norm between the given (inexact) line homography matrix and the one corresponding to the recovered usual motion parameters.

We give one way to perform a least squares estimation. Other algorithms might be possible. Steps 2 and 3 require to fit a skew-symmetric 3×3-matrix $[\mathbf{w}]_\times$ to a general 3×3 matrix W. The following solution minimizes the Frobenius norm of $[\mathbf{w}]_\times - \mathsf{W}$: $\mathbf{w} = \frac{1}{2}(W_{32} - W_{23} \quad W_{13} - W_{31} \quad W_{21} - W_{12})$. Step 4 requires to fit a scaled 3×3 identity matrix $\lambda\mathrm{I}$ to a general 3×3-matrix Λ. The following solution minimizes the Frobenius norm of $\lambda\mathrm{I} - \Lambda$: $\lambda = \frac{1}{3}\sum_i \Lambda_{ii}$.

Table 1. Extracting the homography matrix from the 3D line homography matrix.

Let $\tilde{\mathsf{H}}$ be subdivided in 3×3 blocks as: $\tilde{\mathsf{H}} \sim \left(\begin{smallmatrix} \mathsf{H}_{11} & \mathsf{H}_{12} \\ \mathsf{H}_{21} & \mathsf{H}_{22} \end{smallmatrix}\right)$.

1. $\bar{\mathsf{H}}$: compute $\bar{\mathsf{H}} = \pm\sqrt{|\det(\tilde{\mathsf{H}}_{11})|}\ \tilde{\mathsf{H}}_{11}^{-\mathsf{T}}$
2. $\mathbf{h}_1$: compute $[\mathbf{h}_1]_\times = \pm\tilde{\mathsf{H}}_{12}\bar{\mathsf{H}}^{-1}$
3. $\mathbf{h}_2$: compute $[\mathbf{h}_2]_\times = \mp\bar{\mathsf{H}}^{-1}\tilde{\mathsf{H}}_{21}$
4. h: compute h via $h\mathrm{I}_{3\times3} = \pm(\tilde{\mathsf{H}}_{22} + \mathbf{h}_1\mathbf{h}_2^{\mathsf{T}})\bar{\mathsf{H}}^{-1}$.

Note that extracted values are defined up to a global change of sign.

Table 2. Extracting the affinity matrix from the 3D line affinity matrix.

Let $\tilde{\mathsf{A}}$ be subdivided in 3×3 blocks as: $\tilde{\mathsf{A}} \sim \left(\begin{smallmatrix} \mathsf{A}_{11} & \mathsf{A}_{12} \\ 0 & \mathsf{A}_{22} \end{smallmatrix}\right)$.

1. $\bar{\mathsf{A}}$: compute $\bar{\mathsf{A}} = \tilde{\mathsf{A}}_{22}$
2. $\mathbf{t}$: compute $[\mathbf{t}]_\times = \tilde{\mathsf{A}}_{12}\bar{\mathsf{A}}^{-1}$.

4.2. Affine Space

We give a straightforward algorithm in Table 2. This algorithm is valid for the noise-free case. For the noisy case, one can modify the proposed steps as follows.

For step 1, $\bar{\mathsf{A}}$ can be recovered from $\tilde{\mathsf{A}}_{11}$ as well as $\tilde{\mathsf{A}}_{22}$. Their average, possibly weighted, can be used to recover $\bar{\mathsf{A}}$. Step 2 can be conducted as indicated for the projective case in the previous section.

4.3. Metric Space

We give an algorithm for the noise-free case in Table 3. For the noisy case, one might average the two versions of R available from $\tilde{\mathsf{D}}$ as $\mathsf{R} = \frac{\tilde{\mathsf{D}}_{11}/s^2 + \tilde{\mathsf{D}}_{22}}{2}$ and correct the result so that the obtained matrix exactly represents a rotation by using e.g. (Horn et al., 1988) or $\mathsf{R} \leftarrow \mathsf{U}\mathsf{V}^{\mathsf{T}}$ where $\mathsf{R} = \mathsf{U}\Sigma\mathsf{V}^{\mathsf{T}}$ is the SVD (Singular Value Decomposition) of R. Step 4 can be achieved as indicated for the affine and projective cases.

4.4. Euclidean Space

We give an algorithm for the noise-free case in Table 4. For the noisy case, one might average the two versions of R available from $\tilde{\mathsf{D}}$ as $\mathsf{R} = \frac{\tilde{\mathsf{D}}_{11} + \tilde{\mathsf{D}}_{22}}{2}$ and correct the result so that the obtained matrix exactly represents a

Table 3. Extracting the similarity matrix from the 3D line similarity matrix.

Let $\tilde{\mathsf{S}}$ be subdivided in 3×3 blocks as: $\tilde{\mathsf{S}} \sim \left(\begin{smallmatrix} \mathsf{S}_{11} & \mathsf{S}_{12} \\ 0 & \mathsf{S}_{22} \end{smallmatrix}\right)$.

1. normalize $\tilde{\mathsf{S}}$ such that $\det(\tilde{\mathsf{S}}_{22}) = 1$;
2. s: compute $s = \sqrt[6]{\det(\tilde{\mathsf{S}}_{11})}$;
3. R: compute $\mathsf{R} = \tilde{\mathsf{S}}_{22}$;
4. $\mathbf{t}$: compute $[\mathbf{t}]_\times = \tilde{\mathsf{S}}_{12}\mathsf{R}^{\mathsf{T}}$.

Table 4. Extracting the displacement matrix from the 3D line displacement matrix.

Let $\tilde{\mathsf{D}}$ be subdivided in 3×3 blocks as: $\tilde{\mathsf{D}} \sim \begin{pmatrix} \mathsf{D}_{11} & \mathsf{D}_{12} \\ 0 & \mathsf{D}_{22} \end{pmatrix}$.
1. normalize $\tilde{\mathsf{D}}$ such that $\det(\tilde{\mathsf{D}}_{11}) = 1$;
2. R: compute $\mathsf{R} = \tilde{\mathsf{D}}_{11}$;
3. $\mathbf{t}$: compute $[\mathbf{t}]_\times = \tilde{\mathsf{D}}_{12}\mathsf{R}^\mathsf{T}$.

rotation as in the metric case. Step 3 can be achieved as indicated for the affine and projective cases.

5. Aligning Two Line Reconstructions

We now describe how the 3D line motion matrix can be used to align two sets of N corresponding 3D lines expressed in Plücker coordinates. We examine the projective case but the method can also be used for affine, metric or Euclidean frames. We assume that the two sets of cameras are independently weakly calibrated, i.e. their projection matrices are known up to a 3D homography, so that a projective basis is attached to each set (Luong and Vieville, 1996). Lines can be projectively reconstructed in these two bases. Our goal is to align these 3D lines i.e. to find the projective motion between the two bases, using the line reconstructions.

5.1. General Estimation Scheme

Estimation is performed by finding $\arg\min_{\tilde{\mathsf{H}}} \mathcal{C}$ where $\mathcal{C}$ is the cost function considered. The scale ambiguity is removed by using the additional constraint $\|\tilde{\mathsf{H}}\|^2 = 1$. Non-linear optimization is performed directly on the usual motion parameters (the 16 entries of H and using the constraint $\|\mathsf{H}\|^2 = 1$) whereas the other estimators determine an approximate 3D line homography matrix $\tilde{\mathsf{H}}$ first, then recover the usual homography matrix using the algorithm in Table 1. Most employed cost functions are non-symmetric, taking into account the errors only in the second set of images.

5.2. Estimation Based on a 3D Cost Function

Our first alignment method is "*Lin_3D*". One of the difficulties of using 3D lines for motion estimation is that there is no universally agreed error metric between two 3D lines. For that reason, we propose only one estimator based on 3D entities. This estimator is based on a direct comparison of Plücker coordinates. A measure of the distance between two 3D lines $\mathbf{L}$ and $\hat{\mathbf{L}}$ is given by:

$$d_{3\mathrm{D}}^2(\mathbf{L}, \hat{\mathbf{L}}) = \sum_{k\in\{1\ldots6\}} \left(\sum_{l\in\{1\ldots6\}, l\neq k} (L_k\hat{L}_l - L_l\hat{L}_k)^2 \right).$$

It can be constructed as follows. Consider the 6×6 matrix $\mathsf{Z} = \mathbf{L}\hat{\mathbf{L}}^\mathsf{T} - \hat{\mathbf{L}}\mathbf{L}^\mathsf{T}$. If $\mathbf{L}$ and $\hat{\mathbf{L}}$ are identical, all entries of Z vanish. Therefore, $d_{3\mathrm{D}}(\mathbf{L}, \hat{\mathbf{L}}) = \frac{\|\mathsf{Z}\|_\mathcal{F}}{\sqrt{2}}$, where $\|\cdot\|_\mathcal{F}$ is the Frobenius matrix norm, can be used as a distance measure between $\mathbf{L}$ and $\hat{\mathbf{L}}$.

The distance $d_{3\mathrm{D}}$ induces the following cost function:

$$\mathcal{C}_{3\mathrm{D}} = \sum_i d_{3\mathrm{D}}^2(\mathbf{L}_{2_i}, \hat{\mathbf{L}}_{2_i}),$$

where $\mathbf{L}_{bi}$ is the i-th reconstructed line in the b-th basis ($b \in \{1, 2\}$) and $\hat{\mathbf{L}}_{2_i} \sim \tilde{\mathsf{H}}\mathbf{L}_{1_i}$ is the estimated line after transfer from the first to the second basis.

Finding $\tilde{\mathsf{H}}$ that minimizes $\mathcal{C}_{3\mathrm{D}}$ is a linear least squares problem. Concretely, it may be solved by finding the null-vector of a $30N \times 36$ matrix, using e.g. SVD.

5.3. Estimation Based on 2D (Image-Related) Cost Functions

The following cost functions are based on minimizing discrepancies between reprojected lines and lines extracted in images. Concretely, we consider the projections in the second set of images, of 3D lines in the first set, after transfer using the $\tilde{\mathsf{H}}$ to be estimated. The discrepancy of these reprojected lines and observed ones is measured by comparing 2D lines directly or by computing the distance between reprojected lines and points on observed ones. The following cost functions are expressed in terms of observed image lines, their end-points or an arbitrary number of points along the line, and reprojected lines in the second set of images. These points need not be corresponding points.

If end-points are not available then they can be hallucinated, e.g. by intersecting the image lines with the image boundaries. The linear and quasi-linear methods need at least 7 lines to solve for the motion while the non-linear one needs 5 (which is the minimum number) but requires an initial guess.

We derive a joint projection matrix mapping a 3D line to a set of image lines in the second set of images. Let P'_j be the projection matrices of the M images corresponding to the second set of cameras (expressed in the second basis) and $\tilde{\mathsf{P}}'_j$ the corresponding 3×6 line projection matrices (cf Eq. (2)). Similarly, let P_j and $\tilde{\mathsf{P}}_j$ be the point and line projection matrices for the first set of cameras (expressed in the first basis).

Linear Estimation 1. Our second alignment method "*Lin_2D_1*" directly uses the line equations in the images. End-points need not be defined. We define an algebraic measure of the distance between two image lines $\mathbf{l}$ and $\hat{\mathbf{l}}$ by $d^2(\mathbf{l}, \hat{\mathbf{l}}) = \|\mathbf{l} \times \hat{\mathbf{l}}\|^2$. This distance does not have any direct physical meaning, but it is zero if the two lines are identical and simple in that it is bilinear. If $\mathbf{l}$ and $\hat{\mathbf{l}}$ had unit norm and if they were interpreted as vectors in Euclidean 3D space, then $d^2(\mathbf{l}, \hat{\mathbf{l}})$ would be the absolute value of the sine of the angle between them.

This distance induces the cost function:

$$\mathcal{C}_1 = \sum_i \sum_j d^2(\mathbf{l}_{ij}, \hat{\mathbf{l}}_{ij}),$$

where $\mathbf{l}_{ij}$ is the i-th observed line in the j-th image of the second set and $\hat{\mathbf{l}}_{ij}$ the corresponding reprojection: $\hat{\mathbf{l}}_{ij} \sim \tilde{\mathsf{P}}'_j \tilde{\mathsf{H}} \mathbf{L}_{1_i}$. We normalize observed image lines such that $\|\mathbf{l}_{ij}\|^2 = 1$. Finding $\tilde{\mathsf{H}}$ that minimizes $\mathcal{C}_1$ is a linear least squares problem that may be solved by computing the null-vector of a $3\,MN \times 36$ matrix using e.g. SVD.

Linear Estimation 2. Our third method "*Lin_2D_2*" uses points lying on the lines in the second image set and the algebraic distance $d_a^2(\mathbf{x}, \mathbf{l}) = (\mathbf{x}^\mathsf{T}\mathbf{l})^2$ between an image point $\mathbf{x}$ and a line $\mathbf{l}$. This distance would have a physical meaning, i.e. the orthogonal distance between $\mathbf{x}$ and $\mathbf{l}$, if they were normalized such that $x_3 = 1$ and $l_1^2 + l_2^2 = 1$.

If we consider the end-points $\mathbf{x}_{ij}$ and $\mathbf{y}_{ij}$ of each line i of the j-th image of the second image set, this gives the cost function:

$$\mathcal{C}_2 = \sum_i \sum_j \left(d_a^2(\mathbf{x}_{ij}, \hat{\mathbf{l}}_{ij}) + d_a^2(\mathbf{y}_{ij}, \hat{\mathbf{l}}_{ij})\right).$$

One can observe that an arbitrary number of points could be incorporated in $\mathcal{C}_2$ in a straightforward manner. Finding $\tilde{\mathsf{H}}$ that minimizes $\mathcal{C}_2$ is a linear least squares problem that may be solved by computing the null-vector of a $2MN \times 36$ matrix using e.g. SVD.

Non-Linear Estimation 1. Our fourth method "*NLin_2D_1*" uses a geometrical cost function based on the orthogonal distance between reprojected 3D lines and their measured end-points (Liebowitz and Zisserman, 1998), defined as $d_\perp^2(\mathbf{x}, \mathbf{l}) = \frac{(\mathbf{x}^\mathsf{T}\mathbf{l})^2}{l_1^2 + l_2^2}$ (provided $x_3 = 1$):

$$\mathcal{C}_3 = \sum_i \sum_j \left(d_\perp^2(\mathbf{x}_{ij}, \hat{\mathbf{l}}_{ij}) + d_\perp^2(\mathbf{y}_{ij}, \hat{\mathbf{l}}_{ij})\right).$$

This is non-linear in the reprojected lines $\hat{\mathbf{l}}_{ij}$ and consequently in the entries of $\tilde{\mathsf{H}}$, which implies the use of non-linear optimization techniques. We use the Levenberg-Marquardt algorithm (Gill et al., 1981) with numerical differentiation. The unknowns are minimally parameterized (we optimize directly the entries of H, not $\tilde{\mathsf{H}}$), so no subsequent correction is needed to recover the motion parameters.

Quasi-Linear Estimation. The drawbacks of non-linear optimization are that the implementation may be complicated. For these reasons, we also developed a quasi-linear estimator "*Qlin_2D*" that minimizes the same cost function $\mathcal{C}_3$. Consider the cost functions $\mathcal{C}_2$ and $\mathcal{C}_3$. Both depend on the same data, measured image points on the line (such as end-points) and reprojected lines, the former using an algebraic and the latter the orthogonal distance. We can relate these distances by:

$$d_\perp^2(\mathbf{x}, \mathbf{l}) = w\, d_a^2(\mathbf{x}, \mathbf{l}) \quad \text{where } w = \frac{1}{l_1^2 + l_2^2}, \qquad (7)$$

and rewrite $\mathcal{C}_3$ as:

$$\mathcal{C}_3 = \sum_i \sum_j w_{ij} \left(d_a^2(\mathbf{x}_{ij}, \hat{\mathbf{l}}_{ij}) + d_a^2(\mathbf{y}_{ij}, \hat{\mathbf{l}}_{ij})\right). \qquad (8)$$

The non-linearity is hidden in the weight factors w_{ij}. If they were known, the cost function would be a sum of squares of terms that are linear in the entries of $\tilde{\mathsf{H}}$. This leads to the following iterative algorithm. Weights, assumed unknown, are initialized to unity and iteratively updated. The loop is ended when the weights or equivalently the residual errors converge. The algorithm is summarized in Table 5. It is a quasi-linear optimization that converges from the minimization of an algebraic error to the geometrical one. Its main advantages are that it is simple to implement (as a loop over a linear method) and that it gives reliable results as will be seen in the next sections.

Table 5. Quasi-linear motion estimation from 3D line correspondences.

1. *initialization*: set $w_{ij} = 1$;
2. *estimation*: estimate $\tilde{\mathsf{H}}$ using standard weighted least squares; the 6×6 matrix obtained is corrected so that it represents a motion, see algorithm 1;
3. *weighting*: use $\tilde{\mathsf{H}}$ to update the weights w_{ij} according to Eq. (7);
4. *iteration*: iterate steps 2 and 3 until convergence (see text).

Non-Linear Estimation 2. The cost function $\mathcal{C}_3$ employed in methods *NLin_2D_1* and *QLin_2D* is expressed only in terms of entities of the second set of images. Hence, it is not symmetric with respect to the two sets of images. Our sixth method "*NLin_2D_2*" is based on a cost function $\mathcal{C}_4$, similar to $\mathcal{C}_3$, but that is symmetric with respect to the two sets of images.

Let $\mathbf{x}'_{ij}$ and $\mathbf{y}'_{ij}$ designate the end-points of line i in the j-th image of the first set. In order to write the expression of $\mathcal{C}_4$, we need the lines $\hat{\mathbf{l}}'_{ij}$ projected in the first image set, after transfer from the second basis. They are given by $\hat{\mathbf{l}}'_{ij} \sim \tilde{\mathsf{P}}_j \tilde{\mathsf{H}}^{-1} \mathbf{L}_{2_i}$. The cost function $\mathcal{C}_4$ can then be expressed as:

$$\mathcal{C}_4 = \mathcal{C}_3 + \sum_i \sum_j \left(d_\perp^2(\mathbf{x}'_{ij}, \hat{\mathbf{l}}'_{ij}) + d_\perp^2(\mathbf{y}'_{ij}, \hat{\mathbf{l}}'_{ij}) \right). \quad (9)$$

This is a non-linear function. As for method *NLin_2D_1*, we optimize the entries of H using the Levenberg-Marquardt algorithm with numerical differentiation. At each optimization step, we form matrix $\tilde{\mathsf{H}}$. We do not compute directly the inverse of the 6×6 matrix $\tilde{\mathsf{H}}$ to get $\tilde{\mathsf{H}}^{-1}$, but we rather compute the 4×4 matrix H^{-1} and form the corresponding 3D line homography matrix, which is $\widetilde{\mathsf{H}^{-1}} = \mathsf{H}^{-1}$.

Other Cost Functions. Other distance measures between image lines have been proposed in the literature. In Taylor and Kriegman (1995), the authors proposed the total squared distance, i.e. the sum of squared orthogonal distances along two line segments. In Zhang (1994), the overlap between line segments is considered. These distances could be used for motion estimation within the above described framework, requiring non-linear optimization.

Singularities. It has been shown that there exist critical sets of 3D lines (Buchanan, 1992). In the case of motion estimation from 3D line correspondences, (Navab and Faugeras, 1997) shows that these sets may contain an infinite number of lines. For example, if all observed lines are coplanar, motion estimation is ambiguous. We did not encounter such a singular situation during our experiments.

6. Results Using Simulated Data

We first compare our estimators using simulated data. The test bench consists of four cameras that form two stereo pairs observing a set of $N = 50$ 3D lines randomly chosen in a sphere lying in the fields of view of all cameras. Lines are projected onto the image planes, end-points are hallucinated at the image boundaries and corrupted by additive Gaussian noise, and the equations of the image lines are estimated from these noisy end-points.

A canonical projective basis (Luong and Vieville, 1996) is attached to each camera pair and used to reconstruct the lines in projective space. We then compute the 3D homography between the two projective bases using the estimators given in Section 5. We assess the quality of an estimated motion by measuring the RMS (Root Mean Square) of the Euclidean reprojection error (orthogonal distances between reprojected lines and end-points in both image pairs). This corresponds to the symmetric cost function $\mathcal{C}_4$ (Eq. (9)) minimized by the non-linear algorithm *NLin_2D_2*. We also monitor the computational cost of each method. We show median results over 100 trials.

Accuracy. Figure 3 shows the error as the level of added noise varies. The non-linear method is initialized using the quasi-linear one (an initialization from *Lin_2D_2* gives similar results). We observe that the methods *Lin_3D* (based on an algebraic distance between 3D Plücker coordinates), *Lin_2D_1* and *Lin_2D_2* perform worse than the others. This is due to the fact that the cost functions used in these methods are not physically meaningful and biased compared to $\mathcal{C}_4$. Method *QLin_2D* gives results close to

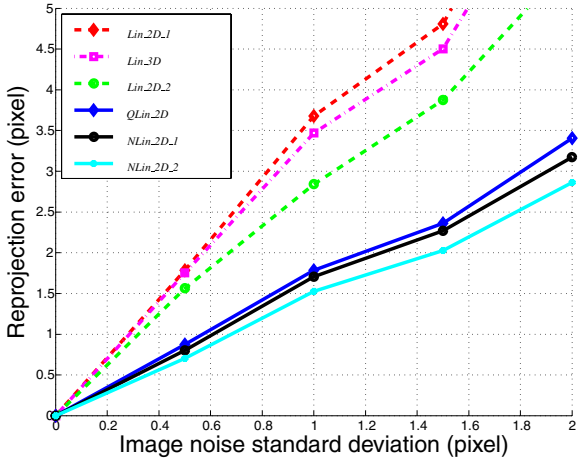


Figure 3. Comparison of the reprojection error versus added image noise for different motion estimators. The order of methods in the legend corresponds to the curves from top to bottom.

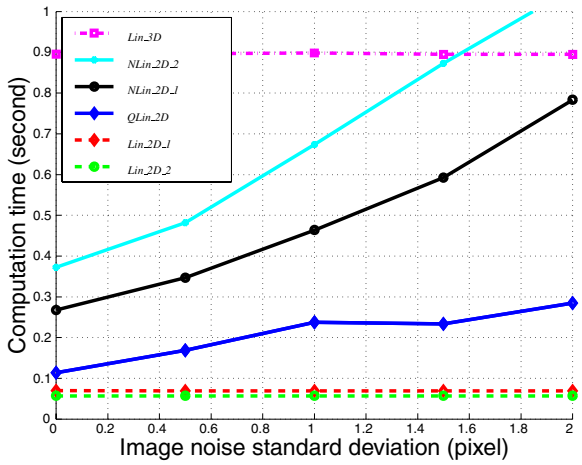


Figure 4. Comparison of the computation time versus added image noise for different motion estimators. The order of methods in the legend corresponds to the curves from top to bottom.

those obtained using *NLin_2D_1*. It is therefore a good compromise between the linear and non-linear methods, achieving good results while keeping simplicity of implementation. However, we observed that in a few cases (about 4%), the quasi-linear method does not enhance the result obtained by *Lin_2D_2* while *NLin_2D* does. *QLin_2D* estimates more parameters than necessary and this may cause numerical instabilities. The method that best minimizes the reprojection error is *NLin_2D_2*. This results could have been expected since this method consists in minimizing the reprojection error.

Computational Cost. Figure 4 shows the computational cost as the level of added noise varies. These results have been obtained using our C implementation, on a 850 Mhz. Pentium III PC running under Windows. As expected, the linear methods (*Lin_2D_1*, *Lin_2D_2* and *Lin_3D*) give constant results, i.e. that do not depend upon the level of added noise. Note that for these methods, the computational cost is dominated by the singular value decomposition needed to solve the linear system. Hence, the computational cost is directly linked to the size of the linear system associated to the method. This explains that method *Lin_3D*, with a $30N \times 36$ linear system to solve, has a much higher computational cost than methods *Lin_2D_1* and *Lin_2D_2* which have respectively $6N \times 36$ and $4N \times 36$ systems to solve.

On the other hand, non-linear methods (*QLin_2D*, *NLin_2D_1* and *NLin_2D_2*) give results depending on the added noise level. Indeed, the noise level influences the number of iterations needed by the method to convergence and hence, the computational cost. As for linear methods, the computational cost for method *QLin_2D* is dominated by the singular value decomposition of successive linear systems, i.e. step 2 of the algorithm shown in Table 5. Hence, the computational cost is roughly given by the number of iterations multiplied by the computational cost of method *Lin_2D_2*, on which *QLin_2D* is based. This corresponds to what can be observed on Fig. 4. For methods *NLin_2D_1* and *NLin_2D_2*, we observe that the computational cost of each Levenberg-Marquardt iteration is dominated by the computation of the differentiation of the cost function, and slightly influenced by the resolution of the normal equations. The fact that *NLin_2D_2* has roughly twice the computational cost of *NLin_2D_1* is explained by the fact that the number of terms in the symmetric cost function $\mathcal{C}_4$ is twice that of the cost function $\mathcal{C}_3$.

7. Results on Real Images

We test our algorithms using real images. Two scenarios are considered, described in the following two sections.

The first one considers two projective line reconstructions obtained from a weakly calibrated stereo rig.

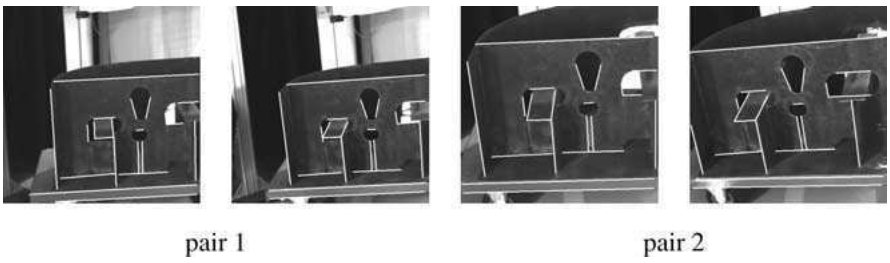


Figure 5. The two image pairs of a ship part used in the experiments, overlaid with extracted lines. Note that the extracted end-points do not necessarily correspond.

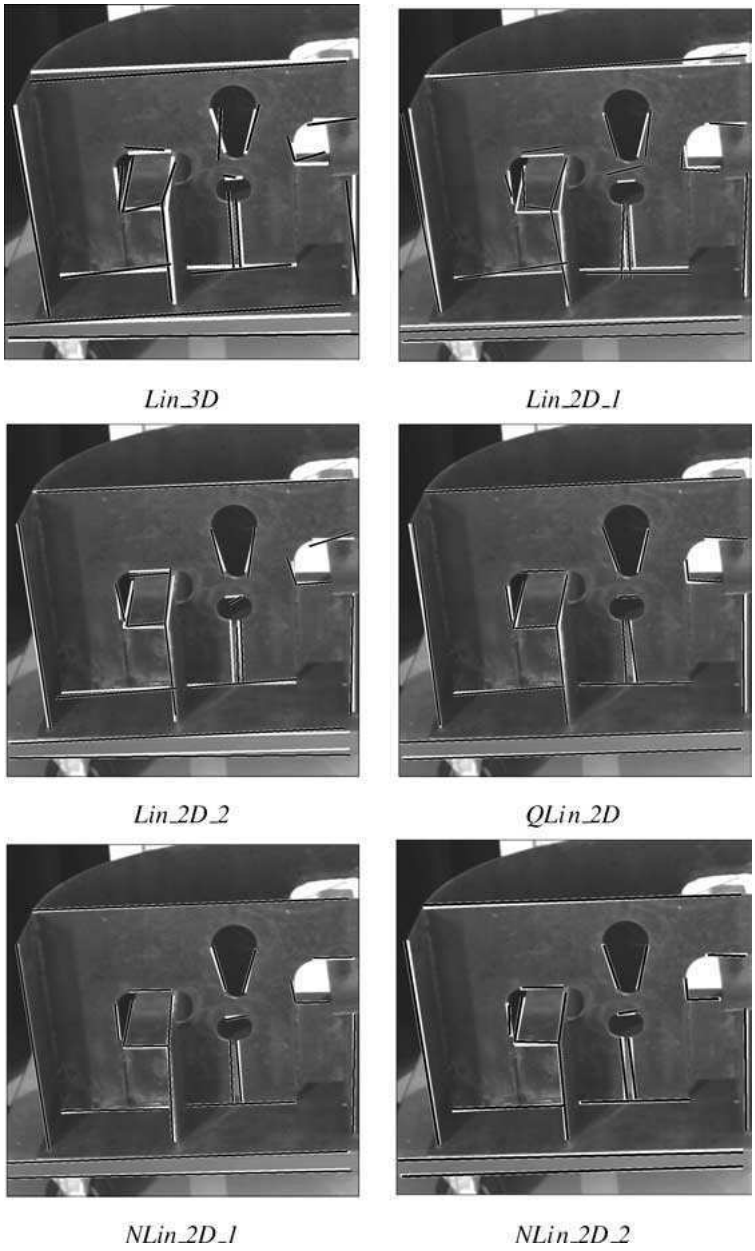


Figure 6. The lines transferred from the first reconstruction and reprojected onto the second image pair are shown overlaid on the left image of the second pair in black for different methods while observed image lines are shown in white.

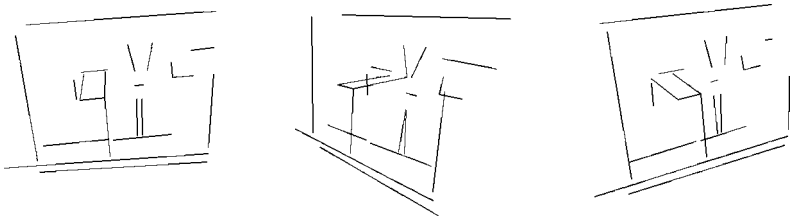


Figure 7. Reconstructed lines after self-calibration.

Figure 8. The first set of images, overlaid with extracted lines.

The overlap between the two reconstructions is large and the recovered motion is expected to be accurate. A stereo self-calibration technique is then applied to upgrade the reconstructions to metric space.

The second scenario is based on metric reconstructions obtained from multiple views of an indoor scene. The overlap between the two reconstructions is small. Hence, the alignment is expected to be unstable.

7.1. Largely Overlapping Reconstructions

We use images taken with a weakly calibrated stereo rig, shown on Fig. 5. Weakly calibrated means that the fundamental matrix between the left and right images is known. In practice, we estimate it from point correspondences using the maximum likelihood estimation technique given in Zhang (1998). The epipolar

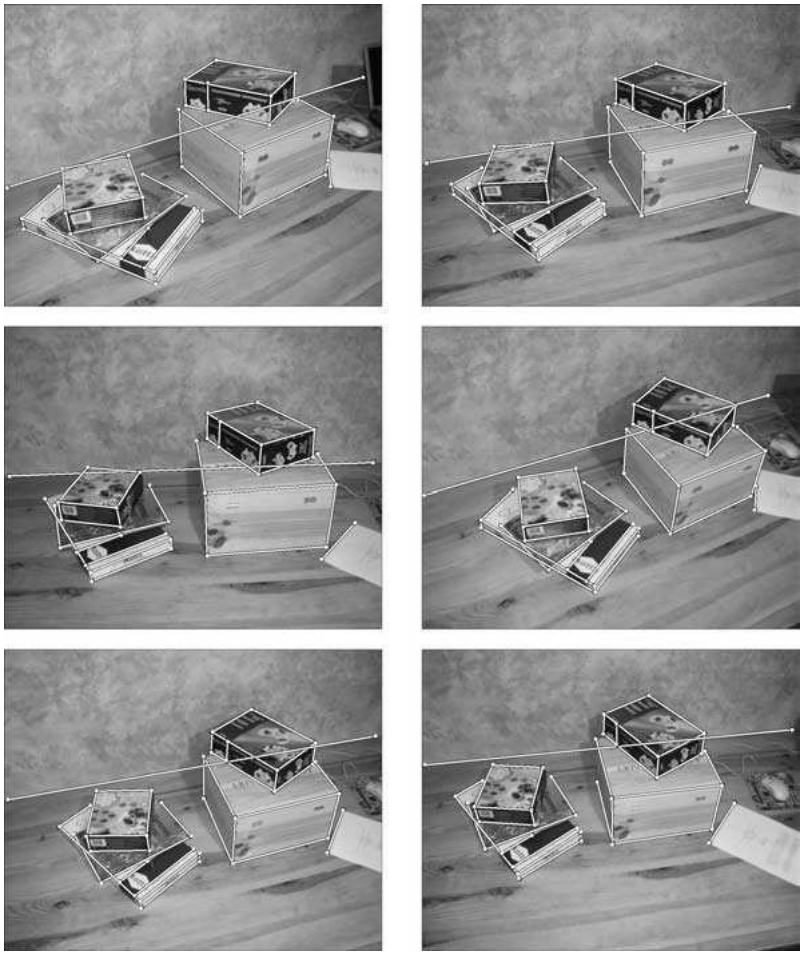


Figure 9. The second set of images, overlaid with extracted lines.

geometry is the same for both image pairs. Hence, stereo self-calibration techniques can be applied to recover camera calibration from the computed 3D motion. From the fundamental matrix, we define a canonical reconstruction basis for each pair (Luong and Vieville, 1996). This also gives the line projection matrices $\tilde{\mathsf{P}}_j$ and $\tilde{\mathsf{P}}'_j$. We track 21 lines across images by hand and projectively reconstruct them for each image pair. One can observe that all lines are visible in each of the 4 images of Fig. 5.

Motion Estimation. We use the methods of Section 5 to estimate the projective motion between the two reconstructions, but since we have no 3D ground truth we will only show the result of transferring the set of reconstructed lines from the first to the second 3D frame, using the 3D line homography matrix, and reprojecting them. Figure 6 shows these reprojections, which confirms that the non-linear and quasi-linear methods achieve better results than the linear ones. Note that the results appear visually slightly worse for method *NLin_2D_2* compared to method *NLin_2D_1* since the former minimizes the reprojection error in both image sets, while the latter uses only the second image set.

We measure the reprojection error (in both image sets) and computational cost for each method:

Method	Reprojection error (pixel)	Computation time (second)
Lin_3D	3.45	0.16
Lin_2D_1	3.36	0.03
Lin_2D_2	2.83	0.02
NLin_2D_1	2.05	0.15
QLin_2D	1.93	0.09
NLin_2D_2	1.53	0.26

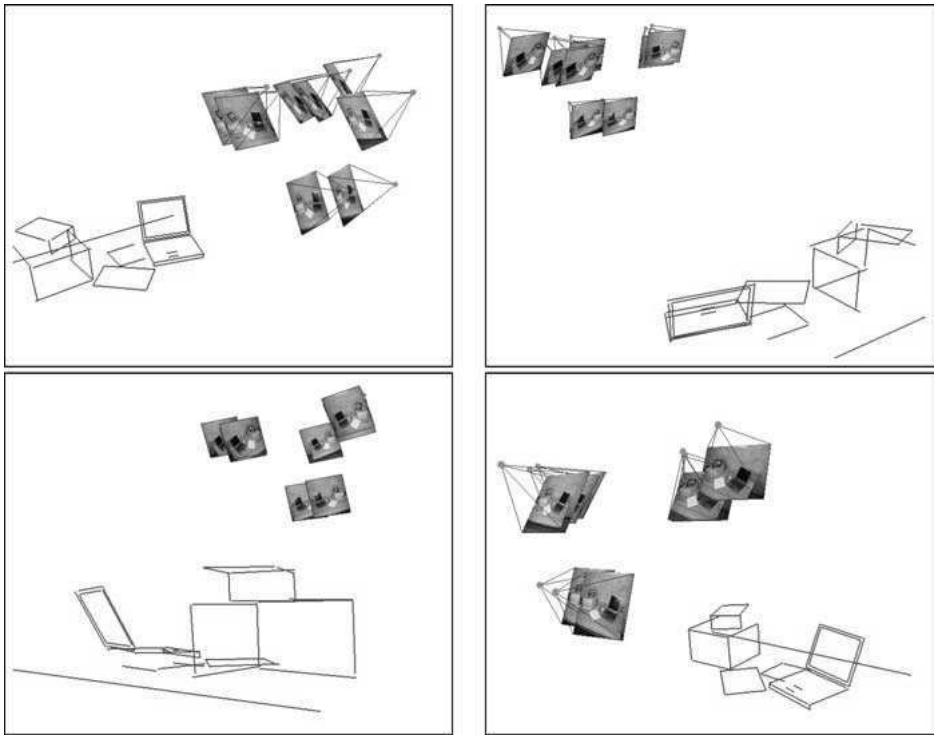


Figure 10. The 40 lines reconstructed from the first set of images.

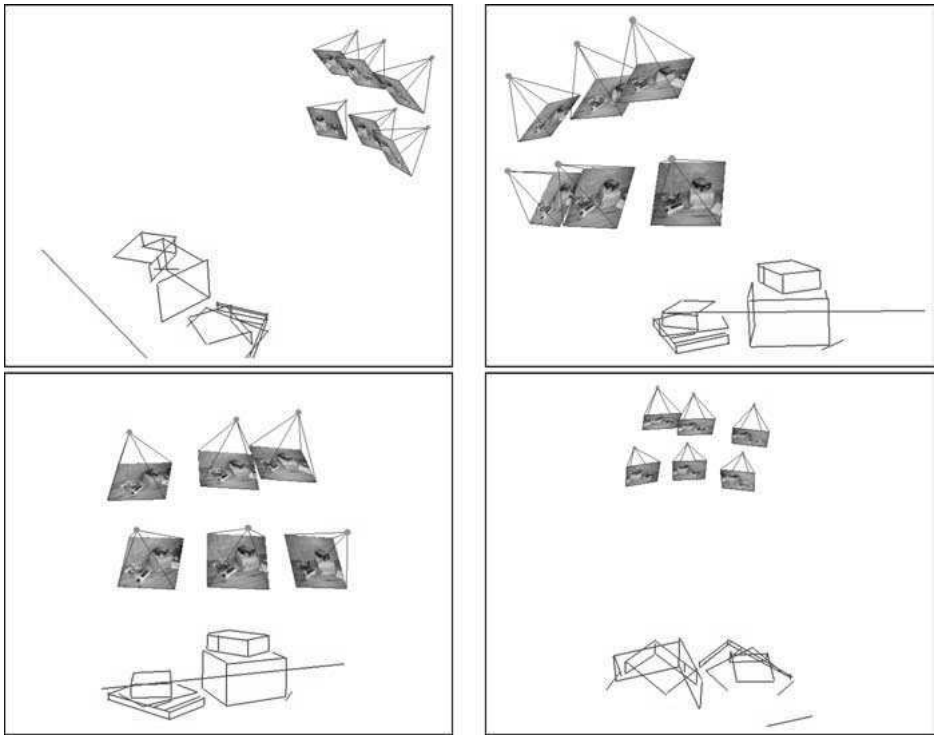


Figure 11. The 40 lines reconstructed from the second set of images.

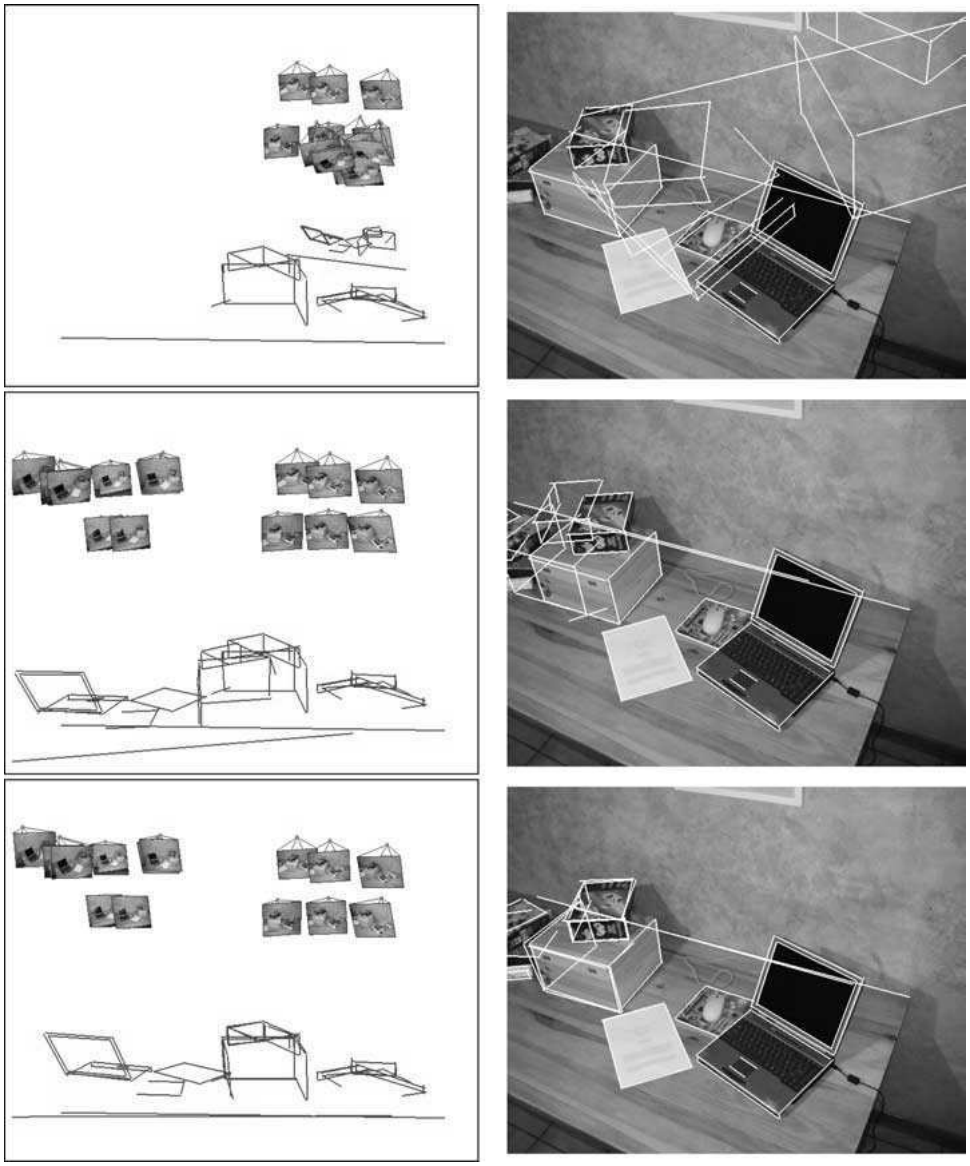


Figure 12. The two reconstructed sets of lines, from top to bottom: without alignment, alignment with linear methods and alignment with non-linear/quasi-linear methods. The left column shows views of the reconstructions, while the right column shows the reprojection in an original image.

These results confirm those observed on Fig. 6: non-linear and quasi-linear methods achieve better results than linear ones. The methods with the lowest computational costs are *Lin_2D_1* and *Lin_2D_2*, while the method with the highest computational cost is *NLin_2D_2*. Methods *Lin_3D* and *NLin_2D_1* have roughly the same computational cost.

Even if there are some differences between the methods, it can be said that all of them give a correct result, i.e. the reprojection error is reasonable.

Self-Calibration. We use the usual 4×4 homography matrix estimated with the method *NLin_2D_1* to self-calibrate the stereo rig using the method described in Horaud et al. (2000) with a three-parameter camera (zero skew and unit aspect ratio).

The usual 4×4 upgrade matrix, which converts a projective point reconstruction into a metric one, has the following form (Horaud et al., 2000), where K is the matrix of intrinsic parameters of the left

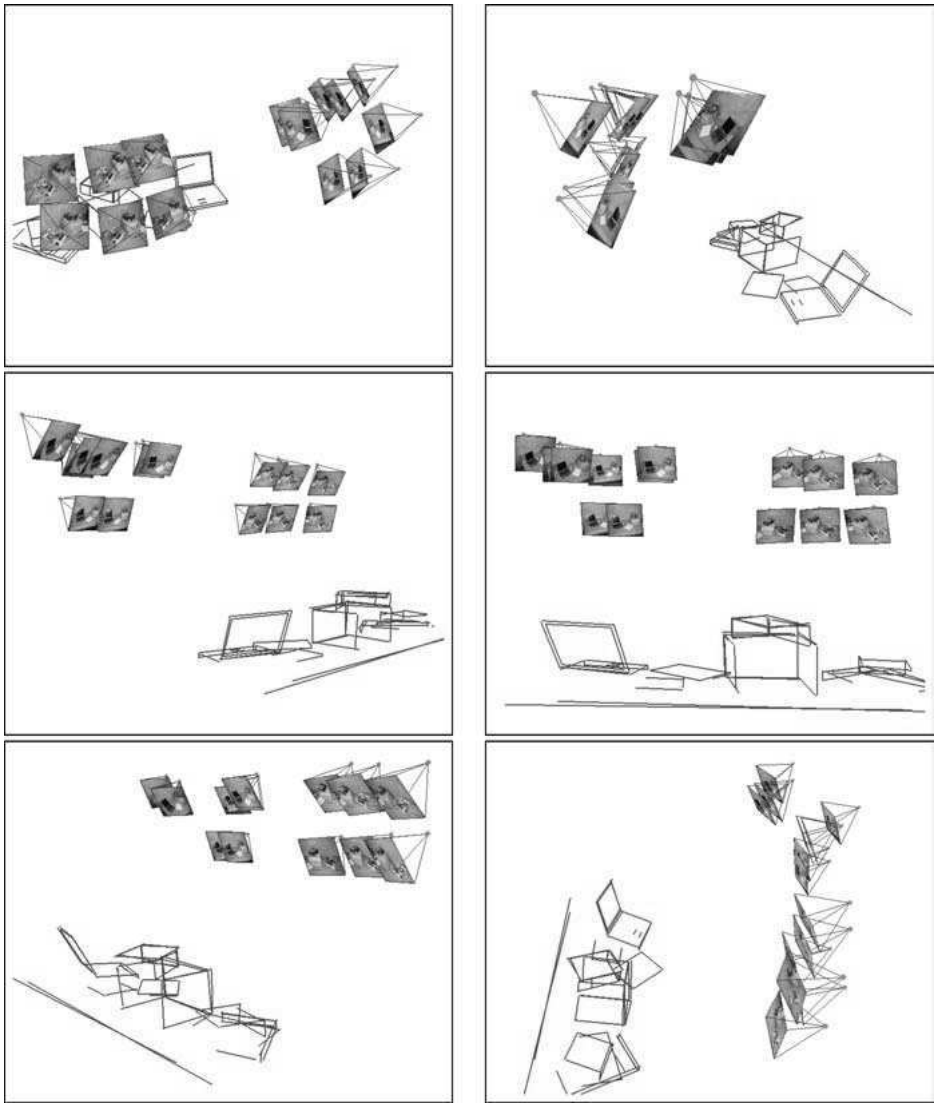


Figure 13. Some views of the reconstructions aligned with the non-linear method *NLin_2D_2*, consisting of 65 lines and 14 cameras.

camera, $\mathbf{a}^\mathsf{T} \sim (\bar{\mathbf{a}}^\mathsf{T}\ a)$ the coordinates of the plane at infinity in the reconstruction basis and f the focal length:

$$\mathsf{H}_u \sim \begin{pmatrix} \mathsf{K}^{-1} & \mathbf{0} \\ \bar{\mathbf{a}}^\mathsf{T} & a \end{pmatrix},$$

which gives the 6×6 *3D line upgrade matrix* as:

$$\tilde{\mathsf{H}}_u \sim \begin{pmatrix} \frac{1}{f^2}\mathsf{K}^\mathsf{T} & 0 \\ -\mathsf{K}^{-1}[\bar{\mathbf{a}}]_\times & a\mathsf{K}^{-1} \end{pmatrix}.$$

We compare the intrinsic parameters recovered for the left camera to those estimated using off-line calibration (Faugeras and Toscani, 1987):

Parameter	Self-calib.	Off-line calib.	% error
f	1514.22	1461.02	3.51
u_0	252.93	267.98	5.95
v_0	250.68	241.03	3.84

where (u_0, v_0) are the image coordinates of the principal point.

Figure 7 shows different points of view of the reconstruction we obtained from the first pair. We hallucinate 3D end-points by intersecting the viewing rays of image end-points with the reconstructed lines from the first pair (we use those from the left image of the first pair). Qualitatively, the result seems to be correct.

7.2. Slightly Overlapping Reconstructions

We use two sets of images, shown on Figs. 8 and 9, taken with a calibrated camera. For each image set, we use point correspondences to get camera positions and used them to reconstruct 3D lines, as shown on Figs.10 and 11.

The observed scene is composed of two stacks of boxes and a laptop. In the first image set, the leftmost stack of boxes is not visible, while in the second image set, the laptop is not visible. Hence, even if 40 lines are reconstructed from each image set, the overlap is constituted by 15 lines only, lying on the middle stack of boxes and closely located in space.

We apply our alignment algorithms to these data. The results are visible on Fig. 12. Visually, the results lie in two categories. The linear methods give very biased results, leading to bad alignment, while the non-linear methods (including the quasi-linear one) give reliable results.

We measure the reprojection error and computational cost for each method:

Method	Reprojection error (pixel)	Computation time (second)
Lin_3D	14.49	0.12
Lin_2D_1	15.10	0.02
Lin_2D_2	13.28	0.01
NLin_2D_1	2.95	0.17
QLin_2D	2.91	0.08
NLin_2D_2	1.76	0.31

These measurements confirm the previous observation: the reprojection error is of an order of 10 pixels for linear methods, which is large, while it is of an order of a few pixels for non-linear/quasi-linear methods. These results are explained by the fact that only a few line correspondences are available and that they are closely located in space.

8. Conclusions

We addressed the problem of estimating the motion between two line reconstructions in the general projective case. We used Plücker coordinates to represent 3D lines and showed that they could be transferred linearly between two reconstruction bases using a 6×6 3D line homography matrix. We specialized this result to the affine, metric and Euclidean cases. We investigated the algebraic properties of this matrix and showed how to extract the usual 4×4 motion matrices (i.e. homography, affinity similarity or rigid displacement) from them.

We then proposed several 3D and image-based estimators for the motion between two line reconstructions. Experimental results on both simulated and real data show that the linear estimators perform worse than the non-linear ones, especially when the cost function is expressed in 3D space. The non-linear and quasi-linear estimators, based on orthogonal image errors give similar good results. Concerning the computational cost, we show that linear methods based on 2D cost functions are not expensive, compared to non-linear methods, while the linear method based on a 3D cost function may be as expensive as a non-linear method.

More specifically, when the overlap between the two reconstructions is large, as it can be expected when a continuous image sequence is processed, the alignement obtained with linear methods is reliable. Hence these methods could be used for real-time stereo tracking of lines, in a manner similar to Demirdjian and Horaud (2000).

Appendix: Proofs and Derivations

In this appendix, we derive and prove some results mentioned in this paper.

Perspective Projection Matrix for Lines. Consider a line with Plücker coordinates $\mathbf{L}^\top \sim (\mathbf{a}^\top \ \mathbf{b}^\top)$ defined by two points $\mathbf{M}^\top \sim (\bar{\mathbf{M}}^\top \ m)$ and $\mathbf{N}^\top \sim (\bar{\mathbf{N}}^\top \ n)$ that are projected onto $\mathbf{m}$ and $\mathbf{n}$ respectively by the perspective projection matrix P. The corresponding image line is $\mathbf{l}$. Expanding its expression leads to the perspective projection matrix for lines, $\tilde{\mathsf{P}}$ as:

$$
\begin{aligned}
\mathbf{l} &\sim \mathbf{m} \times \mathbf{n} \\
&\sim (\mathsf{P}\mathbf{M}) \times (\mathsf{P}\mathbf{N}) \\
&\sim (\bar{\mathsf{P}}\bar{\mathbf{M}} + m\mathbf{p}) \times (\bar{\mathsf{P}}\bar{\mathbf{N}} + n\mathbf{p}) \\
&\sim (\bar{\mathsf{P}}\bar{\mathbf{M}}) \times (\bar{\mathsf{P}}\bar{\mathbf{N}}) + m\mathbf{p} \times (\bar{\mathsf{P}}\bar{\mathbf{N}}) - n\mathbf{p} \times (\bar{\mathsf{P}}\bar{\mathbf{M}})
\end{aligned}
$$

$$\sim \det(\bar{\mathsf{P}})\bar{\mathsf{P}}^{-\mathsf{T}}(\bar{\mathbf{M}} \times \bar{\mathbf{N}}) + [\mathbf{p}]_\times \bar{\mathsf{P}}(m\bar{\mathbf{N}} - n\bar{\mathbf{M}})$$
$$\sim \tilde{\mathsf{P}}\mathbf{L} \text{ where } \tilde{\mathsf{P}} \sim (\det(\bar{\mathsf{P}})\bar{\mathsf{P}}^{-\mathsf{T}}\ [\mathbf{p}]_\times \bar{\mathsf{P}}).$$

Deriving the 3D Line Homography Matrix. Consider a line with coordinates $\mathbf{L}_1^\mathsf{T} \sim (\mathbf{a}_1^\mathsf{T}\ \mathbf{b}_1^\mathsf{T})$ defined by two points $\mathbf{M}_1^\mathsf{T} \sim (\bar{\mathbf{M}}_1^\mathsf{T}\ m_1)$ and $\mathbf{N}_1^\mathsf{T} \sim (\bar{\mathbf{N}}_1^\mathsf{T}\ n_1)$ in the first projective basis and Plücker coordinates $\mathbf{L}_2^\mathsf{T} \sim (\mathbf{a}_2^\mathsf{T}\ \mathbf{b}_2^\mathsf{T})$ defined by points $\mathbf{M}_2^\mathsf{T} \sim (\bar{\mathbf{M}}_2^\mathsf{T}\ m_2)$ and $\mathbf{N}_2^\mathsf{T} \sim (\bar{\mathbf{N}}_2^\mathsf{T}\ n_2)$ in the second projective basis. Expanding the expressions for $\mathbf{a}_2$ and $\mathbf{b}_2$ according to the definition of Plücker coordinates (1) gives respectively the 3×6 upper and lower parts of $\tilde{\mathsf{H}}$:

$$\begin{aligned}
\mathbf{a}_2 &= \bar{\mathbf{M}}_2 \times \bar{\mathbf{N}}_2 \\
&= (\bar{\mathsf{H}}\bar{\mathbf{M}}_1 + m_1\mathbf{h}_1) \times (\bar{\mathsf{H}}\bar{\mathbf{N}}_1 + n_1\mathbf{h}_1) \\
&= \det(\bar{\mathsf{H}})\bar{\mathsf{H}}^{-\mathsf{T}}(\bar{\mathbf{M}}_1 \times \bar{\mathbf{N}}_1) + [\mathbf{h}_1]_\times \bar{\mathsf{H}}(m_1\bar{\mathbf{N}}_1 - n_1\bar{\mathbf{M}}_1) \\
&= \det(\bar{\mathsf{H}})\bar{\mathsf{H}}^{-\mathsf{T}}\mathbf{a}_1 + [\mathbf{h}_1]_\times \bar{\mathsf{H}}\mathbf{b}_1,
\end{aligned}$$

$$\begin{aligned}
\mathbf{b}_2 &= m_2\bar{\mathbf{N}}_2 - n_2\bar{\mathbf{M}}_2 \\
&= \mathbf{h}_2^\mathsf{T}(\bar{\mathbf{M}}_1\bar{\mathsf{H}}\bar{\mathbf{N}}_1 - \bar{\mathbf{N}}_1\bar{\mathsf{H}}\bar{\mathbf{M}}_1) + \mathbf{h}_2^\mathsf{T}(\bar{\mathbf{M}}_1\mathbf{h}_1 n_1 - \bar{\mathbf{N}}_1\mathbf{h}_1 m_1) \\
&\quad + h\bar{\mathsf{H}}(m_1\bar{\mathbf{N}}_1 - n_1\bar{\mathbf{M}}_1) \\
&= -\bar{\mathsf{H}}[\mathbf{h}_2]_\times \mathbf{a}_1 - \mathbf{h}_1\mathbf{h}_2^\mathsf{T}\mathbf{b}_1 + h\bar{\mathsf{H}}\mathbf{b}_1.
\end{aligned}$$

The specialization of this result to the affine, metric and Euclidean spaces shown in Sections 3.2, 3.3 and 3.4 respectively is straightforward.

Deriving the 20 Consistency Constraints on the 3D Line Homography Matrix. We prove 20 non-linear consistency constraints that must hold on the entries of a 3D line homography matrix. Note that there exist other possible constraints. We use the notation defined in Section 3.1.

Consider the product $\tilde{\mathsf{H}}_{11}\tilde{\mathsf{H}}_{12}^\mathsf{T}$. Its expansion leads to:

$$\begin{aligned}
\tilde{\mathsf{H}}_{11}\tilde{\mathsf{H}}_{12}^\mathsf{T} &\sim \bar{\mathsf{H}}^{-\mathsf{T}}\bar{\mathsf{H}}^\mathsf{T}[\mathbf{h}_1]_\times \\
&\sim [\mathbf{h}_1]_\times,
\end{aligned}$$

which is a skew-symmetric matrix. It means that its diagonal entries vanish, which corresponds to the following 3 constraints on the 3D line homography matrix:

$$\left(\mathbf{r}_{11,k}^\mathsf{T}\right)(\mathbf{r}_{12,k}) = 0, \quad k = 1 \ldots 3.$$

Note that 3 other constraints could be derived from this equation based on the off-diagonal entries.

Similarly, consider the product $\tilde{\mathsf{H}}_{21}\tilde{\mathsf{H}}_{22}^\mathsf{T}$. Its expansion leads to:

$$\begin{aligned}
\tilde{\mathsf{H}}_{21}\tilde{\mathsf{H}}_{22}^\mathsf{T} &\sim \bar{\mathsf{H}}[\mathbf{h}_2]_\times\left(h\bar{\mathsf{H}}^\mathsf{T} - \mathbf{h}_2\mathbf{h}_1^\mathsf{T}\right) \\
&\sim \bar{\mathsf{H}}[\mathbf{h}_2]_\times h\bar{\mathsf{H}}^\mathsf{T} - \bar{\mathsf{H}}[\mathbf{h}_2]_\times\mathbf{h}_2\mathbf{h}_1^\mathsf{T} \\
&\sim \bar{\mathsf{H}}[\mathbf{h}_2]_\times\bar{\mathsf{H}}^\mathsf{T} \\
&\sim [\bar{\mathsf{H}}^{-\mathsf{T}}\mathbf{h}_2]_\times,
\end{aligned}$$

where we used the rule:

$$[\mathsf{J}\mathbf{g}]_\times = \det(\mathsf{J})\mathsf{J}^{-\mathsf{T}}[\mathbf{g}]_\times\mathsf{J}^{-1} \sim \mathsf{J}^{-\mathsf{T}}[\mathbf{g}]_\times\mathsf{J}^{-1}. \quad (10)$$

As previously, we end up with a skew-symmetric matrix whose diagonal entries vanish, giving another 3 constraints:

$$\left(\mathbf{r}_{21,k}^\mathsf{T}\right)(\mathbf{r}_{22,k}) = 0, \quad k = 1 \ldots 3.$$

Similarly, observe that:

$$\begin{aligned}
\tilde{\mathsf{H}}_{11}^\mathsf{T}\tilde{\mathsf{H}}_{21} &\sim \bar{\mathsf{H}}^{-1}\bar{\mathsf{H}}[\mathbf{h}_2]_\times \\
&\sim [\mathbf{h}_2]_\times,
\end{aligned}$$

and that:

$$\begin{aligned}
\tilde{\mathsf{H}}_{12}^\mathsf{T}\tilde{\mathsf{H}}_{22} &\sim \bar{\mathsf{H}}^\mathsf{T}[\mathbf{h}_1]_\times\left(h\bar{\mathsf{H}} - \mathbf{h}_1\mathbf{h}_2^\mathsf{T}\right) \\
&\sim \bar{\mathsf{H}}^\mathsf{T}[\mathbf{h}_1]_\times\bar{\mathsf{H}} \\
&\sim [\bar{\mathsf{H}}^{-1}\mathbf{h}_1]_\times,
\end{aligned}$$

which gives, respectively, the following 6 constraints:

$$\begin{aligned}
\left(\mathbf{c}_{11,l}^\mathsf{T}\right)(\mathbf{c}_{21,l}) &= 0, \quad l = 1 \ldots 3 \\
\left(\mathbf{c}_{12,l}^\mathsf{T}\right)(\mathbf{c}_{22,l}) &= 0, \quad l = 1 \ldots 3.
\end{aligned}$$

The first 12 constraints are derived. We derive the remaining 8 contraints as follows. Consider the following equation:

$$\begin{aligned}
\tilde{\mathsf{H}}_{11}\tilde{\mathsf{H}}_{22}^\mathsf{T} + \tilde{\mathsf{H}}_{12}\tilde{\mathsf{H}}_{21}^\mathsf{T} &\sim \det(\bar{\mathsf{H}})\bar{\mathsf{H}}^{-\mathsf{T}}\left(h\bar{\mathsf{H}}^\mathsf{T} - \mathbf{h}_2\mathbf{h}_1^\mathsf{T}\right) \\
&\quad + [\mathbf{h}_1]_\times\bar{\mathsf{H}}[\mathbf{h}_2]_\times\bar{\mathsf{H}}^\mathsf{T}.
\end{aligned}$$

After expansion and by using Eq. (10) for the last term, we obtain:

$$\begin{aligned}
\tilde{\mathsf{H}}_{11}\tilde{\mathsf{H}}_{22}^\mathsf{T} + \tilde{\mathsf{H}}_{12}\tilde{\mathsf{H}}_{21}^\mathsf{T} &\sim h\mathrm{I} - \det(\bar{\mathsf{H}})\bar{\mathsf{H}}^{-\mathsf{T}}\mathbf{h}_2\mathbf{h}_1^\mathsf{T} \\
&\quad + [\mathbf{h}_1]_\times \det(\bar{\mathsf{H}})[\bar{\mathsf{H}}^{-\mathsf{T}}\mathbf{h}_2]_\times \\
&\sim h\mathrm{I} - \bar{\mathsf{H}}^{-\mathsf{T}}\mathbf{h}_2\mathbf{h}_1^\mathsf{T} - [\mathbf{h}_1]_\times[\bar{\mathsf{H}}^{-\mathsf{T}}\mathbf{h}_2]_\times^\mathsf{T}.
\end{aligned}$$

Define $\mathbf{a} = \bar{\mathsf{H}}^{-\mathsf{T}}\mathbf{h}_2$ and $\mathbf{b} = \mathbf{h}_1$ and use the following rule:

$$\mathbf{a}\mathbf{b}^{\mathsf{T}} + [\mathbf{b}]_\times [\mathbf{a}]_\times^{\mathsf{T}} = (\mathbf{a}^{\mathsf{T}}\mathbf{b})\mathrm{I},$$

which gives:

$$\begin{aligned}\tilde{\mathsf{H}}_{11}\tilde{\mathsf{H}}_{22}^{\mathsf{T}} + \tilde{\mathsf{H}}_{12}\tilde{\mathsf{H}}_{21}^{\mathsf{T}} &\sim h\mathrm{I} - \big(\mathbf{h}_2^{\mathsf{T}}\bar{\mathsf{H}}^{-1}\mathbf{h}_1\big)\mathrm{I}\\ &\sim \mathrm{I}.\end{aligned}$$

From this equation, we deduce that all diagonal entries of matrix $\tilde{\mathsf{H}}_{11}\tilde{\mathsf{H}}_{22}^{\mathsf{T}} + \tilde{\mathsf{H}}_{12}\tilde{\mathsf{H}}_{21}^{\mathsf{T}}$ are equal and all off-diagonal entries are zero, which gives the remaining 8 constraints:

$$\begin{aligned}&\big(\mathbf{r}_{11,k}^{\mathsf{T}}\big)(\mathbf{r}_{22,k'}) + \big(\mathbf{r}_{12,k}^{\mathsf{T}}\big)(\mathbf{r}_{21,k'})\\ &\quad = 0,\ k \in 1\ldots 3,\ k' \in 1\ldots 3,\ k \neq k'\\ &\big(\mathbf{r}_{11,1}^{\mathsf{T}}\big)(\mathbf{r}_{22,1}) + \big(\mathbf{r}_{12,1}^{\mathsf{T}}\big)(\mathbf{r}_{21,1})\\ &\quad = \big(\mathbf{r}_{11,2}^{\mathsf{T}}\big)(\mathbf{r}_{22,2}) + \big(\mathbf{r}_{12,2}^{\mathsf{T}}\big)(\mathbf{r}_{21,2})\\ &\quad = \big(\mathbf{r}_{11,3}^{\mathsf{T}}\big)(\mathbf{r}_{22,3}) + \big(\mathbf{r}_{12,3}^{\mathsf{T}}\big)(\mathbf{r}_{21,3}).\end{aligned}$$

Extracting the Usual Homography Matrix From the 3D Line Homography Matrix. To prove the correctness of algorithm 1 we may reform the 3D line homography matrix $\tilde{\mathsf{H}}'$ corresponding to the extracted usual motion parameters (given by Eq. (3)) and verify that each of its blocks corresponds to the original block of $\tilde{\mathsf{H}}$, as follows:

$$\begin{aligned}\tilde{\mathsf{H}}'_{11} &= \det(\bar{\mathsf{H}})\bar{\mathsf{H}}^{-\mathsf{T}}\\ &= \det\Big(\pm\sqrt{|\det(\tilde{\mathsf{H}}_{11})|}\tilde{\mathsf{H}}_{11}^{-\mathsf{T}}\Big)\frac{\pm 1}{\sqrt{|\det(\tilde{\mathsf{H}}_{11})|}}\tilde{\mathsf{H}}_{11}\\ &= \pm\sqrt{|\det(\tilde{\mathsf{H}}_{11})|}^{3}\ \frac{1}{\det(\tilde{\mathsf{H}}_{11})}\frac{\pm 1}{\sqrt{|\det(\tilde{\mathsf{H}}_{11})|}}\tilde{\mathsf{H}}_{11}\\ &= \pm\tilde{\mathsf{H}}_{11}\\ \tilde{\mathsf{H}}'_{12} &= [\mathbf{h}_1]_\times\bar{\mathsf{H}}\\ &= \pm\tilde{\mathsf{H}}_{12}(\pm\bar{\mathsf{H}}^{-1})\bar{\mathsf{H}}\\ &= \pm\tilde{\mathsf{H}}_{12}\\ \tilde{\mathsf{H}}'_{21} &= -\bar{\mathsf{H}}[\mathbf{h}_2]_\times\\ &= -\mp\bar{\mathsf{H}}(\pm\bar{\mathsf{H}}^{-1})\tilde{\mathsf{H}}_{21}\\ &= \pm\tilde{\mathsf{H}}_{21}\\ \tilde{\mathsf{H}}'_{22} &= h\bar{\mathsf{H}} - \mathbf{h}_1\mathbf{h}_2^{\mathsf{T}}\\ &= \pm\big(\tilde{\mathsf{H}}_{22} + (\pm\mathbf{h}_1)\big(\pm\mathbf{h}_2^{\mathsf{T}}\big)\big)(\pm\bar{\mathsf{H}}^{-1})\bar{\mathsf{H}} - \mathbf{h}_1\mathbf{h}_2^{\mathsf{T}}\\ &= \pm\tilde{\mathsf{H}}_{22}.\end{aligned}$$

Note

1. Compare this with the case of fundamental matrix estimation using the 8 point algorithm: the obtained 3×3 matrix is corrected so that its smallest singular value becomes zero (Hartley and Zisserman, 2000).

References

Andreff, N., Espiau, B., and Horaud, R. 2000. Visual servoing from lines. In *International Conference on Robotics and Automation*, San Francisco.

Bartoli, A. and Sturm, P. 2001. The 3D line motion matrix and alignment of line reconstructions. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, Kauai, Hawaii, USA, IEEE Computer Society Press, vol. I, pp. 287–292.

Buchanan, T. 1992. Critical sets for 3D reconstruction using lines. In *Proceedings of the 2nd European Conference on Computer Vision*, G. Sandini (Ed.), Springer-Verlag, Santa Margherita Ligure, Italy, pp. 730–738.

Canny J. 1986. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698.

Csurka, G., Demirdjian, D., and Horaud, R. 1999. Finding the collineation between two projective reconstructions. *Computer Vision and Image Understanding*, 75(3):260–268.

Demirdjian, D. and Horaud, R. 2000. Motion-egomotion discrimination and motion segmentation from image-pair streams. *Computer Vision and Image Understanding*, 78(1):53–68.

Devernay, F. and Faugeras, O. 1996. From projective to euclidean reconstruction. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, San Francisco, California, USA, pp. 264–269.

Faugeras, O. and Mourrain, B. 1995. On the geometry and algebra of the point and line correspondences between *n* images. In *Proceedings of the 5th International Conference on Computer Vision*, Cambridge, Massachusetts, USA, pp. 951–956.

Faugeras, O.D. and Toscani, G. 1987. Camera calibration for 3D computer vision. In *Proceedings of International Workshop on Machine Vision and Machine Intelligence*, Tokyo, Japan.

Gill, P.E., Murray, W., and Wright, M.H. 1981. *Practical Optimization*. Academic Press.

Hager, G. and Toyama, K. 1998. X vision: A portable substrate for real-time vision applications. *CVIU*, 69(1):23–37.

Hartley, R.I. 1997. Lines and points in three views and the trifocal tensor. *International Journal of Computer Vision*, 22(2):125–140.

Hartley, R.I. and Zisserman, A. 2000. *Multiple View Geometry in Computer Vision*. Cambridge University Press.

Horaud, R., Csurka, G., and Demirdjian, D. 2000. Stereo calibration from rigid motions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1446–1452.

Horaud, R., Dornaika, F., and Espiau, B. 1997. Visually guided object grasping. *IEEE Transactions on Robotics and Automation*.

Horn, B.K.P., Hilden, H.M., and Negahdaripour, S. 1988. Closed-form solution of absolute orientation using orthonormal matrices. *Journal of the Optical Society of America A*, 5(7):1127–1135.

Liebowitz, D. and Zisserman, A. 1998. Metric rectification for perspective images of planes. In *Proceedings of the Conference*

on Computer Vision and Pattern Recognition, Santa Barbara, California, USA, pp. 482–488.

Liu, Y., Huang, T.S., and Faugeras, O.D. 1990. Determination of camera location from 2D to 3D line and point correspondences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1):28–37.

Luong, Q.T. and Faugeras, O. 1996. The fundamental matrix: Theory, algorithms and stability analysis. *International Journal of Computer Vision*, 17(1):43–76.

Luong, Q.T. and Vieville, T. 1996. Canonic representations for the geometries of multiple projective views. *Computer Vision and Image Understanding*, 64(2):193–229.

Navab, N. and Faugeras, O.D. 1997. The critical sets of lines for camera displacement estimation: A mixed euclidean-projective and constructive approach. *International Journal of Computer Vision*, 23(1):17–44.

Schmid, C. and Zisserman, A. 1997. Automatic line matching across views. In *Proceedings of the Conference on Computer* Vision and Pattern Recognition, Puerto Rico, USA, pp. 666–671.

Spetsakis, M. and Aloimonos, J. 1990. Structure from motion using line correspondences. *International Journal of Computer Vision*, 4:171–183.

Taylor, C.J. and Kriegman, D.J. 1995. Structure and motion from line segments in multiple images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(11):1021–1032.

Zhang, Z. 1994. Estimating motion and structure from correspondences of line segments between two perspective images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(12):1129–1139.

Zhang, Z. 1998. Determining the epipolar geometry and its uncertainty: A review. *International Journal of Computer Vision*, 27(2):161–195.

Zhang, Z. and Faugeras, O.D.1990. Tracking and grouping 3D line segments. In *Proceedings of the 3rd International Conference on Computer Vision*, Osaka, Japan, pp. 577–580.

Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computer Vision and Image Understanding
ELSEVIER
Computer Vision and Image Understanding 100 (2005) 416–441
www.elsevier.com/locate/cviu

Structure-from-motion using lines: Representation, triangulation, and bundle adjustment

Adrien Bartoli [a,*], Peter Sturm [b]

[a] *LASMEA, 24, Avenue des Landais, 63177 Aubière cedex, France*
[b] *INRIA, 655, Avenue de l'Europe, 38334 St Ismier cedex, France*

Received 12 March 2004; accepted 17 June 2005
Available online 11 August 2005

Abstract

We address the problem of camera motion and 3D structure reconstruction from line correspondences across multiple views, from initialization to final bundle adjustment. One of the main difficulties when dealing with line features is their algebraic representation. First, we consider the triangulation problem. Based on Plücker coordinates to represent the 3D lines, we propose a maximum likelihood algorithm, relying on linearizing the Plücker constraint and on a Plücker correction procedure, computing the closest Plücker coordinates to a given 6-vector. Second, we consider the bundle adjustment problem, which is essentially a nonlinear optimization process on camera motion and 3D line parameters. Previous overparameterizations of 3D lines induce gauge freedoms and/or internal consistency constraints. We propose the orthonormal representation, which allows handy nonlinear optimization of 3D lines using the minimum four parameters with an unconstrained optimization engine. We compare our algorithms to existing ones on simulated and real data. Results show that our triangulation algorithm outperforms standard linear and bias-corrected quasi-linear algorithms, and that bundle adjustment using our orthonormal representation yields results similar to the standard maximum likelihood trifocal tensor algorithm, while being usable for any number of views.

* Corresponding author. Fax: +33 473 407 262.
E-mail addresses: Adrien.Bartoli@gmail.com (A. Bartoli), Peter.Sturm@inria.fr (P. Sturm).

1077-3142/$ - see front matter © 2005 Elsevier Inc. All rights reserved.
doi:10.1016/j.cviu.2005.06.001

A. Bartoli, P. Sturm / Computer Vision and Image Understanding 100 (2005) 416–441 417

Keywords: Structure-from-motion; Lines; Representation; Triangulation; Bundle adjustment

1. Introduction

The goal of this paper is to give methods for reconstruction of line features from image correspondences over multiple views, from initialization to final bundle adjustment. Reconstruction of line features is an important topic since it is used in areas such as scene modeling, augmented reality, and visual servoing. Bundle adjustment is the computation of an optimal visual reconstruction of camera motion and 3D scene structure, where optimal means maximum likelihood in terms of reprojected image error. We make no assumption about the calibration of the cameras. We assume that line correspondences over at least three views are available.[1]

While the multiple-view geometry of lines is well-understood, see, e.g. [5,11], there is still a need for practical structure and motion algorithms. The factorization algorithms [15,18,25] yield reliable results but requires all lines to be visible in all views. We focus on the common three-stage approach, see e.g. [11, Section 17.5] consisting in (i) computing camera motion using inter-image matching tensors, (ii) triangulating the features, and (iii) running bundle adjustment.

There exist reliable algorithms for step (i). In particular, it can be solved by computing trifocal tensors for triplets of consecutive images, using, e.g., the automatic computation algorithm described in [11, Section 15.6], and registering the triplets in a manner similar to [6]. Other integrated motion estimation systems are [20], based on Kalman filtering techniques and [26], registering each view in turn.

In steps (ii) and (iii), one of the main difficulties when dealing with line features arises: the algebraic representation. Indeed, there is no minimal, complete and globally nonsingular parameterization of the four-dimensional set of 3D lines, see, e.g. [11, Section 2.2]. Hence, they are often overparameterized, e.g., as the join of two points or as the meet of two planes (eight parameters), or by the six coefficients of their Plücker coordinates, which must satisfy the bilinear Plücker constraint. Another overparameterization is two images of the line (six parameters). The most appropriate representation depends upon the problem considered. For example, the algorithm in [11, Section 15.2] shows that the 'two image lines' representation is well-adapted to the computation of the trifocal tensor, while the sequential algorithm of [20] is based on Plücker coordinates.

Concerning step (ii), many of the previous works assume calibrated cameras, e.g. [14,21,23,27] and use specific Euclidean representations. The linear three view algorithm of [27] and the algorithm of [23] utilize a 'closest point + direction' representation, while [21] uses the projections of the line on the $x = 0$ and the $y = 0$ planes, which has obvious singularities. These algorithms yield sub-optimal results in that none of them maximizes the individual likelihood of the reconstructed lines.

Bundle adjustment, step (iii), is a nonlinear procedure involving camera and 3D line parameters, attempting to maximize the likelihood of the reconstruction, corresponding to minimizing the reprojection error when the noise on measured features has an

[1] Line correspondences over two views do not constrain the camera motion.

identical and independent (i.i.d.) normal distribution. Previously mentioned overparameterizations are not well-adapted to standard nonlinear optimization engines. The 'two point' and the 'two plane' overparameterizations have four degrees of internal gauge freedoms[2] which may induce numerical instabilities. The 'two image lines' parameterization has two degrees of internal gauge freedoms and implies that one may have to choose different images for different lines since all lines may not be visible in all images. Also, one must check that the chosen images are not too close to each other. Finally, direct optimization of Plücker coordinates makes sense only if a constrained optimization technique is used to enforce the bilinear Plücker constraint. An appropriate representation would not involve internal constraint or gauge freedom.

To summarize, there is a need for an efficient optimal triangulation algorithm, and a representation of 3D lines well-adapted to nonlinear optimization. We address both of these problems through the following contributions.

In Section 3, we give an overview of various 3D line representations an their characteristics.

In Section 4, we propose triangulation methods based on using Plücker coordinates to represent the lines. A simple and optimal algorithm is obtained based on linearizing the bilinear Plücker constraint within an iteratively reweighted least squares approach.

In Section 5, we propose a nonlinear representation of 3D lines that we call the *orthonormal representation*. This representation allows efficient nonlinear optimization since only the minimum four parameters are computed at each step which allows the use of a standard unconstrained optimization engine. With this representation, there is no internal gauge freedom or consistency constraint, and analytic differentiation of the error function is possible.

Finally, Section 6 validates our algorithms and compares them to existing ones. The next section gives some preliminaries and notations and states the problem.

2. Preliminaries and notation

2.1. Notation

We make no formal distinction between coordinate vectors and physical entities. Everything is represented in homogeneous coordinates. Equality up to scale is denoted by $\sim$, transposition and transposed inverse by $^\top$ and $^{-\top}$. Vectors are typeset using bold fonts ($\mathbf{L}$, $\mathbf{l}$), matrices using sans-serif fonts (S, A, R), and scalars in italics. Bars represent inhomogeneous leading parts of vectors or matrices, e.g. $\mathbf{M}^\top \sim (\bar{\mathbf{M}}^\top | m)$. The $\mathscr{L}_2$-norm of vector v is denoted $\|\mathrm{v}\|$. The identity matrix is denoted I. $SO(2)$ and $SO(3)$ denote the 2D and 3D rotation groups.

The 2D orthogonal (Euclidean) distance between point $\mathbf{q}$ and line $\mathbf{l}$ weighted by q_3 is:

$$d_\perp^2(\mathbf{q},\mathbf{l}) = (\mathbf{q}^\top \mathbf{l})^2/(l_1^2 + l_2^2). \tag{1}$$

[2] For the former one, the position of the points along the line, and the free scale factor of the homogeneous representation of these points.

2.2. Matrix factorization

We make use of the singular value decomposition of matrices, dubbed SVD. The SVD of matrix A is $\mathsf{A}_{m\times n} = \mathsf{U}_{m\times n}\Sigma_{n\times n}\mathsf{V}^{\top}_{n\times n}$, where U and V are orthonormal, and Σ is diagonal, containing the singular values of A in decreasing order. The QR factorization of matrix A is $\mathsf{A}_{m\times n} = \mathsf{Q}_{m\times m}\,\mathsf{R}_{m\times n}$, with Q orthonormal and R upper triangular. More details on these matrix factorizations can be read in, e.g. [7].

2.3. Maximum likelihood estimation

As noted in [11, Section 15.7.2], no matter how many points are used to represent an image line $\mathbf{l}$, the quadratic error function on it can be expressed in the form $d^2_\perp(\mathbf{x},\mathbf{l}) + d^2_\perp(\mathbf{y},\mathbf{l})$ for two weighted points $\mathbf{x}$, $\mathbf{y}$ on $\mathbf{l}$. We will use this representation for simplicity. If we have 3D lines $\mathscr{S} = \{\mathbf{L}^1,\ldots,\mathbf{L}^m\}$ and cameras $\mathscr{M} = \{\mathsf{P}^1,\ldots,\mathsf{P}^n\}$, the negative log likelihood function $\mathscr{E}(\mathscr{S},\mathscr{M})$ for the reconstruction, corresponding to the reprojection error, can be written in terms of individual reprojection errors $\mathscr{E}(\mathbf{L}^j,\mathscr{M})$ for each line j:

$$\mathscr{E}(\mathscr{S},\mathscr{M}) = \sum_{j=1}^{m} \mathscr{E}(\mathbf{L}^j,\mathscr{M}), \tag{2}$$

$$\mathscr{E}(\mathbf{L}^j,\mathscr{M}) = \sum_{i=1}^{n} (d^2_\perp(\mathbf{x}^{ij},\mathbf{l}^{ij}) + d^2_\perp(\mathbf{y}^{ij},\mathbf{l}^{ij})). \tag{3}$$

3. Representing 3D lines

We describe several representations for 3D lines in projective space and their characteristics. Some of these representations are 'partial' in the sense that they can only represent a subset of all 3D lines. For example, some work on metric reconstruction, particularly in photogrammetry, assume that the reconstructed lines do not lie at infinity. The goal of this study is to choose a representation for the triangulation and bundle adjustment problems. Concerning the triangulation, the most important criterion is that the reprojected lines is a linear function of the 3D line. Bundle adjustment is a nonlinear procedure allowing more flexibility in the choice of the parameterization. The quality of the parameterization is assessed based on criteria such as the number of internal gauge freedoms or internal constraints. A summary of the reviewed representations is finally provided. The first representation that we describe is the Plücker coordinates. We link all the other representations to Plücker coordinates.

3.1. Complete representations

3.1.1. Plücker coordinates

Given two 3D points $\mathbf{M}^\top \sim (\bar{\mathbf{M}}^\top | m)$ and $\mathbf{N}^\top \sim (\bar{\mathbf{N}}^\top | n)$, one can represent the line joining them by a homogeneous 'Plücker' 6-vector $\mathbf{L}^\top \sim (\mathbf{a}^\top \mid \mathbf{b}^\top)$, see e.g. [11, Section 2.2]:

$$\begin{cases} \mathbf{a} &= \bar{\mathbf{M}} \times \bar{\mathbf{N}}, \\ \mathbf{b} &= m\bar{\mathbf{N}} - n\bar{\mathbf{M}}. \end{cases} \tag{4}$$

Other conventions for Plücker 6-vectors are also possible. Each comes with a bilinear constraint that the 6-vector must satisfy to represent valid line coordinates. For our definition, the constraint is

$$\mathscr{C}(\mathbf{L}) = 0 \quad \text{where} \quad \mathscr{C}(\mathbf{L}) = \mathbf{a}^\top \mathbf{b}. \tag{5}$$

Similarly, one can construct the Plücker coordinates of a line defined as the meet of two planes. The Plücker coordinates of a line defined as the meet of two planes $\mathbf{P}^\top \sim (\bar{\mathbf{P}}^\top | p)$ and $\mathbf{Q}^\top \sim (\bar{\mathbf{Q}}^\top | q)$ are given by:

$$\begin{cases} \mathbf{a} &= p\bar{\mathbf{Q}} - q\bar{\mathbf{P}}, \\ \mathbf{b} &= \bar{\mathbf{P}} \times \bar{\mathbf{Q}}. \end{cases} \tag{6}$$

As an example, triangulation from two views has the following closed-form solution. Let P^1 and P^2 be the two projection matrices and $\mathbf{l}^1$ and $\mathbf{l}^2$ the two imaged lines. The Plücker coordinates of the corresponding 3D line are given as the meet of the two viewing planes $\pi^i \sim \mathsf{P}^{i\top}\mathbf{l}^i$.

Given a standard (3×4) perspective projection matrix $\mathsf{P} \sim (\bar{P}|\mathbf{p})$, a (3×6) matrix projecting Plücker line coordinates [2,5] is given by

$$\tilde{\mathsf{P}} \sim (\det(\bar{\mathsf{P}})\bar{\mathsf{P}}^{-\top} | [\mathbf{p}]_\times \bar{\mathsf{P}}). \tag{7}$$

It can be easily derived by expanding the expression of the 2D line joining the projections of two points:

$$\begin{aligned} \mathbf{l} &\sim \mathbf{m} \wedge \mathbf{n} \\ &\sim (\mathsf{P}\mathbf{M}) \wedge (\mathsf{P}\mathbf{N}) \\ &\sim (\bar{\mathsf{P}}\bar{\mathbf{M}} + m\mathbf{p}) \wedge (\bar{\mathsf{P}}\bar{\mathbf{N}} + n\mathbf{p}) \\ &\sim (\bar{\mathsf{P}}\bar{\mathbf{M}}) \wedge (\bar{\mathsf{P}}\bar{\mathbf{N}}) + m\mathbf{p} \wedge (\bar{\mathsf{P}}\bar{\mathbf{N}}) - n\mathbf{p} \wedge (\bar{\mathsf{P}}\bar{\mathbf{M}}) \\ &\sim \det(\bar{\mathsf{P}})\bar{\mathsf{P}}^{-\top}(\bar{\mathbf{M}} \wedge \bar{\mathbf{N}}) + [\mathbf{p}] \wedge \bar{\mathsf{P}}(m\bar{\mathbf{N}} - n\bar{\mathbf{M}}) \\ &\sim \tilde{\mathsf{P}}\mathbf{L}. \end{aligned}$$

Seo and Hong [20] use the Plücker coordinates representation for sequential structure-from-motion with a Kalman filtering technique. Pottmann et al. [17] use these coordinates for 3D shape reconstruction and understanding from 3D data.

3.1.2. Pair of points or pair of planes

These are two dual representations, described in details in [11, Section 2.2.2]. In the first case, the line is defined as the join of two points **M** and **N**, and in the second case, it is defined as the intersection of two planes **P** and **Q**. These representations have similar characteristics. They have eight parameters, hence four degrees of gauge freedom, the position of the points along the line (respectively, the position of the planes in the pencil of planes around the line) and the scale factors in the homogeneous coordinates of the points or the planes. For metric reconstruction, if one drops

the lines at infinity, the two point representation has six parameters. There is a direct link with Plücker coordinates using Eqs. (4) and (6). The reprojected line $\mathbf{l}$ is a bilinear function of the entries of the point or the plane coordinates. For example, for the two point representation, $\mathbf{l} \sim (\mathsf{P}\mathbf{M}) \times (\mathsf{P}\mathbf{N})$. Hartley [10] proposes a triangulation algorithm based on these representations. Habib et al. [9] use the two point representation for bundle adjustment. They consider that the line is not at infinity. The ambiguity on the position of the points along the line is fixed by constraining them to reprojected near the end-points observed in one of the images.

3.2. Partial representations

3.2.1. Closest point and direction

A 3D line is represented by its closest point to the origin, with coordinates $\mathbf{Q}^\top \sim (\bar{\mathbf{Q}}^\top 1)$, and its direction, with coordinates $\mathbf{Q}_\infty \sim (\bar{\mathbf{Q}}_\infty^\top 0)$, giving a total of six parameters. This representation does not include lines at infinity and hence cannot be used in projective space. The link with the Plücker line coordinates $\mathbf{L}$ is given by

$$\mathbf{L} \sim \begin{pmatrix} \bar{\mathbf{Q}} \times \bar{\mathbf{Q}}_\infty \\ \bar{\mathbf{Q}}_\infty \end{pmatrix}.$$

Reprojecting the line with the camera matrix $\mathsf{P} \sim (\bar{\mathsf{P}}\mathbf{p})$ is a bilinear function of the line parameters: $\mathbf{l} \sim (\bar{\mathsf{P}}\bar{\mathbf{Q}} + \mathbf{p}) \times (\bar{\mathsf{P}}\bar{\mathbf{Q}}_\infty)$. The line reconstruction algorithms proposed by Weng et al. [27] for three views and by Taylor and Kriegman [23] for multiple views use this representation. In the field of photogrammetry, Tommaselli and Lugnani [24] use this representation for bundle adjustment. Mulawa and Mikhail [16] use the additional constraint $\|\bar{\mathbf{Q}}_\infty\| = 1$.

3.2.2. Two projections

A 3D line can be represented by two projections [10,21]. This is related to the fact that reconstructing a line from two views has in general a unique solution.

Spetsakis and Aloimonos [21] use the intersection of two planes, one parallel to the plane $x = 0$, and the other one parallel to the plane $y = 0$. These two planes are formulated using four parameters a, b, c, and d by $x = az + b$ and $y = cz + d$, respectively. The pencil of points $\mathbf{Q}$ on the 3D line is parameterized by the z coordinate

$$\mathbf{Q} \sim \begin{pmatrix} az + b \\ cz + d \\ z \\ 1 \end{pmatrix}.$$

This representation has obvious singularities: lines which are parallel to the plane $z = 0$ cannot be represented. Indeed, the points lying on such lines have a constant z coordinate, and since the points are parameterized by this coordinate, one always gets the same point if the z coordinate is constant. One can link this representation to

the Plücker coordinates $\mathbf{L}$ of the line by considering any two points lying on the line, e.g. for $z = 0$ and $z = 1$, and Eq. (4), giving

$$\mathbf{L} \sim \begin{pmatrix} d \\ -b \\ bc - da \\ a \\ c \\ 1 \end{pmatrix}.$$

Ayache and Faugeras [4] use this representation for mobile robot navigation. In the field of photogrammetry, Habib [8] extends this representation by using different pairs of planes depending on the 3D line, to avoid the singularities.

Hartley [10] uses two images of the line. This representation has the following singularities: all 3D lines lying in an epipolar plane induced by the two cameras have the same images in both views. The 3D lines that cannot be uniquely represented thus form a Linear Line Complex, see e.g. [22]. Note that these singularities can be encountered in practice. The Plücker coordinates corresponding to this representation can be calculated by intersecting the two viewing planes induced by the two image lines using Eq. (6). Hartley shows that the reprojection of the line in other views in a bilinear function of the parameters.

3.2.3. The Denavit–Hartenberg parameters

The Denavit–Hartenberg representation [3] has become the standard way of representing robots and modeling their motions. The idea is to relate each joint to the next by using the minimal four parameters, namely two distances and two angles. A general 3D Euclidean transformation, between two Euclidean coordinate frames, has six degrees of freedom. For using the Denavit–Hartenberg representation, the x-axis of one coordinate frame has to be aligned with the line orthogonal to the z-axes of both coordinate frames, which cancels out two degrees of freedom, one in rotation, and one in translation. This suggests to represent a 3D line by the z-axis of a coordinate frame, and to parameterize it by the four Denavit–Hartenberg parameters with respect to a reference coordinate frame, e.g. the world coordinate frame. The Plücker coordinates corresponding to these parameters can be obtained by e.g. applying the coordinate transformation given by the four parameters to the z-axis $\mathbf{L}_z^\top \sim (0\ 0\ 0\ 0\ 0\ 1)$ of the reference frame using a 3D line rigid displacement matrix [2]. The projection equation is nonlinear in the Denavit–Hartenberg parameters since it involves products and trigonometric operators.

One problem with this parameterization is that two distances are used as parameters, which prevents from representing the lines at infinity. There is also an indeterminacy in the choice of one of the coordinate frame when the line is parallel to the z-axis of the reference coordinate frame.

Roberts [19] proposes to model 3D lines using two distances and two angles. His representation has drawbacks similar to those described above.

Note that there are other representations for modeling robots. For example, Hayati and Mirmirani [12] introduce an extra rotation parameter to the Denavit–Hartenberg representation to model the error due to near parallel axes. This representation is thus not minimal.

3.3. Summary

Table 1 summarizes the characteristics of the aforementioned representations, and of the orthonormal representation that we propose in Section 5. We observe that the only representation for which the reprojected lines is a linear function of the 3D line parameters is the Plücker coordinates. It is also seen that besides our orthonormal representation, no other complete representation allows a minimal update with four parameters, which is due to gauge freedoms and/or internal consistency constraints. Minimal update is an important criterion for using a representation within bundle adjustment.

4. Triangulation

This section discusses computation of structure given camera motion. We propose direct linear and iterative nonlinear methods to recover Plücker line coordinates. These algorithms are general in the sense that they can be used with calibrated, partially calibrated or uncalibrated cameras.

First, we describe a somehow trivial linear algorithm where a biased error function (compared to the reprojection error) is minimized. This algorithm is subject to the same kind of drawback as the eight-point algorithm for computing the fundamental matrix: due to possible noise in the data, the resulting 6-vectors do not generally satisfy the bilinear Plücker constraint (5), similarly to the matrix computed by the eight-point algorithm not being rank deficient [11, Section 10.2]. We propose what we call a *Plücker correction* procedure, which allows to compute the closest Plücker coordinates to a 6-vector.

Second, we propose an algorithm where the reprojection error of the line is minimized. The cornerstone of this algorithm is the linearization of the Plücker constraint.

Table 1
Summary of different representations for 3D lines with their characteristics

Representation	Complete	Gauge freedoms	Constraints	Reprojection	Minimal update
Closest point and direction	No	1	1	Bilinear	No
Two image lines	No	2	0	Bilinear	No
Denavit–Hartenberg	No	0	0	Nonlinear	Yes
Two points or two planes	Yes	4	0	Bilinear	No
Plücker coordinates	Yes	1	1	Linear	No
Orthonormal representation	Yes	0	0	Nonlinear	Yes

The number of gauge freedoms and internal constraints are strongly linked. The 'reprojection' column is about the equation for reprojecting the 3D line with a perspective camera. The column 'minimal update' indicates if the representation can be updated with four parameters.

Since the reconstruction of each line is independent from the others, we drop the j index in this section.

4.1. Linear algorithm

We describe a linear algorithm, 'LIN.' In the reprojection error (3), each term is based on the square of the 2D point-to-line orthogonal distance $d_\perp$, defined by Eq. (1). The denominator of this distance is the cause of the nonlinearity. Ignoring this denominator leads to an algebraic distance denoted d_a, biased compared to the orthogonal distance. It is linear in the predicted line $\mathbf{l}$ and defined by $d_a^2(\mathbf{q},\mathbf{l}) = d_\perp^2(\mathbf{q},\mathbf{l})w^2 = (\mathbf{q}^\top\mathbf{l})^2$, where the scalar factor w encapsulates the bias as $w^2 = l_1^2 + l_2^2$

$$(w^i)^2 = ((\tilde{\mathsf{P}}^i\mathbf{L})_1)^2 + ((\tilde{\mathsf{P}}^i\mathbf{L})_2)^2. \tag{8}$$

We define the biased linear least squares error function:

$$\mathscr{B}(\mathbf{L},\mathscr{M}) = \sum_{i=1}^{n}\left((\mathbf{x}^{i\top}\tilde{\mathsf{P}}^i\mathbf{L})^2 + (\mathbf{y}^{i\top}\tilde{\mathsf{P}}^i\mathbf{L})^2\right) \tag{9}$$

$$= \|\mathsf{A}_{(2n\times 6)}\mathbf{L}\|^2 \text{ with } \mathsf{A} = \begin{pmatrix} \cdots \\ \mathbf{x}^{i\top}\tilde{\mathsf{P}}^i \\ \mathbf{y}^{i\top}\tilde{\mathsf{P}}^i \\ \cdots \end{pmatrix}. \tag{10}$$

Since $\mathbf{L}$ is an homogeneous vector, we add the constraint $\|\mathbf{L}\|^2 = 1$. The $\mathbf{L}$ that minimizes $\mathscr{B}(\mathbf{L},\mathscr{M})$ is then given by the singular vector of A associated to its smallest singular value, that we compute using SVD. Due to noise, the recovered 6-vector does not in general satisfy the Plücker constraint (5).

4.2. Plücker correction

The Plücker correction procedure is analogous to the standard rank correction of the fundamental matrix based on SVD: the eight-point algorithm linearly computes a full-rank matrix F, whose smallest singular value is nullified to obtained the rank-two matrix $\hat{\mathsf{F}}$, see e.g. [11]. Matrix $\hat{\mathsf{F}}$ is the closest rank-two matrix to F, in the sense of the Frobenius norm. It is used to initialize nonlinear algorithms.

The Plücker correction procedure computes the closest Plücker coordinates to a given 6-vector, where closest is to be understood in the sense of the $\mathscr{L}_2$-norm, equivalent to the matrix Frobenius norm. It is also equivalent to the Euclidean distance between two points in $\mathbb{R}^6$. This correction is necessary to initialize the nonlinear algorithms from the solution provided by linear methods ignoring the Plücker constraint. Pottmann et al. [17] use the Euclidean distance between Plücker coordinate vectors to compare 3D lines. They underline the facts that this distance is practical for minimization purposes and is in accordance with visualization in the region of interest, i.e., near the origin.

More formally, let $\mathbf{L}^\top \sim (\mathbf{a}^\top \mid \mathbf{b}^\top)$ be a 6-vector that does not necessarily satisfy the Plücker constraint (5), i.e., $\mathbf{a}^\top\mathbf{b}$ might be nonzero. We seek $\hat{\mathbf{L}}^\top \sim (\mathbf{u}^\top|\mathbf{v}^\top)$, defined

by $\min_{\hat{\mathbf{L}},\mathbf{u}^\top\mathbf{v}=0}\|\hat{\mathbf{L}}-\mathbf{L}\|^2$. This is a linear least squares optimization problem under a nonlinear constraint. Although it has a clear and concise formulation, it is *not* trivial.

Obviously, one can modify one entry of the Plücker coordinates in accordance with the Plücker constraint, e.g. set $a_1 = -(a_2b_2 + a_3b_3)/b_1$. This simple solution has the disadvantage that the entry must be chosen depending on the actual values of the coordinates since the correction rule uses a division. Also, all entries are clearly not treated uniformly.

By comparison, our solution orthogonally projects the 6-vector on the Klein quadric and treats all its entries the same way. Kanatani [13] proposes a general iterative scheme for projecting points on nonlinear manifolds, such as projecting points in $\mathbb{R}^6$ on the Klein quadric. Our algorithm performs this projection in a noniterative manner, which thus guarantees that the optimal projected point on the Klein quadric, i.e., the optimal 3D line, is found. Its derivation is quite tricky but it can be readily implemented with few lines of code from its summary shown in Table 2.

4.2.1. A geometric interpretation

We interpret the 3-vectors **a**, **b**, **u**, and **v** as coordinates of 3D points. These points are not directly linked to the underlying 3D line. This interpretation is just intended to visualize the problem. The Plücker constraint $\mathbf{u}^\top\mathbf{v}$ corresponds to the fact that the lines induced by the origin with **u** and **v** are perpendicular. The correction criterion is the sum of squared Euclidean distances between **a** and **u** and between **b** and **v**. Hence, the problem may be formulated as finding two points **u** and **v**, as close as possible to **a** and **b**, respectively, and such that the lines induced by the origin with **u** and **v** are perpendicular. We begin by rotating the coordinate frame such that **a** and **b** are transferred on the $z = 0$ plane. This is the *reduction of the problem*. We *solve the reduced problem*, by finding two points on the $z = 0$ plane, minimizing the correction criterion and satisfying the Plücker constraint. Finally, we *express the solution* back to the original space.

4.2.2. Reducing the problem

Let us define the (3×2) matrices $\bar{\mathsf{C}} \sim (\mathbf{ab})$ and $\hat{\mathsf{C}} \sim (\mathbf{uv})$. The Plücker constraint is fulfilled if and only if the columns of matrix $\hat{\mathsf{C}}$ are orthogonal. We rewrite the correction criterion as

$$\mathcal{C} = \|\mathbf{L} - \hat{\mathbf{L}}\|^2 = \|\bar{\mathsf{C}} - \hat{\mathsf{C}}\|^2.$$

Table 2
The *Plücker correction* algorithm

- Compute the Singular Value Decomposition $(\mathbf{ab}) = \bar{\mathsf{U}}\bar{\Sigma}\bar{\mathsf{V}}^\top$.
- Let $\bar{\mathsf{Z}} = \bar{\Sigma}\bar{\mathsf{V}}^\top$, form matrix $\mathsf{T} = \begin{pmatrix} z_{21} & z_{22} \\ z_{12} & -z_{11} \end{pmatrix}$.
- Compute the singular vector $\hat{\mathbf{v}}$ associated to the smallest singular value of matrix T.
- Define $\bar{\mathsf{V}} = \begin{pmatrix} \hat{v}_1 & -\hat{v}_2 \\ \hat{v}_2 & \hat{v}_1 \end{pmatrix}$ and set $(\mathbf{uv}) \sim \bar{\mathsf{U}}\hat{\mathsf{V}}\mathrm{diag}(\hat{\mathsf{V}}^\top\bar{\Sigma}\bar{\mathsf{V}}^\top)$.

Given a 6-vector $\mathbf{L}^\top \sim (\mathbf{a}^\top \mid \mathbf{b}^\top)$, this algorithm computes the closest Plücker coordinates $\hat{\mathbf{L}}^\top \sim (\mathbf{u}^\top|\mathbf{v}^\top)$, i.e., $\mathbf{u}^\top\mathbf{v} = 0$, in the sense of the $\mathscr{L}_2$-norm, i.e., $\|\hat{\mathbf{L}} - \mathbf{L}\|^2$ is minimized.

Using the following SVD $\bar{\mathsf{C}}_{(3\times2)} = \bar{\mathsf{U}}_{(3\times2)}\bar{\Sigma}_{(2\times2)}\bar{\mathsf{V}}^{\top}_{(2\times2)}$

$$\mathscr{O} = \|\bar{\mathsf{U}}\bar{\Sigma}\bar{\mathsf{V}}^{\top} - \hat{\mathsf{C}}\|^2 = \|\bar{\Sigma}\bar{\mathsf{V}}^{\top} - \bar{\mathsf{U}}^{\top}\hat{\mathsf{C}}\|^2,$$

since $\bar{\mathsf{U}}$ has orthonormal columns. We define $\bar{\mathsf{Z}} = \bar{\Sigma}\bar{\mathsf{V}}^{\top}$ and $\hat{\mathsf{Z}} = \bar{\mathsf{U}}^{\top}\hat{\mathsf{C}}$. Matrix $\bar{\mathsf{V}}$ is orthonormal and $\bar{\Sigma}$ is diagonal, hence the rows of $\bar{\mathsf{Z}}$ are orthogonal (i.e., $\bar{\mathsf{Z}}\bar{\mathsf{Z}}^{\top}$ is diagonal, but not $\bar{\mathsf{Z}}^{\top}\bar{\mathsf{Z}}$). Note that $\hat{\mathsf{Z}} = \bar{\mathsf{U}}^{\top}\hat{\mathsf{C}}$ implies $\hat{\mathsf{C}} = \bar{\mathsf{U}}\hat{\mathsf{Z}}$, even if $\bar{\mathsf{U}}\bar{\mathsf{U}}^{\top}$ is not the identity.[3] The problem is reduced to finding a column-orthogonal[4] matrix $\hat{\mathsf{Z}}$, as close as possible to the row-orthogonal matrix $\bar{\mathsf{Z}}$.

4.2.3. Solving the reduced problem

We parameterize the column-orthogonal matrix $\hat{\mathsf{Z}}$ as $\hat{\mathsf{Z}} = \hat{\mathsf{V}}\hat{\Sigma}$, where $\hat{\mathsf{V}}$ is orthonormal and $\hat{\Sigma}$ is diagonal. Hence

$$\mathscr{O} = \|\bar{\Sigma}\bar{\mathsf{V}}^{\top} - \hat{\mathsf{V}}\hat{\Sigma}\|^2 = \|\hat{\mathsf{V}}^{\top}\bar{\Sigma}\bar{\mathsf{V}}^{\top} - \hat{\Sigma}\|^2.$$

The diagonal matrix $\hat{\Sigma}$ which minimizes this expression is given by the diagonal entries of $\hat{\mathsf{V}}^{\top}\bar{\Sigma}\bar{\mathsf{V}}^{\top}$, and does not depend on the solution for $\hat{\mathsf{V}}$. The orthonormal matrix $\hat{\mathsf{V}} = (\hat{\mathbf{v}}_1\ \hat{\mathbf{v}}_2)$ is given by minimizing the sum of squares of the off-diagonal entries of $\hat{\mathsf{V}}^{\top}\bar{\mathsf{Z}}$, with $\bar{\mathsf{Z}} = \bar{\Sigma}\bar{\mathsf{V}}^{\top} = (\mathbf{z}_1\mathbf{z}_2)$

$$\mathscr{O} = (\hat{\mathbf{v}}_1^{\top}\mathbf{z}_2)^2 + (\hat{\mathbf{v}}_2^{\top}\mathbf{z}_1)^2.$$

Define the 2D rotation matrix with angle $\pi/2$ by $\mathsf{M} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$ and parameterize the orthonormal matrix $\hat{\mathsf{V}}$ by a unit vector $\hat{\mathbf{v}}$, as:

$$\begin{cases} \hat{\mathbf{v}}_1 &= \hat{\mathbf{v}}, \\ \hat{\mathbf{v}}_2 &= \mathsf{M}\hat{\mathbf{v}}, \end{cases}$$

The correction criterion can be rewritten as

$$\mathscr{O} = (\hat{\mathbf{v}}^{\top}\mathbf{z}_2)^2 + (\hat{\mathbf{v}}^{\top}\mathsf{M}^{\top}\mathbf{z}_1)^2 = \|\mathsf{T}\hat{\mathbf{v}}\|^2 \text{ with } \mathsf{T} = \begin{pmatrix} \mathbf{z}_2^{\top} \\ \mathbf{z}_1^{\top}\mathsf{M} \end{pmatrix}.$$

The unit vector $\hat{\mathbf{v}}$ minimizing this expression is given by the singular vector associated to the smallest singular value of matrix T.

4.2.4. Expressing the solution

From vector $\hat{\mathbf{v}}$ which solves the reduced problem, we form the orthonormal matrix $\hat{\mathsf{V}} = \begin{pmatrix} \hat{v}_1 & -\hat{v}_2 \\ \hat{v}_2 & \hat{v}_1 \end{pmatrix}$. The diagonal matrix $\hat{\Sigma}$ is given by $\hat{\Sigma} = \text{diag}(\hat{\mathsf{V}}^{\top}\bar{\Sigma}\bar{\mathsf{V}}^{\top})$.

[3] Indeed, denote $\mathbf{u}_i$ the columns of matrix $\bar{\mathsf{U}}$ and form $\mathsf{U} = (\mathbf{u}_1\ \mathbf{u}_2\ \mathbf{u}_1\times\mathbf{u}_2)$. We have $\mathsf{U}^{\top}\bar{\mathsf{U}} = (\mathbf{I}_{(2\times2)}\mathbf{0}_{(2\times1)})^{\top}$. Let us multiply the correction criterion by $\mathsf{U}^{\top}$: $\mathscr{O} = \|(\bar{\mathsf{V}}\bar{\Sigma}\mathbf{0}_{(2\times1)})^{\top} - \mathsf{U}^{\top}\hat{\mathsf{C}}\|^2$. Denote $\mathsf{Y}_{(3\times2)} = \mathsf{U}^{\top}\hat{\mathsf{C}}$. The optimal solution has the form $\mathsf{Y}^{\top} = (\hat{\mathsf{Z}}^{\top}\mathbf{0}_{(2\times1)})$, since, according to the geometric interpretation, the corrected points $\mathbf{u}$ and $\mathbf{v}$ must lie on the plane defined by points $\mathbf{a}$, $\mathbf{b}$ and the origin, the plane $z = 0$. Therefore, we obtain $\hat{\mathsf{C}} = \mathsf{U}\mathsf{Y} = \bar{\mathsf{U}}\bar{\mathsf{Y}}$.

[4] The fact that matrix $\hat{\mathsf{Z}} = \bar{\mathsf{U}}^{\top}\hat{\mathsf{C}}$ is column-orthogonal is induced from the Plücker constraint. Indeed, this constraint implies that $\hat{\mathsf{C}}$ is column-orthogonal, hence $\hat{\mathsf{C}}^{\top}\hat{\mathsf{C}}$ is diagonal. Matrix $\mathsf{U}^{\top}\hat{\mathsf{C}}$, where $SO(3) \ni \mathsf{U} = (\mathbf{u}_1\ \mathbf{u}_2\ \mathbf{u}_1\times\mathbf{u}_2) = (\bar{\mathsf{U}}\bar{\mathbf{u}})$, is also column-orthogonal. Observe that $\hat{\mathsf{C}}^{\top}\mathsf{U}\mathsf{U}^{\top}\hat{\mathsf{C}} = \hat{\mathsf{C}}^{\top}\bar{\mathsf{U}}\bar{\mathsf{U}}^{\top}\hat{\mathsf{C}} + \hat{\mathsf{C}}^{\top}\bar{\mathbf{u}}\bar{\mathbf{u}}^{\top}\hat{\mathsf{C}} = \hat{\mathsf{C}}^{\top}\bar{\mathsf{U}}\bar{\mathsf{U}}^{\top}\hat{\mathsf{C}}$ since $\bar{\mathbf{u}}^{\top}\hat{\mathsf{C}} = \mathbf{0}^{\top}$. Hence, matrix $\bar{\mathsf{U}}^{\top}\hat{\mathsf{C}}$ is column-orthogonal.

4.3. Quasi-linear algorithms

We describe algorithms 'QLIN1' and 'QLIN2,' that consider the reprojection error (3). They are based on an iterative bias-correction, through reweighting of the biased error function (9). Such algorithms are coined quasi-linear.

We showed previously that the orthogonal and the algebraic distances are related by a scalar factor, given by Eq. (8), depending on the 3D line. The reprojection error and the biased error functions are therefore related by a set of such factors, one for each image of the line. The fact that these factors depend on the unknown 3D line suggests an iterative reweighting scheme.

The first approach that comes to mind is 'QLIN1.' The linear system considered for method LIN is formed and solved. The resulting 6-vector $\mathbf{L}_0$ is corrected to be valid Plücker coordinates. This yields a biased estimate of the 3D line. Using this estimate, weight factors that contain the bias of the linear least squares error function are computed, and used to reweight the equations. The process is iterated to compute successive refined estimates $\mathbf{L}_k$ until convergence, where k is the iteration counter. Convergence is determined by thresholding the difference between two consecutive errors. It is typically reached in three or four iterations.

Experimental results show that this naive approach performs very badly, see Section 6. This is due to the fact that the Plücker constraint is enforced afterhand and is not taken into account while solving the linear least squares system.

To remedy to this problem, we propose 'QLIN2,' that linearizes and enforces the Plücker constraint (5), as follows. The algorithm is summarized in Table 3. Rewrite the constraint as $\mathscr{C}(\mathbf{L}) = \mathbf{L}^\top \mathsf{G}\mathbf{L}$ where $\mathsf{G}_{(6\times6)} = \begin{pmatrix} 0 & I \\ I & 0 \end{pmatrix}$. By expanding this expression to first order around the estimate $\mathbf{L}_k$, and after some minor algebraic manipulations, we obtain the following linear constraint on $\mathbf{L}_{k+1}$:

$$\mathscr{C}_k(\mathbf{L}_{k+1}) = \mathbf{L}_k^\top \mathsf{G}\mathbf{L}_{k+1} = 0.$$

We follow the constrained linear least squares optimization method summarized in [11, Section A3.4.3] to enforce this linearized constraint, as well as $\|\mathbf{L}_{k+1}\| = 1$. The idea is to find an orthonormal basis of all possible vectors satisfying the constraint and to solve for a 5-vector $\boldsymbol{\gamma}$ expressed in this basis. Such an orthonormal basis is provided by computing the nullspace of $\mathbf{L}_k^\top \mathsf{G}$ using SVD. Let $\bar{\mathsf{V}}$ be a (6×5) orthonormal matrix whose columns span the basis (i.e., $\mathbf{L}_k^\top \mathsf{G}\bar{\mathsf{V}} = \mathbf{0}$), we define $\mathbf{L}_{k+1} = \bar{\mathsf{V}}\boldsymbol{\gamma}$, hence $\mathscr{C}_k(\mathbf{L}_{k+1}) = \mathbf{L}_k^\top \mathsf{G}\bar{\mathsf{V}}\boldsymbol{\gamma} = 0$ and $\|\mathbf{L}_{k+1}\| = \|\boldsymbol{\gamma}\|$. We solve for $\boldsymbol{\gamma}$ by substituting in

Table 3
The quasi-linear algorithm 'QLIN2' for optimal triangulation

1. *Plücker correction procedure described in Section 4.2. Set* $k = 0$
2. *Constraint linearization:* Compute the singular value decomposition $\mathbf{L}_k^\top \mathsf{G} \sim \mathbf{u}^\top \mathrm{diag}(1,0,0,0,0,0)(\mathbf{v}_{(6\times1)}|\bar{\mathsf{V}}_{(6\times5)})^\top$
3. *Estimation:* Compute $\min_{\boldsymbol{\gamma},\|\boldsymbol{\gamma}\|^2=1} \|\mathsf{A}\bar{\mathsf{V}}\boldsymbol{\gamma}\|^2$ and set $\mathbf{L}_{k+1} = \bar{\mathsf{V}}\boldsymbol{\gamma}$
4. *Bias-correction:* Reweight the linear system A by computing the weights according to Eq. (8)
5. *Iteration:* Iterate steps 2, 3, and 4 until convergence

Eq. (10) ($\|\mathsf{A}\mathbf{L}_{k+1}\|^2 = \|\mathsf{A}\bar{\mathsf{V}}\boldsymbol{\gamma}\|^2$). The singular vector associated to the smallest singular value of matrix $\mathsf{A}\bar{\mathsf{V}}$ provides the solution vector with unit $\mathscr{L}_2$-norm such that $\mathscr{B}(\mathbf{L}_{k+1}, \mathscr{M})$ is minimized.

5. Bundle adjustment

Bundle adjustment is the nonlinear minimization of the reprojection error (2), over camera and 3D line parameters. We focus on the parameterization of 3D lines. Parameterizing the camera motion has been addressed in e.g. [1,11, Section A4.6].

5.1. Problem statement

As said in Section 1, there are various possibilities to overparameterize the four-dimensional set of 3D lines. In the context of nonlinear optimization, choosing an overparameterized representation may induce the following problems. First, the computational cost of each iteration is increased by superfluous parameters. Second, artificial freedoms in the parameter set (internal gauge freedoms) are induced and may give rise to numerical instabilities. Third, some internal consistency constraints, such as the Plücker constraint, may have to be enforced.

These reasons motivate the need for a representation of 3D lines allowing nonlinear optimization with the minimum four parameters. In that case, there is no free scale induced by homogeneity or internal consistency constraints, and an unconstrained optimization engine can be used.

5.2. The orthonormal representation

The orthonormal representation has been introduced in [1] for the nonlinear optimization of the fundamental matrix with the minimum seven parameters. It consists in finding a representation involving elements of $SO(n)$ and scalars (hence the term 'orthonormal representation'). In particular, no other algebraic constraints should be necessary, such as the rank-two constraint of fundamental matrices or the bilinear Plücker constraint. Using orthonormal matrices implies that the representation is well-conditioned. Based on such a representation, local update using the minimum number of parameters is possible.

Commonly used nonlinear optimization engine, e.g., Newton type such as Levenberg–Marquardt, often require the Jacobian matrix of the error function with respect to the update parameters. In the orthonormal representation framework, we split it as the product of the Jacobian matrix of the error function considered with respect to the 'standard' entity representation, e.g., the fundamental matrix or Plücker coordinates, and the *orthonormal Jacobian matrix*, i.e., for the 'standard' representation with respect to the update parameters.

5.2.1. Example: representing $\mathbb{P}^1$

We derive the orthonormal representation of the one-dimensional projective space $\mathbb{P}^1$. This is used in Section 5.3 to derive the orthonormal representation of

3D lines. Let $\boldsymbol{\sigma} \in \mathbb{P}^1$. Such a 2-vector is defined up to scale and has therefore only one degree of freedom. We represent it by an $SO(2)$ matrix W defined by

$$\mathsf{W} = \frac{1}{\|\boldsymbol{\sigma}\|} \begin{pmatrix} \sigma_1 & -\sigma_2 \\ \sigma_2 & \sigma_1 \end{pmatrix}. \tag{11}$$

The first column of this matrix is $\boldsymbol{\sigma}$ itself, normalized to unit-norm. Let θ be the update parameter. A local update step is $\mathsf{W} \leftarrow \mathsf{W}\,\mathsf{R}(\theta)$ where $\mathsf{R}(\theta)$ is the 2D rotation matrix of angle θ. The Jacobian matrix $\frac{\partial \boldsymbol{\sigma}}{\partial \theta}$ evaluated at $\theta_0 = 0$ (the update is with respect to a base rotation) is given by

$$\left.\frac{\partial \boldsymbol{\sigma}}{\partial \theta}\right|_{\theta_0} = \left.\frac{\partial \mathbf{w}_1}{\partial \theta}\right|_{\theta_0} = \begin{pmatrix} -\sigma_2 \\ \sigma_1 \end{pmatrix} = \mathbf{w}_2, \tag{12}$$

where $\mathbf{w}_i$ is the ith column of W.

5.2.2. Updating $SO(3)$

A matrix $\mathsf{U} \in SO(3)$ can be locally updated using three parameters by any well-behaved (locally nonsingular) representation, such as three Euler angles $\boldsymbol{\theta}^\top = (\theta_1 \mid \theta_2 \mid \theta_3)$ as

$$\mathsf{U} \leftarrow \mathsf{U}\mathsf{R}(\boldsymbol{\theta}) \text{ with } \mathsf{R}(\boldsymbol{\theta}) = \mathsf{R}_x(\theta_1)\mathsf{R}_y(\theta_2)\mathsf{R}_z(\theta_3), \tag{13}$$

where $\mathsf{R}_x(\theta_1)$, $\mathsf{R}_y(\theta_2)$, and $\mathsf{R}_z(\theta_3)$ are $SO(3)$ matrices representing 3D rotations around the x-, y- and z-axes with angle θ_1, θ_2, and θ_3, respectively. The Jacobian matrix is derived as follows. As in the $SO(2)$ case, the update is with respect to a base rotation. The orthonormal Jacobian matrix is therefore evaluated at $\boldsymbol{\theta}_0 = \mathbf{0}_{(3\times 1)}$

$$\left.\frac{\partial \mathsf{U}}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta}_0} = \left(\left.\frac{\partial \mathsf{U}}{\partial \theta_1}\right|_{\boldsymbol{\theta}_0} \left| \left.\frac{\partial \mathsf{U}}{\partial \theta_2}\right|_{\boldsymbol{\theta}_0} \right| \left.\frac{\partial \mathsf{U}}{\partial \theta_3}\right|_{\boldsymbol{\theta}_0} \right).$$

After minor algebraic manipulations, we obtain

$$\left.\frac{\partial \mathsf{U}}{\partial \theta_1}\right|_{\boldsymbol{\theta}_0} = \left.\frac{\partial (\mathsf{U}\mathsf{R}_x(\theta_1)\mathsf{R}_y(\theta_2)\mathsf{R}_z(\theta_3))}{\partial \theta_1}\right|_{\boldsymbol{\theta}_0} = (\mathbf{0}_3 | \mathbf{u}_3 | - \mathbf{u}_2), \tag{14}$$

where $\mathbf{u}_i$ is the ith column of U. Similarly:

$$\left.\frac{\partial \mathsf{U}}{\partial \theta_2}\right|_{\boldsymbol{\theta}_0} = (-\mathbf{u}_3 | \mathbf{0}_3 | \mathbf{u}_1) \tag{15}$$

$$\left.\frac{\partial \mathsf{U}}{\partial \theta_3}\right|_{\boldsymbol{\theta}_0} = (\mathbf{u}_2 | - \mathbf{u}_1 | \mathbf{0}_3). \tag{16}$$

These expressions are vectorized to form the orthonormal Jacobian matrix.

5.3. The case of 3D lines

The case of 3D lines is strongly linked with the cases of $SO(2)$ and $SO(3)$, as shown by the following result:

Any (projective) 3D line $\mathbf{L}$ *can be represented by*

$$(\mathsf{U},\mathsf{W}) \in SO(3) \times SO(2),$$

where $SO(2)$ and $SO(3)$ are the Lie groups of respectively (2×2) and (3×3) rotation matrices. (U,W) is the orthonormal representation of the 3D line $\mathbf{L}$.

The proof of this result is obtained by showing that any 3D line has an orthonormal representation $(\mathsf{U},\mathsf{W}) \in SO(3)\times SO(2)$, while any $(\mathsf{U},\mathsf{W}) \in SO(3)\times SO(2)$ corresponds to a unique 3D line. The next paragraph illustrates this by means of Plücker coordinates.

Note that this result is consistent with the fact that a 3D line has four degrees of freedom, since an element of $SO(2)$ has one degree of freedom and an element of $SO(3)$ has three degrees of freedom.

Using this representation of 3D lines, we show that there exists a locally nonsingular minimal parameterization. Therefore, 3D lines can be locally updated with the minimum four parameters. The update scheme is inspired from those given above for 2D and 3D rotation matrices, and can be plugged into most of the existing nonlinear optimization algorithms. These results are summarized in Table 4.

5.3.1. Relating Plücker coordinates and the orthonormal representation

The orthonormal representation of a 3D line can be computed from its Plücker coordinates $\mathbf{L}^\top \sim (\mathbf{a}^\top \mid \mathbf{b}^\top)$, as follows. Let $\bar{\mathsf{C}}_{(3\times 2)} \sim (\mathbf{a}|\mathbf{b})$ be factored as

$$\bar{\mathsf{C}} \sim \underbrace{\begin{pmatrix} \frac{\mathbf{a}}{\|\mathbf{a}\|} & \frac{\mathbf{b}}{\|\mathbf{b}\|} & \frac{\mathbf{a}\times\mathbf{b}}{\|\mathbf{a}\times\mathbf{b}\|} \end{pmatrix}}_{SO(3)} \underbrace{\begin{pmatrix} \|\mathbf{a}\| & \\ & \|\mathbf{b}\| \end{pmatrix}}_{(\|\mathbf{a}\|\|\mathbf{b}\|)^\top \in \mathbb{P}^1}.$$

In practice, we use QR decomposition, $\bar{\mathsf{C}}_{(3\times 2)} = \mathsf{U}_{(3\times 3)}\Sigma_{(3\times 2)}$. The special form of matrix Σ, i.e., the zero at the (1,2) entry is due to the Plücker constraint. While $\mathsf{U} \in SO(3)$, the two nonzero entries of Σ defined up to scale can be represented by an $SO(2)$ matrix W, as shown in Section 5.2.

Going back from the orthonormal representation to Plücker coordinates is trivial. The Plücker coordinates of the line are obtained from its orthonormal representation (U,W) as

Table 4
Elements for 3D line optimization using the minimal four parameters through the orthonormal representation

Initialization. The initial guess is given by the Plücker coordinates $\mathbf{L}_0^\top \sim (\mathbf{a}_0^\top|\mathbf{b}_0^\top)$

- Compute the orthonormal representation $(\mathsf{U},\mathsf{W}) \in SO(3)\times SO(2)$ of $\mathbf{L}_0$ by QR decomposition

$$(\mathbf{a}_0|\mathbf{b}_0) = \mathsf{U}\begin{pmatrix} \sigma_1 & \\ & \sigma_2 \end{pmatrix} \text{ and set } \mathsf{W} = \begin{pmatrix} \sigma_1 & -\sigma_2 \\ \sigma_2 & \sigma_1 \end{pmatrix}$$

- The four optimization parameters are $\mathbf{p}^\top = (\boldsymbol{\theta}^\top \mid \theta)$ where the 3-vector $\boldsymbol{\theta}$ and the scalar θ are used to update U and W, respectively

Update. (i.e., one optimization step)

- Current line is $\mathbf{L}^\top \sim (w_{11}\mathbf{u}_1^\top | w_{21}\mathbf{u}_2^\top)$ and $\partial\mathbf{L}/\partial\mathbf{p}$ is given by Eq. (18)
- Compute $\mathbf{p}$ by minimizing some criterion
- Update U and W: $\mathsf{U} \leftarrow \mathsf{U}\,\mathsf{R}(\boldsymbol{\theta})$ and $\mathsf{W} \leftarrow \mathsf{W}\,\mathsf{R}(\theta)$

$$\mathbf{L}^\top \sim (w_{11}\mathbf{u}_1^\top | w_{21}\mathbf{u}_2^\top), \tag{17}$$

where $\mathbf{u}_i$ is the ith column of U.

5.3.2. A 4-parameter update

Consider $(\mathsf{U},\mathsf{W}) \in SO(3) \times SO(2)$, the orthonormal representation of a 3D line. Since $\mathsf{U} \in SO(3)$, as reviewed in Section 5.2, it can not be minimally parameterized but can be locally updated using Eq. (13), as $\mathsf{U} \leftarrow \mathsf{UR}(\boldsymbol{\theta})$ where $\boldsymbol{\theta} \in \mathbb{R}^3$. Matrix $\mathsf{W} \in SO(2)$ can be updated as $\mathsf{W} \leftarrow \mathsf{WR}(\theta)$, where $\theta \in \mathbb{R}$. We define the update parameters by the 4-vector $\mathbf{p}^\top \sim (\boldsymbol{\theta}^\top \mid \theta)$.

We denote J the (6×4) Jacobian matrix of the Plücker coordinates, with respect to the orthonormal representation. Matrix J must be evaluated at $\mathbf{p}_0 = \mathbf{0}_{(4\times1)}$:

$$\mathsf{J}|_{\mathbf{p}_0} = \left(\frac{\partial \mathbf{L}}{\partial \theta_1}\bigg|_{\mathbf{p}_0} | \frac{\partial \mathbf{L}}{\partial \theta_2}\bigg|_{\mathbf{p}_0} | \frac{\partial \mathbf{L}}{\partial \theta_3}\bigg|_{\mathbf{p}_0} | \frac{\partial \mathbf{L}}{\partial \theta}\bigg|_{\mathbf{p}_0} \right).$$

By using the orthonormal representation to Plücker coordinates Eq. (17) and the Jacobian matrices for $SO(2)$ and $SO(3)$, as defined by Eqs. (12), (14)–(16), we obtain, after minor algebraic manipulations:

$$\mathsf{J}_{(6\times4)} = \begin{pmatrix} \mathbf{0}_{(3\times1)} & -\sigma_1\mathbf{u}_3 & \sigma_1\mathbf{u}_2 & -\sigma_2\mathbf{u}_1 \\ \sigma_2\mathbf{u}_3 & \mathbf{0}_{(3\times1)} & -\sigma_2\mathbf{u}_1 & \sigma_1\mathbf{u}_2 \end{pmatrix}. \tag{18}$$

5.3.3. Geometric interpretation

Each of the four above-defined update parameters $\mathbf{p}$ has a geometric interpretation. Matrix W encapsulates the ratio $\|\mathbf{a}\|/\|\mathbf{b}\|$, hence the distance d from the origin $\mathbf{O}$ to $\mathbf{L}$. Thus, parameter θ acts on d. Matrix U is related to a 3D coordinate frame attached to $\mathbf{L}$. Parameter θ_1 rotates $\mathbf{L}$ around a circle with radius d, centered on $\mathbf{O}$, and lying on the plane defined by $\mathbf{O}$ and $\mathbf{L}$. Parameter θ_2 rotates $\mathbf{L}$ around a circle with radius d, centered on $\mathbf{O}$, and lying in a plane containing $\mathbf{O}$, the closest point $\mathbf{Q}$ of $\mathbf{L}$ to $\mathbf{O}$, and perpendicular to $\mathbf{L}$. Parameter θ_3 rotates $\mathbf{L}$ around the axis defined by $\mathbf{O}$ and $\mathbf{Q}$. For the last three cases, the angles of rotation are the parameters themselves. This interpretation allows to easily incorporate a priori knowledge while estimating a line. For example, to leave the direction of the line invariant, one may use the two update parameters θ_2 and θ, while to leave the distance of the line to the origin invariant, one may use the three update parameters $\boldsymbol{\theta}$. This allows to solve constrained line estimation cases, as summarized in the table below, indicating which update parameters to optimize in which case

Scenario	θ_1	θ_2	θ_3	θ
Fixed direction		×		×
Fixed normal to the plane formed with the origin	×			×
Fixed distance to the origin	×	×	×	

6. Experimental results

6.1. Simulated data

Our simulated experimental setup consists of a set of cameras looking inwards at 3D lines randomly chosen in a sphere with a 1 m radius. Cameras are spread widely around the sphere, at a distance of roughly 10 m away from the centre of the sphere. We fix the focal length of the cameras to 1000 (in number of pixels). Note that this information is not used in the rest of the experiments. The end-points of all lines are projected in all views, where their positions are corrupted by an additive Gaussian noise. We vary the parameters of this setup to assess and compare the quality of the different estimators on various scene configurations.

We compare the four methods given in this paper: LIN, QLIN1, QLIN2, and MLE (bundle adjustment based on our orthonormal representation of 3D lines), as well as the method given in [11, Section 15.4.1], denoted by 'MLE_HARTLEY.' This method consists in nonlinearly computing the trifocal tensor as well as reconstructed lines by minimizing the reprojection error (2) and parameterizing the 3D lines by two of their three images. We also compare QLIN2 to a direct Levenberg–Marquardt-based minimization of the reprojection error, dubbed NLIN: the two methods gave undistinguishable results in all our experiments. Note that most existing methods, e.g. [14,21,23,27] can be applied only when camera calibration is available.

We measure the quality of an estimate using the *estimation error*, as described in [11, Section 4], which also provides the theoretical lower bound. The estimation error is equivalent to the value of the negative log likelihood (2) (i.e., the reprojection error).

The results are shown on graphs on Figs. 1 and 2. We observe that the different methods are always in the same order. Three distinct behaviours can be seen. Methods LIN and QLIN1 give similar results since they are subject to the same bias induced by ignoring the Plücker constraint until the final correction. Methods QLIN2 and NLIN are undistinguishable. They give better results than the biased methods. Finally, methods MLE and MLE_HARTLEY are hardly ever distinguishable. Their results are the best since they adjust the camera positions along with the 3D line parameters.

In more details, we vary the added noise level from 0 to 2 pixels, while considering 20 lines and three views on Fig. 1A. One observes that, beyond one pixel noise, methods LIN and QLIN1 behave very badly. This is mainly due to the bias introduced by the Plücker correction procedure. Methods QLIN2, MLE, and MLE_HARTLEY degrade gracefully as the noise level increases. Method QLIN2 gives reasonable results. Methods MLE and MLE_HARTLEY give undistinguishable results, very close to the theoretical lower bound.

We vary the number of lines from 15 to 60, while considering a one pixel noise and three views on Fig. 1B. Similar conclusions as for the previous experiment can be drawn, except for the fact, that when more than 30 lines are considered, methods LIN and QLIN1 give reasonable results. Also, methods MLE and MLE_HARTLEY give results undistinguishable from the theoretical lower bound when more than 45 lines are considered.

Fig. 2A shows the results when the number of images is varied from 3 to 12. The algorithms that do not optimize the cameras, namely LIN, QLIN1, QLIN2, and NLIN,

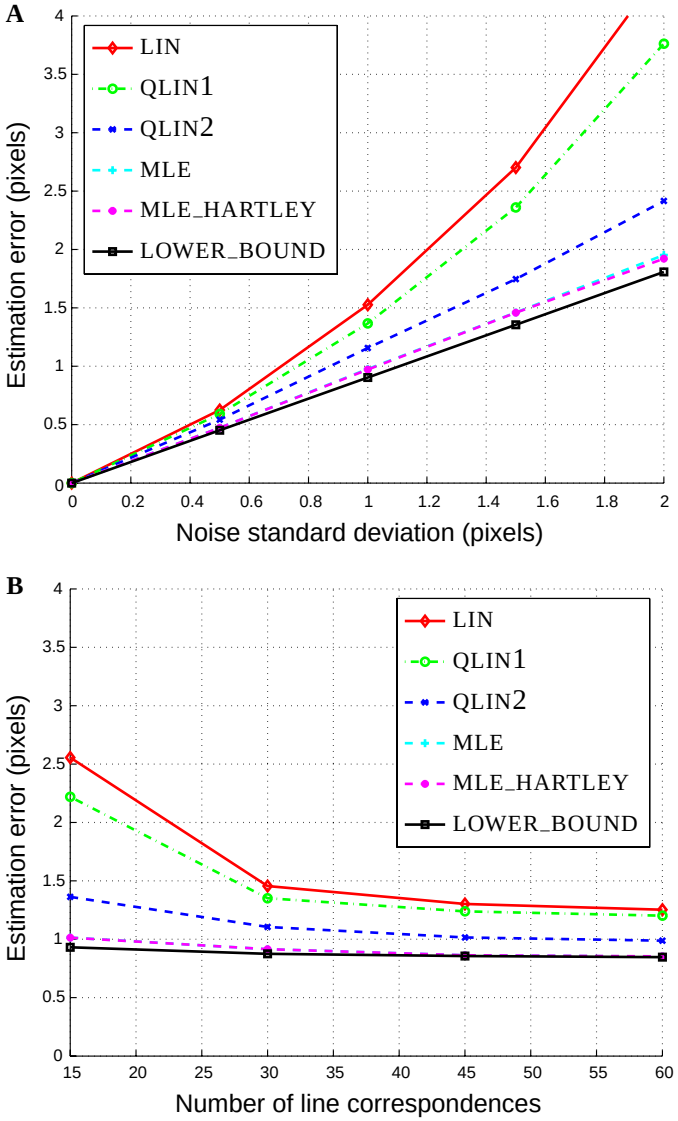


Fig. 1. Estimation error for different methods when varying the variance of added noise on image endpoints (A) and the number of lines considered (B).

have an error which increases with the number of images, whereas the bundle adjustment algorithms, namely MLE and MLE_HARTLEY, have an error which decreases. This is due to the fact that when the number of images increases, the initial camera estimation degrades, which is characteristic of the camera initialization algorithm.

When the distance between the lines and the cameras increases, Fig. 2B shows that the error decreases for all methods. This is explained by the fact that the cloud of 3D lines gets smaller and smaller in the images, which decrease the estimation error, but does not mean that the estimate is better.

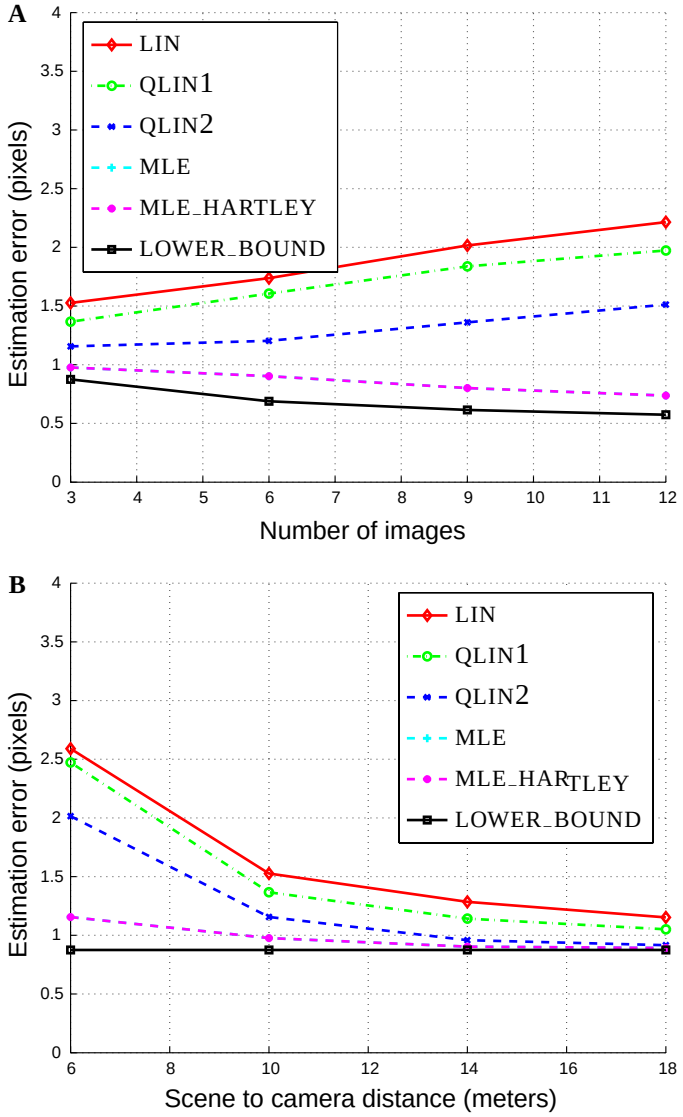


Fig. 2. Estimation error for different methods when varying the number of images (A) and the scene to camera distance (B).

We observed that the quasi-linear methods always converge within five iterations.

6.2. Real data

We tested our algorithms on several image sequences. For two of them, we show results. We compared methods LIN, QLIN1, QLIN2, and MLE, since MLE_HARTLEY is for three views only.

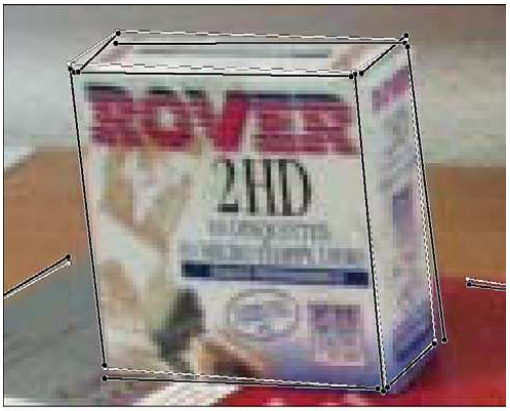
We observed that QLIN1 generally needs more iterations to converge than QLIN2. This is due to the Plücker correction step that significantly modifies the estimate

in QLIN1, while in QLIN2, since the constraint is linearized and enforced in the estimation, the correction applied to the estimate is less important.

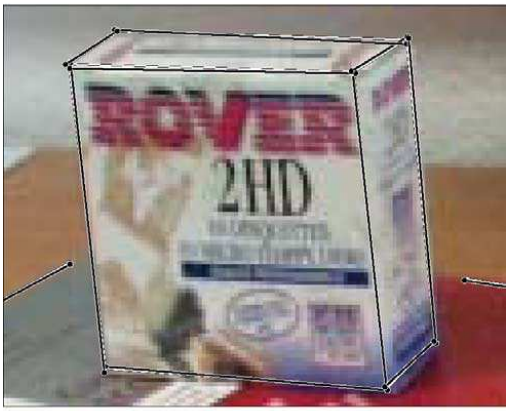
6.2.1. The 'books' sequence

Fig. 3 shows images from this 5-frame sequence. We provided 45 line correspondences by hand. Note that some of them are visible in two views only. We used these line correspondences to compute the trifocal tensor corresponding to each subsequence formed by triplets of consecutive images, using the linear method described in e.g. [11, Section 15.2]. We used method QLIN2 to reconstruct the lines associated with each triplet. We registered these subsequences by using the method given in [2]. At this point, we had a suboptimal guess of metric structure and motion. We further refined it using our triangulation algorithms, to reconstruct each line by taking into account all of its images. The corresponding estimation errors are, respectively

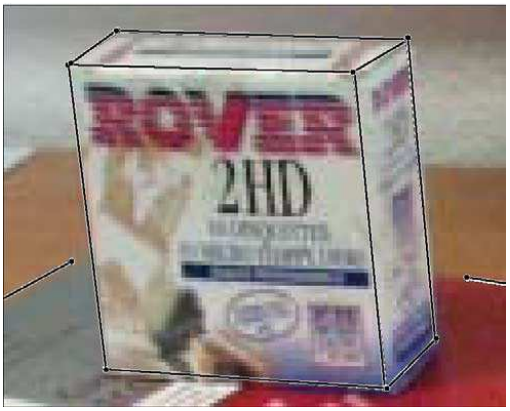
Fig. 3. Sample images out of the 5-frame 'books' sequence overlaid with manually provided lines. Note that the optical distortion is not corrected.



LIN & QLIN1



QLIN2



MLE

Fig. 4. Zoom on some original (white) and reprojected lines (black) for the 'books' sequence for different methods.

for LIN, QLIN1, and QLIN2, 2.30, 2.27, and 1.43 pixels. Note the significant improvement of QLIN2 compared to the biased methods LIN and QLIN1. Methods QLIN1 and QLIN2, respectively, took four and three iterations to converge.

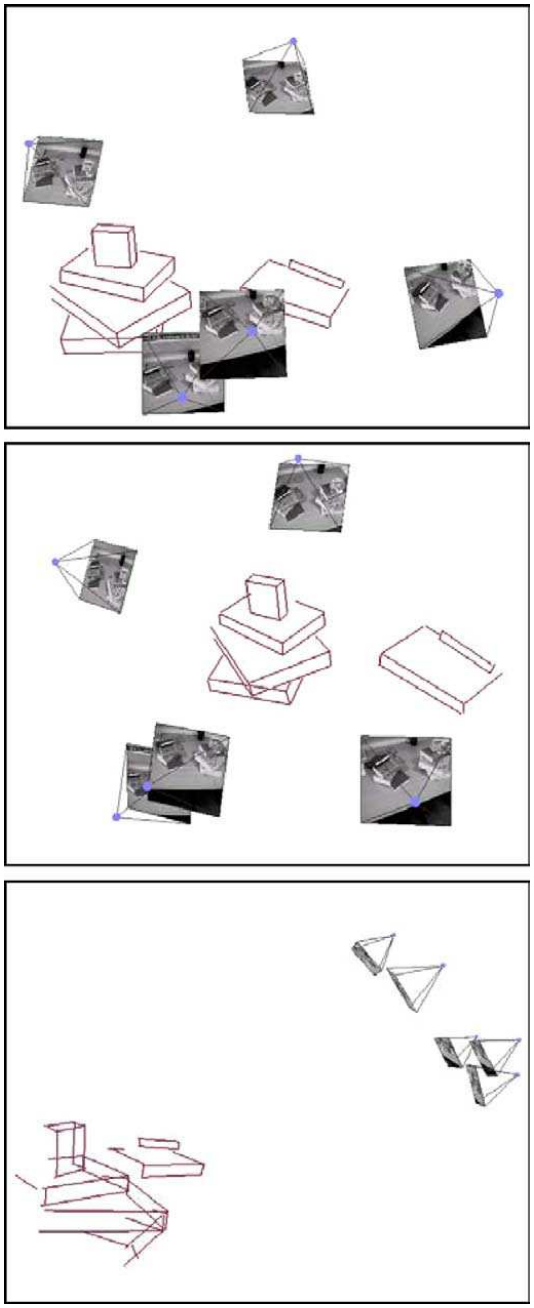


Fig. 5. Snapshots of the cameras and lines reconstructed by method MLE for the 'books' sequence. The images shown in Fig. 3 correspond to the top- and bottom-most cameras.

We used the result of QLIN2 to initialize our maximum likelihood estimator for structure and motion based on the proposed orthonormal representation together with a metric parameterization of the camera motion, which ends up with a 0.9 pixel estimation error.

For each estimation, we reconstructed the end-points corresponding to the first view (shown on the left of Fig. 3). The maximum likelihood end-points are given by orthogonally projecting their images onto the image of the corresponding line.

These results are visible on Fig. 4. Note the significant improvement of method MLE over methods LIN, QLIN1 and QLIN2. The lines predicted by MLE and the original lines are undistinguishable. Fig. 5 shows the cameras and lines reconstructed by MLE. There is visually no difference with the reconstruction provided by algorithm QLIN2, but that reconstructions provided by LIN and QLIN1 appear distorted.

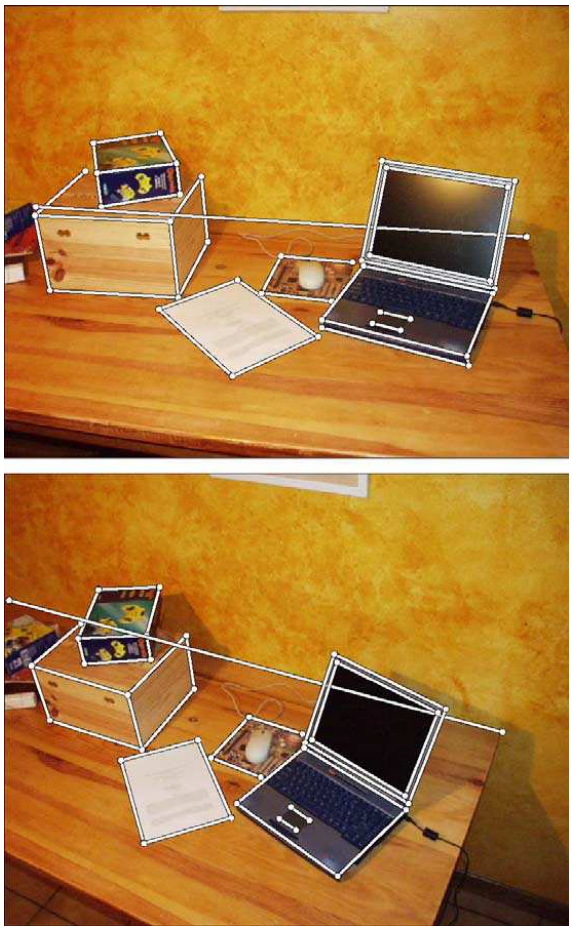


Fig. 6. Sample images out of the 8-frame 'laptop' sequence overlaid with manually provided lines. Note that the optical distortion is not corrected.

6.2.2. The 'laptop' sequence

Fig. 6 shows sample images for the 8-frame 'laptop' sequence, overlaid with the 40 manually entered line correspondences. We performed 3D reconstruction by apply-

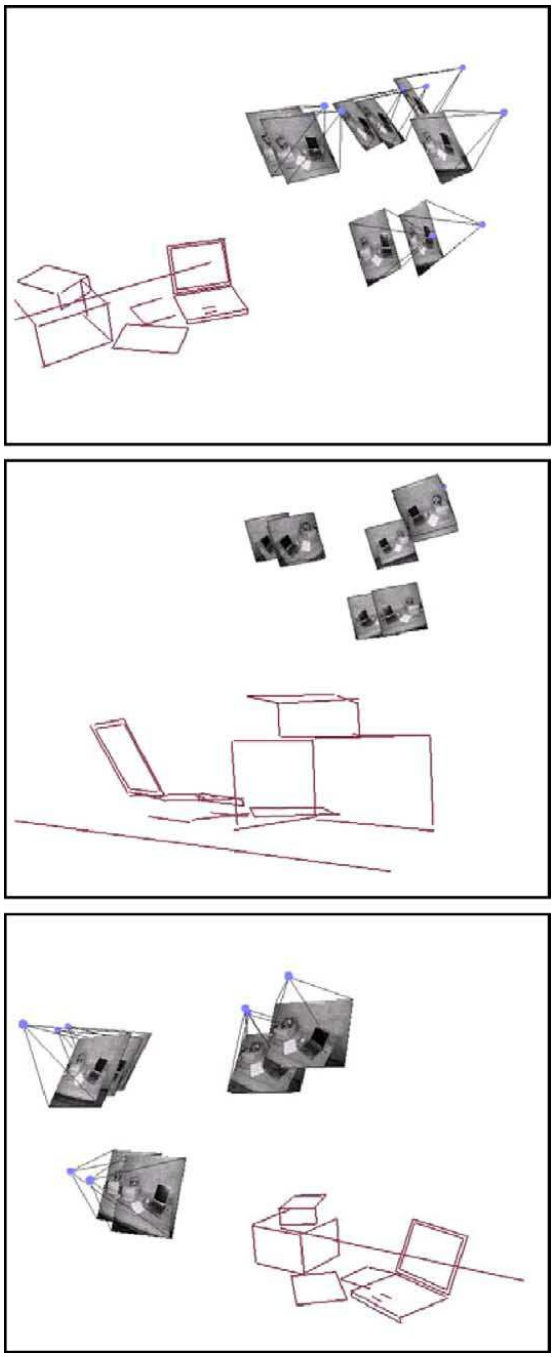


Fig. 7. Snapshots of the cameras and lines reconstructed by method MLE for the 'laptop' sequence.

ing the same algorithms as for the 'books' sequence. We obtained the following estimation errors for the triangulation algorithms, namely LIN: 1.34 pixels, QLIN1: 1.29 pixels, and QLIN2: 1.04 pixels. Methods QLIN1 and QLIN2 took respectively seven and five iterations to converge. For the bundle adjustment algorithms, we obtained an estimation error of 0.82 pixels. Fig. 7 shows snapshots of the reconstructed 3D models.

These results show that accurate reconstructed models can be obtained on real images taken by amateur digital cameras. They also show the importance of running a final bundle adjustment after initial triangulation.

7. Conclusion

We addressed the problem of structure and motion recovery from line correspondences across multiple views.

First, we proposed an optimal triangulation algorithm. Given camera motion, the Plücker coordinates of the 3D lines are estimated by minimizing the reprojection error. The algorithm relies on an iteratively reweighted least squares scheme. We linearized the bilinear Plücker constraint to incorporate it up to first order in the estimation process. A Plücker correction procedure is proposed to find the nearest Plücker coordinates to a given 6-vector.

Second, we proposed the orthonormal representation of 3D lines, which allows nonlinear optimization with the minimal four parameters within an unconstrained optimization engine, contrarily to previously proposed overparameterizations. This representation is well-conditioned and allows analytic differentiation.

Experimental results on simulated and real data show that the standard linear method and its naive bias-corrected extension perform very badly in many cases and should only be used to initialize a nonlinear estimator. Our bias-corrected algorithm including the Plücker constraint performs as well as direct Levenberg–Marquardt-based triangulation. It is therefore a good solution to initialize subsequent bundle adjustment. Based on our orthonormal representation, bundle adjustment gives results close to the theoretical lower bound and undistinguishable from the three-view maximum likelihood estimator of [11, Section 15.4.1], while being usable with any number of views.

References

[1] A. Bartoli, On the nonlinear optimization of projective motion using minimal parameters, in: Proc. 7th Eur. Conf. on Computer Vision, vol. 2, Copenhagen, Denmark, 2002, pp. 340–354.

[2] A. Bartoli, P. Sturm, The 3D line motion matrix and alignment of line reconstructions, Internat. J. Comput. Vision 57 (3) (2004).

[3] J. Denavit, R.S. Hartenberg, A kinematic notation for lower pair mechanisms based on matrices, ASME J. Appl. Mech. 22 (1955) 215–221.

[4] N. Ayache et Faugeras, Maintaining representations of the environment of a mobile robot, IEEE Trans. Robotics Autom. 5 (6) (1989) 804–819.

[5] O. Faugeras, B. Mourrain, On the geometry and algebra of the point and line correspondences between *n* images, in: Proc. 5th Internat. Conf. on Computer Vision, Cambridge, Massachusetts, USA, 1995, pp. 951–956.
[6] A.W. Fitzgibbon, A. Zisserman, Automatic camera recovery for closed or open image sequences, in: Eur. Conf. on Computer Vision, 1998, pp. 311–326.
[7] G.H. Golub, C.F. van Loan, Matrix Computation, The Johns Hopkins University Press, Baltimore, 1989.
[8] A. Habib, Motion parameter estimation by tracking stationary three-dimensional straight lines in image sequences, Internat. Arch. Photogramm. Remote Sensing 53 (1998).
[9] A. Habib, M. Morgan, Y.-R. Lee, Bundle adjustement with self-calibration using straight lines, Photogramm. Rec. (2002).
[10] R.I. Hartley, Lines and points in three views and the trifocal tensor, Internat. J. Comput. Vision 22 (2) (1997) 125–140.
[11] R.I. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, Cambridge University Press, Cambridge, 2000.
[12] S.A. Hayati, M. Mirmirani, Improving the absolute positioning accuracy of robot manipulators, J. Robotic Syst. 2 (4) (1985) 397–441.
[13] Kanatani K, Statistical Optimisation for Geometric Computation: Theory and Practice, Elsevier Science, Amsterdam, 1996.
[14] Y. Liu, T.S. Huang, A linear algorithm for motion estimation using straight line correspondences, Comput. Vision Graph. Image Process. 44 (1) (1988) 35–57.
[15] D. Martinec, T. Pajdla. Line reconstruction from many perspective images by factorization, in: Proc. Conf. on Computer Vision and Pattern Recognition, vol. I, Madison, Wisconsin, USA, IEEE Computer Society Press, 2003, pp. 497–502.
[16] D.C. Mulawa, E.M. Mikhail, Photogrammetric treatment of linear features, Internat. Arch. Photogramm. Remote Sensing 27 (1988) 383–393.
[17] H. Pottmann, M. Hofer, B. Odehnal, J. Wallner. Line geometry for 3D shape understanding and reconstruction, in: Proc. Eur. Conf. on Computer Vision, 2004.
[18] L. Quan, T. Kanade, Affine structure from line correspondences with uncalibrated affine cameras, IEEE Trans. Pattern Anal. Mach. Intell. 19 (8) (1997) 834–845.
[19] K. Roberts, A new representation for a line, in: Proc. Conf. on Computer Vision and Pattern Recognition, San Diego, California, USA, 1988, pp. 635–640.
[20] Y. Seo, K.S. Hong, Sequential reconstruction of lines in projective space, in: Proc. 13th Internat. Conf. on Pattern Recognition, Vienna, Austria, 1996, pp. 503–507.
[21] M. Spetsakis, J. Aloimonos, Structure from motion using line correspondences, Internat. J. Comput. Vision 4 (1990) 171–183.
[22] G.P. Stein, A. Shashua, On degeneracy of linear reconstruction from three views: linear line complex and applications, IEEE Trans. Pattern Anal. Mach. Intell. 21 (3) (1999) 244–251.
[23] C.J. Taylor, D.J. Kriegman, Structure and motion from line segments in multiple images, IEEE Trans. Pattern Anal. Mach. Intell. 17 (11) (1995) 1021–1032.
[24] A. Tommaselli, J. Lugnani, An alternative mathematical model to collinearity equations using straight features, Internat. Arch. Photogramm. Remote Sensing 27 (1998) 765–774.
[25] B. Triggs, Factorization methods for projective structure and motion, in: Proc. Conf. on Computer Vision and Pattern Recognition, San Francisco, California, USA, 1996, pp. 845–851.
[26] T. Viéville, Q.T. Luong, O.D. Faugeras, Motion of points and lines in the uncalibrated case, Internat. J. Comput. Vision 17 (1) (1995).
[27] J. Weng, T.S. Huang, N. Ahuja, Motion and structure from line correspondences: closed-form solution, uniqueness, and optimization, IEEE Trans. Pattern Anal. Mach. Intell. 14 (3) (1992) 318–336.

A Method for Interactive 3D Reconstruction of Piecewise Planar Objects from Single Images

Peter F Sturm* and Stephen J Maybank
Computational Vision Group, Department of Computer Science
The University of Reading, Whiteknights, PO Box 225
Reading, RG6 6AY, United Kingdom
{P.F.Sturm,S.J.Maybank}@reading.ac.uk

Abstract

We present an approach for 3D reconstruction of objects from a single image. Obviously, constraints on the 3D structure are needed to perform this task. Our approach is based on user-provided coplanarity, perpendicularity and parallelism constraints. These are used to calibrate the image and perform 3D reconstruction. The method is described in detail and results are provided.

1 Introduction

Methods for 3D reconstruction from images abound in the literature. A lot of effort has been spent on the development of multi-view approaches allowing for high accuracy and complete modeling of complex scenes. On one hand, research is directed towards completely automatic systems; these are relatively difficult to realize and it is not clear yet if they are ready for use by a non expert. On the other hand, commercial systems exist, but they usually require a high amount of user interaction (clicking on many points in many images) or a special camera setup (e.g. using structured light).

The guideline of the work described here is to provide an intermediate solution, reconstruction from a single image, that needs relatively little user interaction. Naturally, there are limits on the kind of objects possible to be reconstructed and on the achievable degree of completeness of reconstructions. However, our results suggest that such a minimal solution for reconstruction might be quite useful, e.g. for visualization and augmented reality purposes.

Work on reconstruction from single images has been done by others. Shum et al. describe a method similar to ours in [6]. Their method, however, allows to reconstruct only planes whose vanishing lines can be computed from two or more sets of parallel lines, whereas our method can also reconstruct arbitrary planes, thus leading to a wider class of objects that may be reconstructed. Liebowitz et al. describe two different methods for single-view 3D reconstruction in [5]. The first method achieves the reconstruction by measuring heights of points with respect to a ground plane. The drawback of the method is the requirement of the foot point for each 3D point to be reconstructed, i.e. the image of the vertical intersection with the ground plane. This puts a limit to the nature of objects that may be reconstructed. The second method of Liebowitz et al. requires, like the method by Shum et al., the computation of the vanishing lines of all planes to be reconstructed. Their method also appears to be less straightforward than the one we describe in this paper, e.g. it performs intermediate rectifications of the images of planar patches that might perhaps be omitted.

Reconstruction from single images requires geometrical constraints on the 3D structure of the observed object. The approach described in this paper is based on three types of constraints: coplanarity of points, perpendicularity of directions or planes and parallelism of directions or planes. Perpendicularity constraints are used to calibrate the image. Together with parallelism constraints they provide the vanishing geometry of the scene which forms the skeleton of the 3D reconstruction. Coplanarity constraints are used to complete the reconstruction, via alternating reconstruction of points and planes.

The paper is organized as follows. In §2, we describe our camera model and the computation of vanishing points and lines. Details on calibration and 3D reconstruction are given in §§3 and 4 respectively. The complete algorithm is summarized in §5. §6 gives an example of how the algorithm works and presents some results. Conclusions are given in §7.

*This work is supported by the EPSRC funded project GR/K89221 (Vector).

2 Background

2.1 Camera Model

We use perspective projection to model cameras. A projection may be represented by a 3×4 projection matrix P that maps points of 3-space to points in 2-space: $\mathbf{q} \sim \mathsf{P}\mathbf{Q}$. Here, $\sim$ means equality up to a non zero scale factor, which accounts for the use of homogeneous coordinates. Since we consider a single view and may chose the 3D reference frame arbitrarily, we align it with the camera, leading to the simple projection matrix $\mathsf{P} \sim (\ \mathsf{K}\ |\mathbf{0})$. Here, K is the 3×3 calibration matrix:

$$\mathsf{K} = \begin{pmatrix} \tau f & s & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{pmatrix} \ .$$

In general, we distinguish 5 intrinsic parameters for the perspective projection model: the (effective) focal length f, the aspect ratio τ, the principal point (u_0, v_0) and the skew factor s accounting for non rectangular pixels. The skew factor is usually very close to 0 and we ignore it in the following.

2.2 Vanishing Points and Lines

We compute vanishing points as the least squares solution for the intersection of sets of images of parallel 3D line segments. The information of line segments being parallel is provided by the user.

Vanishing lines are determined from vanishing points and parallelism constraints. The assumption that a vanishing point $\mathbf{v}$ belongs to a 3D direction parallel to a plane implies that $\mathbf{v}$ lies on the vanishing line $\mathbf{l}$ of that plane. Hence, two or more vanishing points parallel to a plane define its vanishing line.

A vanishing point $\mathbf{v}$ that belongs to the 3D direction *perpendicular* to a plane, completely defines the vanishing line (if the image is calibrated): $\mathbf{l} \sim \mathsf{K}^{-\mathsf{T}}\mathsf{K}^{-1}\mathbf{v}$.

3 Calibration

In the following, we derive calibration equations that are based on vanishing points of pairs of perpendicular directions. This approach is well known (cf. e.g. [1]); we briefly describe it and give then a closed-form solution for the focal length which is usually the only intrinsic parameter that we calibrate in practice.

Let $\mathbf{v}_1$ and $\mathbf{v}_2$ be the vanishing points of two perpendicular 3D directions. Let the ideal points of the 3D directions be written as: $(\mathbf{V}_1^\mathsf{T}, 0)^\mathsf{T}$ and $(\mathbf{V}_2^\mathsf{T}, 0)^\mathsf{T}$. From the projection equations $\mathbf{v}_1 \sim \mathsf{K}\mathbf{V}_1$ and $\mathbf{v}_2 \sim \mathsf{K}\mathbf{V}_2$ we compute the ideal points as:

$$\begin{aligned} \mathbf{V}_1 &\sim \mathsf{K}^{-1}\mathbf{v}_1 \\ \mathbf{V}_2 &\sim \mathsf{K}^{-1}\mathbf{v}_2 \ . \end{aligned}$$

The 3D directions being perpendicular means that $\mathbf{V}_1^\mathsf{T}\mathbf{V}_2 = 0$, hence:

$$\mathbf{v}_1^\mathsf{T}\mathsf{K}^{-\mathsf{T}}\mathsf{K}^{-1}\mathbf{v}_2 = 0 \ . \tag{1}$$

This equation is homogeneous linear in the coefficients of the symmetric matrix $\omega \sim \mathsf{K}^{-\mathsf{T}}\mathsf{K}^{-1}$ (which represents the image of the Absolute Conic). Having determined ω, using equations (1) or other means, the calibration matrix K may be computed uniquely using Cholesky decomposition [4].

Each pair of perpendicular vanishing points gives one constraint on the intrinsic parameters in K. In a man-made environment, we will typically observe three pairs of mutually perpendicular vanishing points, sometimes more, sometimes only a single pair. This puts a limit on the number of intrinsic parameters that may be computed. The aspect ratio τ can often be assumed to be known. Depending on the number of calibration equations, we may estimate the principal point and the focal length. For the experiments described later, we assumed that the principal point is in the center of the image, and only estimated the focal length.

The equations for the focal length are particularly simple. We may decompose the calibration matrix in its known and unknown parts:

$$\mathsf{K} = \mathsf{K}_1\mathsf{K}_2 = \begin{pmatrix} \tau & 0 & u_0 \\ 0 & 1 & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix} \ .$$

Transforming the vanishing points by K_1:

$$\mathbf{v}'_p \sim \mathsf{K}_1^{-1}\mathbf{v}_p$$

we obtain points $\mathbf{v}'_p$ for which the calibration equation (1) takes on the simple form:

$$(\mathbf{v}'_1)^\mathsf{T} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & f^2 \end{pmatrix} \mathbf{v}'_2 = 0 \ . \tag{2}$$

The least squares solution for a set of equations (2) is given by:

$$f^2 = -\frac{\sum_{\mathbf{v}'_p \perp \mathbf{v}'_q} v'_{p,3} v'_{q,3} (v'_{p,1} v'_{q,1} + v'_{p,2} v'_{q,2})}{\sum_{\mathbf{v}'_p \perp \mathbf{v}'_q} (v'_{p,3} v'_{q,3})^2} \ .$$

Before solving for f, the $\mathbf{v}'_p$ should be normalized to unit norm.

What we observe is that vanishing points that lie on the ideal line in the image are useless for focal length calibration (the term $v'_{p,3} v'_{q,3}$ in the denominator is zero here). This means that the vanishing points that correspond to directions that are parallel to the image plane, are useless, so we need at least two finite vanishing points that are perpendicular. These considerations tell us how to position the camera to successfully calibrate it. Note that vanishing points at infinity will not badly influence the determination of the focal length: the term in the denominator will be zero and the term in the numerator very close to zero, so they won't affect the sums in the equation. Infinite vanishing points might be used for the determination of other intrinsic parameters, if required.

The images used in our experiments (see figures 3 and 4) only show small amounts of optical distortion. However, wide-angle views which are likely to be used for single-view 3D reconstruction, might require distortion removal prior to calibration and reconstruction. The automatic method by Devernay and Faugeras [2] might be used. Distortion removal can also be achieved in a very simple way by manually adjusting the dominant first coefficient of radial distortion, by the aid of a slider provided by the graphical user interface, such as to make line segments in the image roughly straight.

4 3D Reconstruction

The principal aim here is to reconstruct a set of 3D points and planes. Sets of coplanar 3D points define polygons onto which texture can be mapped for visualization purposes.

We assume that vanishing points and lines have been computed where possible and that the image has been calibrated as described in the previous section. This enables us to backproject image points to 3D along their projection rays – a 3D point whose image is given by $\mathbf{q}$, has coordinates:

$$\mathbf{Q} = \begin{pmatrix} \lambda \mathbf{q}' \\ 1 \end{pmatrix} \ , \tag{3}$$

where $\mathbf{q}' \sim \mathsf{K}^{-1}\mathbf{q}$ and $\mathbf{q}'$ has unit norm. The unknown λ expresses the distance of $\mathbf{Q}$ from the optical center and hence defines its position on the projection ray.

If we know the vanishing line of a 3D plane, its position is also defined up to one unknown. Let $\mathbf{l}$ be the vanishing line and $\mathbf{n} \sim \mathsf{K}^\mathsf{T}\mathbf{l}$ such that $\mathbf{n}$ has unit norm. Then, the 3D position of any plane whose vanishing line is $\mathbf{l}$, is given by:

$$\Pi = \begin{pmatrix} \mathbf{n} \\ d \end{pmatrix} \ . \tag{4}$$

The vector $\mathbf{n}$ is the plane's normal and d the plane's distance from the optical center.

Unless a reference distance in the scene is known, 3D reconstruction can be achieved up to a global scale factor only. Hence, we are free to set the position of one point (along its projection ray) or one plane (while preserving its vanishing line). Suppose, we have fixed one point $\mathbf{Q}$. The position of planes with known vanishing lines and containing $\mathbf{Q}$ is then completely defined. Other points lying on these planes may then be reconstructed, by simply intersecting the projection rays with the planes. In turn, other planes may then be reconstructed using the available points, and so on. This alternation scheme allows to reconstruct objects whose parts are sufficiently "interconnected", i.e. the points on the object have to be linked together via coplanarity or other geometrical constraints.

In the following, we describe an extension of this basic reconstruction scheme. Basically, we bootstrap the alternating point-plane reconstruction scheme via the simultaneous reconstruction of a set of points and a set of planes that are linked together in a way described below. This is the central part of our reconstruction method. Other modules used for reconstruction are described in §4.2. The complete algorithm is given in §5 and the way it works is illustrated in §6.

4.1 Simultaneous Reconstruction of Points and Planes

The coplanarity constraints provided by the user are in general overconstrained, i.e. several points may lie on more than one plane. This means that, due to image noise, it is difficult to obtain a 3D reconstruction that satisfies all constraints exactly. This may be achieved by constrained optimization, but there might be no batch method of doing so. Thus, in

the following we describe a direct least squares solution for reconstructing a subset of object planes and points, minimizing the sum of squared distances between planes and points. Usually, the subsets of planes and points that may be reconstructed this way cover already a large part of the object (cf. the example in §6).

Consider sets of images of coplanar points, $S_r = \{\mathbf{q}_{i_{r,1}}, \ldots, \mathbf{q}_{i_{r,n_r}}\}$. A point may belong to more than one set S_r. Let Π_r be the plane corresponding to the set S_r. In the following, we only consider planes with known vanishing lines.

We say that two planes Π_{r_1} and Π_{r_2} are connected if they share a point, i.e. if the intersection of S_{r_1} and S_{r_2} is non empty. This relationship may be visualized by a graph, whose vertices are planes, with edges being drawn between connected planes. We choose a largest subgraph of connected planes (full connection is not required). Let S'_r be the point sets of the selected planes, points lying on one plane only having been eliminated.

We now show how the considered planes and points may be reconstructed simultaneously in a least squares manner. Reconstruction is done via the determination of the scalars λ and d, as given in equations (3) and (4). Let $\mathbf{Q}$ be a point lying on plane Π. The squared distance between them is given by:

$$\left(d + (\mathbf{n}^\mathsf{T}\mathbf{q}')\lambda\right)^2 \quad .$$

We want to minimize the sum of squared distances for pairs of planes and points. The cost function is thus:

$$g = \sum_r \sum_{p=1}^{n_r} \left(d_r^2 + 2\left((\mathbf{n}_r)^\mathsf{T}\mathbf{q}'_{i_{rp}}\right)\lambda_{i_{rp}} d_r + \left((\mathbf{n}_r)^\mathsf{T}\mathbf{q}'_{i_{rp}}\right)^2 \lambda_{i_{rp}}^2 \right) \quad .$$

Its partial derivatives are (divided by 2):

$$\frac{\sigma g}{\sigma d_r} = \left(\sum_{p=1}^{n_r} 1\right) d_r + \sum_{p=1}^{n_r} \left((\mathbf{n}_r)^\mathsf{T}\mathbf{q}'_{i_{rp}}\right)\lambda_{i_{rp}}$$

$$\frac{\sigma g}{\sigma \lambda_p} = \sum_{r, \mathbf{q}_p \in S'_r} \left((\mathbf{n}_r)^\mathsf{T}\mathbf{q}'_p\right) d_r + \left(\sum_{r, \mathbf{q}_p \in S'_r} \left((\mathbf{n}_r)^\mathsf{T}\mathbf{q}'_p\right)^2 \right) \lambda_p$$

Nullifying these leads to a homogeneous linear equation system in the unknowns d_r and λ_p, giving the least squares solution. The solution is defined up to scale, as expected, since reconstruction can only be done up to scale.

The equation system has the following nice structure:

$$\left(\begin{array}{cccc|cccc} D_1 & & & & C_{11} & C_{12} & \cdots & C_{1n} \\ & D_2 & & & C_{21} & C_{22} & \cdots & C_{2n} \\ & & \ddots & & \vdots & \vdots & & \vdots \\ & & & D_m & C_{m1} & C_{m2} & \cdots & C_{mn} \\ \hline C_{11} & C_{21} & \cdots & C_{m1} & L_1 & & & \\ C_{12} & C_{22} & \cdots & C_{m2} & & L_2 & & \\ \vdots & \vdots & & \vdots & & & \ddots & \\ C_{1n} & C_{2n} & \cdots & C_{mn} & & & & L_n \end{array}\right) \left(\begin{array}{c} d_1 \\ d_2 \\ \vdots \\ d_m \\ \hline \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_n \end{array}\right) = \left(\begin{array}{c} 0 \\ 0 \\ \vdots \\ 0 \\ \hline 0 \\ 0 \\ \vdots \\ 0 \end{array}\right)$$

where:

$$D_r = \sum_{p=1}^{n_r} 1 \qquad C_{rp} = (\mathbf{n}_r)^\mathsf{T}\mathbf{q}'_p \qquad L_p = \sum_{r, \mathbf{q}_p \in S'_r} \left((\mathbf{n}_r)^\mathsf{T}\mathbf{q}'_p\right)^2$$

Special sparse solution methods may be used like e.g. in [3], but for small problems (the size of the matrix is the number of planes plus the number of points, which is usually at most a few dozens for single images) we simply use singular value decomposition.

4.2 Other Modules

Our method requires basically two other reconstruction modules, the backprojection of a point onto a 3D plane and the fitting of a plane to a set of 3D points, possibly including ideal points.

Backprojecting a point onto a plane. Backprojection of a point onto a plane is done by computing λ_p via:

$$\lambda_p = -\frac{d_r}{(\mathbf{n}_r)^\mathsf{T}\mathbf{q}'_p} \quad .$$

Fitting a plane to a set of points. Several cases may be considered. In the general case, the cost function to be minimized is the sum of squared distances (we omit here indices referring to the plane):

$$g = \sum_{p=1}^{n} \left(d^2 + 2 \left(\mathbf{n}^\mathsf{T} \mathbf{q}'_p \right) \lambda_p d + \left(\mathbf{n}^\mathsf{T} \mathbf{q}'_p \right)^2 \lambda_p^2 \right) \quad . \tag{5}$$

Nullifying the partial derivatives leads to a linear homogeneous equation system in the unknowns d and $\mathbf{n}$. If we already know the plane's normal $\mathbf{n}$, we obtain the following closed form solution for the unknown d:

$$d = -\frac{\sum_{p=1}^{n} \left(\mathbf{n}^\mathsf{T} \mathbf{q}'_p \right)}{\sum_{p=1}^{n} 1} \quad .$$

5 Complete Algorithm

1. Compute vanishing points and lines (cf. §2.2).
2. Calibrate (cf. §3).
3. Backproject all points up to scale, i.e. compute the vectors $\mathbf{q}'_p \sim \mathsf{K}^{-1}\mathbf{q}_p$. Scale the $\mathbf{q}'_p$ to unit norm and use extended coordinates for 3D points: $\mathbf{Q}_p^\mathsf{T} = (\lambda_p (\mathbf{q}'_p)^\mathsf{T}, 1)$.
4. From vanishing lines, compute plane normals (cf. equation (4)).
5. Partition the planes with known normal in sets of planes which are connected by at least one point (in a transitive manner).
6. Choose the largest partition.
7. Reconstruct plane and point positions (distances from origin) as described in §4.1. Use only points that lie on more than one plane in the actual partition.
8. Backproject points that lie on exactly one plane in the actual partition (cf. §4.2).
9. Reconstruct a plane not reconstructed yet by fitting it to 3D points (cf. §4.2). Each point provides one equation and a vanishing line two. Choose the plane with the most equations.
10. Backproject points lying on the plane just reconstructed.
11. If there are planes not reconstructed yet, go to step 9.

We may optimize the reconstruction, respecting the geometric constraints. We have coded such a bundle adjustment procedure, but in practice there is virtually no improvement in the quality of the reconstruction, so we usually omit this step. From the 3D reconstruction, we automatically create textured VRML models (see examples in §6).

6 Sample Run and Examples of 3D Models

Figure 1 explains the user-provided input to our algorithm for the example shown in figure 3. On the left hand side, the different directions present in the 3D object are represented via the dotted line segments which are used for computing the vanishing points. Additionally, the user should flag perpendicular directions. On the right hand side, 5 groups of parallel planar patches are shown, i.e. groups of patches sharing the same vanishing line. The edges in the middle of the graph show which directions "belong" to which groups of planes. For example, for each of the second, third and fifth groups of planes, we have two vanishing points, allowing us to compute the vanishing lines.

Some of the steps taken by the reconstruction algorithm for our example are shown in figure 2. The upper left figure shows the result of the initial step described in §4.1. Note that a large part of the object is already reconstructed. The upper right figure shows points that are backprojected onto the reconstructed planes (cf. step 8 of the algorithm). The subsequent rows of figures show on the left a reconstructed plane (dark) and the (bold) points used to reconstruct it (step 9 of the algorithm). On the right, backprojected points are shown using bright circles (step 10). The reconstruction of the whole 3D model for this example required 2 additional steps of alternating plane-point reconstruction, not shown here.

Figures 3 and 4 on page 8 show examples of 3D models obtained with our method. The first image in each figure is the original image from which reconstruction was obtained. The other images show rendered views of created VRML models. Texture maps were taken from the original images. For the model shown in figure 4, additional texture maps, taken from frontoparallel views of two of the walls were used to enhance resolution.

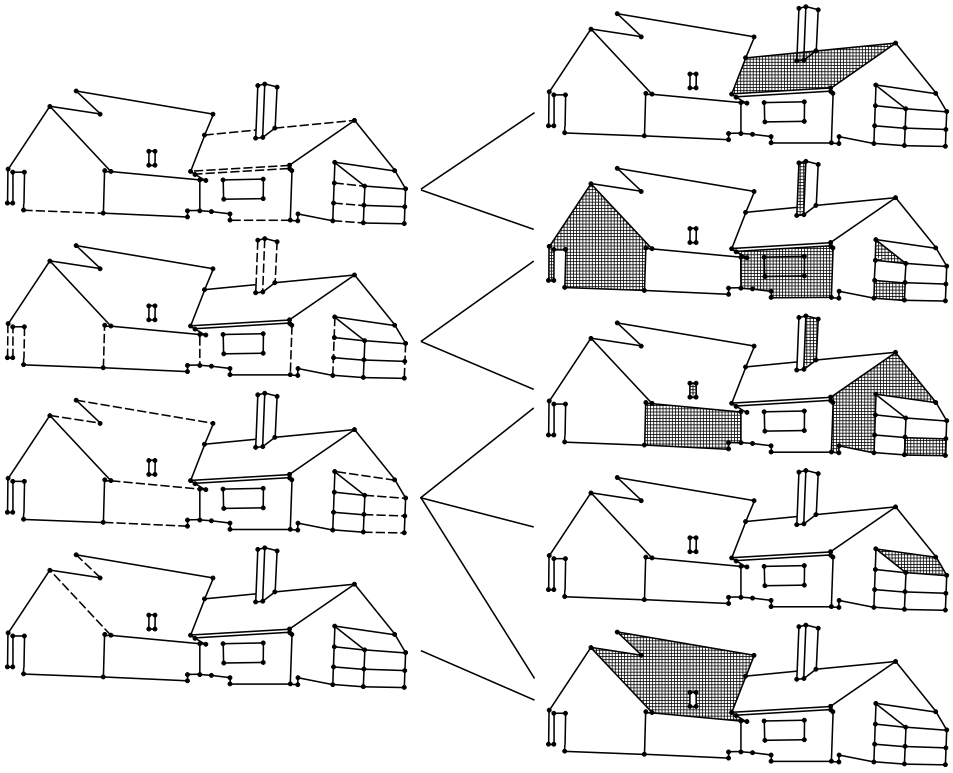


Figure 1: User-provided input: directions of lines and planes.

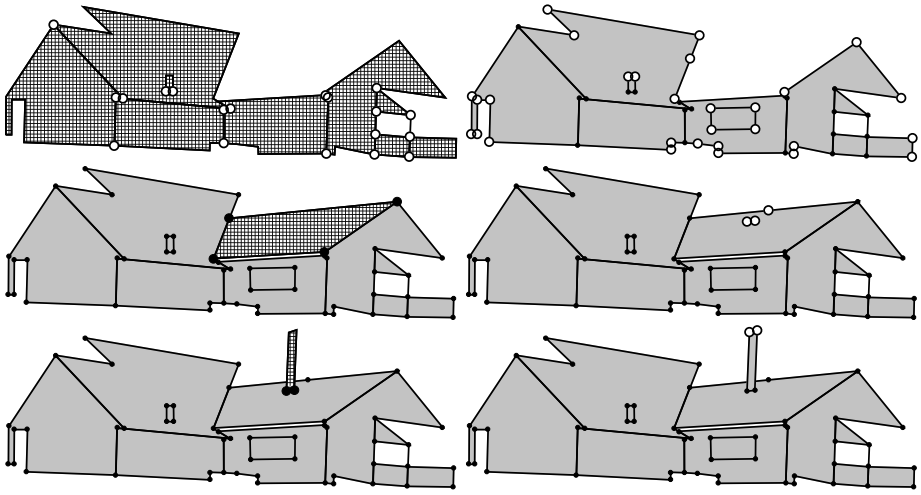


Figure 2: First 6 steps of the reconstruction process.

7 Conclusion

We have presented a method for interactive 3D reconstruction of piecewise planar objects from a single view. Camera calibration and 3D reconstruction are done using geometrical constraints provided by the user, that are simple in nature (coplanarity, perpendicularity and parallelism) and may be easily provided without any computer vision expertise.

The major drawback of single-view 3D reconstruction is of course that only limited classes of objects may be reconstructed and that the reconstruction is usually incomplete. The major advantages however are that it is a quick way of obtaining 3D models, that it is rather easy to implement and to use and that due to user interaction and the small size of the problem the reconstruction process becomes very reliable, compared to more automatic multi-view systems.

One advantage of our method compared to other approaches is that a wider class of objects can be reconstructed (especially, there is no requirement of disposing of two or more vanishing points for each plane). The simultaneous reconstruction of several planes and several points that forms the starting point of our method makes it likely that errors are nicely spread over the whole 3D model, compared to more sequential approaches like [5].

There are several extensions that may be added to our basic method. For example, other primitives than points and planes might be used, like lines, spheres or cylinders. Other types of geometrical constraints, like e.g. ratios of distances, can be added, enlargening the class of objects that can be reconstructed. Also, it might be worth trying to stitch together two or more 3D models obtained from single, possibly non-overlapping views (e.g. from the back and the front of a house), to get complete 3D models.

We already adapted our method to the use of panoramic images, obtained using a parabolic mirror. Thus, we are able to create 360° 3D models from one image, usually of the interior of a room.

Please contact the first author to get hard copies with color figures and images.

References

[1] B. Caprile and V. Torre, "Using Vanishing Points for Camera Calibration," *International Journal on Computer Vision*, Vol. 4, pp. 127–140, 1990.

[2] F. Devernay and O.D. Faugeras, "Automatic calibration and removal of distortion from scenes of structured environments," *Proceedings of the* SPIE *Conference on Investigate and Trial Image Processing, San Diego, California, USA*, Vol. 2567, SPIE - Society of Photo-Optical Instrumentation Engineers, August 1995.

[3] R.I. Hartley, "Euclidean Reconstruction from Uncalibrated Views," *Proceeding of the* DARPA–ESPRIT *Workshop on Applications of Invariants in Computer Vision, Azores, Portugal*, pp. 187-202, October 1993.

[4] A. Jennings, J.J. McKeown, *Matrix Computation,* 2nd edition, Wiley, 1992.

[5] D. Liebowitz, A. Criminisi and A. Zisserman, "Creating Architectural Models from Images," *Proceedings EuroGraphics*, to appear, September 1999.

[6] H.-Y. Shum, R. Szeliski, S. Baker, M. Han, P. Anandan, "Interactive 3D Modeling from Multiple Images Using Scene Regularities," *SMILE Workshop, Freiburg, Germany*, pp. 236-252, June 1998.

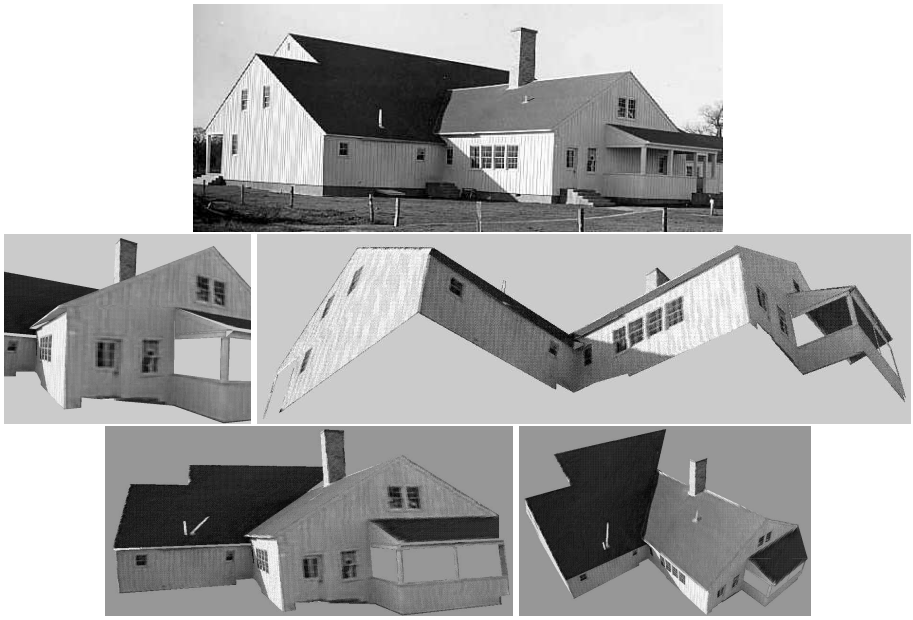


Figure 3: Original image and rendered views of 3D VRML model.

Figure 4: Original image and rendered views of 3D VRML model.

A Method for 3D Reconstruction of Piecewise Planar Objects from Single Panoramic Images

Peter Sturm*
INRIA Rhône-Alpes
655 Avenue de l'Europe
38330 Montbonnot St Martin, France
Peter.Sturm@inrialpes.fr

Abstract

We present an approach for 3D reconstruction of objects from a single panoramic image. Obviously, constraints on the 3D structure are needed to perform this task. Our approach is based on user-provided coplanarity, perpendicularity and parallelism constraints. The method is described in detail for the case of a parabolic mirror-based omnidirectional sensor and results are provided.

1. Introduction

Methods for 3D reconstruction from images abound in the literature. A lot of effort has been spent on the development of multi-view approaches allowing for high accuracy and complete modeling of complex scenes. On one hand, research is directed towards completely automatic systems; these are relatively difficult to realize and it is not clear yet if they are ready for use by a non expert. On the other hand, commercial systems exist, but they usually require a high amount of user interaction (clicking on many points in many images).

The guideline of the work described here is to provide an intermediate solution, reconstruction from a single image, that needs relatively little user interaction. Naturally, there are limits on the kind of objects possible to be reconstructed and on the achievable degree of completeness of reconstructions.

Work on reconstruction from single images has been done before, see e.g. [9, 12, 13]. Most of the existing methods were developed for the use of a pinhole camera (with the exception of [12] where mosaics are used). The scope of several of these methods is limited, e.g. the approaches described in [9, 12] only allow to reconstruct planar surfaces whose vanishing line can be determined in the image. One of the two approaches in [9] achieves the reconstruction by measuring heights of points with respect to a ground plane. The drawback of the method is the requirement of the foot point for each 3D point to be reconstructed, i.e. the image of the vertical intersection with the ground plane.

In this paper, we present an approach for 3D reconstruction from a single panoramic image (work on panoramic stereo and ego-motion estimation is described in e.g. [3, 5, 7, 14]). The concrete example of an image acquired with a parabolic mirror-based omnidirectional camera is described, but the method is easily adapted to other omnidirectional sensors. Reconstruction from a single image requires a priori constraints on the 3D structure. We use constraints that are easy to provide: coplanarity of points, perpendicularity of planes and lines, and parallelism of planes and lines. The parallelism and perpendicularity constraints are used to estimate the "directional geometry" of the scene (line directions and plane normals) which forms the skeleton of the 3D reconstruction. Coplanarity constraints are used to complete the reconstruction, via simultaneous reconstruction of points and planes. With the type of information used we are able to reconstruct piecewise planar objects.

The paper is organized as follows. In §2, we describe the camera model. The input to our reconstruction scheme is explained in §3. The basic idea for 3D reconstruction from a single image is outlined in §4. Details on 3D reconstruction are given in §§5 and 6. The complete algorithm is summarized in §7. §8 shows an experimental result and conclusions are given in §9.

2. Camera Model

We use an omnidirectional camera formed by the combination of a parabolic mirror and an orthographic camera whose viewing direction is parallel to the mirror's axis [10].

*This work is partially supported by the EPSRC funded project GR/K89221 (Vector).

Orthographic projection can be obtained by using telecentric optics [15]. Geometrically speaking, the projection center of the orthographic camera coincides with the infinite one among the two focal points of the paraboloid. Given the image of a point and a small amount of calibration information described below, it is possible to determine the 3D direction of the line joining the original 3D point and the *finite* focal point of the paraboloid. The finite focal point acts as an effective optical center, relative to which correct perspective views of the scene can be created from the panoramic image [1, 11].

In the following, we give formulas needed for calibrating the system and for 3D reconstruction. These formulas are well known [10, 14], but presented here for the sake of completeness.

2.1. Representation of Mirror and Camera

The mirror is a rotationally symmetric paraboloid. Its shape is thus defined by a single parameter a. Without loss of generality, we may represent the paraboloid in usual quadric notation by the following symmetric matrix:

$$\Omega \sim \begin{pmatrix} 4a^2 & 0 & 0 & 0 \\ 0 & 4a^2 & 0 & 0 \\ 0 & 0 & 0 & -2a \\ 0 & 0 & -2a & -1 \end{pmatrix}$$

where $\sim$ means equality up to scale, which accounts for the use of homogeneous coordinates. The mirror's axis is the Z-axis and the finite focal point $\mathbf{F}$ is the coordinate origin, i.e. $\mathbf{F}^\mathsf{T} = (0, 0, 0, 1)$. The parameter a describes the mirror's "curvature".

The viewing direction of the orthographic camera is parallel to the Z-axis, thus the projection matrix can be written as (using homogeneous coordinates):

$$\mathsf{P} \sim \begin{pmatrix} b & 0 & 0 & x_0 \\ 0 & b & 0 & y_0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad .$$

The parameter b is the magnification factor of the orthographic projection. The coefficients x_0 and y_0 describe the relative position of the image plane and the mirror, perpendicular to the viewing direction.

2.2. Projection of a 3D Point

Let $\mathbf{Q}$ be a 3D point with coordinates $(X, Y, Z, 1)$. Its projection can be computed as follows. Let $\mathbf{L}$ be the line joining $\mathbf{Q}$ and the mirror's finite focal point $\mathbf{F}$. Among the two intersection points of $\mathbf{L}$ with the mirror Ω, choose the one which lies on the same half-line as $\mathbf{Q}$, with respect to $\mathbf{F}$. The image of $\mathbf{Q}$ is the orthographic projection of this intersection point, giving the image coordinates:

$$\begin{aligned} x &= x_0 + \frac{b}{2a} X \frac{Z + \sqrt{X^2 + Y^2 + Z^2}}{X^2 + Y^2} \\ y &= y_0 + \frac{b}{2a} Y \frac{Z + \sqrt{X^2 + Y^2 + Z^2}}{X^2 + Y^2} \quad . \end{aligned}$$

2.3. Calibration

The above projection equations show that the mirror's shape parameter a and the magnification b of the orthographic projection can be grouped together in a parameter $r = \frac{b}{2a}$ describing the combined system. To calibrate the system, we thus need to estimate the parameters x_0, y_0 and r. These parameters have a simple geometrical meaning: consider the horizontal circle on the paraboloid at the height of the focal point $\mathbf{F}$ (cf. figure 1). The projection of this circle in the orthographic image is exactly the circle ω with center (x_0, y_0) and radius r.

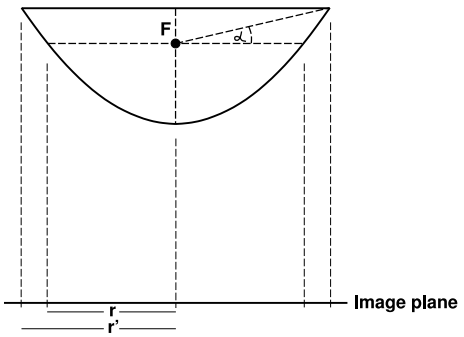


Figure 1. The paraboloidal mirror.

If the mirror's top border does not lie at the height of the focal point (e.g. as shown in figure 1), then we can not directly determine the circle ω in the image. Instead, we fit a circle ω' to the border of the image as shown in figure 2 (a). This circle is cocentric with ω, thus x_0 and y_0 are given by its center. The radius r' of ω', and r are related as follows:

$$r = r' \frac{\cos\alpha}{1 + \sin\alpha}$$

where α is the angle shown in figure 1, which is known by construction.

The calibration procedure has to be done only once for a fixed configuration. Another, more flexible calibration method, is described in [2].

2.4. Backprojection

The most important feature of our mirror-camera system is that from a panoramic image, we may create correct per-

spective images of the scene, as if they had been observed by a pinhole camera with optical center at $\mathbf{F}$.

This is equivalent to being able to backproject image points via $\mathbf{F}$: we are able to determine the projection ray $\mathbf{L}$ (cf. §2.2) of a point $\mathbf{Q}$, given its image point. Given the calibration parameters, the projection rays are determined in Euclidean space, which is useful for obtaining metric 3D reconstructions as described later.

A projection ray may be represented by its ideal point[1] which can be computed from the image coordinates (x, y) as follows:

$$\begin{pmatrix} \mathbf{B} \\ 0 \end{pmatrix} \sim \begin{pmatrix} 2r(x - x_0) \\ 2r(y - y_0) \\ (x - x_0)^2 + (y - y_0)^2 - r^2 \\ 0 \end{pmatrix} . \quad (1)$$

3. Input

To prepare the description of the 3D reconstruction method, we first explain the (user-provided) input. First of course, the system has to be calibrated, as described in §2.3 (also see figure 2 (a)). The basic primitives for our method are interest points (see figure 2 (b)). Based on interest points, coplanarity, parallelism and perpendicularity constraints are provided as follows.

Lines are defined by two or more interest points and they are grouped together into sets of mutually parallel lines (see figures 2 (c) and (d)). In §5.2 it is described how to compute the direction of a set of parallel lines. In the following, the direction of the ith set of parallel lines will be represented via the ideal point $\left(\mathbf{D}_i^\mathsf{T}, 0\right)^\mathsf{T}$.

Planes are also defined by interest points and grouped according to parallelism (see figures 2 (e) and (f)). The normal direction of a set of parallel planes can be computed as described in §5.3. The normal of the jth set of parallel planes will be represented via the 3-vector $\mathbf{n}_j$.

Other useful constraints are:

- parallelism of lines and planes, expressed as: $\mathbf{D}_i^\mathsf{T}\mathbf{n}_j = 0$.
- perpendicularity of lines and planes: $\mathbf{D}_i \sim \mathbf{n}_j$.
- perpendicularity of lines: $\mathbf{D}_{i_1}^\mathsf{T}\mathbf{D}_{i_2} = 0$.
- perpendicularity of planes: $\mathbf{n}_{j_1}^\mathsf{T}\mathbf{n}_{j_2} = 0$.

The input data are rather easy to provide interactively, which typically takes 10-15 minutes per image.

[1] By ideal points and ideal lines we denote points and lines at infinity respectively.

(a) Calibration: the dotted line shows the circle ω (see text).

(b) Interest points.

(c) A set of parallel lines.

(d) A set of parallel lines.

(e) A set of parallel planes.

(f) A set of parallel planes.

Figure 2. Illustration of camera calibration and the input used for 3D reconstruction. Crosses of the same color represent interest points belonging to the same line or plane.

4. Basic Idea for 3D Reconstruction

The principal aim here is to reconstruct a set of 3D points and planes. Sets of coplanar 3D points define polygons onto which texture can be mapped for visualization purposes. The reconstruction process is based on operations which are described in the §§5 and 6. Here, we explain that 3D reconstruction from a single image is indeed possible, given the considered types of constraints – coplanarity, parallelism and perpendicularity.

We assume that the image has been calibrated as described in §2.3. Hence, it is possible to backproject image points to 3D, which means that the position of each 3D point is known up to one parameter, its depth. Parallelism and perpendicularity constraints allow us to compute normal directions of some of the planes in the scene (this is described in §§5.2 and 5.3). Hence, these planes are also determined up to one unknown parameter each.

Unless a reference distance in the scene is known, 3D reconstruction can be achieved up to a global scale factor only. We are thus free to arbitrarily fix the position of one point (along its projection ray) or one plane (while preserving its normal). Suppose, we have fixed one point $\mathbf{Q}$. Planes with known normal and which contain $\mathbf{Q}$ (known from the input) are then completely defined. Other points lying on these planes may then be reconstructed, by simply intersecting the backprojection rays with the planes. In turn, other planes may then be reconstructed by fitting them to the already reconstructed 3D points, and so on. This alternation scheme allows to reconstruct objects whose parts are sufficiently "interconnected", i.e. the points on the object have to be linked together via coplanarity or other geometrical constraints.

This discussion shows that it is possible to obtain a 3D reconstruction from one image and constraints of the types considered. However, the alternation scheme just described might not be the best practical solution, since it favors error propagation throughout the reconstruction process. Instead of reconstructing the scene step by step, we thus developed a simple method for simultaneous reconstruction of potentially large sets of points and planes linked together in a way described in §6. Instead of being accumulated, errors are potentially nicely spread over the 3D model. This initial reconstruction is then completed via the alternating point-plane reconstruction scheme outlined above.

In the following section, the basic modules needed for 3D reconstruction are described. The method for simultaneous reconstruction of points and planes is given in §6. The complete algorithm is summarized in §7.

5. Basic Modules for 3D Reconstruction

5.1. Backprojection of Points

Let $\mathbf{q}_p = (x_p, y_p)^\mathsf{T}$ be an image point and $\mathbf{B}_p$ be the 3D direction of the backprojection ray, given by equation (1). Then, the 3D point may be parameterized as

$$\mathbf{Q}_p = \begin{pmatrix} \lambda_p \mathbf{B}_p \\ 1 \end{pmatrix} \quad . \tag{2}$$

The unknown λ_p expresses the distance of $\mathbf{Q}_p$ from the focal point $\mathbf{F}$ and hence defines its position on the projection ray.

5.2. Computation of the Direction of Parallel Lines

Given the input that two or more 3D lines are parallel, we can compute the lines' 3D direction as follows. We suppose that lines are defined by sets of image points (cf. figures 2 (c) and (d)), i.e. a line $\mathbf{l}_{ik}$ is given by points $\mathbf{q}_{ik,1}, \ldots, \mathbf{q}_{ik,n_{ik}}$. For each line, we may compute the 3D interpretation plane, i.e. the plane spanned by the focal point $\mathbf{F}$ and the 3D line. This plane is given by the backprojection rays of the image points. If more than two points are given, a least squares fit is done to determine the plane: the normal is computed as the right singular vector Λ_{ik} associated to the least singular value [6] of the following matrix:

$$\begin{pmatrix} \mathbf{B}_{ik,1}^\mathsf{T} \\ \mathbf{B}_{ik,2}^\mathsf{T} \\ \cdots \\ \mathbf{B}_{ik,n_{ik}}^\mathsf{T} \end{pmatrix}_{n_{ik} \times 3} .$$

The interpretation plane is then given by $\left(\Lambda_{ik}^\mathsf{T}, 0\right)^\mathsf{T}$.

Given the interpretation planes of two or more parallel 3D lines, we may determine the lines' direction as the ideal point obtained by intersection of the interpretation planes' ideal lines. If more than two interpretation planes are given, a least squares fit is done as above. In the following, let the direction of the ith set of parallel lines be represented by the ideal point $\left(\mathbf{D}_i^\mathsf{T}, 0\right)^\mathsf{T}$.

5.3. Computation of the Normal Direction of a Set of Parallel Planes

Planes are depicted by the user by indicating sets of coplanar points in the image (cf. figures 2 (e) and (f)). Given the knowledge that a plane is perpendicular to a set of parallel lines the plane's normal is directly given by the direction of these lines, which may be determined as described above.

A second method to compute a plane's normal is the use of parallelism constraints: given the knowledge that a plane is parallel to two or more sets of mutually parallel lines, then the planes' normal vector $\mathbf{n}_j$ can be computed in a least squares manner: $\mathbf{n}_j$ is obtained as the right singular vector associated to the least singular value of the matrix consisting of the row vectors $\mathbf{D}_i^\mathsf{T}$.

If, by one of these methods, we are able to compute the normal $\mathbf{n}_j$ of a plane Π_j, then the plane may be represented as:

$$\Pi_j = \begin{pmatrix} \mathbf{n}_j \\ d_j \end{pmatrix} \quad , \tag{3}$$

i.e. the plane's position is determined up to its (oriented) distance $-\frac{d_j}{||\mathbf{n}_j||}$ from the focal point $\mathbf{F}$. In the following, we suppose that the normal vectors $\mathbf{n}_j$ have unit norm.

5.4. Other Modules

Our method requires basically two other reconstruction modules, the backprojection of a point onto a 3D plane and the fitting of a plane to a set of 3D points, possibly including ideal points.

Backprojecting a point onto a plane. Backprojection of a point $\mathbf{Q}_p$ onto a plane Π_j is done by computing λ_p via:

$$\lambda_p = -\frac{d_j}{\mathbf{n}_j^\mathsf{T}\mathbf{B}_p} \quad .$$

Fitting a plane to a set of points. Several cases may be considered. In the general case, the cost function to be minimized is the sum of squared distances (we omit here indices referring to the plane):

$$g = \sum_{p=1}^{n} \left(d^2 + 2\left(\mathbf{n}^\mathsf{T}\mathbf{B}_p\right)\lambda_p d + \left(\mathbf{n}^\mathsf{T}\mathbf{B}_p\right)^2 \lambda_p^2 \right) \quad . \qquad (4)$$

Nullifying the partial derivatives leads to a linear homogeneous equation system in the unknowns d and $\mathbf{n}$ that is readily solved.

If we already know the plane's normal $\mathbf{n}$, we obtain the following closed form solution for the unknown d:

$$d = -\frac{\sum_{p=1}^{n}\left(\mathbf{n}^\mathsf{T}\mathbf{B}_p\right)}{\sum_{p=1}^{n} 1} \quad .$$

6. Simultaneous Reconstruction of Points and Planes

The coplanarity constraints provided by the user are in general overconstrained, i.e. several points may lie on more than one plane. This means that, due to image noise, it is difficult to obtain a 3D reconstruction that satisfies all the constraints exactly. This may be achieved by constrained optimization, but there might be no batch method of doing so. Thus, in the following we describe a direct least squares solution for reconstructing a subset of object planes and points, minimizing the sum of squared distances between planes and points. Usually, the subsets of planes and points that may be reconstructed this way cover already a large part of the object.

Consider sets of coplanar points, $S_j = \{\mathbf{Q}_{j,1}, \ldots, \mathbf{Q}_{j,n_j}\}$. A point may belong to more than one set S_j. Let Π_j be the plane corresponding to the point set S_j. In the following, we only consider planes with known normal direction.

We say that two planes Π_{j_1} and Π_{j_2} are *connected* if they share a point, i.e. if the intersection of S_{j_1} and S_{j_2} is non empty. This relationship may be visualized by a graph, whose vertices are planes, with edges being drawn between connected planes. We choose a largest subgraph of connected planes (full connection is not required). Let S'_j be the point sets of the selected planes, points lying on one plane only having been eliminated.

We now show how the considered planes and points may be reconstructed simultaneously in a least squares manner. Reconstruction is done via the determination of the scalars λ and d, as given in equations (2) and (3). Let $\mathbf{Q}$ be a point lying on plane Π. The squared distance between them is given by:

$$\left(d + (\mathbf{n}^\mathsf{T}\mathbf{B})\lambda\right)^2 \quad .$$

We minimize the sum of squared distances for pairs of planes and points. The general form of the cost function is given in the following equation:

$$g = \sum_j \sum_{p,\mathbf{Q}_p \in S'_j} \left(d_j^2 + 2\left(\mathbf{n}_j^\mathsf{T}\mathbf{B}_p\right)\lambda_p d_j + \left(\mathbf{n}_j^\mathsf{T}\mathbf{B}_p\right)^2 \lambda_p^2 \right) \quad .$$

The partial derivatives (divided by 2) of the cost function are given by:

$$\frac{\sigma g}{\sigma d_j} = \left(\sum_{p,\mathbf{Q}_p \in S'_j} 1 \right) d_j + \sum_{p,\mathbf{Q}_p \in S'_j} \left(\left(\mathbf{n}_j^\mathsf{T}\mathbf{B}_p\right)\lambda_p \right)$$

$$\frac{\sigma g}{\sigma \lambda_p} = \sum_{j,\mathbf{Q}_p \in S'_j} \left(\left(\mathbf{n}_j^\mathsf{T}\mathbf{B}_p\right) d_j \right) + \left(\sum_{j,\mathbf{Q}_p \in S'_j} \left(\mathbf{n}_j^\mathsf{T}\mathbf{B}_p\right)^2 \right) \lambda_p \quad .$$

Nullifying these equations leads to a homogeneous linear equation system in the unknowns d_j and λ_p, giving the least squares solution. The solution is defined up to scale, as expected, since reconstruction can only be done up to scale.

The equation system has the nice structure shown in equation (5), where

$$\begin{aligned} C_{jp} &= \begin{cases} \mathbf{n}_j^\mathsf{T}\mathbf{B}_p & \text{if } \mathbf{Q}_p \in S'_j, \\ 0 & \text{else.} \end{cases} \\ D_j &= \sum_{p,\mathbf{Q}_p \in S'_j} 1 \\ L_p &= \sum_{j,\mathbf{Q}_p \in S'_j} \left(\mathbf{n}_j^\mathsf{T}\mathbf{B}_p\right)^2 \quad . \end{aligned}$$

The equation system is relatively well conditioned since the $\mathbf{n}_j$ and $\mathbf{B}_p$ are unit vectors and the D_j entries (the number of points on plane j) are usually not much larger than 10. Special sparse solution methods may be used like e.g. in [4], but for small problems (the size of the matrix is the number of planes plus the number of points, which is usually at most a few dozens for single images) we simply use singular value decomposition [6].

$$\left(\begin{array}{cccc|cccc} D_1 & & & & C_{11} & C_{12} & \cdots & C_{1n} \\ & D_2 & & & C_{21} & C_{22} & \cdots & C_{2n} \\ & & \ddots & & \vdots & \vdots & & \vdots \\ & & & D_m & C_{m1} & C_{m2} & \cdots & C_{mn} \\ \hline C_{11} & C_{21} & \cdots & C_{m1} & L_1 & & & \\ C_{12} & C_{22} & \cdots & C_{m2} & & L_2 & & \\ \vdots & \vdots & & \vdots & & & \ddots & \\ C_{1n} & C_{2n} & \cdots & C_{mn} & & & & L_n \end{array}\right) \left(\begin{array}{c} d_1 \\ d_2 \\ \vdots \\ d_m \\ \hline \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_n \end{array}\right) = \left(\begin{array}{c} 0 \\ 0 \\ \vdots \\ 0 \\ \hline 0 \\ 0 \\ \vdots \\ 0 \end{array}\right) . \tag{5}$$

7. Complete Algorithm

1. Calibrate the system (cf. §2.3).

2. Backproject all points up to scale, i.e. compute the vectors $\mathbf{B}_p$ (cf. §5.1). Scale the $\mathbf{B}_p$ to unit norm and use extended coordinates for 3D points: $\mathbf{Q}_p^{\mathsf{T}} = (\lambda_p(\mathbf{B}_p)^{\mathsf{T}}, 1)$.

3. Compute line directions and normal directions of planes (cf. §§5.2 and 5.3).

4. Partition the planes with known normal in sets of planes which are connected by at least one point (in a transitive manner).

5. Choose the largest partition.

6. Simultaneously reconstruct plane and point positions as described in §6. Use only points that lie on more than one plane in the actual partition (other points do not add useful redundancy).

7. Backproject the other points that lie on planes in the actual partition (cf. §5.4).

8. Reconstruct a plane not reconstructed yet by fitting it to 3D points (cf. §5.4). Each point provides one equation (and the possibly known normal direction two). Choose the plane with the most (independent) equations.

9. Backproject points lying on the plane just reconstructed.

10. If there are planes not reconstructed yet, go to step 8.

Note that this process is done completely automatically.

From the 3D reconstruction, we may create textured VRML models (see an example in §8).

8. Example

Figure 4 on the last page shows rendered views of a textured 3D model obtained from the image shown in figure 3 on the following page. The input image was obtained with the CycloVision ParaShot system and an Agfa ePhoto 1680 camera. Texture maps were created from the panoramic image using the projection equations in §2.2 and bicubic interpolation [8]. With other images, similar results were obtained.

9. Conclusion

We have presented a method for interactive 3D reconstruction of piecewise planar objects from a single panoramic view. The method was developed for a sensor based on a parabolic mirror, but its adaptation to other sensors is straightforward. 3D reconstruction is done using geometrical constraints provided by the user, that are simple in nature (coplanarity, perpendicularity and parallelism) and may be easily provided without any computer vision expertise.

The major drawback of single-view 3D reconstruction is of course that only limited classes of objects may be reconstructed and that the reconstruction is usually incomplete. The major advantages however are that it is a quick way of obtaining 3D models, that it is rather easy to implement and to use and that due to user interaction and the small size of the problem the reconstruction process becomes very reliable, compared to more automatic multi-view systems. Also, using geometrical constraints on the scene structure is always a good idea in order to obtain realistic 3D models. 3D models from single images might be used to register perspective views of scene details in order to obtain high resolution global 3D models.

One advantage of our method compared to other approaches is that a wider class of objects can be reconstructed (especially, there is no requirement of disposing of two or more ideal points for each plane). The simultaneous reconstruction of several planes and several points that forms the starting point of our method makes it likely that errors are nicely spread over the whole 3D model, compared to more sequential approaches.

Please contact the author for getting a paper version with color figures.

References

[1] S. Baker and S.K. Nayar, "A Theory of Catadioptric Image Formation," *Proceedings International Conference on Computer Vision, Bombay*, pp. 35-42, January 1998.

[2] C. Geyer and K. Daniilidis, "Catadioptric Camera Calibration," *Proceedings International Conference on Computer Vision, Kerkyra, Greece*, pp. 398-404, September 1999.

[3] J. Gluckman and S.K. Nayar, "Ego-Motion and Omnidirectional Cameras," *Proceedings International Conference on Computer Vision, Bombay*, pp. 999-1005, January 1998.

[4] R.I. Hartley, "Euclidean Reconstruction from Uncalibrated Views," *Proceeding of the* DARPA–ESPRIT *Workshop on Applications of Invariants in Computer Vision, Azores, Portugal*, pp. 187-202, October 1993.

[5] H. Ishiguro, M. Yamamoto and S. Tsuji, "Omni-Directional Stereo," IEEE *Transactions on Pattern Analysis and Machine Intelligence*, Vol. 14, No. 2, pp. 257-262, February 1992.

[6] A. Jennings, J.J. McKeown, *Matrix Computation,* 2nd edition, Wiley, 1992.

[7] S.B. Kang and R. Szeliski, "3-D scene data recovery using omnidirectional multi-baseline stereo," *Proceedings* IEEE *Conference on Computer Vision and Pattern Recognition, San Francisco*, pp. 364-370, June 1996.

[8] R.G. Keys, "Cubic Convolution Interpolation for Digital Image Processing," IEEE *Transactions on Acoustics, Speech, and Signal Processing,* Vol. 29, No. 6, pp. 1153-1160, December 1981.

[9] D. Liebowitz, A. Criminisi and A. Zisserman, "Creating Architectural Models from Images," *Proceedings EuroGraphics*, vol. 18, pp. 39-50, September 1999.

[10] S.K. Nayar, "Catadioptric Omnidirectional Camera," *Proceedings* IEEE *Conference on Computer Vision and Pattern Recognition, Puerto Rico*, pp. 482-488, June 1997.

[11] V.N. Peri and S.K. Nayar, "Generation of Perspective and Panoramic Video from Omnidirectional Video," *Proceedings* DARPA *Image Understanding Workshop, New Orleans*, May 1997.

[12] H.-Y. Shum, R. Szeliski, S. Baker, M. Han, P. Anandan, "Interactive 3D Modeling from Multiple Images Using Scene Regularities," *SMILE Workshop, Freiburg, Germany*, pp. 236-252, June 1998.

[13] P. Sturm and S. Maybank, "A Method for Interactive 3D Reconstruction of Piecewise Planar Objects from Single Images," *Proceeding British Machine Vision Conference, Nottingham*, pp. 265-274, September 1999.

[14] T. Svoboda, *Central Panoramic Cameras – Design, Geometry, Egomotion,* PhD Thesis, Faculty of Electrical Engineering, Czech Technical University, Prague, September 1999.

[15] M. Watanabe and S.K. Nayar, "Telecentric Optics for Computer Vision," *Proceedings European Conference on Computer Vision, Cambridge*, pp. 439-451, April 1996.

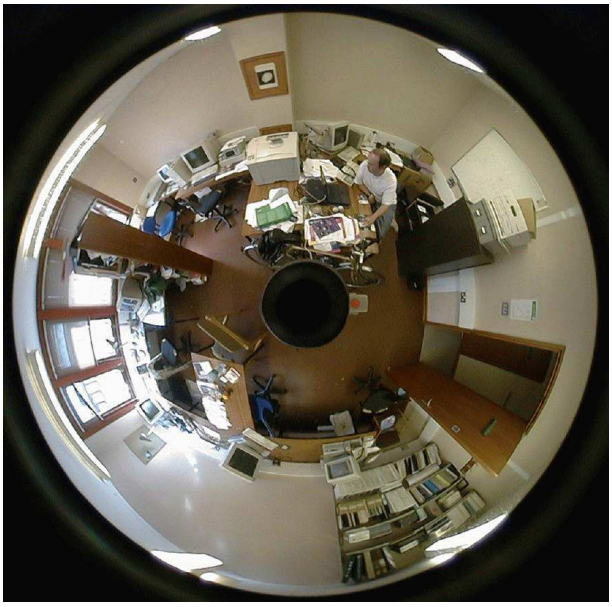
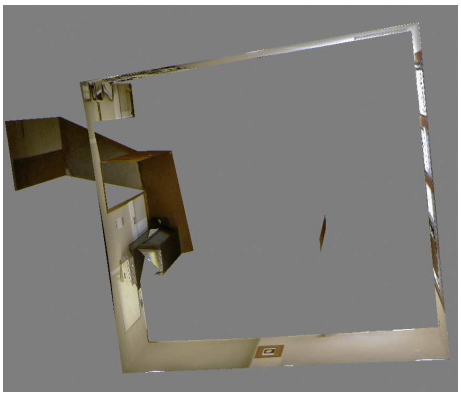
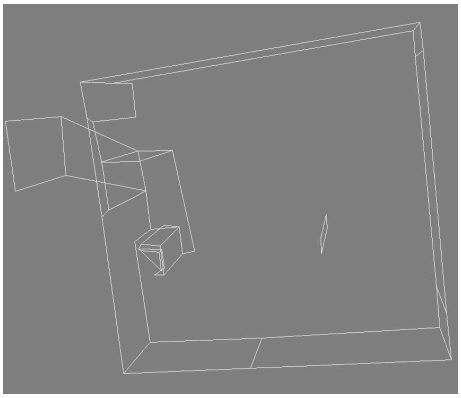


Figure 3. The input image.



(a) An overhead view of the scene. On the left hand side, a part of a hallway outside the office that has been reconstructed, is visible.



(b) A wireframe model of the reconstruction.

(c) A view from below the floor.

(d) A view from below and outside the reconstructed office.

(e) Objects not visible in the input image obviously lead to holes in the 3D model.

Figure 4. Results.

194 IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 27, NO. 2, FEBRUARY 2005

Using Geometric Constraints through Parallelepipeds for Calibration and 3D Modeling

Marta Wilczkowiak, Peter Sturm, and Edmond Boyer

Abstract—This paper concerns the incorporation of geometric information in camera calibration and 3D modeling. Using geometric constraints enables more stable results and allows us to perform tasks with fewer images. Our approach is motivated and developed within a framework of semi-automatic 3D modeling, where the user defines geometric primitives and constraints between them. It is based on the observation that constraints, such as coplanarity, parallelism, or orthogonality, are often embedded intuitively in parallelepipeds. Moreover, parallelepipeds are easy to delineate by a user and are well adapted to model the main structure of, e.g., architectural scenes. In this paper, first a duality that exists between the shape parameters of a parallelepiped and the intrinsic parameters of a camera is described. Then, a factorization-based algorithm exploiting this relation is developed. Using images of parallelepipeds, it allows us to simultaneously calibrate cameras, recover shapes of parallelepipeds, and estimate the relative pose of all entities. Besides geometric constraints expressed via parallelepipeds, our approach simultaneously takes into account the usual self-calibration constraints on cameras. The proposed algorithm is completed by a study of the singular cases of the calibration method. A complete method for the reconstruction of scene primitives that are not modeled by parallelepipeds is also briefly described. The proposed methods are validated by various experiments with real and simulated data, for single-view as well as multiview cases.

Index Terms—3D modeling, calibration, geometric constraints.

1 Introduction

Efficient 3D modeling from images is one of the most challenging issues in computer vision. The tremendous research effort made to develop feasible methods has proven that recovering 3D structures from 2D images is a difficult and often underconstrained problem. Several reasons account for that, including the fundamental fact that, without any prior information on cameras or on the scene to recover, a Euclidean reconstruction is not possible at all [1]. This is why knowledge of the acquisition process or of the scene is required. A number of approaches have been proposed to exploit prior information, both on camera and scene parameters. Such prior information not only solves the projective ambiguity in the reconstruction but also usually stabilizes the sensitive reconstruction process. Furthermore, it often leads to simple and direct solutions for the estimation of both camera and scene parameters, which may eventually be adjusted nonlinearly for higher accuracy. The method proposed in this paper is based on the observation that constraints such as coplanarity, parallelism, or orthogonality are often embedded intuitively in parallelepipeds. Moreover, parallelepipeds are easy to delineate by a user and are well adapted to model the main structure of, e.g., architectural scenes. Using parallelepipeds to constrain the calibration and reconstruction process enables modeling from small sets of images, in particular from single images, thus making possible reconstructions from images not originally taken for that purpose, such as archival images or images from the Internet.

An exhaustive review of the literature on using prior information for self-calibration and Euclidean reconstruction is beyond the scope of this paper. We will concentrate on works which have somehow inspired the method we propose, especially direct approaches giving a good first estimate of camera and scene parameters. There is a large variety of information which can be incorporated into a 3D modeling process. This can be simple knowledge of camera intrinsic parameters or pose (stationarity, pure translation, etc.) or of global 3D scene structure (calibration patterns); it can also be information on scene elements such as points, lines, and planes, as well as on high-level primitives like cubes, prisms, cylinders, etc. Nonetheless, whatever the information is, it can be used at any stage of the 3D modeling process, including the initial calibration, pose estimation, model reconstruction, or an additional nonlinear adjustment of the initial estimate at each step.

Approaches based on calibration patterns. Classical calibration approaches are based on known positions of points in 3D space or known calibration patterns [2]. Unfortunately, such information relies on specific acquisition systems and is thus seldom available in general situations. The use of prior knowledge on some intrinsic parameters, i.e., self-calibration, offers the opportunity to build more flexible systems.

Self-calibration. In standard self-calibration algorithms [3], [4], [5], [6], 3D reconstruction is done in three steps, recovering, in order, the projective, affine, and Euclidean strata, the projective-affine step being considered as the most nonlinear and, thus, the most difficult step. One of the main problems is critical motion sequences, for which self-calibration does not have a unique solution [7]. This problem has been dealt with by restraining the camera motions [8], [9], [10] or by incorporating prior knowledge on the camera [11]

- *M. Wilczkowiak is with the Department of Engineering, University of Cambridge, Cambridge CB2 1PZ, UK. E-mail: mw373@cam.ac.uk.*
- *P. Sturm and E. Boyer are with INRIA Rhône-Alpes, 655 Avenue de l'Europe, 38330 Montbonnot, France. E-mail: {Peter.Sturm, Edmond.Boyer}@inrialpes.fr.*

Manuscript received 5 Jan. 2004; revised 10 June 2004; accepted 25 June 2004; published online 13 Dec. 2004.
Recommended for acceptance by C. Taylor.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0006-0104.

or on the scene. But, to get stable results for self-calibration, a large number of images is usually necessary.

Structure and motion. The basic constraint is that backprojection lines (planes) associated with corresponding image points (lines) intersect in a single space point (line). This observation allows us to formulate the matching tensors, which compactly describe two, three, and four view geometry. When more views are accessible, it is necessary to combine results computed from small subsets of images, which decreases the accuracy of results. An overview of tensor-based structure&motion methods can be found in [12].

Another category of approaches allows the simultaneous recovery of cameras and 3D models via the factorization of a measurement matrix of image points [13], [14], lines [15], [16], or similar methods using planes in the scene [17], [18]. Factorization methods suffer from missing data, i.e., when a primitive is not seen in all images, although some ways of dealing with this problem have been proposed [13], [16]. Using only the above backprojection constraints, it is only possible to recover the scene up to a projective or affine transformation.

Incorporating Euclidean scene constraints. A large variety of geometric constraints can disambiguate the projective reconstruction to a Euclidean one and allow us to decrease the number of images required to obtain a satisfying reconstruction. Many of them can be easily incorporated into a self-calibration framework. A common constraint is given by vanishing points of mutually orthogonal directions, as defined by known cubical structures [19], [20], [21] or by dominating scene directions [22]. Also, knowing the Euclidean structure of scene planes is useful in this context, through rectified planes [23], maps [24], or known plane-to-image homographies [25], [26]. It is also possible to use multiple images of unknown planes, but more images in general position are needed here [27], [28].

When cameras are calibrated, it is relatively easy to reconstruct a 3D structure. However, and as mentioned previously, using geometric constraints may dramatically improve the reconstruction quality, especially when a single or only a few images are considered [29]. Even simple constraints can be very efficient, e.g., in [30], [31], vanishing lines of planes and coplanarity constraints are used for single image reconstruction. However, in general, dealing with different types of scene objects and constraints is a complicated problem. Some authors prefer to model the scene by simple primitives like points, lines, and planes and constraints between them, such as incidence, parallelism, orthogonality, etc. Some direct approaches using the bilinear character of many useful constraints were proposed in [32], [33], [34]. The results can be improved using nonlinear methods applying penalty terms corresponding to the constraints [35], constrained optimization techniques [36], [37], [38], or a minimal scene parameterization [39], [40]. Yet a different approach consists of high-level scene descriptions using complex primitives like cubes, prisms, cylinders, etc. [41], [42]. Recently, some effort has been devoted to the automatic detection of such primitives [43]. All these methods ensure, by the strong inherent geometric constraints, that the final models are visually correct.

The proposed approach. In this paper, we address the first part of the 3D modeling process—intrinsic and extrinsic calibration (pose/motion estimation). In particular, we study the use of a specific calibration primitive: the parallelepipeds. Parallelepipeds are frequently present in man-made environments and they naturally encode the scene's affine structure. Any information about their Euclidean structure (angles or ratios of edge lengths), possibly combined with information about camera parameters, may allow us to recover the entire scene's Euclidean structure. We propose an elegant formalism to incorporate such information, in which camera parameters are dual to parallelepiped parameters, i.e., any knowledge about one entity provides constraints on the parameters of the others. Hence, the image of a known parallelepiped defines the camera parameters and, reciprocally, a calibrated image of a parallelepiped defines its Euclidean shape (up to size). In this paper, we synthesize previous work on parallelepipeds [44], [45] and propose more elegant and efficient approaches.

Camera and parallelepiped parameters are recovered in two steps. First, a factorization-based approach is used to compute their intrinsic and orientation (rotation) parameters. The usual problems of factorization methods— missing data and unknown scale factors—are dealt with rather easily. Then, position and size parameters are recovered simultaneously using linear least squares. The use of well-constrained calibration primitives allows us to obtain good calibration results even from as little as one image. However, depending on the available constraints, singularities might occur. These are described in a detailed catalogue.

Our calibration approach is conceptually close to self-calibration, especially to methods that upgrade an affine structure to Euclidean [5], [6] or methods considering special camera motions [8], [9], [10]. The way Euclidean information on a parallelepiped is used is also similar to vanishing point-based methods [19], [20], [21], [22]. Some properties of our algorithm are also common with plane-based approaches [25], [26], [27], [28], [17], [18]. While more flexible than standard calibration techniques, plane-based approaches still require either Euclidean information or, for self-calibration, many images in general position [27], or at least one plane visible in all images [17]. In this sense, our approach is a generalization of plane-based methods with Euclidean information to three-dimensional parallelepipedic patterns. Finally, our approach can be compared to methods using complex primitives for scene representation. However, unlike most such methods, we use the parallelepiped parameters directly to solve the calibration problem, without requiring nonlinear optimization.

While the main contributions of the paper concern the estimation of camera and parallelepiped parameters, we show that the proposed method can be easily combined with an approach for enhancing reconstructions with primitives other than parallelepipeds [34]. The complete system allows for both calibration and 3D model acquisition from a small number of images with a reasonable amount of user interaction.

The paper is organized as follows: Section 2 gives definitions and some background. Section 3 introduces the concept of camera-parallelepiped duality. Calibration using parallelepipeds and a study on the singular configurations are described in Sections 4 and 5. Sections 6 and 7 describe our approaches for pose estimation and 3D reconstruction. Experimental results are presented in Section 8.

2 Preliminaries

2.1 Camera Parameterization

We represent cameras using the pinhole model. The projection from a 3D point $\mathbf{P}$ to a 2D image point $\mathbf{p}$ is expressed by: $\mathbf{p} \sim \mathrm{M}\mathbf{P}$, where M is a 3×4 matrix, which can be decomposed as:

$$\mathrm{M} = \mathrm{K}(\mathrm{R} \quad \mathbf{t}).$$

The 3×4 matrix $(\mathrm{R} \quad \mathbf{t})$ encapsulates the camera's pose in the world coordinate system or its extrinsic parameters: The rotation matrix R represents its orientation and the vector $-\mathrm{R}^\top\mathbf{t}$ its position. The 3×3 calibration matrix K or, equivalently, $\omega \sim \mathrm{K}^{-\top}\mathrm{K}^{-1}$ represents the camera's intrinsic parameters:

$$\mathrm{K} = \begin{pmatrix} \alpha_u & s & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\omega \sim \mathrm{K}^{-\top}\mathrm{K}^{-1} \sim \begin{pmatrix} 1 & 0 & -u_0 \\ 0 & \tau^2 & -\tau^2 v_0 \\ -u_0 & -\tau^2 v_0 & \tau^2\alpha_v^2 + u_0^2 + \tau^2 v_0^2 \end{pmatrix}, \quad (1)$$

where α_u and α_v stand for the focal length, expressed in horizontal and vertical pixel dimensions, s is a skew parameter considered as equal to zero in the following, (u_0, v_0) are the pixel coordinates of the principal point, and $\tau = \frac{\alpha_u}{\alpha_v}$ is the camera's aspect ratio. ω represents the IAC (image of the absolute conic) and is commonly used to express constraints on the intrinsic parameters. In the following, the term *camera axes* will be used for the axes of the camera coordinate system, i.e., the coordinate system attached to the camera's optical center, two of them being parallel to pixel edges and the third one being orthogonal to the image plane (the optical axis).

2.2 Parallelepiped Parameterization

A parallelepiped is defined by 12 parameters: six extrinsic parameters describing its orientation and position and six intrinsic parameters describing its Euclidean shape: three dimension parameters (edge lengths l_1, l_2, and l_3) and three angles between edges $(\theta_{12}, \theta_{23}, \theta_{13})$. These intrinsic parameters are illustrated in Fig. 1. The parallelepiped may be represented compactly by a 4×4 matrix N:

$$\mathrm{N} = \begin{pmatrix} \mathrm{S} & \mathbf{v} \\ \mathbf{0}^\top & 1 \end{pmatrix} \underbrace{\begin{pmatrix} l_1 & l_2 c_{12} & l_3 c_{13} & 0 \\ 0 & l_2 s_{12} & l_3 \frac{c_{23} - c_{13}c_{12}}{s_{12}} & 0 \\ 0 & 0 & l_3 \sqrt{\frac{s_{12}^2 - c_{13}^2 s_{12}^2 - (c_{23} - c_{13}c_{12})^2}{s_{12}^2}} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}}_{\tilde{\mathrm{L}}},$$

where S is a rotation matrix and $\mathbf{v}$ a vector, representing the parallelepiped's pose (extrinsic parameters). The 4×4 matrix $\tilde{\mathrm{L}}$ represents the parallelepiped's shape (intrinsic parameters) with: $c_{ij} = \cos\theta_{ij}$, $s_{ij} = \sin\theta_{ij}$, $\theta_{ij} \in]0\ \pi[$, $l_i > 0$.

The matrix $\tilde{\mathrm{L}}$ represents the affine transformation between a canonic cube and a parallelepiped with the given shape. Concretely, a vertex $(\pm 1, \pm 1, \pm 1, 1)^\top$ of the canonic cube is mapped, by $\tilde{\mathrm{L}}$, to a vertex of our parallelepiped's intrinsic shape. Then, the pose part of N maps the vertices into the world coordinate system.

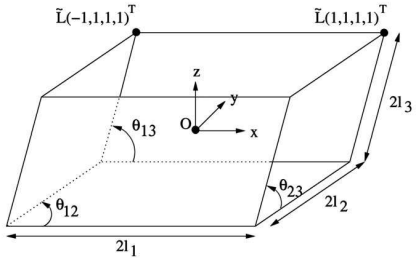


Fig. 1. Parameterization of a parallelepiped: $2l_i$ are the edge lengths, θ_{ij} are the angles between nonparallel edges.

Other parameterizations for $\tilde{\mathrm{L}}$ may be chosen, but the above one is attractive due to its upper triangular form. This underlines the fact that $\tilde{\mathrm{L}}$ plays the same role for the parallelepiped as the calibration matrix K for a camera.

Analogous to a camera's IAC ω is the matrix μ, defined by:

$$\mu \sim \mathrm{L}^\top\mathrm{L} \sim \begin{pmatrix} l_1^2 & l_1 l_2 \cos\theta_{12} & l_1 l_3 \cos\theta_{13} \\ l_1 l_2 \cos\theta_{12} & l_2^2 & l_2 l_3 \cos\theta_{23} \\ l_1 l_3 \cos\theta_{13} & l_2 l_3 \cos\theta_{23} & l_3^2 \end{pmatrix}, \quad (2)$$

where L is the upper left 3×3 matrix of $\tilde{\mathrm{L}}$.

Hence, there is a symmetry between the intrinsic parameters of cameras and parallelepipeds ((1) and (2)). The only difference is that, in some cases, the *size* of a parallelepiped matters, as will be explained in the following. As for cameras, the fact that $K_{33} = 1$ allows us to fix the scale factor in the relation $\omega \sim \mathrm{K}^{-\top}\mathrm{K}^{-1}$ and, thus, to extract K uniquely from the IAC ω, e.g., using Cholesky decomposition. As for parallelepipeds, however, we have no such constraint on its "calibration matrix" L, so the relation $\mu \sim \mathrm{L}^\top\mathrm{L}$ gives us a parallelepiped's Euclidean shape, but not its (absolute) size. This does not matter in general since we are usually only interested in reconstructing a scene up to some scale. However, when reconstructing several parallelepipeds, one needs to recover at least their *relative* sizes.

There are many possibilities to define the size of a parallelepiped. We choose the following definition, motivated by the equations underlying our calibration and reconstruction algorithms below: The **size** of a parallelepiped is defined as $s = (\det \mathrm{L})^{1/3}$. This definition is actually directly linked to the parallelepiped's volume: $s^3 = \det \mathrm{L} = \mathrm{Vol}/8$ (the factor 8 arises since our canonic cube has an edge length of 2).

3 Projections of Parallelepipeds

3.1 One Parallelepiped in a Single View

In this section, we introduce the concept of duality between the intrinsic parameters of cameras and parallelepipeds. Consider the projection of a parallelepiped's vertices into a camera. Let $\mathbf{C}_{i,\ i\in[1..8]}$ be the homogeneous coordinates of the canonic cube's vertices. Using results from Section 2.2, the projection of the corresponding vertex in the image is:

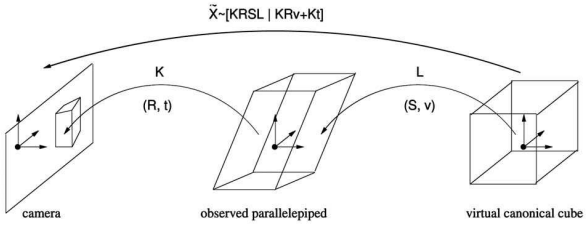


Fig. 2. The projection of the canonic parallelepiped (cube) into the image. Matrices K, L correspond to intrinsic parameters of camera and parallelepiped and $(\mathrm{R}, \mathbf{t})$, $(\mathrm{S}, \mathbf{v})$ correspond to extrinsic parameters of camera and parallelepiped, respectively.

$$\mathbf{p}_i \sim \mathrm{M}\mathbf{P}_i = \underbrace{\mathrm{K}(\,\mathrm{R} \quad \mathbf{t}\,)\begin{pmatrix} \mathrm{S} & \mathbf{v} \\ \mathbf{0}^\top & 1 \end{pmatrix}\tilde{\mathrm{L}}}_{\tilde{\mathrm{X}}}\,\mathbf{C}_i. \tag{3}$$

The matrix $\tilde{\mathrm{X}}$ will be called the *canonic projection matrix*. It represents a perspective projection that maps the vertices of the canonic cube onto the image points of the parallelepiped's vertices. This is illustrated in Fig. 2. Given image points for at least six vertices,[1] the canonic projection matrix can be computed [2], even without prior knowledge on intrinisic or extrinisic parameters. Our calibration and pose algorithms are based on the link between the canonic projection matrix $\tilde{\mathrm{X}}$ (which we suppose given from now on) and the camera's and parallelepiped's intrinsic and extrinsic parameters.

Let us consider this in more detail. First, we may identify the *relative pose* between camera and parallelepiped in (3), represented by the following 3×4 matrix:

$$(\,\mathrm{R} \quad \mathbf{t}\,)\begin{pmatrix} \mathrm{S} & \mathbf{v} \\ \mathbf{0}^\top & 1 \end{pmatrix} = (\,\mathrm{RS} \quad \mathrm{R}\mathbf{v} + \mathbf{t}\,).$$

Second, let us consider the leading 3×3 submatrix X of the canonic projection matrix $\tilde{\mathrm{X}}$, which is given by: $\mathrm{X} \sim \mathrm{K}(\mathrm{RS})\mathrm{L}$.

Due to the orthogonality of the rotation matrices R and S, it is simple to derive the following relation between the camera's IAC ω and the corresponding entity μ of the parallelepiped:

$$\mathrm{X}^\top \omega \mathrm{X} \sim \mu. \tag{4}$$

This equation establishes an interesting duality between the intrinsic parameters of a camera and those of a parallelepiped. It shows (unsurprisingly) that knowing the parallelepiped's shape μ allows us to calibrate the camera. Conversely, knowing the camera's intrinsic parameters allows us to compute the parallelepiped's Euclidean shape, also from a single image. Moreover, even partial information about one set of intrinsic parameters allows us to form equations on the other set [44].

In the next sections, we generalize the use of this duality for calibration and pose estimation to the case of multiple parallelepipeds seen in multiple cameras and to the use of *partial* knowledge about the camera's or parallelepiped's intrinsic parameters. Before doing so, let us describe a few interesting links between our and other (self-) calibration scenarios.

Classical self-calibration usually proceeds in two main steps: First, a projective reconstruction of the scene is obtained from image correspondences. Then, this is upgraded to a Euclidean reconstruction using the available prior knowledge on intrinsic parameters. Sometimes, an intermediate upgrade to an affine reconstruction is performed.

In our scenario, we have a 3D reconstruction of the scene already from a *single* rather than multiple images, which is, furthermore, of *affine* rather than projective nature: We know that the observed parallelepiped's shape is that of a cube, up to some affine transformation. Analogously, our canonic projection matrix is equal to the true one up to an affine transformation. Hence, self-calibration in our scenario does not need to recover the plane at infinity, which is known to be the hardest part of self-calibration. Indeed, our calibration method is somewhat similar to the affine-to-Euclidean upgrade of stratified self-calibration approaches, e.g., [5], [6].

Similarities also exist with (self-) calibration approaches based on special camera motions: Calibrating a rotating camera [8], [9] is more or less equivalent to self-calibrating a camera in general motion once the affine structure is known. Other approaches recover the affine structure by first performing pure translations and then general motions [10], [46].

Our approach is similar to all these. In the following sections, we show how it allows us to efficiently combine the usual self-calibration constraints with constraints on scene structure. This enables us to perform calibration (and 3D reconstruction) from very few images; one image may actually be sufficient.

3.2 n Parallelepipeds in m Views

Let us now consider the general case where n parallelepipeds are seen by m cameras. Let $\tilde{\mathbf{X}}_{ik}$ be the canonic projection matrix associated with the projection of the kth parallelepiped in the ith camera and λ_{ik} a scale factor such that (3) can be written as a component-wise equality:

$$\lambda_{ik}\tilde{\mathbf{X}}_{ik} = \mathrm{K}_i(\,\mathrm{R}_i \quad \mathbf{t}_i\,)\begin{pmatrix} \mathrm{S}_k & \mathbf{v}_k \\ \mathbf{0}^\top & 1 \end{pmatrix}\tilde{\mathrm{L}}_k. \tag{5}$$

We may gather these equations for all m cameras and n parallelepipeds into the following single matrix equation:

$$\underbrace{\begin{bmatrix} \lambda_{11}\tilde{\mathbf{X}}_{11} & \cdots & \lambda_{1n}\tilde{\mathbf{X}}_{1n} \\ \vdots & \ddots & \vdots \\ \lambda_{m1}\tilde{\mathbf{X}}_{m1} & \cdots & \lambda_{mn}\tilde{\mathbf{X}}_{mn} \end{bmatrix}}_{\mathcal{X}_{3m\times 4n}} = \underbrace{\begin{bmatrix} \mathrm{K}_1(\,\mathrm{R}_1 \quad \mathbf{t}_1\,) \\ \vdots \\ \mathrm{K}_m(\,\mathrm{R}_m \quad \mathbf{t}_m\,) \end{bmatrix}}_{\mathcal{M}_{3m\times 4}} \underbrace{\begin{bmatrix} \begin{pmatrix} \mathrm{S}_1 & \mathbf{v}_1 \\ \mathbf{0}^\top & 1 \end{pmatrix}\tilde{\mathrm{L}}_1 & \cdots & \begin{pmatrix} \mathrm{S}_n & \mathbf{v}_n \\ \mathbf{0}^\top & 1 \end{pmatrix}\tilde{\mathrm{L}}_n \end{bmatrix}}_{\mathcal{S}_{4\times 4n}}. \tag{6}$$

This equation naturally leads to the idea of a factorization-based calibration algorithm, which will be developed in Section 4. It is based on the following observation: The matrix $\mathcal{X}$ contains all information that can be recovered from the parallelepipeds' image points alone (below, we

1. In theory, five image points and one image direction are sufficient to determine the 11 parameters of a projection matrix. Additional points make the computation more stable.

discuss the issue of computing the scale factors λ_{ik}). In analogy with [13], we call it the *measurement matrix*. Since the measurement matrix is the product of a "motion matrix" $\mathcal{M}$ of four columns, with a "shape matrix" $\mathcal{S}$ of four rows, its rank can be four at most (in the absence of noise).

We might aim at extracting intrinsic and extrinsic parameters directly from a rank-4-factorization of $\mathcal{X}$. One step of factorization-based methods for structure and motion recovery is to disambiguate the factorization's result: In general, for a rank-r-factorization, motion and shape are recovered up to a transformation represented by an $r \times r$ matrix (here, this would be a 3D projective transformation). The ambiguity can be reduced using, e.g., constraints on intrinsic camera parameters (see more details in Section 4). In our case, we observe that the 4×4 subblocks of the shape matrix $\mathcal{S}$ are affine transformations. We would have to include this constraint into the disambiguation, but, nevertheless, the result would not, in general, exactly satisfy the affine form of these subblocks. We thus cut the problem into two steps, which allows us to easily guarantee that the subblocks of the shape matrix will be affine transformations. In the first step (Section 4), we consider a "reduced measurement matrix" consisting of the leading 3×3 submatrices of the $\tilde{\mathrm{X}}_{ik}$. We extract *intrinsic* and *orientation* parameters of our cameras and parallelepipeds based on a rank-3-factorization and a disambiguation stage using calibration and scene constraints. In the second step (Section 6), we then estimate the *position* of cameras and parallelepipeds, as well as the parallelepipeds' *size*.

Just as a sidenote, we observe that, for two views i and j and a parallelepiped k, the infinite homography between the two views is given by the product $\mathrm{X}_{ik}\mathrm{X}_{jk}^{-1}$.

4 Estimating Intrinsic and Orientation Parameters by Factorization

In this section, we concentrate on the computation of the cameras' and parallelepipeds' intrinsic parameters and orientation (rotation), based on (6) and the observations concerning it, cf. the previous section. As mentioned, we first restrict our attention to the leading 3×3 submatrices of the $\tilde{\mathrm{X}}_{ik}$, as in Section 3.1 for the establishment of the duality between intrinsic parameters of cameras and parallelepipeds. We thus deal with the following subpart of (6):

$$\underbrace{\begin{bmatrix} \lambda_{11}\mathrm{X}_{11} & \cdots & \lambda_{1n}\mathrm{X}_{1n} \\ \vdots & \ddots & \vdots \\ \lambda_{m1}\mathrm{X}_{m1} & \cdots & \lambda_{mn}\mathrm{X}_{mn} \end{bmatrix}}_{\mathcal{X}'_{3m\times 3n}} = \underbrace{\begin{bmatrix} \mathrm{K}_1\mathrm{R}_1 \\ \vdots \\ \mathrm{K}_m\mathrm{R}_m \end{bmatrix}}_{\mathcal{M}'_{3m\times 3}} \underbrace{\begin{bmatrix} \mathrm{S}_1\mathrm{L}_1 & \cdots & \mathrm{S}_n\mathrm{L}_n \end{bmatrix}}_{\mathcal{S}'_{3\times 3n}}. \tag{7}$$

In the following, we describe the different steps of our factorization-based method. We first deal with the problem of missing data. Then, we describe how to compute the scale factors λ_{ik}, needed to construct the measurement matrix $\mathcal{X}'$. The factorization itself is described in Section 4.3, followed by the most important aspect: how to disambiguate the factorization's result in order to extract intrinsic and orientation parameters. A summary of these steps and a discussion of minimal cases and singularities is provided at the end of this section and in Section 5. The subsequent computation of position parameters and parallelepiped size is dealt with in Section 6.

4.1 Missing Data

As is usual with factorization approaches, our method might suffer from the problem of missing data, i.e., missing X_{ik}. Indeed, in practice, the condition that all parallelepipeds are seen in all views is usually not satisfied. However, each missing matrix X_{ik} can be deduced from others if there is one camera j and one parallelepiped l such that X_{jl}, X_{jk}, and X_{il} are known. The missing matrix can be computed using:

$$\mathrm{X}_{ik} \sim \mathrm{X}_{il}\,(\mathrm{X}_{jl})^{-1}\,\mathrm{X}_{jk}. \tag{8}$$

Several equations of this type can be used simultaneously to increase the accuracy. Care has to be taken since (8) is defined up to scale only. This problem can be circumvented very simply though by normalizing all X_{ik} to unit determinant.

These observations motivate a simple recursive method[2] to compute missing matrices X_{ik}: At each iteration, we compute the one for which most equations of type (8) are available. Previously computed matrices X_{ik} can be involved at every successive iteration of this procedure.

4.2 Recovery of Scale Factors

The reduced measurement matrix $\mathcal{X}'$ in (7) is, in the absence of noise, of rank 3, being the product of a matrix of three columns and a matrix of three rows. This, however, only holds if a correct set of scale factors λ_{ik} is used. For other problems, these are often nontrivial to compute, see, e.g., [28], [14]. In our case, however, this turns out to be rather simple.

Let us first write $\mathrm{A}_i = \mathrm{K}_i\mathrm{R}_i$ and $\mathrm{B}_k = \mathrm{S}_k\mathrm{L}_k$. What we know is that (in the absence of noise) there exist matrices $\mathrm{A}_i, i = 1..m$ and $\mathrm{B}_k, k = 1..n$ such that: $\forall i,k : \mathrm{X}_{ik} \sim \mathrm{A}_i\mathrm{B}_k$. Since this equation is valid up to scale only, we also have: $\forall i,k : \mathrm{X}_{ik} \sim (a_i\mathrm{A}_i)(b_k\mathrm{B}_k)$ for any nonzero scale factors $a_i, i = 1..m$ and $b_k, k = 1..n$. Consequently, this is also true for the scale factors a_i and b_k that satisfy:

$$\det(a_i\mathrm{A}_i) = \det(b_k\mathrm{B}_k) = 1.$$

Note that we do not need to know these scale factors; it is sufficient to know they exist!

Hence, there exist scale factors a_i and b_k with:

$$\forall i,k : \mathrm{X}_{ik} \sim a_ib_k\mathrm{A}_i\mathrm{B}_k, \tag{9}$$

$$\forall i,k : \det(a_ib_k\mathrm{A}_i\mathrm{B}_k) = \det(a_i\mathrm{A}_i)\det(b_k\mathrm{B}_k) = 1. \tag{10}$$

As for the sought for scale factors λ_{ik}, we use those that give $\det(\lambda_{ik}\mathrm{X}_{ik}) = 1$. They are computed as:

$$\lambda_{ik} = (\det \mathrm{X}_{ik})^{-1/3}.$$

Due to (9), we have $\lambda_{ik}\mathrm{X}_{ik} \sim a_ib_k\mathrm{A}_i\mathrm{B}_k$ and, since the determinants of both sides of this equation are equal (they are both equal to 1, cf. the definition of λ_{ik} and (10)), the equation not only holds up to scale, but component-wise:[3]

$$\forall i,k : \lambda_{ik}\mathrm{X}_{ik} = (a_i\mathrm{A}_i)(b_k\mathrm{B}_k).$$

2. Compare with the analogous method in [18].
3. Two nonsingular 3×3 matrices that are equal up to scale and whose determinants are equal are also equal component-wise.

This means that the measurement matrix in (7), with the scale factors λ_{ik} as described here, is of rank 3: It is the product of one matrix of three columns (the $a_i\mathrm{A}_i$ stacked on top of each other) and one of three rows (the $b_k\mathrm{B}_k$ side-by-side).

In the following, we assume that the X_{ik} are already scaled to unit determinant, i.e., that $\lambda_{ik} = 1$. Equation (7) becomes:

$$\underbrace{\begin{bmatrix} \mathrm{X}_{11} & \cdots & \mathrm{X}_{1n} \\ \vdots & \ddots & \vdots \\ \mathrm{X}_{m1} & \cdots & \mathrm{X}_{mn} \end{bmatrix}}_{\mathcal{X}'_{3m\times 3n}} = \underbrace{\begin{bmatrix} a_1\mathrm{K}_1\mathrm{R}_1 \\ \vdots \\ a_m\mathrm{K}_m\mathrm{R}_m \end{bmatrix}}_{\mathcal{M}'_{3m\times 3}} \underbrace{\begin{bmatrix} b_1\mathrm{S}_1\mathrm{L}_1 & \cdots & b_n\mathrm{S}_n\mathrm{L}_n \end{bmatrix}}_{\mathcal{S}'_{3\times 3n}}. \tag{11}$$

The scale factors a_i and b_k do not matter for now; all that counts is that they exist and that the measurement matrix $\mathcal{X}'$ containing the normalized X_{ik} is of rank 3 at most and can thus be factorized as shown below.

4.3 Factorization

As usual, we use the SVD (Singular Value Decomposition) to obtain the low-rank factorization of the measurement matrix. Let the SVD of $\mathcal{X}'$ be given as:

$$\mathcal{X}'_{3m\times 3n} = \mathrm{U}_{3m\times 3n}\Sigma_{3n\times 3n}\mathrm{V}^\top_{3n\times 3n}.$$

The diagonal matrix Σ contains the singular values of $\mathcal{X}'$: $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{3n}$. In the absence of noise, $\mathcal{X}'$ is of rank 3 at most and $\sigma_4 = \cdots = \sigma_{3n} = 0$. If noise is present, $\mathcal{X}'$ is of full rank in general. Setting all singular values to zero, besides the three largest ones, leads to the best rank-3 approximation of $\mathcal{X}'$ (in the sense of the Frobenius norm).

In the following, we consider the rank-3 approximation of $\mathcal{X}'$ (for ease of notation, we denote this also as $\mathcal{X}'$):

$$\mathcal{X}' = \mathrm{U}_{3m\times 3n}\ \mathrm{diag}(\sigma_1, \sigma_2, \sigma_3, 0, \ldots, 0)\ \mathrm{V}^\top_{3n\times 3n}.$$

In the matrix product on the right, only columns of U and rows of $\mathrm{V}^\top$ corresponding to nonzero σ_j contribute. Hence:

$$\mathcal{X}' = \mathrm{U}'_{3m\times 3}\ \mathrm{diag}(\sigma_1, \sigma_2, \sigma_3)\ \left(\mathrm{V}'^\top\right)_{3\times 3n},$$

where U′ (resp. V′) consists of the first three columns of U (resp. V). Let us define $\mathrm{U}'' = \mathrm{U}'\,\mathrm{diag}(\sqrt{\sigma_1}, \sqrt{\sigma_2}, \sqrt{\sigma_3})$ and $\mathrm{V}'' = \mathrm{V}'\,\mathrm{diag}(\sqrt{\sigma_1}, \sqrt{\sigma_2}, \sqrt{\sigma_3})$. Thus, we have: $\mathcal{X}' = \mathrm{U}''\mathrm{V}''^\top$. This represents a decomposition of the measurement matrix $\mathcal{X}'$ into a product of a matrix of three columns (U'') with a matrix of three rows ($\mathrm{V}''^\top$). Note, however, that this decomposition is not unique. For any nonsingular 3×3 matrix T, the following is also a valid decomposition:

$$\mathcal{X}' = \left(\mathrm{U}''\mathrm{T}^{-1}\right)\left(\mathrm{T}\mathrm{V}''^\top\right).$$

Making the link with (11), we obtain:

$$\begin{bmatrix} a_1\mathrm{K}_1\mathrm{R}_1 \\ \vdots \\ a_m\mathrm{K}_m\mathrm{R}_m \end{bmatrix} \begin{bmatrix} b_1\mathrm{S}_1\mathrm{L}_1 & \cdots & b_n\mathrm{S}_n\mathrm{L}_n \end{bmatrix} = \left(\mathrm{U}''\mathrm{T}^{-1}\right)\left(\mathrm{T}\mathrm{V}''^\top\right). \tag{12}$$

Let us decompose matrices U'' and V'' in 3×3 submatrices: $\mathrm{U}''^\top = [\mathrm{U}_1^\top \cdots \mathrm{U}_m^\top]$ and $\mathrm{V}''^\top = [\mathrm{V}_1^\top \cdots \mathrm{V}_n^\top]$. Equation (12) thus becomes:

$$\begin{bmatrix} a_1\mathrm{K}_1\mathrm{R}_1 \\ \vdots \\ a_m\mathrm{K}_m\mathrm{R}_m \end{bmatrix} \begin{bmatrix} b_1\mathrm{S}_1\mathrm{L}_1 & \cdots & b_n\mathrm{S}_n\mathrm{L}_n \end{bmatrix} = \begin{bmatrix} \mathrm{U}_1\mathrm{T}^{-1} \\ \vdots \\ \mathrm{U}_m\mathrm{T}^{-1} \end{bmatrix} \begin{bmatrix} \mathrm{T}\mathrm{V}_1^\top & \cdots & \mathrm{T}\mathrm{V}_n^\top \end{bmatrix}. \tag{13}$$

How to estimate T is explained in Section 4.4. Once a correct estimate is given, we can directly extract the matrices $\mathrm{A}_i = a_i\mathrm{K}_i\mathrm{R}_i$ and $\mathrm{B}_k = b_k\mathrm{S}_k\mathrm{L}_k$ from which, in turn, the individual rotation and calibration matrices can be recovered by Cholesky or QR-decompositions. The Cholesky decomposition of $\mathrm{A}_i\mathrm{A}_i^\top$, e.g., results in an upper triangular matrix $\mathrm{M}_i = a_i\mathrm{K}_i$. Based on the requirement $K_{i,33} = 1$, we can compute the unknown scale factor a_i as $a_i = M_{i,33}$. The calibration matrix is finally obtained as[4] $\mathrm{K}_i = \frac{1}{a_i}\mathrm{M}_i$.

As for the parallelepipeds, there is no constraint similar to $K_{i,33} = 1$ on the entries of their calibration matrices L_k. Hence, we can compute them only up to the unknown scale factors b_k. This means that we can compute the *shape* of each parallelepiped, but not (yet) their *size* (or volume). In Section 6, we explain how to compute their (relative) size.

We now briefly discuss the structure and geometric signification of matrix T. Note that T actually represents the nontranslational part of a 3D affine transformation (its upper left 3×3 submatrix). This is just another expression of the previously mentioned fact that, due to the observation of parallelepipeds, we directly have an affine reconstruction (of scene and cameras).

The matrix T can only be computed up to an arbitrary rotation and scale: For any rotation matrix R and scale factor s, $\mathrm{T}' = s\mathrm{R}\mathrm{T}$ cannot be distinguished from T in the factorization since $\mathrm{T}'^{-1}\mathrm{T}' \sim \mathrm{T}^{-1}\mathrm{T}$. This ambiguity is natural and expresses the fact that the global Euclidean reference frame for the reconstruction of parallelepipeds and cameras can be chosen arbitrarily. Without loss of generality, we may thus assume that T is upper triangular. This highlights the fact that our estimation problem has only five degrees of freedom (six parameters for an upper triangular 3×3 matrix minus one for the free scale), which can also be explained in more geometric terms: As explained previously, our problem is somewhat equivalent to self-calibration with known affine structure. The five degrees of the problem can thus be interpreted as the coefficients of the absolute conic on the plane at infinity.

4.4 Disambiguating the Factorization

We now deal with the estimation of the unknown transformation T appearing in (13). As will be seen below and as is often the case in self-calibration problems, it is simpler to not directly estimate T, but the symmetric and

4. In overconstrained situations, the computed calibration matrices will not, in general, exactly satisfy the constraints used for their computation. The best way of dealing with this would be a constrained nonlinear optimization.

positive definite 3×3 matrix Z defined as: $\mathrm{Z} = \mathrm{T}^\top \mathrm{T}$. (We may observe that Z represents the absolute conic on the plane at infinity.) Once Z is estimated, T may be extracted from it using Cholesky decomposition. As described above, T is defined up to a rotation and scale, so the upper triangular Cholesky factor of Z can directly be used as the estimate for T.

The matrix Z (and, thus, T) can be estimated in various ways, using any information about the cameras or the parallelepipeds, e.g., prior knowledge of the relative positioning of some entities. Here, we concentrate on exploiting prior information of the intrinsic parameters of cameras and parallelepipeds. In the following, we consider two types of information, first for cameras and then for parallelepipeds:

- knowledge of the actual value of some intrinsic parameter for some camera or parallelepiped,
- knowledge that two or more cameras (or parallelepipeds) have the same value for some intrinsic parameter. We also sometimes speak of "constant" intrinsic parameters.

4.4.1 Using Information on Camera Intrinsics

From (13), we have: $a_i \mathrm{K}_i \mathrm{R}_i = \mathrm{U}_i \mathrm{T}^{-1}$. Due to the orthogonality of R_i, we get:

$$a_i^2 \underbrace{\mathrm{K}_i \mathrm{K}_i^\top}_{\omega_i^{-1}} = \mathrm{U}_i \underbrace{\mathrm{T}^{-1}\mathrm{T}^{-\top}}_{\mathrm{Z}^{-1}} \mathrm{U}_i^\top.$$

Neglecting the unknown scale factor a_i and taking the inverse of both sides of the equation, we obtain (note that the U_i are not orthogonal in general):

$$\omega_i \sim \mathrm{U}_i^{-\top} \mathrm{Z} \mathrm{U}_i^{-1}. \tag{14}$$

We are now ready to formulate constraints on Z based on information on the cameras' intrinsics.

Known Values of Camera Intrinsics. Knowing the aspect ratio and principal point coordinates of a camera i and substituting ω_i according to (14) and (1), the following linear constraints on Z can be written:

$$\begin{aligned}
\tau_i^2 \left(\mathrm{U}_i^{-\top} \mathrm{Z} \mathrm{U}_i^{-1}\right)_{11} - \left(\mathrm{U}_i^{-\top} \mathrm{Z} \mathrm{U}_i^{-1}\right)_{22} &= 0 \\
u_{i,0} \left(\mathrm{U}_i^{-\top} \mathrm{Z} \mathrm{U}_i^{-1}\right)_{11} + \left(\mathrm{U}_i^{-\top} \mathrm{Z} \mathrm{U}_i^{-1}\right)_{13} &= 0 \\
v_{i,0} \left(\mathrm{U}_i^{-\top} \mathrm{Z} \mathrm{U}_i^{-1}\right)_{22} + \left(\mathrm{U}_i^{-\top} \mathrm{Z} \mathrm{U}_i^{-1}\right)_{23} &= 0.
\end{aligned}$$

A known value of the focal length α_v can only be used to formulate linear equations if all the other intrinsics are also known. In such a fully calibrated case, other algorithms [25] might be better suited, so we neglect that case in the following.

Constant Camera Intrinsics. In the case when two cameras i and j are known to have the same, yet unknown value for one intrinsic parameter, we in general obtain *quadratic* equations on Z. For example, the assumption of equal aspect ratios leads to the quadratic equation:

$$\left(\mathrm{U}_i^{-\top} \mathrm{Z} \mathrm{U}_i^{-1}\right)_{11} \left(\mathrm{U}_j^{-\top} \mathrm{Z} \mathrm{U}_j^{-1}\right)_{22} = \left(\mathrm{U}_j^{-\top} \mathrm{Z} \mathrm{U}_j^{-1}\right)_{11} \left(\mathrm{U}_i^{-\top} \mathrm{Z} \mathrm{U}_i^{-1}\right)_{22}.$$

The situation is different if *all* intrinsic parameters of two (or more) views are known to be identical. In that case, we can obtain linear equations instead of quadratic ones, as shown in [8]: The matrices U^i are first scaled such as to have unit determinant. Then, we can write the following component-wise matrix equality between any pair (i, j) of views:

$$\mathrm{U}_i^{-\top} \mathrm{Z} \mathrm{U}_i^{-1} - \mathrm{U}_j^{-\top} \mathrm{Z} \mathrm{U}_j^{-1} = 0_{3\times 3}.$$

This represents six *linear* equations on Z for each pair of views, among which four are independent.

4.4.2 Information on Parallelepipeds

From (13), we have: $b_k \mathrm{S}_k \mathrm{L}_k = \mathrm{T} \mathrm{V}_k^\top$. Due to the orthogonality of S_k, we get:

$$b_k^2 \underbrace{\mathrm{L}_k^\top \mathrm{L}_k}_{\mu_k} = \mathrm{V}_k \underbrace{\mathrm{T}^\top \mathrm{T}}_{\mathrm{Z}} \mathrm{V}_k^\top.$$

Neglecting the unknown factor b_k:

$$\mu_k \sim \mathrm{V}_k \mathrm{Z} \mathrm{V}_k^\top.$$

Knowledge of parallelepiped intrinsics can be used in analogous ways as for camera parameters. For example, suppose we know the length ratio of two parallelepiped edges $r_{uv} = \frac{l_u}{l_v}$. Referring to (2), we get the following *linear* equation on Z:

$$r_{k,uv}^2 \mu_{k,vv} - \mu_{k,uu} = r_{k,uv}^2 \left(\mathrm{V}_k \mathrm{Z} \mathrm{V}_k^\top\right)_{vv} - \left(\mathrm{V}_k \mathrm{Z} \mathrm{V}_k^\top\right)_{uu} = 0.$$

Similarly, the assumption that θ_{uv} is a right angle, i.e., $\cos\theta_{uv} = 0$, also gives a *linear equation*:

$$\mu_{k,uv} = \left(\mathrm{V}_k \mathrm{Z} \mathrm{V}_k^\top\right)_{uv} = 0.$$

A known angle θ_{uv} that is not a right angle does not lead to a linear, but a *bilinear* equation [44].

Like for cameras, *quadratic* equations may be derived from assumptions about two or more parallelepipeds having the same, yet unknown value for some intrinsic parameter. Also, two parallelepipeds having the same shape give a set of *linear* equations on Z, even if the parallelepipeds are of different size. Equal size of parallelepipeds gives an additional *linear* equation, but which constrains relative pose rather than intrinsic parameters.

Currently, we only exploit constraints on individual parallelepipeds (right angles and length ratios) since they are easier to provide for the user.

4.5 Complete Algorithm

1. Estimate the canonical projection matrices $\tilde{\mathrm{X}}_{ik}$.
2. Compute missing X_{ik}.
3. Normalize the X_{ik} to unit determinant.
4. Construct the measurement matrix and compute its SVD.
5. From the SVD, extract the matrices U_i and V_k.
6. Establish a linear equation system on Z based on prior knowledge of intrinsic parameters of cameras and parallelepipeds and solve it to least squares.
7. If Z is positive, definite extract T from Z using Cholesky decomposition.
8. Extract the $\mathrm{K}_i, \mathrm{R}_i, \mathrm{L}_k, \mathrm{S}_k$ from the $\mathrm{U}_i \mathrm{T}^{-1}$ and the $\mathrm{T}\mathrm{V}_k^\top$ using, e.g., QR-decomposition. Note that, at this stage, the L_k can only be recovered up to

TABLE 1
Structures of ω Depending on Prior Knowledge

Known camera parameters			
(A) Zero skew	(B) τ	(C) u_0, v_0	(D) τ, u_0, v_0
$\begin{pmatrix} a & 0 & d \\ 0 & b & e \\ d & e & c \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & d \\ 0 & 1 & e \\ d & e & c \end{pmatrix}$	$\begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & c \end{pmatrix}$

scale, i.e., the parallelepipeds' (relative) sizes remain undetermined.

This algorithm allows us to calibrate a set of cameras using very little prior knowledge. Indeed, as mentioned in this section, all constraints provided by knowledge of cameras and parallelepipeds can be expressed in terms of the five independent parameters of the matrix Z. Thus, to calibrate the whole system, it is in general sufficient to know values of a total of only five intrinsic parameters of cameras or parallelepipeds. That is why, in practice, we only use the associated linear equations. In most cases, they are sufficient to find a unique solution. In some minimal cases, when the available linear constraints are insufficient, quadratic equations might be used to find a unique solution or a finite set of solutions.

5 Singularities

Many calibration or self-calibration algorithms are subject to more or less severe singularities, i.e., there exist situations where the algorithm is bound to fail. Furthermore, even in situations that are not exactly singular, but close to a singularity, the results usually become very unstable. In this section, we examine the singularities for the linear calibration algorithm described in Section 4.5. We separately study the singularities for a parallelepiped being seen by one and multiple cameras.

5.1 One Parallelepiped in a Single View

We have studied all possible combinations of a priori knowledge, on both camera and parallelepiped intrinsic parameters leading to linear equations (see Sections 4.1.1 and 4.4.2). We first formulate the meaning of a singularity in terms of the ingredients of the calibration algorithm. The existence of a singularity in our case means that (4) has more than one solution for ω and μ conforming to all available a priori information, i.e., that there is at least one solution different from the true one. It is easy to show that the existence of a singularity does not depend on the relative *position* of the camera and the parallelepiped, only on the relative *orientation* and the a priori knowledge on camera and parallelepiped intrinsic parameters. Proofs for the following results are given in [47].

Table 1 explains the four considered cases of different prior knowledge on camera intrinsics. Table 2 shows all singularities for nearly all combinations of known camera and parallelepiped parameters. Singularities are explained in geometric terms by describing the relative orientation of the parallelepiped with respect to the camera. In the following paragraphs, we give a few comments on different cases of prior knowledge on the parallelepiped. Several singular situations that might occur in practice are illustrated in Fig. 3.

TABLE 2
Singular Relative Orientation (One Parallelepided in One View) for Various Combinations of Prior Knowledge

Case	Conditions for singularity
A-3-1	$\mathbf{u}$ is orthogonal to the $\mathbf{x}$ or $\mathbf{y}$ camera axis
B-3-1	$\mathbf{u}$ is parallel to the optical axis $\mathbf{z}$
C-3-1	$\mathbf{u}$ is parallel to any of the three camera axes
D-3-1	$\mathbf{u}$ is parallel to the optical axis
A-3-0	always (3 constraints for 4 camera intrinsics)
B-3-0	any parallelepiped edge is parallel to the image plane
C-3-0	any parallelepiped edge is parallel to any camera axis
D-3-0	any parallelepiped edge is parallel to the optical axis
A-2-2	no intuitive geometrical interpretation
B-2-2	($\mathbf{u} \parallel \Pi$) and ($\mathbf{w} \parallel$ optical axis or Π)
C-2-2	($\mathbf{u} \parallel \mathbf{x}$ or $\mathbf{y}$ axis and $\angle(\mathbf{w}, \Pi) = 45°$) or ($\mathbf{u} \parallel \mathbf{z}$ and $\mathbf{w} \parallel \Pi$ and $\angle(\mathbf{w}, \mathbf{x}) = \angle(\mathbf{w}, \mathbf{y}) = 45°$)
D-2-2	never!
A-2-1	always (three constraints for four camera intrinsics)
B-2-1	$\mathbf{u} \parallel \Pi$
C-2-1	($\mathbf{u} \parallel$ to either camera axis) or ($\mathbf{u}$ and $\mathbf{w} \parallel \Pi$) or ($\mathbf{u}$ and $\mathbf{w} \perp$ to $\mathbf{x}$ or $\mathbf{y}$ camera axis)
D-2-1	$\mathbf{u}$ and $\mathbf{w}$ are parallel to the image plane
A-2-0	always (two constraints for four camera intrinsics)
B-2-0	always (two constraints for three camera intrinsics)
C-2-0	$\mathbf{u}$ orthogonal to the $\mathbf{x}$ or $\mathbf{y}$ camera axis or $\parallel \Pi$
D-2-0	$\mathbf{u}$ parallel to image plane or to optical axis

Cases are denoted X-Y-Z, *where* $\mathrm{X} \in \{\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D}\}$ *refers to Table 1 and* $\mathrm{Y(Z)}$ *is the number of known right angles (length ratios).* Π *means the image plane.*

1. *Three right angles, two length ratios (cases *-3-2 in Table 2).* In this case, the Euclidean structure of the parallelepiped is completely given (up to scale) and it can be used as a classical calibration object. There are singularities proper to the use of a parallelepiped, but, of course, the generic singularities described in [48] apply here, too.
2. *Three right angles, one length ratio (cases *-3-1).* In Table 2, $\mathbf{u}$ represents the direction of the parallelepiped's edge which is not "involved" in the known length ratio.
3. *Two right angles (cases *-2-*).* Here, the parallelepiped can be visualized as built around two rectangles sharing an edge $\mathbf{u}$. In Table 2, $\mathbf{w}$ is one of the edges not parallel to $\mathbf{u}$.

5.2 One Parallelepiped in Multiple Views

Two observations are useful to characterize the singularities in the case when one parallelepiped is seen in multiple images:

Fig. 3. Examples of singular and nonsingular configurations for calibration based on a parallelepiped with three right angles (the house's main body). Left: The parallelepiped's vertical edge is parallel to the camera's y-axis; this configuration is singular if the camera aspect ratio and principal point are not given (cf. cases B-3-0 and C-3-0 in Table 2. Middle: The parallelepiped's vertical edge is parallel to the image plane; this configuration is singular if the camera's principal point is not given (case B-3-0 in Table 2). Right: A rotation of the camera as shown here removes these singularities.

- The way the canonic parallelepiped projection matrix $\tilde{\mathrm{X}}$ is computed implies that there is only an affine ambiguity in the calibration process. Thus, the singularities of calibration of images viewing one parallelepiped are equivalent to singularities of generic self-calibration when the plane at infinity is known.
- It is natural to suppose that the prior knowledge of the intrinsic parameters is the same for all cameras used, thus all matrices ω_i are assumed here to belong to the same among the groups defined in Table 1.

These observations make it possible to adapt the studies on critical motions for self-calibration. In particular, we use the results presented in [49] for scenarios with a known plane at infinity. Depending on the type of knowledge about the camera (cf. Table 1), the following rotations are singular for calibration:

A. Always critical with fewer than five cameras. Critical motions for five or more cameras are hard to describe.
B. Critical if the cameras' optical axes point in at most two different directions.
C. Critical if one axis of each camera is pointing in some common direction.
D. Critical if the optical axes of all cameras point in the same direction.

Results for the cases A, B, D were given in [49] and the proof of case C is given in [47].

6 Estimating Position and Size

In this section, we propose an algorithm for the estimation of the (relative) positions of the cameras and parallelepipeds, as well as the (relative) sizes of the parallelepipeds. After Section 4, this concludes our complete method for intrinsic and extrinsic calibration.

Consider (5):

$$\lambda_{ik}\tilde{\mathrm{X}}_{ik} = \mathrm{K}_i(\,\mathrm{R}_i \quad \mathbf{t}_i\,)\begin{pmatrix}\mathrm{S}_k & \mathbf{v}_k\\ \mathbf{0}^\top & 1\end{pmatrix}\tilde{\mathrm{L}}_k.$$

The leading 3×3 subpart of the two sides of the equation were used in Section 4 to compute the intrinsic camera parameters K_i and the rotation matrices R_i and S_k. The parallelepipeds' intrinsic parameters L_k were computed up to scale only, i.e., up to the "size" of the parallelepipeds.

Let us consider this in detail. In the following, we suppose that the matrices $\tilde{\mathrm{X}}_{ik}$ are already scaled such that the submatrices X_{ik} have unit determinant, as in Section 4, i.e., $\lambda_{ik}=1$. Let $\bar{\mathrm{K}}_i$ and $\bar{\mathrm{L}}_k$ be the calibration matrices scaled to unit determinant. We know all matrices in the following equation: $\mathrm{X}_{ik} = \bar{\mathrm{K}}_i\mathrm{R}_i\mathrm{S}_k\bar{\mathrm{L}}_k$.

What we don't know is the size s_k of the parallelepipeds. Let us observe the following:

$$\tilde{\mathrm{L}}_k = \begin{pmatrix}s_k\bar{\mathrm{L}}_k & \mathbf{0}\\ \mathbf{0}^\top & 1\end{pmatrix} \sim \begin{pmatrix}\bar{\mathrm{L}}_k & \mathbf{0}\\ \mathbf{0}^\top & 1/s_k\end{pmatrix}.$$

We may now rewrite (5):

$$\tilde{\mathrm{X}}_{ik} = \bar{\mathrm{K}}_i(\,\mathrm{R}_i \quad \mathbf{t}_i\,)\begin{pmatrix}\mathrm{S}_k & \mathbf{v}_k\\ \mathbf{0}^\top & 1\end{pmatrix}\begin{pmatrix}\bar{\mathrm{L}}_k & \mathbf{0}\\ \mathbf{0}^\top & 1/s_k\end{pmatrix}.$$

Let $\mathbf{x}_{ik}$ be the fourth column of $\tilde{\mathrm{X}}_{ik}$. We have the following equation:

$$\mathbf{x}_{ik} = \bar{\mathrm{K}}_i(\,\mathrm{R}_i \quad \mathbf{t}_i\,)\begin{pmatrix}\mathrm{S}_k & \mathbf{v}_k\\ \mathbf{0}^\top & 1\end{pmatrix}\begin{pmatrix}\mathbf{0}\\ 1/s_k\end{pmatrix} = \frac{1}{s_k}\bar{\mathrm{K}}_i(\mathrm{R}_i\mathbf{v}_k + \mathbf{t}_i).$$

From this, we get an equation that is linear in all unknowns (s_k, $\mathbf{t}_i$, and $\mathbf{v}_k$):

$$s_k\mathbf{x}_{ik} - \bar{\mathrm{K}}_i\mathrm{R}_i\mathbf{v}_k - \bar{\mathrm{K}}_i\mathbf{t}_i = \mathbf{0}.$$

The unknowns can be computed using linear least squares: minimizing the sum of the squared L_2 norms of the vectors on the left hand side of the above equation, over all camera-parallelepiped pairs. The estimates for the s_k, $\mathbf{t}_i$, and $\mathbf{v}_k$ are, of course, defined up to a single global scale. Note that, at this stage, missing data are not an issue any more, contrary to the computations in Section 4.

7 3D Reconstruction

The presented calibration approach is well adapted to interactive 3D reconstruction from a few images. It has a major advantage over other methods: simplicity. Indeed, only a small amount of user interaction is needed for both calibration and reconstruction: A few points must be picked in the image to define the primitives' image positions. It thus seems to be an efficient and intuitive way to build models from images of any type, in particular from images taken from the Internet for which no information about the camera is known. For the reconstruction of points not belonging to the parallelepiped, we use an iterative approach described in [34]. This approach is actually independent from the calibration method, although it uses the same input in the first step. It allows us to propagate the information on points, lines, and planes defining the model. All the accessible information is processed simultaneously. Interestingly, it allows 3D models to be computed from nonoverlapping photographs (see, e.g., Fig. 10).

The following section illustrates this approach with results obtained by solving linear systems only. Note that, in order to refine the results, nonlinear optimization, taking into account prior information, might be applied.

8 Experimental Results

8.1 Synthetic Data

The main goal of our experiments with synthetic data is to study the performance of the calibration algorithm in the proximity of singular configurations. In this paper, we only report on experiments on the minimal case: One parallelepiped seen in one camera. Additional experiments, evaluating calibration results in the proximity of singular positions with respect to different numbers of parallelepipeds, as well as different types of prior information, are described in [47].

Tests were performed with synthetic 600×400 images, taken by a camera with the following intrinsic parameters:

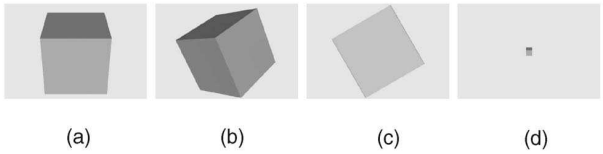


Fig. 4. Parallelepiped orientations in the experiment. (a) Initial orientation (x axes of parallelepiped and camera are parallel), (b) intermediate orientation, (c) final orientation (the y and z axes of the parallelepiped are parallel to the image plane), (d) minimal parallelepiped size.

$(\alpha_u, \alpha_v, s, u_0, v_0) = (1,000, 1,000, 0, 300, 200)$. Parallelepiped parameters were varying over the different tests. The most important parameter of the experiments is the relative orientation parallelepiped-camera. For a given orientation, six parallelepiped vertices were projected into the images and random Gaussian noise was added to image points (for the presented results, noise was of standard deviation 1 pixel). For a given setting (relative orientation, standard deviation of noise, etc.), 100 such data sets were created randomly and used as input for calibration. Calibration was considered to have failed if any of the estimated matrices ω or μ were not positive definite (in that case, K or L cannot be retrieved). We give results by indicating the number of failures, as well as median values for estimated parameters (computed using valid calibration results only). Prior information used was: The parallelepiped has only right angles and known camera parameters were $(s, \tau) = (0, 1)$ (i.e., case B-3-0 in Section 5). This is one of the minimal cases for the calibration.

Tests were performed for different orientations of the parallelepiped, as illustrated in Fig. 4. Orientation varies continuously from that shown in Fig. 4a (x axes of parallelepiped and camera are parallel) to that of Fig. 4c (the y and z axes of the parallelepiped are parallel to the image plane). The continuous rotation between the two positions is parameterized by an angle ranging from $0°$ (Fig. 4a) to $90°$ (Fig. 4c), see the horizontal axis of the graphs in Fig. 5. According to Section 5, both extremal orientations are singular. We also varied the size of the parallelepiped: Maximal and minimal sizes are shown in Fig. 4a and Fig. 4d, respectively.

The results of the calibration method described here are compared to calibration based on vanishing points [19]; vanishing points were determined using the method [50]. In the case tested here, both methods use the same constraints (three right angles); the difference is that [19] uses individually estimated vanishing points, whereas our method, via the estimation of the canonical projection matrix, accounts for the fact that the three vanishing points of a parallelepiped are not fully independent.

Fig. 5 shows the number of successful calibrations, the median of estimated values for α_v, and errors on the estimated pose (median of errors on the angle of the estimated rotational part of pose). Results are shown with respect to the orientation of the parallelepiped (horizontal axis of graphs, see above) and its size (vertical axis, unit is coverage of image by parallelepiped, from 5 percent to 50 percent).

Results are shown for both the parallelepiped-based approach (first column) and the vanishing point approach

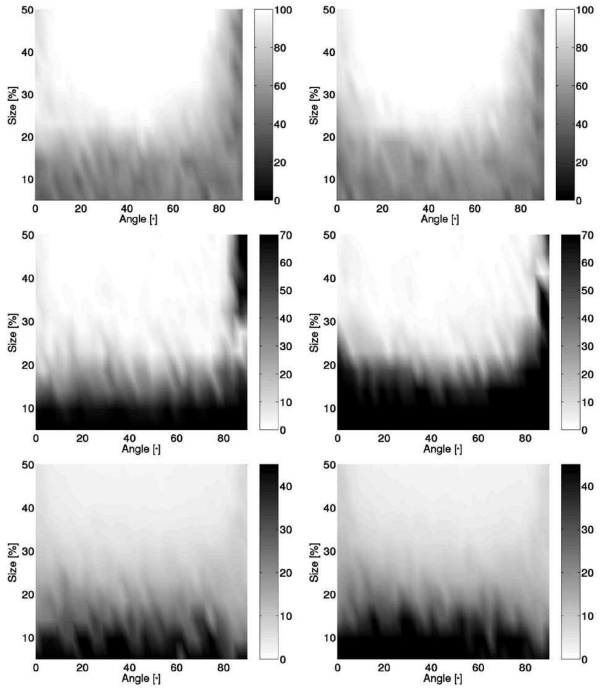


Fig. 5. Calibration results as a function of the size and relative camera-parallelepiped rotation angle. The two columns correspond to parallelepiped-based and vanishing point-based calibration, respectively. From top to bottom, the graphs show: 1) the number of successful calibrations, 2) median values of relative errors on the estimation of α_v, 3) median rotation errors.

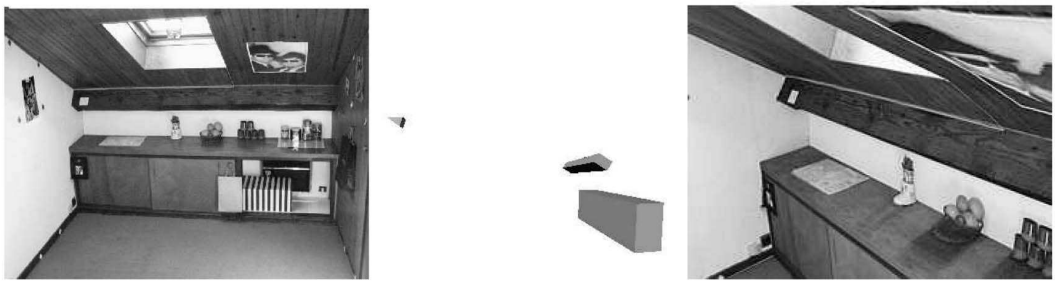
(second column). The effect of singular cases is clearly visible in the upper two rows: Calibration often fails for orientations within $10°$ of the singular ones. However, for the intermediate range of orientations, the relative error of calibration is smaller than 5 percent for both methods, when the parallelepiped covers more than 20 percent of the image. Results in case of successful calibration are slightly better for the parallelepiped method.

8.2 Results on Real Scenes

We present 3D reconstruction results of our methods for indoor and outdoor scenes. These examples correspond to situations where automatic methods are bound to fail: Small sets of images are used and occlusions are frequent. Each reconstruction was performed in two steps: First, one or more parallelepipeds were used to calibrate the intrinsic and extrinsic camera parameters; second, scene points and geometric constraints were used for the reconstruction (cf. Section 7). Results from single as well as multiple images are shown.

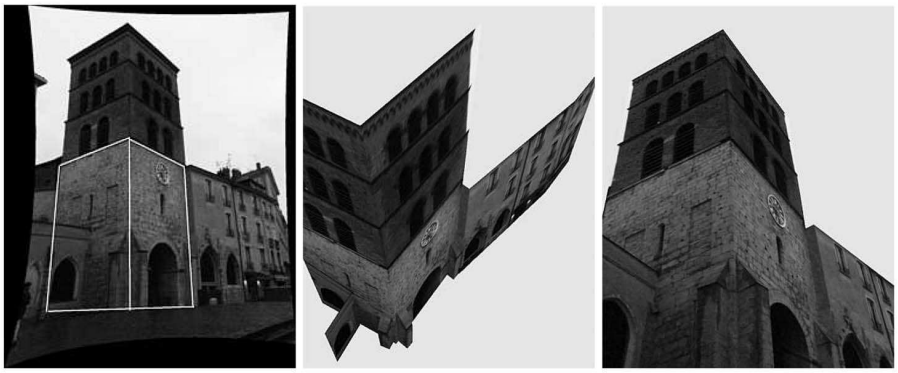
8.2.1 Kitchen Scene

Fig. 6a shows the original image used for the 3D reconstruction. The image was taken with a small focal length, leading, therefore, to a slight optical distortion which was not corrected here. Calibration was based on the cupboard in the central part of the image. Note that the camera was almost frontoparallel with respect to the cupboard and, thus, close to a singular situation. Prior information used for calibration was: right parallelepiped angles, zero camera skew, and



(a) (b) (c)

Fig. 6. Kitchen scene: (a) The original image, (b) the modeled parallelepipeds and camera pose, (c) textured model seen from a different viewpoint.



(a) (b) (c)

Fig. 7. Notre Dame square scene: (a) The original image, (b) and (c) screen-shots of the model obtained using the image on the left only.

principal point in the image center. The full model was reconstructed using 29 points, constrained by two parallelepipeds (the cupboard and the wooden belt), three parallelograms, and six collinearity and coplanarity constraints. Fig. 6b shows the reconstructed parallelepipeds as well as the camera pose. Due to the fact that a single image was used, only the shape of the parallelepipeds can be reconstructed at first, but not their relative size. This was then done using additional constraints such as coplanarity, which were also used to reconstruct the positions of additional points. Fig. 6c shows the complete textured model rendered from a new viewpoint.

8.2.2 Notre Dame Square Scene

In this section, we present reconstructions of the Notre Dame Square in Grenoble. Here, radial image distortion was corrected offline.

Reconstruction from one image. The image and the calibration parallelepiped are shown in Fig. 7a. Prior information used for calibration as: right parallelepiped angles, zero camera skew, and principal point in the image center. The final model is composed of 42 points, three parallelepipeds, four parallelograms, and four lines and planes. Rendered views of the model are shown in Fig. 7.

Reconstruction from multiple images. The sequence used for the reconstruction is composed of 15 images whose sizes vary from $768 \times 1,024$ to $960 \times 1,280$ pixels. Calibration was based on three parallelepipeds (shown in Fig. 8a, Fig. 8b, Fig. 8c, Fig. 8d). Prior information used was: right angles for parallelepipeds 1 and 2, zero camera skew, unit aspect ratios and centered principal points for all images. Parallelepiped 3 is relatively small in those images where both parallelepipeds 2 and 3 appear. Consequently, the estimation of its vertices is unstable and, thus, information about its intrinsic parameters was not used for calibration. Calibration was performed in two steps. First, the proposed linear factorization approach was applied. Second, the parameters of cameras and parallelepipeds obtained from the previous step were nonlinearly optimized by minimizing the reprojection error of vertices in a bundle adjustment.

Then, scene elements were added and reconstructed so that the final model is composed of 194 points, 19 planes, and 25 lines. The mean reprojection error over all the model points was about 8 pixels (only linear methods were used for reconstruction). As expected, the largest errors occurred in images calibrated using parallelepiped 3.

For comparison, an unconstrained bundle adjustment, using the Levenberg-Marquardt optimization method, was performed over all the model points and the camera focal lengths. This reduced the reprojection error to 2 pixels. It did not, however, reduce the small artifacts occurring in the final model.

The calibration primitives and cameras reconstructed using the factorization method, the parallelepiped-based nonliner optimization, and the point-based nonlinear optimization are shown, respectively, in Fig. 8e, Fig. 8f, Fig. 8g. Rendered views of the model reconstructed using the parallelepiped-based calibration are shown in Fig. 8h and Fig. 8j.

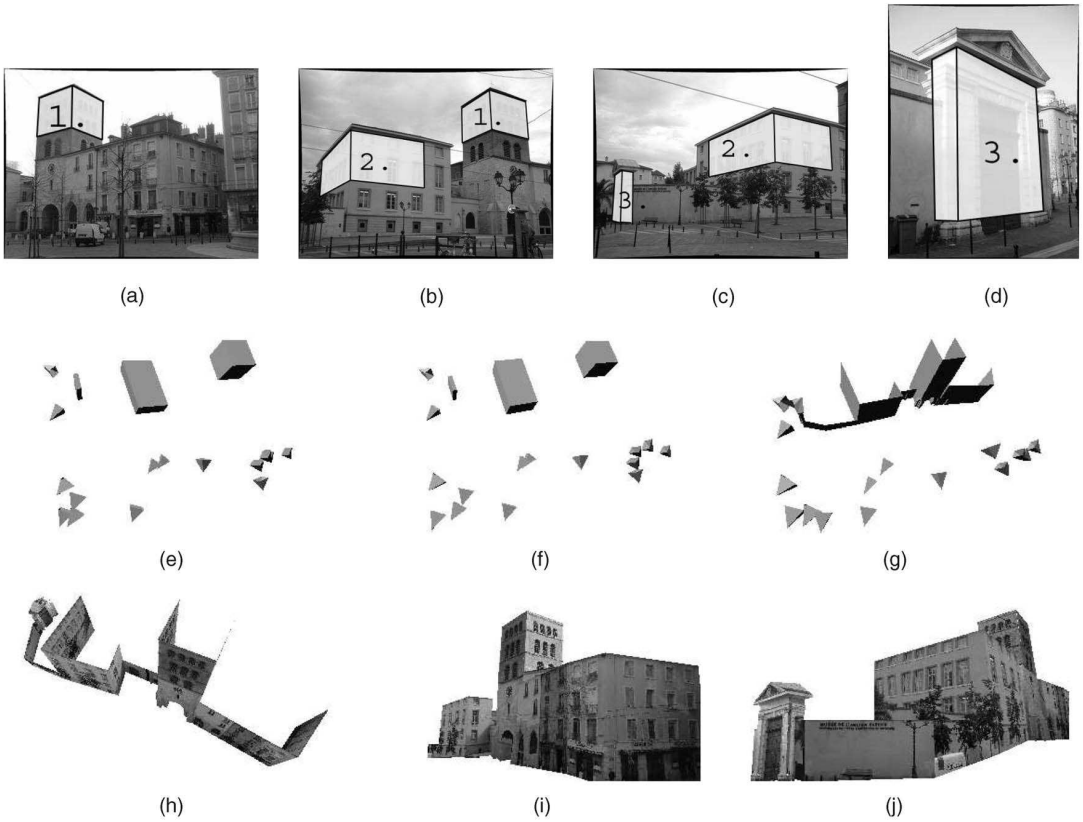


Fig. 8. Notre Dame Square scene: (a)-(d) Four images from the 15 used for the reconstruction. Parallelepipeds used for the reconstruction are marked in white. (e) Cameras and parallelepipeds as estimated by the proposed linear factorization method. (f) Camera and parallelepiped parameters after nonlinear optimization. (g) Cameras and 194 model points optimized by an unconstrained nonlinear method. (h)-(j) Synthetic viewpoints of the textured model.

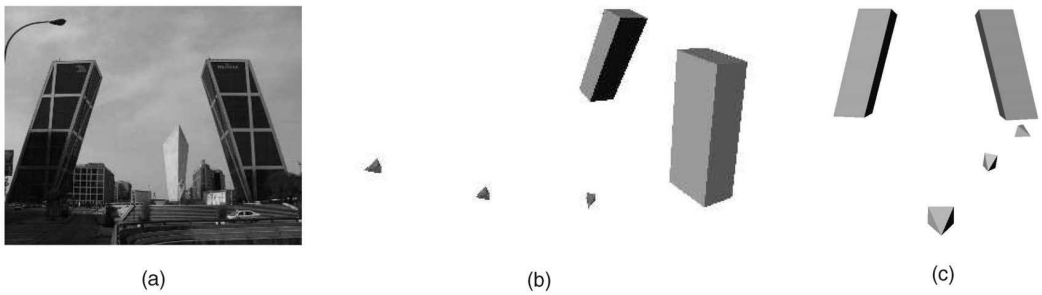


Fig. 9. Kio towers in Madrid: (a) The original image; (b) and (c) reconstructed model and camera poses.

8.2.3 Kio Towers

Reconstruction was based on three images and two calibration primitives. One of the images used for the reconstruction is shown in Fig. 9a. Information used for calibration was: two right angles in each tower, zero camera skew, unit aspect ratio, and centered principal point. The reconstructed cameras and primitives are shown in Fig. 9.

8.2.4 Opposite Viewpoints Scene

Fig. 10 shows the reconstruction of a modern building from two images taken from completely opposite viewpoints. The parallelepiped used for calibration and the estimated cameras' positions are shown in the two original images (Fig. 10a and Fig. 10b). In the first image, intersections of lines were computed to obtain the six points required to define a parallelepiped (see Fig. 10a). The parallelepiped and the cameras reconstructed by the factorization algorithm are shown in Fig. 10c. New viewpoints of the whole model, composed of 32 points, 13 parallelograms, and six planes, are shown in Fig. 10d, Fig. 10e, and Fig. 10f.

9 Conclusion

We have presented an approach for calibration, pose estimation, and 3D model acquisition from several uncalibrated images based on user-provided geometric constraints on the scene. Useful constraints, such as parallelism, coplanarity, and right angles, can often be

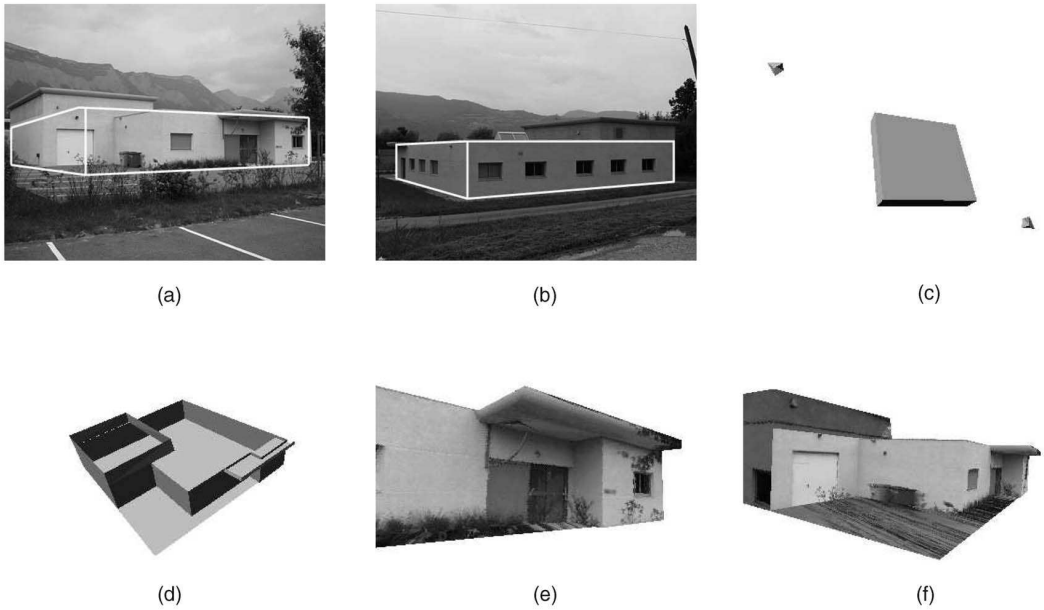


Fig. 10. Opposite viewpoints scene: (a), (b) The original images used for the reconstruction. (c) The reconstruction scenario with the computed model and the cameras' positions. (d), (e), (f) New viewpoints of the model.

nicely modeled via parallelepipeds. Especially, this allows us to couple together constraints between several neighboring scene primitives (points, lines, planes), which potentially brings about higher stability than only using constraints between pairs of primitives. The projections of parallelepipeds already encode the affine structure of the scene. Metric information (length ratios and angles) is then combined with prior information on camera parameters in a self-calibration type approach, performing complete calibration and pose estimation. This is formulated in a factorization framework. The usual problems of missing data and unknown scale factors are dealt with relatively easily and a satisfying solution can already be obtained with a small number of images and correspondences (starting from four correspondences per image pair or six per image and parallelepiped).

A detailed study on singular cases of this approach is also presented: Singularities are derived theoretically and the impact on the method's performance due to the proximity to singular configurations is shown by simulated experiments. Experiments with real images show that our calibration approach gives excellent initial results for general 3D model reconstruction methods.

We believe that an approach such as the one presented here is a useful tool for easily calibrating cameras using images of unknown though constrained scenes. Also, it allows us to efficiently obtain models of the global structure of scenes (including camera pose), which are good starting points for more automatic reconstruction methods.

Acknowledgments

This work was partially supported by the European project VISIRE (IST-1999-10756).

References

[1] O. Faugeras, "What Can Be Seen in Three Dimensions with an Uncalibrated Stereo Rig?" *Proc. European Conf. Computer Vision,* pp. 563-578, 1992.

[2] R. Tsai, "An Efficient and Accurate Camera Calibration Technique for 3D Machine Vision," *Proc. Conf. Computer Vision and Pattern Recognition,* pp. 364-374, 1986.

[3] S. Maybank and O. Faugeras, "A Theory of Self Calibration of a Moving Camera," *Int'l J. Computer Vision,* vol. 8, no. 2, pp. 123-151, 1992.

[4] B. Triggs, "Autocalibration and the Absolute Quadric," *Proc. Conf. Computer Vision and Pattern Recognition,* pp. 609-614, 1997.

[5] R. Hartley, "Euclidean Reconstruction from Uncalibrated Views," *Proc. DARPA-ESPRIT Workshop Applications of Invariants in Computer Vision,* pp. 187-202, 1993.

[6] M. Pollefeys and L. Van Gool, "A Stratified Approach to Metric Self-Calibration," *Proc. Conf. Computer Vision and Pattern Recognition,* pp. 407-412, 1997.

[7] P. Sturm, "Critical Motion Sequences for Monocular Self-Calibration and Uncalibrated Euclidean Reconstruction," *Proc. Conf. Computer Vision and Pattern Recognition,* pp. 1100-1105, 1997.

[8] R.I. Hartley, "Self-Calibration of Stationary Cameras," *Int'l J. Computer Vision,* vol. 22, no. 1, pp. 5-23, 1997.

[9] L. de Agapito, R. Hartley, and E. Hayman, "Linear Self-Calibration of a Rotating and Zooming Camera," *Proc. Conf. Computer Vision and Pattern Recognition,* pp. 15-21, 1999.

[10] M. Armstrong, A. Zisserman, and P. Beardsley, "Euclidean Structure from Uncalibrated Images," *Proc. British Machine Vision Conf.,* vol. 2, pp. 509-518, 1994.

[11] A. Zisserman, D. Liebowitz, and M. Armstrong, "Resolving Ambiguities in Auto-Calibration," *Philosophical Trans. Royal Soc. London, Series A,* vol. 356, no. 1740, pp. 1193-1211, 1998.

[12] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision.* Cambridge Univ. Press, 2000.

[13] C. Tomasi and T. Kanade, "Shape and Motion from Image Streams under Orthography: A Factorization Method," *Int'l J. Computer Vision,* vol. 9, no. 2, pp. 137-154, 1992.

[14] P. Sturm and B. Triggs, "A Factorization Based Algorithm for Multi-Image Projective Structure and Motion," *Proc. European Conf. Computer Vision,* pp. 709-720, 1996.

[15] B. Triggs, "Factorization Methods for Projective Structure and Motion," *Proc. Conf. Computer Vision and Pattern Recognition,* pp. 845-851, 1996.

[16] D. Martinec and T. Pajdla, "Structure from Many Perspective Images with Occlusions," *Proc. European Conf. Computer Vision,* pp. 355-369, 2002.

[17] C. Rother, S. Carlsson, and D. Tell, "Projective Factorization of Planes and Cameras in Multiple Views," *Proc. Int'l Conf. Pattern Recognition,* pp. 737-740, 2002.

[18] P. Sturm, "Algorithms for Plane-Based Pose Estimation," *Proc. Conf. Computer Vision and Pattern Recognition,* pp. 1010-1017, 2000.

[19] B. Caprile and V. Torre, "Using Vanishing Points for Camera Calibration," *Int'l J. Computer Vision,* vol. 4, pp. 127-140, 1990.

[20] R. Cipolla and E. Boyer, "3D Model Acquisition from Uncalibrated Images," *Proc. IAPR Workshop Computer Vision,* pp. 559-568, 1998.

[21] C. Chen, C. Yu, and Y. Hung, "New Calibration-Free Approach for Augmented Reality Based on Parameterized Cuboid Structure," *Proc. Int'l Conf. Computer Vision,* pp. 30-37, 1999.

[22] J. Kosecka and W. Zhang, "Video Compass," *Proc. European Conf. Computer Vision,* pp. 476-491, 2002.

[23] D. Liebowitz and A. Zisserman, "Combining Scene and Auto-Calibration Constraints," *Proc. Int'l Conf. Computer Vision,* pp. 293-300, 1999.

[24] D. Bondyfalat, T. Papadopoulo, and B. Mourrain, "Using Scene Constraints during the Calibration Procedure," *Proc. Int'l Conf. Computer Vision,* pp. 124-130, 2001.

[25] P. Sturm and S. Maybank, "On Plane-Based Camera Calibration: A General Algorithm, Singularities, Applications," *Proc. Conf. Computer Vision and Pattern Recognition,* pp. 432-437, 1999.

[26] Z. Zhang, "Flexible Camera Calibration by Viewing a Plane from Unknown Orientations," *Proc. Int'l Conf. Computer Vision,* pp. 666-673, 1999.

[27] B. Triggs, "Autocalibration from Planar Scenes," *Proc. European Conf. Computer Vision,* pp. 89-105, 1998.

[28] E. Malis and R. Cipolla, "Camera Self-Calibration from Unknown Planar Structures Enforcing the Multi-View Constraints between Collineations," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 9, Sept. 2002.

[29] B. Boufama, R. Mohr, and F. Veillon, "Euclidean Constraints for Uncalibrated Reconstruction," *Proc. Int'l Conf. Computer Vision,* pp. 466-470, 1993.

[30] A. Criminisi, I.D. Reid, and A. Zisserman, "Single View Metrology," *Int'l J. Computer Vision,* vol. 40, no. 2, pp. 123-148, 2000.

[31] P. Sturm and S. Maybank, "A Method for Interactive 3D Reconstruction of Piecewise Planar Objects from Single Images," *Proc. British Machine Vision Conf.,* pp. 265-274, 1999.

[32] H.-Y. Shum, M. Han, and R. Szeliski, "Interactive Construction of 3D Models from Panoramic Mosaics," *Proc. Conf. Computer Vision and Pattern Recognition,* pp. 427-433, 1998.

[33] E. Grossmann and J.S. Victor, "Single and Multi-View Reconstruction of Structured Scenes," *Proc. Asian Conf. Computer Vision,* pp. 93-104, 2002.

[34] M. Wilczkowiak, P. Sturm, and E. Boyer, "The Analysis of Ambiguous Solutions in Linear Systems and Its Application to Computer Vision," *Proc. British Machine Vision Conf.,* pp. 53-62, 2003.

[35] C. McGlone, "Bundle Adjustment with Object Space Geometric Constraints for Site Modeling," *Proc. SPIE Conf. Integrating Photogrammetric Techniques with Scene Analysis and Machine Vision II,* pp. 25-36, 1995.

[36] R. Szeliski and P. Torr, "Geometrically Constrained Structure from Motion: Points on Planes," *Proc. SMILE Workshop,* pp. 171-186, 1998.

[37] P. McLauchlan, X. Shen, A. Manessis, P. Palmer, and A. Hilton, "Surface-Based Structure-from-Motion Using Feature Groupings," *Proc. Asian Conf. Computer Vision,* pp. 699-705, 2000.

[38] E. Grossmann and J. Santos-Victor, "Dual Representations for Vision-Based 3D Reconstruction," *Proc. British Machine Vision Conf.,* pp. 516-526, 2000.

[39] D. Bondyfalat and S. Bougnoux, "Imposing Euclidean Constraints during Self-Calibration Processes," *Proc. SMILE Workshop,* pp. 224-235, 1998.

[40] M. Wilczkowiak, G. Trombettoni, C. Jermann, P. Sturm, and E. Boyer, "Scene Modeling Based on Constraint System Decomposition Techniques," *Proc. Int'l Conf. Computer Vision,* pp. 1004-1010, 2003.

[41] P. Debevec, C. Taylor, and J. Malik, "Modeling and Rendering Architecture from Photographs: A Hybrid Geometry and Image-Based Approach," *Proc. SIGGRAPH,* pp. 11-20, 1996.

[42] D. Jelinek and C. Taylor, "Reconstruction of Linearly Parameterized Models from Single Images with a Camera of Unknown Focal Length," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 23, no. 7, pp. 767-774, July 2000.

[43] A. Dick, P. Torr, S. Ruffle, and R. Cipolla, "Combining Single View Recognition and Multiple View Stereo for Architectural Scenes," *Proc. Int'l Conf. Computer Vision,* pp. 268-274, 2001.

[44] M. Wilczkowiak, E. Boyer, and P. Sturm, "Camera Calibration and 3D Reconstruction from Single Images Using Parallelepipeds," *Proc. Int'l Conf. Computer Vision,* pp. 142-148, 2001.

[45] M. Wilczkowiak, E. Boyer, and P. Sturm, "3D Modelling Using Geometric Constraints: A Parallelepiped Based Approach," *Proc. European Conf. Computer Vision,* pp. 221-236, 2002.

[46] M. Pollefeys, L. Van Gool, and M. Proesmans, "Euclidean 3D Reconstruction from Image Sequences with Variable Focal Lengths," *Proc. European Conf. Computer Vision,* pp. 31-42, 1996.

[47] M. Wilczkowiak, P. Sturm, and E. Boyer, "Using Geometric Constraints through Parallelepipeds for Calibration and 3D Modelling," INRIA, Rapport de Recherche 5055, 2003.

[48] T. Buchanan, "The Twisted Cubic and Camera Calibration," *Computer Vision, Graphics, and Image Processing,* vol. 42, no. 1, pp. 130-132, 1988.

[49] F. Kahl, B. Triggs, and K. Åström, "Critical Motions for Auto-Calibration when Some Intrinsic Parameters Can Vary," *J. Math. Imaging and Vision,* vol. 13, no. 2, pp. 131-146, 2000.

[50] D. Liebowitz and A. Zisserman, "Metric Rectification for Perspective Images of Planes," *Proc. Conf. Computer Vision and Pattern Recognition,* pp. 482-488, 1998.

Marta Wilczkowiak received the PhD degree from INPG (National Polytechnical Institute of Grenoble, France) in 2004 after receiving the MSc degree in engineering from Warsaw University of Technology in 2000 (first class grade). She is currently a research associate at the University of Cambridge, United Kingdom, in the Computer Vision and Robotics Group. Her research interests are in image-based 3D modeling, including camera (self-)calibration, 3D reconstruction, and, more recently, texture description for region filling and classification.

Peter Sturm received the PhD degree from INPG (National Polytechnical Institute of Grenoble, France) in 1997, after receiving the MSc degrees from INPG and the Technical University of Karlsruhe, both in 1994. After a two-year postdoctoral at Reading University, he joined INRIA as a senior researcher in 1999. He has acted as a program committee member for ICCV, CVPR, ECCV, ICIP, ICPR, ACCV, and several other conferences and (co)authored more than 70 scientific publications. He received the SPECIF award for 1998 for his PhD thesis (given to one French PhD thesis in computer science per year). His current main research topics are in computer vision, specifically, related to camera (self-)calibration, 3D reconstruction, and motion estimation, both for traditional perspective cameras and omnidirectional sensors.

Edmond Boyer received the MS degree from the Ecole Nationale Supérieure de l'Electronique et de ses Applications, France, in 1993 and the PhD degree in computer science from the Institut National Polytechnique de Lorraine, France, in 1996. He has been with the Institut National de la Recherche en Informatique et Automatique (INRIA) since 1993. He is currently an assistant professor in computer science at the University Joseph Fourier, Grenoble, France, and a researcher at INRIA Rhône-Alpes, Grenoble. His research interests include 3D modeling, silhouettes, motion capture, motion recognition, and multiple camera environments.

Chapter 11

3D Reconstruction of Dynamic Scenes

Paper 30 [25]: P. Sturm. Structure and motion for dynamic scenes – the case of points moving in planes. In A. Heyden, G. Sparr, M. Nielsen, and P. Johansen, editors, *Proceedings of the 7th European Conference on Computer Vision, Copenhagen, Denmark*, volume 2351 of *Lecture Notes in Computer Science*, pages 867–882. Springer-Verlag, May 2002.

Paper 31 [31]: P. Sturm and L. Quan. Camera calibration and relative pose estimation from gravity. In A. Sanfeliu, J.J. Villanueva, M. Vanrell, R. AlquÃ©zar, J.-O. Eklundh, and Y. Aloimonos, editors, *Proceedings of the 15th International Conference on Pattern Recognition, Barcelona, Spain*, volume 1, pages 72–75, September 2000.

Structure and Motion for Dynamic Scenes – The Case of Points Moving in Planes

Peter Sturm

INRIA Rhône-Alpes, 38330 Montbonnot, France,
Peter.Sturm@inrialpes.fr,
WWW home page: http://www.inrialpes.fr/movi/people/Sturm

Abstract. We consider dynamic scenes consisting of moving points whose motion is constrained to happen in one of a pencil of planes. This is for example the case when rigid objects move independently, but on a common ground plane (each point moves in one of a pencil of planes parallel to the ground plane). We consider stereo pairs of the dynamic scene, taken by a moving stereo system, that allow to obtain 3D reconstructions of the scene, for different time instants. We derive matching constraints for pairs of such 3D reconstructions, especially we introduce a simple tensor, that encapsulates parts of the motion of the stereo system and parts of the scene structure. This tensor allows to partially recover the dynamic structure of the scene. Complete recovery of structure and motion can be performed in a number of ways, e.g. using the information of static points or linear trajectories. We also develop a special self-calibration method for the considered scenario.

1 Introduction

Most existing works on structure and motion from images concentrate on the case of rigid scenes. The rigidity constraint allows to derive matching relations among two or more images, represented by e.g. the fundamental matrix or trifocal tensors. These matching tensors encapsulate the geometry/motion of the cameras which took the underlying images, and thus all the geometric information needed to perform 3D reconstruction. Matching tensors for rigid scenes can also be employed for scenes composed of multiple, independently moving objects [1, 2], which requires however that enough features be extracted for each object, making segmentation, at least implicitly, possible.

Shashua and Wolf introduced the so-called homography tensors [9] – matching constraints that exist between three views of a planar scene consisting of independently moving points (each point being static or moving on a straight line). Basically, given correspondences of projections of such points in three images, the plane homographies between all pairs of images can be computed from the homography tensor, which would not be possible with only two images of the scene. It is important to note that this does not make any assumption about the camera's motion, i.e. the camera is indeed allowed to move freely between image takings. So, this work is maybe the first that considers

scenarios where *everything* is moving independently: the camera as well as any point in the scene [1].

Naturally, the question arises if there are other dynamic scenarios that might be interesting to examine. Wolf et al. considered the case of a rigid stereo system taking stereo pairs of a *threedimensional* scene consisting of points moving on straight lines, but independently from each other [11]. From each stereo pair, a 3D reconstruction of the current state of the scene can be obtained (a projective reconstruction if the cameras are not calibrated). Similarly to the above mentioned work on 2D homography tensors, the aim is now to determine 3D homographies between pairs of 3D reconstructions, that would allow to align them. If the stereo system were static, this would again be no problem: the searched for 3D homography is simply the identity transformation. In case of a *moving* stereo system however, Wolf et al. showed that there exist matching tensors, between three 3D reconstructions, representing the state of the scene at three different time instants. From these so-called join tensors, the 3D homographies between all pairs of 3D reconstructions can be recovered, and the reconstructions can be aligned. These 3D homographies represent in fact the stereo system's motions.

Other works along similar lines include that of Han and Kanade [3, 4], who consider points moving with constant velocities (thus on linear trajectories), for the case of affine or perspective cameras. Wolf and Shashua [12] consider several dynamic scenarios, and derive matching constraints by embedding the problem representations in higher-dimensional spaces than e.g. the usual projective 3-space for rigid scenes.

The work presented in this paper is inspired by these works. We consider the following scenario: a moving stereo system taking 2D views of a 3D scene consisting of moving points, each point moving arbitrarily in what we call its *motion plane*. In addition, all motion planes are constrained to belong to the same pencil of planes. The most practical instance of this kind of scenario is the case where all motion planes are parallel to each other and, say, horizontal. This scenario covers for example all scenes where objects move on a common ground plane.

For each time instant considered, the stereo system gives a 3D view of the current state of the scene, which would be a projective reconstruction for example, if the system is uncalibrated. In this paper, we derive matching constraints that exist between such 3D views, and examine which amount of 3D motion and structure information can be recovered from the associated matching tensors. We show that there already exists a matching tensor between two 3D views, for two different time instants. This tensor is more or less the analogue to the fundamental matrix between pairs of 2D views. However, it does not allow full recovery of the stereo system's and the 3D points' structure and motion. Full recovery of these requires additional information, e.g. the knowledge that certain points are static, or that certain points move on linear trajectories (if three or more 3D views are available). In the latter case, the join tensors [11] may be applied, but in our more constrained scenario (pencil of motion planes), a simpler matching constraint exists, that can be estimated with fewer correspondences.

For the special case of parallel motion planes, we present a simple self-calibration method that overcomes singularities that exist without the knowledge of parallelism.

[1] Of course, if the camera were not moving, two images would be enough to do the job (the plane homography between them, for any plane, is intrinsically known – it is the identity).

2 Background and Notation

We will both use standard matrix-vector notations, and tensor notation. In tensor notation, points are specified by superscripts, e.g. P^i. Transformations mapping points onto points, have one superscript and one subscript, e.g. T_i^m. Mapping the point P by T gives a point Q with $Q^m = T_i^m P^i$. Transformations mapping points to hyperplanes, are denoted as e.g. $\mathcal{L}_{ij}$. Let ϵ denote the $3 \times 3 \times 3$ "cross-product tensor", which is defined as $\epsilon_{ijk} a^i b^j c^k = \det A$ where a, b and c are the three columns of matrix A. Among the 27 coefficients of ϵ, 21 are zero (all coefficients with repeated indices), the others are equal to $+1$ or -1.

A *linear line complex* in 3D is a set of lines that are all incident with a line A, the *axis* of the linear line complex [8].

3D lines may be represented via 6-vectors of *Plücker coordinates*. Plücker coordinate vectors are defined up to scale and they must satisfy one bilinear constraint on their coefficients. The Plücker coordinates of a line A can be determined from any two different points on A, as follows. Let B and C be two points on A (represented by 4-vectors of homogeneous coordinates). Define $\left([A]_\times\right)^{ij} = B^i C^j - C^i B^j$. This is a skew-symmetric 4×4 matrix and has thus only six different non zero coefficients – these are the Plücker coordinates of A. There are several possibilities of ordering the coefficients to get a 6-vector, we choose the following:

$$[A]_\times = \begin{pmatrix} 0 & -A_4 & A_6 & -A_2 \\ A_4 & 0 & -A_5 & -A_3 \\ -A_6 & A_5 & 0 & -A_1 \\ A_2 & A_3 & A_1 & 0 \end{pmatrix} .$$

3 Problem Statement

We consider a dynamic scene of the type described above. Any point of the scene may move freely[2] inside what we call its *motion plane*. All motion planes form a pencil of planes, whose axis is a 3D line A (see figure 1). For ease of expression, we also call A the *horizon* or *horizon line* of the motion (although A need not be a line at infinity in general). Let the positions of some point at three different time instants be represented by the 4-vectors of homogeneous coordinates, P, P' and P''. We call *point motion* the "displacement" of an individual point between different time instants.

We consider that the scene is observed my a *moving* stereo system (consisting of two or more cameras). We suppose that at each time instant, a *3D view* of the current state of the scene can be obtained. In the most general case, this will be a projective reconstruction, based on a weak calibration of the stereo system, for the images taken at the considered instant. The stereo system is considered to be moving[3], so different 3D views are represented in different coordinate frames (see figure 1). We call *stereo motion* the transformation between these coordinate frames. Let T' respectively T'' be

[2] This includes that a point may actually be static.

[3] Note that it is nowhere required that the stereo system be moving rigidly or the individual cameras have constant intrinsic parameters or the like.

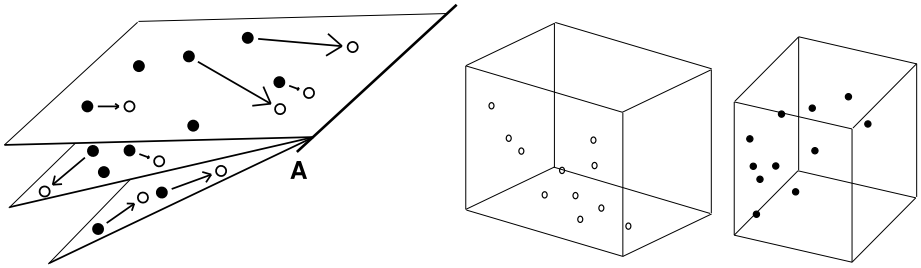


Fig. 1. Left: the considered scenario – points moving in a pencil of motion planes. Right: 3D views at two time instants.

the transformations mapping points from the second respectively third 3D view, into the frame of the first one. Let Q, Q' and Q'' be the coordinates inside the 3D views, of a moving point P at three time instants, i.e. of the points P, P' and P''. The basic question dealt with in this paper is, which amount of stereo and point motion (i.e. scene structure) can be reconstructed, given the input of matching 3D views.

We first study this question for the case of two 3D views, by deriving the associated matching tensor and showing what information on stereo and point motion can be extracted from it. We show that in general, i.e. for unconstrained motion of individual points inside their motion planes, a complete reconstruction is not possible, even if arbitrarily many views (for arbitrarily many time instants) are available. Several ways of obtaining a complete reconstruction, are then described. These are based on additional knowledge about point motion, e.g. knowledge that individual points are actually static or that points are moving on linear trajectories.

4 Two 3D Views – The Projective Case

4.1 The Matching Tensor – A Kind of 3D Epipolar Geometry

The structure of all points, observed at two time instants, may be represented as a linear line complex: the lines spanned by pairs of corresponding points P and P', are all bound to lie in the pencil of motion planes, thus they all intersect the pencil's axis A.

Let us now consider two 3D views of the dynamic scene, taken at two different time instants, by an uncalibrated stereo system. Hence, the 3D views are projective reconstructions of the scene, at the respective time instants. Let point positions in the first 3D view be denoted as Q and in the second one, as Q'. If we knew the stereo motion T' and A, the point motion's horizon line in the first 3D view, then, after mapping all Q' by T', the lines spanned by corresponding points Q and $T'Q'$, would form a linear line complex, with A as axis, as observed above. Let B and C be any two points on A. We must have coplanarity of $Q, T'Q', B$ and C, thus:

$$\det \begin{pmatrix} | & | & | & | \\ B & C & Q & T'Q' \\ | & | & | & | \end{pmatrix} = 0 \ . \tag{1}$$

This equation is bilinear in the coefficients of the reconstructed 3D points Q and Q' and we may rewrite it in the following form:

$$Q^i Q'^j \mathcal{L}_{ij} = 0 \ , \tag{2}$$

where $\mathcal{L}$ is a 4×4 matrix, that depends on the stereo motion and the point motion's horizon line. We might call $\mathcal{L}$ a "Linear Line Complex Tensor", or, L-tensor for short, for the reasons given above. The coefficients of the two points B and C, that appear in $\mathcal{L}$, can all be contracted to the Plücker coordinates of A. It is then easy to derive the following decomposition of $\mathcal{L}$:

$$\mathcal{L} \sim T'^{\mathsf{T}} [A]_\times \ . \tag{3}$$

In the following, we describe several properties of the tensor and in §4.2 we explain, what information can be extracted from it.

The matrix $[A]_\times$ is of rank two at the most, since its coefficients are Plücker coordinates (they satisfy the constraint $A_1 A_4 + A_2 A_5 + A_3 A_6 = 0$). Hence, $\mathcal{L}$ too is of rank two at the most. The right and left null spaces of $\mathcal{L}$ represent nothing else than the horizon line A: the right null space consists of the 3D points that lie on A, in the first 3D view, whereas the left null space contains the 3D points lying on the reconstruction of the horizon line A' in the second 3D view.

In the following, we give some geometric interpretation (cf. figure 2) of the L-tensor, and actually show that there are some analogies to the epipolar geometry between two 2D views of a rigid scene. Let us first consider the action of $\mathcal{L}$ on a point Q in the first 3D reconstruction. The product $([A]_\times)_{ij} Q^i$ gives the motion plane Π_j, that is spanned by the horizon line A and the point Q [8]. The transformation T'^{T} maps planes from the first 3D view, onto planes in the second one: $\Pi'_k \sim \left(T'^{\mathsf{T}}\right)^j_k \Pi_j$. The plane Π' contains the horizon line A'. The correspondence of a point Q' with Q is then expressed as Q' lying on Π', or: $Q'^k \Pi'_k = 0$.

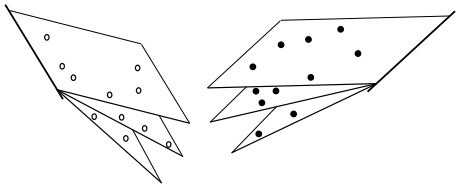


Fig. 2. 3D epipolar geometry.

The analogy to the 2D epipolar geometry is straightforward. The horizon lines A and A' (they represent the same "physical" line, but in 3D views taken at different stereo positions) play the role of the epipoles. In each 3D view, there is a pencil of "epipolar motion planes" containing the horizon line, which is analogous to pencils of epipolar lines in 2D views. Concerning the transformation T', there is an analogous

expression to $\mathcal{L} \sim {T'}^{\mathsf{T}}[A]_\times$ for the 2D epipolar geometry: any plane homography, multiplied by the skew-symmetrix matrix of an epipole, gives the fundamental matrix. Plane homographies are those 2D homographies that map one epipole onto the other and that map corresponding epipolar lines onto each other. Hence, plane homographies are defined up to three parameters. Here, T' is a 3D homography. It is constrained to map epipoles A and A' onto each other and to map corresponding motion planes onto each other. A difference compared to the 2D epipolar geometry is that here, the epipoles (the horizon lines) do represent a part of the dynamic scene structure, and not only the camera geometry. Also, for a given 3D view, the epipole with respect to any other 3D view of the same dynamic scene, is always the same, whereas in the 2D case, the epipoles of one view with respect to several other views, are in general different from each other.

4.2 What Can Be Extracted From The L-Tensor?

It would be desirable to extract, from the tensor $\mathcal{L}$, the stereo system's motion T' and the point motion's horizon line A. Unhappily, this is not entirely possible, which is clear when counting parameters: $\mathcal{L}$ offers at most 11 constraints (it is a rank-2 4×4 matrix, defined up to scale), which is not sufficient to cover the 15+4 parameters for T' and A.

From the decomposition of $\mathcal{L}$ in (3), it is clear that the horizon line A can be extracted via the right nullspace of $\mathcal{L}$ (the horizon line A' in the second 3D view is the left nullspace). The question left is, how much information can be gained on the stereo motion T' ? Let H be any non-singular 3D homography that maps A to the line at infinity consisting of all points $(X, Y, 0, 0)^{\mathsf{T}}$. It can be shown that multiplying equation (3) from the right with the inverse of H leads to:

$$\mathcal{L}H^{-1} \sim {T'}^{\mathsf{T}} \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \end{pmatrix} \sim \begin{pmatrix} 0 & 0 & -T'_{41} & T'_{31} \\ 0 & 0 & -T'_{42} & T'_{32} \\ 0 & 0 & -T'_{43} & T'_{33} \\ 0 & 0 & -T'_{44} & T'_{34} \end{pmatrix} .$$

Hence, $\mathcal{L}$ gives us 7 coefficients of T' (discarding the scale ambiguity).

Let M' be any 4×4 matrix whose third and fourth rows are the same as that of T', but with arbitrary coefficients in the first two rows. Any such M' maps the horizon line A' of the second 3D view onto A in the first 3D view (to be precise, it maps all points on A' onto points on A) and the motion planes of the second 3D view (planes spanned by the Q' and the line A'), onto the corresponding motion planes in the first view. What remains unknown however, is the motion *inside* the individual motion planes.

Mapping the second 3D view by any such M' will in the sequel be called *partial alignment of 3D views*. Methods for obtaining a *full alignment* are described further below. We now describe one method of performing partial alignment. Since everything is defined up to a global projective transformation, we perform the alignment such that the horizon line becomes the line at infinity, consisting of all points $(X, Y, 0, 0)^{\mathsf{T}}$, which leads to simpler expressions in the sequel. Let the Singular Value Decomposition (SVD)

of $\mathcal{L}$ be given as (remember that $\mathcal{L}$ is of rank two):

$$\mathcal{L} = U \begin{pmatrix} a & & & \\ & b & & \\ & & 0 & \\ & & & 0 \end{pmatrix} V^{\mathsf{T}} \ .$$

Define the following projective transformations:

$$M = \begin{pmatrix} & \sqrt{a} & \sqrt{a} \\ & \sqrt{b} & \\ -\sqrt{a} & & \end{pmatrix} V^{\mathsf{T}} \qquad M' = \begin{pmatrix} & & \sqrt{a} \\ \sqrt{a} & & \sqrt{a} \\ & \sqrt{b} & \end{pmatrix} U^{\mathsf{T}} \ .$$

These transformations are by construction non-singular (unless $a = 0$ or $b = 0$). Transforming the first 3D view by M and the second one by M' leads to points MQ and $M'Q'$ that satisfy the following constraint:

$$(M'Q')^{\mathsf{T}} \, \mathcal{L}' \, MQ = 0$$

where

$$\mathcal{L}' = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

is the L-tensor of the partially aligned 3D views.

Before describing methods for full alignment, which will be done in §6, we consider the specialization of our scenario to the Euclidean and affine cases.

5 Two 3D Views – The Euclidean Case

We now consider the case where the 3D views are Euclidean reconstructions, obtained using e.g. a calibrated stereo system. In addition, we concentrate on the case of parallel motion planes, which is probably the most interesting one to study. This means that A is a line at infinity, thus $A_4 = A_5 = A_6 = 0$ and

$$\mathcal{L} \sim {T'}^{\mathsf{T}} [A]_{\times} \sim {T'}^{\mathsf{T}} \begin{pmatrix} 0 & 0 & 0 & -A_2 \\ 0 & 0 & 0 & -A_3 \\ 0 & 0 & 0 & -A_1 \\ A_2 & A_3 & A_1 & 0 \end{pmatrix} \ .$$

The vector $a = (A_1, A_2, A_3)^{\mathsf{T}}$ contains the homogeneous coordinates of the line A, on the plane at infinity. Thus, it also represents the homogeneous coordinates of the normal direction of all motion planes.

For Euclidean 3D views, the stereo motion is a similarity transformation, i.e. a rigid motion possibly followed by a global scale change, which is needed since the two 3D views might have different scales. Thus:

$$T' = \begin{pmatrix} sR & t \\ 0^\mathsf{T} & 1 \end{pmatrix}$$

for a rotation matrix R, a translation vector t and a scalar s. The tensor is thus given by:

$$\mathcal{L} = \begin{pmatrix} 0_{3\times 3} & -sR^\mathsf{T}a \\ a^\mathsf{T} & -t^\mathsf{T}a \end{pmatrix} .$$

It has a particularly simple structure with only 7 non zero coefficients, and no non-linear constraint on them. However, if the global scale s of the second 3D view, is known in advance, e.g. due to constant stereo calibration in which case $s = 1$, then there is one non-linear constraint: the norm of the leading 3-vectors in the 4th column and the 4th row of $\mathcal{L}$ are the same.

What information on stereo and point motion can be extracted from $\mathcal{L}$? The horizon line can be read off directly, as the leading 3-vector of the 4th row. The scale s is obtained as the ratio of the norms of the two leading 3-vectors in the 4th column and 4th row. As for R, it can be seen that it can be determined, up to a rotation about a, the normal direction of the motion planes (see above). Finally, as for the translation t, only its amount along the direction a, can be determined.

Thus, the L-tensor allows, like in the projective case, only partial alignment of 3D views. Here, the ambiguity has three degrees of freedom: let T' be any similarity transformation doing the partial alignment. Adding any transformation consisting of a rotation about a and a translation perpendicular to a, will also result in a valid alignment transformation. Contrary to the projective case, the ambiguous transformation is the same for all motion planes, i.e. if the ambiguity can be cancelled in one motion plane only, then it can be so for the entire 3D scene alignment (in the projective case, full alignment of at least two planes is necessary).

6 Three 3D Views

As discussed previously, two 3D views are not sufficient for full alignment. We now examine if and how additional 3D views, obtained at other time instants, allow to reduce the ambiguity. Let us first note that even with three or more 3D views of our scenario, without additional information, full alignment is not possible. Every 3D view can be partially aligned with the others, as described above, but it is easy to see that the ambiguity in the alignment can not be reduced without further information. In the following, we outline a few types of additional information, that indeed may contribute to full alignment of 3D views.

First, suppose that every point has a linear trajectory. Wolf and Shashua have derived the matching constraints for three 3D views of this scenario [11]. The so-called join tensors, or J-tensors for short, allow to perform full alignment of the three 3D views. This holds even if the linear trajectories are in general position, i.e. if they are not bound to lie on a pencil of planes. The drawback of this general case is that a linear solution of the J-tensors requires at least 60 corresponding point triplets. In §6.1, we specialize the J-tensors to our scenario, and show how this allows full alignment using fewer correspondences.

Second, we remind that until now we did not assume that there are more than one point per motion plane. Thus, it might be interesting to study the case of one or several motion planes containing several points. This can actually be detected after partial alignment, see §7.2. In this case, motion planes with enough moving points on them, can be dealt with individually, e.g. by estimating their homography tensor [9]. Since in our scenario, we already know at least one line correspondence per motion plane (the horizon line), we might consider a simplified version of the H-tensor (see §6.2). Each motion plane for which the H-tensor can be estimated, can thus be fully aligned, and it is easy to show that the alignment of two or more motion planes is sufficient to align the rest of the scene.

Third, knowledge of static points helps of course in the alignment of the 3D views. This will be described briefly in §6.3. Other possibly useful types of additional information could be knowledge of conical trajectories, of motion with constant velocity, of linear trajectories going in the same direction, etc.

6.1 Linear Trajectories

The join tensors, introduced for the general case of unconstrained linear trajectories [11], can also be used here of course. However, in our specialized scenario, we can exploit the additional constraint that the trajectories form a linear line complex (they lie in a pencil of motion planes). It is possible to derive tensors that fully encapsulate this constraint, but they are numerous and not very intuitive. Rather, we suppose in the following that partial alignment of the three 3D views has been performed (e.g. the second and third views have been aligned with the first one), as described in §4.2, and derive matching constraints on the already partially aligned 3D views.

We remind that the horizon line A in the aligned 3D views, is the line at infinity consisting of points $(X, Y, 0, 0)^\mathsf{T}$. Hence, the motion planes are given by 4-vectors of homogeneous coordinates of the form $(0, 0, s, -t)^\mathsf{T}$. We are looking for transformations T' and T'' for the second and third 3D views, that leave the horizon line and all motion planes globally fixed. Hence, the transformations are of the following form:

$$T' = \begin{pmatrix} a' & b' & c' & d' \\ e' & f' & g' & h' \\ 0 & 0 & j' & 0 \\ 0 & 0 & 0 & j' \end{pmatrix} \qquad T'' = \begin{pmatrix} a'' & b'' & c'' & d'' \\ e'' & f'' & g'' & h'' \\ 0 & 0 & j'' & 0 \\ 0 & 0 & 0 & j'' \end{pmatrix} . \tag{4}$$

Let Q, Q' and Q'' represent triplets of corresponding points. The matching constraint used here is that $Q, T'Q'$ and $T''Q''$ have to be collinear, which means that the

rank of the 4×3 matrix composed of these 3 vectors, is two at the most. This constraint can be expressed by a linear family with four degrees of freedom, of $4 \times 4 \times 4$ join tensors [11]. In our case, due to the special form of T' and T'', the four degrees of freedom remain, but some coefficients are known to be zero for all join tensors, i.e. fewer than the 60 correspondences for the general family of join tensors, are needed here to estimate them.

One problem here, that actually turns out as a benefit, is that the point correspondences available to us in the considered scenario, are constrained – all triplets of points Q, Q', Q'' lie in some "horizontal" plane (in the chosen projective frame). Hence, the estimation of the join tensors is underconstrained [4]. Hence, if we were to estimate general $4 \times 4 \times 4$ tensors, there would be a family of solutions of degree higher than four.

We now consider matching constraints for triplets of points lying in a motion plane $(0, 0, s, -t)^\mathsf{T}$, thus $Q \sim (X, Y, t, s)^\mathsf{T}, Q' \sim (X', Y', t, s)^\mathsf{T}$ and $Q'' \sim (X'', Y'', t, s)^\mathsf{T}$. These being collinear after alignment, is expressed by:

$$\text{rank} \left(\begin{array}{c|c|c} X & a'X' + b'Y' + c't + d's & a''X'' + b''Y'' + c''t + d''s \\ Y & e'X' + f'Y' + g't + h's & e''X'' + f''Y'' + g''t + h''s \\ t & j't & j''t \\ s & j's & j''s \end{array} \right) \leq 2 \ .$$

In the general case, four join tensors forming a basis for the 4-degree-of-freedom family, might be extracted by expressing that the four possible 3-minor's determinants vanish. With our input, it is clear that the two minors containing both the third and fourth rows of the above matrix, always vanish. Hence, the corresponding join tensors can not be estimated. As for the other two minors, we may write:

$$\epsilon_{lpq} \left(M_i^l Q^i\right) \left(M_m^p T_j'^m Q'^j\right) \left(M_n^q T_k''^n Q''^k\right) = 0$$
$$\epsilon_{lpq} \left(M_i'^l Q^i\right) \left(M_m'^p T_j'^m Q'^j\right) \left(M_n'^q T_k''^n Q''^k\right) = 0$$

where the 3×4 matrices M respectively M' project points $(X, Y, Z, W)^\mathsf{T}$ to $(X, Y, Z)^\mathsf{T}$ respectively $(X, Y, W)^\mathsf{T}$, i.e.

$$M = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \qquad M' = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \ .$$

We thus obtain the following two join tensors:

$$\mathcal{J}_{ijk} = \epsilon_{lpq} M_i^l \left(M_m^p T_j'^m\right) \left(M_n^q T_k''^n\right) \qquad \mathcal{J}'_{ijk} = \epsilon_{lpq} M_i'^l \left(M_m'^p T_j'^m\right) \left(M_n'^q T_k''^n\right) \ .$$

The slices $\mathcal{J}_{4jk}$ and $\mathcal{J}'_{3jk}$ are zero matrices, and $\mathcal{J}_{3jk}$ and $\mathcal{J}'_{4jk}$ are identical. As for the other two slices, the coefficients with indices lower than 3 inside the slice, are identical in the two tensors. Among the other coefficients, there are several that are the

[4] It is important to note that although T' and T'' conserve motion planes, their join tensors also express the fact that $Q, T'Q'$ and $T''Q''$ may be collinear, for points Q, Q' and Q'' not lying in the same motion plane.

same in both tensors, but that stand at different places. Each one of $\mathcal{J}$ and $\mathcal{J}'$ has only 30 non-zero coefficients. However, again due to the specific type of input correspondences, the tensors can only be estimated up to a 3-degree-of-freedom family of solutions each. Happily, the nature of the ambiguity in the solutions, is known and simple: 24 of the non-zero coefficients for each tensor, can be estimated without ambiguity (up to scale). As for the remaining coefficients, what can be estimated are the following sums: $\mathcal{J}_{134} + \mathcal{J}_{143}, \mathcal{J}_{234} + \mathcal{J}_{243}, \mathcal{J}_{334} + \mathcal{J}_{343}$ and $\mathcal{J}'_{134} + \mathcal{J}'_{143}, \mathcal{J}'_{234} + \mathcal{J}'_{243}, \mathcal{J}'_{434} + \mathcal{J}'_{443}$.

So, 26 point correspondences are in general sufficient to obtain a linear solution for the 24 coefficients and the 3 sums of coefficients (per tensor). The following coefficients of the alignment transformations can be read off directly from the estimated tensor coefficients (after an arbitrary choice for j' and j''): $a', b', e', f', a'', b'', e'', f''$. Having determined them, one can establish, using coefficients of $\mathcal{J}$ and $\mathcal{J}'$, as well as the estimated values of a' etc., a simple linear equation system, to solve for the remaining 8 unknowns, $c', d', g', h', c'', d'', g''$ and h''.

In summary, 26 correspondences are sufficient to determine the alignment transformations T' and T'', and it is nowhere required that there be more than a single moving point per motion plane.

6.2 Using Homography Tensors

We consider the same scenario as in the previous section, i.e. linear trajectories, but now suppose that there are motion planes carrying several points (which can be detected, see §7.2). In this case, we may deal with each motion plane separately.

Consider one motion plane, represented by $(0, 0, s, -t)^\mathsf{T}$. Let Q, Q' and Q'' represent triplets of corresponding points on that plane. Hence, they have the form given in the previous section. The matching constraint for such triplets, corresponds to the homography tensor, or H-tensor for short, introduced in [9]. In that work, the matching constraint was derived for three 2D views of a dynamic planar scene, obtained by a moving 2D camera. Each such 2D view constitutes a projective reconstruction of the planar scene, at the corresponding time instant. Here, we start with three 3D views, which essentially gives us again three projective reconstructions of the motion plane considered.

In order to compute the homography tensor for a motion plane, we first project the three 3D views of the plane by some projection matrix onto some 2D image plane. Let us define the 4×4 projection matrix M:

$$M = \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & t & s \end{pmatrix}$$

whose optical center is guaranteed not to lie on the considered motion plane. We project all three 3D views using M. For the resulting 2D views, there must exist 3×3 transformations H' and H'' such that all triplets $MQ, H'MQ'$ and $H''MQ''$ are collinear. In addition, we know a correspondence of a static line, the motion's horizon line. This line is mapped, by M, to the line at infinity of the 2D views. Hence, H' and H'' must

be affine transformations of the form:

$$H' = \begin{pmatrix} a' & b' & c' \\ e' & f' & g' \\ 0 & 0 & j' \end{pmatrix} \qquad H'' = \begin{pmatrix} a'' & b'' & c'' \\ e'' & f'' & g'' \\ 0 & 0 & j'' \end{pmatrix} \quad .$$

We obtain the following matching equation, in tensorial notation:

$$\epsilon_{lmn} \left(M_i^l Q^i\right) \left(H_p'^m M_j^p Q'^j\right) \left(H_q''^n M_k^q Q''^k\right) = 0$$

We may rewrite the equation:

$$\left(M_i^l Q^i\right) \left(M_j^p Q'^j\right) \left(M_k^q Q''^k\right) \underbrace{\left(\epsilon_{lmn} H_p'^m H_q''^n\right)}_{\mathcal{H}_{lpq}} = 0 \quad .$$

We may identify $\mathcal{H}_{lpq}$ as the $3 \times 3 \times 3$ homography tensor. It can be shown that, due to the constrained form of H' and H'', the tensor has only 19 non-zero coefficients (compared to 27 for the general H-tensor). Hence, a linear solution is possible with 18 or more correspondences. Extracting the individual transformations H' and H'' from $\mathcal{H}$ can be done analogously to what is described in [9].

The tensor $\mathcal{H}$, for one motion plane, allows to partially determine $\mathcal{J}$ and $\mathcal{J}'$ (valid for all motion planes), dealt with in §6.1. Several coefficients of $\mathcal{H}$ occur identically in $\mathcal{J}$ or $\mathcal{J}'$, and the others give linear equations on coefficients of the join tensors.

It is easy to show that the alignment of two motion planes is sufficient to fully align the entire 3D views: for any 3D point in, say, the second 3D view, which we will call Q', let D' be a line passing through it, but that is not contained in Q's motion plane. Let B' and C' be the intersection points of D' with the two motion planes for which full alignment is possible. We thus may compute the positions B and C of the points B' and C', after alignment. The aligned position Q of Q' is finally given by the intersection of Q's motion plane, with the line joining B and C.

6.3 Using Static Points

Given the special form of the alignment matrices (see (4)), it is clear that one static point (that is known to be static), provides two independent equations on each of them. Hence, correspondences associated to four static points in general position, should be sufficient to achieve full alignment of the 3D views (compared to five correspondences that would be required without the specific nature of our scenario). In the Euclidean case, two point correspondences (actually, one and a half) are sufficient for full alignment (compared to three that would be required without the specific nature of our scenario). It would be interesting to study the general case of mixed static and moving points.

7 Other Issues

7.1 Projections $P^5 \rightarrow P^3$

The derivation of the matching tensor for the two-view scenario (see §4.1), could also be performed in the framework of higher dimensional projection matrices used in [12].

Without loss of generality (we deal with projective space), suppose that the horizon A is the line at infinity containing all points $(X, Y, 0, 0)^{\mathsf{T}}$. Let $P \sim (X, Y, Z, W)^{\mathsf{T}}$ be a 3D point at the first time instant and let $P' \sim (X + a, Y + b, Z, W)^{\mathsf{T}}$ be the same point, at the second instant, after having moved in the plane spanned by A and P. We may form the following 6-vector that represents P and its moved version P': $S^{\mathsf{T}} = \begin{pmatrix} X & Y & a & b & Z & W \end{pmatrix}$. We define two projection matrices $P^5 \rightarrow P^3$:

$$M \sim \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \qquad M' \sim \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} .$$

We may observe that $MS \sim P$ and $M'S \sim P'$. In our scenario, we do not observe P and P' directly, but have projective 3D views of them, i.e.: $\lambda Q = TP$ and $\lambda' Q' = T'P'$ for some 4×4 projective transformations T and T' and scale factors λ and λ'. We may derive the matching constraints for Q and Q' in the way shown e.g. in [10]: due to

$$\underbrace{\left(\begin{array}{c|c|c} TM & Q & 0 \\ T'M' & 0 & Q' \end{array}\right)}_{X_{8\times 8}} \left(\begin{array}{c} S \\ \hline -\lambda \\ \hline -\lambda' \end{array}\right) = 0$$

we know that the matrix X is rank-deficient, i.e. that its determinant is equal to zero. By developing the determinant, one obtains the same 4×4 tensor $\mathcal{L}$ as in §4.1 (if we set T to the identity).

7.2 Segmentation of Points Moving in the Same Motion Plane

After partial alignment (see §4.2), the segmentation of points that move in the same plane, is straightforward and can in principle be done in a single 3D view. This might be done by checking, in 3D, if points are on the same motion plane. An alternative would be to compute plane homographies between the 2D views inside a stereo system, for individual motion planes, and check if corresponding projections of 3D points in the 2D views, are consistent with the plane homographies.

7.3 Self-Calibration

We briefly describe a self-calibration algorithm for the scenario of two projective 3D views, under the assumption that the true motion planes are parallel to each other, i.e. the true horizon line is a line at infinity. Using the L-tensor, the horizon line can be determined in the 3D views. Since the true line is a line at infinity, it has two intersection points with the absolute conic – the circular points of all motion planes. We may perform partial self-calibration by searching for the circular points, on the reconstructed horizon lines in our 3D views.

Consider one of the 3D views, after partial alignment as described in §4.2. We suppose that this 3D view has been obtained using two perspective cameras, with unknown

and possibly different focal lengths, but known other intrinsic parameters. The two focal lengths can in general be recovered from the epipolar geometry [5], but this is nearly always singular in practice, due to optical axes passing close to each other [7]. The knowledge of a line at infinity in the projective reconstruction, however, can be used to overcome the singularity, as described in the following.

Let M and M' be the 3×4 projection matrices of the two 2D views. We suppose that the known parts of the calibration matrices (containing aspect ratio and principal point) have been undone, i.e. the unknown calibration matrices of M and M' are $K = \text{diag}(f, f, 1)$ and $K' = \text{diag}(f', f', 1)$. We parameterize the problem in the circular points on the horizon line, which, in the partially aligned 3D view, have coordinates $C_\pm \sim (a \pm I, b, 0, 0)^\mathsf{T}$ for real a, b and $b \neq 0$. Our self-calibration constraints are that the projections of C_+ and C_- lie on the images of the absolute conic in the respective views, which leads to:

$$(am_1 + bm_2 + Im_1)^\mathsf{T} K^{-\mathsf{T}} K^{-1} (am_1 + bm_2 + Im_1) = 0$$
$$(am_1 + bm_2 - Im_1)^\mathsf{T} K^{-\mathsf{T}} K^{-1} (am_1 + bm_2 - Im_1) = 0$$

where m_i is the ith column of M, and similar equations for the second view. Separating the real and imaginary parts of the equations leads to two equations, whose resultant with respect to f^2 is quadratic in a and b. We get a similar equation for the second view. The resultant of these two equations, with respect to a, finally, is the product of the term b^2 and a term that is linear in b^2. Since $b \neq 0$, we thus get a single solution for b^2, which gives us b up to sign (the sign does not matter). From b, unique solutions for a and the squared focal lengths may then be obtained.

We performed simulated experiments with this method. Twenty moving points on each of three planes were simulated. The 3D points were projected in two stereo pairs, and centered Gaussian noise with a standard deviation of 1 pixel was added to the image coordinates. For each stereo pair, the fundamental matrix was computed using the 8-point method [6], projective reconstruction was performed, and the L-tensor between the two resulting point sets estimated. The point sets were then partially aligned. For several stereo configurations (varying vergence angle), 100 simulation runs each were performed. Self-calibration gave focal lengths with an average relative error of about 6% (excepting between 0 and 4 runs were computation failed).

8 Experimental Results

We conducted the following experiment using four stereo pairs of a dynamic scene (see figure 3). About 60 points on the moving objects were manually extracted in all eight images. The experiment was performed for the Euclidean case: the calibration grid visible in the images, was used to obtain full stereo calibration, and thus Euclidean 3D reconstruction of the points. Each such 3D view underwent an arbitrary Euclidean transformation, otherwise they would already have been aligned, since stereo calibration was with respect to the static calibration grid.

From this input, the Euclidean L-tensor between the first and second 3D views was estimated and these 3D views were partially aligned (see §4.2). Then, the other two 3D

views were aligned with the two first one, using a simpler variant of the method of §4.2 (the horizon line in the first two views is already known), not described here.

Full alignment was done for the first three 3D views, based on the knowledge that individual points moved on linear trajectories (see §6.1), and by estimating join tensors specialized to Euclidean alignment transformations. Finally, the fourth 3D view was fully aligned with the others, again using a simplification of the method of §6.1. Some recovered point trajectories are shown in figure 4. Qualitatively, the result seems to be correct, although a quantitative evaluation should definitely be carried out.

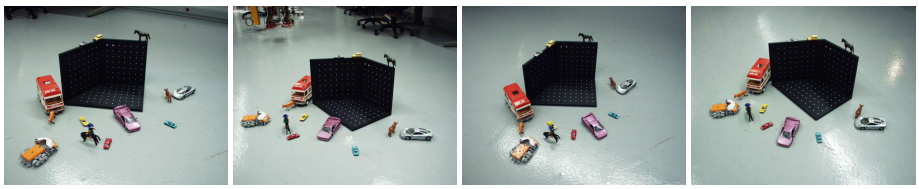


Fig. 3. Two stereo pairs used in the experiments.

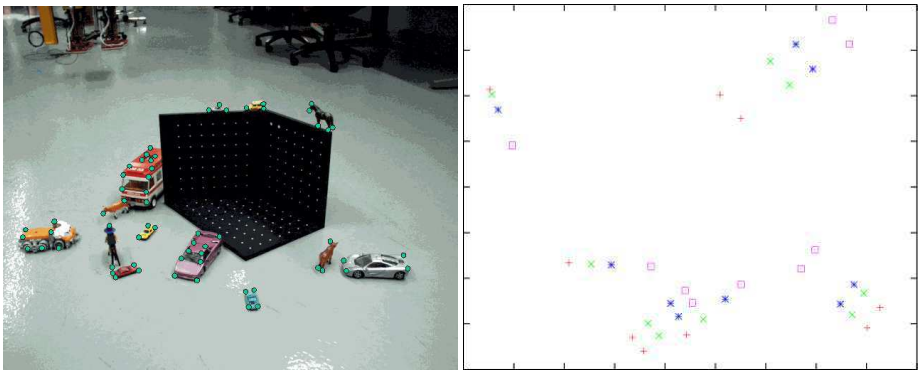


Fig. 4. Left: the moving points used in the experiment. Right: recovered linear trajectories of several points (4 positions each), orthogonally projected onto the ground plane. The point group on the left corresponds to a point on the horse, the group at the bottom to the caravan (3 moving points shown), the group in between, to one of the cars on the grid. The other two point groups belong to the truck and to the sportscar in front of the grid (2 moving points each).

9 Conclusion

We have considered the structure and motion problem for a dynamic scene, consisting of individually moving points, with the restriction that motion happens in a pencil of

motion planes. The scene is supposed to be observed by a moving stereo system, resulting in 3D views of the scene, at different time instants. We have derived the matching constraints between two such 3D views, and shown that full alignment of the views is not possible without further information. Information useful to fully recover the motion of the stereo system as well as the motion and structure of the scene, are for example knowledge of static points or linear trajectories. We have especially discussed how to take into account linear trajectories, to achieve full recovery of structure and motion. A preliminary experiment has shown that it may be feasible to solve the problem in practice, at least in the calibrated case.

Among issues for further work on this topic, minimum numbers of correspondences for the mixed case of known/unknown static and moving points, should be established, and a more thorough experimentation is needed.

Acknowledgement. I wish to thank Adrien Bartoli for preparing the experimental data of section 8.

References

1. Costeira, J., Kanade, T.: A Multi-Body Factorization Method for Motion Analysis. ICCV (1995) 1071–1076
2. Fitzgibbon, A.W., Zisserman, A.: Multibody Structure and Motion: 3-D Reconstruction of Independently Moving Objects. ECCV (2000) 891–906
3. Han, M., Kanade, T.: Reconstruction of a Scene with Multiple Linearly Moving Objects. CVPR **II** (2000) 542–549
4. Han, M., Kanade, T.: Multiple Motion Scene Reconstruction from Uncalibrated Views. ICCV **I** (2001) 163–170
5. Hartley, R.I.: Estimation of Relative Camera Positions for Uncalibrated Cameras. ECCV (1992) 579–587
6. Hartley, R.: In Defence of the 8-Point Algorithm. ICCV (1995) 1064–1070
7. Newsam, G.N., Huynh, D.Q., Brooks, M.J., Pan, H.P.: Recovering Unknown Focal Lengths in Self-Calibration: An Essentially Linear Algorithm and Degenerate Configurations. XVIIIth ISPRS Congress **Part B3** (1996) 575–580
8. Semple, J.G., Kneebone, G.T.: *Algebraic Projective Geometry,* Oxford Science Publ. (1952)
9. Shashua, A., Wolf, L.: Homography Tensors: On Algebraic Entities That Represent Three Views of Static or Moving Planar Points. ECCV (2000) 507–521
10. Triggs, B.: Matching Constraints and the Joint Image. ICCV (1995) 338–343
11. Wolf, L, Shashua, A., Wexler, Y.: Join Tensors: On 3D-to-3D Alignment of Dynamic Sets. ICPR (2000) 388–391
12. Wolf, L., Shashua, A.: On Projection Matrices $P^k \rightarrow P^2, k = 3, \cdots, 6$, and their Applications in Computer Vision. ICCV **I** (2001) 412–419

Camera Calibration and Relative Pose Estimation from Gravity

Peter F. Sturm and Long Quan
INRIA Rhône-Alpes
655 Avenue de l'Europe
38330 Montbonnot St Martin, France
Peter.Sturm@inrialpes.fr, Long.Quan@inrialpes.fr

Abstract

We examine the potential use of gravity for camera calibration and pose estimation purposes. Concretely, objects being launched or dropped follow trajectories dictated by the law of gravity. We examine if video sequences of such trajectories give us exploitable constraints for estimating the imaging geometry. It is shown that it is possible to estimate the infinite homography and the epipolar geometry between pairs of views from this input, from which we can estimate (some) intrinsic parameters and relative pose. There are less singularities compared to approaches that do not use the information that the observed trajectories follow gravity. In this paper, we sketch the geometric principles of our idea and validate them by numerical simulations.

1. Introduction

In this paper we consider the exploitation of gravity for tasks such as camera calibration and relative pose estimation for stereo systems. The basic idea is simple: if we take video sequences of objects being dropped or launched, then the image trajectories, combined with the assumption that the 3D trajectories follow the law of gravity, might provide us with information useful for estimating the imaging geometry. Two types of trajectory are of interest here: the trajectory of an object dropping down in a straight line (e.g. two or more such trajectories give us the vertical vanishing point) and that of an object being launched in a non vertical direction (i.e. with non zero horizontal velocity). In the latter case, the 3D trajectory is a parabola with vertical symmetry axis and we will see later that this information is useful to recover the vanishing geometry of stereo systems.

The paper is organized as follows. In §2, the camera model and some notations are introduced. The problem dealt with in this paper is described concretely in §3 and the geometric ideas leading to its solution are sketched in §4. In §§5 to 8, several ways of extracting information from image sequences of the described type are presented, which are useful for epipolar geometry estimation, calibration, pose estimation and also synchronization of cameras. Algorithms are briefly described in §9 and results of numerical simulations are presented in §10. Some practical issues are discussed in §11, followed by conclusions.

2. Background

Camera model. We use perspective projection to model cameras. A projection may be represented by a 3×4 matrix P mapping points of 3-space to points in 2-space: $\mathbf{q} \sim \mathsf{P}\mathbf{Q}$. Here, $\sim$ means equality up to a non zero scale factor, which accounts for the use of homogeneous coordinates. The projection matrix incorporates the so-called extrinsic and intrinsic camera parameters; it may be decomposed as $\mathsf{P} \sim \mathsf{K}\mathsf{R}(\ \mathtt{I}_3\ \ |-\mathbf{t})$, where $\mathtt{I}_3$ is the 3×3 identity matrix, R a 3×3 orthogonal matrix representing the camera's orientation, $\mathbf{t}$ a 3-vector representing its position, and K the 3×3 *calibration matrix* containing the camera's intrinsic parameters: the (effective) focal length f, the aspect ratio τ, the principal point (u_0, v_0) and the skew factor s accounting for non rectangular pixels.

Infinite homography. Consider the projections of a set of coplanar features in two images. The image features are linked by a 3×3 projective transformation, or homography. If the 3D features considered are located on the plane at infinity, the associated homography between the images is often referred to as the *infinite homography* [1]. This homography depends only on the two cameras' intrinsic parameters and their relative rotation R, as follows:

$$\mathsf{H}_\infty \sim \mathsf{K}_2\mathsf{R}\mathsf{K}_1^{-1} \ . \tag{1}$$

In this paper, we will estimate the infinite homography using corresponding vanishing points and lines and then use it for calibration.

3. Problem Description

We consider one or several static video cameras and our aim is to calibrate these and estimate their relative pose. We examine the potential use of gravity for these tasks. Input are video sequences of objects being dropped or launched. The videos consist thus of "snapshots" of the objects, at different times during their trajectory. From each snapshot, an image point is determined that is assumed to be the projection of the object's center of mass (e.g. for a spherical object, the center of the contour ellipse in the image is a good approximation). Hence, the basic features we will use for calibration etc. are image points.

We assume that the cameras' frame rates are constant and identical. The frame rates need not be known however, and

it is not required that the cameras are synchronized. Based on these simple assumptions, we examine how to use the image points and the knowledge that the corresponding 3D points lie on trajectories dictated by gravity, for calibration and pose estimation. Gravity gives us basically two useful pieces of information: first, all trajectories contain the same point at infinity (the vertical direction) and second, objects travel at constant horizontal velocity while their vertical velocity varies according to the law $v(t) = v(0) - gt$. Once the vertical direction is known, this law allows us to compute ratios of point coordinates to e.g. compute the point of zero velocity of a linear trajectory (see below).

4. Geometric Ideas

Two types of trajectory are considered: linear ones, obtained by letting an object drop, and parabolic trajectories which are obtained when objects are launched. These trajectories are projected to lines and conics. We will use the trajectories as a whole, but also correspondences between "snapshots" of the object taken during the trajectories.

Consider a camera taking an image sequence of dropping objects. The objects drop vertically of course which means that their linear trajectories all have the same point at infinity. Consequently, the image trajectories are a set of concurrent lines, the incidence point being the vanishing point corresponding to the vertical direction, which can thus be easily determined. However, we can not obtain calibration constraints from linear trajectories, even if several cameras observe the trajectories.

Consider now a camera observing objects moving on parabolic trajectories. The 3D trajectories are parabolas with vertical symmetry lines, i.e. all these 3D parabolas contain the vertical point at infinity, like the linear trajectories. Hence, the conics obtained by projecting the trajectories contain the vertical vanishing point. Interestingly, the tangent of any such conic at the vertical vanishing point is nothing else than the vanishing line of the motion plane, i.e. the plane supporting the 3D parabola. Hence, if we consider more than one camera, we can obtain correspondences of one vanishing point and several vanishing lines (one for each parabola). This alone is not sufficient to compute the infinite homography, since all the considered vanishing lines are concurrent. However, together with the epipolar geometry, two correspondences of vanishing lines are already enough to compute the infinite homography, and obtain the associated calibration constraints (see §7).

The epipolar geometry can be estimated using snapshots of the objects, which can be used as point correspondences across the images. Projections of a minimum number of 3D points on at least two planes are sufficient, i.e. two parabolic or three linear trajectories in general position are enough.

5. Using Linear Trajectories

If the camera is calibrated, it is simple to determine the horizon (the vanishing line of the horizontal planes) which might be used for example to orient a camera such that its gaze direction is horizontal. If $\mathbf{q}$ is the vertical vanishing point, then the horizon line $\mathbf{l}$ is given by $\mathbf{l} \sim \mathsf{K}^{-\mathsf{T}}\mathsf{K}^{-1}\mathbf{q}$. The camera could then be oriented by rotating it while tracking the horizon line, until it is horizontal and passes through the camera's principal point.

A second and maybe more interesting potential use of linear trajectories concerns the synchronization of cameras. Suppose we have determined the vertical vanishing points in the images, by intersecting the linear trajectories. It is then possible to determine, for each of the trajectories, the image point which corresponds to the position of zero velocity along the trajectory (three points on each trajectory are sufficient to do so). In addition, we can determine the time stamp of this event, given in frame numbers. Hence, we can establish correspondence of one (virtual) frame between video sequences, which is enough to determine the time lag between them and thus to synchronize.

This helps us to find point matches along trajectories: given a point in one image, we can compute the frame number for another camera, which corresponds to the same time instant. The frame number will in general not be an integer, i.e. the corresponding image does not exist. However, using observed points, we can interpolate the image position that would have been observed at the required time instant. Hence, we are able to establish correspondences between cameras, even if they are not synchronized. These might be used to compute e.g. the epipolar geometry.

6. Using Parabolic Trajectories

As described above, projections of parabolic trajectories are conics containing the vertical vanishing point. Important for us is that the conics' tangents at that vanishing point are the vanishing lines of the supporting planes of the 3D parabolas. Hence, observing parabolic trajectories in several cameras provides us with correspondences of vanishing lines. Given the epipolar geometry (see §5), two line correspondences are sufficient to compute the infinite homography. To see this, we note that the epipolar geometry gives us *point-wise* correspondence along the vanishing lines. If we establish point correspondences on at least two vanishing lines, we already have enough constraints to compute the infinite homography. How to use it for calibration and pose estimation is described in §§7 and 8.

We can thus obtain calibration constraints by observing parabolic trajectories in several views. What about single views: do the projections of parabolic trajectories in a single camera provide us with calibration constraints? The answer is no. The quintessence of the proof for this statement is

that it is possible to obtain the affine structure of the parabolas (we know the vanishing line), but not more (any affine transformation that leaves the vertical direction fixed, maps a set of points on a parabolic trajectory to a set of points on another trajectory that also respects the law of gravity).

7. Calibration and Singularities

The infinite homography gives calibration constraints that have been used for calibrating cameras by rotating them about their optical centers [4] and for stratified self-calibration of cameras [3, 7]. There are a total of 5 constraints: the 8 coefficients of the homography (9 minus 1 scale factor) cover 3 parameters for the rotation (cf. equation (1)) and the remaining 5 constraints can be used to estimate the intrinsic parameters.

The constraints can be used in several ways. It would be possible e.g. to "transfer" calibration information from one camera to another: from equation (1), we obtain $\mathsf{K}_2\mathsf{K}_2^\mathsf{T} \sim \mathsf{H}_\infty\mathsf{K}_1^\mathsf{T}\mathsf{K}_1\mathsf{H}_\infty^\mathsf{T}$ and from $\mathsf{K}_2\mathsf{K}_2^\mathsf{T}$ we may get the calibration matrix K_2 using e.g. Choleski decomposition [8]. The most relevant practical situation however, is probably the case where the focal lengths of the two cameras have to be calibrated, given the other intrinsic parameters. A quasi closed-form solution is easy to derive from equation (1), but omitted here due to lack of space.

As in many (self-) calibration scenarios, there exist generic singularities in the form of relative pose configurations for which there is no unique solution for calibration. For the focal length calibration described above, the only singularity occurs when the optical axes of the two cameras are parallel (proof omitted due to lack of space). In that case, there are infinitely many pairs of values for the two focal lengths that are mathematically valid solutions.

It is known that the focal lengths can be estimated from the epipolar geometry, without knowing the infinite homography [2, 6]. However, this problem is subject to more singularities: whenever the two optical axes are coplanar (i.e. the two cameras are fixated) and in some other cases, the calibration problem has no unique solution [6]. In practice, static stereo systems used for e.g. surveillance, will nearly always be approximately fixated, which will cause numerical instability for the calibration.

Another advantage of being able to use the infinite homography is that more calibration constraints are available: from the epipolar geometry, a maximum of 2 constraints on the intrinsic parameters can be extracted, whereas the infinite homography provides up to 5 constraints.

8. Pose Estimation

Once calibration has been determined, there are several possibilities for estimating the relative pose between cameras, from the infinite homography, calibration and the epipolar geometry. The rotational component can be estimated e.g. using the infinite homography and calibration (cf. equation (1)). Another possibility is to extract pose information from the so-called essential matrix, which represents the calibrated epipolar geometry (see e.g. [5]).

9. Implementation

We briefly describe several parts of the implementation of the ideas described. A first version computes the vertical vanishing point in an image as the intersection of the available linear trajectories. Conics are fitted to obtain the projections of trajectories known to be parabolic. The tangents of these conics are computed and used, together with the epipolar geometry (see §5), to compute the infinite homography and then to calibrate.

An alternative algorithm does the computations in a more direct manner and gives superior results: let $\mathbf{q}_i$ be the image coordinates of points arising from the *projection* of a parabolic trajectory. Let $\mathbf{Q}_i$ be points distributed on an *arbitrary* 3D parabolic trajectory, spaced according to the frame numbers of the $\mathbf{q}_i$. We may compute the homography H that maps the $\mathbf{Q}_i$ to the $\mathbf{q}_i$. Even if the $\mathbf{Q}_i$ do not represent the "true" 3D trajectory, the homography H still enables computation of the true vanishing line, i.e. the line obtained by mapping the line at infinity by H is indeed the projection of the line at infinity of the true 3D plane of motion. If several images are used, correspondences of vanishing lines are thus established, and in addition, the homographies H allow to establish correspondences between vanishing points, which makes it unnecessary to compute the epipolar geometry to determine the infinite homography.

10. Experimental Results

We tested our algorithms by numerical simulation. Here, we present results for the second algorithm of the previous section and the camera configuration shown in figure 1. Two parabolic trajectories (i.e. the minimum data) were created at random and image points were obtained by sampling the trajectory at time instants corresponding to a frame rate of 25 fps. The intrinsic parameters of the cameras were $f = 10mm, \tau = 1, u_0 = v_0 = 256$, for a 512×512 image plane. The image points were perturbed by zero mean Gaussian noise of a standard deviation between 0 and 1 pixels. For different noise levels and elevation angles β (see figure 1), 100 random experiments each were performed.

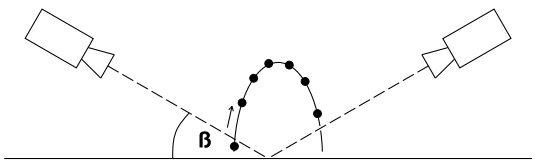


Figure 1. Configuration used for simulations.

Figure 2 shows the median relative errors on the focal lengths. In more than 60-95% of the experiments (depending on noise level), the errors were below 10%. These results are encouraging, especially since only linear algorithms were used (computation of homographies and focal lengths). With more than 2 trajectories and an optimization stage, the results should become significantly better.

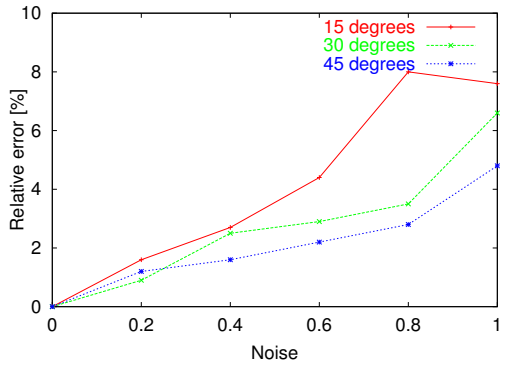


Figure 2. Errors of focal length calibration for different elevation angles and noise levels.

An elevation angle of 0° would mean that the cameras gaze at each other and, in particular, that the optical axes are parallel. This situation is singular (cf. §7), which was reflected by huge errors in the simulations (not shown in the graph). The graphs in figure 2 suggest that the results get better the further away the camera configuration is from the singularity, but there might be other reasons for this, e.g. for higher elevation angles, the vertical vanishing points are closer to the image centers, leading to a better conditioned computation of the infinite homography.

11. Practical Issues

To apply our approach in practice, we have to consider several issues. First of course, the frame rate of the cameras has to be high enough, i.e. the objects should be captured several times during their trajectories. The required frame rate depends mainly on the distance between the camera and the observed scene, and the camera's viewing angle. A simple computation shows that even in close-range conditions, a frame rate of 25 images per second is highly sufficient. Another issue is that we neglect air friction when making the assumption of 3D trajectories obeying perfectly the law of gravity. This should not introduce significant errors, since the most interesting trajectories will be those close to the vertex (point of zero vertical velocity), where the friction is minimal. Other critical issues, concerning the underlying image processing, are shutter speed and contrast. If the shutter speed is too low, motion blur is introduced, making the extraction of objects more difficult. Of course, sufficient contrast between the "calibration object" and the background is needed. This can be achieved by using fluorescent objects. The natural choice for calibration objects are spherical objects: they are perceived identically (if uniformly textured) independently of their orientation and the projections of their centers of gravity are well approximated by the centers of the ellipses perceived in the images.

12. Conclusions

We have presented ways of exploiting image sequences of moving objects if it is known that their trajectories obey the law of gravity. In particular, we have addressed the tasks of camera synchronization, computation of epipolar geometry, calibration and pose estimation. We mainly considered the calibration problem: while there are no useful constraints for single cameras, we obtain more constraints for stereo systems than would be possible without the exploitation of gravity. Maybe most importantly, the additional constraints suffer from fewer calibration singularities. Especially, convergent (or fixated) two-camera systems do not represent a singular configuration, as it is the case for focal length calibration from epipolar geometry alone.

This research is not mature enough yet for practical application. The simulation results however, are encouraging and suggest that our approach might be useful e.g. for calibrating surveillance systems consisting of several cameras at distant locations.

Some parts of the discussion in this paper were given in informal style, due to lack of space. Please contact the first author for proofs and more details.

We thank Bill Triggs for discussions and Frédérick Martin for help with initial experiments using video sequences.

References

[1] O. Faugeras, "Stratification of Three-Dimensional Vision: Projective, Affine and Metric Representations," *Journal of the Optical Society of America A*, Vol. 12, 465-484, 1995.

[2] R.I. Hartley, "Estimation of Relative Camera Positions for Uncalibrated Cameras," *ECCV*, 579-587, 1992.

[3] R.I. Hartley, "Euclidean Reconstruction from Uncalibrated Views," DARPA-ESPRIT *Workshop on Applications of Invariants in Computer Vision*, 187-202, 1993.

[4] R.I. Hartley, "Self-Calibration from Multiple Views with a Rotating Camera," *ECCV*, 471-478, 1994.

[5] H.C. Longuet-Higgins, "A Computer Program for Reconstructing a Scene from Two Projections," *Nature*, Vol. 293, 133-135, 1981.

[6] G.N. Newsam, D.Q. Huynh, M.J. Brooks, H.P. Pan, "Recovering Unknown Focal Lengths in Self-Calibration: An Essentially Linear Algorithm and Degenerate Configurations," ISPRS-*Congress, Vienna*, Vol. XXXI, 575-580, 1996.

[7] M. Pollefeys, L. van Gool, A. Oosterlinck, "The Modulus Constraint: A new Constraint for Self-Calibration," *ICPR*, Vol. I, 349-353, 1996.

[8] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, *Numerical Recipes in C,* Cambridge University Press, 1992.

Chapter 12

Multi-View Dense 3D Reconstruction

Paper 32 [11]: P. Gargallo and P. Sturm. Bayesian 3D modeling from images using multiple depth maps. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, San Diego, USA*, volume 2, pages 885–891, June 2005.

Paper 33 [19]: T. Rodriguez, P. Sturm, P. Gargallo, N. Guilbert, A. Heyden, J.M. Menendez, and J.I. Ronda. Photorealistic 3d reconstruction from handheld cameras. *Machine Vision and Applications*, 16(4):246–257, 2005.

Bayesian 3D Modeling from Images using Multiple Depth Maps

Pau Gargallo and Peter Sturm

INRIA Rhône-Alpes, GRAVIR-CNRS, Montbonnot, France

Abstract

This paper addresses the problem of reconstructing the geometry and color of a Lambertian scene, given some fully calibrated images acquired with wide baselines. In order to completely model the input data, we propose to represent the scene as a set of colored depth maps, one per input image. We formulate the problem as a Bayesian MAP problem which leads to an energy minimization method. Hidden visibility variables are used to deal with occlusion, reflections and outliers. The main contributions of this work are: a prior for the visibility variables that treats the geometric occlusions; and a prior for the multiple depth maps model that smoothes and merges the depth maps while enabling discontinuities. Real world examples showing the efficiency and limitations of the approach are presented.

1. Introduction

This paper addresses the problem of recovering high-resolution 3D models of a scene from a small collection of images. Reconstruction of 3D models from images has been widely studied in computer vision. Many algorithms have been proposed. Differences between them lie in the model used to represent the scene, the prior on this model and the optimization method used for estimating it. The scene representation is a very important factor that practically determines the strengths and weaknesses of the approaches.

Volumetric models, such as voxel-based ones [8, 1, 10] or using level-sets [3], are based on a discretization of 3D space and their goal is to determine the full and the empty cells. These methods can use a large number of images taken from arbitrarily placed viewpoints. Any shape can be represented and the visibility problem is handled in a deterministic geometric manner. However, the initial discretization limits their resolution. The only way of increasing the resolution is to increase the size of the voxel grid. On the other hand, mesh representations [6, 9, 19] can, in theory, adapt their resolution to best reconstruct detailed shapes, but have problems dealing with self-intersections and topological changes during the search.

Depth maps have been mainly studied for two views with a small baseline [12, 7, 15, 18]. The small baseline makes it impossible to get accurate results and these methods are forced to use strong priors that usually introduce fronto-parallel bias. The results of these methods are not precise continuous depth maps but piece-wise planar surfaces. Recently depth map reconstruction from multiple wide-baseline images has been developed with impressive results [14, 13]. The wide-baseline configuration allows astonishingly accurate results without the discretization and topological problems of other methods.

These nice properties of the depth map representation encourage us to use it. However, a single depth map is usually not enough to represent the whole scene: only the parts viewed in a reference view are modeled. A depth map for every input image [17] is needed to ensure that every input pixel is used and modeled. This is probably the model best adapted to the resolution of the input and is the model treated in this work. Alternatively to computing each depth map independently and merging them in a postprocessing step [11, 13], we will compute all the depth maps at the same time which permits an efficient geometric visibility/occlusion reasoning and ensures that the output depth maps will be coherent.

In [13], depth map recovery was formulated as a maximum a posteriori (MAP) problem using the framework proposed in [4] for the novel view synthesis problem showing that the two problems are intrinsically the same. Here we adopt this framework and adapt it to the case of multiple reference views.

The main contributions of this paper to this framework are: First, a reflection on and modification of the likelihood formula. Second, a geometric visibility prior. We use the current depth maps estimation to determine the prior on visibility of the model points. And finally, a multiple depth maps prior that smoothes and merges the depth maps while preserving discontinuities.

1.1. Problem Statement

Our goal is to find a 3D representation of a scene, from a given set of images with full calibration information, i.e. known intrinsic and extrinsic parameters. The model we use to represent the scene consists of a set of colored depth maps. For every pixel in the input images, we want to infer the depth and color of the 3D point that this pixel is seeing.

2. Modeling and Estimation

We treat the problem as a Bayesian MAP search. Input images $\mathcal{I}$ are regarded as a noisy measurement of the model θ. The researched model is defined as the one that maximizes the posterior probability $p(\theta|\mathcal{I}) \propto p(\mathcal{I}|\theta)p(\theta)$.

We first define the relevant variables of the problem in section 2.1. Next, in section 2.2 we decompose the joint probability of the variables, determining the statistical dependencies between them. In sections 2.3 to 2.5 we give a form to each term of the decomposition. Finally in 2.6 we present the optimization method used to estimate the MAP.

2.1. Depth and Color Maps and Visibility Variables

The set of n input images is noted as $\{\mathcal{I}_i\}_{i=1..n}$. $\mathcal{I}_i(\mathbf{x})$ is the color of pixel $\mathbf{x}$ in the i^{th} image and lives in some color space (graylevel, RGB, etc.). The cameras are represented by a set of projection matrices $\{\mathrm{P}_i\}_{i=1..n}$. These matrices have the usual form $\mathrm{P}_i = \mathrm{K}_i(\mathrm{R}_i|\mathrm{t}_i)$ and we scale them so that $(\mathrm{K}_i)_{3,3} = 1$. The depth of a point $\mathbf{X} = (X, Y, Z)^\top$ with respect to a camera position P_i is then defined as $d_i(\mathbf{X}) = (\mathrm{P}_i\bar{\mathbf{X}})_3$, where $\bar{\mathbf{X}} = (X, Y, Z, 1)^\top$. Conversely, if pixel $\mathbf{x} = (x, y)^\top$ of image i has a depth d, then the euclidean coordinates of the corresponding 3D point are $\mathbf{X}_i(\mathbf{x}, d) = d(\mathrm{K}_i\mathrm{R}_i)^{-1}\bar{\mathbf{x}} - \mathrm{R}_i^\top\mathrm{t}_i$, where $\bar{\mathbf{x}} = (x, y, 1)^\top$.

For every pixel in the input images we will compute its depth and color. Depths will be stored in a set of depth maps $\{\mathcal{D}_i\}_{i=1..n}$ and colors in a set of ideal images $\{\mathcal{I}_i^*\}_{i=1..n}$. $\mathcal{D}_i(\mathbf{x})$ and $\mathcal{I}_i^*(\mathbf{x})$ will then be the depth and the color of the point seen by the pixel $\mathbf{x}$ of the i^{th} image. Sometimes it will be more illustrative to think of the set of colored depth maps as a representation of the 3D point cloud $\{\mathbf{X}_i(\mathbf{x}, \mathcal{D}_i(\mathbf{x})) : i = 1..n, \mathbf{x} \in \mathcal{I}_i\}$ and treat all the points of the cloud in the same manner, ignoring their origin, i.e. the image by whose depth map a point is parameterized.

For simplicity, given a point $\mathbf{X} = \mathbf{X}_i(\mathbf{x}, \mathcal{D}_i(\mathbf{x}))$ in the cloud, its estimated color $\mathcal{I}_i^*(\mathbf{x})$ will be noted by $C(\mathbf{X})$. The value of other images on its projection will be noted as $\mathcal{I}_j(\mathbf{X})$ instead of $\mathcal{I}_j(\mathrm{P}_j\bar{\mathbf{X}})$. Similarly, we write $\mathcal{D}_j(\mathbf{X})$ instead of $\mathcal{D}_j(\mathrm{P}_j\bar{\mathbf{X}})$. It is important to note that this is the estimated depth of the *pixel* of image $\mathcal{I}_j$, onto which the 3D point $\mathbf{X}$ is projected, and not the actual depth of the *3D point* $\mathbf{X}$ itself. The latter will be noted as $d_j(\mathbf{X})$, see above and figure 1. As $\mathbf{X}$ is parameterized by the depth map of image $\mathcal{I}_i$, of course, $\mathcal{D}_i(\mathbf{X}) = d_i(\mathbf{X})$.

Due to geometric occlusions, specular reflections or other effects not all the points of the cloud will be visible in every input image. As proposed by Strecha et al. [13] we introduce a boolean variable $\mathcal{V}_{i,\mathbf{X}}$ for each model point $\mathbf{X}$ and each image $\mathcal{I}_i$, that signals whether $\mathbf{X}$ is visible or not in image $\mathcal{I}_i$. These variables are hidden and only their probabilities will be computed.

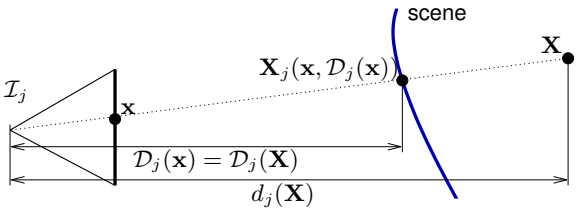


Figure 1. For a given 3D point $\mathbf{X}$, $d_j(\mathbf{X})$ denotes its depth relative to image $\mathcal{I}_j$. $\mathbf{D}_j(\mathbf{X})$ denotes the estimated depth of the pixel onto which $\mathbf{X}$ is projected by P_j, $\mathbf{x} = \mathrm{P}_j\bar{\mathbf{X}}$.

2.2. Decomposition

Having all the variables defined, the next step in a Bayesian modeling task is to choose a decomposition of their joint probability. The decomposition will define the statistical dependencies between the variables that our model is considering. For completeness, we add to the previously defined variables, a variable $\tau = \{\Sigma, \sigma, \sigma', v, l\}$, that represents the set of all the parameters that will be used in our approach, see below. The joint probability of all the variables is then $p(\mathcal{I}, \mathcal{V}, \mathcal{I}^*, \mathcal{D}, \tau)$ and the proposed decomposition (fig. 2):

$$p(\tau)\; p(\mathcal{I}^*|\tau)\; p(\mathcal{D}|\tau)\; p(\mathcal{V}|\mathcal{D}, \tau)\; p(\mathcal{I}|\mathcal{V}, \mathcal{I}^*, \mathcal{D}, \tau) \qquad (1)$$

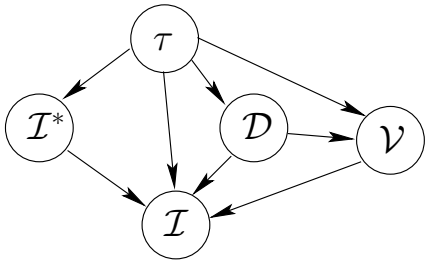


Figure 2. Network representation of the joint probability decomposition. Arrows represent statistical dependencies between variables.

1. $p(\tau)$ is the prior probability of the parameters. We assume a uniform one in this work and ignore this term.
2. $p(\mathcal{I}^*|\tau)$ is the prior on the colors of the depth maps. This term was used by Fitzgibbon et al. [4] to regularize the novel view synthesis problem with great success. The so-called image-based priors were introduced to enforce the computed images $\mathcal{I}^*$ to look like natural images, which in practice was enforced by looking like images of a catalogue of examples [5]. In this work, we adopt a uniform prior, centering the regularization on the depth maps like [13].
3. $p(\mathcal{D}|\tau)$ is the prior on depth maps. Its work is to smooth and integrate the different depth maps. It is developed in section 2.5. Note that in contrast with [13], no statistical dependence between $\mathcal{I}^*$ and $\mathcal{D}$ is used here. Modeling this dependence can help when dealing with constant

albedo surfaces were image and depth discontinuities are correlated. On the other hand, this can produce undersmoothing of textured smooth surfaces.

4. $p(\mathcal{V}|\mathcal{D},\tau)$ is the visibility prior. We propose to consider visibility as dependent on $\mathcal{D}$, to enable geometric reasoning on occlusions (section 2.4). In the E-step of the EM algorithm described below, this geometric visibility prior will be probabilistically mixed with photometric evidence, giving an estimate of the visibility that is more robust to geometric occlusions than using a uniform prior [13].
5. $p(\mathcal{I}|\mathcal{V},\mathcal{I}^*,\mathcal{D},\tau)$ is the likelihood of the input images. Particular attention is paid to this term (section 2.3), because we find that usual formulae are not satisfactory for the wide-baseline case.

The variables can be classified in three groups: the known variables (or data) $\mathcal{I}$ and τ, the wanted variables (or model) $\theta = (\mathcal{I}^*, \mathcal{D})$ and the hidden ones $\mathcal{V}$. The inference problem is now stated as finding the most probable value of the wanted variables, given the value of the known ones and marginalizing out the hidden ones. That is, we want to estimate

$$\theta^* = \arg\max_\theta p(\theta|\mathcal{I},\tau) = \arg\max_\theta \int p(\mathcal{I},\mathcal{V},\mathcal{I}^*,\mathcal{D},\tau)d\mathcal{V}$$

The following sections give a form to each term of the decomposition.

2.3. Likelihood

Pixels in input images are treated as noisy observations of the model. We suppose the noise to be independently identically distributed. The likelihood can be decomposed as the product of the per-pixel likelihoods:

$$p(\mathcal{I}|\mathcal{V},\theta) = \prod_i \prod_{\mathbf{x}} p(\mathcal{I}_i(\mathbf{x})|\mathcal{V},\theta) \qquad (2)$$

Note that this product is extended over the pixels in the input images and not over the points in the 3D model, as opposed to many of the previous works on Bayesian modeling of the stereo problem that define the likelihood as

$$p(\mathcal{I}|\mathcal{V},\theta) = \prod_{\mathbf{X}} \prod_i p(\mathcal{I}_i(\mathbf{X})|C(\mathbf{X}),\mathcal{V}) \qquad (3)$$

Although this has the great advantage of clearly representing the contribution of every model point to the total likelihood, it is, strictly speaking, incorrect.

The problems related to this approximation are sketched in figure 3. In the first case, many 3D points instantiated by the first image's depth map project to the same pixel in the second image. Computing the product over the 3D points as in (3) will overuse the second image's pixel. This is not a good idea given that the viewing angle of this pixel is really steep, hence its color is quite random and depends on the camera sensors. In the second case, only a few points of the first image's depth map project to the second image, so many pixels of the second image will be unused even if these pixels were seeing the scene better than any other.

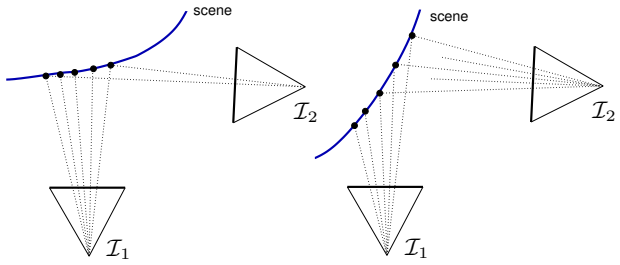


Figure 3. On the left, many 3D points instantiated by the first image project to the same pixel in the second one. On the right, many pixels on the second image have no 3D point instantiated by the first image that is projected onto them.

In small-baseline situations, where there is almost a bijection between pixels in each image and 3D model points from any other image's depth map, these effects are minimal and can be ignored. However, in our wide-baseline applications it is desirable to deal with them. In the following, we propose an approximation to the per-pixel product likelihood (2).

The per-pixel likelihood $p(\mathcal{I}_i(\mathbf{x})|\mathcal{V},\theta)$ measures the similarity between the color $\mathcal{I}_i(\mathbf{x})$ observed in the pixel $\mathbf{x}$ of image i, and the color that the model would predict for that pixel, let us call it $C_i^*(\mathbf{x})$. Remember that all 3D points are used to explain all images, hence $C_i^*(\mathbf{x})$ is computed from the colors of all 3D points that are projected onto that pixel, and may be different from $\mathcal{I}_i^*(\mathbf{x})$. Let us call $S_{i,\mathbf{x}}$ the set of points that are projected to $\mathbf{x}$ in image i. The color $C_i^*(\mathbf{x})$ is hard to define, because $S_{i,\mathbf{x}}$ may contain many points; its definition corresponds to a rendering problem. It seems natural to define $C_i^*(\mathbf{x})$ as the mean color of all visible ($\mathcal{V}_{i,\mathbf{X}} = 1$) points in $S_{i,\mathbf{x}}$: since they are currently considered to be visible by the pixel, they should contribute to predicting its color. Sadly, the resulting expression of the likelihood is difficult to deal with and in particular, the EM formulas become intractable.

To approximate this solution with a more usable expression, we define the per-pixel likelihood as the geometric mean of the likelihoods that the pixel would have if only one of the points in $S_{i,\mathbf{x}}$ was used,

$$p(\mathcal{I}_i(\mathbf{x})|\theta) = \prod_{\mathbf{X}\in S_{i,\mathbf{x}}} p(\mathcal{I}_i(\mathbf{x})|C(\mathbf{X}),\Sigma)^{\frac{1}{|S_{i,\mathbf{x}}|}}.$$

Computing the geometric mean of probabilities is equivalent to computing the arithmetic mean of energies. The idea behind is to cut the pixel's information in $|S_{i,\mathbf{x}}|$ parts and give one to each point in $S_{i,\mathbf{x}}$. This is justified as a manner

of using all the points in $S_{i,\mathbf{x}}$ without overusing the pixel $\mathbf{x}$. It is a heuristic approximation of the correct solution (2) but it solves the problems commented above and permits writing the likelihood as a per-point product

$$p(\mathcal{I}|\theta) = \prod_{\mathbf{X}} \prod_{i} p(\mathcal{I}_i(\mathbf{X})|C(\mathbf{X}), \Sigma)^{\frac{1}{|S_{i,\mathbf{x}}|}} \tag{4}$$

We refer to the term $p(\mathcal{I}_i(\mathbf{X})|C(\mathbf{X}), \Sigma)$ as the *pixel-point likelihood* and we model it by a mixture between a normal distribution in the case that $\mathcal{V}_{i,\mathbf{X}} = 1$ and a uniform distribution over the color space in case that $\mathcal{V}_{i,\mathbf{X}} = 0$. Since we work with probabilities for the visibility variables, this is:

$$\begin{aligned} p(\mathcal{I}_i(\mathbf{X})|C(\mathbf{X}), \Sigma) &= p(\mathcal{V}_{i,\mathbf{X}} = 1|\mathcal{D})\mathcal{N}(\mathcal{I}_i(\mathbf{X})|C(\mathbf{X}), \Sigma) \\ &+ p(\mathcal{V}_{i,\mathbf{X}} = 0|\mathcal{D})\mathcal{U}(\mathcal{I}_i(\mathbf{X})) \end{aligned} \tag{5}$$

When the prior on the visibility variables is constant, this distribution is called a *contaminated Gaussian* [16]. The following section describes the non-constant form that we give to this visibility prior.

2.4. Geometric Visibility Prior

The mixture of the pixel-point likelihood (5) is balanced by the visibility prior $p(\mathcal{V}_{i,\mathbf{X}}|\mathcal{D})$. This models the prior belief on whether the point $\mathbf{X}$ is visible or not in image $\mathcal{I}_i$, before taking into consideration the colors $C(\mathbf{X})$ or $\mathcal{I}_i(\mathbf{x})$. A uniform distribution is usually used for such a situation [13, 17]. However, our decomposition (1) of the joint probability, allows using the depth maps' information to give a more interesting form to this prior.

$\mathcal{D}_i(\mathbf{X})$ is the estimated depth of the pixel in image $\mathcal{I}_i$ onto which $\mathbf{X}$ is projected, which is not the same (see section 2.1) as the actual depth $d_i(\mathbf{X})$ of $\mathbf{X}$. If $d_i(\mathbf{X})$ is similar to $\mathcal{D}_i(\mathbf{X})$, it suggests that $\mathbf{X}$ is near the point seen by $\mathbf{x}$, so it should be more likely that $\mathbf{X}$ is visible. Symmetrically, if $d_i(\mathbf{X})$ is very different from $\mathcal{D}_i(\mathbf{X})$ the idea of image $\mathcal{I}_i$ seeing $\mathbf{X}$ seems unlikely. Thanks to this simple observation the geometric visibility can be easily and efficiently handled, in a multiple depth map approach. In [17] a threshold was used to strictly determine the visibility. Here we quantify the above idea by the (smooth) expression

$$p(\mathcal{V}_{i,\mathbf{X}} = 1|\mathcal{D}) = v \exp\left(-\frac{(d_i(\mathbf{X}) - \mathcal{D}_i(\mathbf{X}))^2}{2\sigma^2}\right)$$

where $v \in [0, 1]$ is the visibility prior for points at the estimated depth $\mathcal{D}_i(\mathbf{X})$ and σ models the tolerance that we give to points that are not exactly at this depth.

The effect of this prior on the pixel-point likelihood is in agreement with the above intuition. For points near the depth $\mathcal{D}_i(\mathbf{X})$, the prior is large and the normal distribution centered at $C(\mathbf{X})$ of the pixel-point likelihood mixture (5) is weighted up. This makes pixel colors similar to $C(\mathbf{X})$ more probable. For points far from the depth $\mathcal{D}_i(\mathbf{X})$, the uniform distribution is favored. The color $C(\mathbf{X})$ becomes irrelevant, which is logical given that we don't believe that the pixel $\mathrm{P}_i\bar{\mathbf{X}}$ is seeing the point $\mathbf{X}$.

2.5. Multiple Depth Map Prior

The multiple depth map prior $p(\mathcal{D}|\tau)$ is supposed to evaluate the plausibility of a set of depth maps without using any other information but the depth maps themselves. Two main properties are desired:

1. Each depth map should be mostly smooth but (strong) discontinuities have to be allowed.
2. The 3D points clouds belonging to the different depth maps should be *overlapping*.

Instead of using separate terms to measure smoothness and overlap, we evaluate the two properties in a single expression. To do so, we think of the set of depth maps as a point cloud forgetting, for a moment, the 2D neighborhood relation existing in the images. Smoothness and overlap will be reached by letting points attract one another, independently if they originate from the same depth map or not.

We express the probability of the point cloud as a Markov network:

$$p(\mathcal{D}) \propto \prod_{\mathbf{X} \in \mathcal{D}} \prod_{\mathbf{Y} \in N(\mathbf{X})} \varphi(\mathbf{X}, \mathbf{Y}) \tag{6}$$

where $N(\mathbf{X})$ denotes the neighborhood of $\mathbf{X}$ and $\varphi(\mathbf{X}, \mathbf{Y})$ is the compatibility probability for the $(\mathbf{X}, \mathbf{Y})$ pair. For the moment, this neighbourhood extends to the totality of points, $N(\mathbf{X}) = \mathcal{D} \setminus \{\mathbf{X}\}$.

Like for the pixel-point likelihood (5), we model the compatibility probabilities as mixtures of a normal and a uniform distribution, balanced by a hidden line process $\mathcal{L}$:

$$\begin{aligned} \varphi(\mathbf{X}, \mathbf{Y}) &\propto p(\mathcal{L}_{\mathbf{X},\mathbf{Y}} = 1)\, \mathcal{N}(\mathbf{Y}|\mathbf{X}, \sigma') \\ &+ p(\mathcal{L}_{\mathbf{X},\mathbf{Y}} = 0)\, \mathcal{U}(\mathbf{Y}) \end{aligned}$$

where $p(\mathcal{L}_{\mathbf{X},\mathbf{Y}})$ is the constant prior on the line process. $l = p(\mathcal{L}_{\mathbf{X},\mathbf{Y}} = 1)$ is a parameter of the method. σ' is the variance of the isotropic three dimensional normal distribution $\mathcal{N}$. $\mathcal{U}$ is a uniform distribution over a volume containing the scene ($\mathcal{U}(\mathbf{Y}) = \mathcal{U}(\mathbf{X})$).

The underlying idea is that the process $\mathcal{L}_{\mathbf{X},\mathbf{Y}}$ signals if the two points should attract each other or not. If $\mathcal{L}_{\mathbf{X},\mathbf{Y}} = 1$, we regard $\mathbf{Y}$ as a noisy measurement of $\mathbf{X}$ and its probability distribution is set to a normal distribution centered on $\mathbf{X}$ and with variance σ'. Note that this relationship is symmetrical. Otherwise, if $\mathcal{L}_{\mathbf{X},\mathbf{Y}} = 0$ a uniform distribution is used to reflect the idea that $\mathbf{X}$ and $\mathbf{Y}$ are not related.

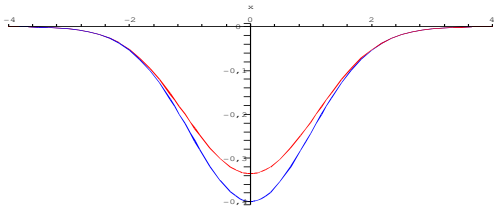


Figure 4. In red, a plot of the clique potentials of our prior, $-\log(\mathcal{N}(x|0,1)+1)$. In blue, the kernel correlation based one, $-\mathcal{N}(x|0,1)$.

This prior is computationally expensive. If m is the number of points, there are $O(m^2)$ compatibility probabilities. However, for all the points far enough from $\mathbf{X}$, $\mathcal{N}(\mathbf{Y}|\mathbf{X},\sigma')$ will be very small and $\varphi(\mathbf{X},\mathbf{Y})$ will be constant. We can thus restrict the neighborhood to the points near enough to $\mathbf{X}$. We define the neighborhood as the points inside a sphere centered at $\mathbf{X}$ with a radius ρ dependent on σ'. Finding this neighborhood is in itself a hard problem that can be expensive. Luckily, our point cloud comes from a set of depth maps where points are ordered. The projection of the neighborhood sphere in each image is an ellipse. The set of 3D points instantiated by the pixels inside these ellipses contain all neighbors of $\mathbf{X}$, greatly facilitating the task of finding them.

As desired, the proposed prior smoothes and integrates all the depth maps at the same time. Discontinuities are allowed thanks to the hidden line process $\mathcal{L}$ that avoids distant points to attract one another.

Kernel Correlation. Our prior is closely related to *leave-one-out* kernel correlation. Tsin and Kanade showed the capacities of the KC prior in smoothing while keeping discontinuities and applied it successfully to the stereo problem [18]. The KC prior can be written as a Markov network with

$$\varphi_{KC}(\mathbf{X},\mathbf{Y}) \propto exp(\mathcal{N}(\mathbf{X}|\mathbf{Y},\sigma'))$$

In figure 4, the negative logarithms of our compatibility probability and the KC-based one are plotted to show the similar shape they have. The advantage of the mixture prior over the KC is that it is defined in a probabilistic framework that permits the incorporation of new cues of information. We could, for example, use a statistical relation between the color of points and the line process $\mathcal{L}$, that makes points of the same color have a better chance to be attracted to one another.

2.6. Optimization

We maximize the posterior probability with the Expectation Maximization algorithm [2]. Direct non-linear optimization of our posterior is not only possible but also less expensive than EM. However, EM is known to often be more stable and easier to monitor as hidden variables are explicitly estimated. EM alternates between estimating the hidden variables' probabilities and optimizing the model. We start with a given initial model θ^0 (see section 3) and repeat the next steps until convergence.

E-step. In the expectation step we compute the posterior probabilities of our hidden variables $\mathcal{V}$ given the current estimate of the model. We store them as a set of visibility maps $f_{i,\mathbf{X}} = p(\mathcal{V}_{i,\mathbf{X}} = 1|\mathcal{I},\theta^t)$ and, by Bayes' rule,

$$f_{i,\mathbf{X}} = \frac{p(\mathcal{V}_{i,\mathbf{X}}=1|\mathcal{D})\mathcal{N}}{p(\mathcal{V}_{i,\mathbf{X}}=1|\mathcal{D})\mathcal{N} + p(\mathcal{V}_{i,\mathbf{X}}=0|\mathcal{D})\mathcal{U}}$$

where $\mathcal{N} = \mathcal{N}(\mathcal{I}_i(\mathbf{X})|C(\mathbf{X}),\Sigma)$ and $\mathcal{U} = \mathcal{U}(\mathcal{I}_i(\mathbf{X}))$ (see (5)). It is at this moment that the geometric visibility prior is mixed with the photometric evidence to give an estimation of the current visibility.

M-step. In the maximization step the expected visibility maps are used to maximize the expected log-posterior,

$$\theta^{t+1} = \arg\max_{\theta} \langle \log p(\mathcal{I}|\mathcal{V},\theta)\rangle_f + \log p(\mathcal{D})$$

i.e. the sum of the expected log-likelihood (cf. (4) and (5)),

$$\sum_{\mathbf{X}}\sum_{i} \frac{1}{S_{i,\mathbf{x}}} \left(f_{i,\mathbf{X}} \log\mathcal{N} + (1-f_{i,\mathbf{X}})\log\mathcal{U}\right)$$

and the log-prior (cf. (6)), $\sum_{\mathbf{X}}\sum_{\mathbf{Y}} \log\varphi(\mathbf{X},\mathbf{Y})$.

The maximum is searched by gradient descent. Analytical derivation of the log-posterior with respect to the model variables can be easily computed from the above equations. In our implementation, only one gradient descent iteration is done at each M-step. The iteration finds a better guess for θ^{t+1} but not the best. This method is called the Generalized EM algorithm. The motivation for doing this is that each iteration of the gradient descent method is as expensive as an E-step. Rapid alternation between E and M steps permits a faster actualization of the visibility maps.

3. Experiments

We have implemented the algorithm in a pyramidal scheme to speed up convergence and reduce the probability of being trapped in irrelevant local minima. We start using reduced versions of the original input images, and thus reduced versions of the colored depth maps. When convergence of EM is achieved, a higher resolution level is initialized with the obtained results, using bilinear interpolation.

In all our experiments, the noise variance Σ (see section 2.3) was included to the wanted variables and estimated during the optimization process. The visibility prior, v, was set to 0.9 expressing the idea that a point is likely to be visible in an image if it is at a similar depth to that estimated for that image (see section 2.4). σ' was set to the same value as

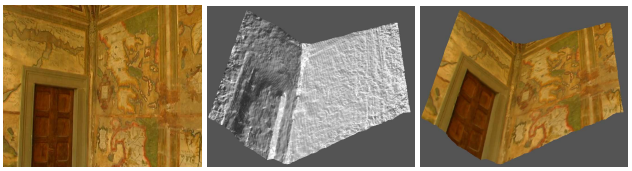


Figure 5. **Loggia**: One of the three input images (left) and renderings of its recovered depth maps.

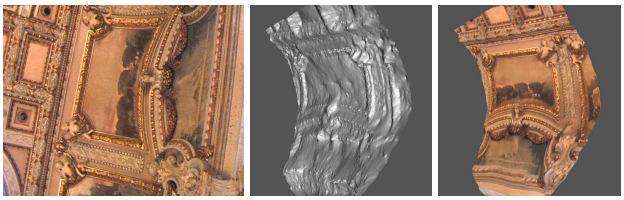


Figure 6. **Casino**: One of the five input images on the left and untextured and textured renderings of the recovered surface viewed from a very different angle.

σ (see sections 2.4 and 2.5). This value was heuristically set to two times the robust mean of the distance between pairs of 3D points instantiated from consecutive pixels in the images. The parameter l (see section 2.5) was the only one to be specially adapted for each experiment. We present the results on several datasets of increasing complexity.

Easy. The Loggia data set (figure 5) consists of three wide-baseline images of a scene with rich textures and simple geometry. Initial depth maps were set to a constant value (i.e. fronto-parallel) and the algorithm converged to the correct surface. The Casino data set (figure 6) contains five images with small baseline. Constant depth initialization was also used. The results show the potential of the method in capturing fine details. In both cases, large enough values of l ($l > 0.1$) gave similar results.

Medium. We tested our method's performance for the Cityhall scene [1] to prove that the algorithm can achieve state-of-the-art results in wide-baseline matching but with several depth maps at once. Images 3, 4 and 5 of the dataset were used. In this case, the model was initialized using the 3D feature point positions from the calibration step. Pixels with known depth were fixed while successive Gaussian blurs were applied to the rest of the depth map pixels. From this coarse initialization the algorithm converged, merging the depth maps into a single surface. The results (figure 7) show fine and rich details and the strong discontinuity between the foreground statues and the door was preserved.

Hard. To show the potential of the algorithm in dealing with strong discontinuities and geometric occlusions, we tested its performance on the challenging statue data set (figure 8). The scene contains a statue in front of a far wall. A single depth map is not enough to model the scene because none of the images sees the whole statue or wall. We

[1] The Cityhall images with full calibration can be downloaded from `http://www.esat.kuleuven.ac.be/~cstrecha/testimages/`

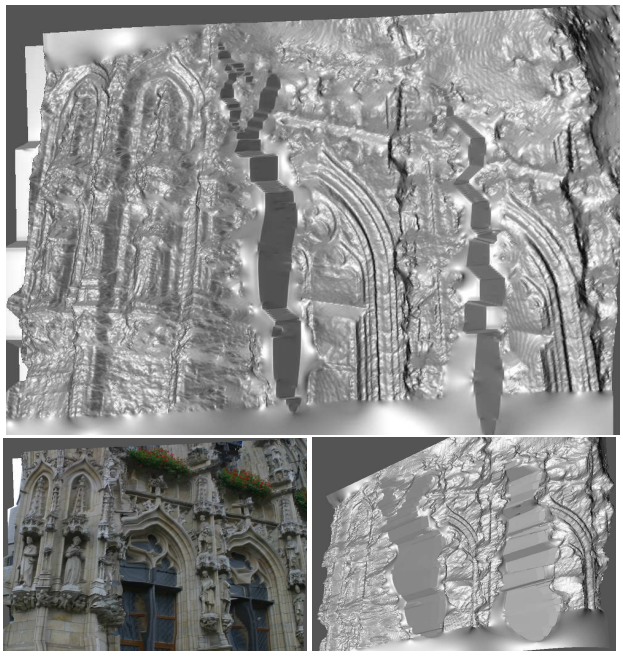


Figure 7. **Cityhall**: Untextured, textured and relighted renderings of an estimated depth map viewed from two different angles. No points were removed. The oversmoothed part at the bottom of the model corresponds to points seen only in one image. The two flat regions in the center correspond to discontinuities of the depth map.

used the same coarse initialization method as for the Cityhall scene.

The main difficulty was to strictly estimate the large discontinuity between the statue and the wall. Smoothing in this region would produce incorrect 3D points between the foreground and the background. We set the l parameter to a small value ($l = 0.2$) to motivate the points not to attract each other too much (see section 2.5). The discontinuity was then well preserved, but not at the exact position. Some background points remained attached to the statue. In addition, when initializing a finer level of the pyramid from a coarser one, we used bilinear interpolation which smoothed out the discontinuity.

To solve these problems we alternated several EM iterations with the following heuristic global search. For each pixel $\mathbf{x}$ and image i, we consider all the depths of the 3D points $S_{i,\mathbf{x}}$ that are projected to that pixel (see section 2.3). Then we test if the likelihood will be improved if we change the depth of pixel $\mathbf{x}$ to any of these values. The value producing the best improvement is kept. The large discontinuity between the statue and the wall was detected by the EM algorithm from the coarser level. The global search heuristic placed this discontinuity at the correct position and mantained it there in the finer levels.

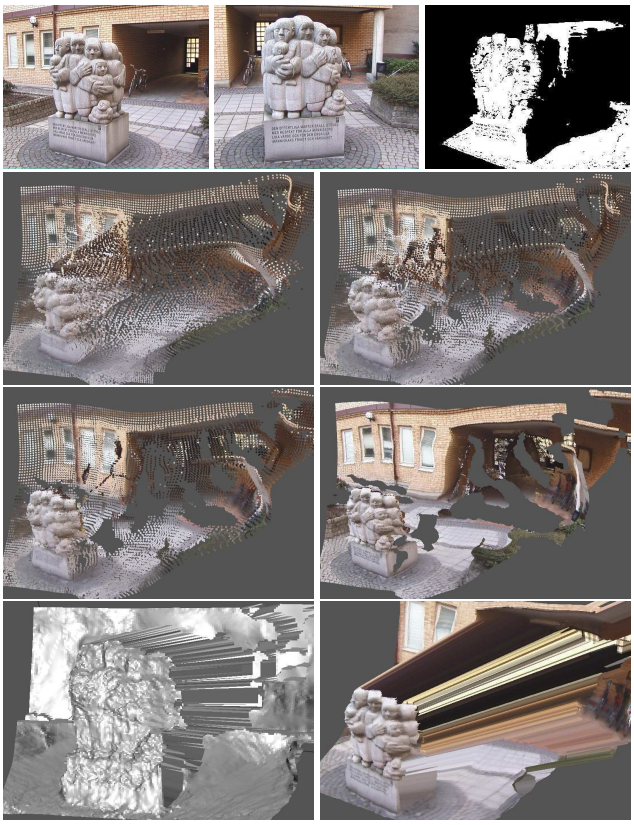


Figure 8. **Statue**: On top, first and last of the five input images and the visibility map of the first image with respect to last, i.e. the estimated probabilities of the 3D points instantiated by the first depth map to be visible in the last image. The next two rows show a point rendering of the set of all the depth maps at the same time during the evolution of the algorithm, from a very coarse initialization, to the final model. On the last row, two renderings of the estimated depth map $\mathcal{D}_2$ are shown. Note the well-preserved large discontinuities between statue and background.

4. Discussion

The proposed method was formulated in a rigorous probabilistic framework extending previous works. The experiments proved the pertinence of this extensions. However, there are still some issues to solve in order to make the method more usable.

The probabilistic approach permits the parameters of the method to be learned during the optimization. In effect, treating the parameters as random variables we can either estimate their most probable value or marginalize them out. Our current implementation needs to manually set three parameters. Although these parameters represent well defined concepts it will be preferable that the algorithm automatically sets them.

The other issue of the method, like in any other gradient descent based method, is the initialization. The pyramidal implementation of the EM algorithm converges well in cases where the strong discontinuities are captured from earlier small resolution levels. However, without a good initialization, it seems likely that for images such as the ones used in [12], the EM algorithm does not reach the global optimum but a local one. Interestingly, one of the best performing methods in this field [15], uses the same Bayesian scheme, but the optimization is done with the Loopy Belief Propagation algorithm. It is our interest to study the possibility of applying this or other global maximization techniques to our posterior probability definition.

Acknowledgements. This work used resources developed by partners of the European project VISIRE (IST-1999-10756). We especially would like to thank Martin Johansson and Anders Heyden for providing us with the data for the statue sequence.

References

[1] A. Broadhurst, T. W. Drummond, R. Cipolla. A probabilistic framework for space carving. *ICCV*, 2001.

[2] A. P. Dempster, N. M. Laird, D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *J. R. Statist. Soc. B*, 39:1–38, 1977.

[3] O. Faugeras, R. Keriven. Complete dense stereovision using level set methods. *ECCV*, 1998.

[4] A. Fitzgibbon, Y. Wexler, A. Zisserman. Image-based rendering using image-based priors. *ICCV*, 2003.

[5] W. T. Freeman, E. C. Pasztor. Learning low-level vision. *IJCV*, 40:25 – 47, 2000.

[6] P. Fua, Y. Leclerc. Object-centered surface reconstruction: combining multi-image stereo shading. *IUW*, 1993.

[7] V. Kolmogorov, R. Zabih, S. J. Gortler. Generalized multi-camera scene reconstruction using graph cuts. *EMMCVPR*, 2003.

[8] K. Kutulakos, S. Seitz. A theory of shape by space carving. *IJCV*, 38(3):199–218, 2000.

[9] D. Morris, T. Kanade. Image-consistent surface triangulation. *CVPR*, 2000.

[10] S. Paris, F. Sillion, L. Quan. A surface reconstruction method using global graph cut optimization. *ACCV*, 2004.

[11] M. Pollefeys. *Self-Calibration and Metric 3D Reconstruction from Uncalibrated Image Sequences*. PhD thesis, 1999.

[12] D. Scharstein, R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV*, 47:7–42, 2002.

[13] C. Strecha, R. Fransens, L. Van Gool. Wide-baseline stereo from multiple views: a probabilistic account. *CVPR*, 2004.

[14] C. Strecha, T. Tuytelaars, L. Van Gool. Dense matching of multiple wide-baseline views. *ICCV*, 2003.

[15] J. Sun, H.Y. Shum, N.N. Zheng. Stereo matching using belief propagation. *PAMI*, 25(7), July 2003.

[16] R. Szeliski. Bayesian modeling of uncertainty in low-level vision. *IJCV*, 5(3):271–301, 1990.

[17] R. Szeliski. A multi-view approach to motion and stereo. *CVPR*, 1999.

[18] Y. Tsin, T. Kanade. A correlation-based model prior for stereo. *CVPR*, 2004.

[19] G. Vogiatzis, P.H.S. Torr, R. Cipolla. Bayesian stochastic mesh optimization for 3d reconstruction. *BMVC*, 2003.

Machine Vision and Applications (2005) 16(4): 246–257
DOI 10.1007/s00138-005-0179-4

ORIGINAL PAPER

Tomás Rodríguez · Peter Sturm · Pau Gargallo · Nicolas Guilbert · Anders Heyden · Fernando Jauregizar · J. M. Menéndez · J. I. Ronda

Photorealistic 3D reconstruction from handheld cameras

Received: 24 March 2004 / Accepted: 29 March 2005 / Published online: 10 June 2005

Abstract One of the major challenges in the fields of computer vision and computer graphics is the construction and representation of life-like virtual 3D scenarios within a computer. The VISIRE project attempts to reconstruct photorealistic 3D models of large scenarios using as input multiple freehand video sequences, while rendering the technology accessible to the non-expert.

VISIRE is application oriented and hence must deal with multiple issues of practical relevance that were commonly overlooked in past experiences. The paper presents both an innovative approach for the integration of previously unrelated experiences, as well as a number of novel contributions, such as: an innovative algorithm to enforce closedness of the trajectories, a new approach to 3D mesh generation from sparse data, novel techniques dealing with partial occlusions and a method for using photo-consistency and visibility constrains to refine the 3D mesh.

Keywords Photo-realistic 3D reconstruction · Self calibration · Structure from motion · Image based rendering (IBR) · Video analysis

1 Introduction

Traditionally, reconstructing large scenarios in 3D has been costly, time consuming, and required expert personnel. Usually the results showed artificial look and produced unmanageable heavy models. However, recent advances in the areas of video analysis, camera calibration, and texture fusion allow us to think in a more satisfying scenario, where the user just needs to wander around, aiming his camera, making shoots, following the provided guidelines, and the system will automatically do the 3D reconstruction of the desired scenario for him. Our objective is to come closer to this ideal scenario, but it is our belief that current state of the art does not allow still for reliable full automatic 3D reconstruction. For that reason we avoid dogmatic views and accept human cooperation in the 3D reconstruction process whenever it can lead to better results, faster processing, or a personal touch.

In this document, the results of the EC funded project VISIRE (IST-1999-10756) are presented. VISIRE [1] attempts to reconstruct in 3D photorealistic interiors of large scenarios from multiple freehand video sequences, while rendering the technology accessible to the non-expert. VISIRE offers an advanced authoring tool that empowers the user to interact effortlessly with the underlying Computer Vision (CV) software with the aim to process the acquired video material off-line and obtain lightweight 3D models highly resembling the original. VISIRE observed certain basic assumptions that greatly influenced the design of the system: no expert CV personnel should be needed, no knowledge about the camera was assumed (i.e. unknown intrinsic and extrinsic parameters), no proper calibration should be required, and the system should work with the only aid of a domestic camcorder (i.e. professional cameras, tripods, lighting or measurement devices were discarded).

VISIRE deals with several CV disciplines: auto calibration, structure from motion, non-linear robust and iterative methods, texture and geometry representation, etc. There is a general belief in the scientific community that these issues have been mostly solved. This statement is correct with respect to the basic principles, but there is still a big gap that must be filled between the scientific demonstration and a technology that “works." The fact is the problem of automatic 3D reconstruction of complex scenarios remains largely unsolved and the technology never found its way to the market in spite of its unquestionable interest.

T. Rodríguez (✉)
Eptron SA. R&D Dpt. Madrid, Spain
E-mail: tomasrod@eptron.es

P. Sturm · P. Gargallo
INRIA Rhône-Alpes Montbonnot, France

N. Guilbert · A. Heyden
Centre for Mathematical Science, Lund University, Sweden

F. Jauregizar · J. M. Menéndez · J. I. Ronda
E.T.S.I. Telecomunicaciones, Universidad Politécnica de Madrid, Spain

VISIRE is application oriented and hence must approach multiple issues of practical relevance. As opposed to previous experiences, aimed at technological demonstrations based on ad-hoc solutions that must be modified by the experts for every new situation or partial 3D reconstructions of elements of the scenario specially selected for the task, VISIRE offers the innovation to consider a "global" approach and proposes instead methods and tools able to solve a number of general situations. Obviously this approach is more challenging and the price to pay is the need to define certain restrictions of use and consider new difficulties previously ignored in this type of application, that now acquires the category of critical problems: reliable tracking in sparse environments, introducing constraints such as planarity and closedness, occlusion reasoning, optimized autocalibration, combining and adapting textures from multiple viewpoints, multiresolution, integrating manual and automatic mesh generation methods, etc.

In that sense, existing autocalibration [2] methods are mainly based on a small set of images, ranging from 2 to 20 in today's very complex systems. VISIRE aims to break this barrier in several orders of magnitude, mainly because it can use the thousands of images typically found in video streams. No system so far has ever tried to accomplish such a complex scenario, in any of the above-mentioned tasks. Another important goal is to produce photorealistic 3D models, i.e. models that may be realistically rendered from synthetic viewpoints. One way of doing so [2–4] is to apply IBR (Image-Based Rendering). The other main method [5] is to enhance a geometrical 3D model with texture maps or other information e.g. surface reflectance properties. Up to now, we concentrate on the second solution, which produces a more compact scene description. The challenge for the 3D mesh generation process is that most existing methods are designed for sets of dense and regularly distributed 3D points. This is usually not the case in automatic structure from motion, so we have developed methods that use information provided by the input images, via visibility and photoconsistency constraints.

VISIRE follows a standard process division (see Fig. 1): we start out in Sect. 2 with Feature Analysis. We continue with a description of Calibration methods in Sect. 3. Next, the approach to 3D Registration and Mesh Generation is introduced in Sect. 4. The authoring tool is illustrated in Sect. 5. Results and experimental evaluation are presented in Sect. 6. Finally, we end in Sect. 7 with the conclusions.

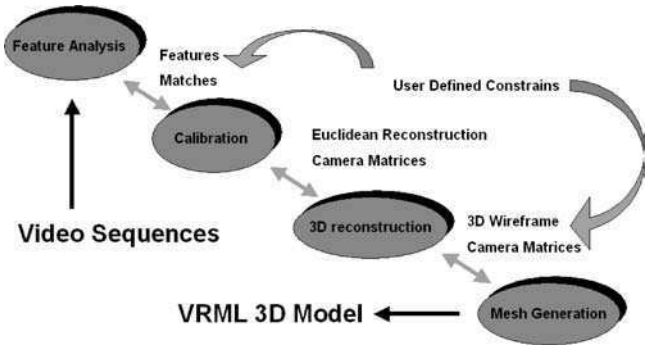


Fig. 1 VISIRE computer vision chain

2 Feature analysis

The method selected for feature extraction in VISIRE relies on the Tomasi/Kanade [6] approach, which first smoothes the image by convolving it with a Gaussian and then the gradients (in x and y) are computed by convolving the resulting image with the derivative of a Gaussian. Later, a measure of cornerness is applied for each pixel, evaluating the minimum eigenvalue of the 2×2 gradient matrix computed in a 7×7 window around the pixel. This measure is used to sort in descending order all pixels in the image, ensuring that selected features are at least 10 pixels away from each other.

Feature tracking is applied through a robust method, based on the Kanade–Lucas–Tomasi approach [6] that relies on the assumption of an affine motion field in the projection from the 3D point motion to the 2D surface of the image, and on the computation of a weighted *dissimilarity* function between consecutive images that is minimized using a Newton–Raphson iterative method, over a limited spatial window. After the trajectories are generated along time, they are validated by checking their compliance to the assumed model of rigid motion of the scene. To this purpose the set of trajectories is processed in two steps corresponding, respectively, to a local processing and a global processing. For the local processing the sequence is divided into non-overlapping temporal windows and for each of them a RANSAC-based calibration is performed, which is later optimized by bundle adjustment. The temporal windows must be short enough to ensure that enough trajectories remain complete within it but long enough to include enough motion. Local data from different temporal windows are consolidated and reoptimized in the global processing step, which operates iteratively consolidating data from pairs of adjacent windows. After this analysis, a trajectory or part of a trajectory results validated when it is successfully approximated by the reprojection of a scene feature.

3 On-line calibration

Once a sufficient amount of reliable image correspondences have been established the overall structure of the scene may be recovered. This recovery is performed in two major steps, namely the recoveries of projective and Euclidian structure, respectively. Projective structure is recovered by extracting camera matrices from the trifocal tensor, see e.g. [7] and followed by a series of the resectionings in order to obtain each of the subsequent cameras. However, the result is only defined up to a projective transformation, and is consequently useless for visualization purposes. However, as shown in [8], very general constraints such as assuming square pixels suffice to establish the in- and extrinsic camera

calibration parameters. In VISIRE, the actual implementation makes use of the Cheirality inequalities, see e.g. [7] followed by an identification of the plane at infinity and eventually the recovery of the intrinsic and extrinsic camera parameters and Euclidian structure.

Nevertheless, the estimation of the initial set of cameras depend on the solution of a linearized problem, and are consequently subject to errors. Hence, in order to achieve a maximum likelihood solution, so called bundle adjustment is applied, see eg. [9] for details. This involves minimizing the reprojection error

$$\sum_{i,j} \|(x_{ij} - p(P_i, X_j))\|^2$$

where x_{ij} indicates the jth image point in the ith image, and $p{:}(P, X) \mapsto \mathbb{R}^2$ projects the 3D homogeneous point X using the camera matrix P. In VISIRE, bundle adjustment is implemented using the Levenberg–Marquardt method and a sparse system solver, allowing for significantly more effective processing and for longer sequences than previously, i.e. sequences of up to 300 views and 15,000 3D points.

In building complete systems for solving structure and motion, new questions and research subjects arise naturally. One unique feature of the VISIRE system is the ability to apply the constraint of *closedness* to a sequence. In a long sequence, the same image feature is likely to appear on several occasions, but will, however, under normal conditions be reconstructed as a different 3D feature each time. For the general case, enforcing identity on these features turns out to be indispensable. Also, small errors and degrees of freedom from partial reconstructions might accumulate to a very large error so that the scene structure or camera motion obtained from the feature the first time it is encountered might not fit at all when it is re-projected to the images where the feature appears later. One way to deal with this problem is to make partial reconstructions from subsequences and then stitch these together by minimizing the distances between corresponding points in 3D via homographies of the substructures as it is done in [10]. Another way would be to impose soft constraints as described in [11], where a penalty term on the difference between expected identical parameters is included as a Lagrange multiplier in the error function.

Both methods have important drawbacks we intended to overcome. Our initial approach was to distribute the accumulated error equally on all the parameters, although in the norm given by their covariance. Specifically, the reprojection error vector and its associated covariance structure are projected onto their respective lower-dimensional manifolds corresponding to the reduced system, i.e. the system where identity of the parameters has been enforced. Using the resulting values, the optimal reduced parameters are calculated through the equivalent of a Levenberg–Marquardt iteration.

3.1 Batch reconstruction from sparse data

This approach evolved to a method that takes closedness constraints into account in the very first reconstruction step (*Batch* process). This is in itself interesting for robustness reasons, since as many constraints as possible should be enforced as early as possible to avoid ending up in a erratic situation. Another robustifying feature of the algorithm is that the auto-calibration step is performed from affine to Euclidian, which is significantly simpler than the original projective to Euclidian. However, the algorithm has turned out to play a more important role: basically, the sequential approach originally used in the VISIRE project (and most other state-of-the-art structure from motion systems) is best suited for applications where decisions need to be made as soon as a new frame becomes available (i.e. robotics). There has previously been no alternative, since existing batch algorithms [12–14], in practice, have required *all* features to be visible in *all* images; something that is unlikely in a real scenario.

The new method developed [15] proposes a batch algorithm that would work on sparse data. The basic idea is to compute matching tensors between the images (fundamental matrices, trifocal or quadrifocal tensors) and finally determine all of the camera matrices in a single computational step. The notion of $\mathbf{F} - \mathbf{e}$ closure constraint, i.e. $\mathbf{F}_{12}\mathbf{P}_2 + [\mathbf{e}_{21}]_\times \mathbf{P}_1 = \mathbf{0}$ was introduced in [16], denoting bilinear constraints between camera parameters and matching tensors. We derived an alternative closure constraint, the $\mathbf{F}-$closure:

$$\mathbf{X}^{\mathrm{T}} \underbrace{\mathbf{P}_1^{\mathrm{T}} \mathbf{F}_{12} \mathbf{P}_2}_{\varphi} \mathbf{X} = \mathbf{0}, \ \forall \mathbf{X} \in \mathbb{P}^3. \tag{1}$$

where $\mathbf{P}_1$ and $\mathbf{P}_2$ denote the camera matrices, $\mathbf{F}_{12}$ is the fundamental matrix, and $\mathbf{X}$ a set of 3D homogeneous points. In the affine case, the structure of φ becomes particularly simple:

$$\varphi = \mathbf{P}_2^{\mathrm{T}} \mathbf{F}_{12} \mathbf{P}_1 = \begin{bmatrix} \mathbf{0}_{3\times 3} & \mathbf{a} \\ -\mathbf{a}^{\mathrm{T}} & \mathbf{0} \end{bmatrix} \tag{2}$$

By re-arranging (2) and by denoting the elements of $\mathbf{F}_{12}$ by

$$\mathbf{F}_{12} = \begin{bmatrix} 0 & 0 & a \\ 0 & 0 & b \\ c & d & e \end{bmatrix}, \tag{3}$$

we obtain four linear constraints on the coefficients of $\mathbf{P}_1$ and $\mathbf{P}_2$:

$$\begin{bmatrix} a & b & c & d \end{bmatrix} \begin{bmatrix} \mathbf{P}_1 \\ \mathbf{P}_2 \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{0}_3 & -e \end{bmatrix}}_{\mathbf{r_{12}}} \tag{4}$$

These constraints apply for each pair of views $\mathbf{P}_{i_1}$ and $\mathbf{P}_{i_2}$, $i_1 \neq i_2$, provided $\mathbf{F}_{i_1 i_2}$ is defined. We construct a

linear system of equations using 4 with the form $\mathbf{S}P = R$:

$$\begin{bmatrix} \mathbf{s}_{12} \\ \mathbf{s}_{1i_1} \\ \vdots \\ \mathbf{s}_{i_k i_m} \end{bmatrix} \begin{bmatrix} \mathbf{P}_1 \\ \mathbf{P}_2 \\ \vdots \\ \vdots \\ \vdots \\ \mathbf{P}_m \end{bmatrix} = \begin{bmatrix} \mathbf{r}_{12} \\ \mathbf{r}_{1i_1} \\ \vdots \\ \mathbf{r}_{i_k i_m} \end{bmatrix} \tag{5}$$

where $\mathbf{r}_{i_1 i_2}$ is the right hand side of Eq. 4 and $\mathbf{s}_{i_1 i_2}$ are $1 \times 2m$ row vectors.

$$\mathbf{s}_{i_1 i_2} = \begin{bmatrix} \cdots & \underbrace{a \;\; b}_{\text{FirstBlock}} & \cdots & \underbrace{c \;\; d}_{\text{SecondBlock}} & \cdots \end{bmatrix} \tag{6}$$

One important advantage of the proposed algorithm is the ability to include different types of constraints, such as equality of given cameras. This feature is illustrated in Fig. 2 where a cubic point cloud is reconstructed given views taken on a circular trajectory.

The point cloud consists of 300 3D points evenly distributed in the cube $[0.5] \times [0.5] \times [0.5]$ and of 30 cameras with focal length $f = 100$ equidistantly placed on a circular path centered at $(0, 0, 0)$. Each frame contains features which are visible in the nine following frames. Gaussian noise with $\sigma = 1$ is present in the images. Figure 2b shows the initial reconstruction of the camera trajectory using affine approximation and in Fig. 2c an alternative reconstruction where equality has been assumed between the first and the last camera in the sequence. The perspective equivalents of the affine cameras were obtained by choosing a focal length $< \infty$ ensuring that all the 3D points would lie in front of the cameras where they had been observed.

Clearly, the initial reconstructions capture the overall structure of the scene and the motion, thus allowing for the subsequent bundle adjustment to converge to the global minimum. One point of special interest is the fact that within this framework, the affine camera model approximates the perspective camera sufficiently well, even though the depth of the object is approximately the same as the distance to the object, i.e. a lot more than the 10% that are usually considered the upper limit.

4 3D registration and mesh generation

Creation of photorealistic models is done in two major steps: based on 3D points reconstructed during on-line calibration or subsequently, we first generate a triangular mesh describing the scene's surfaces; then, texture maps for the surface patches are extracted using all available images.

After on-line calibration, we are provided with a set of 3D points, projection matrices of a set of images, and 3D-to-2D point correspondences. Usually, only interest points that could be tracked reliably, were used for on-line calibration and metric 3D reconstruction. However, once projection matrices are known, additional point tracks can be checked more easily for outliers if the correspondingly reconstructed 3D points are reliable. We may thus enrich the set of 3D points before proceeding to mesh generation.

4.1 Geometric consistency constraints

Most existing methods for mesh generation from 3D points rely mainly on 3D geometric reasoning, e.g. proximity constraints between points, see e.g. [17, 18] and references therein. These methods give unusable results for our input data, because they are designed for rather dense and regular point sets. In order to work with more difficult data, other information besides pure 3D geometry should be used. Since the 3D points are obtained by reconstruction from images, we have such additional information. First, visibility constraints can be imposed to prune incorrect surface patches, e.g. a hypothetical patch that would lie between a 3D point and the optical center of a view where that point is visible, can be rejected (see left part of Fig. 3). Other, more complicated, visibility constraints are also used: especially, a surface patch that partially occludes another one, without occluding an actual 3D point, is rejected (see right part of Fig. 3). The drawback of verifying this situation is quite a large computation time (naively, each hypothetical triangle has to be tested against each triangle already in the mesh). Nevertheless, and although this situation is rare, it occurred in all our experiments; not taking it into account would result in visually unpleasing models. Such visibility constraints were also used in [19], where a surface mesh is built

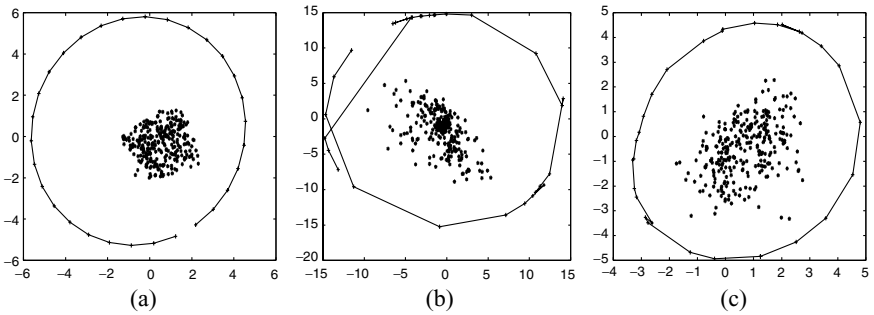


Fig. 2 Reconstruction, object centered configuration **a** original configuration, **b** initial reconstruction using affine approximation, **c** same as, **b** but assuming equality between the first and last camera

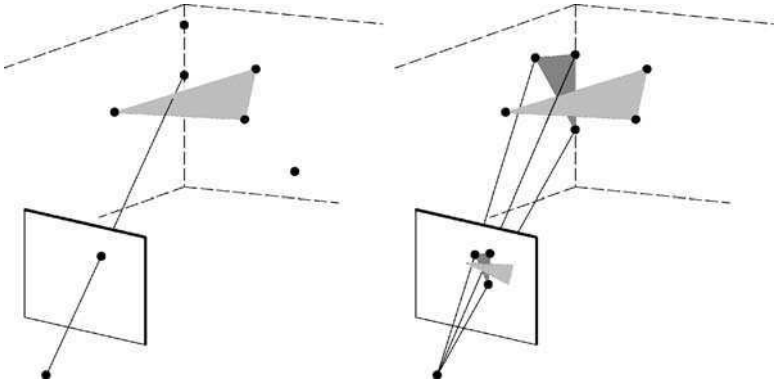


Fig. 3 Visibility constraints. *Left*: the triangle will not be accepted since it would occlude a 3D point from a view where that point is visible (a corresponding 2D interest point was extracted). *Right*: the light triangle would not be accepted since it would partially occlude a triangle already existing in the mesh

incrementally, starting with a mesh obtained by a Delaunay triangulation in one view, and then rejecting and adding triangles based on visibility constraints of one additional view after the other. We proceed differently, by iteratively adding new triangles to automatically selected seed triangles, and thus by letting a mesh grow, directly ensuring all available visibility constraints (and other constraints, see below). This way, we may end up with a better connected surface mesh, which may be easier to edit/complete if necessary.

4.2 Photometric consistency constraints

Another constraint we use to test hypothetical surface patches, is photoconsistency: a planar patch is acceptable, if its projections into all images where the patch's vertices are visible, correspond to image regions with the same "texture." This is verified by the following process: image regions corresponding to a planar 3D surface patch, are warped (using homographies associated to the 3D plane) to some common frame (to undo perspective effects). The simplest method to measure photoconsistency would then be a "multi-image cross-correlation" using all warped image regions (e.g. compute variance of greylevels) [20]. This can be problematic, for example in cases where some images of a surface patch are partially occluded or show specular highlights. Also, individual images are taken from different viewpoints which usually result in changes of perceived intensity values. To this end, we measure photoconsistency using the following general and robust approach: we estimate an "average" texture map for the considered patch as well as parameters for intensity transformations for the input images. This is a non-linear optimization problem, whose cost function, in its most general form, is as follows (for a patch with m pixels, seen in n images):

$$\sum_{p=1}^{m}\sum_{i=1}^{n}\sum_{k=R,G,B}\rho(I_{ikp}-\alpha_{ik}T_{kp}-\beta_{ik}) \tag{7}$$

Here, T_{kp} is the intensity value of the p-th pixel of the generated mean texture map, for color channel k. I_{ikp} is the corresponding intensity value, measured in image i. The α_{ik} and β_{ik} are parameters for affine intensity transformations, for image i and color channel k (we have implemented several modes for intensity transformations: one affine transformation per color channel, but also restricted modes with the same affine transformation for all channels or only intensity scaling or offset for example). Finally, $\rho(\cdot)$ is an influence function, that serves for weighting residuals; $\rho(x) = x^2$ corresponds to a least squares cost function which is highly non-robust. Here, we use robust influence functions [21], for example the Huber-function, that downweight the influence of outliers, which thus allows to handle specular highlights in some images etc.

Optimization is done for the T_{kp}, α_{ik}, and β_{ik} and is carried out using an M-estimator [21] (IRLS, Iteratively Reweighted Least Squares). The estimation is initialized as follows: we initialize the T_{kp} by the average of the corresponding input greylevels I_{ikp}, i.e.:

$$T_{kp} = \frac{1}{n}\sum_{i=1}^{n} I_{ikp}$$

We then compute the initial values for the affine transformation coefficients α_{ik} and β_{ik} by minimizing (7) over these parameters (keeping the T_{kp} fixed), and with $\rho(x) = x^2$ as influence function. This is thus a linear least squares problem, solved using an SVD (Singular Value Decomposition) [22]. After this initialization, we optimize the T_{kp}, α_{ik}, and β_{ik} using IRLS, as mentioned above, now using a robust influence function for ρ. This proceeds in iterations, as follows [21]: at each iteration, we first compute weights w_{ikp} by evaluating the influence function for each residual (each term $I_{ikp} - \alpha_{ik}T_{kp} - \beta_{ik}$ in (7)), to be precise by evaluating it at residuals after they are scaled by a global factor (see [21] for details). Then, we solve the weighted least squares problem:

$$\sum_{p=1}^{m}\sum_{i=1}^{n}\sum_{k=R,G,B} w_{ikp}(I_{ikp}-\alpha_{ik}T_{kp}-\beta_{ik})^2$$

This is a non-linear least squares problem, which we solve using the Levenberg–Marquardt method [22]. Here, we exploit the sparse structure of the normal equations to

drastically reduce the computation time, as it is common practice for e.g. bundle adjustment, cf. [7].

The process of computing weights and solving the weighted least squares problem, is iterated until convergence (we use a small, fixed number of iterations, which proved to be sufficient in practice). This optimization process is rather time-consuming and we thus do not use it routinely for testing hypotheses of surface patches. However, it is sometimes employed as such to generate texture maps for the final surface mesh, depending on the desired visual quality.

4.3 Overall procedure for mesh generation

We have so far described the geometric and photometric constraints used for mesh generation. The overall process is as follows. One or several "seed triangles" are created. This can be done manually, since it creates little overhead, but we also tried a simple automatic procedure: determine the smallest roughly equilateral triangles in the reconstructed point cloud, and accept them if they have a good photoconsistency measure. The mesh is thus initialized as the set containing one or several such seed triangles. The edges that are at the border of the mesh, are stored in a list. As soon as the list is not empty, the following operations are run. We first randomly pick one edge of the list. Then, all 3D points are determined for which the triangle formed by a point and the edge is not too thin (no angle smaller than 10°, for example). Sort these points by increasing distance to the edge. For the closest point, check if the triangle it would form with the edge, satisfies all geometric and photometric constraints (see above). If this is the case, accept the triangle in the mesh and update the associated data structures (e.g. remove the edge from the list, and add the new outer edges to it). In the opposite case, proceed with the next point. If none of the points satisfies all criteria, the edge is removed from the list. After having thus processed one border edge of the mesh, we iterate by randomly selecting another one, as explained above.

4.4 Other mesh generation approaches

Other approaches for mesh generation were also developed. Instead of iteratively growing a surface mesh, an alternative approach is to perform a volumetric reconstruction: starting with a discretization of 3-space (typically, a 3D Delaunay tetrahedrization), one iteratively prunes volumes (here, tetrahedral), based on similar constraints as those used above. The outer surface of the final volume is then taken as surface mesh. Constraints used for pruning tetrahedra are visibility constraints (same as above), and photoconsistency constraints, which are now applied differently: a tetrahedron is kept if its visible faces have a good photoconsistency score; if a face has a low score, the tetrahedron is pruned only if this would increase the photoconsistency of the entire model (pruning a tetrahedron makes other tetrahedra visible, which might have an even lower score). This is still work in progress, but it has already produced better results than the mesh-growing method in cases where feature points were not tracked over many frames. In the opposite case, however, the mesh-growing method tended to stick better to the true object surface.

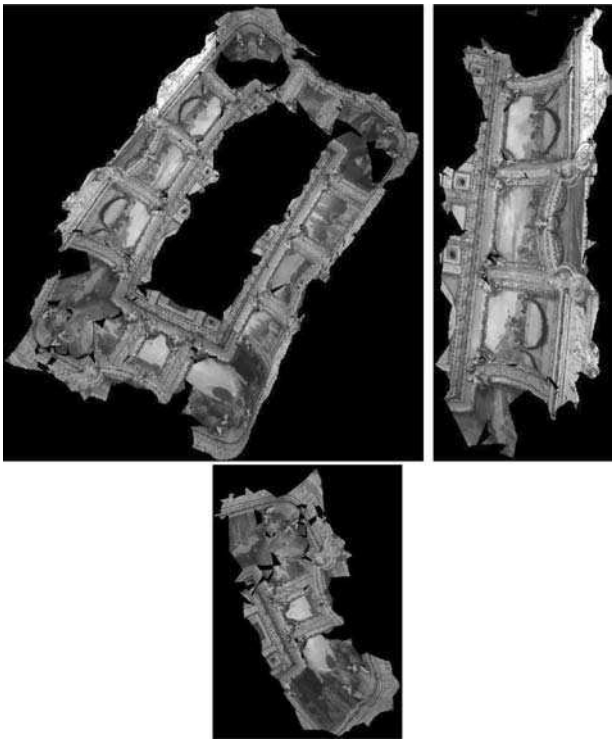


Fig. 4 Ceiling of the Casino Royal Hall. Raw Automatic 3D Model

4.5 Refining a mesh

We recently have developed an approach to refine 3D structure to get more dense reconstructions. The approach is inspired by [23], and roughly works as follows. 3D structure is estimated as a set of depth maps (as opposed to using a single depth map as in [23]). These are initialized from the coarse triangular mesh, or even by directly interpolating from the 3D point cloud. Then, they are optimized based on photoconsistency and visibility reasoning, as well as a 3D shape prior. The method resembles that of [23], but many details are novel. It would be beyond the scope of this article to completely describe this approach; it will be published separately. Results are shown in Sect. 6.

Results are shown in Figs. 4–6. Figure 4 shows a detail of the Casino scene, cf. Fig. 8. Figures 5 and 6 show results for outdoor scenes. Especially the scene of Fig. 6 is very challenging, due to the enormous depth discontinuities. For Fig. 5, the calibration provided by the authors of [23], was used, whereas for Fig. 6, the self-calibration and reconstruction tools described in this paper, were applied.

5 The authoring tool

Fully automatic 3D reconstruction of complex environments is currently not a practical possibility for a number of

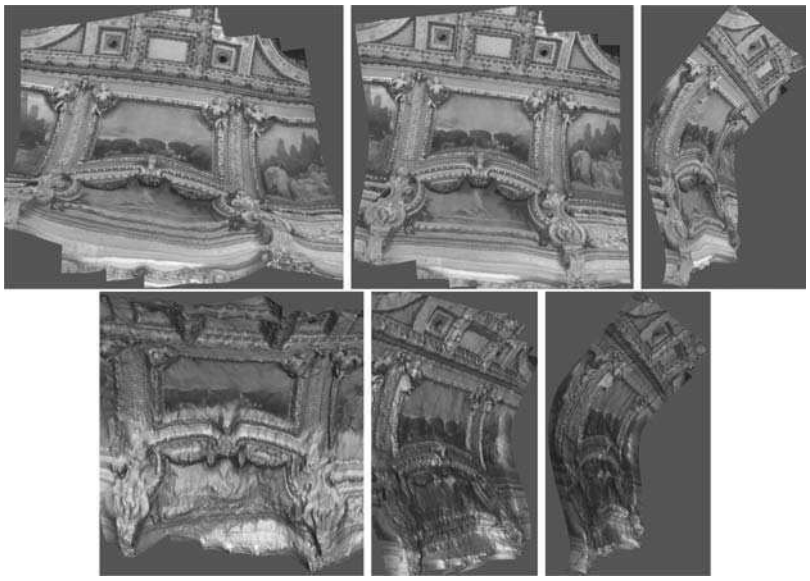


Fig. 5 Casino sequence. *Top row*: rendered images. *Bottom row*: rendered images with artificial specular component added to textures, to better show the underlying geometry

Fig. 6 Cityhall sequence [23]. *Top row*: the three input images. *Middle and bottom rows*: rendered images

reasons. Manual intervention is still required whenever CV software is not able to cope with singular situations, when automatic procedures are too slow, or simply when users desire to add a personal touch; there are situations when a few keystrokes may save hours of computational time or significantly improve the quality of the results. The VISIRE Authoring Tool (VAT) has been designed specifically to operate the underlying CV software. The tool (Fig. 7) supports the following functions: manipulate the video material required to construct a 3D model, guide and operate automatic CV processes selecting the various parameters, and visualize 2D and 3D results.

The VAT offers options to monitor the different CV steps, visualize intermediate results, and modify parameters

Fig. 7 VISIRE Authoring Tool Screenshot

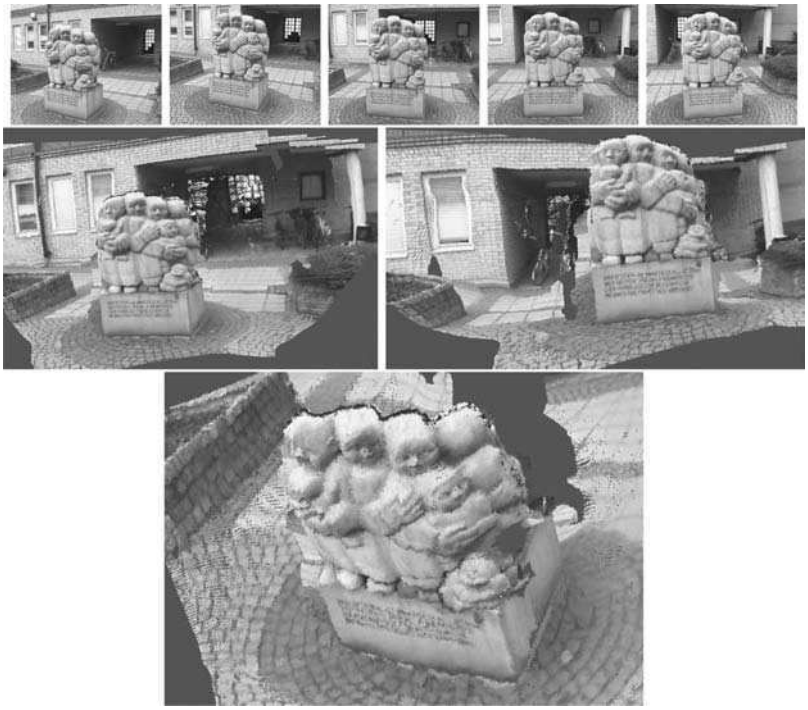


Fig. 8 Statue sequence. *Top row*: the five input images. *Middle and bottom rows*: rendered images. The background is not perfectly reconstructed, since much of it is only seen in two or three images

with the aim to improve the finished results. The user may, for example, visualize trajectories in the track editor and manually insert or delete features that are automatically tracked by the system. The VAT follows a "project" approach: a project gathers all the information necessary to manipulate and process one or more video sequences. When the user modifies a parameter, the system automatically recalculates obsolete links and updates the resulting 3D mesh.

Project information is stored in three main structures: Videos (2D image sequences), Tracks (structures containing the information required to track individual features in one or multiple video sequences), and Geometries (3D points, 3D meshes, and textures). The process of constructing a 3D model is progressive. Different parts of the scenario are processed independently and then stitched together. Three containers store the respective structures as they are created and allow the user to manipulate them using a

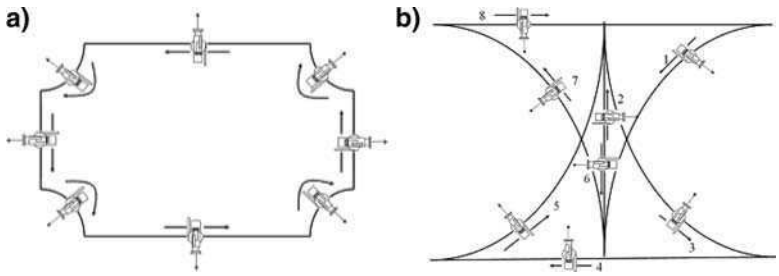


Fig. 9 Example of a shooting plan: **a**) Large room, **b**) Small room

drag and drop interface. Several combinations are possible: object structures can be added, deleted, and joined (i.e. it is possible to join videos, tracks, or geometries). This last is a unique feature since it is almost impossible to cover a complete scenario in a single shot.

The VAT implements the following additional characteristics: integrated video editor, support for multiple input video sequences, compatible with most popular video formats, VRML output, integrated 3D browser, multiple interactive tools (feature analysis, calibration, 3D rendering, texture processing), and hot display capabilities (the result of changed parameters is immediately updated in the 3D model).

6 Results and evaluation

An exhaustive evaluation of VISIRE methods and software in real conditions was performed using video material acquired in different museums (Casino de Madrid, Uffizi Gallery, and Palazzo Pitti in Florence), as well as available test image libraries. Early in the project stage it was acknowledged that image shooting procedures would become critical and might determine the quality of the resulting 3D models. For that reason precise guidelines (Fig. 9) were produced so that any person with minimum skills and basic training could do the shooting. In most cases a mid size room could be completed in no more than 2 h if guidelines are closely followed. Tests were made using professional digital Betacam cameras and domestic minDV camcorders. Due to unusual shooting requirements, miniDV cameras performed better than their professional counterparts since miniDV cameras are lighter and easier to aim and differences in image quality were hardly noticeable.

The abundant video material compiled was used to construct several complete 3D models. Experience showed VISIRE performs better in richly textured scenarios. Painted walls (i.e. frescos) or textured materials (i.e. marble) produced very good results. The system offers remarkable accuracy in the reconstruction of vaulted scenarios and irregular shapes where human modellers would have difficulties to achieve similar results. In planar surfaces where extremely good accuracy in the assembly of the geometry is required and even small errors are highly noticeable, performance was more noisy; but nevertheless very good for an automatic method.

As compared with the state of the art, VISIRE does a great job handling occlusions by using information from alternative views when part of the scenario is occluded. However, in certain circumstances the algorithm incorrectly joins polygons from different objects. In most cases, those problems are due to irregular sampling and can be corrected manually during post-processing (i.e. inserting seed features manually). VISIRE also has difficulties with reflections (i.e. mirrors) or unstable illumination. It must be considered the system was required to use handheld cameras and no professional lighting or measurement devices were allowed. Ill posed situations, can be found in some cases due to improper shooting of the images; most frequently when disparity between the acquired images is not sufficient. Fortunately, the system detects this situation automatically. There were also rare situations when the algorithm joining sequences did not perform as expected and the model required manual adjustments for a proper joining.

Generally speaking, VISIRE textures and lighting are more realistic than human produced models. The geometry, while closer to the real thing, tends to be more noisy and error prone. In that sense the system obtains a typical RMS of no more than 0.338 in the reprojected points after projective bundle and a maximum of 0.5 after Euclidian bundle. Those figures are very good considering they were obtained using fully automatic methods.

Results are shown in Figs. 4–6 and Fig. 10. Figure 4 shows a detail of the Casino scene, cf. Fig. 8. Figures 5 and 6 show results for outdoor scenes. Especially the scene of Fig. 6 is very challenging, due to the enormous depth discontinuities. For Fig. 5, the calibration provided by the authors of [23], was used, whereas for Fig. 6, the self-calibration and reconstruction tools described in this paper, were applied. In Fig. 10 an almost complete model of the ceiling of the Palazzo Pitti is presented.

Next, we will analyze in more detail one of the video sequences evaluated (Fig. 8). The Casino sequence consisted of four slightly overlapping shots describing the edges of a near-rectangular camera path. The whole sequence lasts several min and was in the process reduced to 281 views and 16,000 3D points. In order to bundle efficiently, only the 835 longest point tracks were kept in the global bundle adjustment. Figure 11a shows the magnitude of the trajectory mismatch is such that usual bundle turned out to be insufficient. Results improved dramatically when constraint enforcement was performed as it is apparent in Figs. 11b

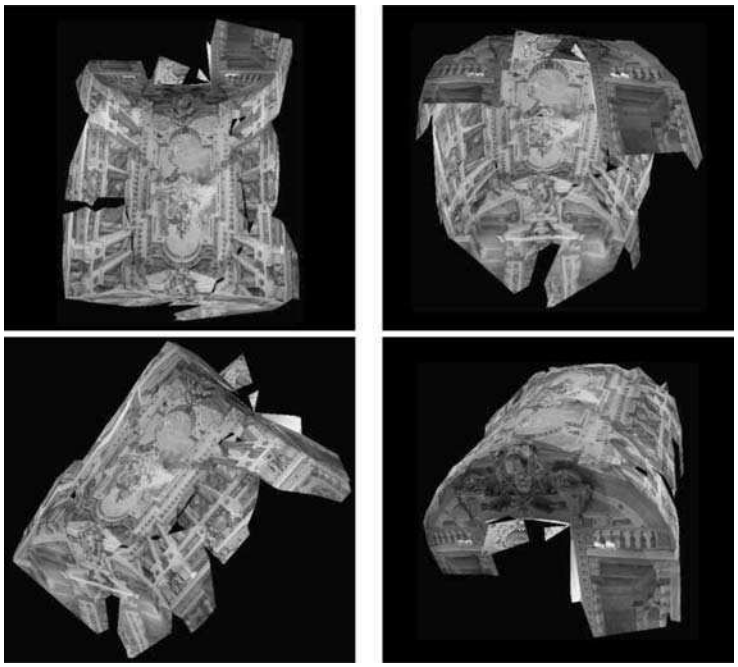


Fig. 10 Different views of a model from Museo degli Argenti in Palazzo Pitti

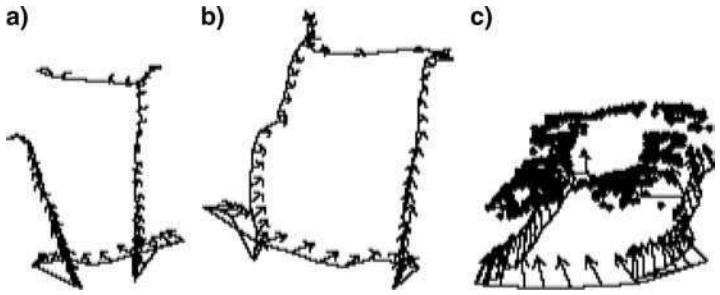


Fig. 11 Casino sequence: **a** open sequence trajectory, **b** trajectory after closing, **c** final reconstruction of closed sequence with 3D points

Table 1 Reprojection errors (RMS) for the Casino sequence

(Pixels)	Open sequence	Constraint enforcement	Closed sequence
Overall	0.3340	15.54	0.3667
Closure	777.6	10.55	0.6098

and c. The first row in Table 1 presents the overall RMS reprojection error for all image points, while the sequence is still open, after imposing the closure constraint and after bundle convergence of the closed sequence, respectively. The second row provides similar information when reprojecting the five 3D points used for merging (hence visible in both the last and the first images) onto the first image. Note that the overall reprojection error increases when the constraint is enforced, since the error on the constraining image points is in a sense distributed on the whole structure. The important measure is, however, how well the algorithm subsequently minimizes the reprojection error, which is seen to fall very close to the error for the non-constrained structure.

7 Conclusion

In the paper a “complete" approach to 3D reconstruction in real environments has been presented. The VISIRE system succeeded in the use of video information, acquired from handheld camcorders, as input to a near-automatic 3D mesh generation system. A full functional authoring tool has been developed to allow graphics professionals create photo-realistic 3D models with less effort and better quality than it was possible before. The system is specially well suited for scenarios where architecture is rich and textured, with few first plane objects occluding the view. Vaulted scenarios, painted walls, or irregular geometries are excellent candidates, while scenarios with simpler geometries, fewer textures, or showing big symmetries (i.e. where replicated geometrical primitives can be used) are better suited for manual modelling. In general, best performance is obtained when VISIRE is used as an initial automatic step that is completed by human post-processing. To some extent manual and automatic processes are

complementary, but there still exists important challenges as to how to combine both approaches in a most efficient way.

Even if VISIRE achieved important advances, the field is still open to new improvements. In particular, it will be highly desirable to implement methods to apply constraints (i.e. planarity and pure rotations) to the produced 3D models in order to simplify the 3D reconstruction process and improve the accuracy. One of our goals for future research is to combine 3D and IBR approaches; applying IBR for scene parts whose geometry cannot be modelled well. Another possible point of improvement is to develop more advanced methods for representing objects. Instead of triangular meshes, more general object surfaces could be used, e.g. level-set frameworks. Here it would also be desirable to include other surface properties, such as reflectance and local geometry. More research will be also required to improve some of the still pending problems arising from: severe occlusions, texture-less or repetitive structures, irregular sampling, etc. Finally, it must be mentioned it still remains a difficult task to acquire sequences that cover every part of a scene from several viewpoints. It is imaginable to develop a system that detects parts of the scene that were not modelled precisely, and guides the user to acquire the additional video material.

Our main conclusion is the system is capable to perform a good job in a broad range of conditions. There are still problems related to CV, which presents limits that may cause certain defects on the reconstructed models. This is certainly a limitation, but is well compensated by the automatic functioning of the system and the processing speed, which allows to achieve a model of reasonable good quality in a very short time. In our opinion the system can indeed simplify the process of building up 3D models or at least provide a good prototype of a model to work on.

References

1. Rodríguez, T., Sturm, P., Heyden, A., Menéndez, J.M., et al.: Visire. photorealistic 3d reconstruction from video sequences. In: IEEE International Conference on Image Processing, pp. 705–708. Barcelona, Spain (2003)
2. Pollefeys, M., Gool, L.V., Vergauwen, M., Verbiest, F., Cornelis, K., Tops, J., Koch, R.: Visual modeling with a hand-held camera. IJCV **59**(3), 207–232 (2004)
3. Gortler, S., Grzeszczuk, R., Szeliski, R., Cohen, M.: The lumigraph. In: Proceedings of the 23rd Conference on Computer graphics and Interactive Techniques, pp. 43–54 (1996)
4. Matusik, W., Pfister, H., Ngan, A., Beardsley, P., Ziegler, R., McMillan, L.: Image-based 3d photography using opacity hulls. In: Proceedings of the ACM SIGGRAPH 2002, p. 427–437 (2002)
5. Takashi Machida, H.T.: Dense estimation of surface reflectance properties based on inverse global illumination rendering. In: ICPR'04, vol. 2, pp. 895–898. Cambridge, UK (2004)
6. Shi, J.C.T.: Good features to track. In: CVPR'94, pp. 593–600 (1994)
7. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press, Cambridge, UK (2000)
8. Heyden, A., Åström, K.: Euclidean reconstruction from image sequences with varying and unknown focal length and principal point. In: Proceedings of the Conference on Computer Vision and Pattern Recognition, pp. 438–443 (1997)
9. Triggs, W., McLauchlan, P., Hartley, R., Fitzgibbon, A.: Bundle adjustment: A modern synthesis. In: Vision Algorithms: Theory and Practice. Springer, Berlin Heidelberg New York (2000)
10. Fitzgibbon, A., Zisserman, A.: Automatic camera recovery for closed or open image sequences. In: Proceedings of the European Conference on Computer Vision, vol. I, pp. 311–326. Freiburg, Germany (1998)
11. Triggs, B., McLauchlan, P., Ri, H., Fitzgibbon, A.: Bundle adjustment—a modern synthesis. In: Vision Algorithms'99, pp. 298–372. in conjunction with ICCV'99, Kerkyra, Greece (1999)
12. Reid, I., Murray, D.: Active tracking of foveated feature clusrters using affine structure. In: International Journal of Computer Vision, pp. 41–60. Seattle, WA (1996)
13. Sturm, P., Triggs, B.: A factorization based algorithm for multi-image projective structure and motion. In: Buxton, B., Cipolla, R. (eds.) Computer Vision – ECCV'96, Lecture Notes in Computer Science, vol. 1065, pp. 709–720. Springer, Berlin Heidelberg New York (1996)
14. Tomasi, C., Kanade, T.: Shape and motion from image streams under orthography: a factorization method. Int. J. Comput. Vis. **9**(2), 137–154 (1992)
15. Guilbert, N., Bartoli, A.: Batch recovery of multiple views with missing data using direct sparse solvers. In: British Machine Vision Conference. Norwich, UK (2003)
16. Triggs, B.: Linear projective reconstruction from matching tensors. Image Vis. Comput. **15**(8), 617–625 (1997)
17. Hoppe, H.: Surface reconstruction from unorganized points. Ph.D. thesis, Department of Computer Science and Engineering, University of Washington (1994)
18. Petitjean, S., Boyer, E.: Regular and non-regular point sets: Properties and reconstruction. Comput. Geom.—Theor. Appl. **19** (2001)
19. Manessis, A., Hilton, A., Palmer, P., McLauchlan, P., Shen, X.: Reconstruction of scene models from sparse 3d structure. In: Proceedings of the Conference on Computer Vision and Pattern Recognition. Hilton Head, USA (2000)
20. Morris, D., Kanade, T.: Image-consistent surface triangulation. In: Proceedings of the Conference on Computer Vision and Pattern Recognition. Hilton Head, USA (2000)
21. Huber, P.: Robust Statistics. Wiley, New York (1981)
22. Press, W., Flannery, B., Teukolsky, S., Vetterling, W.: Numerical Recipes in C. Cambridge University Press, Cambridge, UK (1988)
23. Strecha, C., Fransens, R., Gool, L.V.: Wide-baseline stereo from multiple views: a probabilistic account. In: Proceedings of the Conference on Computer Vision and Pattern Recognition, pp. 552–559 Washington, DC (2004)

Tomas Rodriguez was born in Madrid in 1961. Bachelor in Physics and Master in Electronics by the Universidad Complutense de Madrid. He started his career in the private R&D sector in 1987, when he specialized in computer vision and parallel processing systems. In the early nineties he participated in the EVA project; one of the most outstanding traffic monitoring system of the time. For more than 10 years, he has been involved in international research projects within the ambit of EUREKA, ESPRIT, V and VI Framework programmes. During this time, he

coordinated eight international projects (CAMELOT, CITRONE, ON-LIVE, SAID, VISIRE, EVENTS, ITALES, HOLONICS) and acted as principal investigator in two additional ones (CITRUS and VICTORIA). Since the early days, he had the opportunity to collaborate with some of the most prestigious research institutions in Europe: Franhoufer Inst., INRIA, CNRS, University of Oxford, University of Lund, DFKI, Siemens C-Lab, Philips Research Labs, etc. Evaluator of R&D projects for the Spanish Ministry for Science and reviewer of international scientific journals, he is currently the R&D manager and coordinator for European projects at Eptron SA. His recent interests include: computer vision, real time software, industrial control, parallel processing, iTV, and mobile technologies, etc.

Chapter 13

3D Reconstruction of Specular Surfaces

Paper 34 [6]: T. Bonfort and P. Sturm. Voxel carving for specular surfaces. In *Proceedings of the 9th International Conference on Computer Vision, Nice, France*, volume 1, pages 591–596. IEEE Computer Society Press, October 2003.

Paper 35 [7]: T. Bonfort, P. Sturm, and P. Gargallo. General specular surface triangulation. In *Proceedings of the Asian Conference on Computer Vision, Hyderabad, India*, volume II, pages 872–881, January 2006.

Paper 36 [27]: P. Sturm and T. Bonfort. How to compute the pose of an object without a direct view? In *Proceedings of the Asian Conference on Computer Vision, Hyderabad, India*, volume II, pages 21–31, January 2006.

Voxel Carving for Specular Surfaces

Thomas Bonfort and Peter Sturm
MOVI - GRAVIR - INRIA, Grenoble, France
thomas.bonfort,peter.sturm@inrialpes.fr

Abstract

*We present an novel algorithm that reconstructs voxels of a general 3D specular surface from multiple images of a calibrated camera. A calibrated scene (*i.e. *points whose 3D coordinates are known) is reflected by the unknown specular surface onto the image plane of the camera. For every viewpoint, surface normals are associated to the voxels traversed by each projection ray formed by the reflection of a scene point. A decision process then discards voxels whose associated surface normals are not consistent with one another. The output of the algorithm is a collection of voxels and surface normals in 3D space, whose quality and size depend on user-set thresholds. The method has been tested on synthetic and real images. Visual and quantified experimental results are presented.*

1. Introduction

3D shape reconstruction techniques obtain models of real-world objects, that can then be used in computer graphics, CAD, multimedia databases, etc... Most reconstruction methods rely on the identification and matching of pixels corresponding to a same object feature, and return the 3D coordinates of the object's feature using different geometric constraints. In the case of specular objects, these non-specific methods will always fail to reconstruct the object's surface. Indeed, the observed texture of a specular object is the reflection of the object's surrounding environment, rather than the texture of the object itself. As the viewpoint changes, the observed texture *moves* along the object's surface, thus invalidating the geometric constraints used by classical reconstruction methods.

This article describes a method recovering points of a specular surface. Texture and shading contributions of the surface are ignored, our focus being only on perfect mirrors like objects made of polished metal. We place ourselves in the case of several views of calibrated cameras observing the specular surface, the images seen by these cameras being the reflection of the object's surrounding environment. We assume that the object's environment (typically a printed target, from now on referred to as *scene points*) is calibrated, *i.e.* we know the 3D coordinates of a number of scene points.

A realworld application of our reconstruction method uses a printed target attached to a camera taking images of a specular surface. Camera pose is obtained by taking images of the system from a stereo rig. Once the matching of the points on the target with the image of their reflection has been accomplished, a set of voxels belonging to the surface is obtained by our method, along with their corresponding surface normals. A model of the object can then be obtained by fitting a surface to the obtained points and normals.

1.1. Previous Work

Specularities have interested researchers in the field of computer vision for the past 20 years. For example, Blake *et al.* [1] studied the disparity of highlights on a specular surface seen from two viewpoints. Zisserman *et al.* [17] tracked the motion of specularities to obtain information on the surface. Healey [5] used static images and a reflectance map to recover 3D points on a specular surface. In [8], Oren and Nayar studied the classification of real and reflected features, and recover the profile of a specular surface by tracking an unknown scene point. The profile can be recovered without further hypotheses only if the motion of the camera, the scene point and its reflection in the mirror surface are coplanar, thus limiting practical applications. Halstead *et al.*, in [3], fit a spline surface to a set of normals, iteratively refining the result. Their method requires an initial seed point on the specular surface, and was applied to the sub-micronic reconstruction of the human cornea. Schultz recovers in [11] the ocean's surface given three calibrated images, an irradiance map of the illumination, and known seed points. An elevation map is obtained by propagating around the known points using observed surface normals, while minimising the difference between real and rendered images. In [9], Ripsman and Jenkin recover specular planes using three views and active illumination. The aimed application is the automatic inspection of orbital objects. Savarese and Perona detail in [10] the information available for one view of a specular object reflecting three or more intersecting calibrated lines. The second order surface geometry can be obtained up to one unknown parameter. In the case of general specular surfaces, Zheng and Murata in [16] reconstruct a rotating specular object by studying the motion of the illu-

mination created by two circular light sources.

The method we propose makes no assumptions on the specular surface, and doesn't need any initial seed points. It extracts voxels of the surface independently from one another, therefore preventing the accumulation of errors that can occur in other methods.

2. Geometric Constraints

This paragraph presents the basic geometric constraints used by our method. The following notation will be used throughout the article:

- $\mathbf{u}$ is a vector of 3D space, *i.e.* $\mathbf{u} = (x\ y\ z)^T$, and $\hat{\mathbf{u}}$ is a normalised vector.
- s is a scalar.
- subscripts are used as follows:
  - $\mathbf{O_c}$ is the *camera*'s projection centre.
  - $\mathbf{x_s}$ is a calibrated *scene* point.
  - $\mathbf{x_m}$ is the position of the reflection of $\mathbf{x_s}$ on the *mirror* surface (therefore it depends on the viewpoint).
  - $\mathbf{x_i}$ is the projection of $\mathbf{x_m}$ on the *image* plane of the camera.

For the sake of simplicity, we assume that $\mathbf{x_i}$ is a point of 3D space, *i.e.* the 3D coordinates of the image point $\mathbf{x_i}$ when the position of the image plane is known (see figure 2). This assumption causes no loss of generality, as $\mathbf{x_i}$ is on the projection ray formed by $\mathbf{x_m}$ for a given camera viewpoint $\mathbf{O_c}$.

2.1 Ideal Specular Surfaces

We have chosen to consider only purely specular surfaces such as mirrors, objects made of polished metal, etc... With this assumption, the law of reflection shown in figure 1 links the surface normal $\hat{\mathbf{n}}_\mathbf{m}$, the incoming light direction $\hat{\mathbf{r}}_\mathbf{i}$, and the reflected light ray $\hat{\mathbf{r}}_\mathbf{r}$ by the following constraint:

$$\hat{\mathbf{r}}_\mathbf{r} = \hat{\mathbf{r}}_\mathbf{i} - 2(\hat{\mathbf{n}}_\mathbf{m}.\hat{\mathbf{r}}_\mathbf{i})\hat{\mathbf{n}}_\mathbf{m}$$

The formation of an image of a specular surface is shown in figure 2: given a known scene point $\mathbf{x_s}$, a known mirror point $\mathbf{x_m}$, and a known image point $\mathbf{x_i}$, the surface normal $\mathbf{n_m}$ at $\mathbf{x_m}$ is constrained by:

- $\mathbf{n_m}$ belongs to the plane formed by $\mathbf{x_s}$, $\mathbf{x_m}$ and the projection ray formed by $\mathbf{O_c}$ and $\mathbf{x_i}$.
- the angle α_i between the incident line of sight and the surface normal is equal to the angle α_r between the surface normal and the reflected line of sight.

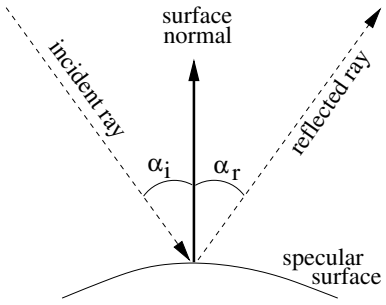


Figure 1: **Law of reflection.** The angle formed by the incident line of sight and the surface normal equals the angle formed by the surface normal and the reflected line of sight. The surface normal and the rays are coplanar.

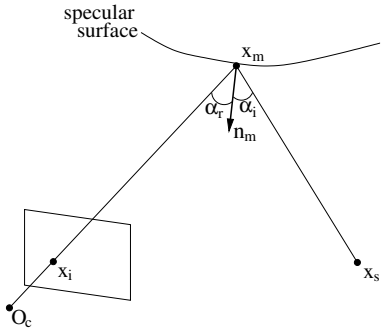


Figure 2: **Image formation.** A calibrated scene point is reflected by the specular surface onto the image plane of a calibrated camera.

Thus, the surface normal at $\mathbf{x_m}$ is given by the bisector of the angle formed by the scene point, the voxel and the camera. This constraint is on its own clearly insufficient to determine the surface's position and orientation: given any point on the projection ray formed by the camera's projection centre $\mathbf{O_c}$ and the image point $\mathbf{x_i}$, we can find the orientation of a specular surface passing by that point that would lead to the same observation. Inversely, given any surface normal, the position of a surface leading to the same observation can be obtained. Our aim being to reconstruct a specular surface, we need further constraints so as to find surface position and orientation simultaneously. The method we present in the following paragraphs uses multiple images of the specular surface, and is based on the fact that the normal at a given point on a specular surface is independent of the viewpoint.

3. Reconstruction Method

This section presents our reconstruction method, using multiple views of a calibrated camera. We start by discretizing the 3D space around the specular object, to obtain a voxelic representation of the working space. The reconstruc-

tion then takes place in two main phases: the first phase associates $n \geq 0$ normals to every voxel of the 3D space surrounding the specular surface. In the second phase, along each projection ray, the voxel whose associated normals are the most consistent with one another is kept. In essence, our method is very similar to Seitz and Dyer [12] or Seitz and Kutulakos [7] who rely on surface color to reject incorrect voxels.

3.1. Phase 1: Normal Accumulation Process

In the previous section, we have seen that a single view of the reflection of scene points was not sufficient to obtain a specular surface's position and orientation simultaneously.

Figure 3 shows the information available for one viewpoint, once the 3D space has been discretized: for each voxel traversed by the projection ray corresponding to an image point and a scene point,we compute the surface normal associated to that voxel. This surface normal corresponds to the normal of a surface passing through the centre of the voxel, that would produce the same observation. In other words, for a given scene point $\mathbf{x_s}$ reflected by the specular surface onto the image plane at $\mathbf{x_i}$, we associate for every voxel $\mathbf{x_m}$ traversed by the projection ray $[\mathbf{O_c x_i})$ the normal of a specular surface passing by the centre of that voxel, that would reflect $\mathbf{x_s}$ onto $\mathbf{x_i}$.

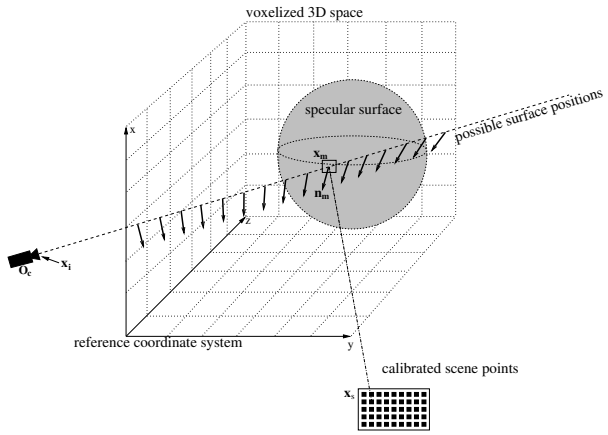


Figure 3: Normal construction for each voxel traversed by a projection ray. The process is repeated for each calibrated scene point $\mathbf{x_s}$, and for each camera position $\mathbf{O_c}$.

The normal calculation is repeated for every voxel traversed by every projection ray corresponding to the reflection of the scene points, and for every camera position. Each voxel in the 3D space surrounding the specular object will then have zero or more surface normals associated to it.

In the next paragraph, we will explain how the disparity of the surface normals stored for one voxel can determine whether the voxel belongs to the specular surface or not.

3.2. Phase 2: Discarding Incorrect Voxels

Let us consider the voxels traversed by the projection ray formed by the reflection of the scene point $\mathbf{x_s}$ by the specular point $\mathbf{x_m}$ onto the image point $\mathbf{x_i}$. There are two cases occuring for the computed normal at a given voxel:

- **case 1:** the voxel does not belong to the specular surface. In this case, the associated normal has no physical reality. If the voxel is traversed by a second ray originating from another viewpoint, the second associated normal has no reason to be similar to the first one.
- **case 2:** the voxel belongs to the specular surface. In this case, (assuming the voxelic discretization is infinitely precise, and there is no noise in the measures of $\mathbf{x_s}$ and $\mathbf{x_i}$), the associated normal is the normal of the specular surface at that point. If the voxel is traversed by another projection ray, the voxel's second associated normal will be identical to the first one.

Intuitively, voxels belonging to the surface can be determined by looking at the disparity of their associated normals, as illustrated on figure 4. Furthermore, as for each viewpoint a single correct voxel corresponds to each projection ray, incorrect voxels can be discarded by stepping along each projection ray and keeping the voxel with the least disparity.

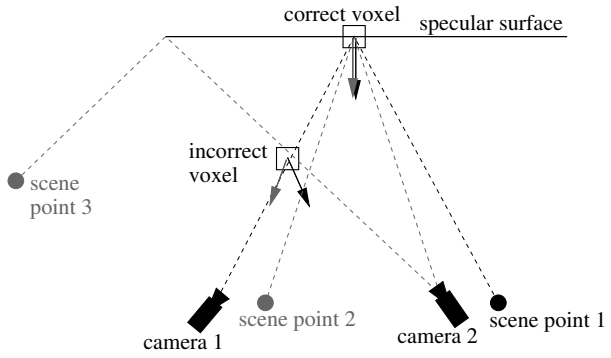


Figure 4: Normals associated to a correct and incorrect voxel. The normals associated to a voxel that does not belong to the specular surface are dissimilar, whereas those associated to a voxel belonging to the surface are consistent with one another.

Quantifying Normal Disparity

So as to discard incorrect voxels, a quantifiable disparity measure is necessary. The simplest disparity measure is given by the mean angle between each normal and the mean normal:

$$disparity = \frac{\sum_{i=1}^{n} \arccos\left(\hat{\mathbf{n}}_\mathbf{i} . \overline{\hat{\mathbf{n}}}\right)}{n} \qquad (1)$$

where $\overline{\hat{\mathbf{n}}}$ stands for the mean normal associated to a voxel, *i.e.*

$$\overline{\hat{\mathbf{n}}} = \frac{\frac{\sum_{i=1}^{n} \hat{\mathbf{n_i}}}{n}}{\left\| \frac{\sum_{i=1}^{n} \hat{\mathbf{n_i}}}{n} \right\|}.$$

This disparity measure is in practice insufficiently discriminant. As well as normal disparity, depth information must be taken into account: as illustrated in figure 5, the distance of a voxel to the cameras plays an important role in the disparity associated to a voxel's normals. Two different approaches have been used to obtain correct results, as explained in the following paragraphs.

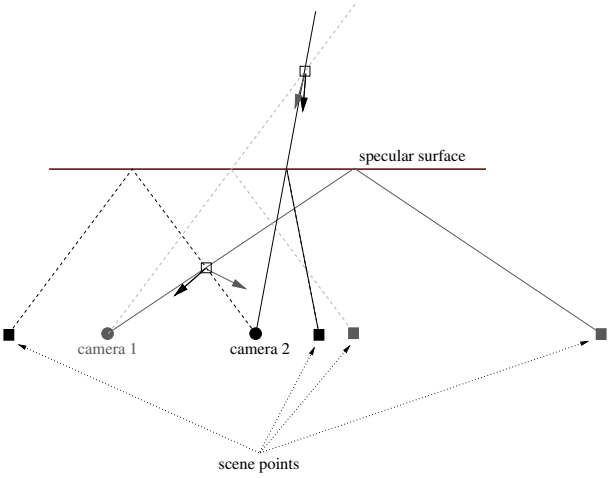


Figure 5: Disparity variation due to camera distance. Two equally incorrect voxels have an associated normal disparity that depends on their distance to the cameras. A threshold-based decision will tend to keep incorrect voxels if they are further away from the camera viewpoints.

Reprojection Error

The error measure seen in (1) relies only on the disparity of the normals stored at a voxel to determine if the voxel is correct. Additional information is available if we compare the reconstructed surface to the images we have of it, by checking that the images of every scene point reflected by a given voxel (whose surface normal is the mean normal seen in the previous paragraph) are close in the participating cameras to the images of the same scene point reflected by the original surface.

A given voxel is assumed to be a plane passing by the centre of the voxel, whose normal is the voxel's mean normal. Every participating scene point associated to the voxel is then geometrically reflected by this plane onto the image plane of the participating cameras. The reprojection error measure becomes the mean square distance of the reflections to the original pixel.

This error measure does not take normal disparity into account, and therefore is not sensitive to depth variations. Qualitatively correct reconstructions have been obtained with this method, but even better results have been obtained with the following heuristics.

Heuristic Disparity Measures

We have tested several disparity measures that take depth information of the voxels into account:

- Because of the perspective projection used by cameras, voxels that are close to the region where the views were taken are associated with a higher number of normals than those which are far away. A first working disparity measure is to divide the variance calculated in (1) by the number of associated normals, which will have the effect of associating greater disparity measures to the further voxels.

- The disparity measure that leads to the best results is obtained by dividing the disparity obtained in (1) by the mean angle formed by the scene points, the voxel, and the different camera viewpoints. As these angles tend to decrease as the voxels are further from the camera viewpoints, this disparity measure penalises the furthest voxels. The results we present were obtained using this measure in the next sections.

Limitation

The reconstruction of a specular object is not straightforward: figure 6 shows that a projection ray that intersects the object's surface twice will cause correct voxels to be associated with incorrect normals, thus invalidating a decision process based on normal disparity. This imposes that the reconstruction take place in several steps, or that a robust disparity measure be used: a correct voxel will be associated with a number of inconsistent normals originating from a double intersection, plus a number of identical normals corresponding to the surface normal. Thus, this limitation could be overcome by selecting consistent normals using robust statistics, and applying the disparity measure on these selected normals. The results we present in the next sections were obtained by placing the cameras in positions assuring no double intersections took place, therefore reconstructing a patch on the specular surface.

4. Results on Simulation Data

We have tested our method on the reconstruction of specular spheres and planes, using simulation data and ray-traced images. We deliberately limited ourselves to these simple

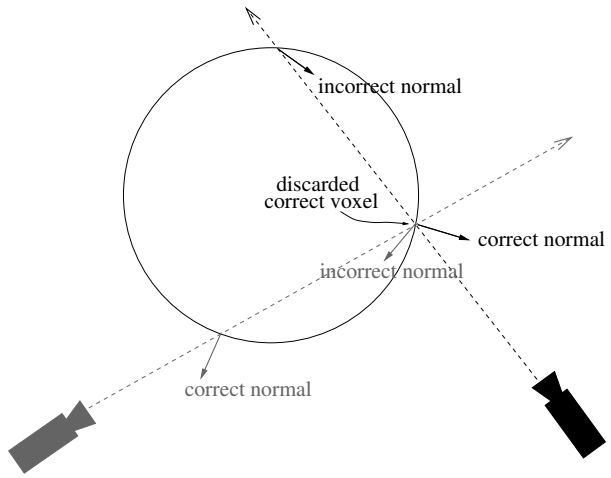


Figure 6: double intersections with the object's surface cause correct voxels to be associated with inconsistent normals, and therefore be discarded when measuring their disparity.

objects so as to easily and automatically perform the correspondence problem between the scene points and the image points.

4.1. Setup

The results shown in the next paragraph were obtained using 40 ray-traced images of a specular sphere. A black on white checkerboard target was reflected by the sphere onto the image plane of the cameras, and a Harris corner detector [4] was used to extract the corners in the obtained images, thus creating some noise in the measures. The extracted points were then analytically matched with their corresponding scene points on the target, using the known geometry of the sphere.

4.2. Quantified Results

We have tested our reconstruction method with different thresholds, as shown in table 1, using the heuristic disparity measure seen above. The quality of the output depends on the minimum number of normals a voxel must be associated with before the disparity measure can be trusted. With less than 4 normals per voxel, a number of outliers can appear. With at least 8 normals, the extracted voxels are never further than two voxel to the original surface.

4.3. Visual Results

The images we present in the appendix on page 8 show the distribution of the extracted voxels on the specular sphere's surface. We have chosen to show all the voxels below a given disparity measure, and not only the best voxel on each projection ray, to show how this disparity evolves around the specular surface.

	strict	medium	laxist
4	363 93.4%	969 95.3%	3103 94.45%
8	161 100%	589 100%	1982 100%

Table 1: Results obtained using strict, medium and laxist thresholds on normal disparity. The lines represent the minimum number of normals associated to a voxel before this voxel can be considered. For each combination, the first number is the number of accepted voxels, and the second one is the percentage of these voxels that are either intersecting the specular surface or less than two voxels away from an intersecting voxel.

Figure 11 shows the extracted voxels for a very strict tolerance on normal disparity. Despite the relatively small number of extracted voxels, surface shape can be recovered thanks to the homogeneous distribution of the voxels on the surface.

Figure 13 shows a medium threshold on normal disparity. A greater number of voxels are extracted, and no outliers have appeared.

Figure 14 shows the result after a laxist threshold on normal disparity. Practically all the intersecting voxels of the surface have been extracted, but a greater number of adjacent voxels have appeared. No outliers were extracted.

Figure 15 shows the following extracted voxels when the threshold is increased again. We see that the extracted voxels are further from the surface, however never being extremely false.

Figure 12 shows a cut through the reconstruction of the specular sphere.

5. Results on Real Images

We have tested our method on the reconstruction of a part of a specular spoon made of low end polished metal, thus exhibiting consequent normal variation on some parts (see figure 7, a little left from the center for example). In the next paragraphs we present the experimental setup, detail the calibration process which is non-trivial due to the limited information available, and present a couple of visual results of the reconstruction.

5.1. Experimental Setup

Figure 8 shows the experimental setup. A color coded [6] printed target was attached to an Olympus C2500 digital camera taking images of the specular spoon from 56 different viewpoints, while a stereo rig was used to obtain the pose of the target + camera system for every viewpoint. The unique color code allowed us to automatically match a pixel in the image corresponding to a circle, to the point on the target that was reflected onto this pixel. We then extracted

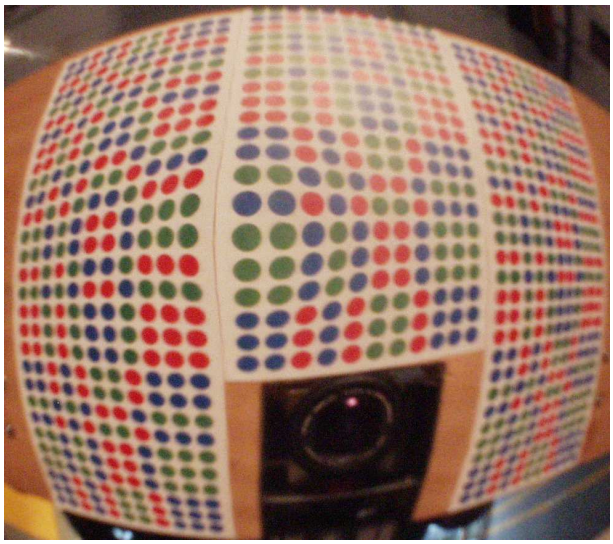


Figure 7: One of the 56 images used for reconstruction. Around 2 out of 3 colored circles were automatically extracted for this image.

1890 voxels corresponding to the specular surface out of a 100x100x100 cube of voxels, each of these voxels being 1 mm^3. We wish to point out that we used low end imaging devices, and that the automatic segmentation of the images for feature extraction induces consequent noise in the location of the pixels corresponding to the reflection of the target points. Therefore, pose estimation of the printed target and the camera at the different viewpoints, and projection lines corresponding to pixels were unlikely to be very accurate, making us believe that this reconstruction method is relatively robust to medium noise.

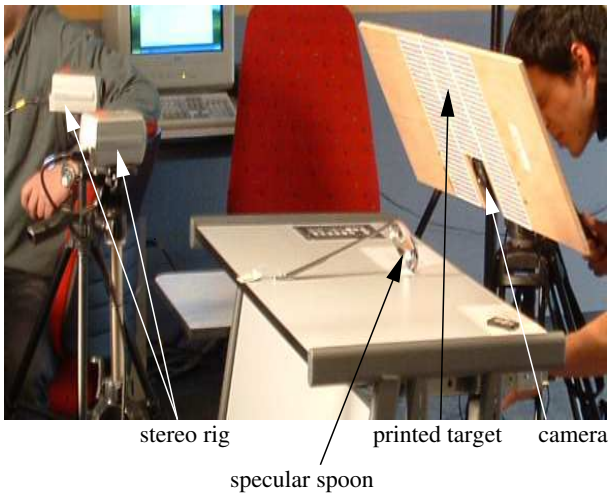


Figure 8: Experimental setup. The printed target is reflected by the specular sphere onto the image plane of the camera. The system's pose is determined by the stereo rig.

One point that should be mentionned is blur. The images we used exhibited large out of focus regions, thus complicating feature extraction on the specular surface. While this defocus did not prevent the reconstruction, having multiple images with a different focus setting, for every viewpoint, would be a simple method to resolve the problem.

5.2. Calibration

The main device used is the combination of the digital camera and the target plane attached to it, whose reflection in the specular object is what the camera sees. For each image acquired using this device, we need to know the (relative) pose of both, camera and target plane. Since they are rigidly linked, the problem reduces to determining the pose of either one of them for the current image, and the relative pose between them, that is fixed.

Several potential solutions come readily to mind. For example, we might put the specular object on a planar platform with targets printed on it. The camera's pose could then be computed using these targets. We rejected this solution because due to the specular object's convex form, the camera had to zoom deeply onto the object, in order to extract the reflected patterns in the image. So, it was not sure that targets on the platform would be systematically visible.

We thus chose to use a fixed stereo system, placed behind the specular object and that observes the camera + target device (see figure 8). The stereo system gives us the pose of the target plane for each acquisition position (the method of [14] was used). Actually, the stereo system was calibrated (intrinsics and relative pose) using directly the images of the target plane acquired during the experiments (using the methods described in [15, 13, 14]).

The remaining problem, estimating the relative position of the camera and the target plane, is not trivial, since the camera has no direct view of the targets. We solved the problem in a way analogous to [2]. Instead of the specular object, we let the camera observe a 3D calibration grid. A dozen images were acquired. The camera's and the target plane's ego-motions across the viewing positions, were computed as follows. For the camera, this is done using the 3D calibration grid and a classical camera calibration + pose algorithm. As for the target plane, its pose and thus ego-motion is output by the stereo system, as described above. The two sequences of ego-motions were then aligned using a method inspired from [2] and whose formulation is closely related to classical hand-eye calibration. This alignment transformation allows finally to compute the rigid transformation that links the target plane and the camera.

Developing this calibration process was rewarding in itself, but details have to be omitted due to lack of space.

5.3. Results

Reconstruction results for a patch of the specular spoon are shown in figures 9 and 10. What these images do not clearly show but can be noticed in the 3D model is that there are no holes greater than 1-2 voxels on the patch, and that the extracted thickness is around 1-3 voxels. The extracted voxels are qualitatively correct, and quantitatively correct up to what we could measure.

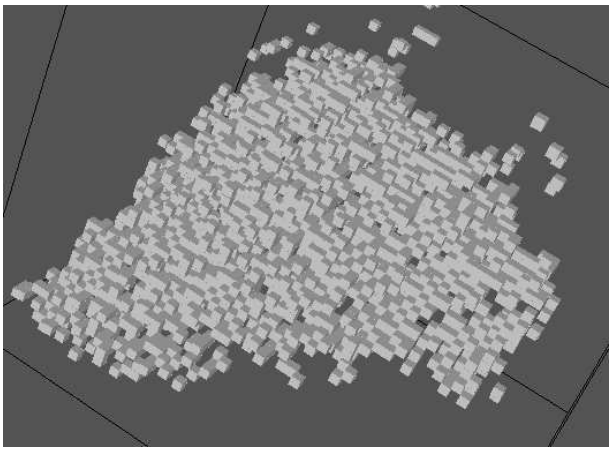


Figure 9: Front view of the reconstructed voxels.

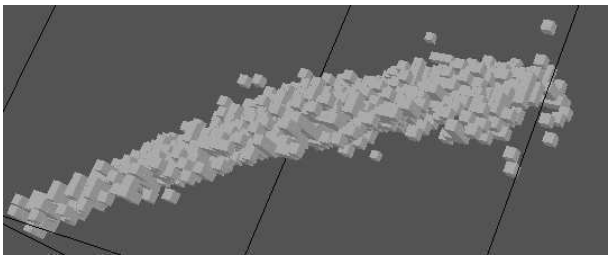


Figure 10: Side view of the reconstructed voxels. The spoon's curvature is clearly visible in one dimension. Views from other directions show curvature is correct on all the reconstructed surface.

6. Conclusion and Future Work

We have developed a method reconstructing voxels and normals of a specular surface, requiring multiple views of a calibrated camera and the matching of calibrated scene points with their reflection in the specular surface. We have tested this method on simulated data and obtained qualitatively correct results, and on real images and obtained correct results as far as we could measure.

The underlying foundation is based on the consistency of surface normals whatever the viewpoint. A reprojection error measure and several heuristic disparity measures have been tested to extract surface voxels, one of the latter giving the best results. The principal advantage we see of the method is that it makes no assumption on the surface, and does not need any initial seed points to complete. It is therefore not subject to error accumulation that can occur when assumptions on surface continuity or derivability are made.

An interesting challenge still remains the information available for specular surfaces from uncalibrated views and/or an uncalibrated reflected scene.

References

[1] A. Blake and G. Brelstaff, Geometry from Specularities, *ICCV*, p 394–403, 1988

[2] Y. Caspi and M. Irani, Alignment of Non-Overlapping Sequences, *I*CCV, pp. 76-83, 2001

[3] M. Halstead, B. Barsky, S. Klein and R. Mandell, Reconstructing Curved Surfaces from Specular Reflection Patterns using Spline Surface Fitting of Normals, *SIGGRAPH*, 1996

[4] C.G. Harris and M. Stephens, A Combined Corner and Edge Detector, *4th Alvey Vision Conference*, p 147–151, 1988

[5] G. Healey and T. Binford, Local Shape from Specularities, *Computer Vision, Graphics, and Image Processing* **42**, p 62–86, 1988

[6] P.M.Griffin, S. Narasimhan and S.R. Yee, Generation of Uniquely Encoded Light Patterns for Range Data Acquisition, *Pattern Recognition* **6**,p 609–616, 1992

[7] K. N. Kutulakos and S. M. Seitz, A Theory of Shape by Space Carving, International Journal of Computer Vision, 38(3), pp 199-218, 2000

[8] M. Oren and S. Nayar, A Theory of Specular Surface Geometry, *International Conference on Computer Vision*, p 740- , 1995

[9] A. Ripsman and M. Jenkin, Local Surface Reconstruction of Objects in Space, *IEEE Int. Symposium on Computational Intelligence*, 2001

[10] S. Savarese and P. Perona, Local Analysis for 3D Reconstruction of Specular Surfaces – Part II, *European Conf. on Computer Vision*, 2002

[11] H. Schultz, Shape Reconstruction from Multiple Images of the Ocean Surface, *Photogrammetric Engineering and Remote Sensing*, vol. 62, no. 1, 1996

[12] S. M. Seitz and C. R. Dyer, Photorealistic Scene Reconstruction by Voxel Coloring *International Journal of Computer Vision*, 35(2), pp 151-173, 1999

[13] P. Sturm and S. Maybank, On Plane-Based Camera Calibration: A General Algorithm, Singularities, Applications, *C*VPR, pp. 432-437, 1999

[14] P. Sturm, Algorithms for Plane-Based Pose Estimation, *C*VPR, p.. 1010-1017, 2000

[15] Z. Zhang, A Flexible New Technique for Camera Calibration, *I*EEE PAMI, Vol. 22, No. 11, pp. 1330-1334, 2000

[16] J. Zheng and A. Murata, Acquiring a Complete 3D Model from Specular Motion under the Illumination of Circular-Shaped Light Sources, *IEEE Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, 2000

[17] A. Zisserman, P.Giblin, and A. Blake, The Information Available to a Moving Observer from Specularities, *Image and Video Computing* **7**, p 38–42, 1989

A. Results using Synthetic Images

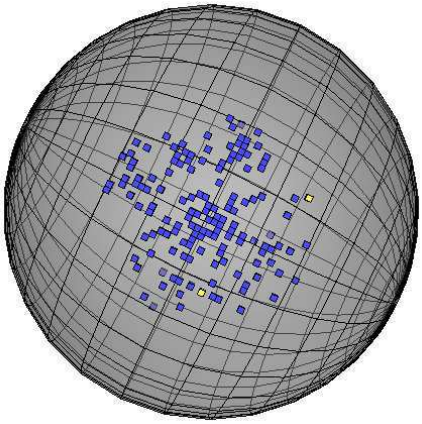


Figure 11: **synthetic reconstruction.** 161 voxels extracted on the sphere's surface using a strict tolerance on normal disparity. 158 dark (blue) coloured voxels intersect the specular surface, while 3 light (yellow) coloured ones are adjacent to an intersecting voxel.No further voxels have been extracted.

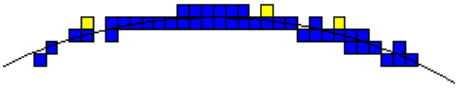


Figure 12: **synthetic reconstruction.** Cut through a reconstruction of 973 voxels. Voxels on the edges are nearly as close to the surface as the ones near the centre.

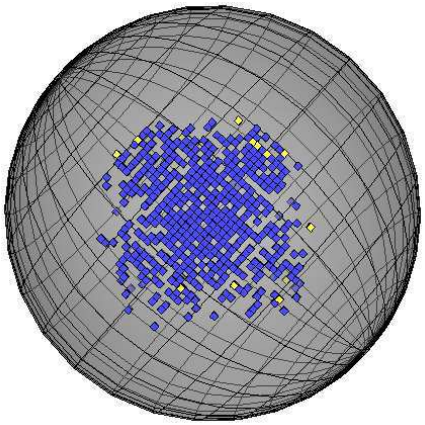


Figure 13: **synthetic reconstruction.** 589 voxels extracted on the sphere's surface, using a medium tolerance on normal disparity. 573 dark (blue) coloured voxels intersect the specular surface, while 16 light (yellow) coloured ones are adjacent to an intersecting voxel.No further voxels have been extracted.

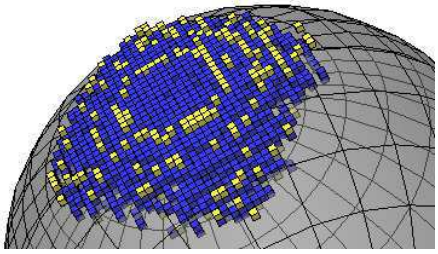


Figure 14: **synthetic reconstruction.** 1982 voxels extracted on the sphere's surface, using a broad tolerance on normal disparity. 1659 dark (blue) coloured voxels intersect the specular surface, while 323 light (yellow) coloured ones are adjacent to an intersecting voxel. Nearly all intersecting voxels were extracted, while no voxels that were further to the surface have appeared.

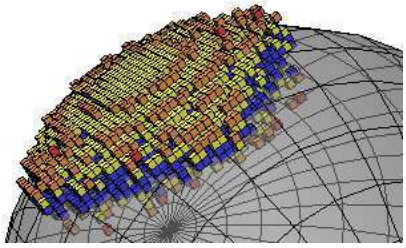


Figure 15: **synthetic reconstruction.** 4162 extracted voxels, using a very laxist threshold on normal disparity. 1934 dark (blue) coloured voxels intersect the specular surface, while 1765 light coloured ones are adjacent to an intersecting voxel. An additional 463 voxels (in orange) were more distant, however never being further than 5 voxels to the correct surface.

General specular Surface Triangulation

Thomas Bonfort, Peter Sturm, and Pau Gargallo

MOVI - GRAVIR - INRIA - 38330 Montbonnot, FRANCE
http://perception.inrialpes.fr

Abstract. We present a method for the reconstruction of a specular surface, using a single camera viewpoint and the reflection of a planar target placed at two different positions. Contrarily to most specular surface reconstruction algorithms, our method makes no assumption on the regularity or continuity of the specular surface, and outputs a set of 3D points along with corresponding surface normals, all independent from one another. A point on the specular surface can be reconstructed if its corresponding pixel in the image has been matched to its source in both of the target planes. We present original solutions to the problem of dense point matching and planar target pose estimation, along with reconstruction results in real-world scenarii.

1 Introduction

Reconstructing surfaces from images usually relies on the identification and matching of pixels corresponding to a same 3D point on the surface. On unpolished surfaces, matching can be fulfilled by analyzing surface texture, and assuming that identical texture patches correspond to identical points on the surface. In the case of specular surfaces, the apparent surface texture is the reflection of the object's surroundings, being *de facto* viewpoint-dependent, thus invalidating the geometric constraints used by all non-specific reconstruction algorithms. Even standard laser scanners are unable to acquire specular surfaces, as all of the laser energy is reflected symmetrically to the normal of the surface, and therefore cannot be detected by the sensor [1]. Consequently, specularities, and even more importantly specular objects, are usually discarded as noise by most surface reconstruction algorithms. However, specular reflections give rise to strong constraints on surface depth and orientation, and we take advantage of these additional cues to reconstruct a precise model of the surface.

We describe a method recovering points of a specular surface, independently from one another. We assume an internally calibrated pinhole camera viewing the reflection of a planar target, and a dense matching of the camera pixels with the points on the target. While the camera is rigidly attached to the specular surface, we acquire images of the reflection of the target placed at two different unknown locations. The foundation of our method is closely related to the work on general (*i.e.* non central) cameras, as the reconstruction of the specular surface from the images of a calibrated camera is equivalent to the calibration of a non-central catadioptric system. The output of the algorithm is a collection of 3D points of the specular surface, and the two transformations (rigid displacements) from the camera reference coordinate system to the target plane coordinate systems.

1.1 Previous Work

Though less actively than for lambertian surfaces, the reconstruction of specular surfaces from images has interested researchers in the field of computer vision for the past 20 years. For example, Blake and Brelstaff [2] study the disparity of highlights on a specular surface in a stereoscopic framework. Zisserman *et al.* [3] tracked the motion of specularities obtaining a degree-1 family of curvatures along the tracked path.

In [4], Oren and Nayar study the classification of real and reflected features, and recover the profile of a specular surface by tracking an unknown scene point. The work was extended to complete object models by Zheng and Murata in [5], who reconstruct a rotating specular object by studying the motion of the illumination created by two circular light sources.

Halstead *et al.*, in [6], fit a spline surface to a set of normals, iteratively refining the result. Their method requires an initial seed point on the specular surface, and was applied to the sub-micronic reconstruction of the human cornea. The approach was extended by Tarini *et al.* [7] who integrate around a seed point, and use a global self-coherence measure to estimate the correct depth for the seed point. Under a distant light configuration, Solem *et al.* [8] fit a level-set surface with a variational approach.

Savarese *et al.* detail in [9] the mathematical derivations allowing the recovery of surface parameters up to 3rd order from one view of a smooth specular object reflecting two intersecting calibrated lines, when scale and orientation can be measured in the images.

Bonfort and Sturm [10] present a space carving approach using surface normals instead of color as a consistency measure.

1.2 Notation

The following notation will be used throughout the article: **bold** letters represent a vector in 3D space, while *italic* letters represent scalars. Matrices are represented by CAPITAL letters.

2 Approach

Suppose a calibrated pinhole camera located at $\mathbf{O_c} = \mathbf{0}^\mathsf{T}$ observing the reflection in an unknown specular surface of a known 3D feature $\mathbf{Q}$. As the camera is calibrated, recovering the position of the surface at the point $\mathbf{p}$ of reflection is simply the estimation of its depth along the corresponding projection ray. This already constrained scenario is still insufficient in order to obtain a solution to the depth estimation, as for every point $\mathbf{P}$ along the projection ray, we can compute a surface orientation that would produce an identical observation: depth estimation of a point on a specular surface from one image gives rise to a one dimensional solution, function of surface depth and orientation.

Now consider the same setup, except that for a given camera pixel $\mathbf{p}$, two 3D point correspondences $\mathbf{Q_1}$ and $\mathbf{Q_2}$ are given. This constraint is sufficient to uniquely determine the depth of the specular surface at $\mathbf{p}$, namely as the intersection of the lines formed by the camera's projection center and $\mathbf{p}$ on the one hand, and $\mathbf{Q_1}$ and $\mathbf{Q_2}$ on the other.

If we consider the (camera + specular surface) system as a general camera, finding two points $\mathbf{Q_1}$ and $\mathbf{Q_2}$ for each $\mathbf{p}$, and therefore obtaining a reconstruction of the surface, is equivalent to calibrating this camera, as this is usually done as a one-to-one mapping of image pixels with lines in 3D space. In [11] or [12], such a calibration is achieved by using points on calibration planes: pixels in the image are matched with their 2D correspondent in the target planes, then the only step necessary in order to obtain 3D coordinates of these points is to estimate the pose of the planes in the camera reference coordinate system. Figure 1 summarizes our reconstruction method for 3 point correspondences: reconstructing the specular surface sums down to matching camera pixels with their source in the target planes, then estimating the two transformation matrices T_1 and T_2, that map points from the target reference coordinate system to the camera one.

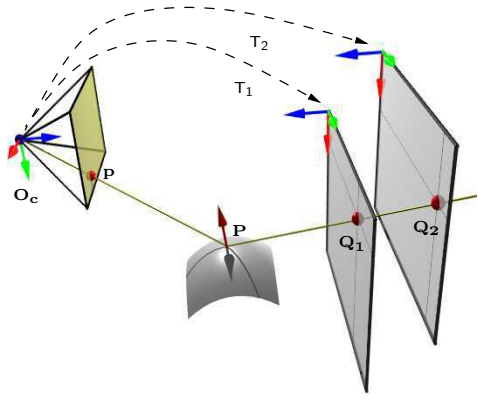


Fig. 1: **Reconstruction Approach.** Matching of image pixels with their source in the targets and estimating two plane poses is sufficient to reconstruct the surface.

3 Dense Matching

The 3D position of a point on the specular surface corresponding to a given pixel in the camera image plane can only be computed if a correspondence can be found in both of the target planes. As such, in order to obtain a dense reconstruction of the specular surface, each pixel of the specular surface must be matched to its target correspondence.

3.1 Initial Matching

We use a standard computer monitor displaying Gray codes, once original and once inverted [13]. The total number of images taken for each pose of the target is therefore twice the binary resolution in each direction.

The resolution of the codes and the width of the low order stripes must be chosen according to the shape of the specular object and the resolution of the camera. Too high resolution codes tend to be blurred out and become unusable, whereas too coarse ones lack in precision. In most cases, multiple pixels in the camera image correspond to the same code in the target planes. Figure 3 (top right) shows the result of a reconstruction if we apply this initial matching directly.

3.2 Sub-pixel Matching

From the Gray code decoding we get an initial integer-valued estimate of the pixel matching. To get more accurate correspondences, this initialization has to be refined. Let $u(x, y)$ and $v(x, y)$ denote the coordinates of the target point corresponding to the camera pixel (x, y). Instead of directly smoothing u and v as in [13], we use an energy minimization approach to ensure that the smoothed correspondences will still link camera pixels with their corresponding origin on the target planes.

We minimize the following energy functional with respect to u and v:

$$E(u,v) = \sum_k \int_\Omega (\mathcal{G}_k(u,v) - \mathcal{I}_k(x,y))^2 \, dx\, dy + \lambda \int_\Omega |\nabla u|^2 + |\nabla v|^2 \, dx\, dy$$

where Ω is the mirror image region, $\mathcal{G}_k$ are the Gray code images and $\mathcal{I}_k$ are the images captured by the camera.

The first energy term is the data term. It penalize correspondences for which the color $\mathcal{I}_k(x, y)$ captured by the camera and it's corresponding Gray code $\mathcal{G}_k(u, v)$ are not the same. We first scale the camera images intensities pixel-wise, so that 0 and 1 intensities correspond to pure black and pure white. This referential is computed by displaying entirely black and entirely white images on the planar targets. For non-integer values of u and v, $\mathcal{G}_k(u, v)$ is computed using bilinear interpolation.

The second term is a homogeneous regularizer. It penalizes large variations on the correspondence functions. The λ parameter sets the compromise between data evidence and smoothing.

The energy functional is minimized by a steepest descent. The descent direction is given by the Euler-Lagrange equations,

$$\frac{\partial u_i}{\partial t} = -\sum_k 2(\mathcal{G}_k - \mathcal{I}_k)\frac{\partial \mathcal{G}_k}{\partial u_i} + \lambda\, 2\Delta u_i$$

for $u_1 = u$ and $u_2 = v$.

Figure 3 (bottom right) shows the result of the reconstruction after having smoothed the orginal matches.

4 Target Pose Estimation

Our reconstruction algorithm requires knowledge of the relative pose between the camera and target plane, in its different positions.

The first and simplest method is to ensure that the target plane is partially visible in the camera, as seen in figure 3, and apply any pose estimation method [14]; we use the method proposed in [15].

To ensure a much higher flexibility, we wanted to be able to work with setups where the camera hasn't any direct view of the target plane; if this was possible then one would be able to take "better" images of the

specular surface to be reconstructed. The second solution is to estimate the pose of the targets through the reflection by a known mirror. We therefore suppose having a means of estimating the pose of the planar mirror: this can either be done by placing markers on the mirror and performing a classical plane pose estimation, or in our case by using a hard-drive platter, whose known interior and exterior radii allow an ellipse based pose to be estimated. More details on the reflection by a known plane can be found in the next paragraph.

4.1 Pose Through Reflection by 3 Unknown Planes

We acquire images by holding a planar mirror in front of the camera in different positions, such that the target plane's reflection is seen by the camera. We now briefly describe how to solve the relative pose between camera and target plane, from three or more such images, or one image of three or more such mirrors.

In the following, we adopt a global reference frame such that the target plane is at $Z = 0$, and first carry out a pose estimation for each image, as if the image were a direct view of the target plane.

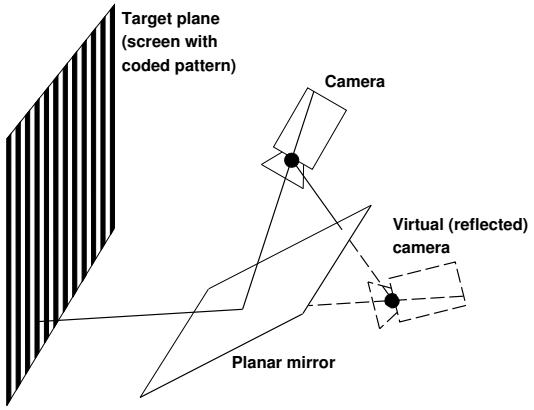


Fig. 2: **Reflected pose.** The estimated pose of a reflected plane is equivalent to its pose viewed from a virtual reflected camera.

This procedure gives us the pose of the virtual camera that would be produced by reflecting the real camera in the planar mirror, cf. figure 2. If we knew the pose of the planar mirror, we could of course immediately recover the camera's true pose, as follows. Let the recovered pose of the virtual camera for image i be given via the projection matrix:

$$\mathsf{P}_i^v \sim \mathsf{S}_i \left(\mathtt{I} | - \mathbf{t}_i \right)$$

where S_i is a reflection matrix (a rotation matrix multiplied by -1), and let the associated pose of the planar mirror be represented by homogeneous coordinates

$$\mathit{\Pi}_i \sim \begin{pmatrix} \mathbf{n}_i \\ d_i \end{pmatrix}$$

where we distinguish the plane's normal vector $\mathbf{n}_i$ (of unit norm), and its distance d_i from the origin. The true camera's pose can be recovered by multiplying P_i^v with the transformation modeling the reflection in the plane $\mathit{\Pi}_i$:

$$\begin{aligned} \mathsf{P}_i &\sim \mathsf{P}_i^v \begin{pmatrix} \mathtt{I} - 2\mathbf{n}_i\mathbf{n}_i^\mathsf{T} & -2d_i\mathbf{n}_i \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \\ &\sim \mathsf{S}_i \left(\mathtt{I} - 2\mathbf{n}_i\mathbf{n}_i^\mathsf{T} | - \mathbf{t}_i - 2d_i\mathbf{n}_i \right) \end{aligned} \tag{1}$$

We now have to address the question how to recover the true camera's pose, knowing that with the correct mirror positions $\mathit{\Pi}_i$, the camera poses P_i computed according to (1), have to be equal to one another: $\mathsf{P}_i \sim \mathsf{P}_j$. Due to $\det\left(\mathtt{I} - 2\mathbf{n}_i\mathbf{n}_i^\mathsf{T}\right) = \det \mathsf{S}_i = -1$, we can safely eliminate the scale ambiguity in the equation $\mathsf{P}_i \sim \mathsf{P}_j$, and obtain element-wise equalities:

$$\forall i, j : \mathsf{S}_i \left(\mathtt{I} - 2\mathbf{n}_i\mathbf{n}_i^\mathsf{T} \right) = \mathsf{S}_j \left(\mathtt{I} - 2\mathbf{n}_j\mathbf{n}_j^\mathsf{T} \right) \tag{2}$$

$$\forall i, j : \mathsf{S}_i \left(\mathbf{t}_i + 2d_i\mathbf{n}_i \right) = \mathsf{S}_j \left(\mathbf{t}_j + 2d_j\mathbf{n}_j \right) \tag{3}$$

Computing mirror plane normals $\mathbf{n_i}$. Let $\mathsf{X}_i = \mathtt{I} - 2\mathbf{n}_i\mathbf{n}_i^\mathsf{T}$, which is of course a symmetric matrix. From (2), we get:

$$\mathsf{X}_i = \underbrace{\mathsf{S}_i^\mathsf{T}\mathsf{S}_j}_{\mathsf{R}_{ij}}\mathsf{X}_j \tag{4}$$

Furthermore, X_j is a reflection, *i.e.* $\mathsf{X}_j\mathsf{X}_j = I$, therefore:

$$\mathsf{R}_{ij} = \mathsf{X}_i\mathsf{X}_j \tag{5}$$

Let $\mathbf{a}_{ij}$ be a vector orthogonal to $\mathbf{n}_i$ and $\mathbf{n}_j$. We therefore have:

$$\begin{aligned}\mathsf{R}_{ij}\mathbf{a}_{ij} &= \mathsf{X}_i\mathsf{X}_j\mathbf{a}_{ij}\\ &= \left(\mathtt{I} - 2\mathbf{n}_i\mathbf{n}_i^\mathsf{T}\right)\left(\mathtt{I} - 2\mathbf{n}_j\mathbf{n}_j^\mathsf{T}\right)\mathbf{a}_{ij}\\ &= \left(\mathtt{I} - 2\mathbf{n}_i\mathbf{n}_i^\mathsf{T}\right)\mathbf{a}_{ij}\\ &= \mathbf{a}_{ij}\end{aligned}$$

which implies that $\mathbf{a}_{ij}$ is the eigenvector to the eigenvalue 1 of R_{ij}, *i.e.* that $\mathbf{a}_{ij}$ is the rotation axis of R_{ij}.

We now have the means to compute all mirror normals $\mathbf{n}_i$, provided at least 3 mirrors are used.

1. Compute the pose eq. (1) of all virtual cameras, as described above.
2. For all pairs of mirrors (i, j), compute R_{ij}, as per eq. (4). Compute their eigenvectors to the eigenvalue $+1$, i.e. vectors $\mathbf{a}_{ij}$.
3. For every mirror i, stack all $\mathbf{a}_{ij}^\mathsf{T}$ (respectively $\mathbf{a}_{ki}^\mathsf{T}$) in a matrix A of size $(n-1)\times 3$ (where n is the number of mirrors), and compute $\mathbf{n}_i$ as the unit eigenvector to the smallest eigenvalue of $\mathsf{A}^\mathsf{T}\mathsf{A}$.

Computing the true camera's pose. The last step is to compute the least squares solution for the d_i of the linear equation system composed of one equation (3) per pair of mirrors. The system's design matrix is of size $3n(n-1)\times n$ and very sparse.

We now know all mirror planes Π_i, and can compute the camera pose from any one of them, according to eq. (1). In practice, we do this computation for every mirror, and then "average" the resulting rotation matrices and position vectors that represent camera pose. We then apply a bundle adjustment style procedure for simultaneously optimizing the pose of the camera and the planar mirrors. The cost function minimized here is the reprojection error of target points, projected in the camera after reflection in the mirrors.

5 Optimization

In practice, we also perform a global non-linear optimization of the poses T_1 and T_2 of the target planes, before the triangulation. The cost function to be minimized is the distance between matching lines in 3D space which we minimize using a Levenberg Marquardt algorithm.

$$cost(\mathsf{T}_1, \mathsf{T}_2) = \sum_{i\in\{matches\}} dist^2((\mathbf{O_c}, \mathbf{p_i}), (\mathsf{T}_1\mathbf{Q_{1i}}, \mathsf{T}_2\mathbf{Q_{2i}}))$$

6 Results

We tested our reconstruction method on real specular surfaces, using the different pose estimation methods presented in section 4. As seen on figure 3, no continuity or regularity is assumed.

Having no ground truth results, we evaluated the correctness of the method by fitting a plane to the part of the reconstruction we knew was planar, *i.e.* the hard drive platter (linear least squares fitting, without outlier removal). In the reconstruction shown on figure 3, over 98% of the computed points were less than 0.2 mm away from the surface, and 64% less than 0.1 mm. The approximate diameter of the reconstructed part of the platter was 80 mm, resulting in a maximum 0.3% relative error in the reconstruction.

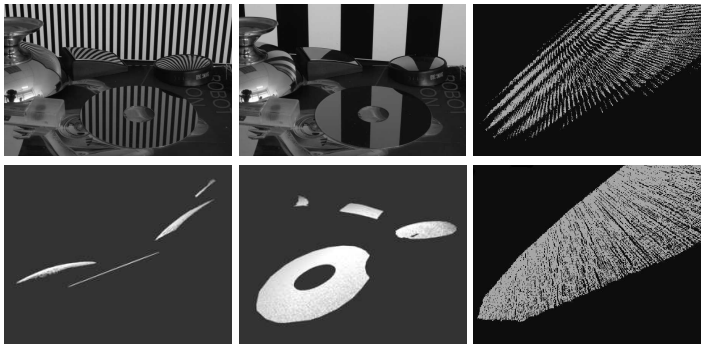


Fig. 3: **Validation Setup and Results.** The top row shows two of the images used for the reconstruction. Notice the 3 curved mirrors (an ice-cream cup and two small wide-angle rear-view mirrors, the planar hard drive platter, and a direct view of the target plane, in the upper part of the image. The second row shows the reconstruction viewed from two locations. The model contains over 525 000 independent points. Note the planarity of the reconstructed hard drive platter in the left image. Only a few points could be computed on the ice-cream cup, as its surface covered by the exploitable Gray codes was limited. The two small rear-view mirrors (one with circular, the other with rectangular based shape) were completely reconstructed (apart from a non specular dent in the circular one). The two images on the right show the effect of the sub-pixel matching and constrained smoothing: top image shows result using raw gray codes, while the bottom one shows results after the smoothing step.

The accuracy of the reconstruction also depends on the quality of the pixel matching. Indeed, when experimenting with purely piecewise planar surfaces, where the sub-pixel matching was "easy" to compute, the distances to the fitted planes dropped down to 99.9% of the computed points less than 0.1 mm away from the surface, and 88% less than 0.05 mm. This is because the average quality of the matches is higher compared to when the scene also contains curved specular surfaces. Hence the pose of the target planes and finally the reconstruction are more precise.

We tested the reconstruction on another setup composed only of planes with the different pose estimation techniques presented in section 4. Although the initial estimation of the poses given by the different techniques are not exactly identical, the non-linear optimization converged to very similar poses in all cases. The histogram of the point-plane distance, with the poses estimated with the three unknown planes (section 4.1), without global optimization, can be seen in figure 4.

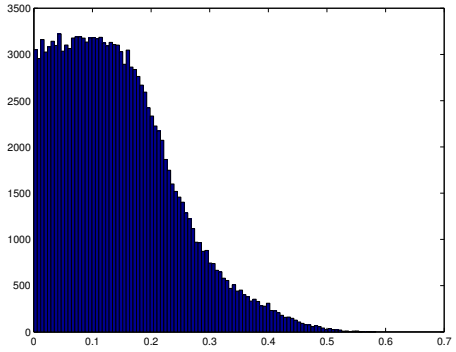


Fig. 4: **Point-plane distance.** Histogram of the distance in of each point to the linear least squares fitted plane (in millimeters) with the poses estimated with the three unknown mirror planes (section 4.1).

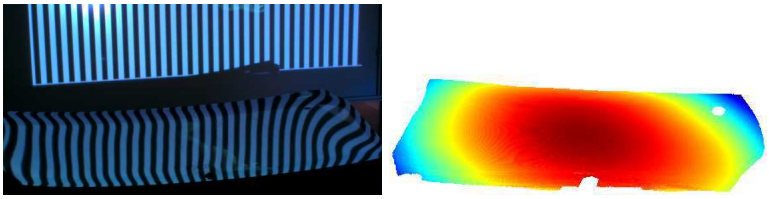


Fig. 5: **Real World Reconstruction.** Reconstruction of a car windshield. The method allowed us to easily obtain a 800 000 + point model using a classical video projector, on a large scale reflective surface. The hole in the middle is due to a non-specular patch on the surface.

7 Conclusion

We have presented a novel method that reconstructs a specular surface from two views. Compared to other reconstruction methods, we attain a high level of accuracy, without having the need to suppose surface continuity or regularity. We believe it could easily be implemented in an industrial surface inspection application, at least to provide an accurate initialization for integration based reconstruction methods, probably the only purely vision based ones able to detect surface micro-structure. We also proposed a novel method for the pose estimation of a target plane even if it is never directly seen in the images, requiring the view of its reflection through unknown planar mirrors.

The drawback of the method is the need to obtain a dense matching over the *complete* surface we want reconstructed. This in practice is difficult to obtain with only two positions of the target plane, meaning multiple reconstructions have to be computed then stitched together.

References

1. Chen, F., Brown, G.M., Song, M.: Overview of three-dimensional shape measurement using optical methods. Optical Engineering **39** (2000)
2. Blake, A., Brelstaff, G.: Geometry from specularities. In: Second International Conference on Computer Vision (Tampa,, FL), Washington, DC,, Computer Society Press (1988) 394–403
3. Zisserman, A., Giblin, P., Blake, A.: The information available to a moving observer from specularities. Image and Vision Computing **7** (1989) 38–42
4. Oren, M., Nayar, S.K.: A theory of specular surface geometry. In: International Conference on Computer Vision. (1995) 740–747
5. Zheng, J.Y., Murata, A.: Acquiring 3D object models from specular motion using circular lights illumination. In: Procedings of the Sixth International Conference on Computer Vision (ICCV-98). (1998) 1101–1108
6. Halstead, M., Barsky, B., Klein, S., Mandell, R.: Reconstructing curved surfaces from specular reflection patterns using spline surface fitting of normals. In: SIGGRAPH 96 Conference Proceedings. (1996) 335–342
7. Tarini, M., Lensch, H., Goesele, M., Seidel, H.: 3D acquisition of mirroring objects. In: Research Report MPI-I-2003-4-001, Max-Planck-Institut fr Informatik (2003)
8. Solem, J.E., Aanæs, H., Heyden, A.: A variational analysis of shape from specularities using sparse data. In: 3DPVT, IEEE Computer Society (2004) 26–33
9. Savarese, S., Chen, M., Perona, P.: Recovering local shape of a mirror surface from reflection of a regular grid. In: European Conference on Computer Vision. (2004)
10. Bonfort, T., Sturm, P.: Voxel carving for specular surfaces. In: International Conference on Computer Vision. (2003) 591–596
11. Grossberg, M., Nayar, S.: A general imaging model and a method for finding its parameters. In: International Conference on Computer Vision. (2001) 108–115
12. Sturm, P., Ramalingam, S.: A generic concept for camera calibration. In: Proceedings of the European Conference on Computer Vision. Volume 2., Springer (2004) 1–13
13. Scharstein, D., Szeliski, R.: High-accuracy stereo depth maps using structured light. In: CVPR (1), IEEE Computer Society (2003) 195–202
14. Haralick, R., Lee, C., Ottenberg, K., Nolle, M.: Review and analysis of solutions of the three point perspective pose estimation problem. IJCV **13** (1994) 331–356
15. Sturm, P.: Algorithms for plane-based pose estimation. In: Proceedings of the Conference on Computer Vision and Pattern Recognition, Hilton Head Island, South Carolina, USA. (2000) 1010–1017

How to Compute the Pose of an Object without a Direct View?

Peter Sturm and Thomas Bonfort

INRIA Rhône-Alpes, 38330 Montbonnot St Martin, France
{Peter.Sturm, Thomas.Bonfort}@inrialpes.fr

Abstract. We consider the task of computing the pose of an object relative to a camera, for the case where the camera has no direct view of the object. This problem was encountered in work on vision-based inspection of specular or shiny surfaces, that is often based on analyzing images of calibration grids or other objects, reflected in such a surface. A natural setup consists thus of a camera and a calibration grid, put side-by-side, i.e. without the camera having a direct view of the grid. A straightforward idea for computing the pose is to place planar mirrors such that the camera sees the calibration grid's reflection. In this paper, we consider this idea, describe geometrical properties of the setup and propose a practical algorithm for the pose computation.

1 Introduction

Consider a calibration grid or any other known object, planar or not, and a camera. We would like to determine their relative pose, but for the case where **the camera does not see the object directly**. This is an unusual setting, but it is quite natural for the task of reconstructing specular or shiny surfaces, as explained in the following. Modeling of specular or shiny surfaces is an important application in inspection of industrial parts, especially in the car manufacturing industry (control of wind shields and bodywork) but also in the control of optical lenses or mirrors, glasses of watches etc. Vision-based reconstruction of specular surfaces is usually based on acquiring images of known patterns or light sources, reflected in the surface to be reconstructed [3, 4, 6, 7, 11, 14].

It is thus rather natural to place the camera and pattern such that the camera does not have a direct view of the latter, or at most sees a small part of it. We have proposed practical approaches for the reconstruction of specular surfaces where such an arrangement is indeed used. The question of how to compute the pose of an object without a direct view is thus important for us and in addition scientifically appealing.

Our initial solution consisted in attaching the pattern rigidly to the camera and to move the two to a few locations. During this, the camera acquired images of a calibration grid, and a secondary camera (static) acquired images of our pattern. With this input, the pattern's 3D trajectory was computed as well as the main camera's one. By registering the two trajectories into a common coordinate frame, along the lines of [2] and of the classical hand-eye calibration problem, we finally computed the pose of the pattern relative to the main camera. This approach was found to be too cumbersome in practice. A second camera is required and especially, having to move the camera–pattern pair is not desirable, as we currently use an LCD monitor to produce the pattern(s).

We are thus aiming at a lighter procedure. A natural idea is to proceed as follows: place a planar mirror in different positions in front of the camera such that the pattern's reflection is seen, and acquire images. The question arises if this input is sufficient to solve our pose problem, and if yes, how many positions of the planar mirror are required? We show in this paper that our pose problem can be solved up to 1 degree of freedom from two positions, and can be fully solved from three or more positions.

2 Background

2.1 Camera model

We consider perspective projection as camera model. The projection of 3D points is modeled by a 3×4 projection matrix $\mathsf{P} = \mathsf{K}\mathsf{R}\left(\mathtt{I}|-\mathbf{t}\right)$, where K is the usual 3×3 calibration matrix with the camera's intrinsic parameters, and the orthogonal matrix R and the vector $\mathbf{t}$ represent camera orientation and position. For simplicity, we assume that the camera is calibrated, i.e. that K is known (this will be relaxed later). We thus directly work with geometric image coordinates, i.e. consider that 3D points $\mathbf{Q}$ are projected to image points $\mathbf{q}$ via the canonical projection matrix $\mathbf{q} \sim \mathsf{R}\left(\mathtt{I}|-\mathbf{t}\right)\mathbf{Q}$. 2D and 3D points are expressed in homogeneous coordinates and $\sim$ means equality of vectors or matrices, up to scale.

2.2 Pose computation

A classical task of photogrammetry and computer vision is to compute the pose of a calibrated camera, relative to an object of known structure. In this work, we use planar reference objects. There exist many algorithms for the planar pose problem; we use [10].

2.3 Reflections in planes

Consider a plane $\Pi = (\mathbf{n}^\mathsf{T}, d)^\mathsf{T}$ in 3-space, i.e. consisting of points satisfying the equation $n_1X+n_2Y+n_3Z+d = 0$. In the following, we will always suppose that the plane's normal vector is of unit norm. The reflection in Π can be represented by the following transformation matrix:

$$\mathsf{S} = \begin{pmatrix} \mathtt{I} - 2\mathbf{n}\mathbf{n}^\mathsf{T} & -2d\mathbf{n} \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix}$$

Let us denote the upper left 3×3 matrix of S by $\bar{\mathsf{S}}$. It is an orthogonal matrix, with determinant -1 (whereas a rotation matrix has determinant $+1$). Further, it has $+1$ as double eigenvalue and -1 as single eigenvalue. The plane normal $\mathbf{n}$ is an eigenvector of $\bar{\mathsf{S}}$ to the eigenvalue -1. Note also that $\mathsf{S}^{-1} = \mathsf{S}$.

2.4 Planar motion and fixed-axis rotation

Planar motion usually means a translation in some direction, followed by a rotation about an axis that is orthogonal to the translation direction. Such a motion can always be expressed as just a rotation about an axis that is parallel to the above rotation axis; we thus prefer to call such motions *fixed-axis rotations*. It is easy to show that any euclidian transformation that preserves some line point-by-point, is a fixed-axis rotation, whose axis is that line.

Let the axis be represented by its direction vector $\mathbf{D}$ and a footpoint $\mathbf{A}$ such that $\mathbf{A} + \lambda\mathbf{D}$ represents the points (in non-homogeneous coordinates) on the axis. Any finite point on the axis can serve as footpoint; we always choose the one that is "orthogonal" to $\mathbf{D}$: $\mathbf{A}^\mathsf{T}\mathbf{D} = 0$. This is the point on the axis that is closest to the origin.

Let α be the angle of rotation and R be the rotation matrix representing rotation by α about $\mathbf{D}$. Then, the 4×4 matrix representing the complete fixed-axis rotation, is:

$$\mathsf{T} = \begin{pmatrix} \mathtt{I} & \mathbf{A} \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \begin{pmatrix} \mathsf{R} & \mathbf{0} \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \begin{pmatrix} \mathtt{I} & -\mathbf{A} \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} = \begin{pmatrix} \mathsf{R} & \mathbf{A} - \mathsf{R}\mathbf{A} \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix}$$

2.5 Reflection in two planes

Consider successive reflections in two planes. It can be shown that this is a fixed-axis rotation, with the intersection line of the two planes as rotation axis: the transformation preserves the intersection line of the two planes point-by-point, and thus is a fixed-axis rotation.

Further, the rotation angle is twice the angle between the two planes. This is also easy to see: let the transformation be the sequence $\mathsf{S}_2\mathsf{S}_1$ of reflections in two planes. Let us apply this transformation to the point at infinity $(\mathbf{n}_1^\mathsf{T}, 0)^\mathsf{T}$, i.e. the normal direction of the first plane. This is a fixed point of S_1, hence the transformation gives the point's reflection in S_2. The angle between the original point at infinity, and the transformed one, i.e. the fixed-axis rotation angle, is thus twice the angle between the original point at infinity and the second reflection plane. Hence, as said above, the sequence of reflections in two planes is a fixed-axis rotation, whose angle is twice the angle between the planes.

2.6 Horopter

The horopter of a stereo system is the set of 3D points that project to points with identical coordinates in the two cameras. Let P_1 and P_2 be the two cameras' projection matrices. The horopter thus consists of all 3D points $\mathbf{Q}$ with $\mathsf{P}_1\mathbf{Q} \sim \mathsf{P}_2\mathbf{Q}$. This is in general a quartic curve. If the two cameras have identical calibration and are separated by a fixed-axis rotation, then the horopter "degenerates" into the union of a straight line and a circle: the motion's rotation axis and the circle in the motion plane that contains the two optical centers and that cuts the rotation axis [5].

3 Outline of the Proposed Approach

We consider a camera and an object in fixed position, put a planar mirror in the scene in n different positions, and take an image for each of those. We suppose that the camera sees the object's reflection in each image. We further suppose that the object's structure is known and that correspondences between object and image points can be obtained.

In the first step of our approach, the views of the reflected object are treated as if they were direct views. We may thus compute a camera pose, from the camera's calibration and the given point correspondences. This will actually give the pose of a "virtual" camera that is the reflection of the true camera, in the planar mirror (cf. figure 1). Overall, we thus get the pose of n virtual cameras, relative to the object.

In the second step, we try to infer the positions of the planar mirrors. The underlying constraint is that reflecting the virtual cameras in mirrors with the correct positions, will lead to n identical cameras – the true one. We show how the mirror positions can be computed using the above notions of horopter and fixed-axis rotation. We further show that for $n = 2$, the problem can be solved up to 1 degree of freedom, and that with $n > 2$ a unique solution can be found. These steps are described in the following sections.

4 Computing Pose of Virtual Cameras

In the following, we adopt the object's coordinate system as our reference system, in which the pose of true and virtual cameras will be expressed. Let the pose of the true camera be represented by the projection matrix

$$\mathsf{R}\left(\mathtt{I}|-\mathbf{t}\right)$$

Consider now a planar mirror, defined by the plane

$$\mathit{\Pi} = \begin{pmatrix}\mathbf{n}\\ d\end{pmatrix}$$

Object points $\mathbf{Q}$ are projected into the true camera as follows:

$$\mathbf{q} \sim \mathsf{R}\left(\mathtt{I}|-\mathbf{t}\right)\begin{pmatrix}\mathtt{I}-2\mathbf{n}\mathbf{n}^\mathsf{T} & -2d\mathbf{n}\\ \mathbf{0}^\mathsf{T} & 1\end{pmatrix}\mathbf{Q}$$

From point correspondences $(\mathbf{Q}, \mathbf{q})$, we can run any pose computation algorithm and compute the projection matrix of the virtual camera:

$$\mathsf{P} = \mathsf{R}\left(\mathtt{I}|-\mathbf{t}\right)\begin{pmatrix}\mathtt{I}-2\mathbf{n}\mathbf{n}^\mathsf{T} & -2d\mathbf{n}\\ \mathbf{0}^\mathsf{T} & 1\end{pmatrix} = \mathsf{R}\left(\mathtt{I}-2\mathbf{n}\mathbf{n}^\mathsf{T}\right)\left(\mathtt{I}\;2d\mathbf{n}-\left(\mathtt{I}-2\mathbf{n}\mathbf{n}^\mathsf{T}\right)\mathbf{t}\right) \tag{1}$$

One issue needs to be considered: pose algorithms for perspective cameras compute a pose consisting of a rotation and a translation component, whereas the above projection matrix contains a reflection part. What the pose computation will compute is thus a rotation matrix R' and a camera position $\mathbf{t}'$, with:

$$\mathsf{P} \sim \underbrace{\mathsf{R}\left(2\mathbf{n}\mathbf{n}^\mathsf{T}-\mathtt{I}\right)}_{\mathsf{R}'}\left(\mathtt{I}\;-\mathbf{t}'\right) \quad \text{with} \quad -\mathbf{t}' = 2d\mathbf{n}-\left(\mathtt{I}-2\mathbf{n}\mathbf{n}^\mathsf{T}\right)\mathbf{t}$$

Our input for the following steps is thus a set of n projection matrices P_i (we drop the ' above the R_i and $\mathbf{t}_i$):

$$\mathsf{P}_i = \mathsf{R}_i\left(\mathtt{I}\;-\mathbf{t}_i\right)$$

The basic constraint for solving our pose problem is the following (cf. §3): we try to compute n planes $\mathit{\Pi}_i$ and associated reflection matrices S_i such that

$$\forall i, j : \mathsf{P}_i\mathsf{S}_i \sim \mathsf{P}_j\mathsf{S}_j$$

If there is a unique solution for the set of planes, then any $\mathsf{P}_i\mathsf{S}_i$ gives the pose of the true camera. In the above equation, we may actually replace the equality up to scale by a component-wise equality, since the determinants of the left 3×3 submatrices of the $\mathsf{P}_i\mathsf{S}_i$ are all equal to -1. Hence, our constraint becomes:

$$\forall i, j : \mathsf{P}_i\mathsf{S}_i = \mathsf{P}_j\mathsf{S}_j$$

5 Two Mirror Positions

In this section, we investigate what can be done from just two mirror positions. Our basic constraint is:

$$\mathsf{P}_1\mathsf{S}_1 = \mathsf{P}_2\mathsf{S}_2$$

Instead of directly trying to compute the reflections S_1 and S_2, we first concentrate on:

$$\mathsf{P}_1 = \mathsf{P}_2\mathsf{S}_2\mathsf{S}_1^{-1} = \mathsf{P}_2\mathsf{S}_2\mathsf{S}_1$$

We have seen above that the sequence of two reflections gives a fixed-axis rotation. Let us thus compute R and $\mathbf{t}$ in the following euclidian transformation between the two virtual cameras:

$$\mathsf{P}_1 = \mathsf{P}_2 \begin{pmatrix} \mathsf{R} & \mathbf{t} \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix}$$

We get $\mathsf{R} = \mathsf{R}_2^\mathsf{T}\mathsf{R}_1$ and $\mathbf{t} = \mathbf{t}_2 - \mathsf{R}_2^\mathsf{T}\mathsf{R}_1\mathbf{t}_1$. In the following, we analyze what R and $\mathbf{t}$ reveal about the individual reflections S_1 and S_2.

Let α be the rotation angle of R. We already know that it equals twice the angle between the two mirror planes. Further, we want to compute the fixed axis (the intersection of the two mirror planes). Let us represent it by its direction $\mathbf{D}$ and a footpoint $\mathbf{A}$, cf. §2.4. The direction $\mathbf{D}$ is identical with the rotation axis of R and can for example be computed as its eigenvector to the eigenvalue $+1$. Let $\mathbf{D}_1$ and $\mathbf{D}_2$ be an orthonormal basis of the complement of $\mathbf{D}$, such that:

$$\mathsf{R}\begin{pmatrix}\mathbf{D}_1 & \mathbf{D}_2\end{pmatrix} = \begin{pmatrix}\mathbf{D}_1 & \mathbf{D}_2\end{pmatrix}\begin{pmatrix}\cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha\end{pmatrix}$$

As for the footpoint $\mathbf{A}$, we compute it as follows. Since we want $\mathbf{A}$ to be "orthogonal" to $\mathbf{D}$, we can parameterize it by two scalars a_1 and a_2:

$$\mathbf{A} = \begin{pmatrix}\mathbf{D}_1 & \mathbf{D}_2\end{pmatrix}\begin{pmatrix}a_1 \\ a_2\end{pmatrix}$$

The translation part of the fixed-axis rotation would thus be (cf. §2.4):

$$\mathbf{A} - \mathsf{R}\mathbf{A} = \begin{pmatrix}\mathbf{D}_1 & \mathbf{D}_2\end{pmatrix}\begin{pmatrix}1-\cos\alpha & \sin\alpha \\ -\sin\alpha & 1-\cos\alpha\end{pmatrix}\begin{pmatrix}a_1 \\ a_2\end{pmatrix}$$

In the absence of noise, this would be equal to $\mathbf{t}$. However, with noise, the computed R and $\mathbf{t}$ will in general not exactly represent a fixed-axis rotation. We thus determine a_1 and a_2 that minimize the L_2 norm of:

$$\mathbf{t} - \begin{pmatrix}\mathbf{D}_1 & \mathbf{D}_2\end{pmatrix}\begin{pmatrix}1-\cos\alpha & \sin\alpha \\ -\sin\alpha & 1-\cos\alpha\end{pmatrix}\begin{pmatrix}a_1 \\ a_2\end{pmatrix}$$

This is a linear least squares problem, with the following closed-form solution:

$$\begin{pmatrix}a_1 \\ a_2\end{pmatrix} = \frac{1}{2}\begin{pmatrix}1 & -\cot\frac{\alpha}{2} \\ \cot\frac{\alpha}{2} & 1\end{pmatrix}\begin{pmatrix}\mathbf{D}_1^\mathsf{T} \\ \mathbf{D}_2^\mathsf{T}\end{pmatrix}\mathbf{t}$$

So far, we have computed the axis and angle of the fixed-axis rotation being the sequence of S_2 and S_1. What does this tell us about the mirror planes Π_1 and Π_2? The axis being the planes' intersection line, we know that both planes must contain it; this determines each plane up to a rotation about the axis. Further, we know the angle between the planes ($\alpha/2$). In addition, not explained here in more detail, we know the "ordering" of the two planes, i.e. the second plane has always to be on the same side of the first (in terms of rotation about their intersection line). Overall, we thus have computed the two mirror planes up to a single unknown. It can be shown (not done due to lack of space) that this can not be reduced further with only two planes.

A geometric illustration of the situation is given in figure 1. For simplicity, we show the scene as seen from the direction of the mirror planes' intersection line. On the right, the ambiguity in the inferred pose of the true camera is shown: its position can lie anywhere on the circle that is centered in the fixed-axis rotation axis, is

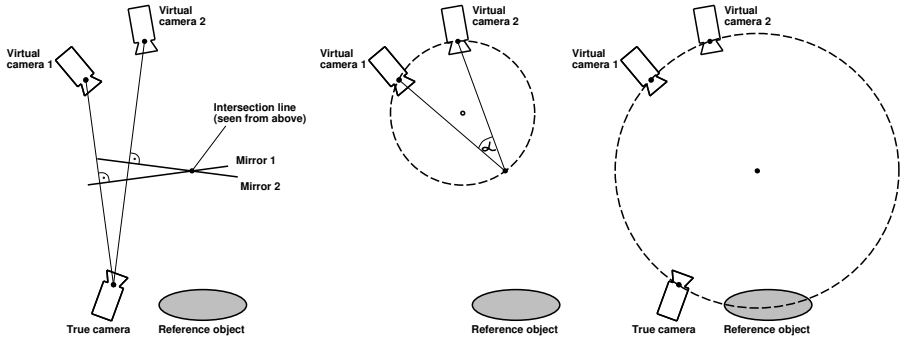


Fig. 1. Illustration of the case of two planar mirrors. **Left:** the virtual cameras are the reflections of the true one in the planar mirrors. **Middle:** the horopter curve of the two virtual cameras is the union of the shown circle and the axis of the fixed-axis rotation, i.e. the mirror planes' intersection axis. Further shown is the angle α of the fixed-axis rotation. **Right:** the true camera pose can be recovered up to one degree of freedom. The reconstructed camera position is only constrained to lie on the shown circle.

"orthogonal" to the latter and passes through the two virtual camera positions. Let us call this circle the *pose circle*. For every possible camera position on the pose circle though, the camera's orientation is uniquely defined.

All possible poses for the true camera can be parameterized by an angle β as follows. Any plane containing the axis of the fixed-axis rotation, can be written

$$\Pi \sim \begin{pmatrix} \mathbf{D}_1 & \mathbf{D}_2 \\ -a_1 & -a_2 \end{pmatrix} \begin{pmatrix} \cos\beta \\ \sin\beta \end{pmatrix}$$

for some β. We can thus parameterize the possible poses of the true camera by β, by reflecting any of the virtual ones, say the first, in the family of planes Π.

6 Three or More Mirror Positions

With three or more mirror positions, our pose problem will in general be solvable. Different approaches are possible. One could for example use the solution of the previous section for all available pairs of mirror positions. The problem could be geometrically expressed as one of finding the common point of a set of circles in 3-space (the circles as sketched in the right part of figure 1). A few special cases need to be discussed:

- Consider the case where the planar mirror is rotated about some axis contained in the mirror plane. In that case, the fixed-axis rotations for pairs of mirror positions will all have the same axis, and the resulting pose circles will all be identical. The pose of the true camera will remain ambiguous. Note that this case also refers to mirror positions that are parallel to each other (this can be seen as rotating the mirror about a line at infinity).
- The case where the mirror moves in such a way that it remains tangent to some cylinder. This implies that the lines of intersection for pairs of positions, will be parallel to one another. Hence, all pose circles lie on the same plane but since we have three or more of them, there will be a single common point: the position of the true camera.
- If there are intersection lines for pairs of mirror positions that are not parallel, then there are pose circles with different supporting planes. It can be shown that there will be pose circles in at least three supporting planes with different normals. Consequently, the set of pose circles can only have a single common point (the intersection point of all supporting planes), meaning that again, the pose problem can be solved.

In the following, we present a less geometrical method for combining results from pairs of mirror positions. Consider mirror position i. Using §5, we can compute the intersection lines of the mirror in that position, with any other position. The plane at position i has to contain all these lines, and can thus be uniquely computed from two or more lines, unless all of them are identical (cf. the above discussion). In the presence of noise, there will not be a plane that exactly contains all these lines, and we perform a fitting procedure, as follows. First, perform a least-squares fit to the direction vectors of all available lines. This will be used as the plane's normal vector. Then, compute the plane position (the scalar d appearing elsewhere in this paper) that minimizes the sum of squared distances to the available footpoints. This method fails if all lines are parallel. An alternative procedure for that case is simple to devise though.

Once mirror plane positions are computed, we reflect the virtual cameras in the corresponding planes, to obtain the true camera's projection matrix. The complete method is summarized in the next section.

7 Complete Approach

1. For each mirror position, compute the pose of the virtual camera, relative to the object.
2. For each pair of mirror positions, compute the fixed-axis rotation between the virtual cameras.
3. For each mirror position, compute the mirror plane by fitting it to the associated axes of fixed-axis rotations.
4. Compute the true camera's projection matrix by reflecting any virtual camera in the associated plane.
5. Do a non-linear bundle adjustment: minimize the sum of squared reprojection errors, over the pose of the true camera and the positions of the mirror plane. This is implemented in the usual sparse manner [12].

For conciseness, we did not mention that in practice we also compute intrinsic camera parameters during this procedure: in the first step, we also calibrate the camera. In practice, we use a planar reference object; we thus use the method of [13, 9] to calibrate the camera from the reflected views (the reflection does not alter the intrinsic parameters), prior to computing pose in step 1. Further, the bundle adjustment in the last step also optimizes intrinsic camera parameters.

8 Experiments

8.1 Setup

We use an LCD monitor as reference object, considering it effectively as a planar surface. A structured light type approach [8] is used to get correspondences between the screen and the image plane: for each position of the planar mirror, we actually take a set of images, with the screen displaying a sequence of different patterns (cf. figure 2). Patterns are designed such that for each pixel in the image plane, we can directly compute the matching "point" on the screen, from the sequence of black-and-white (dark-and-light) greylevels received at the pixel.

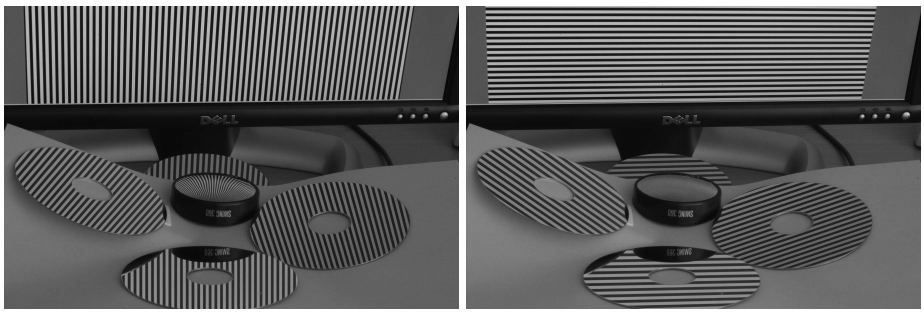


Fig. 2. Two images of our setup. Four planar mirrors (hard disk platters) are placed simultaneously in the scene. The object in the middle is a curved mirror, which was not used for the experiments reported here. The LCD monitor is partly visible in the image, but only its reflections are used to compute camera pose.

8.2 Surface Reconstruction

We tested our method on real images, cf. figure 2. It was difficult to evaluate the estimated pose, so we evaluate it indirectly as follows. In [1], we describe an approach for the reconstruction of general specular surfaces from two images of the reference object's reflections. Here, to perform a quantitative evaluation, we reconstruct a planar specular surface (a hard drive platter), without making use of the planarity information for the reconstruction. Images are taken with a fixed pose of the camera and the specular surface (cf. figure 3), but with two different positions of the LCD monitor. Each of the two positions is estimated using the approach presented in this paper, by placing planar mirrors in the scene and making use of the knowledge of planarity.

The specular surface is reconstructed as a dense point cloud [1], to which we fit a plane (linear least squares fitting without outlier removal). Over 98% of the roughly 525,000 computed points were less than 0.2 mm away from the computed plane and 64% less than 0.1 mm. The approximate diameter of the reconstructed part of the platter was 80 mm, resulting in a 0.3% relative error in the reconstruction. Refer to figure 4 for the histogram of point-plane distances.

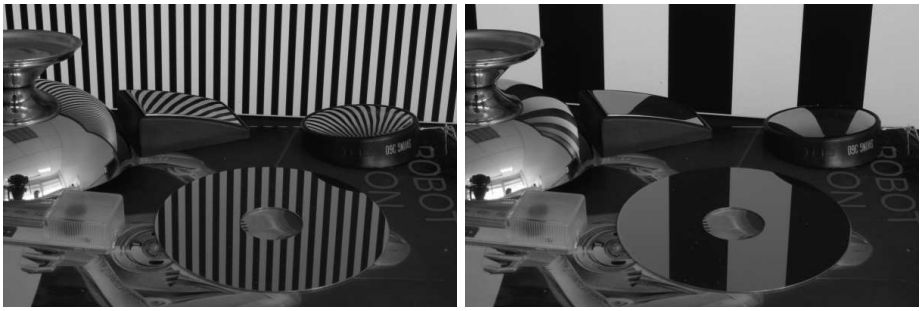


Fig. 3. Two of the images used for the reconstruction of the planar hard drive platter.

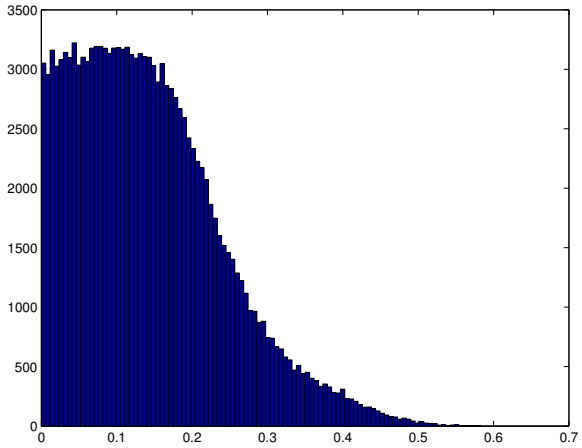


Fig. 4. Point-plane distance. Histogram of the distance of each point to the linear least squares fitted plane (in mm).

9 Conclusions

We have addressed the problem of computing the pose of an object relative to a camera, without any direct view of the object. This problem has to our knowledge not been studied yet. A theoretical study and a practical algorithm have been provided, making use of planar mirrors put in unknown positions in the scene. It was shown that with three mirror positions or more, the problem can in general be solved.

Although rather specific, this problem is very relevant for our work on specular surface reconstruction, which like many similar works uses setups where the camera has not direct view of the reference object.

The method was shown to work with real images, although by an indirect evaluation via a specular surface reconstruction method. A more in-depth evaluation using simulated data should be done, but it seems to be reasonable to assume that the performances will be similar to those of calibration and pose estimation of a camera from several images of a planar calibration grid [13, 9] (the number of parameters and the geometries of the problems are similar).

References

1. T. Bonfort, P. Sturm, P. Gargallo. General Specular Surface Triangulation. ACCV, 2006.
2. Y. Caspi, M. Irani. Alignment of Non-Overlapping Sequences. ICCV, 76-83, 2001.
3. M.A. Halstead, B.A. Barsky, S.A. Klein, R.B. Mandell. Reconstructing Curved Surfaces from Specular Reflection Patterns Using Spline Surface Fitting of Normals. SIGGRAPH, 335-342, 1996.
4. S. Kammel, F. Puente León. Deflectometric measurement of specular surfaces. IEEE Instrumentation and Measurement Technology Conference, 531-536, 2005.
5. S. Maybank. Theory of Reconstruction from Image Motion. Springer Verlag, 1993.
6. M. Oren, S.K. Nayar. A Theory of Specular Surface Geometry. IJCV, 24(2), 1996.
7. S. Savarese, M. Chen, P. Perona. Recovering local shape of a mirror surface from reflection of a regular grid. ECCV, 2004.
8. D. Scharstein, R. Szeliski. High-accuracy stereo depth maps using structured light. CVPR, 195-202, 2003.

9. P. Sturm, S. Maybank. On Plane-Based Camera Calibration. CVPR, 432-437, 1999.
10. P. Sturm. Algorithms for Plane-Based Pose Estimation. CVPR, 706-711, 2000.
11. M. Tarini, H. Lensch, M. Goesele, H.-P. Seidel. 3D acquisition of mirroring objects. Research Report MPI-I-2003-4-001, Max-Planck-Institut für Informatik, 2003.
12. B. Triggs, P.F. McLauchlan, R.I. Hartley, A. Fitzgibbon. Bundle Ajustment – A Modern Synthesis. Vision Algorithms, 298-372, 1999.
13. Z. Zhang. A flexible new technique for camera calibration. PAMI, 22(11), 2000.
14. J.Y. Zheng, A. Murata. Acquiring 3D object models from specular motion using circular lights illumination. ICCV, 1101-1108, 1998.

Chapter 14

Modelling of 3D Geometry and Reflectance Properties

Paper 37 [5]: N. Birkbeck, D. Cobzaş, P. Sturm, and M. Jägersand. Variational shape and reflectance estimation under changing light and viewpoints. In H. Bischof and A. Leonardis, editors, *Proceedings of the 9th European Conference on Computer Vision, Graz, Austria*, Lecture Notes in Computer Science, May 2006.

Variational Shape and Reflectance Estimation Under Changing Light and Viewpoints

Neil Birkbeck[1], Dana Cobzas[1], Peter Sturm[2], and Martin Jagersand[1]

[1] Computer Science, University of Alberta, Canada
{birkbeck, dana, jag}@cs.ualberta.ca
[2] INRIA Rhone-Alpes, France
peter.sturm@inrialpes.fr

Abstract. Fitting parameterized 3D shape and general reflectance models to 2D image data is challenging due to the high dimensionality of the problem. The proposed method combines the capabilities of classical and photometric stereo, allowing for accurate reconstruction of both textured and non-textured surfaces. In particular, we present a variational method implemented as a PDE-driven surface evolution interleaved with reflectance estimation. The surface is represented on an adaptive mesh allowing topological change. To provide the input data, we have designed a capture setup that simultaneously acquires both viewpoint and light variation while minimizing self-shadowing. Our capture method is feasible for real-world application as it requires a moderate amount of input data and processing time. In experiments, models of people and everyday objects were captured from a few dozen images taken with a consumer digital camera. The capture process recovers a photo-consistent model of spatially varying Lambertian and specular reflectance and a highly accurate geometry.

1 Introduction

The automatic computation of 3D geometric and appearance models from images is one of the most challenging and fundamental problems in computer vision. While a more traditional point-based method provides accurate results for camera geometry, a surface representation is required for modeling and visualization applications. Most surface-based approaches reconstruct the model based on stereo correlation data [1, 2, 3]. That works well for textured Lambertian surfaces but fails in the presence of specular highlights or uniform texture. Additionally, stereo-based techniques reconstruct only the shape and not the surface reflectance properties even though some approaches can handle specular objects using robust scores [4, 5].

We are proposing a surface reconstruction method that uses texture and shading information to successfully reconstruct both textured and non-textured objects with general reflectance properties. The similarity cost functional uses a parametric reflectance model that is estimated together with the shape. There exist other approaches that combine stereo for textured regions with shape from

A. Leonardis, H. Bischof, and A. Prinz (Eds.): ECCV 2006, Part I, LNCS 3951, pp. 536–549, 2006.

shading cues for texture-less regions [6, 7], but, in those works, the two scores are separate terms in the cost function and the combination is achieved either using weights [6] or by manually assigning regions [7]. Additionally, they only exploit diffuse objects whereas our method can also handle specular objects. Like photometric stereo, our method is able to reconstruct the surface of spatially varying or uniform material objects by assuming that the object is moving relative to the light source. Zhang et al. [8] and Weber et al. [9] also use light variation for reconstructing spatially varying albedo. But, in contrast to our approach, they do not consider the challenge of dealing with specular surfaces.

With respect to recovering specular surfaces, most of the approaches either filter or remove specularities and use only diffuse observations in the reconstruction [5]. Another option is to design similarity scores that account for specular highlights either by assuming a uniform surface material [10] or by enforcing dimensionality constraints on the observed intensity variations [11]. A more general approach is to explicitly model surface reflectance either with a parametric model [12] or a non-parametric model (BRDF map). Obtaining a BRDF map requires carefully calibrated lights and many samples [13]. For our system we made the choice of using a parametric model for reflectance as we are interested in reconstructing both shape and reflectance parameters.

Different representations have been proposed for shape reconstruction; they can be divided in two main classes - image-based (depth/disparity) and object-based (voxel grid, mesh, level set). Image-based representations are suitable for single view or binocular stereo techniques, but object based representations, which are not tied to a particular image, are more suitable for multi-view reconstruction. Mesh and level set techniques have the advantage over voxel representations that they give readily computable normals (essential in recovering shading). Additionally, the regularization terms can be easily integrated into a mesh or level set. An implicit level set representation leads to an elegant algorithm [2], but despite various efficient numerical solutions proposed for the level set methods [14], they are still slow compared to mesh based approaches that can take advantage of graphics hardware acceleration. We therefore decided to implement our method using an adaptive deformable mesh that allows for topological changes. The mesh is evolved in time based on a variational algorithm. Fua and Leclerc [6] and Duan et al. [15] have presented related variational mesh-based approaches but not as general as they only reconstruct diffuse objects.

Due to the high dimensionality, reconstruction can be difficult, slow and require lots of image data. To ameliorate these problems, we propose a multi-resolution algorithm that alternates between shape and reflectance estimation. Although in theory a general reflectance model can be estimated at every step, in practice we noticed that similar results are obtained more efficiently if the shape reconstruction is performed on filtered diffuse pixels assuming Lambertian reflectance. A Phong parametric model is then calculated using the final shape. Experiments show that the proposed method is able to reconstruct accurate and photo-realistic models that can be rendered in novel illumination conditions. To summarize, the main contributions of the paper are:

- We designed a photo-consistency functional suitable for surfaces with non-uniform general reflectance based on a parametric reflectance model;
- We present a variational method implemented as a PDE-driven mesh evolution interleaved with reflectance estimation. Our particular mesh implementation is robust to self-intersections while allowing topological changes;
- We designed a practical setup that provides the necessary light variation, camera and light calibration and requires only commonly available hardware: a light source, a camera, and a glossy white sphere.

2 Shape Refinement

We present the shape refinement problem beginning with a general continuous formulation that is then discretized on the mesh triangles. Next, we describe a numeric solution to the resultant optimization problem for an object with Lambertian or specular reflectance.

2.1 Problem Definition

The proposed capture setup consists of a single camera viewing an object placed on a turntable illuminated by a desk lamp. We take two sets of images of a full rotation, each with a different light position. Considering the proposed capture setup, the shape recovery problem takes the following as input:

- a set of n images $\mathcal{I} = \{I_i | i = 1 \cdots n\}$;
- the associated projection matrices P_i;
- the illumination information $L_i = (\mathbf{l}_i, l_i)$, assuming a single distant light source with direction $\mathbf{l}_i$ and color l_i;
- an initial shape S_0;

and computes a refined shape, S, and the corresponding reflectance parameters that best agree with the input images. A practical method for automatically calibrating the camera and the light is presented in Section 4.

Given the projection matrix $P_i = K[R_i, \mathbf{t}_i]$, the image coordinates $\mathbf{p}_i = (u_i, v_i, 1)^T$ for a 3D point $\mathbf{x}$ are expressed as $\mathbf{p}_i = \Pi(P_i\mathbf{x})$. Π represents the non-linear operator that transforms homogeneous coordinates into Cartesian ones (division with the homogeneous component).

We assume that surface reflectance is a parametric function implied by the surface (and surface normals) and imaging conditions. Therefore, the shape reconstruction problem is to recover a shape and its implied reflectance parameters that best agree with the input images. The shape and reflectance are estimated in an alternate fashion (see Section 4).

2.2 Shape Functional

We use a variational formulation for the shape recovery problem similar to the one from Faugeras and Keriven [2].

$$E(S) = \int_S g(\mathbf{x}, \mathbf{n}) dS = \int_v \int_u g(\mathbf{x}, \mathbf{n}) \|S_u \times S_v\| du dv \tag{1}$$

where $\mathbf{x} = (x(u,v), y(u,v), z(u,v))^T$ is a point on the surface and $\mathbf{n} = \frac{S_u \times S_v}{\|S_u \times S_v\|}$ is the surface normal at point $\mathbf{x}$.

The photo-consistency function g encodes the similarity between a point on the surface, and the images in which it is observed. We investigate a similarity function of the form:

$$g(\mathbf{x}, \mathbf{n}) = \sum_i h(\mathbf{x}, \mathbf{n}, P_i, L_i) \left(I_i(\Pi(P_i \mathbf{x})) - R(\mathbf{x}, \mathbf{n}, L_i)\right)^2 \tag{2}$$

where R is a rendering equation returning the color of point $\mathbf{x}$ under light conditions L_i. The function h is a weighting function that accounts for visibility and discrete sampling effects. Refer to Fig. 1 for a explanation of our notations.

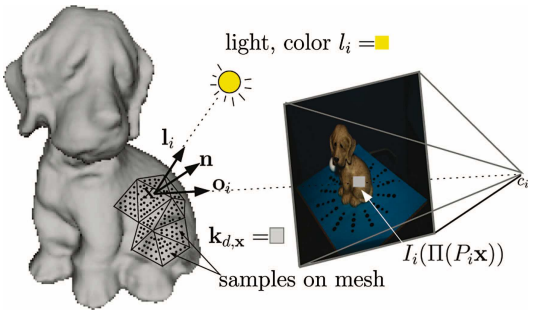


Fig. 1. An illustration of the sample points and the angles used in the shading equation

Rendering function. The function R encodes the reflectance model at a point $\mathbf{x}$ on the surface. In fact, R is a function of the entire surface as it should account for inter-reflections and shadowing of a point $\mathbf{x}$. In our capture setup we minimized self shadowing and inter-reflections and therefore ignored these subtleties. We model R with a parametric BRDF which is fitted to Eq. 2 (assuming known shape and imaging conditions).

For modeling the parametric BRDF we chose the Lambertian model to represent diffuse reflectance and the Phong model for the specular reflectance. The two models are briefly summarized below[1].

Lambertian model assumes constant BRDF and effectively models matte objects, such as clay, where the observed shading is a result of the foreshortening contribution of the light source. Integrating the Lambertian BRDF model into the reflectance equation we get the following expression for the observed color at a particular point $\mathbf{x}$ with normal $\mathbf{n}$:

$$R_{lamb}(\mathbf{x}, \mathbf{n}, L_i) = (\langle \mathbf{n}, \mathbf{l}_i \rangle l_i + a_i) k_{d,\mathbf{x}} \tag{3}$$

[1] The proposed method works with color images but for simplicity reasons we present the theory for one color channel. In practice the colors are vectors in RGB space.

where $k_{d,\mathbf{x}}$ represents the Lambertian color (albedo). For better modeling of light effects in a normal room we incorporate an ambient term to capture the contribution of indirect light in each image a_i.

Specular reflectance is typically modeled as an additive component to the Lambertian model. We chose to represent the specular BRDF using the Phong model. Letting $\mathbf{o}_i$ be the outgoing direction from the point $\mathbf{x}$ to the center of the camera i (i.e., the view direction), and $\mathbf{h}_i$ the bisector of the angle between the view and the light directions $\mathbf{h}_i = \frac{\mathbf{o}_i+\mathbf{l}_i}{\|\mathbf{o}_i+\mathbf{l}_i\|}$ the shading model for a specular pixel is (refer to Fig. 1 for an illustration):

$$R_{spec}(\mathbf{x}, \mathbf{n}, L_i) = (\langle \mathbf{n}, \mathbf{l}_i \rangle l_i + a_i) k_{d,\mathbf{x}} + \langle \mathbf{n}, \mathbf{h}_{i,\mathbf{x}} \rangle^m l_i k_s \tag{4}$$

where k_s is the specular color and m is the specular exponent. The specular parameters are not indexed per point due to the fact that several observations are needed for reliably estimating the BRDF. Instead (as discussed in Section 3) we compute the specular parameters for groups of points having similar diffuse component, thus likely to have the same material.

Weight function. The similarity measure with respect to an image should be computed only for the visible points. This can be easily represented by setting the weight function, h, to the binary visibility function $V(\mathbf{x}, S, P_i)$.

To ensure that only relevant image information is used in evaluation of g, we use a subset of image observations for each point on the surface. In particular, we use the $n_{cameras}$ closest cameras to the median camera [5], where the median camera is chosen based on the azimuthal angle. This camera selection gives another binary weight function V'. Another sampling issue arises because a surface patch projects to a different area in each image. We compensate for this by giving more weight to observations that have frontal views and less weight to grazing views. This is accomplished by weighting the samples by $\langle \mathbf{n}, \mathbf{o}_i \rangle$. Cumulating visibility and sampling into the function h we get:

$$h(\mathbf{x}, \mathbf{n}, P_i, L) = \langle \mathbf{n}, \mathbf{o}_i \rangle V'(\mathbf{x}, S, P_i) \tag{5}$$

2.3 Surface Evolution

Optimizing the photo-consistency function in Eq. 1 with respect to the surface S results in a surface evolution problem. The gradient flow PDE is derived from the Euler-Lagrange equation of Eq. 1. The PDE contains higher order terms [2] resulting from the general form of g being a function of $\mathbf{n}$. Instead of using the full PDE, complete with the higher order terms, we use a simplified PDE containing only the first order terms. This flow is accurate for a g that is only a function of surface position $\mathbf{x}$. Similar PDE's were used by [16, 15] but with different g functions.

$$\frac{\partial S}{\partial t} = (2g\kappa - \langle \nabla g, \mathbf{n} \rangle)\mathbf{n} \tag{6}$$

where κ is the mean curvature. The flow will move each point along the current estimate for the normal. The first component of the motion in Eq. 6, $2g\kappa$, is

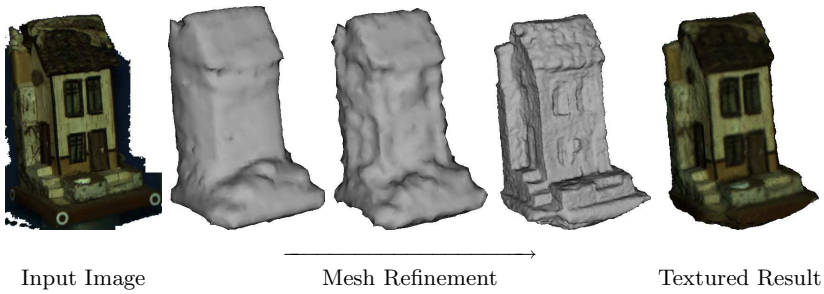


Fig. 2. An example of the mesh evolving to a refined shape

essentially a smoothing term, reducing the mean curvature of the object, whereas the second component ensures the evolution decreases the error function on the surface.

The shape refinement then proceeds by iteratively updating the initial shape, S_0, using Eq. 6 until convergence. We stop the evolution when there is no significant change in the error function for several steps. Fig 2 gives an example of our surface evolution algorithm starting from the visual hull.

2.4 Discretization on the Triangular Mesh

The numerical solution for the surface evolution depends on the chosen representation. As we explore the use of a mesh based representation, we must first break the integral into a sum of integrals over the triangles. Let $\Delta = (\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3)$ be a triangle having vertices $\mathbf{v}_1, \mathbf{v}_2$ and $\mathbf{v}_3$. An interior point on the triangle can be expressed using the barycentric coordinates $\lambda_1, \lambda_2, \lambda_3$ satisfying $\lambda_1 + \lambda_2 + \lambda_3 = 1$ and $\lambda_k \geq 0$ for $k \in \{1, 2, 3\}$: $\mathbf{x} = \lambda_1\mathbf{v}_1 + \lambda_1\mathbf{v}_2 + \lambda_1\mathbf{v}_3$. The triangle normal $\mathbf{n}$ is then computed by smoothly interpolating the normals $\mathbf{n}_1, \mathbf{n}_2, \mathbf{n}_3$ of the vertices: $\mathbf{n} = \lambda_1\mathbf{n}_1 + \lambda_2\mathbf{n}_2 + \lambda_3\mathbf{n}_3$.

The integrals are then composed into a sum of regularly spaced sample points over the triangles, giving:

$$E(S) \approx \sum_{\{\mathbf{v}_1,\mathbf{v}_2,\mathbf{v}_3\}\in\Delta} \sum_{\{\lambda_1,\lambda_2,\lambda_3\}} g(\lambda_1\mathbf{v}_1 + \lambda_2\mathbf{v}_2 + \lambda_3\mathbf{v}_3, \lambda_1\mathbf{n}_1 + \lambda_2\mathbf{n}_2 + \lambda_3\mathbf{n}_3) \quad (7)$$

The method of computing the error on sampling points within the triangles relates our work to other mesh based approaches [6, 17, 10, 12]. An alternative approach, used in the work of Duan et al. [15], is to sample the error on the tangent plane of the mesh vertices.

Although a small number of samples points (e.g., using only the mesh vertices) may be sufficient for textureless surfaces, a textured surface may require a dense sampling that matches the image resolution. We use a dense sampling to ensure the method works on either textured or textureless surfaces.

One way to implement the gradient flow given by Eq. 6 is to derive the analytic gradient of g. But, there are several problems with the analytic gradient.

First, the visibility changes are not taken into account. While moving vertices it is possible that some parts of the surrounding triangles become occluded or visible (un-occluded), which is not taken into account by the analytic gradient. A second remark is that the formulas do not account for reflectance changes as the reflectance properties could only be computed after taking the step. Similar to the visibility case, moving a vertex results in changes in the shading. For these reasons we use a numerical computation for the gradient.

Numerical Gradient. The gradient of the similarity function along the direction of the normal, $\nabla g \cdot \mathbf{n}$, is computed numerically using central differences. Letting $g_{\mathbf{v}^+}$ (resp. $g_{\mathbf{v}^-}$) be the error computed on the mesh when a vertex $\mathbf{v}$ is replaced with $\mathbf{v}^+ = \mathbf{v} + \mathbf{n}\Delta n$ (resp. $\mathbf{v}^- = \mathbf{v} - \mathbf{n}\Delta n$), then:

$$\nabla g \cdot \mathbf{n} \approx \frac{g_{\mathbf{v}^+} - g_{\mathbf{v}^-}}{2\Delta n}$$

where $\Delta n = c_\Delta \sigma_{mesh}$ and $c_\Delta \in (0, 1]$, to ensure that the derivative step size is bounded by the minimum edge length (a tuning parameter σ_{mesh} explained in Section 4.1).

In order to compute the gradient efficiently, without displacing each vertex individually and computing the error over the entire mesh, we compute the gradient for a set of vertices simultaneously [18]. The idea is to partition the mesh into disjoint sets of vertices such that moving a vertex from a set does not influence the error for the rest of the vertices in that set. Ignoring visibility issues, displacing a vertex $\mathbf{v}$ affects all triangles within distance 2 from $\mathbf{v}$. Therefore, the gradient computation for a vertex $\mathbf{v}$ must do the reflectance fitting and error computation for these affected triangles. This means that we can displace other vertices at the same time as long as they do not both affect the same triangles.

3 Reflectance Fitting

As previously mentioned, we assume that the reflectance function is implied by the shape and imaging conditions. We experimented with two parametric reflectance models briefly introduced in Section 2.2 : Lambertian for diffuse and Phong for specular surfaces. We describe here how we practically recover the reflectance parameters from a set of images given a shape S, illumination conditions L_i and calibration parameters P_i.

3.1 Lambertian Reflectance

Lambertian reflectance has only one parameter per point $\mathbf{x}$ (the albedo $k_{d,\mathbf{x}}$). The albedo for each point $\mathbf{x}$ on the mesh with normal $\mathbf{n}$ is fit to the image observations for the current shape by minimizing

$$g_{lamb}(\mathbf{x}, \mathbf{n}) = \sum_i \langle \mathbf{n}, \mathbf{o}_i \rangle V'(\mathbf{x}, P_i) \left(I_i(\Pi(P_i \mathbf{x})) - (\langle \mathbf{n}, \mathbf{l}_i \rangle l_i + a_i) k_{d,\mathbf{x}} \right)^2 \quad (8)$$

which has a simple closed form solution using least squares.

3.2 Specular Reflectance

The parameters of the specular reflectance can be estimated given a set of input images, an object surface, and illumination information, by minimizing the similarity measure (Eq. 2). For a low parameter BRDF model, as the Phong model, given enough observations, the parameters can be estimated efficiently using an indirect iterated linear approach [19] or by a more direct non-linear method [20].

In practice, with only a limited number of input images, it is not always possible to fit a full reflectance model at each surface point. Instead of fitting the full model at each surface point, we chose to use an interpolation method that first attempts to fit the Phong model to the observations at each point. A reliable fitting is only possible when a point has several observations with a small angle between the surface normal and bisector of viewing and illumination direction. If there are not enough observations, the specular parameters will not be estimated correctly, leaving only a correctly fit Lambertian model. These points are assigned the specular parameters of a point where the specular fitting was successful. This assignment is based on the diffuse color of the point.

3.3 Filtering Specular Highlights

In practice, it is inefficient to fit a full reflectance model to each surface point during the optimization. Instead of fitting the full reflectance model, we choose to filter out the specular highlights during the optimization and perform the shape refinement only for diffuse observations.

It is known that specular highlights occur at points having a large $\langle \mathbf{n}, \mathbf{h}_i \rangle$. As a consequence, one approach is to give smaller weights (in the h function) to those observations [21]. But, for a surface estimation method it is not the best approach as it relies on the current estimate of $\mathbf{n}$. Another approach, and the one used in this work, is to use the fact that specular highlights typically cause a bright image observation. Therefore, a fraction of the samples having the brightest intensity (typically 1/3) are excluded from the computation of the albedo and the g measure for a point. This type of filtering is essentially another binary function, like the visibility function V.

4 System and Implementation Details

Recall that our formulation of the shape refinement problem requires calibrated input images, a calibrated light source, and an initial shape. We use a turntable based capture setup as an easy way to capture many views of an object, while automatically providing light variation, and allowing for an initial shape to be computed from the object's silhouette.

Our particular capture setup consists of a single camera viewing an object rotating on a turntable (see Fig. 3). Each set of images observes a full rotation of the object but has a different light position. In practice, the elevation of the light is varied between the two sets of images, and the light is positioned in a

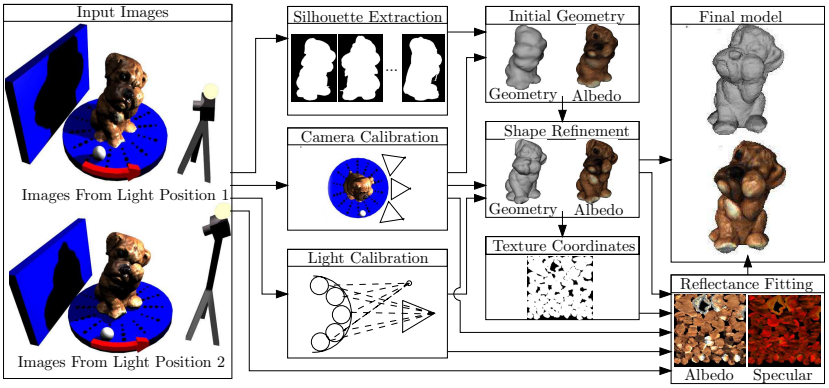


Fig. 3. Overview of the system used to scan objects

manner to avoid cast shadows (i.e., the source is placed close to the camera, implying that the camera also changes between the two sets of images).

The camera position is obtained through the automatic detection of a calibration pattern that is similar to the one used by Baumberg et al. [22]. A regular desk lamp is used as the light source and provides the majority of the illumination. The object rotates in front of a solid colored background, and a PCA based color segmentation is used to extract a set of silhouette images, which are used with shape from silhouette (SFS) to provide an initial shape.

The light source position and color are calibrated using a single glossy white sphere, which rotates along with the object on the turntable. Our approach is similar to other approaches that use a set of metallic spheres to calibrate a light source (e.g., [23]). The image of the specular highlight on the sphere in several views is used to triangulate the position of the source. As we used a white sphere, the non-specular pixels of the sphere are used to calibrate the light source color.

In order to make the recovered model useful in computer graphics applications, the reflectance model is represented in texture maps. As a prerequisite, we first need to obtain texture coordinates for the refined model. For this task, we have implemented a method similar to that of Lévy et al. [24].

4.1 Overview of the Shape Refinement Algorithm

The two components of the refinement in Eq. 6 are the gradient of the cost function and the regularizing component. The gradient is approximated per vertex using central differences, which was discussed in Section 2.4. The driving force behind the regularizing term is the mean curvature on the object, κ, which can be effectively approximated using a paraboloid method [25]. For a particular vertex, the mean curvature is computed by first finding the transformation taking the vertex to the origin and aligning its normal with the positive z axis. This transformation is applied to the neighboring vertices, and a paraboloid,

$z = ax^2 + bxy + cy^2$, is then fit to the transformed points. The mean curvature at the vertex is $\kappa = a + c$.

To handle topological changes in the mesh, we use the method proposed by Lachaud and Montanvert [26]. The mesh has a consistent global resolution, where edge lengths are confined to be within a certain range, i.e., if e is an edge in the mesh then $\sigma_{mesh} \leq \|e\| \leq 2.5\sigma_{mesh}$. A simple remesh operation ensures that the edges are indeed within this range and also performs the necessary operations related to topology changes. The global resolution of the mesh can be adjusted by altering this edge length parameter, σ_{mesh}.

The refinement starts with a low resolution mesh (i.e., large σ_{mesh}) and the corresponding low resolution images in a Gaussian pyramid. When the progress at a particular mesh resolution slows, the mesh resolution (and possibly the corresponding resolution in the Gaussian pyramid) is increased. This multi-resolution approach improves convergence, as there are fewer vertices (i.e., degrees of freedom), and enables the mesh to recover larger concavities.

5 Experiments

We have performed several experiments on synthetic and real image sequences to demonstrate the effectiveness of the method described in this paper. For the real sequences, the images were captured with either a consumer Canon Powershot A85 digital camera or a Point Grey Research Scorpion firewire camera. We used roughly 6 mesh resolutions during the refinement, and the total time for refinement was typically between 20 minutes and 1 hour. The captures contained roughly 60 input images and we found that using $n_{cameras} = 12$ simultaneous images provided sufficient results for many of the sequences. In the final stages of the refinement this parameter was increased to 24.

The first experiment demonstrates the refinement of an object that a standard correlation based method would have problems with: a 3D printout of the Stanford bunny model with uniform Lambertian reflectance. An initial shape obtained from SFS is a good approximation to the bunny, but several indentations near the legs of the bunny are not recovered (Fig. 4). These indentations are recovered by our method as illustrated by comparing the distance from the ground truth surface to the initial shape and the refined model (Fig. 5).

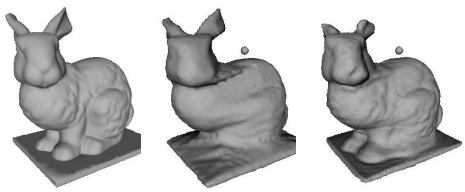


Fig. 4. From left to right a ground truth rendering, the recovered shape from SFS, and the refined model

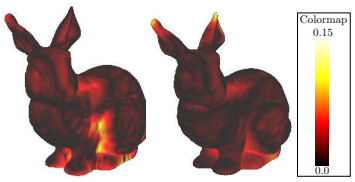


Fig. 5. A portrayal of the distance from the ground truth object to the SFS model (left) and the refined model

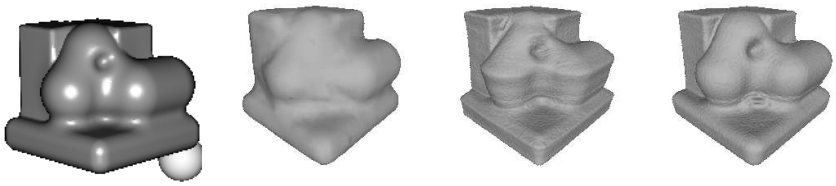


Fig. 6. From left to right, an input image of a synthetic specular object, the reconstruction from SFS, the reconstruction without specular filtering, and the reconstruction with specular filtering

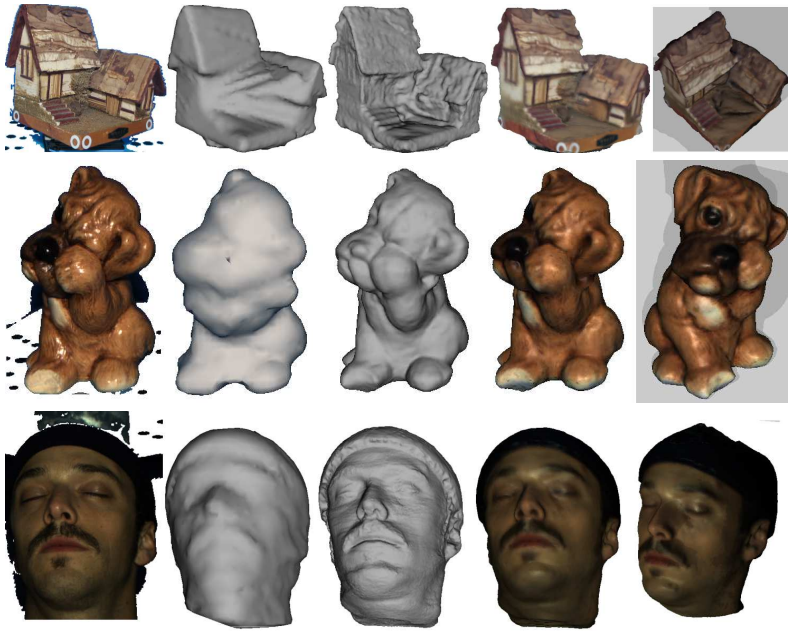


Fig. 7. Several reconstructed objects: a model house, a sad dog, and a human head

A second experiment, designed to test the effectiveness of the specular filtering, was performed on a synthetic object. The object has several concavities that were not reconstructed by the initial SFS shape (Fig. 6). The reconstruction obtained without specular filtering has artifacts. The most noticeable artifact is a sharp crease where the specularity was observed (second from the right of Fig. 6). On the other hand, the refinement that used specular filtering successfully recovered the indentations.

We have also tested the method on several real objects with both textured and glossy surfaces (Fig. 7). Our method was capable of recovering an accurate geometry on all the objects. Notice the large concavity that was recovered in the

house sequence. The fitted specular parameters give realistic highlights on the reconstructed results (see the sad dog and human head results). Unfortunately, the reconstructed specular component was not always as sharp as the true specular component, which is noticeable on the sad dog object (a similar observation was made by Yu et al. [12]).

Fig. 8. An image of a real chess board (left), followed by a novel rendering of the captured models combined into a chess game

Our high quality results are easily integrated into realistic computer graphics applications. To illustrate this, we have captured several real models of a chess game and combined them into a computer chess game (Fig. 8).

6 Discussion

We have presented a variational method that alternatively reconstructs shape and general reflectance from calibrated images under known light. The surface evolution is implemented on a deformable mesh at multiple resolutions. We have demonstrated the usefulness of the proposed method on controlled sequences, where an object was rotated relative to a light source. The results are quite accurate, proving that the method is able to reconstruct a variety of objects.

The capture setup used in this work provides an efficient way to capture a 3D model of an object, but currently we need to be able to rotate this object in front of the camera. As future work, we would like to extend our method to work on objects where this form of light variation cannot be obtained. For small outdoor statues, it may be sufficient to use the flash on the camera, or capture images on a sunny day at different times to obtain the light variation on the object. A less restrictive method would be required for larger objects (e.g., buildings).

Other future directions include finding a more efficient way to utilize the information in specular highlights instead of filtering them out and to compare the advantages of a level set implementation. We would also like to have some guarantee that the recovered surface is at (or at least near) a global minimum of the functional.

References

1. Scharstein, D., Szeliski, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. Int. J. Comput. Vision **47** (2002) 7–42
2. Faugeras, O., Keriven, R.: Variational principles, surface evolution, pde's, level set methods and the stereo problem. IEEE Trans. Image Processing **7** (1998) 336–344
3. Robert, L., Deriche, R.: Dense depth map reconstruction: A minimization and regularization approach which preserves discontinuities. In: ECCV '96. (1996) 439–451
4. Yang, R., Pollefeys, M., Welch, G.: Dealing with textureless regions and specular highlights - a pregressive space carving scheme using a novel photo-consistency measure. In: ICCV. (2003)
5. Esteban, C.H., Schmitt, F.: Silhouette and stereo fusion for 3d object modeling. Computer Vision and Image Understanding **96** (2004) 367–392
6. Fua, P., Leclerc, Y.: Object-centered surface reconstruction: combining multi-image stereo shading. In: Image Understanding Workshop. (1993) 1097–1120
7. Jin, H., Yezzi, A., Soatto, S.: Stereoscopic shading: Integrating shape cues in a variational framework. In: CVPR. (2000) 169–176
8. Zhang, L., Curless, B., Hertzmann, A., Seitz, S.M.: Shape and motion under varying illumination: Unifying structure from motion, photometric stereo, and multi-view stereo. In: ICCV. (2003)
9. Weber, M., Blake, A., Cipolla, R.: Towards a complete dense geometric and photometric reconstruction under varying pose and illumination. In: BMVC. (2002)
10. Yu, T., Xu, N., Ahuja, N.: Shape and view independent reflectance map from multiple views. In: ECCV. (2004)
11. Jin, H., Soatto, S., Yezzi, A.: Multi-view stereo reconstruction of dense shape and complex appearance. IJCV **63** (2005) 175–189
12. Yu, T., Xu, N., Ahuja, N.: Recovering shape and reflectance model of non-lambertian objects from multiple views. In: CVPR. (2004)
13. Debevec, P.E., Malik, J.: Recovering high dynamic range radiance maps from photographs. In: Siggraph. (1997)
14. Sethian, J.: Level Set Methods. Cambridge University Press (1996)
15. Duan, Y., Yang, L., Qin, H., Samaras., D.: Shape reconstruction from 3d and 2d data using pde-based deformable surfaces. In: ECCV. (2004)
16. Caselles, V., Kimmel, R., Sapiro, G., Sbert, C.: Minimal surfaces based object segmentation. PAMI **19** (1997) 394–398
17. Zhang, L., Seitz, S.: Image-based multiresolution modeling by surface deformation. Technical Report CMU-RI-TR-00-07, Carnegie Mellon University (2000)
18. Zach, C., Klaus, A., Hadwiger, M., Karner, K.: Accurate dense stereo reconstruction using graphics hardware. In: Eurographics 2003. (2003) 227–234
19. Ikeuchi, K., Sato, K.: Determining reflectance properties of an object using range and brightness images. PAMI **13** (1991) 1139–1153
20. Lafortune, E.P., Foo, S.C., Torrance, K.E., Greenberg, D.P.: Non-linear approximation of reflectance functions. In: SIGGRAPH. (1997)
21. Marschner, S.R.: Inverse rendering for computer graphics. PhD thesis, Cornell University (1998)
22. Baumberg, A., Lyons, A., Taylor, R.: 3D S.O.M. - a commercial software solution to 3d scanning. In: Vision, Video, and Graphics (VVG'03). (2003) 41–48

23. Lensch, H.P.A., Goesele, M., Kautz, J., Heidrich, W., Seidel, H.P.: Image-based reconstruction of spatially varying materials. In: Eurographics Workshop on Rendering Techniques. (2001) 103–114
24. Lévy, B., Petitjean, S., Ray, N., Maillot, J.: Least squares conformal maps for automatic texture atlas generation. In: SIGGRAPH '02. (2002) 362–371
25. Surazhsky, T., Magid, E., Soldea, O., Elber, G., Rivlin, E.: A comparison of gaussian and mean curvatures triangular meshes. In: ICRA '03. (2003) 1021–1026
26. Lachaud, J.O., Montanvert, A.: Deformable meshes with automated topology changes for coarse-to-fine 3D surface extraction. Medical Image Analysis **3** (1999) 187–207

Part V

Other Works

Chapter 15

Object Tracking

Paper 38 [14]: A. Jacquot, P. Sturm, and O. Ruch. Adaptive tracking of non-rigid objects based on color histograms and automatic parameter selection. In *Proceedings of the* IEEE *Workshop on Motion and Video Computing, Breckenridge, Colorado, USA*, pages 103–109, January 2005.

Adaptive Tracking of Non-Rigid Objects Based on Color Histograms and Automatic Parameter Selection

A. Jacquot, P. Sturm
INRIA Rhône Alpes
Grenoble, FRANCE

O. Ruch
Thales Optronique
Guyancourt, FRANCE

Abstract

One of the main difficulties in visual tracking is to take into account appearance changes (not only of the target but also of or due to the scene, illumination for example). The use of a Bayesian framework is very flexible and has proven to be very efficient in visual tracking. Moreover, color or greylevel histograms allow to track an objet with a low computational cost. The recently proposed color-based trackers integrated in a probabilistic framework [1, 3] are efficient for a given application (face tracking for example) but can not be generalized easily, due to the initialization and the adjustment of the different tracker parameters that are dependent on the input sequence. This paper presents a method based on color integrated in a particle filter that allows to cope with some of the usual problems of visual tracking (occlusions, target appearance changes, changes in resolution or in illumination) and to adapt easily to different applications (tracking of structures in aerial imagery as well as football players). The novelty of the tracker is its ability to automatically regulate all the parameters needed for tracking, which makes it flexible and easily usable for different applications.

1. Introduction

Whatever the object we want to track, tracking is based on some model describing its appearance: this model can include prior information on the target as well as information extracted from the previous frames in the sequence. The model can contain geometric contours, image patches, global descriptors or other features. One of the main factors that limits the performance of visual tracking algorithms is the lack of a suitable appearance model for the target. Template matching methods can not directly cope with appearance changes and motion estimation based methods allow the appearence model to change rapidly but tend to drift away from targets.

This paper proposes a robust appearance model for tracking using color distributions. Histograms are robust to partial occlusion, rotation and have a low computational cost. Furthermore, particle filters have proven to be efficient and reliable in cases of clutter and occlusions. Several trackers based on color histograms integrated in probabilistic frameworks have been proposed recently [1, 2, 3]; to the best of our knowledge these algorithms are efficient for a given application but can not be adapted easily to another target. Other techniques proposed by Bradski (the "Camshift" [4]), Comaniciu (the "Mean Shift" [5]) or more recently Zivkovic (who proposes an extension of the mean shift algorithm in [7]) do not use probabilities but make a deterministic search of the region whose color content best matches the reference model. These methods have the same limitations as the previous ones. The novelty of the proposed tracker lies in the integration of some criteria which allow to automatically select the number of bins of the histogram needed for the tracking and in a new way to update the model. Our tracker is robust to occlusions, changes in illumination as well as changes of appearance of the target, and has shown to be efficient in tracking objects with a hand-held camera as well as tracking buildings or static structures in aerial imagery.

The outline of this paper is as follows: in Section 2 we briefly describe the basic method: particle filtering, the way to use color for tracking and the way to integrate both. Section 3 describes the improvements we have made: the gain of spatial information obtained by dividing the patch of interest, the model update and the automatic selection of the number of bins of the model's color histogram and parameters needed for the tracking. Finally some results are presented in Section 4.

2. Color-based probabilistic tracking

The aim of this section is to present the basis of the tracking of non rigid objects using color histograms in conjunction with a probabilistic framework [1, 3].

2.1. Recalls on particle filtering

We use the Bayesian framework to track objects in the case where the posterior density $P(X_t \mid Z_t)$ and the observation model $P(Z_t \mid X_t)$ are not necessarily Gaussian. The object tracked is characterized by its state vector X_t, and the

observations up to time t are defined by the vector Z_t.

The idea of particle filtering is to approximate the probability distribution of the object state by a weighted sample set. Each sample is an element which represents the hypothetical state s of the object, associated with a weight π. The sample set can be written as: $S = \left\{\left(s^{(i)}, \pi^{(i)}\right), i = 1, ..., n\right\}$ where $\sum_{i=1}^{n} \pi^{(i)} = 1$

The evolution of a sample set is given by propagating each sample according to a system model (here a motion model, see below). Each element of the set is then weighted in terms of the observations, and the mean state of the object is estimated at each step as:

$$E\left[S\right] = \sum_{i=1}^{n} \pi^{(i)} s^{(i)} \tag{1}$$

One of the advantages of particle filtering is that it models uncertainty, thus making it more robust in case of occlusion or clutter.

2.2. Color histogram as a model

As said previously, we use color distributions to model our target because of their robustness to rotation, partial occlusion and non rigidity of the target. Suppose the distributions are discretized into K bins (see Section 3.1 for the automatic selection of K). In our approach, we model the target by an ellipsis. The histograms can be calculated in RGB or any other color space, or simply in grey level space, depending on the input sequence. To partially cope with the loss of spatial information when using histograms, Nummiaro [1] and Pérez [2] assigned different weights to the pixels of the ellipsis to increase the reliability of the color distributions; smaller weigths are given to the pixels far away from the ellipsis center, using the following weighting function:

$$k(x) = \begin{cases} 1 - x^2 & \text{if } x \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

where x is the distance from the ellipsis center. Note that other weighting functions can be used: Comaniciu, for example in [5], uses the Epanechnikov kernel.

A color or greylevel distribution $p_x = \{p_x^{(j)}\}_{j=1,...,K}$ at a location x is calculated as

$$p_x^{(j)} = C \sum_{x_i \in E} k\left(\frac{\|x - x_i\|}{\sqrt{(l_x^2 + l_y^2)}}\right) \delta\left(h(x_i) - j\right) \tag{2}$$

where δ is the Kronecker delta function, E is the set of pixels in the ellipsis, l_x and l_y are the ellipsis half lengths, $h(x_i)$ assigns one of the K bins of the histogram to a given color at location x_i and C is the normalization factor, which ensures that $\sum_{j=1}^{K} p_x^{(j)} = 1$. The expression of C is given by

$$C = \left[\sum_{x_i \in E} k\left(\frac{\|x - x_i\|}{\sqrt{(l_x^2 + l_y^2)}}\right)\right]^{-1}$$

The similarity of two distributions p and q is measured by the Bhattacharyya coefficient

$$\rho\left[p, q\right] = \sum_{j=1}^{K} \sqrt{p^{(j)} q^{(j)}} \tag{3}$$

For two identical distributions, we have $\rho = 1$, which corresponds to a perfect match. We use the Bhattacharyya distance $d = \sqrt{1 - \rho\left[p, q\right]}$ in our algorithm.

2.3. How to combine color histograms and particle filtering?

We want to track a patch of interest in the image plane. We choose to parameterize this patch by an ellipsis

$$s = \left\{x, y, \dot{x}, \dot{y}, \theta, l_x, l_y, \dot{l}_x, \dot{l}_y\right\}$$

where x and y represent the ellipsis center, $\dot{x}$ and $\dot{y}$ the velocities of the center, θ the ellipsis orientation, l_x and l_y the lengths of the ellipsis half axes, and $\dot{l}_x$ and $\dot{l}_y$ the velocities of l_x and l_y. This model is flexible in that the ellipsis parameters can vary independently.

To propagate the sample set, we use a first order model given by

$$s_t = A s_{t-1} + b_{t-1} \tag{4}$$

where b_t is a multivariate Gaussian random variable and A is a matrix designed in order to describe an object moving with constant velocity for x, y, l_x and l_x.

The tracker works as follows: in the first image the model distribution is calculated, and the set of particles is initialized. Then, for each image of the input sequence, we propagate the set of particles using the dynamic model previously defined. For each sample of the set, the Bhattacharyya distance between the model distribution and the sample distribution is calculated and used to compute the weight π of the sample. The weight associated to each particle of the set favors samples whose color distributions are similar to the target model. The weights are calculated using

$$\pi^{(i)} = \gamma \exp(-\beta d^{(i)}) \tag{5}$$

for each particle i of the set, where γ and β are some fixed constants and $d^{(i)}$ represents the Bhattacharyya distance between the i^{th} particle and the target model.

The last step is to re-sample the particles to ensure the efficiency of the evolution, and to determine the mean state of the object. During the re-sampling step, the particles

are eliminated or duplicated according to their weight: the higher the weight of a particle, the more likely it is to be duplicated. Different methods exist, we chose to use a systematic re-sampling [9]. It consists in dividing the interval $[0, 1]$ into n segments. Then a uniform random variable U is generated on $\left[0, \frac{1}{n}\right]$; we define $U_1 = U$ and $U_i = U_{i-1} + \frac{1}{n}$ for $i = 2, .., n$. If U_i belongs to the j^{th} segment, then we pose $\Xi^i = X^j$, where Ξ^i is an element of the new sample set. The advantage of this method is that only $O(n)$ comparison tests are needed to produce the new sample set.

3. Our contributions

Many approaches have been proposed for tracking objects (with or without shape deformations) based on color histograms integrated in a probabilistic framework as described in the last section. But none of them are flexible enough to automatically regulate all the parameters in order to make them easily usable for different applications. We propose in this section some criteria that allow to automatically determine the tracking parameters.

3.1. The appropriate number of bins

The number of bins in our histograms is a crucial parameter and should be determined automatically. Too many bins in a histogram do not cope with changes in illumination or in the model appearence and most of the time the algorithm drifts away from the target. On the opposite, too few bins do not allow a good discrimination of the target, and the tracking fails. This evidence has been confirmed by our experiments, as shown in Section 4.

In most of the existing approaches, the number of bins seems to be chosen arbitrarily and kept fixed during the tracking. Nothing indicates that such a partition is optimal given the n-sample density we want to estimate. If we could find the optimal partition, the tracker should be more robust.

There have been many attempts in the past to solve the problem of determining the optimal number of bins from the data. Generally these methods are based on some asymptotic considerations. The problem with these approaches is that they do not perform very well in the case of small sample sizes due to their asymptotic nature. Moreover, many of them assume some prior information about the density. Recently, Birgé and Rozenholc [10] have generalized Akaike's estimator. Akaike's theorem is a statistical measure for model selection which states that if two models fit the data equally well, the simpler model will usually predict better. In the following, we briefly summarize their method of determining the optimal number of bins of our histograms. For the theoretical arguments underlying it, refer to [10].

The purpose is to find a histogram estimator $\hat{f}$ based on some partition $\{I_1, ..., I_K\}$ of $[0, 1]$ into K intervals of equal length. $X_1, X_2, ..., X_n$ are n samples from the unknown density f we want to estimate. K is given by

$$K = \arg\max_K (L_n(K) - penalty(K)) \tag{6}$$

where $L_n(K)$ is the log-likelihood of the histogram with K bins, given by

$$L_n(K) = \sum_{j=1}^{K} M_j \log(\frac{KM_j}{n}) \quad \text{with} \quad M_j = \sum_{i=1}^{n} \mathbf{1}_{I_j}(X_i)$$

where $\mathbf{1}_{I_j}$ is the indicator function defined by

$$\mathbf{1}_{I_j}(x) = \begin{cases} 1 & \text{if } x \in I_j \\ 0 & \text{otherwise} \end{cases}$$

The penalty function is given by

$$penalty(K) = K - 1 + (log(K))^{2.5} \quad \text{for } K \geq 1$$

This approach is thus a typical example of model selection methods, making a compromise between the complexity of the model and its fidelity to the data.

3.2. Incorporating spatial information

The problem with the use of histograms is that all spatial information is lost, as opposed to templates, which use the whole spatial information. As said previously, assigning different weights to the pixels of the ellipsis according to their distance to the center allows to integrate some spatial information. We wanted our tracker to work with more spatial information. We decided to divide our ellipsis in four quarters, and proceed as previously explained for each one of the ellipsis quarters. This division of the ellipsis increases the robustness of the tracker since we have four measures of similarity between a hypothesis and the model that can be combined easily and allows a better discrimination between the object and the rest of the scene.

Another advantage of dividing our ellipsis is to use the criterion for the automatic selection of the number of bins for each one of the quarters. The number of bins can be different in each quarter of the ellipsis according to the amount of data available (an ellipsis quadrant containing an homogenous region does not need as many bins as a highly textured region).

Finally, the division of the ellipsis into quarters makes it easier to handle or detect partial occlusions: if the Bhattacharyya coefficient is bad for one of the quarters but very good for the others then a partial occlusion is detected. To determine the mean state of the object, we combine the four measures by calculating their median value.

3.3. The model update

The apparent color of an object can vary over time due to changes in illumination, in camera parameters or in object

motion. To deal with these appearance changes, the model has to be updated. Particle filtering has already been used with static [8] or adaptive [6] models. Most of the time, the model is updated for each frame where the probability of the mean state is above a threshold fixed arbitrarily at the initialization. The risk with this method is to gradually drift from the target.

The idea we propose is the following: why should we update the model when it is still good? The model should only be updated when its appearance changes too much. So we use the following criterion: if the mean state is under a threshold π_T (see next paragraph for its setup), then we update the model using the following equation:

$$p_{E[S_t]} \leq \pi_T \Rightarrow q_t^j = (1-\alpha)q_{t-1}^j + \alpha p_{E[S_t]}^j \qquad (7)$$

Setting up of the threshold π_T: This threshold depends on different parameters more or less connected: the ellipsis size and the number of bins of the histograms. In fact, the larger the ellipsis is, the larger in general the number of bins will be. This results in a lower Bhattacharyya coefficient and the threshold for the model update should be lower too. To set it up automatically, we make the hypothesis that in the beginning of the sequence (the first images) the mean state of the object is well estimated and we set the threshold empirically: $\pi_T = \rho - c$ where c is a fixed constant.

The global scheme of the algorithm is given on Figure 1.

1. Initialization:
   - Selection of the ellipsis in the first image
   - Automatic determination of the number of bins of each histogram according to equation (6)
   - Computation of the model distribution in each ellipsis quarter with equation (2)
   - Initialization of the sample set
2. For each new image:
   - Propagate the sample set using the dynamic model according to equation (4)
   - For each sample of the set S_t, compute
     - The color distribution with equation (2)
     - The Bhattacharyya coefficients with (3)
     - The weights with equation (5)
   - Estimate the mean state according to equation (1)
   - Update of the target model if necessary with (7)
   - Re-sampling step (see Section 2.3)

Figure 1: The algorithm.

4. Experiments

This section evaluates the performances of our tracker. We first show how performant the automatic selection of the number of bins is, then we present the experimental results of our tracker on various sequences. The description of the sequences is made in Section 4.2.

4.1. Evalutation of our contributions

To give experimental evidence that our contributions lead to better tracking, we ran our algorithm several times (15) with the automatic selection of the numbers of bins and with fixed number of bins in the case of the tracking on the entire ellipsis. We use the mean value for our results, presented in Table 1; the numbers indicate the image number from which the tracking fails and bold numbers correspond to the results for the automatically chosen number of bins. To measure that the tracking fails, we established a "ground truth" of each one of the sequences by keeping in a file the coordinates of a good tracking. Tracking was then declared as failed if for one image, the size of the ellipse is too large or if the ellipsis center is too far away from the ground truth. The results show that fixing arbitrarily a number of bin can't be a good solution for tracking various objects. But the automatic selection leads most of the time to the optimal tracking.

# of bins	Football	Indoor	License plate
2	16	19	31
4	87	23	46
6	107	111	55
8	109	125	41
10	95	172	60
12	118	169	94
14	90	**168**	92
16	121	164	84
18	**117**	132	**93**
20	111	217	61
22	75	108	78
24	88	135	62
26	72	90	63
28	86	107	73
30	76	78	74
32	69	85	82
34	102	70	61
36	92	92	86
38	102	76	69
40	87	59	91

Table 1: Performances of the tracker using the entire ellipsis with various numbers of bins.

Figure 2 presents the results obtained for each sequence with the automatic selection of the number of bins, in the case of a tracking with the entire ellipsis or the 4 quadrants. The graphs shows that using 4 quadrants improve largely the performances of our tracking.

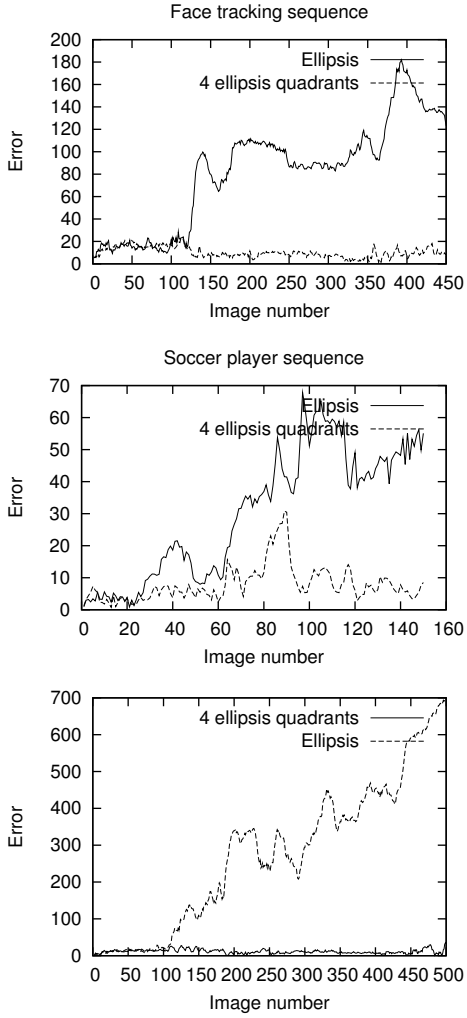


Figure 2: Influence of the ellipsis division into quadrants for the 3 sequences.

4.2. Tracking results for different sequences

We tested our tracker on different types of input: the results show that our approach allows us to use it efficiently for various applications independently of parameter initialization. Three sets of results are provided; in each experiment grey level sequences are used.

1. *Face tracking*: The sequence presents a person entering and moving around a room (http://www.ee.oulu.fi/~mikak/tracking/FaceColor.html). There are some important appearance changes since the person turns around 360° during the sequence. The results show that the way to update the model in order to cope with appearance changes of the target is efficient. Furthermore, the algorithm is able to track even in the cases where the person moves with changes in speed.

Figure 3: Face tracking results: images 1, 140, 395, 412 and 449 of the sequence.

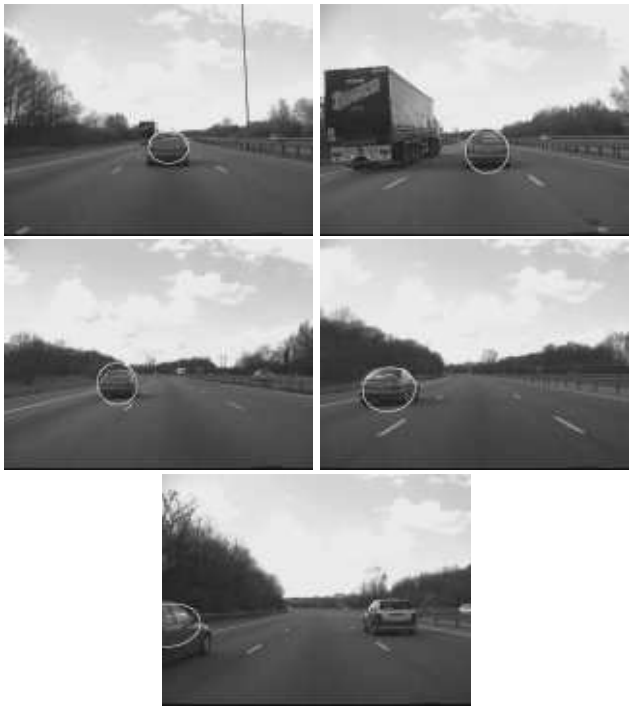


Figure 4: Car tracking results: images 1, 187, 347, 462 and 563 of the sequence.

Figure 5: License plate tracking results: images 1, 187, 347, 417, 520 of the sequence, and a zoom made on image 417.

3. *Car tracking*: The sequence is one of the PETS 2001 database (ftp://pets2001.cs.rdg.ac.uk/) in the context of a driver assistance application. We made two experiments, the results are presented in Figures 4 and 5: the particle filter has to deal with rapid movements of the target and the camera. The difficulties of this sequence are the different viewing angles of the tracked car, the changes in scale and the out of plane rotations; but we can see the good performance of our algorithm during the whole sequence. For the first experiment we tracked the entire car, achieving equivalent results to [1]. For the second one, we only tracked the rear license plate of the car. The additional difficulty of this experiment is the small size of the object to be tracked. The results presented in Figure 5 show that our tracker is able to track efficiently even small objects.

4. *Football player tracking*: The sequence is taken from a football match; the difficulties of this sequence are the fast motions of the players and the occasional occlusion of some players by others players on the ground. In frame 78, a player is falling down and another player attempts to catch the ball so a player is entirely occluded. Our tracker remains efficient even in this case.

5. Conclusions and discussion

These results suggest that our system is able to track:

- an object with large appearance changes such as shape and/or orientation changes.
- an object in a scene with scale changes.
- an object that moves with varying velocity as illustrated by the football player sequence.
- a deformable object (the football player for instance).

We also tested our algorithm on aerial sequences; the results show the robustness of our tracker for various applications.

The proposed tracker adds a criterion which allows to detect the optimal number of bins needed for the histograms in order to achieve robust tracking of various objects. Moreover we set up rules allowing the algorithm to work automatically for various applications. Our approach is a step towards a fully automatic and adaptative tracking. The tracking algorithm is based on color distributions integrated in a probabilistic framework. The experiments show that the tracker is robust to partial and complete occlusions, to appearance changes of the target as well as changes in illumination of the scene. Furthermore, the division of the ellipsis into smaller regions increases the robustness of the tracker.

Notice that we divided the ellipsis into quarters; it would be interesting to set up a criterion similar to the one used to select the number of bins in order to find the best compromise between the amount of spatial information in the model and the flexibility of histograms. Furthermore, we think about updating the number of bins during the sequence if appropriate (for example if the target grows in the image, updating the model with more bins in order to profit from the increasing of information). Also, the model update using the color distributions at the current mean state is expected to run into problems if a similar looking object is nearby; we work at solutions for this problem.

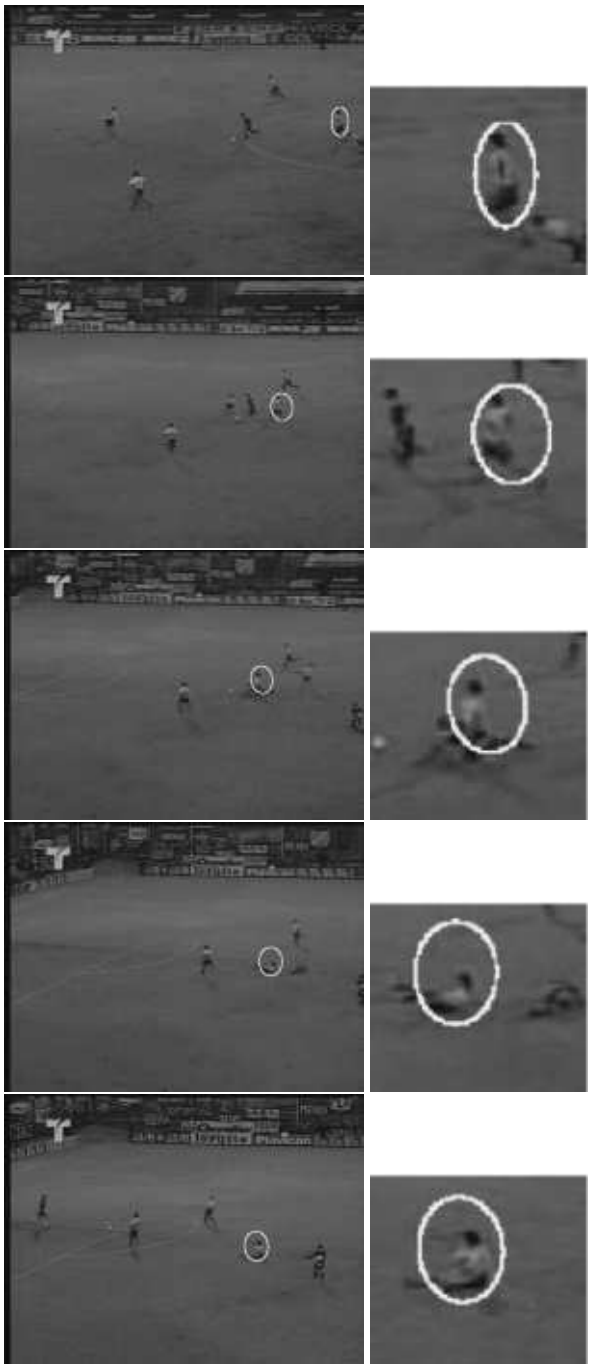


Figure 6: Football player tracking results: images 1, 55, 76, 97 and 141 of the sequence. In frame 76, the white player occludes the black one.

References

[1] K. Nummiaro, E. Koller-Meier, L. Van Gool, "Color Features for Tracking Non-Rigid Objects", *ACTA Automatica Sinica*, 2003.

[2] P. Pérez, C. Hue, J. Vermaak, M. Gangnet, "Color-based probabilistic tracking", *In European Conference on Computer Vision*, LNCS 2350, pp 661-675, Copenhagen, Denmark,2002.

[3] A.D. Jepson, D.J. Fleet, T.F. El-Maraghi, "Robust Online Appearance Models for Visual Tracking", *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, Vol I, pp 415-422, 2001.

[4] G.R. Bradski, "Computer vision face tracking as a component of a perceptual user interface", *In Workshop on Applications of Computer Vision*, pp 214-219, Princeton, NJ, 1998.

[5] D. Comaniciu, V. Ramesh, P. Meer, "Kernel-Based Object Tracking", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol 25, no 5, 2003.

[6] K. Nummiaro, E. Koller-Meier, L. Van Gool, "Object Tracking with an Adaptative Color-Based Particle Filter", *Image and Vision Computing*, 2002.

[7] Z. Zivkovic, B. Krose, "An EM-like algorithm for color-histogram-based object tracking", *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, 2004.

[8] T. Heap, D. Hogg, "Wormholes in Shape Space: Tracking through Discontinuous Changes in Shape", *Sixth International Conference on Computer Vision*, Bombay, India, 1998.

[9] F. Legland, "Filtrage particulaire", *Proceedings 19ème Colloque GRETSI sur le Traitement du Signal et des Images*, VolI, pp 1-8, Paris, 2003.

[10] L. Birgé, Y. Rozenholc, "How many bins should be put in a regular histogram", *Technological Report, Laboratoire Probabilités et Modèles Aléatoires, Université Pierre et Marie Curie*, Paris, France, PMA-721, 2002.

Chapter 16

Model Selection for Two-View Geometry

Paper 39 [15]: S.J. Maybank and P. Sturm. MDL, collineations and the fundamental matrix. In T. Pridmore and D. Elliman, editors, *Proceedings of the 10th British Machine Vision Conference, Nottingham, England*, pages 53–62. British Machine Vision Association, September 1999.

MDL, Collineations and the Fundamental Matrix

S.J. Maybank and P.F. Sturm, Department of Computer Science, The University or Reading, Whiteknights, Reading, Berkshire, RG6 6AY, UK.

(S.J.Maybank,P.F.Sturm)@reading.ac.uk

Abstract

Scene geometry can be inferred from point correspondences between two images. The inference process includes the selection of a model. Four models are considered: background (or null), collineation, affine fundamental matrix and fundamental matrix. It is shown how Minimum Description Length (MDL) can be used to compare the different models. The main result is that there is little reason for preferring the fundamental matrix model over the collineation model, even when the former the 'true' model.

1 Introduction

Model selection is a central task in computer vision: given data obtained from images and given a number of models, which model is most strongly supported by the data? Is it better to have *i)* a simple model fitting the data approximately; or *ii)* a complicated model fitting the data very closely [1, 6, 7, 9, 10, 15, 18, 20]? Accuracy of fit to the data is by itself not a sufficient criterion for choosing a model. The fit can always be improved by allowing a greater flexibility or generality in the model. In many cases, a sufficiently general model fits the data with zero error.

In simple cases, the allowable models are specified by probability density functions $p(x|\theta)$ defined on the data x and depending on a parameter vector θ with a fixed dimension. In such cases the Maximum Likelihood (ML) principle is a good, widely used strategy: given x, select the value of θ at which $p(x|\theta)$ attains its maximum value. The ML principle fails if the dimension of θ can vary. Tt is necessary to introduce a penalty for the number of model parameters, otherwise a model with a large number of parameters will always be chosen in preference to models with only a few parameters.

In the literature there are several suggestions for penalising overparameterisation, for example [1, 18]. The Bayesian Information Criterion (BIC) of [18] applies if the allowable models can be divided into separate families such that the ML principle holds for each family separately. The BIC yields for each family an error criterion of the form $\log(p(x|\hat{\theta})) - (1/2)D\log(N)$ where $\hat{\theta}$ is the maximum likelihood estimate of θ for the family, D is the total number of parameters in the model and N is the total number of measurements. If the number of measurements is low, then the second term dominates and models with low D are favoured. If the number of measurements is high, then the first term dominates, because it depends on the fit of the model to the large number of measurements, and the value of D is less important.

The BIC is applicable only if probability density functions are defined on the space of possible errors in the measurements and on the space of parameter values for each family

of models. These density functions are prior information which can strongly affect the model choice.

Minimum Description Length (MDL) [9, 10] is an alternative way of comparing models. Unlike Bayesian methods it does not require explicit probability density functions for the data and the parameters. In MDL the data are first expressed as a bit string s. If a model $\mathcal{M}$ fits the data, then s contains internal structure depending on $\mathcal{M}$. This structure is removed, to give a compressed string. The compressed string and a bit string description of $\mathcal{M}$ are concatenated to give a string $U_{\mathcal{M}}(s)$. The model $\mathcal{M}$ is strongly favoured if $U_{\mathcal{M}}(s)$ is much shorter than s. The compression must be information preserving, in that s can be recovered exactly from $U_{\mathcal{M}}(s)$. Overparameterised models are penalised simply because they require a long description.

The Shannon-Fano code [10] is an example of MDL. The code is for symbols supplied randomly and independently. If a symbol a has probability p then the optimal 0,1 code for a has length $\lceil -\log(p) \rceil$, where log is the logarithm to base 2. The key point here is the close link between the model (a supplied with probability p) and the code length.

Applications of MDL to computer vision can be found in [3, 5, 8, 11, 13, 14]. In many cases, for example [8, 11], MDL is used to assign a prior probability to a model. The deviations of the data from the model are given probabilities assuming a standard distribution such as the Gaussian. The probability of a model conditional on the data is derived using Bayes' theorem, and then maximised over the model parameters.

In this paper MDL is applied to sets of pairs of corresponding points obtained from images of a room taken by a digital camcorder. The aim is to investigate a test case application of MDL, and to see if it performs as expected. The models are $\mathcal{B}$, $\mathcal{C}$, $\mathcal{A}$, $\mathcal{F}$, as listed in §1.1. No explicit probabilistic assumptions are made. The only criterion for comparing the models is the length of the compressed bit strings $U_{\mathcal{B}}(s)$, $U_{\mathcal{C}}(s)$, $U_{\mathcal{A}}(s)$, $U_{\mathcal{F}}(s)$. The main new result is that $\mathcal{C}$ achieves a good compression in all cases, in particular,

$$|U_{\mathcal{F}}(s)| \approx |U_{\mathcal{C}}(s)| \text{ when the 'true' model is } \mathcal{F}. \tag{1}$$

Without more information, for example, additional images or constraints on the shapes of objects, there is little reason for ever preferring $\mathcal{F}$ to $\mathcal{C}$.

1.1 The models

To be specific, suppose the data are the pixel coordinates of corresponding points $q_i \leftrightarrow q_i'$, $1 \leq i \leq n$ in two images of the same scene. Two points q, q' in different images correspond if and only if they are projections of the same scene point [4]. The images are embedded in the projective plane, $\mathbb{P}^2$, by adding 1 as a third coordinate, $(x, y) \mapsto (x, y, 1)$. There are many possible models, each of which involves assumptions about the relative position of the two cameras or the scene geometry [18, 19]. In this paper the following models are considered.

- $\mathcal{B}$) Background: the image points have no discernable structure.
- $\mathcal{C}$) Collineation: there is a collineation $\omega : \mathbb{P}^2 \to \mathbb{P}^2$ such that $\omega(q_i) = q_i'$, $1 \leq i \leq n$.
- $\mathcal{A}$) Affine fundamental: there is a 3×3 matrix A with rank 2 such that $A_{11} = A_{12} = A_{21} = A_{22} = 0$ and $q_i^\top A q_i' = 0$, $1 \leq i \leq n$ [17].
- $\mathcal{F}$) Fundamental: there is a 3×3 matrix F with rank 2 such that $q_i^\top F q_i' = 0$, $1 \leq i \leq n$.

1.2 Notation

A bit string is an element of $\{0,1\}^*$. The length of a bit string s is $|s|$. The floor function is $x \mapsto \lfloor x \rfloor$ and the ceiling function is $x \mapsto \lceil x \rceil$, where $\lfloor x \rfloor$ is the greatest integer such that $\lfloor x \rfloor \leq x$, and $\lceil x \rceil$ is the least integer such that $x \leq \lceil x \rceil$.

The fixed length code of length b for a non-negative integer n is $d(n,b)$. The usual binary code for n is padded with zeros on the right to give a bit string of length b. If $n \geq 1$, then $d(n,b)$ is defined only if $b \geq \lceil \log(n) \rceil + 1$, where log is the logarithm to base 2.

The logstar prefix code $r : \mathbb{N}^+ \to \{0,1\}^*$ is implemented as described in [2], §4.3.2. Note that $|r(n)| \approx \log^*(n) + 1.51857\ldots$. A general discussion of the logstar code, including the definition of $\log^*$, can be found in [10], with further information in [2]. The map zton : $Z \to \mathbb{N}^+$ is $\text{zton}(n) = 2n$ for $n \geq 1$, and $\text{zton}(n) = 2\,\text{abs}(n) + 1$ for $n < 1$, and the logstar code is redefined on Z by $e(n) = r(\text{zton}(n))$.

2 Coding the Data

In this section the strings $U_{\mathcal{B}}(s)$, $U_{\mathcal{C}}(s)$, $U_{\mathcal{A}}(s)$, $U_{\mathcal{F}}(s)$ are defined. All the codes are constructed using rational arithmetic, in order to avoid inaccuracies arising from floating point approximations. The algorithms are implemented in Mathematica [21].

The image points are defined for $1 \leq i \leq n$ by $q_i = (x_i, y_i, 1)$, $q_i' = (x_i', y_i', 1)$ where x_i, y_i, x_i', y_i' are integers. If the feature points are located in each image to an accuracy of 1 pixel then the x_i, y_i etc. are the pixel coordinates. If feature points are located with subpixel accuracy, then the x_i, y_i etc. are scaled pixel coordinates. The code $c : Z^n \to \{0,1\}^*$ used in this section is defined below in §3.

2.1 Background $\mathcal{B}$

Let $x, y, x', y' \in Z^n$ be the vectors with respective components x_i, y_i, x_i', y_i'. The code for $q_i \leftrightarrow q_i'$, $1 \leq i \leq n$ under the background model $\mathcal{B}$ is $U_{\mathcal{B}}(s) = c(x).c(y).c(x').c(y')$.

2.2 Collineation $\mathcal{C}$

Let $\omega : \mathbb{P}^2 \to \mathbb{P}^2$ be a collineation, ie. a map of the the form $q \mapsto Hq$, where H is an invertible 3×3 matrix, and let a_i, b_i, ϵ_i, δ_i be defined for $1 \leq i \leq n$ by

$$\begin{aligned} \omega(q_i) &= (a_i, b_i, 1)^\top \\ (\epsilon_i, \delta_i) &= (\lfloor x_i' - a_i + 0.5 \rfloor, \lfloor y_i' - b_i + 0.5 \rfloor) \end{aligned}$$

The point q_i' can be recovered exactly from ω, q_i and the integers ϵ_i, δ_i.

Let $\epsilon, \delta \in Z^n$ be the vectors with respective coordinates ϵ_i, δ_i and let code(ω) be a coding of ω. The q_i, q_i', $1 \leq i \leq n$ are coded by the string

$$t = c(x).c(y).\text{code}(\omega).c(\epsilon).c(\delta) \tag{2}$$

If ω is a good fit to the data, then ϵ, δ are small and compression is achieved.

How should code(ω) be constructed? A key issue is the precision with which the components of ω are specified. If the precision is low, then $|\text{code}(\omega)|$ is small but $|c(\epsilon).c(\delta)|$ is

large. If the precision is high, then $|\text{code}(\omega)|$ is large but $|c(\epsilon).c(\delta)|$ is small. The problem of choosing the best precision is circumvented by using RANSAC [16].

Let $u_i \leftrightarrow v_i$, $1 \leq i \leq 4$ be pairs of corresponding points in $\mathbb{P}^2$, such that no three of the u_i are collinear and no three of the v_i are collinear. There is a unique collineation ω such that $\omega(u_i) = v_i$, $1 \leq i \leq 4$. A coding of the u_i, v_i, $1 \leq i \leq 4$ yields a coding of ω. Ideally, all quadruples $q_{i_j} \leftrightarrow q'_{i_j}$, $1 \leq j \leq 4$ of corresponding points should be examined to find the quadruple for which

$$|\text{code}(\omega).c(\epsilon).c(\delta)| \tag{3}$$

is a minimum. In practice there are too many quadruples, so a random selection of N quadruples is made and (3) is minimised over the chosen quadruples.

An advantage of RANSAC is that the precision of ω is appropriate for the data; in addition, the code for ω is redundant because it includes the points q_{i_j}, $1 \leq j \leq 4$ already coded in $c(x).c(y)$. The redundancy is removed and compression achieved by omitting the q_{i_j} from code(ω), and instead coding the index of the four-tuple $\iota = (i_1, i_2, i_3, i_4)$, $i_1 < i_2 < i_3 < i_4$ in the list of all ordered four-tuples with distinct entries drawn from n. The code length for ι is at most $\lceil \log(b(n,4)) \rceil + 1$ bits where $b(n,4)$ is the binomial coefficient.

Further compression of t in (2) is achieved by omitting from ϵ, δ the eight entries known to be zero, yielding the code $U_{\mathcal{C}}(s)$.

2.3 Affine fundamental matrix A

Let A be an affine fundamental matrix, and let l be the line $l' = q^\top A$. The geometrical interpretation of the equation $q^\top A q' = 0$ is that q' lies on l'. If q, A are given, then q' can be coded by giving its position on l'. Compression is achieved because only one coordinate is needed rather than two.

As with $\mathcal{C}$, RANSAC is used to find a suitable matrix A compatible with the $q_i \leftrightarrow q'_i$, $1 \leq i \leq n$. Let $u_i \leftrightarrow v_i$, $1 \leq i \leq 4$ be pairs of corresponding points in $\mathbb{P}^2$ such that no three of the u_i are collinear, no three of the v_i are collinear, none of the u_i, v_i are on the line at infinity and there is no affine transformation T such that $Tu_i = v_i$, $1 \leq i \leq 4$. Then there is a unique affine fundamental matrix A such that $u_i^\top A v_i = 0$, $1 \leq i \leq 4$.

The point q'_i is specified relative to an origin which depends on i, the q_j and A. In detail, there is a three dimensional family of collineations which preserve the epipolar lines associated with A in that if ρ is any one of the collineations and l is an epipolar line of A in the first image, then $\rho(l)$ is the corresponding epipolar line in the second image [12]. The three dimensional family is spanned the collineations associated with any four linearly independent matrices H for which

$$AH + H^\top A^\top = 0 \tag{4}$$

Let $q_{i_j} \leftrightarrow q'_{i_j}$, $1 \leq j \leq 4$ be the pairs of corresponding points which define A. From the q_{i_j} select the three points q_{i_j}, q_{i_k}, q_{i_l} which define a triangle with the greatest area. A unique collineation is specified by the matrix H for which $Hq_{i_j} = q'_{i_j}$, $Hq_{i_k} = q'_{i_k}$, $Hq_{i_l} = q'_{i_l}$ and (4) holds.

Let ν_i be a unit vector in $\mathbb{R}^2$ parallel to $q_i^\top A$, let $\nu_i^\perp$ be a unit vector perpendicular to ν_i, and define r_i, s_i by $q'_i - Hq_i = (r_i \nu_i + s_i \nu_i^\perp, 1)$ as shown in Figure 1. Define integers

ϵ_i, δ_i by

$$(\epsilon_i, \delta_i) = (\lfloor 2r_i + 0.5 \rfloor, \lfloor 2s_i + 0.5 \rfloor) \tag{5}$$

The factor 2 on the right hand side of (5) is needed to remove quantisation errors.

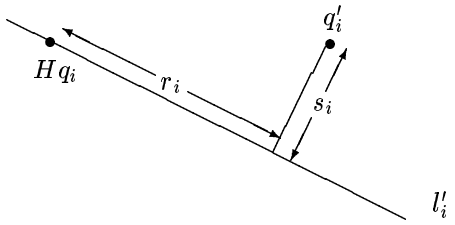


Figure 1. Definition of r_i, s_i.

The code for the $q_i \leftrightarrow q_i'$, $1 \leq i \leq n$ is

$$c(x).c(y).\text{code}(A).c(\epsilon).c(\delta) \tag{6}$$

The matrix A is specified by giving the index of $\iota = (i_1, i_2, i_3, i_4)$ in the list of ordered four-tuples of distinct elements drawn from n. When coding δ, the four entries known to be zero are omitted.

A random selection of N quadruples is made and the length of the code (6) minimised over the quadruples. The code with the minimum length is $U_{\mathcal{A}}(s)$.

In a full reconstruction of the 3D scene the collineation H described above defines a plane in space which passes near to the 3D points projecting down to corresponding points in the two images.

2.4 Fundamental matrix $\mathcal{F}$

The coding of s as $U_{\mathcal{F}}(s)$ is similar to the coding as $U_{\mathcal{A}}(s)$, with one significant change, due to the fact that four pairs of image correspondences are not sufficient to specify a unique fundamental matrix. Let $q_{i_j} \leftrightarrow q'_{i_j}$, $1 \leq j \leq 7$ be seven pairs of corresponding points. There are in general exactly two linearly independent 3×3 matrices F_1, F_2 such that $q_{i_j}^\top F q'_{i_j} = 0$, $1 \leq j \leq 7$. The fundamental matrices compatible with the $q_{i_j} \leftrightarrow q'_{i_j}$, $1 \leq i \leq 7$ are obtained by solving the cubic polynomial equation in t [19],

$$\det(F_1 + tF_2) = 0 \tag{7}$$

There are at most three real roots. To specify a unique fundamental matrix it is necessary to record the appropriate root of (7), which requires two bits.

Let t_j be a real root of (7) and let $\tilde{F} = F_1 + t_j F_2$. The matrix $\tilde{F}$ is replaced by a rational approximation F, retaining the constraint $\det(F) = 0$. Let $\tilde{u}$ be the eigenvector of $\tilde{F}^\top \tilde{F}$ with the least eigenvalue, let u be a rational approximation to $\tilde{u}$ and let G be a rational approximation to $\tilde{F}$. The matrix F is defined by

$$F = G - (u.u)^{-1} Gu \otimes u$$

3 Code for Vectors in Z^n

If $c_1, \ldots, c_p$ are different codes for vectors $x \in Z^n$, then a new code c can be constructed by first finding the index j such that

$$|c_j(x)| = \min\{|c_i(x)|, 1 \leq i \leq p\}$$

and then setting $c(x) = d(j,b).c_j(x)$, where d is defined in §1.4. If p is small and n is large, then c may give shorter average code lengths than any single code c_i.

The code c in §2 is constructed from four separate codes c_1, c_2, c_3, c_4, which are described in turn.

3.1 Codes c_1 and c_2

Let C_σ be defined for $\sigma \in \mathbb{N}$ by

$$C_\sigma = \{x \in Z^n, |x_i| \leq \sigma, 1 \leq i \leq n\}$$

The set C_σ contains $(2\sigma+1)^n$ points. The elements of C_σ are enumerated by a function $\zeta_\sigma : C_\sigma \to \mathbb{N}^+$ constructed such that $\zeta_0(0) = 1$, and such that ζ_σ is an extension of $\zeta_{\sigma-1}$ for $\sigma \geq 1$. The functions ζ_σ, $\sigma \geq 0$ together define a single function $\zeta : Z^n \to \mathbb{N}^+$.

Let m be the median of the x_i and let v be the vector with components $v_i = x_i - m$, $1 \leq i \leq n$. The codes c_1, c_2 are defined by

$$c_1(x) = r(\zeta(x)) \qquad \text{and} \qquad c_2(x) = r(\text{zton}(m)).c_1(v)$$

3.2 Code c_3

Let $\sigma = \text{abs}(x_j)$ for some j, let u be the vector of components x_i such that $\text{abs}(x_i) \leq \sigma$, and let v be the vector of components $x_i - \text{sign}(x_i)\sigma$ for those i such that $|x_i| > \sigma$. Let $w_\sigma \in \{0,1\}^n$ be defined such that $(w_\sigma)_i = 1$ if x_i is a component of u and $(w_\sigma)_i = 0$ if $x_i - \text{sign}(x_i)\sigma$ is a component of v.

The code d_σ is defined by $d_\sigma(x) = w_\sigma.c_1(u).c_1(v)$. Let θ be the value of σ at which $|d_\sigma(x)|$ is a minimum over all the distinct values of $\sigma = \text{abs}(x_i)$, $1 \leq i \leq n$, that is $|d_\theta(x)| = \min_\sigma\{|d_\sigma(x)|\}$. The code c_3 is defined by $c_3(x) = d_\theta(x)$.

3.3 Code c_4

Let m be the median of x, let $y_1, \ldots, y_p$ be the distinct integers appearing in the set $\{x_i - m, 1 \leq i \leq n\}$, and let y_i occur k_i times, $1 \leq i \leq p$. Let S be the set of all permutations of x. The number b of elements of S is given by the multinomial $b = n!/(k_1! \ldots k_m!)$. The elements of S are ordered in any convenient way. Let ι be the index of x in the chosen order, let y, k be the vectors with the respective components y_i, k_i, and let m_k be median of k. The code c_4 is defined by

$$\begin{aligned} f(x) &= e(k_1 - m_k).e(y_1). \ldots .e(k_p - m_k).e(y_p) \\ c_4(x) &= e(m).e(m_k).f(x).d(\iota, b) \end{aligned}$$

If the x_i are independent realisations of a random variable and n is large, then $|c_4(x)|/n$ is, with a high probability, close to the shortest possible expected length of a code word.

See for example [10], §1.11.4. In practice the effectiveness of c_4 is reduced because of the extra code needed for y, k.

Figure 2. Images for collineation model.

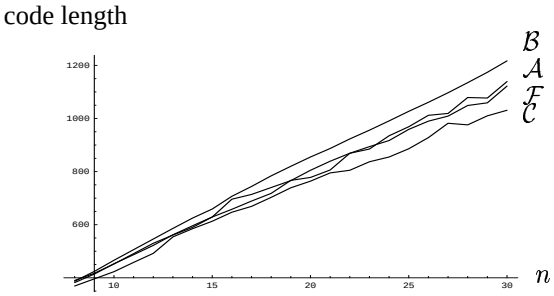


Figure 3. Code lengths when the 'true' model is a collineation

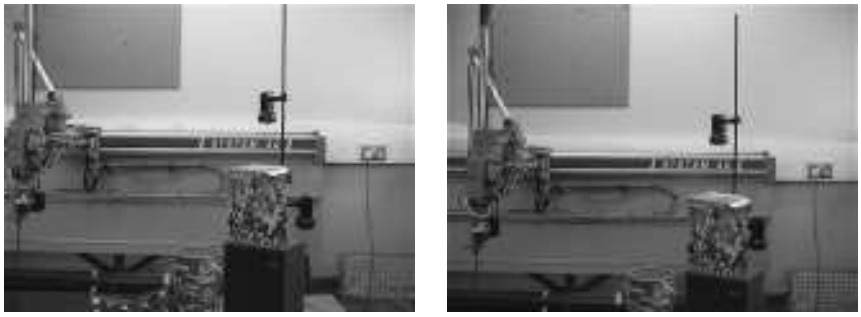


Figure 4. Images for affine fundamental matrix model.

4 Experiments

Images of a laboratory were taken by a Canon MV-1 Camcorder mounted on a tripod. Typical pairs of images are shown in Figures 2, 4, 6 with the 'true' models shown in the captions. In each case the 'true' model is known, but only because of the prior information available to a human observer. Figure 2 shows two images of a flat poster, Figure 4 is

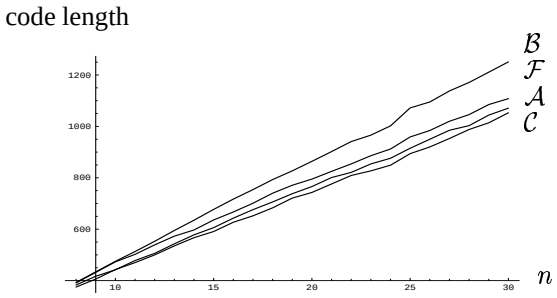


Figure 5. Code lengths when the 'true' model is an affine fundamental matrix

obtained by translating the camera parallel to the image plane, and in Figure 6 the camera is moved forwards to produce significant projective distortions of the image.

The task of the program is to make the best choice of model using only the data $q_i \leftrightarrow q_i'$, $1 \leq i \leq n$ and the models $\mathcal{B}, \mathcal{C}, \mathcal{A}, \mathcal{F}$. This best choice can and does differ from the 'true' model.

The size of the original images in pixels is 640×480. Feature points were located in each image and matched to obtain pairs of corresponding points $q_i \leftrightarrow q_i'$, $1 \leq i \leq n$. The graphs of code length against n for $8 \leq n \leq 30$ are shown in Figures 3, 5, 7. The maximum number on the vertical scale is 1200 bits, and the spacing between numbers is 200 bits. The number of random samples in the RANSAC algorithm was $N = 10$. Increasing the value of N as far as 20 did not produce significant changes in the graphs. Higher values were not investigated because of the long run times.

It is apparent from the graphs that $\mathcal{C}$ is always the preferred model even when the 'true' model is $\mathcal{F}$. The models $\mathcal{A}$, $\mathcal{F}$ show a similar performance, and $\mathcal{B}$ is always the worst model.

Figure 6. Images for fundamental matrix model.

5 Discussion

The experiments show that the collineation model $\mathcal{C}$ is a good choice even for sets of image correspondences for which the 'true' model is a fundamental matrix. This is in agreement with the comment in [18], Section 4, that for two images, simple models are strongly favoured over more complex ones. Why does the model $\mathcal{F}$ perform so badly

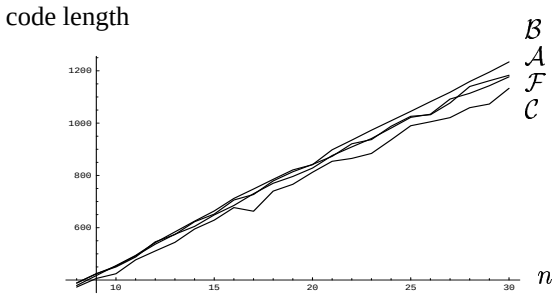


Figure 7. Code lengths when the 'true' model is a fundamental matrix

under MDL? The reason can be seen in Figure 1. In the usual methods for assessing the fit of $\mathcal{F}$ to the data, the error measure is the sum of the squares of the s_i, and the r_i, measuring distances along the epipolar lines, are ignored. In MDL the r_i must be included, to obtain a loss free coding of the data. The extra code length needed for the r_i reduces the preference for $\mathcal{F}$, so that in these experiments $\mathcal{C}$ is almost always preferred. This argument suggests that $\mathcal{C}$ will still be preferred if all the image pixel values are used rather than than just the locations of a few salient points.

If additional information is given, then $\mathcal{F}$ may become the preferred model. For example, suppose that $\mathcal{C}$ is augmented by an assumption that $q_i' - \omega(q_i)$ is a realisation of a Gaussian random variable with a known covariance, and that $q_i' - \omega(q_i)$ is coded under this assumption. When the 'true' model is a fundamental matrix, then the code for $q_i' - \omega(q_i)$ will be long and $\mathcal{F}$ will be preferred to $\mathcal{C}$.

Simple parameter counting, in agreement with general arguments based on the BIC, suggest that for long image sequences a generalisation of $\mathcal{F}$ will be preferred over a model in which pairs of images are related by a collineation. For example, suppose that images 1,2,3 are given with fundamental matrices F_{12}, F_{23}, F_{13}. Let $q \leftrightarrow q' \leftrightarrow q''$ be a triple of corresponding points. Then q'' is determined by q, q' and F_{13}, F_{23}, because it is the intersection of the epipolar lines $q^\top F_{13}$ and $q'^\top F_{23}$. If the F_{ij} are replaced by collineations H_{ij} such that H_{ij} preserves the epipolar lines associated with F_{ij}, then there is no certain way of locating q'' given only q, q' and the H_{ij}. This suggests that the fundamental matrix model will yield a shorter code for the points in the third image.

References

[1] H. Akaike, "A new look at statistical model identification," *IEEE Trans. Automation and Control*, Vol. 19, pp. 716-723, 1974.

[2] R. Baxter, "Minimum message length inductive inference - theory and applications," *Phd Thesis, Department of Computer Science, Monash University, Aus.*, 1996.

[3] T. Darrell and A. Pentland, "Cooperative robust estimation using layers of support," *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 17, pp. 474-487, 1995.

[4] O.D. Faugeras, *Three-Dimensional Computer Vision: a geometric viewpoint*. MIT Press, 1993.

[5] P. Fua and A.J. Hanson, "An optimization framework for feature extraction," *Machine Vision and Applications*, Vol. 4, pp. 59-87, 1991.

[6] K. Kanatani, *Statistical Optimization for Geometric Computation: Theory and Practice*. Elsevier, 1996.

[7] K. Kanatani, "Comments on nonparametric segmentation of curves into various representations." *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 19, pp. 1391-1392, 1997.

[8] Y.G. Leclerc, "Constructing simple stable descriptions for image partitioning," *International Journal of Computer Vision*, Vol. 3, pp. 73-102, 1989.

[9] M. Li and P.M.B. Vitányi, "Inductive reasoning and Kolmogorov complexity," *J. of Computer and System Sciences*, Vol. 44, pp. 343-384, 1982.

[10] M. Li and P. M. B. Vitányi, *An Introduction to Kolmogorov Complexity and Its Applications*, Graduate Texts in Computer Science, Springer, 2nd edition, 1997.

[11] T. Lindeberg and M.-X. Li, "Segmentation and classification of edges using minimum description length approximation and complementary junction clues," *Computer Vision and Image Understanding*, Vol. 67, pp. 88-98, 1997.

[12] Q.-T. Luong and T. Viéville, "Canonic representations for the geometries of multiple projective views," In J.-O. Eklundh (ed.) *Computer Vision-ECCV'94, Vol I*, Lecture Notes in Computer Science, Vol. 800, pp. 589-599, Springer, 1994.

[13] S.J. Maybank and R. Fraile, "Minimum description length method for facet matching," *Proc. International Symposium on Multispectral Image Processing, ISMIP'98*, SPIE Vol. 3545, Wuhan, China, pp. 330-335, 1998.

[14] A. Pentland, "Part segmentation for object recognition," *Neural Computation*, Vol. 1, pp. 82-91, 1989.

[15] P.L. Rosin and G.A.W. West, "Response to Kanatani," *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 19, pp. 1393-1394, 1997.

[16] P.J. Rousseeuw, *Robust Regression and Outlier Detection*, John Wiley, 1987.

[17] L.S. Shapiro, A. Zisserman and M. Brady "Motion from point matches using affine epipolar geometry". In J.-O. Eklundh (ed.) *Computer Vision - ECCV'94, Vol. II*, Lecture Notes in Computer Science, Vol. 801, pp. 73-84, Springer, 1994.

[18] P. H. S. Torr and A. Zisserman, "Concerning Bayesian motion segmentation, model averaging, matching and the trifocal tensor," In H. Burkhardt and B. Neumann (eds.) *Computer Vision - ECCV'98, Vol. I*, Lecture Notes in Computer Science, Vol. 1406, pp. 511-527, Springer, 1998.

[19] P.H.S. Torr, A. Zisserman and S.J. Maybank, "Robust detection of degenerate configurations whilst estimating the fundamental matrix," *Computer Vision, Graphics, and Image Processing*, Vol. 71, pp. 312-333, 1998.

[20] C.S. Wallace and P.R. Freeman, "Estimation and inference by compact coding," *J. Royal Stat. Soc. Series B*, Vol. 49, pp. 240-265, 1987.

[21] S. Wolfram, *The Mathematica Book*, 3rd Edition, Cambridge University Press, Cambridge, 1996.

Bibliography

[1] A. Bartoli and P. Sturm. Constrained structure and motion from multiple uncalibrated views of a piecewise planar scene. *International Journal of Computer Vision*, 52(1):45–64, 2003.

[2] A. Bartoli and P. Sturm. The 3D line motion matrix and alignment of line reconstructions. *International Journal of Computer Vision*, 57(3):159–178, 2004.

[3] A. Bartoli and P. Sturm. Non-linear estimation of the fundamental matrix with minimal parameters. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 26(4):426–432, 2004.

[4] A. Bartoli and P. Sturm. Structure from motion using lines: Representation, triangulation and bundle adjustment. *Computer Vision and Image Understanding*, 100(3):416–441, December 2005.

[5] N. Birkbeck, D. Cobzaş, P. Sturm, and M. Jägersand. Variational shape and reflectance estimation under changing light and viewpoints. In H. Bischof and A. Leonardis, editors, *Proceedings of the 9th European Conference on Computer Vision, Graz, Austria*, Lecture Notes in Computer Science, May 2006.

[6] T. Bonfort and P. Sturm. Voxel carving for specular surfaces. In *Proceedings of the 9th International Conference on Computer Vision, Nice, France*, volume 1, pages 591–596. IEEE Computer Society Press, October 2003.

[7] T. Bonfort, P. Sturm, and P. Gargallo. General specular surface triangulation. In *Proceedings of the Asian Conference on Computer Vision, Hyderabad, India*, volume II, pages 872–881, January 2006.

[8] O. Chum, T. Pajdla, and P. Sturm. The geometric error for homographies. *Computer Vision and Image Understanding*, 97(1):86–102, January 2005.

[9] D. Cobzas and P. Sturm. 3D SSD tracking with estimated 3D planes. In *Proceedings of the Second Canadian Conference on Computer and Robot Vision, Victoria, Canada*, May 2005.

[10] O. Faugeras, L. Quan, and P. Sturm. Self-calibration of a 1d projective camera and its application to the self-calibration of a 2d projective camera. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 22(10):1179–1185, October 2000.

[11] P. Gargallo and P. Sturm. Bayesian 3D modeling from images using multiple depth maps. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, San Diego, USA*, volume 2, pages 885–891, June 2005.

[12] P. Gurdjos and P. Sturm. Methods and geometry for plane-based self-calibration. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Madison, Wisconsin, USA*, volume 1, pages 491–496, June 2003.

[13] P. Hammarstedt, P. Sturm, and A. Heyden. Closed-form solutions and degenerate cases for camera calibration with one-dimensional objects. In *Proceedings of the 10th International Conference on Computer Vision, Beijing, China*, October 2005.

[14] A. Jacquot, P. Sturm, and O. Ruch. Adaptive tracking of non-rigid objects based on color histograms and automatic parameter selection. In *Proceedings of the* IEEE *Workshop on Motion and Video Computing, Breckenridge, Colorado, USA*, pages 103–109, January 2005.

[15] S.J. Maybank and P. Sturm. MDL, collineations and the fundamental matrix. In T. Pridmore and D. Elliman, editors, *Proceedings of the 10th British Machine Vision Conference, Nottingham, England*, pages 53–62. British Machine Vision Association, September 1999.

[16] S. Ramalingam, P. Sturm, and S. Lodha. Theory and calibration algorithms for axial cameras. In *Proceedings of the Asian Conference on Computer Vision, Hyderabad, India*, volume I, pages 704–713, January 2006.

[17] S. Ramalingam, P. Sturm, and S.K. Lodha. Towards complete generic camera calibration. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, San Diego, USA*, volume 1, pages 1093–1098, June 2005.

[18] S. Ramalingam, P. Sturm, and S.K. Lodha. Towards generic self-calibration of central cameras. In *Proceedings of the 6th Workshop on Omnidirectional Vision, Camera Networks and Non-Classical Cameras, Beijing, China*, pages 20–27, October 2005.

[19] T. Rodriguez, P. Sturm, P. Gargallo, N. Guilbert, A. Heyden, J.M. Menendez, and J.I. Ronda. Photorealistic 3d reconstruction from handheld cameras. *Machine Vision and Applications*, 16(4):246–257, 2005.

[20] P. Sturm. Algorithms for plane-based pose estimation. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Hilton Head Island, South Carolina, USA*, pages 1010–1017, June 2000.

[21] P. Sturm. A case against Kruppa's equations for camera self-calibration. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 22(10):1199–1204, October 2000.

[22] P. Sturm. A method for 3D reconstruction of piecewise planar objects from single panoramic images. In *Proceedings of the* IEEE *Workshop on Omnidirectional Vision, Hilton Head Island, South Carolina*, pages 119–126. IEEE, June 2000.

[23] P. Sturm. Critical motion sequences for the self-calibration of cameras and stereo systems with variable focal length. *Image and Vision Computing*, 20(5-6):415–426, 2002.

[24] P. Sturm. Mixing catadioptric and perspective cameras. In *Proceedings of the Workshop on Omnidirectional Vision, Copenhagen, Denmark*, pages 37–44, June 2002.

[25] P. Sturm. Structure and motion for dynamic scenes – the case of points moving in planes. In A. Heyden, G. Sparr, M. Nielsen, and P. Johansen, editors, *Proceedings of the 7th European Conference on Computer Vision, Copenhagen, Denmark*, volume 2351 of *Lecture Notes in Computer Science*, pages 867–882. Springer-Verlag, May 2002.

[26] P. Sturm. Multi-view geometry for general camera models. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, San Diego, USA*, volume 1, pages 206–212, June 2005.

[27] P. Sturm and T. Bonfort. How to compute the pose of an object without a direct view? In *Proceedings of the Asian Conference on Computer Vision, Hyderabad, India*, volume II, pages 21–31, January 2006.

[28] P. Sturm, Z.L. Cheng, P.C.Y. Chen, and A.N. Poo. Focal length calibration from two views: Method and analysis of singular cases. *Computer Vision and Image Understanding*, 99(1):58–95, July 2005.

[29] P. Sturm and S. Maybank. On plane-based camera calibration: A general algorithm, singularities, applications. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Fort Collins, Colorado, USA*, pages 432–437, June 1999.

[30] P. Sturm and S.J. Maybank. A method for interactive 3D reconstruction of piecewise planar objects from single images. In T. Pridmore and D. Elliman, editors, *Proceedings of the 10th British Machine Vision Conference, Nottingham, England*, pages 265–274. British Machine Vision Association, September 1999.

[31] P. Sturm and L. Quan. Camera calibration and relative pose estimation from gravity. In A. Sanfeliu, J.J. Villanueva, M. Vanrell, R. Alquézar, J.-O. Eklundh, and Y. Aloimonos, editors, *Proceedings of the 15th International Conference on Pattern Recognition, Barcelona, Spain*, volume 1, pages 72–75, September 2000.

[32] P. Sturm and S. Ramalingam. A generic concept for camera calibration. In T. Pajdla and J. Matas, editors, *Proceedings of the 8th European Conference on Computer Vision, Prague, Czech Republic*, volume 3022 of *Lecture Notes in Computer Science*, pages 1–13. Springer-Verlag, May 2004.

[33] P. Sturm and S. Ramalingam. Géométrie d'images multiples pour des modèles de caméra généraux. *Traitement du Signal*, 22(5), October 2005.

[34] P. Sturm, S. Ramalingam, and S.K. Lodha. On calibration, structure-from-motion and multi-view geometry for general camera models. In R. Reulke and U. Knauer, editors, *Proceedings of the 2nd ISPRS Panoramic Photogrammetry Workshop, Berlin, Germany*. International Society for Photogrammetry and Remote Sensing, February 2005. Published in the Int. Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. XXXVI-5/W8.

[35] P. Sturm, S. Ramalingam, and S.K. Lodha. On calibration, structure from motion and multi-view geometry for generic camera models. In K. Daniilidis, R. Klette, and A. Leonardis, editors, *Imaging Beyound the Pinhole Camera*. Kluwer Academic Publishers, 2006.

[36] J.-P. Tardif and P. Sturm. Calibration of cameras with radially symmetric distortion. In *Proceedings of the 6th Workshop on Omnidirectional Vision, Camera Networks and Non-Classical Cameras, Beijing, China*, pages 44–51, October 2005.

[37] J.-P. Tardif, P. Sturm, and S. Roy. Self-calibration of a general radially symmetric distortion model. In H. Bischof and A. Leonardis, editors, *Proceedings of the 9th European Conference on Computer Vision, Graz, Austria*, Lecture Notes in Computer Science, May 2006.

[38] M. Urbanek, R. Horaud, and P. Sturm. Combining off- and on-line calibration of a digital camera. In *Proceedings of the Third International Conference on 3D Digital Imaging and Modeling, Québec City, Canada*, pages 99–106, May 2001.

[39] M. Wilczkowiak, P. Sturm, and E. Boyer. Using geometric constraints through parallelepipeds for calibration and 3D modelling. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 27(2):194–207, February 2005.