Generic Imaging Models: Calibration and 3D Reconstruction Algorithms

Srikumar Ramalingam

INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

Numéro attribué par la bibliothèque

THÈSE

pour obtenir le grade de

DOCTEUR DE L'INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

Spécialité : Imagerie, Vision et Robotique
Ecole Doctorale : Mathématiques, Sciences et Technologie de l'Information

présentée et soutenue publiquement

par

Srikumar RAMALINGAM

le 17 Novembre 2006

Generic Imaging Models: Calibration and 3D Reconstruction Algorithms

---

Directeurs de thèse : Peter STURM et Suresh K. LODHA

JURY

M. Crowley JAMES	Président
M. Andrew FITZGIBBON	Rapporteur
M. Tomas PAJDLA	Rapporteur
M. Peter STURM	Directeur de thèse
M. Suresh K. LODHA	Directeur de thèse
M. Hai TAO	Examinateur
M. Augustin LUX	Examinateur

Thèse préparée sous une convention de co-tutelle internationale, dans le laboratoire GRAVIR-IMAG (INRIA Rhône-Alpes, 655 avenue de l'Europe, 38330 Montbonnot, France) et le Department of Computer Science de l'Université de Californie à Santa Cruz (1156 High Street, Santa Cruz, CA 95064, USA)

Contents

List of Figures

List of Tables

Abstract

Vision applications such as surveillance and virtual reality have been using cameras which are beyond pinhole: stereo, fisheye cameras, catadioptric systems (combination of lenses with mirrors), multi-camera setups and other non-central cameras. These novel cameras have interesting properties such as larger field of view, better motion estimation, absolute scale recovery and better navigational capabilities. Camera calibration and 3D reconstruction algorithms are two fundamental blocks for computer vision; the former refers to parametric modeling of the transformation of a 3D scene to a 2D image, and the latter refers to the inverse process of reconstructing the 3D scene using 2D images. Models and algorithms for these two problems are usually parametric, camera dependent and seldom capable of handling heterogeneous camera networks, that are useful for complementary advantages. In order to solve these problems a generic imaging model is introduced, where every camera is modeled as a set of image pixels and their associated projection rays in space. Every image pixel measures the light travelling along a (half-) ray in 3-space, associated with that pixel. Calibration is non-parametric and refers to the computation of the mapping between pixels and the associated 3D rays. This mapping is essentially computed using images of calibration grids, which are objects with known 3D geometry, taken from unknown positions. 3D reconstruction algorithms (motion estimation, triangulation, and bundle adjustment) are developed as camera independent simple ray-intersection problems. Generic self-calibration, which uses arbitrary scenes rather than calibration grids, is studied and practical algorithms are proposed for central and radially symmetric cameras. Promising results are shown and contributions of this work are discussed.

Chapter 1

Introduction

1.1 History of pinhole and panoramic images

As a historical introduction, I briefly mention early works of art, especially paintings, which use perspective and panoramic ideas. It is surprising to know that perspective and panoramic images were explored as early as in the fourteenth and seventeenth centuries respectively.

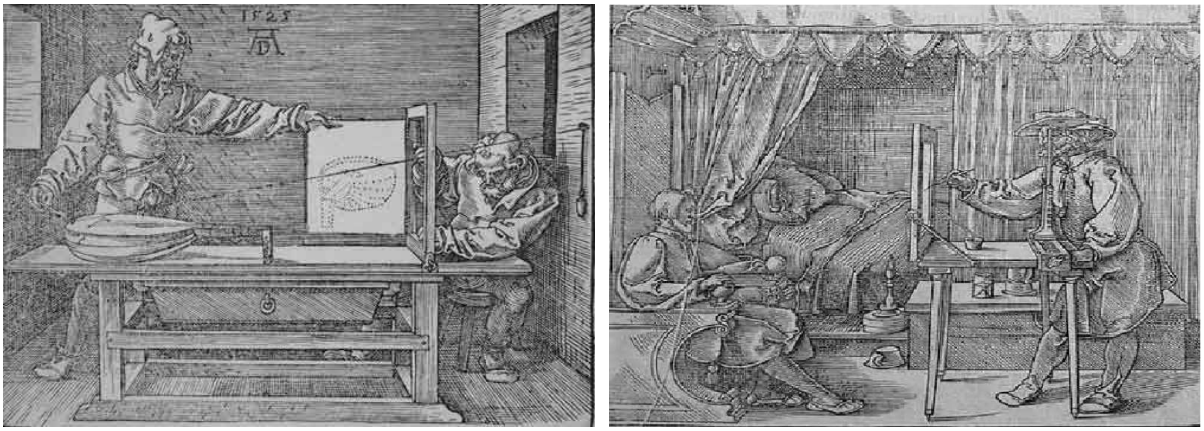


Figure 1.1: Left: One of Albrecht Dürer's perspective machines, which was used to draw a lute in the year 1525 [24]. Right: An artist uses the perspective principle to accurately draw a man sitting on a chair, by looking at him through a peep hole with one eye, and tracing his features on a glass plate.

Perspective images. Albrecht Dürer, a german artist, published a treatise on measurement using a series of illustrations of drawing frames and perspective machines. On the left side of Figure 1.1 we show an apparatus for drawing a lute. One end of a thread is attached to a pointer and the other end to a pulley on a wall. The thread also passes through a frame in between the lute and the pulley. When the pointer is fixed at different points on the lute, the vertical and horizontal co-ordinates of the thread, as it passes through the frame, are marked. By meticulously marking the coordinates for each point on the lute the perspective image of the lute is created. It is obvious to see the intuition behind this setup, i.e., its similarity to a pinhole camera. The pulley is equivalent to the single viewpoint, the frame replaces the image plane, and finally the thread is nothing but the light ray emerging from the scene. Though the principle is correct, the procedure is quite complicated.

On the right side of Figure 1.1, we illustrate another perspective machine. We observe an artist squinting through a peep hole with one eye to keep a single viewpoint, and tracing his sitter's features onto a glass panel. The idea is to trace the important features first and then transfer the drawing for further painting.

Leonardo da Vinci defines the perspective projection in the following manner:

"Perspective is nothing else than the seeing of an object through a sheet of glass, on the surface of which may be marked all the things that are behind the glass".

Panoramic paintings. Robert Barker, an Irishman, was walking on Carlton Hill with the whole vista of the city of Edinburgh. The Figure 1.2 shows his painting of the entire city on a single canvas. For public viewing, the painting was wrapped inside a large circular building, called a rotunda. The viewers were admitted through a spiral staircase and advised not to see the top or bottom of the painting to improve the illusion. This work was patented in 1787, mainly concerning the viewing of the panorama rather than its production.

Figure 1.2: The first panoramic painting by the Irish painter Robert Barker in the year 1792 (Courtesy of Edinburgh Virtual Environment Center).

1.2 Advantages of novel sensors

The pinhole camera is one of the most successful mathematical models in the field of computer vision, image processing and graphics. People naturally accepted this imaging model because of its extreme simplicity and its closeness to an image perceived by the human visual system. In a pinhole camera the projection rays from scene points (light rays) intersect at a single point (optical center). Typically these conventional cameras have a very small field of view, around 50°. Omnidirectional cameras have a larger field of view and have been extremely useful in several applications like video conferencing, augmented reality, surveillance and large scale 3D reconstruction. These cameras can be constructed in a simple manner, for they can be made from conventional cameras by using additional lenses or mirrors. For example in Figure 1.3(a), we show an E8 Nikon Coolpix camera appended with a fisheye lens having a field of view of $183^\circ \times 360^\circ$. Another possibility is to use mirrors in addition to lenses to increase the field of view. These configurations are referred to as *catadioptric*, where 'cata' comes from mirrors (reflective) and 'dioptric' comes from lenses (refractive). In Figure 1.3(b) and (c) we show two catadioptric configurations with hyperbolic and parabolic mirrors respectively. In these configurations the projection rays still pass through a single viewpoint in space. On relaxing this single viewpoint constraint, i.e., by allowing the projection rays to be arbitrary we introduce non-central cameras. Non-centrality provides more flexibility in designing omnidirectional cameras with larger field of view. In theory these novel sensors bring complementary advantages, some of which are given below.

- *Larger field of view:* Figure 1.4 shows images captured by three different omnidirectional cameras. The first image is captured by a fisheye camera, the second is captured by a catadioptric system constructed using a hyperbolic mirror and a perspective camera, and finally, the third is captured by another catadioptric camera constructed using a parabolic mirror and an orthographic camera. We can make the following observation from the omnidirectional images. A very large scene, that usually requires several pinhole images, can be captured in a single omnidirectional image although of course at a lower resolution.

(a) (b) (c) (d) (e)

Figure 1.3: (a) fisheye camera (b) catadioptric configuration using a hyperbolic mirror and a pinhole camera. (c) catadioptric configuration using a parabolic mirror and an orthographic camera (d) multi-camera setup (e) stereo configuration.

Figure 1.4: Omnidirectional images captured by three different cameras are shown: (a) image of a fisheye camera (b) image of a catadioptric camera using a hyperbolic mirror and a pinhole camera. (c) image of a catadioptric camera using a parabolic mirror and an orthographic camera.

- *Stable motion estimation:* Motion estimation is a challenging problem for pinhole images, especially when a larger number of images are involved. On the other hand, omnidirectional cameras stabilize the motion estimation and improves its accuracy [12]. In the case of small rigid motions, two different motions can yield nearly identical motion fields for classical perspective cameras. However, this is impossible in the case of omnidirectional cameras. By improving the stability of motion estimation, omnidirectional cameras also contribute to a stable 3D reconstruction.
- *Absolute scale recovery:* Central cameras can only provide 3D reconstruction up to a scale. In contrast, a calibrated non-central camera will provide the absolute scale in the 3D reconstruction and thereby avoid the need to use additional ground control points.
- *Non-rigid scene reconstruction:* In order to reconstruct a non-rigid scene, we can use stereo cameras to capture two images at the same time and to handle any motion in the scene.
- *Resolution enhancement:* Non-central catadioptric cameras can be designed to have less image blur and uniform spatial resolution.

1.3 The need for generic imaging models

Figure 1.5 demonstrates a remote video-conferencing scenario where the people sitting at a circular table are imaged as people sitting at a straight table. This interesting effect can be achieved by observing the scene through a specially designed mirror. By modifying the shape of the mirror, different scene to image

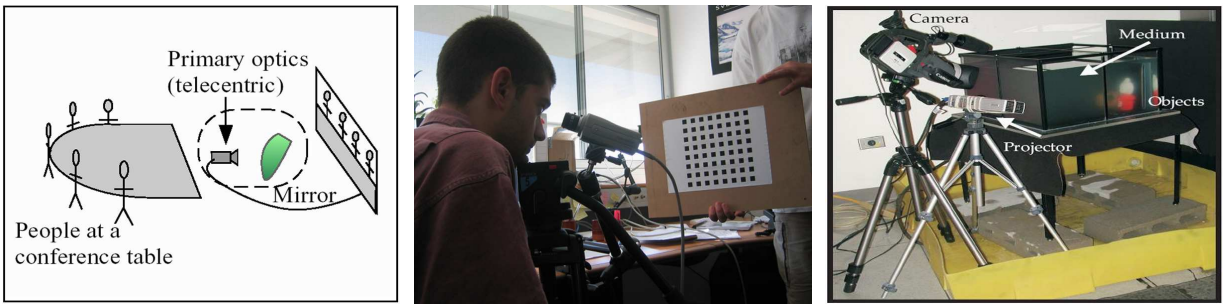


Figure 1.5: Three interesting scenarios showing the importance of generic imaging models. Left: Designing a catadioptric configuration (using a mirror) for custom viewing of a scene [109]. Middle: Shape reconstruction of the cornea of the eye by capturing the image of the scene reflected by the cornea [43]. Right: Camera looking at a scattering media to study structured light algorithms [68].

mappings can be achieved [109]. In Figure 1.5(b), the structure of the eye is computed by studying the image of a known scene reflected on the cornea. This would be useful for medical applications. In Figure 1.5(c) the nature of structured light methods in a scattering media can be studied after calibrating a camera in such a media. Each of the above three scenarios can be compared to a novel imaging model.

We appreciate the fact that novel sensors provide a wide variety of complementary advantages as listed above. Despite this we do not have a unified theory to handle different cameras. Each of the existing cameras has a different parameterization. For example, pinhole cameras are represented using five internal parameters, which correspond to focal length, aspect ratio, skew and principal point (2 parameters). On the other hand omnidirectional cameras like fisheye cameras need additional parameters to handle larger distortions. These parameters also differ based on the distortion function such as field of view (FOV) and division model (DM) (see section 3.1.6). The number of parameters in each of these distortion models can vary. Catadioptric cameras have completely different parameterizations based on the shape of the mirrors. It is obvious that different cameras lead to different parameterizations and eventually to different calibration and structure-from-motion algorithms. It requires considerable effort to stabilize these algorithms in practice. The whole cycle has to be repeated for every novel sensor. The passion to use novel sensors without encountering the above-mentioned difficulties acted as the primary motivation for our work. Alternatively we may say that we address the following questions in this thesis.

- Is there a generic imaging model to represent any arbitrary imaging system?
- Does there exist a practical calibration algorithm to calibrate the generic imaging model?
- Can we develop multi-view geometry and structure-from-motion algorithms for the generic imaging model?

1.4 Overview and Organization

- Chapter 3 gives a taxanomy of camera models going from from simple perspective cameras to generic imaging models. We give a brief introduction to generic imaging models, which is the topic of this thesis. In such a model, an image is considered as a collection of pixels, and each pixel measures the light traveling along a (half-) ray in 3-space associated with that pixel. Calibration consists of the following information:
  - 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 traveling along straight lines. 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. The generic imaging model consists of three important sub-classes based on the geometry of projection rays: central, axial and non-central. The central class refers to cameras where all the projection rays intersect at a single viewpoint. The axial class refers to cameras where all the projection rays intersect a line in space. The non-central class refers to cameras where the projection rays can be completely arbitrary. Several examples for each of these cameras will be given.

Several past works in camera calibration and 3D reconstruction algorithms for different camera models are also presented. We believe that this background would enable the readers to get a better understanding of our contribution in this thesis. Calibration and 3D reconstruction algorithms for generic imaging models are discussed in the remaining chapters from 4 to 10.

- Chapter 4 focuses on the generic calibration. We propose a concept for calibrating the above 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 planar calibration objects. The validity of the concept is shown by experiments with synthetic and real data.

- Chapter 5 focuses on the generic calibration of general *central* cameras. The calibration algorithm developed for general non-central cameras in chapter 4 gives degeneracy problems (or ambiguities in the calibration solution) when applied to single viewpoint cameras. This implies that the single viewpoint constraint has to be imposed in formulating the generic calibration problem to remove the degeneracies. In this chapter we consider this central variant of generic calibration using planar calibration grids. Central cameras like pinhole, fisheye, and central catadioptric cameras are tested using our algorithm. We also demonstrate high quality distortion correction results for fisheye images, and thereby, assert the fact that the central model for fisheye images is indeed a good assumption. Experimental results are also shown for slightly non-central cameras using a central initialization followed by a non-central bundle adjustment.

- Chapter 6 studies a new class of cameras which we call *axial cameras*. Previous works exist on calibration and structure-from-motion algorithms 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. 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 use simulations and real experiments to prove the validity of our theory.

- In Chapter 8 we introduce a structure-from-motion approach for our general imaging model, that allows to reconstruct scenes from calibrated images, possibly taken by cameras of different types (cross-camera scenarios). Structure-from-motion is naturally handled via camera independent ray intersection problems, solved via linear or simple polynomial equations. We also propose two approaches for obtaining optimal solutions using bundle adjustment, where camera motion, calibration and 3D point coordinates are refined simultaneously.

- In Chapter 9 we consider the self-calibration problem for the generic imaging model. 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 chapter talks about our first steps towards generic self-calibration; we consider specific camera motions, concretely, pure translations and rotations. 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.

- Chapter 10 presents a novel approach for self-calibration of radially symmetric cameras. We model these camera images using notions of distortion center and concentric distortion circles around it. The rays corresponding to pixels lying on a single distortion circle form a right circular cone. Each of these cones is associated with two unknowns; optical center and focal length (opening angle). In the central case, we consider all distortion circles to have the same optical center, whereas in the non-central case they have different optical centers lying on the same optical axis. Based on this model we provide a factorization based self-calibration algorithm for planar scenes using dense image matches. Our formulation provides a rich set of constraints to validate the correctness of the distortion center. We also propose possible extensions of this algorithm in terms of non-planar scenes, non-unit aspect ratio and multi-view constraints. Experimental results are shown.

- Chapter 11 summarizes the contributions and possible extensions of this work. We also list a few related challenging problems that are worth investigating.

- In Appendix A.3 and A.4 we study underconstrained cases for 2D and 3D cameras respectively. For example, we investigate what happens when we use the non-central calibration algorithm, but with data that stems from a central camera. Appendix A.1 summarises the nature of the results (unique, inconsistent, ambiguous) on applying the various calibration algorithms, tailormade for specific camera models (central, non-central, axial), on other camera models. Additional material on generic camera calibration using restricted motion sequences is given in Appendix A.2.

Chapter 2

Résumé en Français

2.1 Histoire des imageries perspective et panoramique

Pour donner une introduction historique, je mentionne brièvement des oeuvres d'art, particulièrement des peintures, dont la création s'appuyait sur des principes de la perspective ou de l'imagerie panoramique. Il est surprenant d'apprendre que ces idées étaient explorées aussi tôt qu'aux quatorzième et dix-septième siècles respectivement.

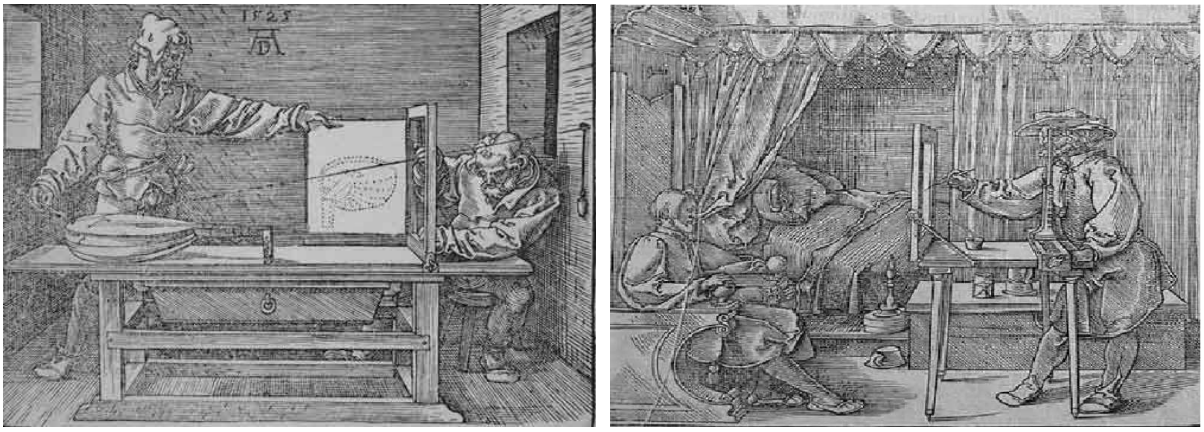


Figure 2.1: Gauche : Une des "machines perspectives" développes par Albrecht Dürer qui a été utilisée pour dessiner un luth en 1525 [24]. Droite : Un artiste utilise le principe de la perspective afin de dessiner précisément un homme assis sur une chaise, en le regardant au travers d'une ouverture avec un œil et en traçant ses traits sur une plaque en verre.

Images perspective. Albrecht Dürer, un artiste allemand, a publie un traité sur la mesure à partir d'une série d'illustrations obtenues utilisant des machines perspectives. Sur la gauche de la figure 2.1 est montré un appareil pour dessiner un luth. Une extrémité d'un fil est attachée à un pointeur et l'autre à un point fixe sur un mur. Le fil passe par un cadre entre le luth et le mur. Quand le pointeur est arrêté sur différents points du luth, les coordonnées verticale et horizontale du fil, au point de passage par le cadre, sont notées. En dessinant des points selon les coordonnées de points caractéristiques du luth sur un support, une image perspective du luth est générée. La similarité entre ce procédé et le modèle de caméra perspective ou sténopé est évidente. Le point fixe sur le mur est équivalent au centre optique d'une caméra, le cadre remplace le plan de l'image et le fil ne correspond à rien d'autre qu'une ligne de vue.

Sur la droite de la figure 2.1, un autre appareil similaire est esquissé. On y voit un artiste regardant

avec un œil à travers un trou tout en traçant les traits du modèle sur une plaque en verre. Le principe est de tracer d'abord les traits principaux sur la plaque en verre, puis de les transférer sur un autre support afin de compléter la peinture. Evidemment, le trou correspond au centre optique d'une caméra perspective. Par ailleurs, Leonardo da Vinci définit la projection perspective ainsi :

"La perspective n'est rien d'autre que de regarder un objet à travers une plaque en verre, sur laquelle on marque toutes les choses qui se trouvent derrière le verre".

Peintures panoramiques. Robert Barker, un Irlandais, en se promenant sur Carlton Hill, d'où il avait une vue panoramique d'Edimbourg, se décida de la reproduire en peinture. La figure 2.2 montre sa peinture de la ville entière sur un seul canevas. Afin de la montrer au public, elle a été posée sur les murs intérieurs d'un large bâtiment circulaire. Les spectateurs montait au centre par un escalier en forme de spirale et on leur conseillait de ne pas regarder le haut ou le bas de la peinture, afin d'améliorer l'illusion d'immersion. Ce procédé a été breveté en 1787, principalement en ce qui concerne l'affichage de la peinture panoramique plutôt que sa création.

Figure 2.2: La première peinture panoramique, créée par le peintre Irlandais Robert Barker en 1792 The first panoramic painting by the Irish painter Robert Barker in the year 1792 (Propriété du *Edinburgh Virtual Environment Center*).

2.2 Avantages de nouveaux types de caméras

La caméra perspective est l'un des modèles les plus utilisés dans les domaines de la vision par ordinateur, du traitement d'images et de l'infographie. Ce modèle est accepté de par sa simplicité est la similarité aux images perçues par le système de vision humain. Dans une caméra perspective les rayons de projection émanent de points dans la scène (rayons de lumière) se coupent tous en un même point (centre optique). Les caméras qui réalisent ce modèle ont typiquement des champs de vue relativement restreints, jusqu'à environ $50°$.

Les caméras omnidirectionnelles ont un champs de vue beaucoup plus large et sont donc utiles dans diverses applications telle la vidéoconférence, la réalité augmentée, la vidéosurveillance ou encore la modélisation 3D d'environnements étendus. Ces caméras peuvent être construites de différentes manières, parfois en combinant une caméra habituelle avec des objectifs particuliers ou des miroirs. Par exemple, la figure 2.3(a) montre une caméra E8 Nikon Coolpix avec un objectif *fish-eye* ayant un champs de vue de $183° \times 360°$. Une autre possibilité est d'utiliser des miroirs afin d'étendre le champs de vue. De telles constructions sont appelées *caméras catadioptriques*, "cata" désignant un élément réflechissant (miroir) et "dioptrique" un élément réfractant (lentille). Les figures 2.3(b) et (c) montrent deux caméras catadioptriques, utilisant un miroir de forme hyperboloïdale et paraboloïdale respectivement. Pour ces configurations, les rayons de projection (ou lignes de vue) se coupent toujours tous en un seul point (centre optique effectif). Des configurations plus générales sont possibles, pour lesquelles un centre optique unique n'existe plus ; c'est le cas si les rayons de projection peuvent être répartie de manière plus générale dans l'espace. Nous parlons alors de *caméras non centrales*. Les caméras non centrales, en renonçant à la contrainte d'un centre optique unique, permettent une plus grande flexibilité dans la conception de caméras avec un champs de vue étendu. Tous ces capteurs omnidirectionnels ont des avantages par rapport aux caméras habituelles, dont

certains sont résumés ci-dessous.

(a) (b) (c) (d) (e)

Figure 2.3: (a) Caméra avec un objectif fish-eye. (b) Capteur catadioptrique utilisant un miroir de forme hyperboloïdale et une caméra perspective. (c) Capteur catadioptrique utilisant un miroir de forme paraboloïdale et une caméra orthographique. (d) Configuration multi-caméras. (e) Système stéréo.

- *Champs de vue étendu :* La figure 2.4 montre des images acquises par trois caméras omnidirectionnelles différentes. La première a été acquises par une caméras fish-eye, la deuxième par une caméras "hyper-catadioptrique" et la troisième par une caméras "para-catadioptrique". Nous pouvons faire l'observation qu'une scène large, dont une couverture complète requiert plusieurs images perspectives, peut être représentée sur une seule image omnidirectionnelles, quoique à une résolution diminuée.

- *Estimation du mouvement :* L'estimation du mouvement d'une caméra est un problème important en vision par ordinateur et qui n'est pas toujours stable pour des caméras perspectives. Une raison principale en est le fait que les mouvements apparents induits par une rotation latérale ou une translation latérale sont assez similaires. Avec des caméras omnidirectionnelles par contre, il existe forcément des régions sur l'image où les mouvements apparents dus aux parties translationnelle et rotationnelle du mouvement de la caméra sont très différents. Ainsi, l'estimation du mouvement est toujours stable et peut être plus précise qu'avec une caméra perspective [12]. Grâce à la stabilité de l'estimation du mouvement, les caméras omnidirectionnelles peuvent également contribuer à une reconstruction 3D stable.

- *Calcul de l'échelle absolue :* Des caméras centrales (caméras avec un centre optique unique) ne peuvent fournir une reconstruction 3D qu'à l'échelle près. Une caméra non centrale calibrée par contre, "porte" sur elle l'échelle absolue ; elle peut par exemple être donnée par la distance entre deux rayons de projection divergents, information donnée par le calibrage. En théorie, lorsqu'on utilise une caméra non centrale, des points de contrôle ne sont alors pas nécessaires pour retrouver l'échelle absolue d'une reconstruction 3D. En pratique par contre, cela est uniquement fiable si la taille de la scène reconstruite et sa distance par rapport à la caméra ne dépassent pas l'échelle de référence de plus d'un ordre de grandeur.

- *Reconstruction de scènes non rigides :* Afin de reconstruire une scène non rigide, nous pouvons utiliser des systèmes stéréo ou multi-caméras, qui captent deux ou plusieurs images simultanément.

- *Résolution :* Il est possible de concevoir des caméras catadioptriques non centrales ayant une résolution spatiale uniforme.

(a) (b) (c)

Figure 2.4: Images omnidirectionnelles acquises par trois caméras différentes : (a) Caméra avec un objectif fish-eye. (b) Capteur catadioptrique utilisant un miroir de forme hyperboloïdale et une caméra perspective. (c) Capteur catadioptrique utilisant un miroir de forme paraboloïdale et une caméra orthographique.

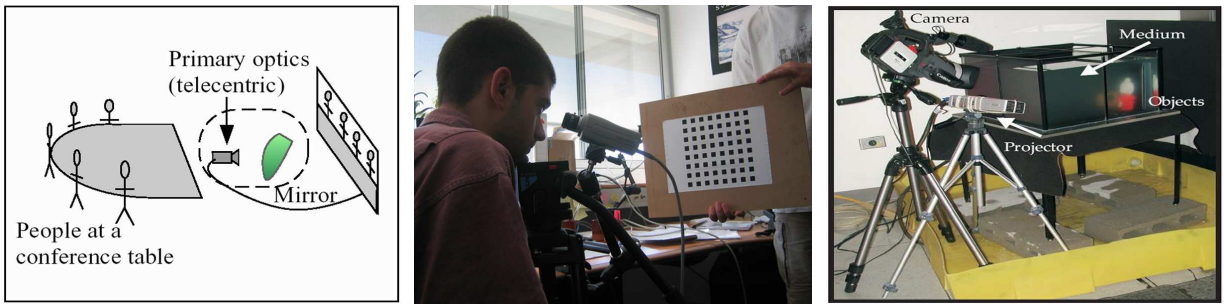


Figure 2.5: Trois scénarios intéressants qui montrent l'importance de modèles de caméra génériques. Gauche : Conception d'un capteur catadioptrique qui produit une image désirée d'une scène d'un certain type [109]. Milieu : Reconstruction de la cornée de l'œil humain en utilisant des images d'une scène réfléchie dans la cornée [43]. Droite : Caméra regardant au travers d'un médium réfractant [68].

2.3 Modèles de caméra génériques

La figure 2.5 montre un scénario de vidéoconférence où les participants dans un site sont assis autour d'une table circulaire, mais sont vus dans un autre site comme s'ils étaient assis à une table rectiligne. Cet effet peut être obtenu en utilisant une caméra équipée d'un miroir spécialement conçu pour obtenir l'image désirée. De manière générale, des miroirs de différentes formes permettent de réaliser différentes projections de la scène sur l'image [109]. La figure 2.5(b) illustre comment la forme d'un œil peut être calculée à partir d'images d'un objet réfléchi dans la cornée. La figure 2.5(c) montre un système de lumière structurée conçu pour être utilisé dans un milieu composé de différents médias réfractants. Tous ces scénarios peuvent être assimilés à des modèles de caméra particuliers.

Il existe alors de nombreux modèles de caméra, certains assez inhabituels, et dont certains ont des propriétés intéressantes comme décrits plus haut. Malgré cela, il n'existe aucune théorie unifiante dans la littérature, qui permette de modéliser et traiter tous les différents modèles de caméra en un seul cadre. On trouve bien sûr des paramétrisations faites sur mesure pour les divers modèles. Par exemple, les caméras perspectives sont représentées par cinq paramètres intrinsèques, la distance focale, le rapport d'aspect, le "skew" et les coordonnées du point principal. La modélisation des caméras omnidirectionnelles, quant à

elles, nécessite souvent plus de paramètres afin de gérer les distorsions non perspectives. La littérature contient beaucoup de modèles de caméra, le plus souvent faits sur mesure pour un certain type de caméras, mais parfois aussi des modèles unifiés pour une classe de caméras, notamment pour les caméras catadioptriques à centre optique unique. De plus, beaucoup de méthodes différentes de calibrage et de "structure-from-motion" ont été développées, souvent faites sur mesure pour un modèle de caméra à la fois. Ainsi, la conception d'un nouveau capteur est souvent accompagnée par le développement d'un nouveau modèle de caméra et de nouvelles méthodes de calibrage. Notre passion pour l'utilisation de nouveaux types de capteurs et notre souhait de contourner difficultés mentionnées ont été les principales motivations pour le travail présenté dans cette thèse. Elles peuvent être résumées par les questions suivantes que nous considérons dans cette thèse :

- Existe-t-il un modèle de caméra générique qui permette de représenter tout type de caméra ?
- Existe-t-il un algorithme de calibrage pratique qui permette de calibrer un tel modèle de caméra générique ?
- Pouvons-nous développer des théories et algorithmes de géométrie d'images multiples et de "structure-from-motion" pour un tel modèle de caméra générique ?

2.4 Aperçu et structure de la thèse

- Le chapitre 3 présente une taxonomie de modèles de caméra, allant du modèle perspectif jusqu'à des moèles génériques. Nous donnons une brève introduction aux modèles de caméra génériques, le sujet principal de cette thèse. Dans ces modèles, une image est considérée comme une collection de pixels, où chaque pixel capte la lumière arrivant le long d'une (demi-) droite en 3D associée à ce pixel (son rayon de projection ou encore sa ligne de vue). Le calibrage de ces modèles consiste en le calcul des informations suivantes :
  - Les coordonnées des rayons de projection, données dans un repère commun ;
  - L'association entre les rayons de projection et les pixels, sous forme d'une "look-up-table".

  Ces modèles permettent de représenter presque toutes les caméras qui captent des rayons de lumière se propageant sur des lignes droites, dont les caméras habituellement utilisées en vision par ordinateur et photogrammétrie : caméras perspectives et affines, ayant ou non des distorsion non perspectives, systèmes stéréo, ou encore les capteurs catadioptriques, centraux ou non centraux. Nous définissons trois classes de modèles de caméra génériques, selon la répartition des rayons de projection : modèle central, axial ou non central. Dans le modèle central, tous les rayons de projection se coupent en un seul point, le centre optique, mais peuvent sinon être arbitraires. Dans le modèle axial, ils coupent tous une ligne particulière dans l'espace, appelée dans la suite *axe de caméra*. Finalement, le modèle non central concerne toutes les caméras encore plus générales dont les rayons de projection peuvent être complètement arbitraires. Des exemples pour chacun des trois modèles sont donnés.

  Dans ce chapitre nous décrivons également certains travaux de la littérature sur le calibrage de caméras et la reconstruction 3D à partir d'images, pour différents modèles de caméra. Le calibrage et la reconstruction 3D pour nos modèles génériques sont décrits dans les chapitres 4 à 10.

- Le chapitre 4 concerne le calibrage de notre modèle non central générique. Nous proposons un concept de calibrage à partir de plusieurs images d'un objet de référence (mire de calibrage), acquises depuis des points de vue inconnus. Il permet en principe de calibrer n'importe quelle caméra qui peut être représentée par le modèle non central générique, avec un unique algorithme. Nous présentons

d'abord un algorithme utilisant des mires de calibrage tridimensionnelles. Ensuite, un algorithme basé sur des mires planes est proposé. La validité de notre concept est montrée à travers des résultats expérimentaux sur des données synthétiques et réelles.

- Le chapitre 5 est dédié au calibrage du modèle de caméra *central*. Les algorithmes du chapitre 4 ne fonctionnent qu'avec des caméras non centrales et donnent lieu à des dégénérescences/ambiguïtés si appliqués à des caméras centrales (ou axiales). Nous montrons comment imposer la propriété d'un centre optique unique, ce qui permet d'éviter ces problèmes. Dans ce chapitre nous décrivons un algorithme de calibrage utilisant des mires planes. Il a été appliqué à des caméras centrales telles des caméras perspectives, fish-eye et des caméras catadioptriques centrales. Nous montrons de bons résultats pour la correction de distorsions non perspectives à partir des résultats du calibrage, pour des caméras fish-eye. Ceci montre par ailleurs qu'un modèle central peut être suffisant pour représenter de telles caméras.

- Le chapitre 6 concerne les *caméras axiales*. Le modèle axial (voir ci-dessus) comprend les systèmes stéréo, beaucoup de caméras catadioptriques et des capteurs de type push-broom par exemple. Nous étudions la géométrie des caméras axiales et en proposons un algorithme de calibrage. Nous décrivons également les configurations catadioptriques qui ne sont pas centrales mais axiales. Finalement, nous montrons des résultats expérimentaux sur des données simulées et réelles pour prouver la validité de notre théorie.

- Dans le chapitre 8 nous proposons une approche de "structure-from-motion" pour notre modèle de caméra générique qui permet de reconstruire des scènes en 3D à partir d'images calibrées, éventuellement acquises par des caméras de différentes types. La reconstruction 3D et d'autres problèmes de "structure-from-motion" sont formulés sur la base d'intersections de lignes droites (les rayons de projection) et sont résolus via des systèmes d'équations linéaires ou polynomiales. Nous proposons également deux approches pour l'ajustement de faisceaux, c'est-à-dire l'optimisation simultanée des poses des caméras et des coordonnées 3D des points reconstruits.

- Au chapitre 9 nous considérons le problème de l'auto-calibrage pour le modèle de caméra central générique. Il s'agit de calculer tous les rayons de projection, et ce uniquement à partir de correspondances entre des images, c'est-à-dire sans connaissance aucune sur la scène (pas d'utilisation d'une mire de calibrage). Ce chapitre décrit nos premiers pas vers un auto-calibrage générique ; nous considérons des mouvements de caméra particuliers, des translations et rotations pures. La connaissance du type de mouvement, ensemble avec les correspondances entre des images, donnent des contraintes géométriques sur les rayons de projection. Il est montré qu'à partir de mouvements translationnels uniquement, l'auto-calibrage est déjà possible, mais uniquement à une transformation affine près de l'ensemble des rayons de projection. Nous proposons ensuite un algorithme pratique pour un auto-calibrage complet, utilisant des mouvements de translation et rotation.

- Le chapitre 10 décrit une approche pour l'auto-calibrage de caméras radialement symétriques. Ces caméras sont modélisées en utilisant les notions de centre de distorsion et de cercles de distorsion, centrés dans le centre de distorsion. La propriété de symétrie radiale se manifeste par le fait que les rayons de projection associés aux pixels sur un même centre de distorsion, forment un cône circulaire. Tous les cônes partagent l'axe. Ainsi, chacun de ces cônes compte pour deux inconnues dans le modèle de caméra : position du centre optique (vertex du cône) et distance focale (ouverture du cône). Dans la variante centrale du modèle radialement symétrique tous les cônes ont le même centre optique tandis que dans le cas non central ils peuvent avoir des centres optiques différents. Nous proposons un algorithme d'auto-calibrage, basé sur des correspondances denses associées à une scène plane et utilisant une factorisation de matrice. Nous proposons également des extensions possibles pour des

scènes non planes, un rapport d'aspect différent de 1 ou encore des contraintes d'appariement multi-images. Des résultats expérimentaux pour l'auto-calibrage sont montrés.

- Le chapitre 11 résume nos contributions et décrit des perspectives de notre travail. Nous citons également quelques problèmes importants liés qui valent investigation.

- Dans les annexes A.3 et A.4 nous étudions les dégénérescences mentionnées plus haut, lorsqu'un algorithme de calibrage non central est appliqué à une caméra centrale par exemple. Ceci est d'abord fait pour le cas d'un monde 2D, plus simple à décrire, puis pour l'espace 3D habituel.

  L'annexe A.1 résume la nature du résultat (unique, inconsistant, ambigu) lorsqu'on applique un algorithme de calibrage conçu pour un modèle (central, axial, non central) à une caméra d'un autre modèle.

  L'utilisation de mouvements particuliers pour le calibrage des modèles génériques est esquissé dans l'annexe A.2.

Chapter 3

Background

In this chapter we present a hierarchy of camera models ranging from the perspective model to the most general *generic imaging models*. The camera model, usually given in parametric form, is required to possess two important capabilities: forward projection and back-projection. In forward projection, for a given 3D point in space, the model provides a mechanism to compute its corresponding image point. In back-projection, for a given image point, the model provides a method to compute the set of 3D points which map to the image point. Concisely, the former refers to the projection of a 3D point to its corresponding 2D image point and the latter refers to the back-projection of a 2D image point to its corresponding 3D ray.

In addition to introducing different camera models we also provide details on their calibration, epipolar geometry and 3D reconstruction algorithms. These algorithms have been studied exhaustively for perspective cameras. Though our background might suffice to maintain the flow and continuity, this is barely sufficient to give a complete overview of various works in multi-view geometry for perspective cameras. Most of the advances in this topic, especially the ones before the millennium year can be found in [28, 49], and a rather brief review follows.

3.1 Camera Models

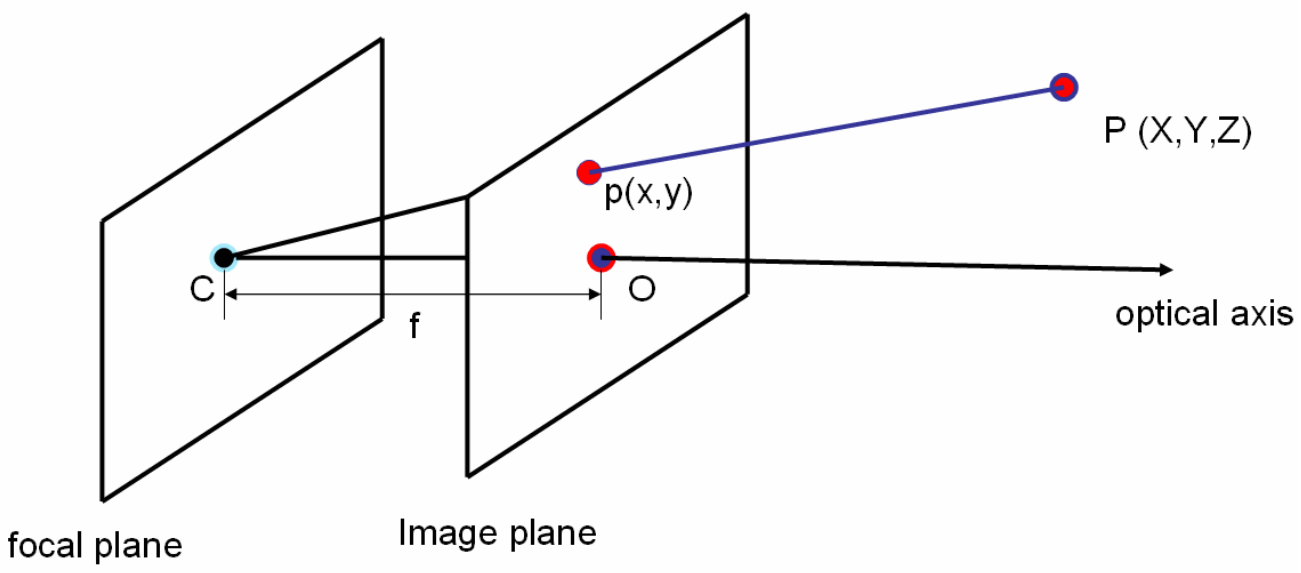
3.1.1 Perspective model

Let us consider the perspective model that is shown in Figure 3.1. Every 3D scene point $\mathbf{P}(X, Y, Z)$ gets projected onto the image plane to a point $\mathbf{p}(x, y)$ through the optical center $\mathbf{C}$. The Optical axis is the perpendicular line to the image plane passing through the optical center. The center of radial symmetry in the image or principal point, i.e., the point of intersection of the optical axis and the image plane is given by $\mathbf{O}$. The distance between $\mathbf{C}$ (optical center) and the image plane is the focal length f. We define the camera coordinate system as follows. The optical center of the camera is the origin of the coordinate system. The image plane is parallel to the XY plane, held at a distance of f from the origin. Using the basic laws of trigonometry we observe the following:

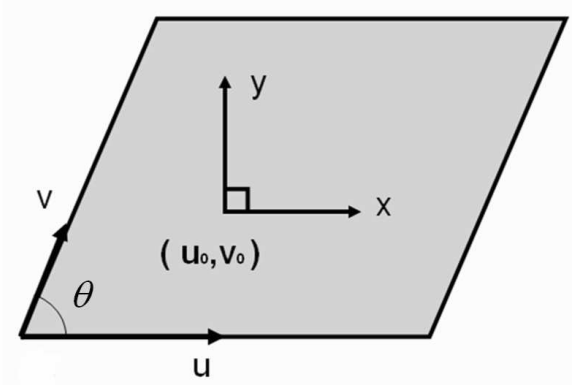
$$x = \frac{fX}{Z}, y = \frac{fY}{Z}$$

Once expressed in homogeneous coordinates the above relations transform to the following:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$



(a)



(b)

Figure 3.1: (a) Perspective camera model. (b) The relationship between (u, v) and (x, y) is shown.

where the relationship $\sim$ stands for 'equal upto a scale'.

Practically available CCD cameras deviate from the perspective model. First, the principal point (u_0, v_0) does not necessarily lie on the geometrical center of the image. Second, the horizontal and vertical axes (u and v) of the image are not always perfect perpendicular. Let the angle between the two axes be θ. Finally, each pixel is not a perfect square and consequently we have f_u and f_v as the two focal lengths that are measured in terms of the unit lengths along u and v directions. By incorporating these deviations in the camera model we obtain the following scene (X, Y, Z) to image (u, v) transformation:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim \begin{bmatrix} f_x & f_v cot\theta & u_0 & 0 \\ 0 & \frac{f_v}{sin\theta} & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

In practice the 3D point is available in some world coordinate system that is different from the camera coordinate system. The motion between these coordinate systems is given by $(\mathsf{R}, \mathbf{t})$:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim \begin{bmatrix} f_x & f_v cot\theta & u_0 \\ 0 & \frac{f_v}{sin\theta} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathsf{R} & -\mathsf{R}\mathbf{t} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{3.1}$$

$$\mathsf{M} = \begin{bmatrix} f_x & f_v cot\theta & u_0 \\ 0 & \frac{f_v}{sin\theta} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathsf{R} & -\mathsf{R}\mathbf{t} \end{bmatrix}$$

$$\mathsf{K} = \begin{bmatrix} f_x & f_v cot\theta & u_0 \\ 0 & \frac{f_v}{sin\theta} & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

The 3×4 matrix M that projects a 3D scene point $\mathbf{P}$ to the corresponding image point $\mathbf{p}$ is called the projection matrix. The 3×3 matrix K that contains the internal parameters $(u_0, v_0, \theta, f_x, f_y)$ is generally referred to as the *intrinsic* matrix of a camera.

In back-projection, given an image point $\mathbf{p}$, the goal is to find the set of 3D points that project to it. The back-projection of an image point is a ray in space. We can compute this ray by identifying two points on this ray. The first point can be the optical center $\mathbf{C}$, since it lies on this ray. Since $\mathsf{M}\mathbf{C} = \mathbf{0}$, $\mathbf{C}$ is nothing but the right nullspace of M. Second, the point M^+p, where M^+ is the pseudoinverse[1] of M, lies on the back-projected ray because it projects to point $\mathbf{p}$ on the image. Thus the back-projection of $\mathbf{p}$ can be computed as follows.

$$\mathsf{P}(\lambda) = \mathsf{M}^+\mathbf{p} + \lambda\mathbf{C}$$

The parameter λ allows us to get different points on the back-projected ray.

3.1.2 Orthographic model

We show an orthographic camera model in Figure 3.2. This is an affine camera model that has a projection matrix M in which the last row has the form $(0, 0, 0, 1)$. In particular, orthographic camera model has a projection matrix M of the following form:

[1]The pseudoinverse A^+ of a matrix A is a generalization of the inverse and it exists for general (m, n) matrix. If $m > n$ and if A has full rank (n) then $\mathsf{A}^+ = (\mathsf{A}^\mathsf{T}\mathsf{A})^{-1}\mathsf{A}^\mathsf{T}$.

$$\mathsf{M} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \mathsf{R} & \mathbf{t} \\ 0 & 1 \end{pmatrix}$$

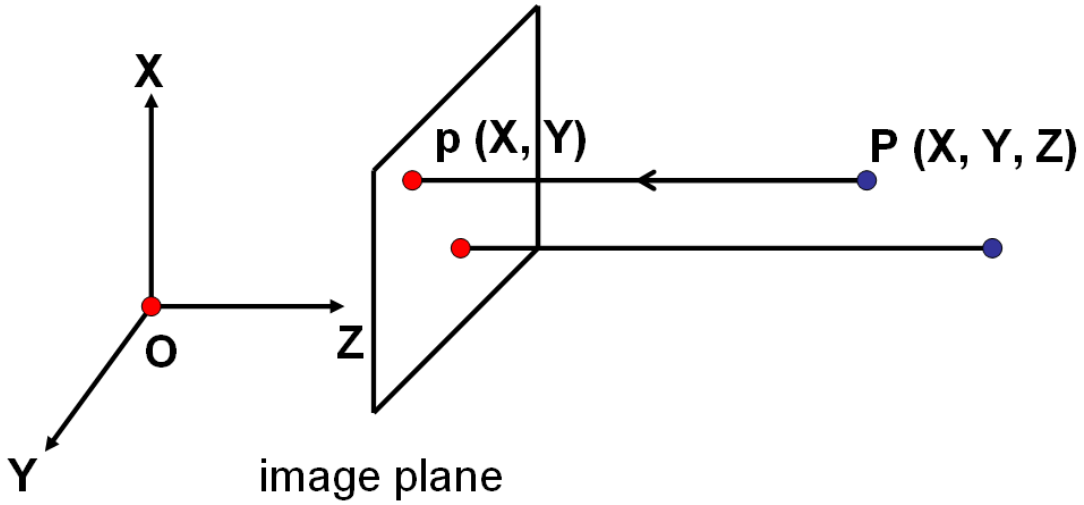


Figure 3.2: Orthographic camera model. The 3D point $\mathbf{P}(X, Y, Z)$ is projected onto the 2D point $\mathbf{p}(x, y)$. The optical center is a point at infinity. When both the camera and the world coordinate systems are the same, i.e. when $\mathsf{R} = \mathsf{I}$ and $\mathbf{t} = 0$, then a 3D point $\mathbf{P}(X, Y, Z)$ gets projected onto the 2D point $\mathbf{p}(X, Y)$.

The projection of a 3D point $\mathbf{P}$ on to the image point $\mathbf{p}$ is given below:

$$\mathbf{p} = \mathsf{M}\mathbf{P}$$

Similar to the perspective camera the back-projected ray is given below:

$$\mathbf{P}(\lambda) = \mathsf{M}^{+}\mathbf{p} + \lambda\mathbf{C}$$

However the optical center $\mathbf{C}$, which is the right nullspace of M, is a point at infinity in an orthographic camera.

3.1.3 Pushbroom model

We show a linear pushbroom camera [42] in Figure 3.3. We briefly explain a method to create a linear pushbroom image by translating a perspective camera. A perspective camera is moved in a straight line and a sequence of images are captured. We extract a single column (for example, the center column) from every frame and concatenate these columns to obtain a linear pushbroom image. As a result, a linear pushbroom image can be considered as a perspective image along one direction (vertical) and as an orthographic image along the other direction (horizontal).

3.1.4 Crossed-slit model

In a crossed-slit camera [31], every projection ray passes through two lines ('slits') (cf. Figure 3.4). Crossed-slit images can be obtained from translating perspective cameras. We sample different columns from translated perspective images and mosaic them to obtain a crossed-slit image. Note that we sample the same column in all the perspective images while constructing a pushbroom image.

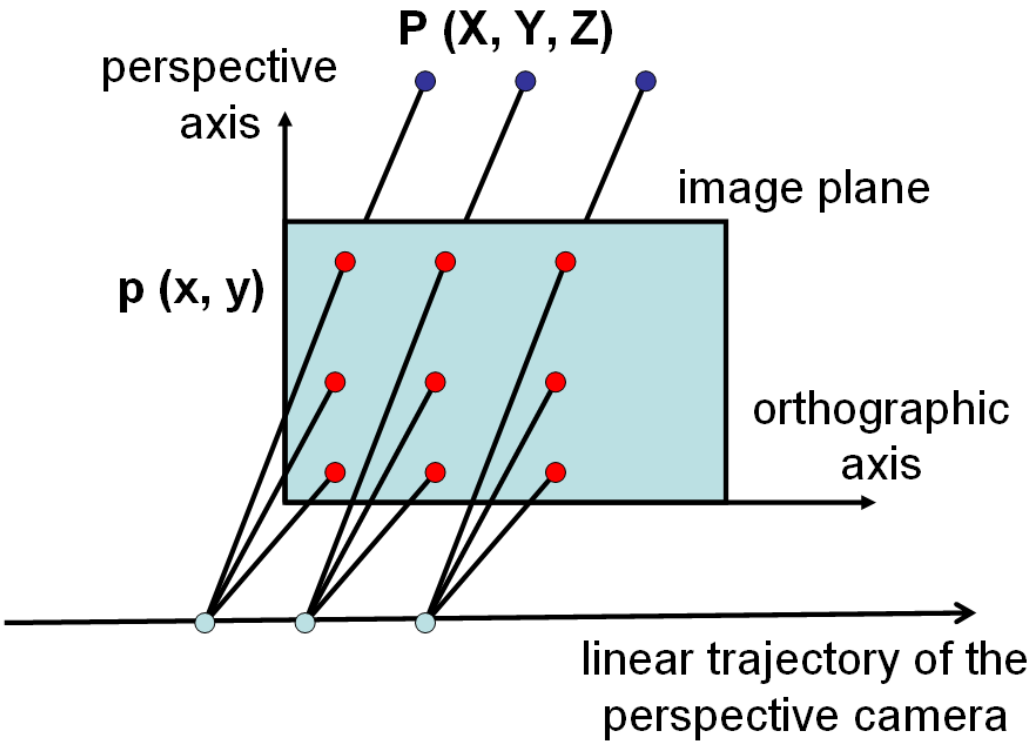


Figure 3.3: Pushbroom camera model. The rays in the pushbroom camera intersects two lines in space, one is the trajectory of the underlying perspective camera used in creating the pushbroom image, the other the line at infinity corresponding to the sampled column in the perspective images.

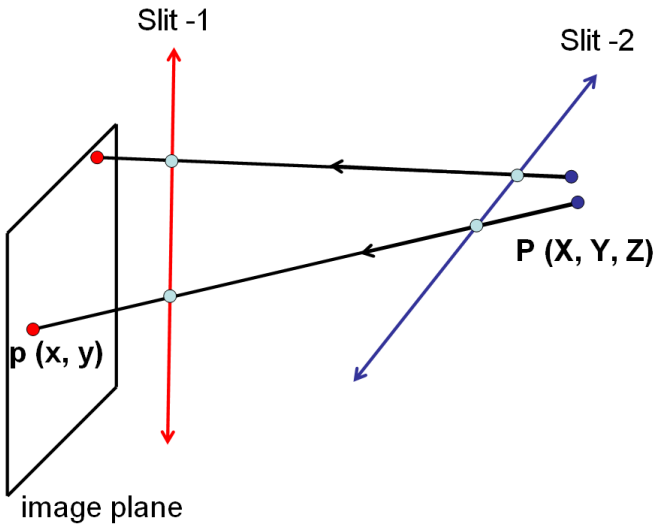


Figure 3.4: Crossed-slit camera model. A 3D point $\mathbf{P}(X, Y, Z)$ gets projected onto a 2D point $\mathbf{p}(x, y)$ through the two slits. The projection of $\mathbf{P}$ depend on the geometry of the two slits and the image plane.

Pushbroom cameras can thus be considered as special cases of crossed-slit cameras. Each of the projection rays in a pushbroom camera lies on one of the parallel planes (cf. Figure 3.3). Each plane consists of coplanar rays corresponding to a single column, which is extracted from one of the original perspective images. These parallel planes intersect at a line at infinity. As a result, each projection ray intersects two lines: the pushbroom axis and the line at infinity.

The projection equation is more complex in a crossed-slit model than the earlier ones. The projection matrix M is a 3×10 matrix that depends on the two slits and the image plane. The 3D point is expressed as a 10-vector using a Veronese mapping [2] as given below.

$$v(\mathbf{P}(X,Y,Z,W)) = \begin{pmatrix} X^3 & XY & XZ & XW & Y^2 & XZ & YW & Z^2 & ZW & W^2 \end{pmatrix}^T$$

$$\mathbf{p} = \mathsf{M}v(\mathbf{P})$$

The back-projected ray for the image point $\mathbf{p}$ is a ray that passes through $\mathbf{p}$ and intersects the two slits.

3.1.5 Rotating imaging model

A conventional camera is rotated about its center of projection to capture several perspective images. These images are then stitched together to obtain an omnidirectional image (c.f Figure 3.5(a)). Several researchers have investigated this approach [19, 55, 60, 128]. The main disadvantage comes from the restriction that the camera has to be precisely positioned and rotated. Secondly the total time taken by the whole process to capture the images and stitch them together is a serious drawback. As a consequence this technique can only be applied to static scenes.

3.1.6 Fisheye model

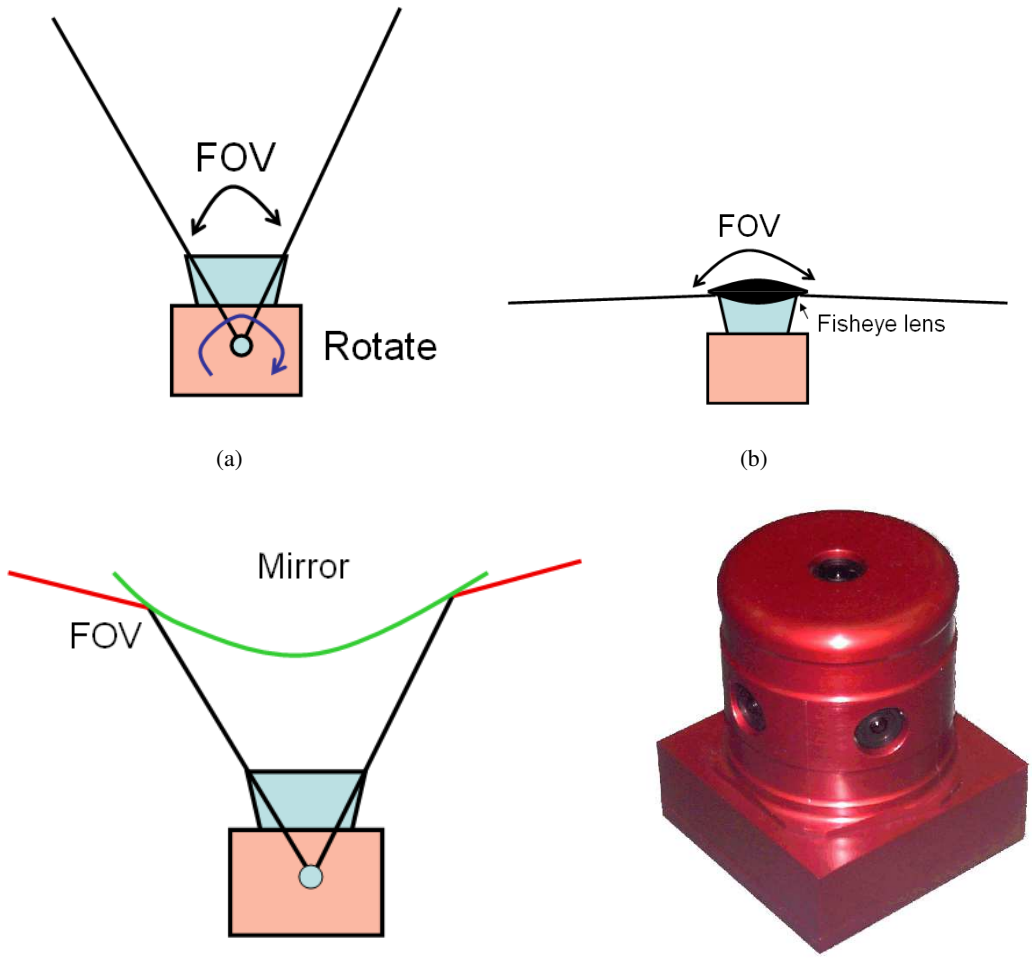
Fisheye lenses have a short focal length and a very large field of view (c.f Figure 3.5(b)). However, when the field of view is greater than 180°, the concept of focal length is not defined. For example, the focal length is not defined for the E8 fisheye lens of Nikon which has a field of view of $183^\circ \times 360^\circ$. Several works have used fisheye lenses for creating omnidirectional images [64, 96, 121]. Geometrically, omnidirectional cameras can be either single viewpoint or non-central. Single viewpoint configurations are preferred to non-central systems because they permit the generation of geometrically correct perspective images from the image(s) captured by the camera. In addition, most theories and algorithms developed for conventional cameras hold good for single-center omnidirectional cameras. In theory, fisheye lenses do not provide a single viewpoint imaging system [67]. The projection rays pass through a small disk in space rather than a single point. Nevertheless in practice it is usually a good assumption to consider these cameras as single viewpoint cameras [125]. Perspective images synthesized from fisheye images are visibly very accurate without any distortions. Some distortion corrections of fisheye images using a single viewpoint assumption are available in chapter 5. Several distortion functions can be used to model fisheye and central catadioptric images [20, 22]. We briefly mention some of them.

- *Stereographic projection:* Several radially symmetric models [34] were used for fisheye images. One of them is the stereographic projection. This model gives a relation between θ, the angle made by a scene point, the optical center and the optical axis, and the distance r between the associated image point and the distortion center.

  $$r = ktan\frac{\theta}{2}$$

  where k is the only parameter to be estimated.

[2] Veronese mappings can be considered as mappings in projective space from one dimension to another. This mapping can be used to reduce certain problems on hypersurfaces (non-linear functions) to the case of hyperplanes (linear functions).



(a) (b) (c) (d)

Figure 3.5: Omnidirectional cameras can be constructed from existing perspective cameras using additional lenses and mirrors. We broadly classify the available construction techniques into four types. (a) Rotation of an imaging system about its viewpoint. (b) Appending a fisheye lens in front of a conventional camera. (c) Using a mirror to image a scene. (d) Ladybug spherical video camera, which has 6 cameras, with five on a horizontal rig and one pointing upwards. Some of the images are adapted from [69].

- *Equidistant projection:*

$$r = k\theta$$

- *Equisolid angle projection:*

$$r = ksin\frac{\theta}{2}$$

- *Sine law projection:*

$$r = ksin\theta$$

On fitting the Nikon FC-E8 fisheye lens with the four radially symmetric models (stereographic, equidistant, equisolid angle and sine law), it was found that the stereographic projection gave the lowest error [6]. The error is the Euclidean distance between the original image pixels and the projected image pixels using the models.

- *Combined stereographic and equisolid angle model:* In [6] Bakstein and Pajdla followed a model fitting approach to identify the right projection model for fisheye cameras. Their model is a combination of stereographic and equisolid angle models. The following relation was obtained with four parameters.

$$r = a\ tan\frac{\theta}{b} + c\ sin\frac{\theta}{d}$$

On the whole they used 13 camera parameters: six external motion parameters $(\mathsf{R}, \mathbf{t})$, one aspect ratio (β), two parameters for the principal point (u_0, v_0) and the four parameters of the above projection model (a, b, c, d).

- *Polynomial lens distortion model:* Most distortion corrections assume the knowledge of the distortion center. Let r_d refer to the distance of an image point from the distortion center. The distance of the same image point in the undistorted image is given by:

$$r_u = r_d(1 + k_1 r_d^2 + k_2 r_d^4 + ...)$$

where k_1 and k_2 are distortion coefficients [13].

- *Field of View (FOV):* The distortion function is given by:

$$r_u = \frac{tan(r_d w)}{2tan\frac{w}{2}}$$

The above distortion correction function is based on a single parameter w. It is a good idea to correct the distortion using the polynomial model followed by the field of view model [22].

- *Division Model (DM):* The distortion correction function is given by:

$$r_u = \frac{r_d}{(1 + k_1 r_d^2 + k_2 r_d^4 + ...)}$$

where the k_i are the distortion coefficients.

3.1.7 Catadioptric model

Vision researchers have been interested in catadioptric cameras because they allow numerous possibilities in constructing omnidirectional cameras [10, 17, 51, 66, 67, 89, 123, 124] (cf. Figure 3.5(c)). The possibilities arise from the differences in size, shape, orientation and positioning of the mirrors with respect to the camera.

Single-viewpoint: Catadioptric configurations allow the possibility of single viewpoint imaging systems. In 1637, René Descartes observed that the refractive and reflective 'ovals' (conical lenses and mirrors) have the ability to focus light into one single point on illumination from a chosen point [21]. Recently Baker and Nayar presented the complete class of catadioptric configurations having a single viewpoint with detailed solutions and degenerate cases [3, 70].

- *Planar mirror:* In the works [3, 70], we observe that by using planar mirrors along with a perspective camera we obtain a single viewpoint configuration. Since planar mirrors do not increase the field of view of the system they are not very interesting for our study. Using four planar mirrors in a pyramidal configuration along with four perspective cameras, Nalwa [67] produced an omnidirectional sensor of field of view of $360^\circ \times 50^\circ$. The optical centers of the four cameras and the angles made by the four planar faces are adjusted to obtain a single effective viewpoint for the system.

- *Conical mirrors:* By positioning the optical center of a perspective camera at the apex of a cone we can obtain a single center configuration. Nevertheless the only light rays reaching the camera after a reflection in the mirror, are those grazing the cone. This case is thus not useful to enhance the field of view while conserving a single center of projection. However, in the work [57], it was proved that conical mirrors can be used to construct a non-degenerate single viewpoint omnidirectional cameras. We briefly describe the principal idea. The outer surface of the conical mirror forms a virtual image corresponding to the real scene behind the conical mirror. On placing the optical center of the pinhole camera at the vertex of the cone, the camera sees the world through the reflection on the outer surface of the mirror. In other words, the cone is not blocking the view. On the other hand, the cone is the view.

- *Spherical mirror:* If the optical center of a perspective camera is fixed at the center of a spherical mirror we obtain a single viewpoint configuration. Unfortunately, all that the perspective camera sees is its own reflection. As a result the spherical mirror produces a degenerate configuration without any advantage. Remember that by positioning the perspective camera outside the sphere we obtain a useful *non-central* catadioptric camera.

- *Parabolic mirror:* Figure 3.6 shows a single viewpoint catadioptric system with a parabolic mirror and an orthographic camera. It is easier to study a catadioptric configuration by considering the back-projection rather than the forward projection. We consider the back-projection of an image point $\mathbf{p}$. The back-projected ray from the image pixel $\mathbf{p}$, starting from the optical center at infinity, is parallel to the axis of the parabolic mirror. This ray intersects and reflects from the surface of the mirror. The reflection is in accordance with the laws of reflection. This reflected light ray is nothing but the incoming light ray from a scene point $\mathbf{P}$ in forward projection. The incoming ray passes through the focus $\mathbf{F}$ if extended on the inside of the mirror. This point where all the incoming light rays intersect (virtually) is called the effective viewpoint. We show an example of a para-catadioptric image in Figure 1.4(a). The physical camera setup is shown in figure 1.3(c).

- *Elliptical mirror:* Figure 3.7 shows a central catadioptric system with an elliptical mirror and a perspective camera. The optical center of the perspective camera is placed at the upper focus of the elliptical mirror. When we back-project an image point $\mathbf{p}$ we observe the following. The back-projected

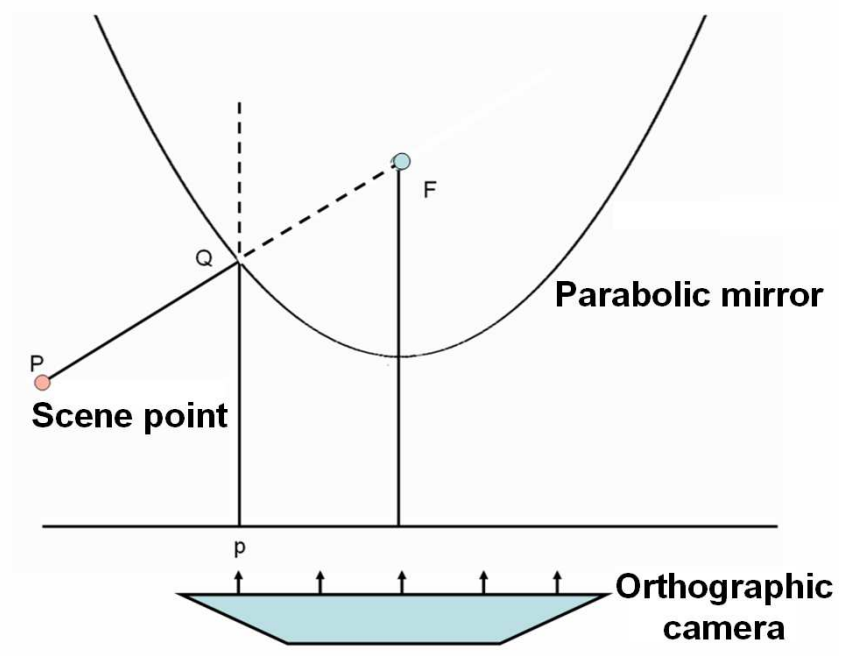


Figure 3.6: Parabolic mirror + orthographic camera [70]. $\mathbf{P}$ refers to the 3D scene point. $\mathbf{F}$, the focus of the parabolic mirror, is the effective viewpoint.

ray, starting from the optical center at the upper focus of the elliptical mirror, intersects and reflects from the surface of the elliptical mirror. The reflected back-projected ray, or the incoming light ray, virtually passes through the lower focus of the mirror. Thus we see that the lower focus ($\mathbf{F}$) is the effective viewpoint of the system.

- *Hyperbolic mirror:* In Figure 3.8 we show a catadioptric system with a hyperbolic mirror and a perspective camera. The optical center of the perspective camera is placed at the external focus of the mirror $\mathbf{F}'$. The back-projected ray of the image point $\mathbf{p}$ start from the optical center, which is the external focus $\mathbf{F}'$ of the mirror, of the perspective camera. Using the same argument as above we observe that the lower focus $\mathbf{F}$ is the effective viewpoint. We show an example of an image captured by this system in Figure 1.4(b). The physical camera setup is shown in figure 1.3(b). The first known work to use a hyperbolic mirror along with a perspective camera at the external focus of the mirror to obtain a single effective viewpoint configuration is [89]. Later in 1995 a similar implementation was proposed in [124].

Non-central catadioptric cameras:
Single viewpoint configurations are extremely delicate to construct, handle and maintain. By relaxing this single viewpoint constraint we obtain greater flexibility in designing novel systems. In fact most real catadioptric cameras are geometrically non-central, and even the few restricted central catadioptric configurations are usually non-central in practice [63]. For example, in the case of para-catadioptric cameras, the telecentric lens is never truly orthographic and it is difficult to precisely align the mirror axis and the axis of the camera. In hyperbolic or elliptic configurations, precise positioning of the optical center of the perspective camera in one of the focal poinnts of the hyperbolic or elliptic mirror is practically infeasible. We briefly mention a few non-central catadioptric configurations. Analogous to the single viewpoint in central cameras, we have a *viewpoint locus* in non-central cameras. It can be defined as follows: a curve or other

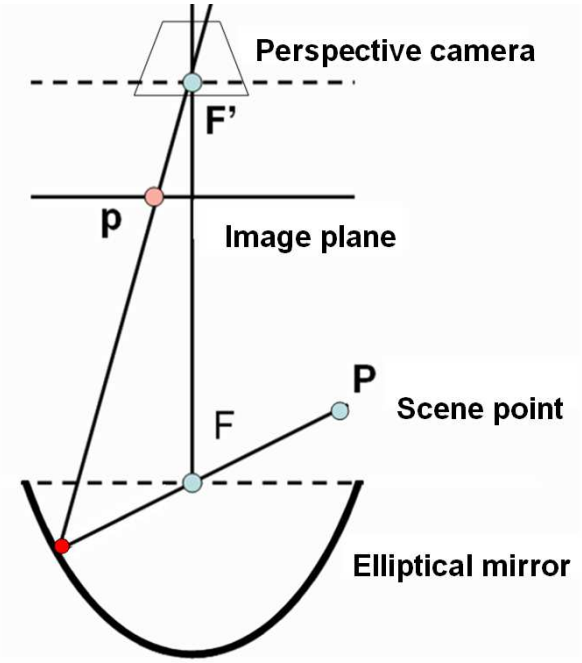


Figure 3.7: Elliptical mirror + perspective camera [3]. $\mathbf{P}$ refers to the 3D scene point. $\mathbf{F}$ and $\mathbf{F}'$ refer to the two focii of the mirror and $\mathbf{p}$ refers to the image point. $\mathbf{F}$ is the effective viewpoint

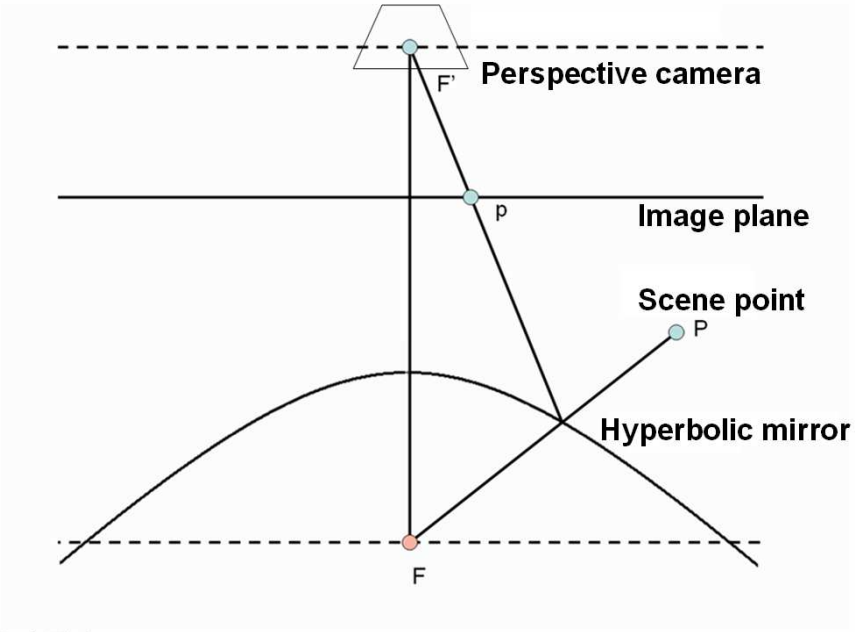


Figure 3.8: Hyperbolic mirror + perspective camera [3]. $\mathbf{P}$ refers to the 3D scene point. $\mathbf{F}$ and $\mathbf{F}'$ refer to the two focii of the mirror and $\mathbf{p}$ refers to the image point. $\mathbf{F}$ is the effective viewpoint.

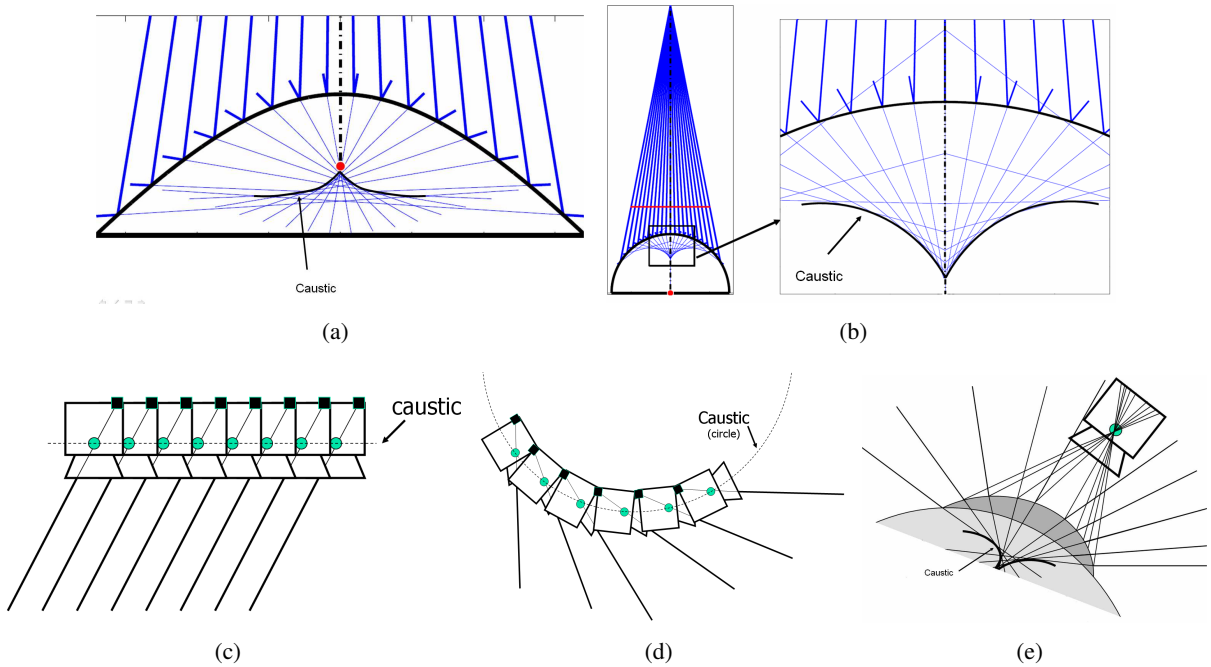
set of points such that all projection rays cut at least one of the points in the viewpoint locus. Usually, one tries to find the "simplest" such set of points.

- *Conical mirror:* On using a conical mirror in front of a perspective camera we obtain an omnidirectional sensor [10, 123]. Nevertheless this configuration does not obey the single viewpoint restriction (besides in the degenerate case of the perspective optical center being located at the cone's vertex). If the optical center lies on the mirror axis, then the viewpoint locus is a circle in 3D, centered in the mirror axis (it can be pictured as a halo over the mirror). An alternative choice of viewpoint locus is the mirror axis. Otherwise, the viewpoint locus is more general.
- *Spherical mirror:* On using a spherical mirror along with a perspective camera we can enhance the field of view of the imaging system [10, 51, 66]. Again this configuration does not obey the single viewpoint restriction (besides in the degenerate case of the perspective optical center being located at the sphere center).
- *Digital micro-mirror array:* Another interesting camera is the recently introduced programmable imaging device using a digital micro-mirror array [71]. A perspective camera is made to observe a scene through a programmable array of micro-mirrors. By controlling the orientations and positions of these mirrors we obtain an imaging system where we have complete control (both in terms of geometric and radiometric properties) over the incoming light ray for every pixel. However there are several practical issues which make it difficult to realize the full potential of such an imaging system. First, current hardware constraints prohibit the usage of more than two possible orientations for each micro-mirror. Second, arbitrary orientations of the micro-mirrors would produce a discontinuous image which is unusable for many image processing operations.

Caustics of catadioptric cameras: In a single viewpoint imaging system the geometry of the projection rays is given by the effective viewpoint and the direction of the projection rays. In a non-central imaging system, the *caustic*, a well-known terminology in the optics community, can be utilized for representing the geometry of projection rays [11]. A caustic refers to the loci of viewpoints in 3D space to represent a non-central imaging system. Concretely, the envelope of all incoming light rays that are eventually imaged is defined as the caustic. A caustic is referred to as diacaustic for dioptric (lens based systems) and catacaustic (mirror based systems) for catadioptric systems. A complete study of conic catadioptric systems has been done [108]. Once the caustic is determined, each point on the caustic represents a light ray by providing its position and the direction. Position is given by the point on the caustic, and orientation is related to the concept of tangent. In Figure 3.9 we show the caustic for several non-central imaging systems. Note that the caustic is same as the viewpoint locus that was already considered in page 3.1.7. For a single viewpoint imaging system the caustic is a degenerate one being a single point. Simple methods exist for the computation of the caustic from the incoming light rays such as local conic approximations [14] and the so-called Jacobian method [15].

3.1.8 Oblique cameras

An ideal example for a non-central camera is an oblique camera. No two rays intersect in an oblique camera [78]. In addition to developing multi-view geometry for oblique cameras Pajdla also proposed a physically realizable system which obeys oblique geometry. The practical system consists of a rotating catadioptric camera that uses a conical mirror and a telecentric optics. The viewpoint locus is equivalent to a two dimensional surface or a set of points, where each of the projection rays passes through at least one of the points.



(a) (b) (c) (d) (e)

Figure 3.9: Caustics for several imaging systems a) Hyperbolic catadioptric system b) Spherical catadioptric system c) Pushbroom camera d) concentric mosaic e) Eye based catadioptric system.

3.1.9 A unifying camera model for central catadioptric systems

In [36], a unifying theory was proposed for all central catadioptric systems with parabolic, elliptical, hyperbolic and planar mirrors. This theory suggests that all catadioptric systems are isomorphic to a projective mapping from a sphere to a plane with a projection center on the perpendicular to the plane. Figure 3.10 shows the unifying projection model. We first identify the two antipodal points on the sphere corresponding to some 3D point ($\mathbf{P}$). This is done by intersecting the sphere by the line joining $\mathbf{P}$ and the center of the sphere $\mathbf{O}$. These points are projected on the horizontal image plane through the projection center ($\mathbf{C}$), lying on the vertical axis of the sphere. The projection of 3D points on to the image plane in the above four central catadioptric configurations can all be described using this model. The projection expressions mainly depend on two parameters.

- Distance between the projection center $\mathbf{C}$ and the center of the sphere $\mathbf{O}$
- Distance between the image plane and the center of the sphere $\mathbf{O}$

The nature of the mirror (parabolic, hyperbolic, elliptic, planar) and type of camera (perspective, orthographic) decide the value of these parameters. For example in the planar case with a perspective camera, the projection center is the center of the sphere. In a parabolic scenario the projection center is the north pole of the mirror. Ying and Hu [125] proposed a unifying model, which is basically an extension of [36], to unify fisheye lenses along with the central catadioptric systems. The unifying model is based on a projection with a more general quadratic surface, in contrast to the spherical surface in [36].

3.1.10 Radially symmetric camera model

Tardiff and Sturm [112, 113] introduced a generic radially symmetric model. Radially symmetric images are modelled using a unique *distortion center* and concentric distortion circles centered about this point

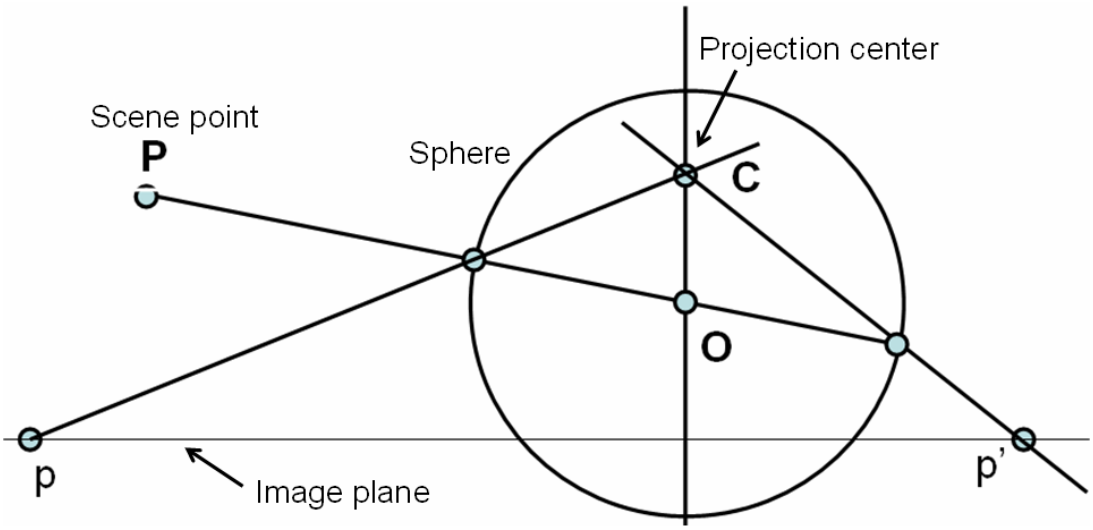


Figure 3.10: Unifying model for central catadioptric systems

as shown in Figure 3.11(a). The projection rays of pixels lying on a specific distortion circle form a right viewing cone (see Figure 3.11(b)). These viewing cones may have different focal lengths (opening angles) and optical centers. However all the optical centers lie on the same line – the optical axis – which also corresponds to the axis of all viewing cones. As shown in Figure 3.11(b), we assume the z axis to be the optical axis. The viewing cones intersect the xy plane in concentric circles. Variables d_i are used to parameterize the positions of optical centers and, together with radii r_i, the focal lengths. Note that this model is sufficiently general to include both central and non-central radially symmetric cameras.

For simplicity we translate the coordinate system of the image such that the distortion center becomes the origin. Let us consider a pixel on the distortion circle with radius $\breve{r}$ at an angle θ. Its location can be specified by $(\breve{r}cos(\theta), \breve{r}sin(\theta))$. The corresponding 3D ray in the viewing cone is specified by the optical center $(0, 0, d)$ and the point $(rcos(\theta), rsin(\theta), 0)$ in the xy plane.

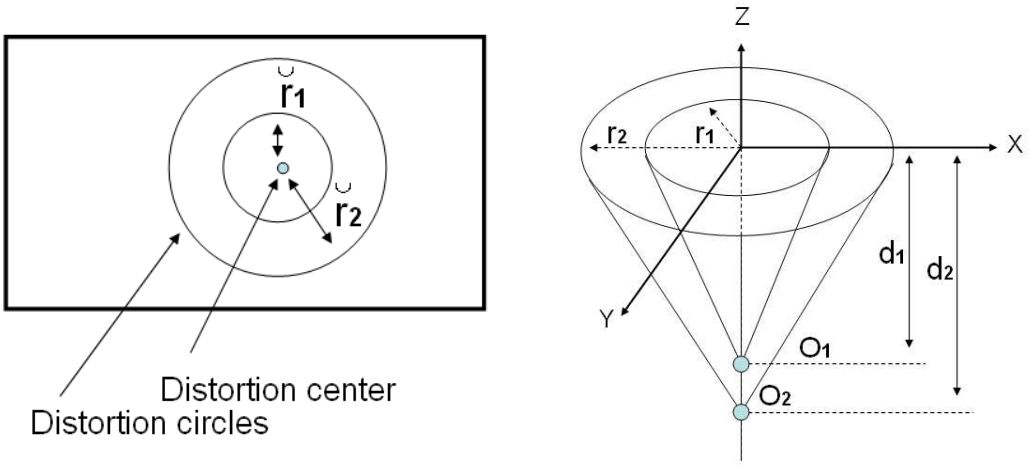


Figure 3.11: Radially symmetric camera model. Left: Distortion center and two distortion circles. Right: Corresponding viewing cones. They may have different vertices (optical centers) and opening angles.

3.1.11 1D radial model

The 1D radial model of [114] generalizes the central radially symmetric model. It consists of an optical axis and a distortion center. The only requirement on the projection is that all points in a plane that contains the distortion center. In [114], it is shown that the relation between such 3D planes π and 2D lines l can be expressed as a 2×4 projection matrix.

$$\lambda\pi = P^T l$$

3.1.12 Rational function model

A rational function model was proposed by Claus and Fitzgibbon for radial lens distortion in wide angle catadioptric lenses [20]. In general, the mapping between image coordinates and 3D rays is highly non-linear for omnidirectional images created using fisheye lenses and mirrors. A linear mapping between the 'lifted' image (pixel) coordinates (non-linear functions of the image coordinates) and 3D rays is proposed in this work. Lifted image coordinates have been previously used for computing the back-projection matrices for para-catadioptric cameras [37, 100].

We briefly explain the lifting strategy that is used in the rational model. A 2D image point $\mathbf{p}(x, y)$ is lifted to a 6-vector $\chi(x, y)$ as follows.

$$\chi(x, y) = \begin{pmatrix} x^2 & xy & y^2 & x & y & 1 \end{pmatrix}^T$$

The imaging model, describing the relation between an image pixel $p(x, y)$ and its 3D ray is given using a 3×6 matrix A.

$$\mathbf{P}(\lambda) = \lambda \mathsf{A} \chi(x, y)$$

The imaging model already describes the back-projection of the image point to its corresponding 3D ray. Refer to [20] for details about forward projection, where the 2D point $\mathbf{p}(x, y)$ is computed corresponding to a 3D point $\mathbf{P}(X, Y, Z)$, using the relation $\mathbf{P} = \mathsf{A}\chi(x, y)$.

3.1.13 Generic imaging model

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 tailor-made for specific camera models.

The aim of our work is to relax the last two 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 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 system).
- the mapping between pixels and rays; this is basically a simple indexing.

The generic imaging model is shown in Figure 3.12. The non-parametric nature of this model adds one difficulty: how to compute 3D rays for an image point with non-integer image coordinates. To do so, the only possibility is to add continuity assumptions, e.g. that neighboring pixels have neighboring 3D rays. Under this or more restrictive assumptions, 3D rays for arbitrary image points can be computed by interpolation. Similarly, the projection of 3D points onto images, is not straightforward, but can for example be solved analogously, by interpolation. For the time being, we do not address these issues in more detail. For more details refer to section 4.2.

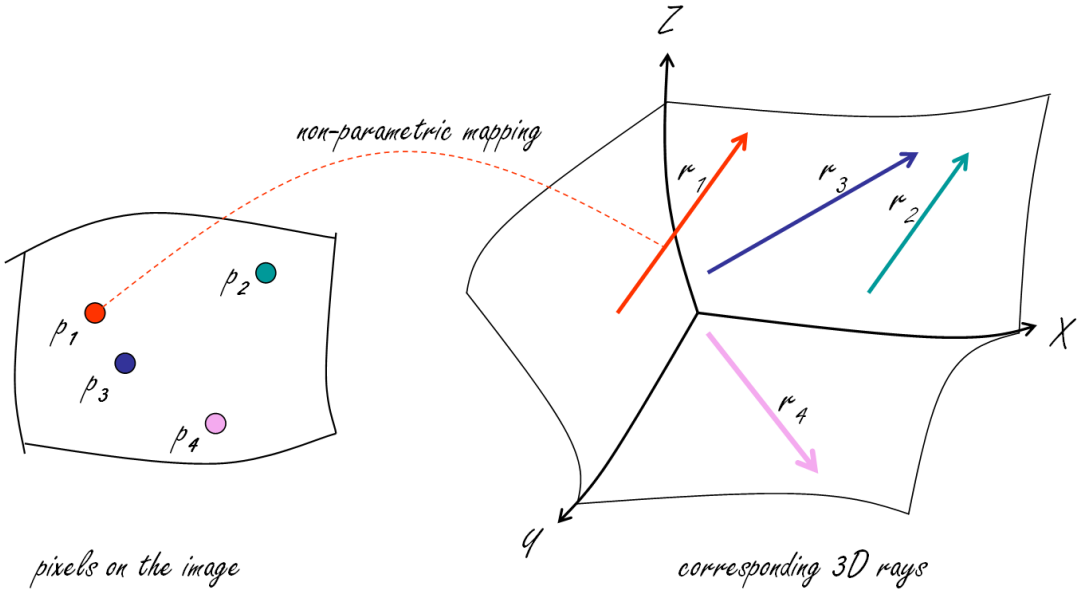


Figure 3.12: The main idea behind the generic imaging model: The relation between the image pixels $(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_n)$ and their corresponding projection rays $(\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3, \mathbf{r}_n)$ is *non-parametric*.

The above general camera model allows to describe all above models and virtually any camera that captures light rays traveling along straight lines[3]. Examples:

- 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 [50]. Such systems, as opposed to central catadioptric systems [3] composed of cameras and parabolic mirrors, do not in general have a single 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) may be considered as a **single** camera.
- other acquisition systems, see e.g. [5, 79, 95], or eyes of some insects.

Relation to previous work. The above imaging model has already been used, in more or less explicit form, in various works [41, 73, 78, 79, 81, 91, 95, 108, 119, 120], and is best described in [41]. 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 [56] or lumigraph [40] may be interpreted as a single image, acquired by a camera of an appropriate design.

Taxonomy of generic imaging models. In this thesis we consider the problems of calibration and generic structure-from-motion algorithms for generic imaging models. The focus is to develop algorithms in a

[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 by any motion that is not a pure translation parallel to the water's surface, the refraction effect causes the set of rays to not move rigidly, hence the calibration would be different for each camera position.

generic manner, i.e., they need to be camera independent and applicable to all the cameras in the same manner. Nevertheless, we observed three important classes of cameras which require independent algorithms, or more precisely, modifications in the general algorithm. The differences arise from the formulation of constraints and estimation of tensors by solving linear systems. For example the theory developed for generally non-central cameras produces ambiguous solutions for central cameras. The three classes are referred to as central, axial and non-central [101]. We briefly explain each one of these classes.

Central model

All the projection rays go through a single point, the optical center. Examples are mentioned below:

- The conventional perspective camera forms the classical example for a central camera
- perspective+radial or decentering distortion
- Central catadioptric configurations using parabolic, hyperbolic or elliptical mirrors.
- Fisheye cameras can be considered as approximate central cameras.

Axial model

All the projection rays go through a single line in space, the *camera axis*. Examples of cameras falling into this class are:

- stereo systems consisting of 2, 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 perspective), lies on its axis of revolution. It is easy to verify that in this case, all the 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 ones.
- x-slit cameras [31] (also called two-slit or crossed-slit cameras), and their special case of linear push-broom cameras [42].

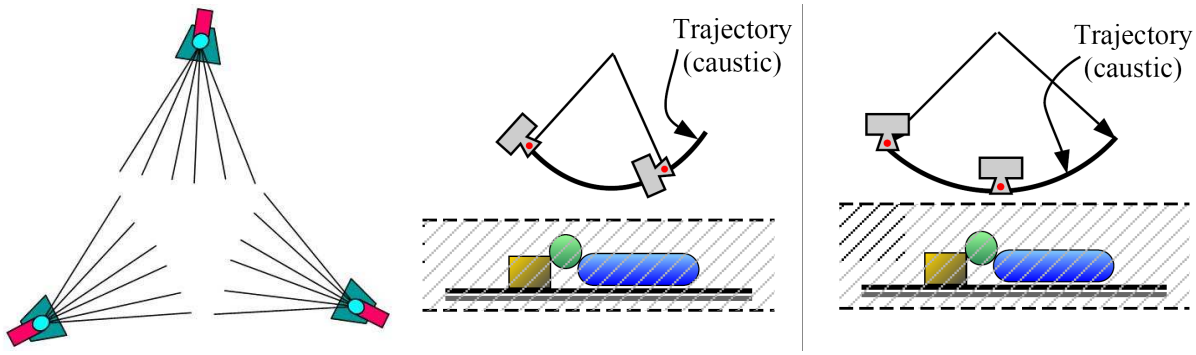
Non-central cameras

A non-central camera may have completely arbitrary projection rays. Common examples are given below:

- multi-camera system consisting of 3 or more cameras, all of whose optical centers are not collinear.
- Oblique camera: This is an ideal example for a non-central camera. No two rays intersect in an oblique camera [78].
- Imaging system using a micro-mirror array [71]. A perspective camera is made to observe a scene through a programmable array of micro-mirrors. By controlling the orientations and positions of these mirrors we obtain an imaging system where we have complete control (both in terms of geometric and radiometric properties) over the incoming light ray for every pixel.
- Non-central mosaic: An image sequence is captured by moving the optical center of a perspective camera in a circular fashion as shown in Figure 3.13(b) [108]. The center columns of the captured images are concatenated to create a non-central mosaic image.

- Center strip mosaic: The optical center of the camera is moved as shown in Figure 3.13(c) [108]. The center columns of the captured images are concatenated to form a center strip mosaic. The resulting mosaic corresponds to a non-central camera.

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 the projection rays: $d = 0$ defines central, $d = 1$ axial and $d = 2$ general non-central cameras.



(a) (b) (c)

Figure 3.13: Examples of non-central camera models: (a) multi-camera scenario (b) concentric mosaic (c) center-strip camera

3.2 Perspective cameras

3.2.1 Calibration

The process of computing the intrinsic matrix (K) along with the pose ($\mathsf{R}, \mathbf{t}$) of the camera is called camera calibration. Concretely the estimation of K is termed intrinsic calibration and the estimation of pose $(\mathsf{R}, \mathbf{t})$, the extrinsic calibration. Sometimes we also model radial or other distortions using additional parameters. Depending on the nature of the information used there are two main approaches for camera calibration.

Pre-calibration

The first approach focuses on calibrating the camera by taking images of a known physical object, referred to as a calibration grid [30, 103, 118, 127]. While using this approach the cameras have to be calibrated before we use them for an actual task. Consequently we can not change the camera parameters (like focal length, zoom, etc) while capturing the images.

Self-calibration

The second technique *self-calibration*, which does not need the use of calibration grids, calibrates using images of unknown scenes [82]. Nevertheless, one of the main issues is the critical motion sequences [97, 98], during which the camera can not be self-calibrated. In other words, the basic idea is that some motion sequences do not allow self-calibration. For example, in the case of pure translation, pure rotation and planar motion of the camera, self-calibration is ambiguous. Self-calibration depends on the constraint that there is only one possible reconstruction consistent with both the image sequences and a priori constraints on the internal parameters. However in the case of critical motion sequences, there are at least two possible reconstructions of the scene that satisfy all the constraints.

3.2.2 3D reconstruction

In this section we focus on the problem of obtaining depth information (3D coordinates) from a sequence of images.

Factorization: An affine factorization based approach was taken by Tomasi and Kanade in [115], where 3D information was extracted from a sequence of images for an orthographic projection case. In this method, image correspondences across multiple views are used to construct a so-called measurement matrix. The decomposition of this matrix directly provides the 3D structure and motion. Nevertheless the assumption of orthographic projection is a serious limitation. Later in 1996, Sturm and Triggs proposed an algorithm for obtaining projective structure and motion from perspective cameras using a factorization based method [106].

Epipolar geometry: Uncalibrated image sequences can be used to produce a reconstruction up to an arbitrary projective transformation [27, 32, 46]. To understand projective transformations one should first understand projective space. In an n dimensional projective space $\mathbb{P}^n$ every point is represented in homogeneous coordinates ($(n+1)$-vector). In $\mathbb{P}^3$ a projective transformation is a linear transformation $\mathsf{H}_{4\times4}$ that transforms a 4-vector $\mathbf{X}$ to another 4-vector $\mathbf{X}'$.

$$\mathbf{X}' = \mathsf{H}_{4\times4}\mathbf{X}$$

Projective geometry is an extension of Euclidean geometry. In a projective space every pair of lines always intersect at a point. Projective transformations do not preserve sizes or angles but do preserve incidence and cross-ratio. More details about projective geometry, cross-ratio and ideal points can be found in [49]. The important point to remember here is that a projective reconstruction is the best result that one can achieve without camera calibration or metric information about the scene. In other words, the reconstruction is only known up to an unknown 4×4 non-singular projective transformation matrix. Using ground control points the reconstruction can be upgraded to a Euclidean reconstruction and the camera parameters can be computed [46].

The epipolar constraint [49] is a useful geometric constraint that exists between a pair of images captured by a camera from different viewpoints. We briefly explain the concept of epipolar geometry. Consider Figure 3.14(a). The underlying idea is that for any point in the first image, its corresponding point in the other image lies on a line called the epipolar line. Let the points $\mathbf{p}$ and $\mathbf{p}'$ be the corresponding points in the images I anad I' respectively. The two points correspond to the same physical point $\mathbf{P}$. Let $\mathbf{C}$ and $\mathbf{C}'$ denote the optical centers of the two cameras respectively. The point of intersection of the line connecting $\mathbf{C}$ and $\mathbf{C}'$ and the image plane I is called the epipole $\mathbf{e}$. Similarly $\mathbf{e}'$ is the epipole in I'. It is easy to visualize $\mathbf{C}$, $\mathbf{C}'$, $\mathbf{p}$, $\mathbf{p}'$ and $\mathbf{P}$ on a single plane, i.e., they are all coplanar. We refer to this plane as *epipolar plane*. This coplanarity property results in the following constraint, commonly referred to as the *epipolar constraint*. For every point $\mathbf{p}$ in the first image, its corresponding point $\mathbf{p}'$ can only lie on the so-called *epipolar line* of the point $\mathbf{p}$ in the second image. The epipolar line is nothing but the intersection of the epipolar plane with I'. All the epipolar lines pass through the epipole in an image. This arises from the well-known *co-planarity* constraint.

The epipolar constraint can be represented in terms of a 3×3 rank 2 matrix called Fundamental matrix. Let K and K' be the intrinsic matrices of the first and the second cameras respectively. The motion between the first and second camera is $(\mathsf{R}, \mathbf{t})$, where R is the 3×3 rotation matrix with three degrees of freedom and $\mathbf{t}$ is the 3×1 translational matrix. Without loss of generality we assume that $\mathbf{P}$ is expressed in the coordinate system of the first camera. Under the assumptions of the perspective model we get the following equations.

$$s\mathbf{p} = \mathsf{K}\begin{bmatrix} \mathsf{I} & \mathbf{0} \end{bmatrix}\mathbf{P}$$

$$s'\mathbf{p}' = \mathsf{K}'\begin{bmatrix} \mathsf{R} & -\mathsf{R}\mathbf{t} \end{bmatrix}\mathbf{P}$$

where s and s' are the scale factors used to remove the scale ambiguity in Equation 3.1. We eliminate $\mathbf{P}$, s and s'.

$$\mathbf{p}'^{T}\mathsf{K}'^{-T}[\mathbf{t}]_{X}\mathsf{R}\mathsf{K}^{-1}\mathbf{p} = 0$$

where $[\mathbf{t}]_{\times}$ is an antisymmetric matrix. The *essential matrix* E, which is the algebraic centerpiece of the classical motion estimation problem, was developed in [58]. The fundamental matrix, essential matrix and their relations are given below.

$$\mathsf{F} = \mathsf{K}'^{-T}[\mathbf{t}]_{\times}\mathsf{R}\mathsf{K}^{-1} \tag{3.2}$$

$$\mathsf{E} = [\mathbf{t}]_{\times}\mathsf{R} \tag{3.3}$$

$$\mathsf{F} = \mathsf{K}'^{-T}\mathsf{E}\mathsf{K}^{-1}$$

We can observe that these matrices depend only on the configuration of the cameras (intrinsic parameters, position and orientation). The Fundamental matrix is a rank-2 homogeneous matrix with 7 degrees of freedom. There are several methods to estimate the fundamental matrix, and the simplest method is the 8-point algorithm, which we briefly describe below. Let

$$\mathbf{p} = \begin{pmatrix} u \\ v \\ 1 \end{pmatrix}, \ \mathbf{p}' = \begin{pmatrix} u' \\ v' \\ 1 \end{pmatrix}$$

We expand Equation(3.2).

$$uu'f_{11} + u'vf_{12} + u'f_{13} + v'uf_{21} + v'vf_{22} + v'f_{23} + uf_{31} + vf_{32} + f_{33} = 0$$

We denote the 9-vector from F by $\mathbf{f}$.

$$\begin{pmatrix} u'u & u'v & u' & v'u & v'v & v' & u & v & 1 \end{pmatrix}\mathbf{f} = 0$$

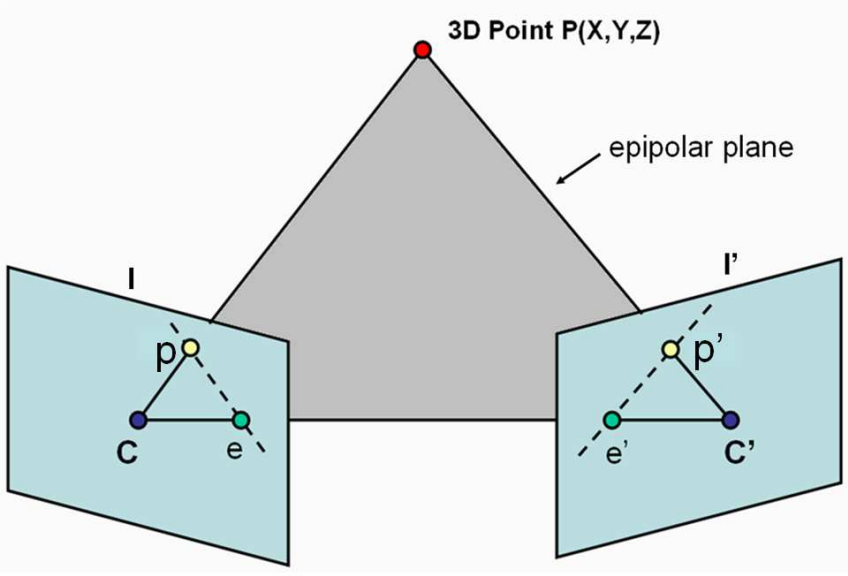
In the presence of n feature matches we have the following linear system.

$$\begin{pmatrix} u'_1u_1 & u'_1v_1 & u'_1 & v'_1u_1 & v'_1v_1 & v'_1 & u_1 & v_1 & 1 \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ u'_nu_n & u'_nv_n & u'_n & v'_nu_n & v'_nv_n & v'_n & u_n & v_n & 1 \end{pmatrix}\mathbf{f} = \mathbf{0}$$

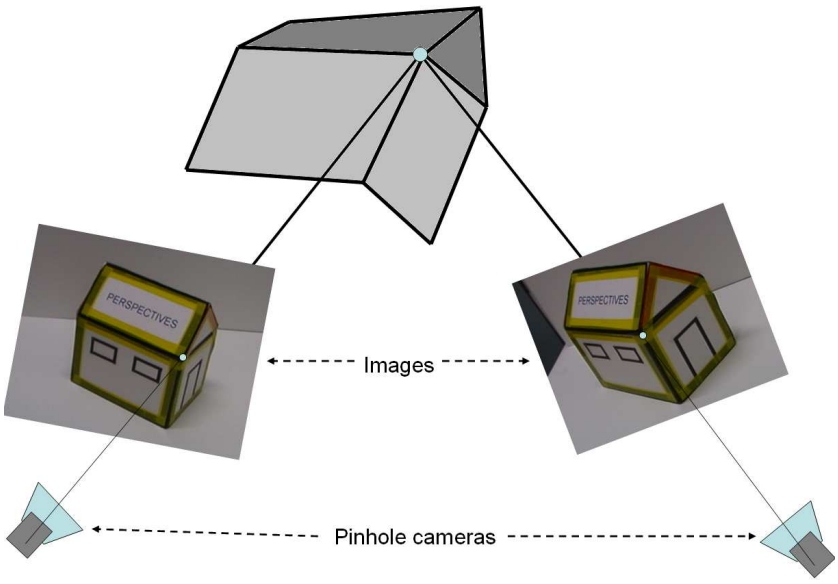
$$\mathsf{A}_{n\times 9}\mathbf{f} = \mathbf{0}$$

From the above homogeneous set of equations, we can extract $\mathbf{f}$ up to a scale. The rank of A must be atmost 8 to have a solution to $\mathbf{f}$. If the rank of A is 8 then the solution to A is the singular vector corresponding to the smallest singular value of A. In other words, the solution is the last column of V in the singular value decomposition of A ($\mathsf{A} = \mathsf{U}\mathsf{D}\mathsf{V}^{T}$). The minimum number of points required to solve the fundamental matrix is 7. Several algorithms exist for the robust computation of F. For more details please refer to [49]. In order to obtain a 3D reconstruction we need to find the projection matrices used in the two images. M and M' are the two projection matrices. Once we have the projection matrices, we can proceed to compute the 3D coordinates of the points. Using the perspective model we have the following equations.

$$s\mathbf{p} = \mathsf{M}\mathbf{P},$$



(a)



(b)

Figure 3.14: (a) Epipolar geometry for a pair of images. For an image pixel $\mathbf{p}$ we show its corresponding epipolar line in the second image. (b) We show the triangulation process in obtaining a 3D model from a pair of images.

$$s'\mathbf{p}' = \mathsf{M}'\mathbf{P}$$

We denote the vectors corresponding to the i^{th} row of M and M' by m_i and m_i' respectively. The scalars s and s' can be represented in terms of the projection matrices (M and M') and $\mathbf{P}$.

$$s = \mathbf{m}_3^T P,$$

$$s' = \mathbf{m}_3'^T P'$$

After eliminating s and s' from the equations we will have $AP = 0$ where

$$A = \begin{bmatrix} \mathbf{p}_1 - u_1\mathbf{p}_3 \\ \mathbf{p}_2 - v_1\mathbf{p}_3 \\ \mathbf{p}_1' - u_2\mathbf{p}_3' \\ \mathbf{p}_2' - v_2\mathbf{p}_3' \end{bmatrix}$$

$\mathbf{P}$ can be found up to a scale factor by finding the eigenvector of the matrix $\mathbf{A}^T\mathbf{A}$ associated with the smallest eigenvalue. If M and M' are known upto a projective transformation, then $\mathbf{P}$ is computed upto a projective transformation.

Analogous to epipolar geometry, which is a bilinear constraint between two views, there is a trilinear constraint which relates corresponding image points in three images [93]. Following this work a systematic study on the constraints on both lines and points in any number of views was done [29, 116].

3.3 Omnidirectional cameras

Omnidirectional cameras, having very large field of view, are useful for vision applications such as autonomous navigation, video surveillance, video conferencing, augmented reality and site modeling. There are several ways to construct omnidirectional cameras. Geometrically, omnidirectional cameras can be either single viewpoint or non-central. Single viewpoint configurations are preferred to non-central systems because they permit the generation of geometrically correct perspective images from the image(s) captured by the camera. In addition, most theories and algorithms developed for conventional cameras hold good for single-center omnidirectional cameras.

3.3.1 Calibration

Similarly to the perspective camera, there are two groups of calibration algorithms for omnidirectional cameras. The first group uses some knowledge about the scene such as provided by calibration grids or plumb-lines (straight line patterns).

Pre-calibration

- *calibration grids (images of known points):* Views of the calibration grids are first captured. Then the camera is calibrated by computing the motion between the grids and computing the parameters for the specific camera model [6, 9, 92].

  Shah and Agarwal developed a Lagrange minimization based calibration method for fisheye lenses as to compute effective focal length, pixel size, radial and tangential distortions [92]. The distortions were modeled by a polynomial of degree 5 and the optical center is computed using a laser beam.

  Hartley and Kang [47] proposed a plane-based calibration algorithm for correcting radial distortion in non-parametric radially symmetric model. The method renounces on parametric distortion models.

In contrast to many calibration approaches, they don't assume the knowledge of the distortion center. Their method can estimate the center of radial distortion using calibration grids.

Tardif and Sturm proposed a calibration algorithm for radially symmetric cameras (see section 3.1.10) using planar calibration grids [112]. The calibration essentially computes the focal lengths of the individual viewing cones corresponding to different distortion circles. Two different calibration approaches were proposed. The first approach is based on a geometrical constraint that links the different viewing cones and the associated ellipses in the calibration plane. The second approach is based on existing plane-based calibration methods to directly calibrate the individual viewing cones. The second approach gave more stable results compared to the first one.

- *Plumb line methods (images of known lines):* In non-perspective images lines appear as curves. Plumb line methods calibrate (estimate the parameters) by correcting the curves to straight lines [13, 35, 38, 111].

  Geyer and Daniilidis calibrated para-catadioptric cameras to obtain all intrinsic parameters from the images of three lines without any metric information [35, 38]. The paracatadioptric model is described in section 3.1.7 (cf. Figure 3.6). In central para-catadioptric cameras the lines in 3D scenes get imaged as circles in images. Note that the images are circles only if the orthographic camera has aspect ratio 1 and zero skew. By observing three or more 3D lines in the corresponding para-catadioptric images the calibration parameters are computed. Three 3D lines are observed in an image and calibration is performed without any metric information about the environment [36]. The authors have proved that this information is highly insufficient to calibrate a conventional camera and thereby suggest one other advantage of central catadioptric cameras compared to conventional ones. Barreto and Araujo extended this algorithm to all catadioptric cameras and proposed a calibration algorithm [7].

  Recently Tardif and Sturm [113] also proposed a plumbline based calibration technique for the same model. Linear constraints are formulated from the images of line patterns to extract the individual focal lengths up to a global focal length, which is already sufficient for image rectification.

Self-calibration

The second group of calibration methods do not use any knowledge about the scene [25, 26, 39, 53, 122].

Kang calibrated para-catadioptric cameras without any of the following: calibration object, knowledge of camera motion and any knowledge of the scene structure [53]. He minimized the point distances to their corresponding epipolar curves to obtain the calibration parameters. The concept of epipolar lines manifest to epipolar curves in central omnidirectional cameras. We describe epipolar curves in section 3.3.2.

Conventional camera self-calibration algorithms compute radial distortion at the final stage, i.e. bundle adjustment. Fitzgibbon used a division model, given in section 3.1.6, and showed how to estimate simultaneously the perspective epipolar geometry and one radial distortion coefficient. Previously, Zhang [126] addressed that problem, but offered a more complicated solution. This approach is not very effective for omnidirectional images with very large field of view of about $180^\circ \times 360^\circ$, for a single parameter is not sufficient to capture the large distortion. The work by Fitzgibbon was generalized in [62], where an appropriate omnidirectional camera model incorporating lens distortion was used to estimate the parameters from epipolar geometry.

In [63], Micusik and Pajdla auto-calibrate catadioptric cameras with parabolic, hyperbolic and spherical mirrors. The catadioptric models are described in section 3.1.7. First point matching is done using a central approximation followed by auto-calibration from epipolar geometry. The internal calibration parameters used in the calibration algorithm include the center of radial symmetry (u_0, v_0), the principal point (assumed to be the radial distortion center), mirror shape parameters, pose of the perspective camera $(\mathsf{R}, \mathbf{t})$ and its focal length(f).

Thirthala and Pollefeys [114] propose a method for recovering radial distortion which can also include non-central cameras. The model is described in section 3.1.11. Two different calibration approaches are proposed. The first is based on 15 point correspondences from 4 images, that works for both central and non-central cameras. The second approach uses 7 point correspondences in three images and works only for central cameras. The second method is demonstrated to be well suited for automatic calibration of fisheye images and central catadioptric cameras. Both approaches use non-parametric modeling for camera calibration. First the scene is reconstructed and the reconstructed scene is used as the calibration object for performing non-parametric camera calibration.

In the presence of restricted camera motions or planar scenes, Tardif and Sturm [113] demonstrate self-calibration from images of a completely unknown scene for radially symmetric camera model (see section 3.1.10).

3.3.2 3D reconstruction

Epipolar geometry:

- *Single viewpoint cameras:* The concept of epipolar geometry for central panoramic cameras is exactly the same as for perspective cameras when we only consider the projection rays in 3D. A single projection ray from the first camera, and a set of projection rays from the second camera, all lie on the same epipolar plane. The difference between the perspective and other central cameras arises when we consider their images. The epipolar lines in perspective cameras manifest themselves as epipolar curves in panoramic images because of the non-linear mapping between the image and rays. The central omnidirectional cameras and other non-classical cameras have extended this mapping from lines to other conics (circles, ellipses, parabolas, hyperbolas, etc.). A study on the epipolar geometry for a restricted case of pure rotation of a hyperbolic mirror around the center of a perspective camera was done by Nene and Nayar [72]. This also focuses on getting the epipolar geometry for the case of a parabolic mirror in the case of pure translation of the orthographic camera.

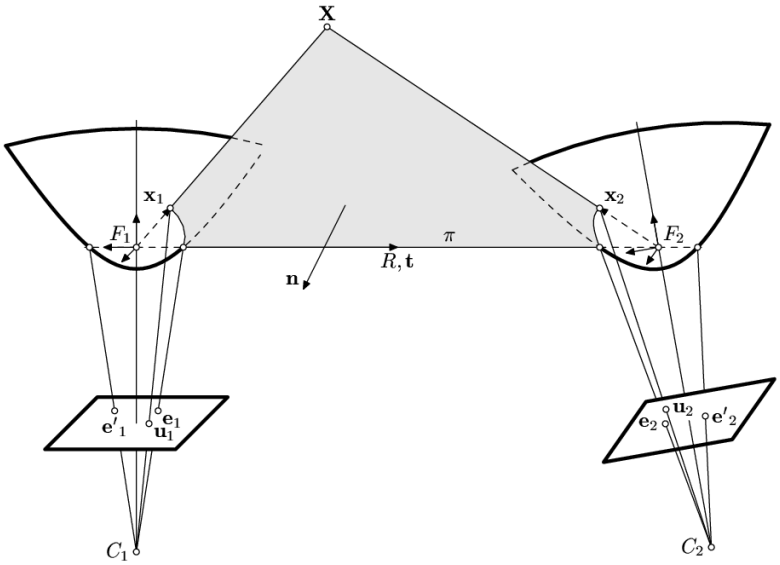
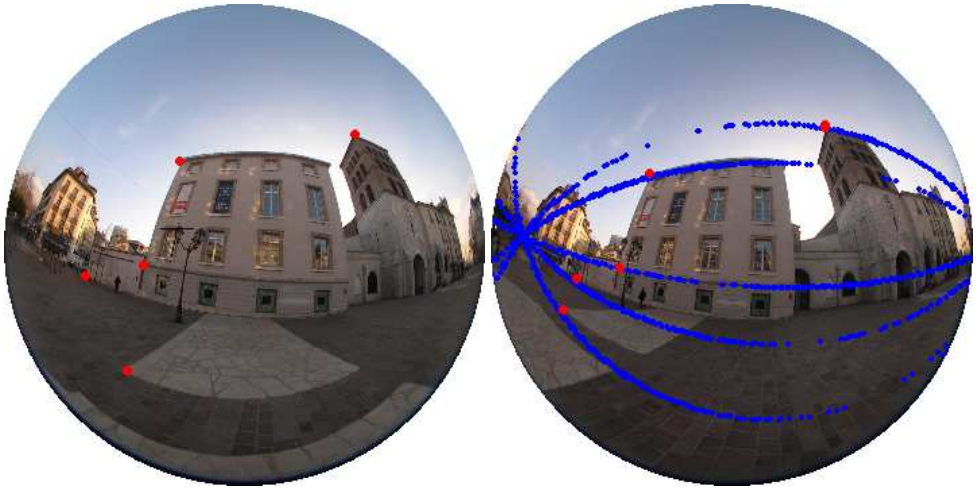


Figure 3.15: The epipolar geometry of two central catadioptric cameras with hyperbolic cameras. This image is adapted from [107].

In Figure 3.15, we show the epipolar geometry for two central catadioptric cameras with hyperbolic mirrors [107]. Let $(\mathsf{R}, \mathbf{t})$ be the motion between the two coordinate systems $\mathbf{F}_1$ and $\mathbf{F}_2$ and let $\mathbf{p}$



(a) (b)

Figure 3.16: Epipolar curves are shown for a pair of fisheye images.

and $\mathbf{p}'$ be the corresponding image points observing the same 3D point $\mathbf{P}$. The corresponding points on the mirror are given by $\mathbf{x}_1$ and $\mathbf{x}_2$. $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{t}$ form the epipolar plane Π. There are many differences in the epipolar relationship for perspective cameras and the central catadioptric cameras. The shape of the conics (lines, circles, ellipses, parabolas or hyperbolas) depends on the shape of the mirrors, the relative position of the cameras, and on the point considered in the image. Algebraically the fundamental constraint on the corresponding points between two central catadioptric images is given below.

$$\mathbf{p}'^{T}\mathsf{A}_2(\mathsf{E},\mathbf{p})\mathbf{p}' = 0$$

where $\mathsf{A}_2(\mathsf{E},\mathbf{u}_1)$ is a nonlinear function of the essential matrix E, the point $\mathbf{p}_1$, and the calibration parameters of the central catadioptric camera. In contrast to a single point in perspective cameras all epipolar conics pass through two points which are the images of the intersections of the mirrors with the line $\mathbf{F}_1\mathbf{F}_2$ (line joining the effective viewpoints). These points, denoted by $\mathbf{e}_1$ and $\mathbf{e}_1^{'}$, are the two epipoles of the first camera. Similarly we have two epipoles $\mathbf{e}_2$ and $\mathbf{e}_2^{'}$ for the second camera. In Figure 3.16 we show the epipolar curves for a pair of fisheye images, which are calibrated using the generic calibration algorithm given in chapter 5.

In [100] the epipolar geometry is derived for combinations of para-catadioptric, perspective and affine cameras. Note that in these cases, the matrix A_2 in the epipolar constraint does not depend on $\mathbf{P}$ anymore. However, the epipolar constraint can be written in the usual bilinear form only if lifted coordinates are used.

- *Non-central cameras:* The most general non-central camera was introduced in [78]. In an oblique camera no two rays intersect each other. The notion of epipolar geometry becomes more complex for oblique cameras. Every ray in one camera is associated with a set of rays of the other camera which form a *doubly ruled quadric*, which we briefly describe now. First, a quadric is nothing but a second order surface. Depending on the nature of coefficients in the second order equation we have different types of quadrics such as sphere, ellipsoid and hyperboloid. A doubly ruled quadric is a special type of second order surface where every point on the surface lies on two distinct lines which also lie on the surface. The plane, the hyperbolic paraboloid and the hyperboloid of one sheet are the only doubly ruled quadric surfaces. In an oblique camera epipoless do not exist. We briefly explain the reason

for such a behaviour. The definition of epipoles for central cameras is: they are the images of the optical center of the respective other camera. This can be extended to non-central cameras: images of the 'viewpoint locus' of the respective other camera. However, epipoles are no longer (sets of) points here, but they are curves. For oblique cameras however, it was shown earlier that the viewpoint locus is (if it constrained to be a continuous surface/curve) a complete two dimensional surface. Hence, the image of the viewpoint locus of an oblique camera is the entire image surface, thus no meaningful epipole exists.

Reconstruction methods:

- *Rotating cameras:* In 1992 Ishiguro et al. [52] constructed non-central panoramic images by rotating a vertical linear camera (also often called slit), about a vertical axis not going through the linear camera's optical center. Two such images were used to demonstrate 3D reconstruction on non-central panoramic cameras. Here, the image is a non-central image. In principle, 3D reconstruction is possible from a single image. Kang and Szeliski followed a similar idea to extract depth from omnidirectional images constructed by rotating a vertical slit [54]. In contrast to the earlier approach this approach was used to construct central panoramic images using a calibrated camera. Again these images were used for 3D reconstruction. Omnidirectional images, with known camera motion, obtained from a GPS or a robot, have also been used in 3D reconstruction [61]. In this work a camera is rotated about its focal point to obtain a hemispherical mosaic. About 47 images are used to construct a single mosaic. As mentioned earlier, these systems with rotating components require mechanical scanning and they are considered to be impractical for real-time applications.

- *Fisheye cameras:* Recently, Micusik et al. extended multi-view metric 3D reconstruction to central fish-eye cameras [63]. The fisheye lens used is a Nikon Coolpix E8 with a field of view larger than 180°. The camera model used in this work [62] was an extension of the work by Fitzgibbon [33] (already explained in section 3.3.1). Image correspondences were computed automatically for calibrating the lens and to eventually obtain 3D reconstruction.

- *Catadioptric cameras:* Let us consider the 3D reconstruction works using calibrated omnidirectional images. Planar and curved mirrors (elliptical, hyperbolic and parabolic) were used along with a single camera to produce computational stereo and later to obtain 3D reconstruction [16, 72]. A few catadioptric images from sensors, with approximately known calibration parameters, were used to obtain 3D reconstruction [23]. There are several works on 3D reconstruction using uncalibrated omnidirectional images. 3D reconstruction of piecewise planar objects was obtained from a single panoramic view [99]. This work also shows a simple calibration approach for para-catadioptric cameras using a single view with known mirror parameters. Geometric constraints such as coplanarity, perpendicularity and parallelism were provided by the user to finally obtain a 3D reconstruction. Central catadioptric cameras such as para-catadioptric systems (orthographic camera facing a parabolic mirror) were calibrated and utilized in 3D reconstruction by Geyer and Daniilidis [37].

3.4 Unifying efforts

- *Mixing catadioptric and perspective cameras:* By expressing the image coordinates in lifted coordinates, the back-projection of rays from images are obtained for perspective, affine and para-catadioptric cameras. The theory of fundamental matrix, trifocal and quadrifocal tensors is extended for any combination of para-catadioptric, perspective or affine cameras [100].

- *Generic epipolar geometry:* The concept of epipolar geometry is described in section 3.3.2. A non-parametric approach for computing the epipolar geometry was proposed in [119]. The input to this

algorithm is multiple image pairs acquired by stereo cameras with fixed configuration. By performing dense matching in each of these image pairs, a dense map of the epipolar curves can be determined on the images. The main advantage of this approach is that the algorithm is not based on any parametric model for computing the epipolar geometry. As a result this work can be extended to different omnidirectional cameras.

- *Generic imaging model:* A non-parametric model and approach to camera calibration, referred to as the general imaging model, was recently introduced by Grossberg and Nayar [41]. The contributions of their work are, first, the formulation of a generic imaging model to represent any arbitrary imaging system, second, a simple calibration method to compute the parameters of the unknown imaging model. Their model consists of a set of photo-sensitive elements on the image detector and their corresponding incoming light rays.

  - *Calibration in the presence of known motion:* The computation of a mapping between these photo-sensitive elements and their corresponding projection rays is nothing but the calibration. The photo-sensitive elements are referred to as *raxels*. In practice each raxel actually collects the light energy from a bundle of closely packed light rays. For studying the geometrical properties we associate each pixel to a single principal ray. In addition to geometrical properties these raxels also have their own radiometric (brightness and wavelength) response as well as optical (point spread) properties. The calibration method of [41] is able to compute both the geometric and radiometric properties. The geometric calibration is achieved using two parallel planes which are separated by a known distance. These planes have active displays which alternatively show horizontal and vertical binary coding patterns of varying resolutions. Using this approach we can compute the 3D projection ray corresponding to every pixel. Nevertheless, this calibration technique is restricted to laboratories: the whole process requires active displays and knowledge of the motion between them.
  - *Generic calibration in the presence of unknown motion:* We developed a simple calibration technique to calibrate a generic imaging model without the restriction of known motion. In our work we focus only on the geometric calibration. It only requires taking images of calibration objects, from completely unknown viewpoints [86, 105]. The thesis is based on several aspects of generic calibration and generic structure-from-motion algorithms [84–88]. Generic self-calibration is a very hard problem. Very few works have addressed this problem [76, 87]. These works assume that the camera model is central and the motions are restricted, either pure translation or pure rotation.

Chapter 4

Calibration of Non-Central Cameras

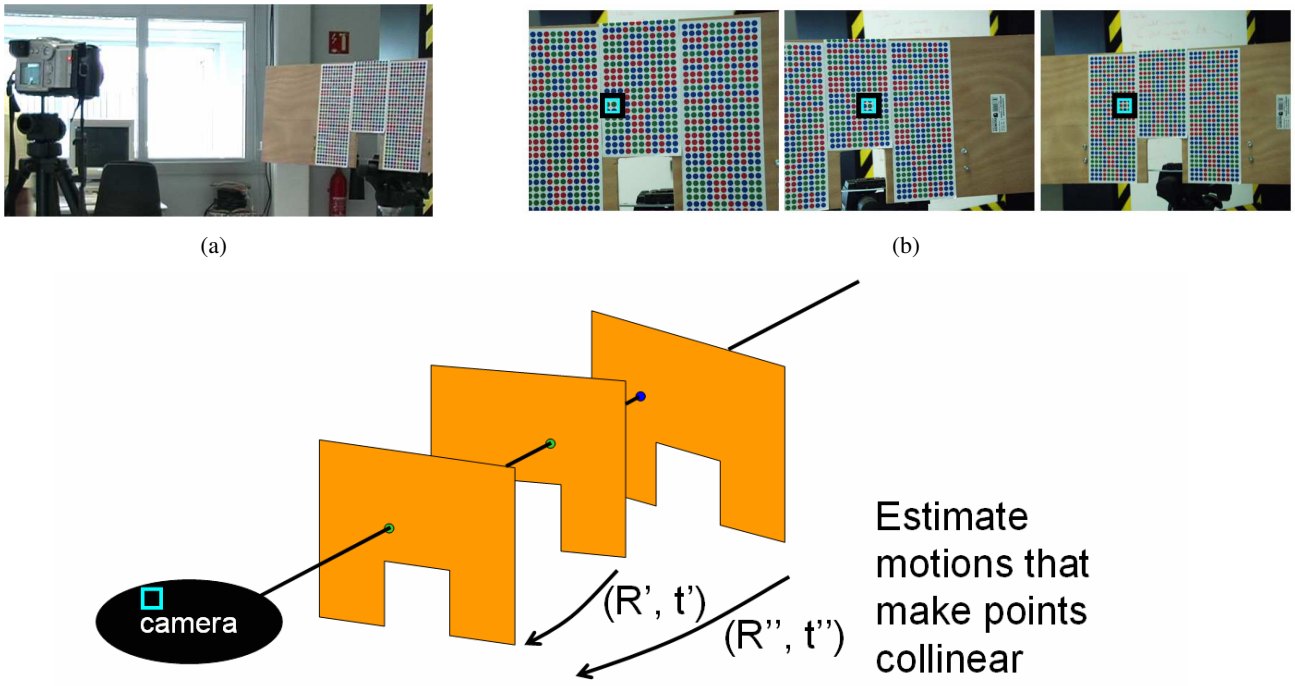


Figure 4.1: The concept of generic calibration. (a) The setup consists of a general camera observing a calibration grid. (b) Three images of a calibration grid observed from different viewpoints and orientations are shown. (c) A schematic diagram showing a single pixel and its 3D ray. This ray is eventually computed by estimating the poses of the individual grids.

This chapter will focus on the theoretical and practical aspects of the calibration algorithm for non-central cameras. Non-central cameras are the most general form of cameras where the projection rays can be completely arbitrary. It is very important to understand the purpose of camera calibration, i.e. the estimation of a camera's intrinsic parameters. In simple words this will exactly enable us to compute a 3D ray, along which light travels, for every image pixel. Before getting into the details of the algorithm, we briefly describe the main idea behind our generic calibration (see Figure 4.1). Three images of a calibration grid are captured by the general camera that needs to be calibrated. The images are taken from unknown viewpoints. During our calibration experiment every image pixel observes three 3D points in different calibration grids. These 3D points are obtained in three different coordinate systems. However, these points are collinear if they are

expressed in the same coordinate system. This will enable us to compute the motion between the views and eventually the projection rays for the image pixels.

For a better understanding of our algorithm we first introduce the underlying idea for 2D cameras in section 4.1.1 and 4.1.2. Then we focus on the more practical scenario involving 3D cameras in section 4.1.3. By 3D cameras we don't meant cameras providing three-dimensional images, but cameras evolving in 3D space. The algorithms are developed for two kinds of calibration grids: 3D and planar. In practice, planar calibration grids are more common and convenient than 3D calibration grids. In section 4.2 we discuss the practical issues concerning feature extraction and 2D to 3D matching. In section 4.3 we provide the details of the experiments that were conducted to test the theory. We present the results of calibration of a multi-camera system, which belongs to the class of generally non-central cameras.

4.1 Generic Calibration of Non-Central Cameras

4.1.1 2D cameras

We consider here a camera and scene living in a 2D plane, i.e. camera rays are lines in that plane.

Input. We acquire two images of an object undergoing some motion. Consider here a single pixel and its camera ray, as illustrated in figure 4.2(a). Figures 4.2 (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, first in some local coordinate frame attached to the object ("matching").

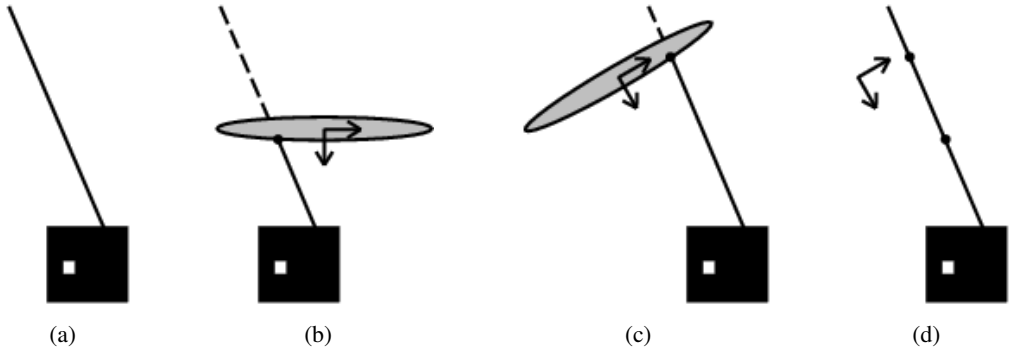


Figure 4.2: (a) The camera as black box, with one pixel and its camera ray. (b) The pixel sees a point on a calibration object, whose coordinates are identified in a frame associated with the object. (c) Same as (b), for another position. (d) Due to known motion, the two points on the calibration object can be placed in the same coordinate frame (have the same as in (c)). The camera ray is then determined by joining them.

The case of known motion. If the object's motion between the image acquisitions is known, then the two object points can be mapped to a single coordinate frame, e.g. the object's coordinate frame for the second image, as shown in figure 4.2 (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 [41], applied here for the 2D case. Camera rays are thus initially expressed in a coordinate frame attached to one of the calibration object's positions. This does not matter (all that counts are the relative positions of the camera rays), but for convenience, one would typically try to 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).

The case of unknown motion. The above approach is no longer applicable, and we need to estimate, implicitly or explicitly, the unknown motion. We now show how to do this, given three images. Let us note the three points on the calibration objects, that are seen in the same pixel, by $\mathbf{Q}, \mathbf{Q}'$ and $\mathbf{Q}''$. 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, that allow to map the second and third frames onto the first one, are given by 2×2 rotation matrices R' and R'' and translation vectors $\mathbf{t}'$ and $\mathbf{t}''$. Note that for the rotation matrices we have $R'_{11} = R'_{22}$ and $R'_{12} = -R'_{21}$ (and similarly for R''). The calibration points, after mapping them to the common frame, are given as:

$$\mathbf{Q} \qquad \begin{pmatrix} \mathsf{R}' & \mathbf{t}' \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \mathbf{Q}' \qquad \begin{pmatrix} \mathsf{R}'' & \mathbf{t}'' \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \mathbf{Q}''$$

They must all lie on the pixel's camera ray, hence they must be collinear. Algebraically, this is expressed by the fact that the determinant of the 3×3 matrix composed of the above three coordinate vectors, vanishes:

$$\begin{vmatrix} Q_1 & R'_{11}Q'_1 + R'_{12}Q'_2 + t'_1Q'_3 & R''_{11}Q''_1 + R''_{12}Q''_2 + t''_1Q''_3 \\ Q_2 & R'_{21}Q'_1 + R'_{22}Q'_2 + t'_2Q'_3 & R''_{21}Q''_1 + R''_{22}Q''_2 + t''_2Q''_3 \\ Q_3 & Q'_3 & Q''_3 \end{vmatrix} = 0 \tag{4.1}$$

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 4.1. In the following, we sometimes call this the "calibration tensor". It is somewhat related to the *homography tensor* derived in [94].

i	C_i	V_i
1	$Q_1Q'_1Q''_3 + Q_2Q'_2Q''_3$	R'_{21}
2	$Q_1Q'_2Q''_3 - Q_2Q'_1Q''_3$	R'_{22}
3	$Q_1Q'_3Q''_1 + Q_2Q'_3Q''_2$	$-R''_{21}$
4	$Q_1Q'_3Q''_2 - Q_2Q'_3Q''_1$	$-R''_{22}$
5	$Q_3Q'_1Q''_1 + Q_3Q'_2Q''_2$	$R'_{11}R''_{21} - R''_{11}R'_{21}$
6	$Q_3Q'_1Q''_2 - Q_3Q'_2Q''_1$	$R'_{11}R''_{22} - R''_{12}R'_{21}$
7	$Q_1Q'_3Q''_3$	$t'_2 - t''_2$
8	$Q_2Q'_3Q''_3$	$-t'_1 + t''_1$
9	$Q_3Q'_1Q''_3$	$R'_{11}t''_2 - R'_{21}t''_1$
10	$Q_3Q'_2Q''_3$	$R'_{12}t''_2 - R'_{22}t''_1$
11	$Q_3Q'_3Q''_1$	$R''_{21}t'_1 - R''_{11}t'_2$
12	$Q_3Q'_3Q''_2$	$R''_{22}t'_1 - R''_{12}t'_2$
13	$Q_3Q'_3Q''_3$	$t'_1t''_2 - t''_1t'_2$

Table 4.1: Coupled variables in the trifocal calibration tensor for the general 2D camera. Coefficients not shown here are always zero.

Among the $3 \cdot 3 \cdot 3 = 27$ coefficients of the calibration tensor, 8 are always zero and among the remaining 19 ones, there are 6 pairs of identical ones. The columns of table 4.1 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 (4.1):

$$\sum_{i=1}^{13} C_i V_i = 0 \tag{4.2}$$

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 (4.2). Note that we have verified using simulated data, that we indeed can obtain a unique solution (up to scale) for the tensor. The main questions now are, if the motion coefficients R', R'', $\mathbf{t}'$ and $\mathbf{t}''$ can be extracted from the tensor. This is indeed possible, as shown below. Once the motions are determined, the above approach can be readily applied to compute the camera rays.

We now describe an algorithm for extracting the motion parameters. Let the estimated tensor coefficients be V_i'; they are equal to the coefficients of table 4.1 up to an unknown scale: $V_i' = \lambda V_i, i = 1 \dots 13$. The proposed algorithm works as follows:

- Estimate λ : $\lambda = \sqrt{(V_1')^2 + (V_2')^2)}$ (exploiting orthonormality of 2×2 rotation matrix R'). λ is defined up to sign.
- Compute $V_i = \frac{V_i'}{\lambda}, i = 1 \dots 13$. We resolve the sign ambiguity in λ in the following manner.
- Compute R': $R'_{11} = R'_{22} = V_2$ and $R'_{21} = -R'_{12} = V_1$
- Compute R'': $R''_{11} = R''_{22} = -V_4$ and $R''_{21} = -R''_{12} = -V_3$
- Compute $\mathbf{t}$ and $\mathbf{t}'$:

  From table 4.1 we have:

$$\begin{pmatrix} 0 & 1 & 0 & -1 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & -V_1 & V_2 \\ 0 & 0 & -V_2 & -V_1 \\ -V_3 & V_4 & 0 & 0 \\ -V_4 & V_3 & 0 & 0 \end{pmatrix} \begin{pmatrix} t'_1 \\ t'_2 \\ t''_1 \\ t''_2 \end{pmatrix} = \begin{pmatrix} V_7 \\ V_8 \\ V_9 \\ V_{10} \\ V_{11} \\ V_{12} \end{pmatrix}$$

  Due to the previous steps of the algorithm this translates to:

$$\underbrace{\begin{pmatrix} 0 & 1 & 0 & -1 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & -R'_{21} & R'_{11} \\ 0 & 0 & -R'_{22} & R'_{12} \\ -R''_{21} & -R''_{11} & 0 & 0 \\ R''_{22} & -R''_{12} & 0 & 0 \end{pmatrix}}_{\mathsf{M}} \begin{pmatrix} t'_1 \\ t'_2 \\ t''_1 \\ t''_2 \end{pmatrix} = \begin{pmatrix} V_7 \\ V_8 \\ V_9 \\ V_{10} \\ V_{11} \\ V_{12} \end{pmatrix}$$

  We can solve for $\mathbf{t}$ and $\mathbf{t}'$ by solving this linear equation system to least squares:

$$\begin{pmatrix} t'_1 \\ t'_2 \\ t''_1 \\ t''_2 \end{pmatrix} = (M^T M)^{-1} M^T \begin{pmatrix} V_7 \\ V_8 \\ V_9 \\ V_{10} \\ V_{11} \\ V_{12} \end{pmatrix}$$

  The solution is always well-defined: $\mathsf{M}^T\mathsf{M}$ is the following non-singular matrix:

$$M^T M = \begin{pmatrix} 2 & 0 & -1 & 0 \\ 0 & 2 & 0 & -1 \\ -1 & 0 & 2 & 0 \\ 0 & -1 & 0 & 2 \end{pmatrix}$$

4.1.2 2D cameras using linear calibration grids

i	C_i	V_i
1	$Q_1Q'_1Q''_3$	R'_{21}
2	$Q_1Q'_3Q''_1$	$-R''_{21}$
3	$Q_1Q'_3Q''_3$	$t'_2 - t''_2$
4	$Q_3Q'_1Q''_1$	$R'_{22}R''_{21} - R'_{21}R''_{22}$
5	$Q_3Q'_1Q''_3$	$R'_{22}t''_2 - R'_{21}t''_1$
6	$Q_3Q'_3Q''_1$	$R''_{21}t'_1 - R''_{22}t'_2$
7	$Q_3Q'_3Q''_3$	$t'_1t''_2 - t''_1t'_2$

Table 4.2: Coefficients of the trifocal calibration tensor for the general 2D camera and a linear calibration object.

It is equally worthwhile to specialize our concept to the case of a linear calibration object. We now consider again the general, non-central camera model. Without loss of generality, we suppose that the calibration points lie on a line with $Y = 0$ in the local coordinate frame, i.e. $Q_2 = Q'_2 = Q''_2 = 0$. The collinearity equation(4.1) gives then rise to a $2 \times 2 \times 2$ trifocal calibration tensor. Among its 8 coefficients, only 1 is always zero, and among the others, none are identical to one another, cf. table 4.2.

We observe that the rotation coefficients R'_{22} and R''_{22} do not appear individually, contrary to the tensor for the general case. Hence, the scale factor λ can no longer be determined as easily as in the above algorithm. The motion parameters can nevertheless be extracted, but the algorithm is more complicated.

4.1.3 3D cameras

Figure 4.1 shows the general idea behind the generic calibration. We extend the concept described in section 4.1.1 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 section 4.1.1, cf. [41]. 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 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 following points must be collinear (they must lie on the associated pixel's projection ray):

$$\mathbf{Q} \qquad \begin{pmatrix} \mathsf{R}' & \mathbf{t}' \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \mathbf{Q}' \qquad \begin{pmatrix} \mathsf{R}'' & \mathbf{t}'' \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \mathbf{Q}''$$

We may stack them in the following 4×3 matrix:

$$\begin{pmatrix} Q_1 & R'_{11}Q'_1 + R'_{12}Q'_2 + R'_{13}Q'_3 + t'_1Q'_4 & R''_{11}Q''_1 + R''_{12}Q''_2 + R''_{13}Q''_3 + t''_1Q''_4 \\ Q_2 & R'_{21}Q'_1 + R'_{22}Q'_2 + R'_{23}Q'_3 + t'_2Q'_4 & R''_{21}Q''_1 + R''_{22}Q''_2 + R''_{23}Q''_3 + t''_2Q''_4 \\ Q_3 & R'_{31}Q'_1 + R'_{32}Q'_2 + R'_{33}Q'_3 + t'_3Q'_4 & R''_{31}Q''_1 + R''_{32}Q''_2 + R''_{33}Q''_3 + t''_3Q''_4 \\ Q_4 & Q'_4 & Q''_4 \end{pmatrix} \tag{4.3}$$

The collinearity constraint means that this matrix must be of rank less than 3, which implies that all subdeterminants of size 3×3 vanish. There are 4 of them, obtained by leaving out one row at a time from the

original matrix. 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 4.1.3 shows the coefficients of the first two calibration tensors. The equations are $\sum_i V_i C_i = 0$ and $\sum_i W_i C_i = 0$. In both, 34 of the 64 coefficients are always zero. One may observe that the two tensors share several coefficients, e.g. $V_8 = W_1 = R'_{31}$. The situation is similar for the other two tensors, which are not shown here since the first two are sufficient to compute the motion parameters and thus to perform calibration.

The tensors can be estimated by solving linear equation systems, 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 (up to scale) tensors. The algorithm proceeds as follows.

1. Estimate scale factors based on the orthonormality of R':

$$\lambda = \sqrt{V'^2_8 + V'^2_9 + V'^2_{10}}$$

$$\mu = \sqrt{W'^2_1 + W'^2_2 + W'^2_3}$$

   This defines λ and μ up to sign.

2. Compute $V_i = \frac{V'_i}{\lambda}$ and $W_i = \frac{W'_i}{\mu}, i = 1 \ldots 37$. If $(V_8 W_1 + V_9 W_2 + V_{10} W_3) < 0$ then set $\mu = -\mu$ and rescale W.

3. Compute R' and R'':

$$\mathsf{R}' = \begin{pmatrix} -W_{15} & -W_{16} & -W_{17} \\ -V_{15} & -V_{16} & -V_{17} \\ V_8 & V_9 & V_{10} \end{pmatrix} \qquad \mathsf{R}'' = \begin{pmatrix} W_{18} & W_{19} & W_{20} \\ V_{18} & V_{19} & V_{20} \\ -V_{11} & -V_{12} & -V_{13} \end{pmatrix}$$

   If $det(\mathsf{R}') < 0$ then scale V and W by -1 apdate R and R' accordingly. In the presence of noise these matrices will in general not be orthonormal. We "correct" this by computing the orthonormal matrices that are closest to the original matrices (in the sense of the Frobenius norm). To do so, let $\mathsf{U}\sigma^{\mathsf{T}}V$ be the SVD of one of the original matrices. The required orthonormal matrix is then given by $\mathsf{U}\mathsf{V}^{\mathsf{T}}$.

4. Compute $\mathbf{t}'$ and $\mathbf{t}''$ by solving the following linear system using least squares:

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}''
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 4.3: Coupled variables in the trifocal calibration tensors for a general 3D camera. Coefficients not shown here are always zero.

$$\underbrace{\begin{pmatrix} 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & -1 & 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -R'_{31} & R'_{21} \\ 0 & 0 & 0 & -R'_{31} & 0 & R'_{11} \\ 0 & 0 & 0 & 0 & -R'_{32} & R'_{22} \\ 0 & 0 & 0 & -R'_{32} & 0 & R'_{12} \\ 0 & 0 & 0 & 0 & -R'_{33} & R'_{23} \\ 0 & 0 & 0 & -R'_{33} & 0 & R'_{13} \\ 0 & R''_{31} & -R''_{21} & 0 & 0 & 0 \\ R''_{31} & 0 & -R''_{11} & 0 & 0 & 0 \\ 0 & R''_{32} & -R''_{22} & 0 & 0 & 0 \\ R''_{32} & 0 & -R''_{12} & 0 & 0 & 0 \\ 0 & R''_{33} & -R''_{23} & 0 & 0 & 0 \\ R''_{33} & 0 & -R''_{13} & 0 & 0 & 0 \end{pmatrix}}_{\mathsf{A}} \begin{pmatrix} t'_1 \\ t'_2 \\ t'_3 \\ t''_1 \\ t''_2 \\ t''_3 \end{pmatrix} = \underbrace{\begin{pmatrix} V_{14} \\ V_{21} \\ W_{21} \\ V_{25} \\ W_{25} \\ V_{29} \\ W_{29} \\ V_{33} \\ W_{33} \\ V_{34} \\ W_{34} \\ V_{35} \\ W_{35} \\ V_{36} \\ W_{36} \end{pmatrix}}_{\mathbf{b}}$$

The least squares solution $\left(\mathsf{A}^\mathsf{T}\mathsf{A}\right)^{-1}\mathsf{A}^\mathsf{T}\mathbf{b}$ is well defined since it can be shown that, due to the orthonormality of the rotation matrices R' and R'', the product $\mathsf{A}^\mathsf{T}\mathsf{A}$ is the following invertible matrix:

$$\mathsf{A}^\mathsf{T}\mathsf{A} = \begin{pmatrix} 2 & 0 & 0 & -1 & 0 & 0 \\ 0 & 2 & 0 & 0 & -1 & 0 \\ 0 & 0 & 3 & 0 & 0 & -1 \\ -1 & 0 & 0 & 2 & 0 & 2 \\ 0 & -1 & 0 & 0 & 2 & 0 \\ 0 & 0 & -1 & 0 & 0 & 3 \end{pmatrix}$$

This implies that a unique solution for motion can always be estimated and that it is correct if the calibration tensors are correct.

4.1.4 3D cameras with a planar calibration object

In the general calibration algorithm, if we feed the data from a planar grid, the linear systems for estimating the tensors are more rank-deficient than they ought to be, i.e. the tensors can not be estimated uniquely (same when having data from a central, chapter 5 or axial camera, chapter 6). In order to remove this extra rank deficiency, we make the following change to incorporate the planarity constraint.

$$Q_3 = Q'_3 = Q''_3 = 0$$

We again form the 4×3 matrix with the triplet in one common coordinate frame as follows.

$$\begin{pmatrix} Q_1 & R'_{11}Q'_1 + R'_{12}Q'_2 + t'_1Q'_4 & R''_{11}Q''_1 + R''_{12}Q''_2 + t''_1Q''_4 \\ Q_2 & R'_{21}Q'_1 + R'_{22}Q'_2 + t'_2Q'_4 & R''_{21}Q''_1 + R''_{22}Q''_2 + t''_2Q''_4 \\ 0 & R'_{31}Q'_1 + R'_{32}Q'_2 + t'_3Q'_4 & R''_{31}Q''_1 + R''_{32}Q''_2 + t''_3Q''_4 \\ Q_3 & Q'_4 & Q''_4 \end{pmatrix}$$

We obtain four tensors by removing one row at a time and expanding the matrix. Using simulations, we found that the removal of the third row does not provide a unique solution. The reason might be that, while

i	C_i	V_i	W_i
1	$Q_1Q'_1Q''_4$	0	$R'_{3,1}$
2	$Q_1Q'_2Q''_4$	0	$R'_{3,2}$
3	$Q_1Q'_4Q''_1$	0	$-R''_{3,1}$
4	$Q_1Q'_4Q''_2$	0	$-R''_{3,2}$
5	$Q_1Q'_4Q''_4$	0	$t'_3 - t''_3$
6	$Q_2Q'_1Q''_4$	$R'_{3,1}$	0
7	$Q_2Q'_2Q''_4$	$R'_{3,2}$	0
8	$Q_2Q'_4Q''_1$	$-R''_{3,1}$	0
9	$Q_2Q'_4Q''_2$	$-R''_{3,2}$	0
10	$Q_2Q'_4Q''_4$	$t'_3 - t''_3$	0
11	$Q_4Q'_1Q''_1$	$-R''_{2,1}R'_{3,1} + R'_{2,1}R''_{3,1}$	$R'_{1,1}R''_{3,1} - R''_{1,1}R'_{3,1}$
12	$Q_4Q'_1Q''_2$	$R'_{2,1}R''_{3,2} - R''_{2,2}R'_{3,1}$	$R'_{1,1}R''_{3,2} - R''_{1,2}R'_{3,1}$
13	$Q_4Q'_1Q''_4$	$R'_{2,1}t''_3 - t''_2R'_{3,1}$	$R'_{1,1}t''_3 - t''_1R'_{3,1}$
14	$Q_4Q'_2Q''_1$	$-R''_{2,1}R'_{3,2} - R''_{2,2}R'_{3,2}$	$R'_{1,2}R''_{3,1} - R''_{1,1}R'_{3,2}$
15	$Q_4Q'_2Q''_2$	$R'_{2,2}R''_{3,2} - R''_{2,2}R'_{3,2}$	$R'_{1,2}R''_{3,2} - R''_{1,2}R'_{3,2}$
16	$Q_4Q'_2Q''_4$	$R'_{2,2}t''_3 - t''_2R'_{3,2}$	$R'_{1,2}t''_3 - t''_1R'_{3,2}$
17	$Q_4Q'_4Q''_1$	$-R''_{2,1}t'_3 + t'_2R''_{3,1}$	$t'_1R''_{3,1} - R''_{1,1}t'_{3,1}$
18	$Q_4Q'_4Q''_2$	$t'_2R''_{3,2} - R''_{2,2}t'_3$	$t'_1R''_{3,2} - R''_{1,2}t'_{3,1}$
19	$Q_4Q'_4Q''_4$	$t'_2t''_3 - t''_2t'_3$	$t'_1t''_3 - t''_1t'_3$

Table 4.4: Coupled variables in the 3D general camera with a planar grid pattern

removing the third row, we do not actually utilize the planarity constraint. However we use only the first two tensor equations for computing the motion parameters.

The extraction of individual motion parameters is slightly more complicated compared to the earlier cases. First of all we do not have any direct constraint to compute the scale parameters. However we may compute V and W up to a common scale λ, since they share some common variables. Let $V' = \lambda V$ and $W' = \lambda W$ be the estimated coefficients. We then have the following relations:

$$\lambda R'_{3,1} = V'_6$$

$$\lambda R'_{3,2} = V'_7$$

$$\lambda R''_{3,1} = -V'_8$$

$$\lambda R''_{3,2} = -V'_9$$

Let us compute two scalars u' and u'' as follows:

$$u' = \lambda t'_3 = \frac{(-V'_9V'_{17} + V'_8V'_{18})V'_6}{-V'_9V'_{11} + V'_8V'_{12}}$$

$$u'' = \lambda t''_3 = (\lambda t'_3) - V'_{10}$$

For extracting the remaining parameters we follow an indirect approach. From table 4.4 we get:

$$\begin{pmatrix} -V'_8 & 0 & -V'_6 & 0 \\ -V'_9 & 0 & 0 & -V'_6 \\ 0 & -V'_8 & -V'_7 & 0 \\ 0 & -V'_9 & 0 & -V'_7 \end{pmatrix} \begin{pmatrix} R'_{2,1} \\ R'_{2,2} \\ R''_{2,1} \\ R''_{2,2} \end{pmatrix} = \begin{pmatrix} V'_{11} \\ V'_{12} \\ V'_{14} \\ V'_{15} \end{pmatrix}$$

The rank of the above system is 3 and hence we obtain the solution in the subspace spanned by two vectors:

$$\begin{pmatrix} R'_{2,1} \\ R'_{2,2} \\ R''_{2,1} \\ R''_{2,2} \end{pmatrix} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{pmatrix} + l_1 \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{pmatrix}$$

Similarly for extracting $R'_{1,1}$,$R'_{1,2}$,$R''_{1,1}$ and $R''_{1,2}$, we form the following linear system.

$$\begin{pmatrix} -V'_8 & 0 & -V'_6 & 0 \\ -V'_9 & 0 & 0 & -V'_6 \\ 0 & -V'_8 & -V'_7 & 0 \\ 0 & -V'_9 & 0 & -V'_7 \end{pmatrix} \begin{pmatrix} R'_{1,1} \\ R'_{1,2} \\ R''_{1,1} \\ R''_{1,2} \end{pmatrix} = \begin{pmatrix} W'_{11} \\ W'_{12} \\ W'_{14} \\ W'_{15} \end{pmatrix}$$

As in the previous case, the rank of the above system is 3 and the solution is defined up to an unknown l_2.

$$\begin{pmatrix} R'_{1,1} \\ R'_{1,2} \\ R''_{1,1} \\ R''_{1,2} \end{pmatrix} = \begin{pmatrix} a_5 \\ a_6 \\ a_7 \\ a_8 \end{pmatrix} + l_2 \begin{pmatrix} b_5 \\ b_6 \\ b_7 \\ b_8 \end{pmatrix}$$

We estimate the values of l_1 and l_2 using orthonormal properties of the rotation matrices R' and R''.

$$\begin{aligned} R'_{1,1}R'_{1,2} + R'_{2,1}R'_{2,2} + R'_{3,1}R'_{3,2} &= 0 \\ R''_{1,1}R''_{1,2} + R''_{2,1}R''_{2,2} + R''_{3,1}R''_{3,2} &= 0 \\ R'^2_{1,1} + R'^2_{2,1} + R'^2_{3,1} &= 1 \\ R'^2_{1,2} + R'^2_{2,2} + R'^2_{3,2} &= 1 \\ R''^2_{1,1} + R''^2_{2,1} + R''^2_{3,1} &= 1 \\ R''^2_{1,2} + R''^2_{2,2} + R''^2_{3,2} &= 1 \end{aligned}$$

On substituting the rotation variables and simplifying the expressions we get the following system.

$$\begin{pmatrix} a_1b_2 + b_1a_2 & b_1b_2 & a_5b_6 + b_5a_6 & b_5b_6 & V'_6V'_7 \\ a_3b_4 + b_3a_4 & b_3b_4 & a_7b_8 + b_8a_7 & b_7b_8 & V'_8V'_9 \\ 2a_1b_1 & b_1^2 & 2a_5b_5 & b_5^2 & V'^2_6 \\ 2a_2b_2 & b_2^2 & 2a_6b_6 & b_6^2 & V'^2_7 \\ 2a_3b_3 & b_3^2 & 2a_7b_7 & b_7^2 & V'^2_8 \\ 2a_4b_4 & b_4^2 & 2a_8b_8 & b_8^2 & V'^2_9 \end{pmatrix} \begin{pmatrix} l_1 \\ l_1^2 \\ l_2 \\ l_2^2 \\ \frac{1}{\lambda^2} \end{pmatrix} = \begin{pmatrix} -a_5a_6 - a_1a_2 \\ -a_7a_8 - a_3a_4 \\ 1 - a_1^2 - a_5^2 \\ 1 - a_2^2 - a_6^2 \\ 1 - a_3^2 - a_7^2 \\ 1 - a_4^2 - a_8^2 \end{pmatrix}$$

The rank of the above system is 3 and we obtain solutions for l_1,l_2, l_1^2,l_2^2 and $\frac{1}{\lambda^2}$ in a subspace spanned by three vectors.

$$\begin{pmatrix} l_1 \\ l_1^2 \\ l_2 \\ l_2^2 \\ \frac{1}{\lambda^2} \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \end{pmatrix} + m_1 \begin{pmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \end{pmatrix} + m_2 \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \end{pmatrix}$$

However, l_1 and l_2 are obtained uniquely. This behaviour is tested using simulations. In other words, e_1, e_3, f_1 and f_3 are all zeros. Thus we have the following.

$$l_1 = d_1$$

$$l_2 = d_3$$

Finally we use the values of l_1 and l_2 to compute $R'_{1,1}$, $R'_{1,2}$, $R'_{2,1}$,$R'_{2,2}$,$R''_{1,1}$, $R''_{1,2}$,$R''_{2,1}$ and $R''_{2,2}$, t' and t'':

$$\lambda = \pm\sqrt{\frac{-R'_{3,1}R'_{3,2}}{R'_{1,1}R'_{1,2} + R'_{2,1}R'_{2,2}}}$$

$$R'_{3,1} = \frac{V'_6}{\lambda} \qquad R'_{3,2} = \frac{V'_7}{\lambda}$$

$$R''_{3,1} = \frac{-V'_8}{\lambda} \qquad R''_{3,2} = \frac{-V'_9}{\lambda}$$

$$t'_1 = \frac{W'_{17} + R''_{1,1}u'}{-V'_8} \qquad t'_2 = \frac{V'_{17} + R''_{3,1}u'}{-V'_8} \qquad t'_3 = \frac{u'}{\lambda}$$

$$t''_1 = \frac{R'_{1,1}u'' - W'_{13}}{V'_6} \qquad t''_2 = \frac{R'_{2,1}u'' - V'_{13}}{V'_6} \qquad t''_3 = \frac{u''}{\lambda}$$

4.1.5 Summary of the calibration algorithms

- Take three images of the calibration grid from different viewpoints.
- For pixels, match calibration points.
- Estimate tensors (see Tables 4.1, 4.2 4.2 and 4.4).
- Extract motion.
- Compute the projection rays.

4.2 2D to 3D matching

The physical location of the actual photosensitive elements that correspond to the pixels, does in principle not matter at all in the generic camera model. On the one hand, this means that the camera ray corresponding to some pixel, need not pass through that pixel. On the other hand, neighborhood relations between pixels are in theory 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: we suppose that pixels are indexed by two integers (the pixel's *coordinates*) like in traditional cameras and that pixels with neighboring coordinates have associated camera rays that are "close" to one another.

In our experiments we used planar calibration objects consisting of black dots or squares on a white paper. Figure 4.3 shows one of the grids used with a pinhole camera. We use Harris corner detection [45] to extract the dots and later process the nearby pixels to compute the center of the dots up to a subpixel level accuracy.

In each image, we thus get matches for the pixels lying in the center of calibration dots, i.e. we can determine the 2D plane coordinates of the corresponding calibration point. To determine the matching

calibration point for *any* pixel, we proceed as follows. We determine the four closest pixels which correspond to centers of calibration dots and such that no three of them are collinear. We then compute the unique 2D projective transformation (homography) that maps these pixels to the matched calibration points. This homography is then applied on the coordinates of the considered pixel, to find its matching calibration point on the planar grid. This approach comes down to making the assumption that very locally (in the neighborhood covered by quadruplets of calibration dots), the projection is perspective. We found this assumption to be reasonable. It may be relaxed if necessary by using a structured light type approach, e.g. as in [112] by projecting series of binary patterns on a flat screen, which acts effectively as calibration grid.

In Figure 4.3, for every image pixel corresponding to every dot in the first grid, we compute the 2D positions in the second and the third grids using the 4-point homography for the neighboring pixels. It is very important to note that the four neighboring dots which are selected should not have any subset of 3 collinear dots.

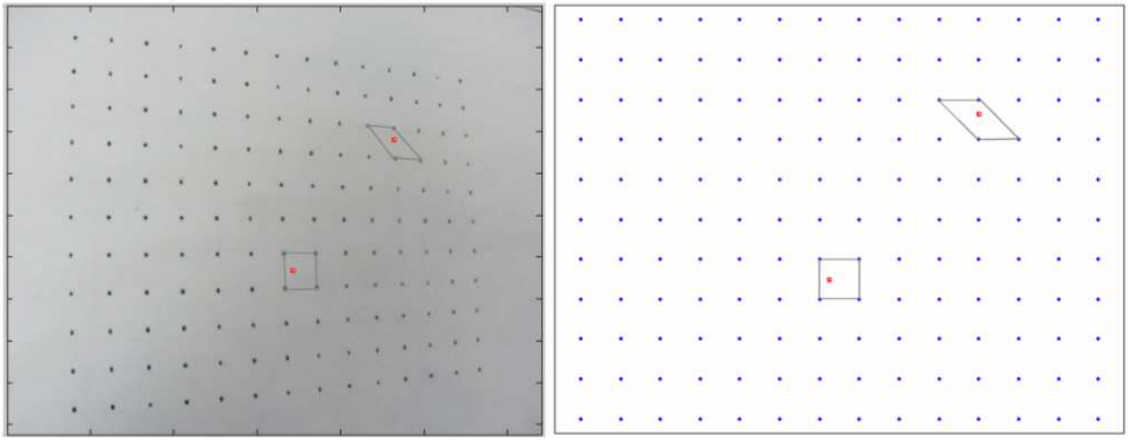


Figure 4.3: On the left we show the image of the calibration object in the second view. On the right we show the 2D coordinates of the dots in the calibration plane. We see two pixels on the left, which do not lie on a dot, and thus for which the matching calibration points are not directly available. We use the 4-point homography associated with the nearby dots to obtain the 3D locations for the matching calibration points.

4.3 Experimental Evaluation

4.3.1 Practical Issues

The first important issue is the design of the calibration grid. We found that grids with circular targets provide stable calibration compared to checkerboard patterns. In the case of point targets, we can compute its projected image (small ellipse) using a simple thresholding method. The center of the projected image can be computed by computing its centroid. When the images are not focused (which often happens with omnidirectional cameras), the corners of the checkerboard pattern are difficult to estimate. This was found to be a problem e.g. for the calibration of a multi-camera setup, whereas circular targets led to a stable calibration.

A second issue is the orientation of calibration grids. 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).

We briefly give the reasons for this observation: the smaller the distance between corresponding points in different grids (i.e. points seen by the same pixel), the less stable the calibration. First, even if grid pose can be estimated stably using other points, the projection ray computed from points which are very close together, is bound to be inaccurate. Second, the whole calibration process is based on collinearity constraints. It is intuitive that the further apart corresponding points, the stabler the calibration (not only of the individual projection ray associated with these points, but of the overall calibration). One can give a few examples. For example, consider a grid which is translated in the same plane between different image acquisitions. This implies that corresponding points are actually identical in 3D, giving no support to the collinearity constraints underlying the calibration. Another example: consider a grid rotating several times about a line contained in the grid's plane. Even if the grid poses can be estimated accurately, the subsequent computation of projection rays will give random results for pixels lying on the image of the rotation axis. Further, for other pixels the projection rays will be the more inaccurate the closer the pixels are to the image of the rotation axis.

From this, one can conclude that a good strategy might be to put a grid at different distances from the camera; in practice we do this, using a different grid for each distance, the further away the larger, in order to always cover sufficiently large image regions. However, it is not advisable to put the grids such that they are all parallel to one another: in that case, the calibration tensors are not estimated uniquely and calibration would fail. This is not surprising, since even in the special case of perspective cameras, parallel calibration grids constitute a degenerate case [103, 127]. Hence, successful calibration requires putting calibration grids at different distances and with different orientations.

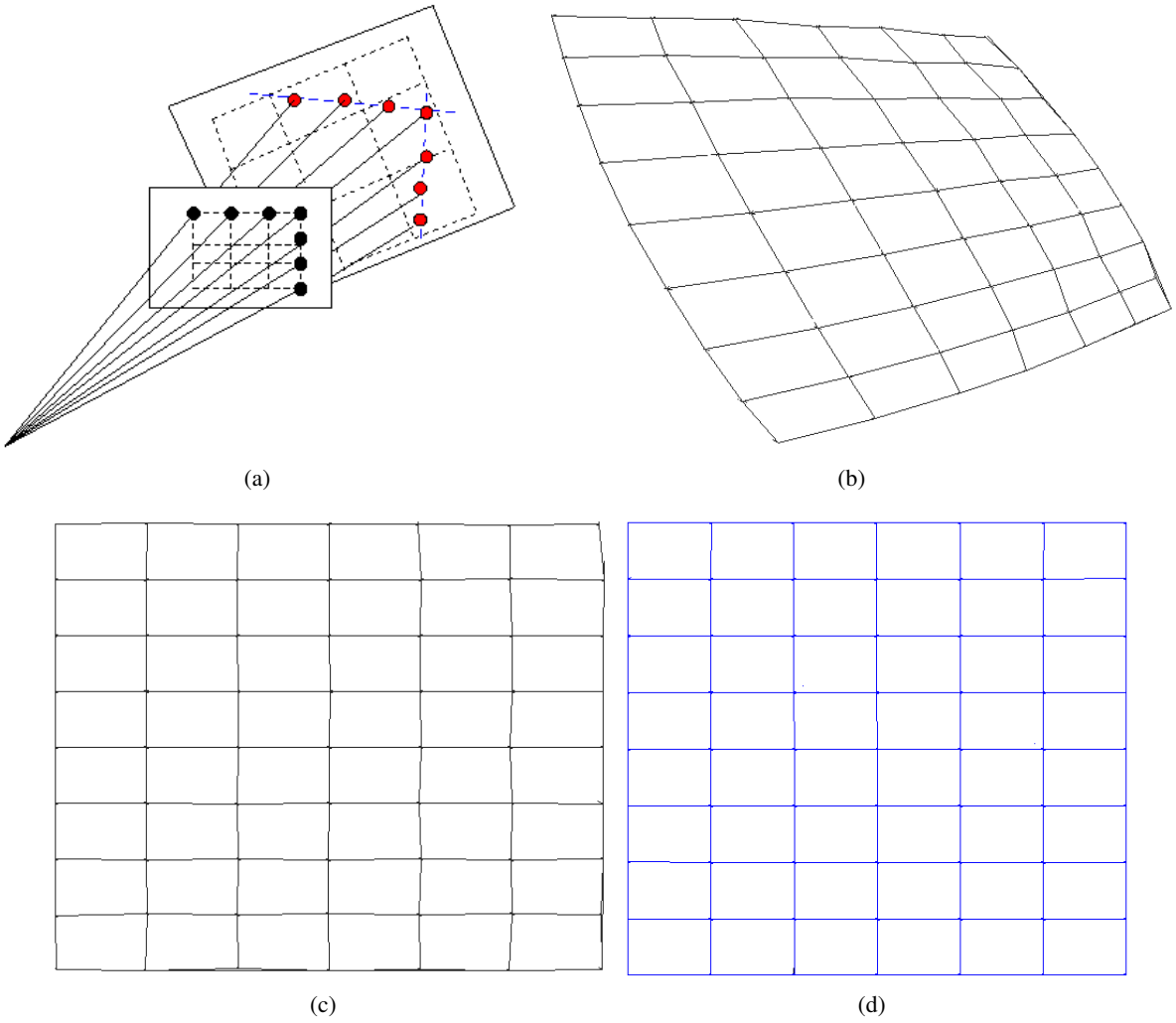
The third practical issue is related to the automation of dense image-to-grid matching where we match the 2D image features to 3D grid coordinates. By using a combination of *local* 4-point homography based prediction, local collinearity and orthogonality constraints, we start from four features (circular targets or corners), located at the corners of a square, and incrementally extend the matching along all directions. This automatic approach worked successfully for all pinhole and fisheye images as well as several images obtained using hyperbolic and spherical catadioptric systems. However we also had to use manual input and correction 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). Noise in the feature extraction, the nature of the camera model, etc. usually introduce errors in the interpolation. Thus we use a global collinearity constraint in the case of central cameras and a local collinearity constraint in the case of non-central ones during the process of interpolation. We observed a significant improvement in the stability of the tensors. This is illustrated by the example in Figure 4.4, which shows the perspective view synthesis (or, distortion correction) for the image of a grid that was not used in the calibration process.

4.3.2 Calibration of a multi-camera system

A multi-camera system can be considered as a single generic imaging system. As shown in Figure 4.5 (a), 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 4.5 (b)), using the algorithm described in § 4.1.4.

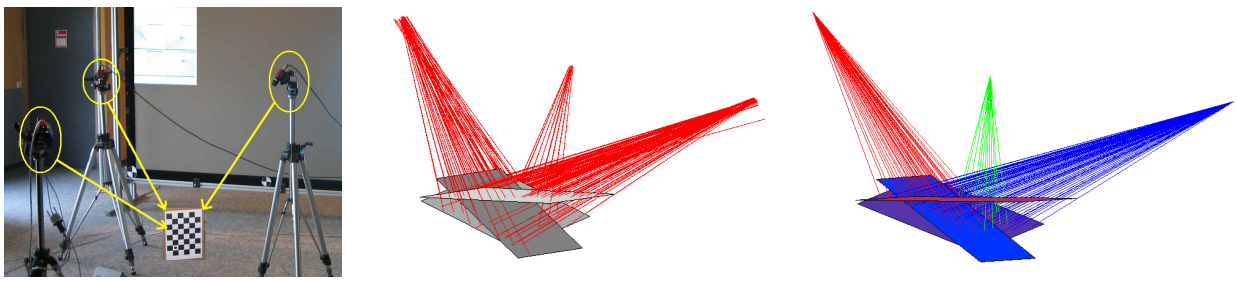
In order to evaluate the calibration, we compared results with those obtained by plane-based calibration [103, 127], that used the knowledge that the three cameras are approximately pinholes. In both, our multi-camera calibration, and plane-based calibration, the first grid was used to fix the global 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



(a) (b)

(c) (d)

Figure 4.4: Application of collinearity constraints (see text). a) interpolated grid points of a calibration grid, b) grid in a fisheye image, c) perspectively synthesized grid using calibration, obtained *without* the collinearity constraint, d) perspectively synthesized grid, using calibration, obtained *with* the collinearity constraint.



(a) (b) (c)

Figure 4.5: (a) Multi-camera setup consisting of 3 cameras. (b) Calibrated projection rays. (c) Three central clusters of rays.

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 4.5: Evaluation of non-central multi-camera calibration relative to plane-based calibration. See text for more details.

relative rotation between the rotation matrices given by the two methods, see columns R_i in Table 4.5). As for translation, we measure the distance between the estimated 3D positions of the grids' centers of gravity (columns t_i in Table 4.5) expressed in percent, relative to the overall size of the scene (distance between two farthest points in the scene). Here, plane-based calibration is done separately for each camera, leading to the three rows of Table 4.5.

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 4.5 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.

The non-central calibration algorithm works only for non-central models. Thus, we test it for a multi-camera system, which is clearly non-central. We tested the non-central approach on other cameras too, less non-central (fisheye, catadioptrics) or even central. In that case the non-central algorithm has a very high rank deficiency in the linear system and it degenerates. Appendix A.1 lists different calibration algorithms and the nature of solutions (unique, degenerate, inconsistent, etc.), that will be obtained, by applying various algorithms for different camera models. Besides the ray distribution in the actual camera, stability further depends on: number of images used, error/noise in data. Related to the issue of stability is overfitting: if too much noise, too few data, or data following a "smaller" model, then the fitting of the more complex model gives a meaningless result. We will discuss these issues in detail in section 7.5.

Finally, the provided evaluation of the non-central method is not extremely exhaustive, but validates and illustrates the concept. One would need many more experiments to fully validate the approach, but we concentrated on the central approach (cf. chapter 5) which we think is much more relevant in practice.

4.4 Conclusions

This chapter discussed the theory and calibration algorithms for generally non-central camera models. The calibration algorithm was first introduced for 2D scenarios and then extended to 3D scenarios. An important and practically useful variant of the calibration algorithm that uses planar calibration grids was presented. Finally we validated our theory by calibrating a multi-camera system, which is an example of a generally non-central camera, using our algorithm. The central and axial variant of the calibration algorithm will be introduced in chapters 5 and 6 respectively. We consider the problem of calibration of non-central cameras with more than three images in chapter 7.

Chapter 5

Calibration of Central Cameras

In the previous chapter we studied the theory of generic calibration for non-central cameras. In this chapter we examine a more restricted model where all the projection rays pass through a single point in space, in other words, the *central* scenario. Examples of this model include conventional perspective cameras, fisheye cameras and the central catadioptric cameras. The primary reason for having a seperate chapter on central cameras is because the theory and algorithms developed for non-central scenarios can not be applied directly to central cameras. Similar to the chapter on non-central cameras we start our study with a 2D scenario for simplicity. Like in the previous chapter we develop several variants of the calibration algorithms for different scenarios.

In section 5.1 and 5.2 we study the theory of generic calibration of 2D and 3D cameras. A practically useful variant of this algorithm using planar calibration grids for 3D cameras is also presented. We then present results of real experiments testing our theory, in section 5.4. Next we test our theory using some real experiments in section 5.4. That section considers three interesting applications: distortion correction, motion estimation and structure recovery. A more general framework for structure-from-motion analysis is presented in chapter 8.

5.1 Generic Calibration of a Central 2D Camera

As described in the previous chapter we formulate collinearity constraints to compute the projection rays for image pixels. In this chapter we study *central cameras* , i.e. all projection rays pass through a single point in space called the *optical center* $\mathbf{O}$ (O_1, O_2). Since the point O lies on all the rays we only need two calibration points to formulate collinearity constraints. In other words two calibration grids are sufficient to calibrate a central 2D camera. We use the constraint that with the correct estimates of the grid poses and the optical center, the latter is collinear with the two calibration points, for any pixel:

$$\begin{vmatrix} O_1 & Q_1 & R'_{11}Q'_1 + R'_{12}Q'_2 + t'_1Q'_3 \\ O_2 & Q_2 & R'_{21}Q'_1 + R'_{22}Q'_2 + t'_2Q'_3 \\ 1 & Q_3 & Q'_3 \end{vmatrix} = 0$$

This can be written as $\sum_{i=1} C_i V_i = 0$, where C_i and V_i are given in Table 5.1. There are only 7 coupled variables, which we compute up to a scale λ using the least squares method. After the computation of the scale factor, the remaining variables are extracted from the coupled variables. We give the algorithm below.

1. $\lambda = \pm\sqrt{(V'_1)^2 + (V'_2)^2}; V_i = \frac{V'_i}{\lambda}$. Do steps 2 and 3 for both signs of λ.

i	C_i	V_i
1	$Q_1Q'_1 + Q_2Q'_2$	R'_{21}
2	$Q_1Q'_2 - Q_2Q'_1$	R'_{22}
3	$Q_1Q'_3$	$t'_2 - O_2$
4	$Q_2Q'_3$	$O_1 - t'_1$
5	$Q_3Q'_1$	$-O_1R'_{21} + O_2R'_{11}$
6	$Q_3Q'_2$	$-O_1R'_{22} + O_2R'_{12}$
7	$Q_3Q'_3$	$-O_1t'_2 + O_2t'_1$

Table 5.1: Coupled variables in the bifocal matching tensor for a central 2D camera.

2. From table 5.1 we get:
$$\mathsf{R}' = \begin{pmatrix} V_2 & -V_1 \\ V_1 & V_2 \end{pmatrix}$$

3. From table 5.1 we get:
$$\begin{pmatrix} 0 & 1 & 0 & -1 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & -V_1 & V_2 \\ 0 & 0 & -V_2 & -V_1 \end{pmatrix} \begin{pmatrix} t'_1 \\ t'_2 \\ O_1 \\ O_2 \end{pmatrix} = \begin{pmatrix} V_3 \\ V_4 \\ V_5 \\ V_6 \end{pmatrix}$$
We obtain the following closed-form solution, due to the relation between V_1, V_2 and R' as per step 2.
$$\begin{pmatrix} t'_1 \\ t'_2 \\ O_1 \\ O_2 \end{pmatrix} = \begin{pmatrix} 0 & -1 & -V_1 & -V_2 \\ 1 & 0 & V_2 & -V_1 \\ 0 & 0 & -V_1 & -V_2 \\ 0 & 0 & V_2 & -V_1 \end{pmatrix} \begin{pmatrix} V_3 \\ V_4 \\ V_5 \\ V_6 \end{pmatrix}$$
This implies that the solution is always well-defined.

4. Select sign of λ for which $V_7 = -O_1t'_2 + O_2t'_1$.

The projection rays are then computed for all pixels. In fact only their directions are computed (they are constrained to pass through the computed optical center).

5.2 Calibration of 3D cameras

As in the 2D case, we have a 3D point $\mathbf{O}(O_1, O_2, O_3)$, which lies on all rays. We construct a matrix with $\mathbf{O}$, $\mathbf{Q}$ and $\mathbf{Q}'$ in the same coordinate frame as shown below.
$$\begin{pmatrix} O_1 & Q_1 & R'_{11}Q'_1 + R'_{12}Q'_2 + t'_1Q'_4 \\ O_2 & Q_2 & R'_{21}Q'_1 + R'_{22}Q'_2 + t'_2Q'_4 \\ O_3 & Q_3 & R'_{21}Q'_1 + R'_{22}Q'_2 + t'_3Q'_4 \\ 1 & Q_4 & Q'_4 \end{pmatrix}$$

The collinearity of these three points gives four constraints by removing one of the 4 rows at a time from the above matrix (two of which are shown in Table 5.2). Using some simulations, we observed that the third constraint failed to give a unique solution for the coupled variables. However to extract the motion variables we used only the first and the second constraints.

i	C_i	V_i	W_i
1	$Q_1Q'_1$	0	$R'_{3,1}$
2	$Q_1Q'_2$	0	$R'_{3,2}$
3	$Q_1Q'_3$	0	$R'_{3,3}$
4	$Q_1Q'_4$	0	$-O_3 + t'_3$
5	$Q_2Q'_1$	$R'_{3,1}$	0
6	$Q_2Q'_2$	$R'_{3,2}$	0
7	$Q_2Q'_3$	$R'_{3,3}$	0
8	$Q_2Q'_4$	$-O_3 + t'_3$	0
9	$Q_3Q'_1$	$-R'_{2,1}$	$-R'_{1,1}$
10	$Q_3Q'_2$	$-R'_{2,2}$	$-R'_{1,2}$
11	$Q_3Q'_3$	$-R'_{2,3}$	$-R'_{1,3}$
12	$Q_3Q'_4$	$O_2 - t'_2$	$O_1 - t'_1$
13	$Q_4Q'_1$	$-O_2R'_{3,1} + O_3R'_{2,1}$	$-O_1R'_{3,1} + O_3R'_{1,1}$
14	$Q_4Q'_2$	$-O_2R'_{3,2} + O_3R'_{2,2}$	$-O_1R'_{3,2} + O_3R'_{1,2}$
15	$Q_4Q'_3$	$-O_2R'_{3,3} + O_3R'_{2,3}$	$-O_1R'_{3,3} + O_3R'_{1,3}$
16	$Q_4Q'_4$	$-O_2t'_3 + O_3t'_2$	$-O_1t'_3 + O_3t'_1$

Table 5.2: Coupled variables in the bifocal matching tensors for a 3D single center camera.

Now we explain the procedure to extract the individual motion parameters from the coupled coefficients.

$$\lambda_1 = \pm\sqrt{V'^2_5 + V'^2_6 + V'^2_7},$$

$$V_i = \frac{V'_i}{\lambda_1},$$

$$\lambda_2 = \pm\sqrt{W'^2_1 + W'^2_2 + W'^2_3},$$

$$W_i = \frac{V'_i}{\lambda_2},$$

$$\mathsf{R}' = \begin{pmatrix} -W_9 & -W_{10} & -W_{11} \\ -V_9 & -V_{10} & -V_{11} \\ V_5 & V_6 & V_7 \end{pmatrix},$$

If $det(\mathsf{R}') = +1$ then $\lambda_2 = -\lambda_2$ and $W_i = -W_i$. If $(W_9V_9 + W_{10}V_{10} + W_{11}V_{11}) < 0)$ then $\lambda_1 = -\lambda_1$ and $V_i = -V_i$.

$$O_3 = \frac{V_{13}R'_{3,2} - V_{14}R'_{3,1}}{R'_{2,1}R'_{3,2} - R'_{2,2}R'_{3,1}},$$

$$O_2 = \frac{O_3R'_{2,1} - V_{13}}{R'_{3,1}},$$

$$O_1 = \frac{O_3R'_{1,1} - W_{13}}{R'_{3,1}},$$

$$t'_1 = O_1 - W_{12},$$

$$t'_2 = O_2 - V_{12},$$

$$t'_3 = W_4 + O_3$$

5.3 Calibration of Central 3D cameras using planar calibration patterns

It is possible to obtain constraints with just two views as in the 2D case (section 5.1). However the constraints are insufficient to estimate the motion variables. This is to be expected since even for the pinhole model, full calibration using a planar calibration grid requires three views at least [103, 127]. We thus have to consider three views at least. We build a 4×4 matrix consisting of $\mathbf{Q}$, $\mathbf{Q}$,$\mathbf{Q}'$ and $\mathbf{Q}''$ as shown below.

$$\begin{pmatrix} O_1 & Q_1 & R'_{11}Q'_1 + R'_{12}Q'_2 + t'_1Q'_4 & R''_{11}Q''_1 + R''_{12}Q''_2 + t''_1Q''_4 \\ O_2 & Q_2 & R'_{21}Q'_1 + R'_{22}Q'_2 + t'_2Q'_4 & R''_{21}Q''_1 + R''_{22}Q''_2 + t''_2Q''_4 \\ O_3 & 0 & R'_{31}Q'_1 + R'_{32}Q'_2 + t'_3Q'_4 & R''_{31}Q''_1 + R''_{32}Q''_2 + t''_3Q''_4 \\ 1 & Q_4 & Q'_4 & Q''_4 \end{pmatrix} \tag{5.1}$$

The collinearity of these four points implies that the 4x4 matrix has to be of rank less than three. As a result all 16 3x3 submatrices are singular. In contrast to the earlier 3D scenarios, here we get 16 constraints by removing one row and one column at a time.

Each constraint is a linear equation in coupled variables, depending on motion parameters and the optical center, whose coefficients depend on the known coordinates of calibration points. Using simulations, it was observed that only for six of the sixteen constraints, the associated linear equation systems gave a unique solution for the coupled variables. Finally, we found that four of these allow to estimate the motion variables and the optical center, as below. Note that there may be other combinations of constraints that allow to estimate the unknowns; all complete analysis would have to be the topic of future research.

To distinguish each constraint from the set of 16 constraints we use $T_{i,j}$ to refer to the constraint obtained by removing the i_{th} row and j_{th} column. The constraints corresponding to $T_{4,1}, T_{4,2}, T_{3,1}$ and $T_{3,2}$ are given below.

$$T_{4,1} : \sum_{i=1} C4_i V_i = 0 \tag{5.2}$$

$$T_{4,2} : \sum_{i=1} C4_i W_i = 0 \tag{5.3}$$

$$T_{3,1} : \sum_{i=1} C3_i M_i = 0 \tag{5.4}$$

$$T_{3,2} : \sum_{i=1} C3_i N_i = 0 \tag{5.5}$$

where $C4_i = Q_j Q'_k$,$C3_i = Q_j Q''_k$ and the other variables are given in Table 5.3. The tensor coefficients can only be computed up to scale. The estimated coefficients are V', W', M', and N'.

Again in this scenario the motion estimation is slightly complicated. We first compute some rotation components up to a scale as shown below.

$$\lambda_1 R'_{3,1} = V_4, \;\; \lambda_1 R'_{3,2} = V_5,$$

$$\lambda_2 R''_{3,1} = M_4, \lambda_2 R''_{3,2} = M_5$$

Note that V, W, M and N are not known. Only V', W', M' and N' are known. We obtain the following equations by multiplying certain variables in Table 5.3.

i	j	k	V_i	W_i	M_i	N_i
1	1	1	0	$R'_{3,1}$	0	$R''_{3,1}$
2	1	2	0	$R'_{3,2}$	0	$R''_{3,1}$
3	1	4	0	$-O_3 + t'_3$	0	$-O_3 + t''_3$
4	2	1	$R'_{3,1}$	0	$R''_{3,1}$	0
5	2	2	$R'_{3,2}$	0	$R''_{3,2}$	0
6	2	4	$-O_3 + t'_3$	0	$-O_3 + t''_3$	0
7	4	1	$-O_2R'_{3,1} + O_3R'_{2,1}$	$-O_1R'_{3,1} + O_3R'_{1,1}$	$-O_2R''_{3,1} + O_3R''_{2,1}$	$-O_1R''_{3,1} + O_3R''_{1,1}$
8	4	2	$-O_2R'_{3,2} + O_3R'_{2,2}$	$-O_1R'_{3,2} + O_3R'_{1,2}$	$-O_2R''_{3,2} + O_3R''_{2,2}$	$-O_1R''_{3,2} + O_3R''_{1,2}$
9	4	4	$-O_2t'_3 + O_3t'_2$	$-O_1t'_3 + O_3t'_1$	$-O_2t''_3 + O_3t''_2$	$-O_1t''_3 + O_3t''_1$

Table 5.3: Coupled variables in four of the bifocal matching tensors in a 3D single center camera with a planar calibration grid. Among the 16 tensors, there are 4 trifocal and 12 bifocal tensors.

$$\begin{pmatrix}
0 & -V_4 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -V_5 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -V'_6 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-V_4 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-V_5 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-V'_6 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -M_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & -M_5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & -M'_6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
-M_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
-M_5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
-N'_6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
O_1 \\
O_2 \\
\lambda_1 O_3(t'_1 - O_1) \\
\lambda_1 O_3(t'_2 - O_2) \\
\lambda_1 O_3 R'_{1,1} \\
\lambda_1 O_3 R'_{1,2} \\
\lambda_1 O_3 R'_{2,1} \\
\lambda_1 O_3 R'_{2,2} \\
\lambda_2 O_3(t''_1 - O_1) \\
\lambda_2 O_3(t''_2 - O_2) \\
\lambda_2 O_3 R''_{1,1} \\
\lambda_2 O_3 R''_{1,1} \\
\lambda_2 O_3 R''_{2,1} \\
\lambda_2 O_3 R''_{2,2}
\end{pmatrix} =
\begin{pmatrix}
V'_7 \\ V'_8 \\ V'_9 \\ W'_7 \\ W'_8 \\ W'_9 \\ M'_7 \\ M'_8 \\ M'_9 \\ N'_7 \\ N'_8 \\ M'_9
\end{pmatrix}$$

$$\mathsf{A}_{12\times 14}\mathbf{u} = \mathbf{b} \tag{5.6}$$

We consider the above equation system on 14 coupled variables (couplings of the scale factors, motion parameters and the optical center). Then, since the equation system is of rank 12, one may express the 14 coupled variables as a linear combination of three vectors, one satisfying the matrix equation 5.6 and the two null-vectors of $\mathsf{A}_{12\times 14}$. The method to compute the solution of equations of the form $\mathsf{A}\mathbf{u} = \mathbf{b}$ using a

linear combination of vectors including the null vectors is given in the appendix of [49].

$$\begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ u_7 \\ u_8 \\ u_9 \\ u_{10} \\ u_{11} \\ u_{12} \\ u_{13} \\ u_{14} \end{pmatrix} = \begin{pmatrix} O_1 \\ O_2 \\ \lambda_1 O_3(t'_1 - O_1) \\ \lambda_1 O_3(t'_2 - O_2) \\ \lambda_1 O_3 R'_{1,1} \\ \lambda_1 O_3 R'_{1,2} \\ \lambda_1 O_3 R'_{2,1} \\ \lambda_1 O_3 R'_{2,2} \\ \lambda_2 O_3(t''_1 - O_1) \\ \lambda_2 O_3(t''_2 - O_2) \\ \lambda_2 O_3 R''_{1,1} \\ \lambda_2 O_3 R''_{1,1} \\ \lambda_2 O_3 R''_{2,1} \\ \lambda_2 O_3 R''_{2,2} \end{pmatrix} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \\ a_9 \\ a_{10} \\ a_{11} \\ a_{12} \\ a_{13} \\ a_{14} \end{pmatrix} + l_1 \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \\ b_8 \\ b_9 \\ b_{10} \\ b_{11} \\ b_{12} \\ b_{13} \\ b_{14} \end{pmatrix} + l_2 \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \\ c_8 \\ c_9 \\ c_{10} \\ c_{11} \\ c_{12} \\ c_{13} \\ c_{14} \end{pmatrix}$$

We now use the orthonormality properties of the rotation matrices R' and R'' to compute the values of l_1 and l_2.

$$u_5 u_6 + u_7 u_8 + V_4 V_5 O_3^2 = 0$$

This is because the scalar project of two columns of rotation matrix is zero.

$$u_{11} u_{12} + u_{13} u_{14} + M_4 M_5 O_3^2 = 0$$

$$u_5^2 + u_7^2 + (V_4 O_3)^2 = \lambda_1^2 O_3^2$$
$$u_6^2 + u_8^2 + (V_4 O_3)^2 = \lambda_1^2 O_3)^2$$
$$u_{11}^2 + u_{13}^2 + (M_4 O_3)^2 = \lambda_2^2 O_3)^2$$
$$u_{12}^2 + u_{14}^2 + (M_5 O_3)^2 = \lambda_2^2 O_3)^2$$

The above system can be expressed with the following eight variables (u_i's): l_1,l_2,$l_1 l_2$,l_1^2,l_2^2,O_3^2, $(\lambda_1 O_3)^2$ and $(\lambda_2 O_3)^2$.

$$A_{12} = \begin{pmatrix} a_5 b_6 + b_5 a_6 + a_7 b_8 + b_7 a_8 & a_5 c_6 + c_5 a_6 + a_7 c_8 + c_7 a_8 \\ a_{11} b_{12} + b_{11} a_{12} + a_{13} b_{14} + b_{13} a_{14} & a_{11} c_{12} + c_{11} a_{12} + a_{13} c_{14} + c_{13} a_{14} \\ 2a_5 b_5 + 2a_7 b_7 & 2c_5 a_5 + 2c_7 a_7 \\ 2a_6 b_6 + 2a_8 b_8 & 2c_6 a_6 + 2c_8 a_8 \\ 2a_{11} b_{11} + 2a_{13} b_{13} & 2c_{11} a_{11} + 2c_{13} a_{13} \\ 2a_{12} b_{12} + 2a_{14} b_{14} & 2c_{12} a_{12} + 2c_{14} a_{14} \end{pmatrix}$$

$$A_{345} = \begin{pmatrix} b_5 c_6 + c_5 b_6 + b_7 c_8 + c_7 b_8 & b_5 b_6 + b_7 b_8 & c_5 c_6 + c_7 c_8 \\ b_{11} c_{12} + c_{11} b_{12} + b_{13} c_{14} + c_{13} b_{14} & b_{11} b_{12} + b_{13} b_{14} & c_{11} c_{12} + c_{13} c_{14} \\ 2b_5 c_5 + 2c_7 b_7 & {b_5}^2 + {b_7}^2 & {c_5}^2 + {c_7}^2 \\ 2b_6 c_6 + 2c_8 b_8 & {b_6}^2 + {b_8}^2 & {c_6}^2 + {c_8}^2 \\ 2b_{11} c_{11} + 2c_{13} b_{13} & {b_{11}}^2 + {b_{13}}^2 & {c_{11}}^2 + {c_{13}}^2 \\ 2b_{12} c_{12} + 2c_{14} b_{14} & {b_{12}}^2 + {b_{14}}^2 & {c_{12}}^2 + {c_{14}}^2 \end{pmatrix}$$

$$A_{678} = \begin{pmatrix} V4V5 & 0 & 0 \\ M4M5 & 0 & 0 \\ V4^2 & -1 & 0 \\ V5^2 & -1 & 0 \\ M4^2 & 0 & -1 \\ M4^2 & 0 & -1 \end{pmatrix}$$

$$\begin{pmatrix} A_{12} & A_{345} & A_{678} \end{pmatrix} \begin{pmatrix} l_1 \\ l_2 \\ l_1 l_2 \\ {l_1}^2 \\ {l_2}^2 \\ O3^2 \\ (l_1 O3)^2 \\ (l_2 O3)^2 \end{pmatrix} = 0$$

As a result we have eight variables in 6 equations. The rank of this system is 5 and thus we obtain the solution for these 8 variables in the subspace spanned by three vectors and linear combination factors m_1 and m_2 (similar to the approach used for equation 5.6).

$$\begin{pmatrix} l_1 \\ l_2 \\ l_1 l_2 \\ {l_1}^2 \\ {l_2}^2 \\ O3^2 \\ (l_1 O3)^2 \\ (l_2 O3)^2 \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \\ d_7 \\ d_8 \end{pmatrix} + m_1 \begin{pmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \\ e_7 \\ e_8 \end{pmatrix} + m_2 \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \\ f_8 \end{pmatrix}$$

However, the solutions for l_1 and l_2 are obtained without any ambiguities. In other words, e_1, e_2, f_1 and f_2 are always zeros. Such a behaviour is because of the internal constraints that exist between the different variables. This is verified using simulations with random data. As a result we have the following.

$$l_1 = d_1$$

$$l_2 = d_2$$

Next we estimate the 14 variables (u_i's). From the u_i's we estimate the individual motion parameters.

$$O_3 = \sqrt{\frac{u_5 u_6 + u_7 u_8}{-V_4 V_5}}, \lambda_1 = \pm \frac{\sqrt{u_5^2 + u_7^2 + (V_4 O_3)^2}}{O3}, \lambda_2 = \pm \frac{\sqrt{u_{11}^2 + u_{13}^2 + (M_4 O_3)^2}}{O3}$$

$$R'_{1,1} = \frac{u_5}{O_3 \lambda_1}, R'_{1,2} = \frac{u_6}{O_3 \lambda_1}, R'_{2,1} = \frac{u_7}{O_3 \lambda_1}, R'_{2,2} = \frac{u_8}{O_3 \lambda_1},$$

$$R''_{1,1} = \frac{u_{11}}{O_3 \lambda_2}, R''_{1,2} = \frac{u_{12}}{O_3 \lambda_2}, R''_{2,1} = \frac{u_{13}}{O_3 \lambda_2}, R''_{2,2} = \frac{u_{14}}{O_3 \lambda_2},$$

$$R'_{3,1} = \frac{V_4}{\lambda_1}, R'_{3,2} = \frac{V_5}{\lambda_1}, R''_{3,1} = \frac{M_4}{\lambda_2}, R''_{3,2} = \frac{M_5}{\lambda_2},$$

$$t'_1 = \frac{u_3 + O_1 O_2 \lambda_1}{O_3 \lambda_1}, t'_2 = \frac{u_4 + O_2 O_3 \lambda_1}{O_3 \lambda_1}, t'_3 = \frac{O_3 t'_2 - V_9}{O_2},$$

$$t_1'' = \frac{u_9 + O_1O_2\lambda_2}{O_3\lambda_2}, t_2'' = \frac{u_{10} + O_2O_3\lambda_2}{O_3\lambda_2}, t_3' = \frac{O_3t_2'' - M_9}{O_2},$$

Now we have to fit the rays to a single optical center **O**. We use least squares approach to compute the optimal ray. Let D represent the direction of a projection ray passing through Q_i, $i = 1..3$, on the ray. We have to minimize the following expression to compute D and λ_i is a parameter corresponding to the closest point on the ray to a given point Q_i.

$$\min_{\lambda_i D} \sum_i^n |O + \lambda_i D - Q_i|^2$$

Let us replace $Q_i - O$ by V_i. We apply the constraint $D^T D = 1$. We have $\lambda_i = V_i^T D$.

$$\min_{\lambda_i D} \sum_i^n |O + \lambda_i D - Q_i|^2 = \min_D \sum_i^n ((V_i^T D)D - V_i)^T (V_i^T D)D - V_i)$$

$$= \min_D \sum ((V_i^T D)^2 D^T D - 2(V_i^T D)^2 + V_i^T V_i) = \min_D \sum (V_i^T D)^2 + \sum V_i^T V_i$$

$$= \min_D \sum V_i^T V_i - D^T (\sum V_i V_i^T) D = \min_D \sum V_i^T V_i - D^T S D$$

The minimum has to be computed under the condition $D^T D = 1$. Let us look at the eigenvalues m_i of S.

$$\min_{|D|=1} (-D^T S D) = -D^T (S - m_i I) D$$

Thus the eigenvector corresponding to the largest eigenvalue, scaled to norm 1, will be the least squares solution for vector D.

5.4 Experimental Evaluation

5.4.1 Pinhole and fisheye cameras

In Figure 5.2 we can observe that we have calibrated a major portion of the image with just three grids. The calibrated camera rays and the pose of the calibration objects, which is given by the estimated motion parameters, are shown. The calibration grids, which are of different sizes, used in the calibration are shown in Figure 5.1. 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. Using singular value analysis of the tensors we found that three differently sized calibration grids produce stable solutions. The different sizes allow the calibration grids to be non intersecting, well aligned to each other and still have a large common area of intersection.

5.4.2 Applications

We have considered 3 different applications to evaluate our calibration approach.

- Distortion correction
- structure and motion recovery
- epipolar curve analysis

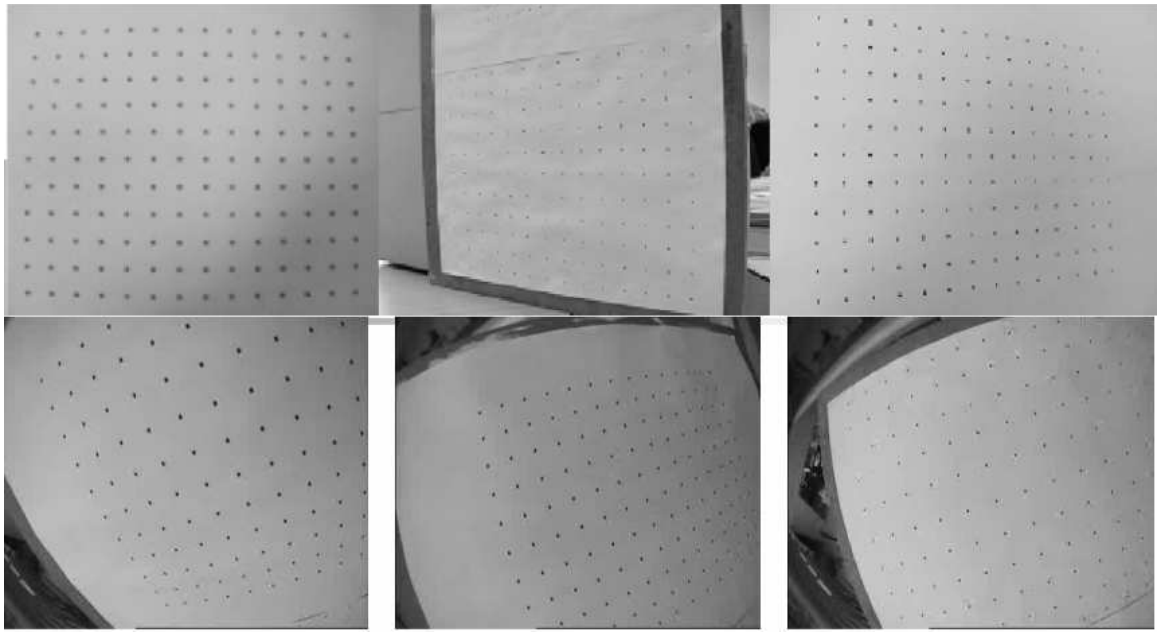
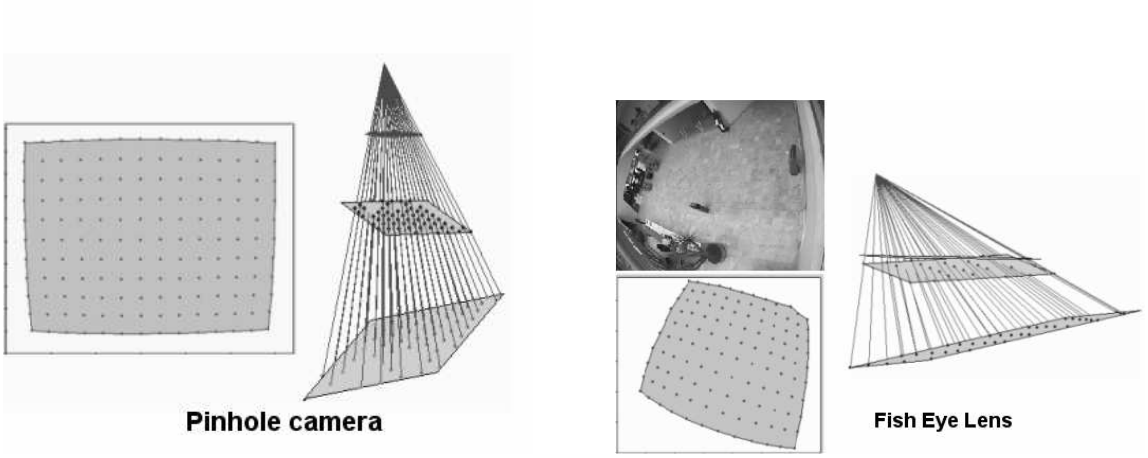


Figure 5.1: Images of three grids, which are of different sizes, captured by a pinhole camera and a fish eye lens are shown in top and bottom respectively.



(a) (b)

Figure 5.2: The shaded region shows the calibrated region and the 3D rays along with the calibration grids.

Distortion correction

At the end of the calibration for a central camera, we obtain a bundle of rays passing through the optical center. To each can be associated a specific color, borrowed from its image pixel. To obtain a perspective view we first intersect the bundle of rays with a plane on which we want the perspective view. We give the color of the ray to its point of intersection on the plane. Finally we compute the bounding box for the points of intersection on the plane and use interpolation or nearest neighbor color for the rest of the points inside the bounding box. We show a simple technique to synthesize partial perspective images from various parts of a fisheye image in Figure 5.3.

We show perspective image synthesis for two fisheye images in Figure 5.4.

Motion Recovery

After obtaining image correspondences between two images of an arbitrary scene, we compute the motion between the two viewpoints by applying the constraint that the corresponding 3D rays intersect (see Figure 5.5). Let $\mathbf{O}$ and $\mathbf{O}'$ be the camera centers for the first and the second views. Let us consider a point $\mathbf{X}$, which is seen by both views. The point at infinity along the ray $\mathbf{OX}$ is $\mathbf{a}$. Let $\mathbf{a}'$ be the point at infinity in a different coordinate system. Let the motion from the second view to the first view be given by $(\mathsf{R}, \mathbf{t})$. Let $\mathbf{D}$ be the point at infinity along the ray $\mathbf{OO}'$. Since a point at infinity is not affected by translation, $\mathsf{R}\mathbf{a}'$ lies along the ray $\mathbf{O}'\mathbf{X}$. Now we know that $\mathbf{a}$, $\mathsf{R}\mathbf{a}'$ and D lie on a single line, which is a line at infinity. This can be expressed as:

$$\mathbf{a}'^{T} \underbrace{\mathsf{R}^{T}[\mathbf{D}]_{\times}}_{\mathsf{E}} \mathbf{a} = 0$$

We need 8 correspondences to extract the motion parameters from the 3×3 essential matrix E. Note that the above algorithm is applicable only to central cameras. The algorithm for non-central cameras is given in chapter 8.

Epipolar Curves

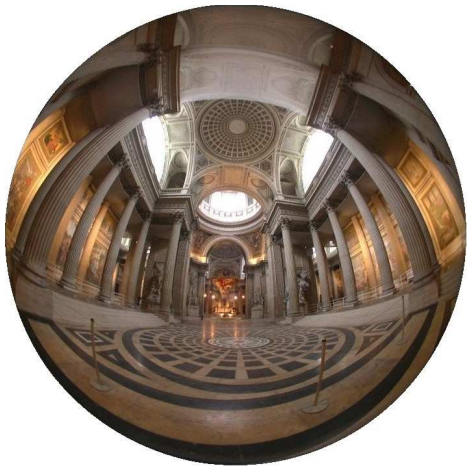
For computing the epipolar curve in the second view for a pixel x in the first view, we first compute the rays of the second view which intersect the ray corresponding to x. Image pixels of these intersecting rays form the epipolar curve. In Figure 5.6 we show the epipolar curves for pinhole and fisheye cameras, computed after motion recovery. Note that this concept is valid for non-central cameras too.

Structure Recovery

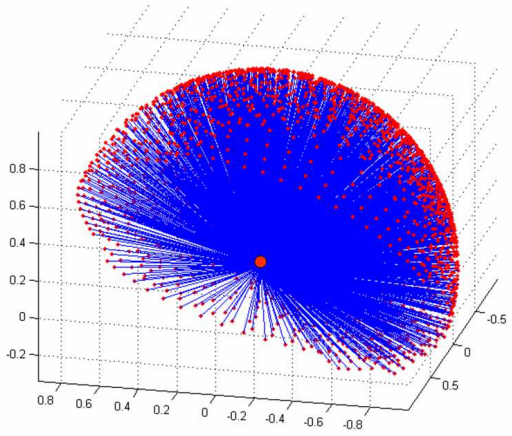
After computing image correspondences, we intersect the rays corresponding to matching pixels to compute the associated 3D point in space. Some of the results of this 3D reconstruction algorithm using ray intersection for images are shown in Figure 5.7. We found the results to be reasonably accurate. The angle between the right angled faces in the house is estimated to be 89.5° without doing any nonlinear refinement. A more detailed analysis on the accuracy of structure recovery is given in chapter 8.

5.5 Conclusions

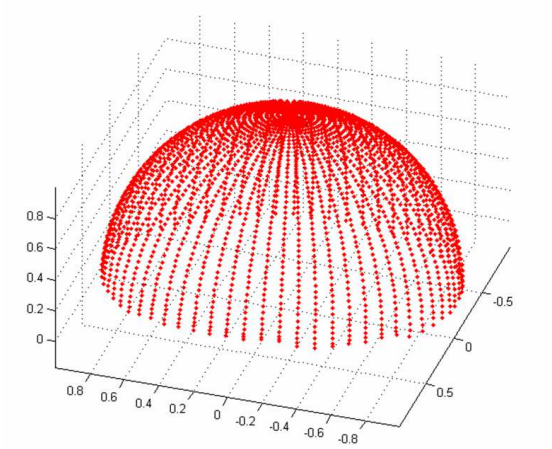
Experiments show that our approach allows to calibrate central cameras without using any analytical distortion model, with applications in distortion correction, motion estimation and structure recovery.



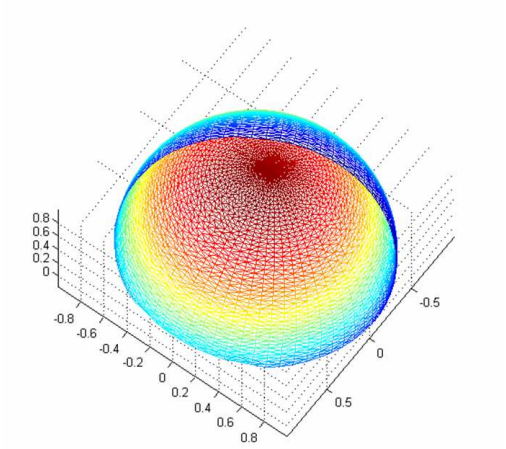
(a)



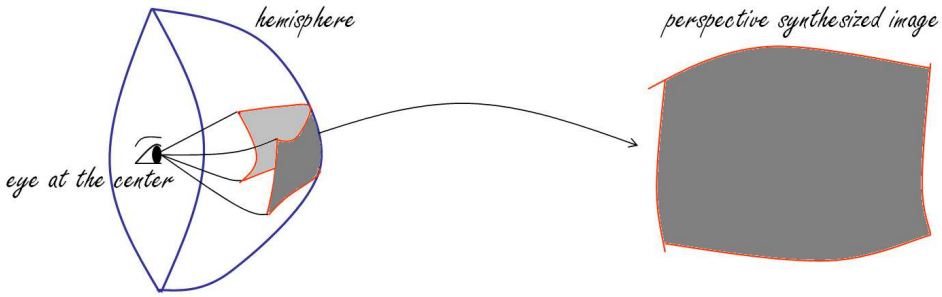
(b)



(c)

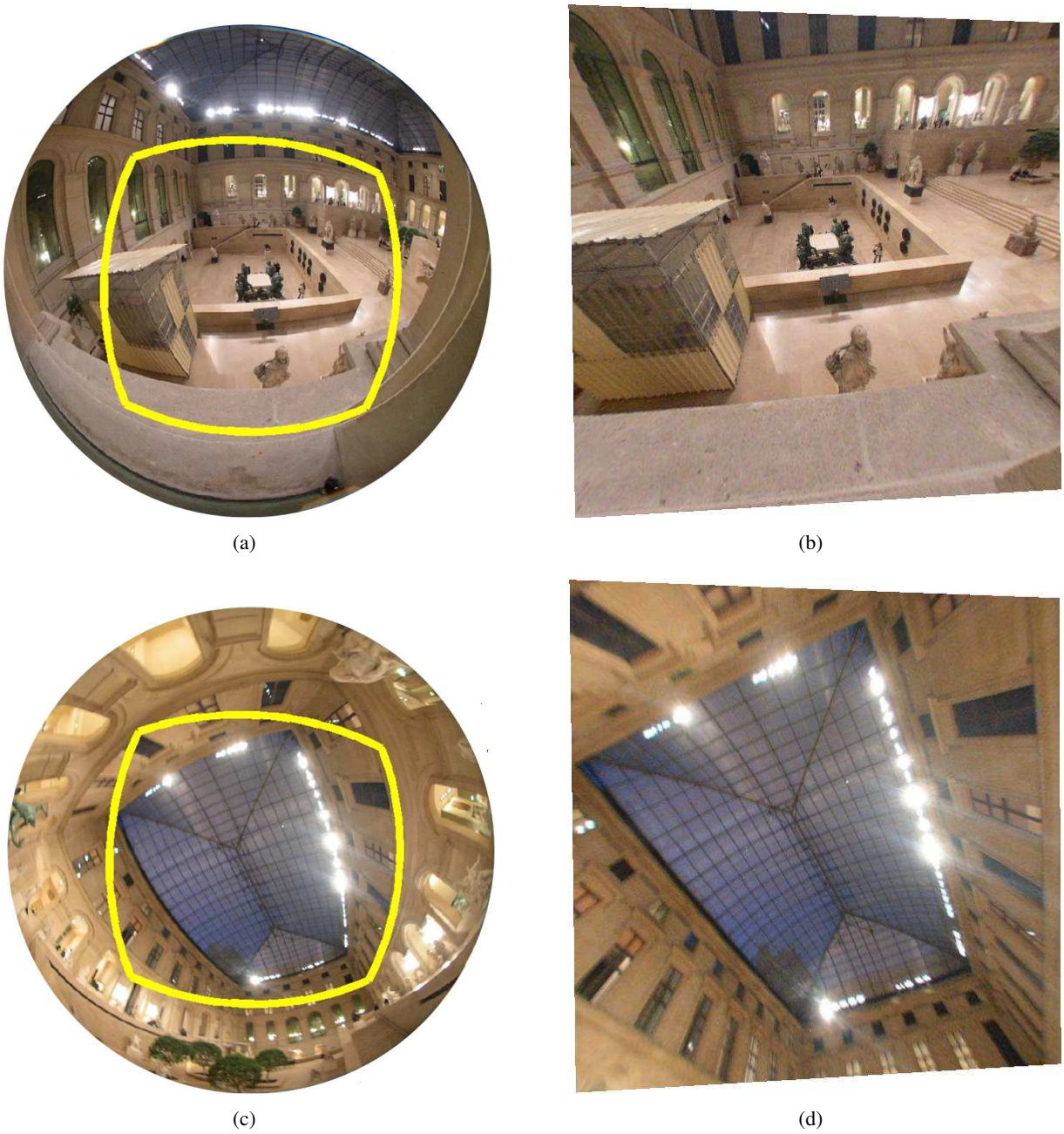


(d)



(e) (f)

Figure 5.3: We briefly explain our technique for distortion correction. (a) Original fisheye image. (b) reconstructed projection rays (c) uniformly distributed projection rays (only the end points are shown to represent the directions) (d) Uniformly distributed points are triangulated. (e) The triangulated mesh is texture mapped with the fisheye image. (e) The viewpoint of the synthetic perspective camera is fixed at the center of the hemisphere (i.e. the optical center of the original image). Perspectively correct images can now be obtained by rendering the textured mesh onto the perspective camera. Different parts of the original image may thus be corrected for distortions by letting the synthetic perspective camera rotate. The intrinsic parameters of the perspective camera determine the size of the section of the original image that is corrected.



(a) (b)

(c) (d)

Figure 5.4: Perspective image synthesis for a partially calibrated fisheye image is shown. (a) The calibrated region of a fisheye image of Louvre museum in Paris. (b) Distortion corrected perspective image. (c) and (d) show the fisheye image and distortion corrected image of the ceiling in Louvre respectively.

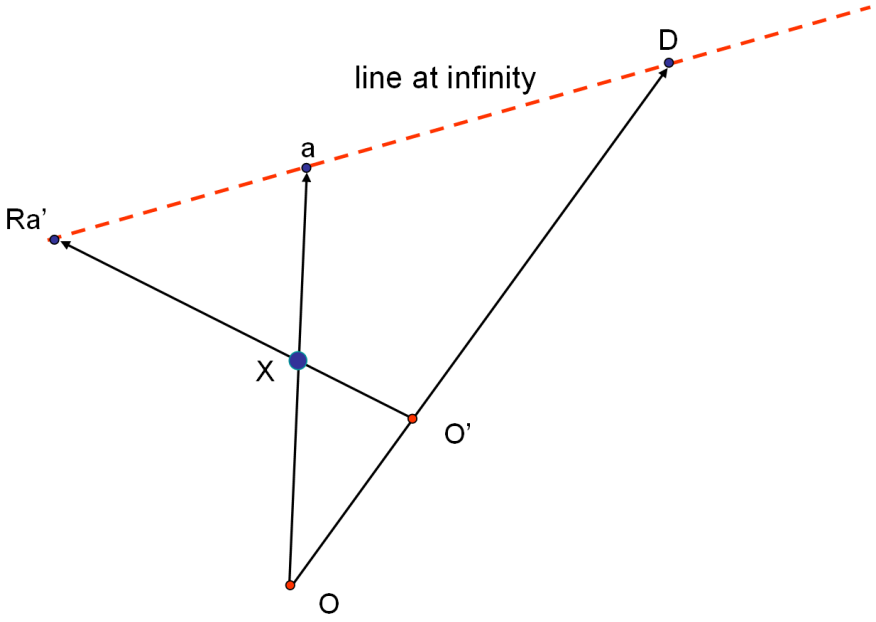


Figure 5.5: Motion Recovery.

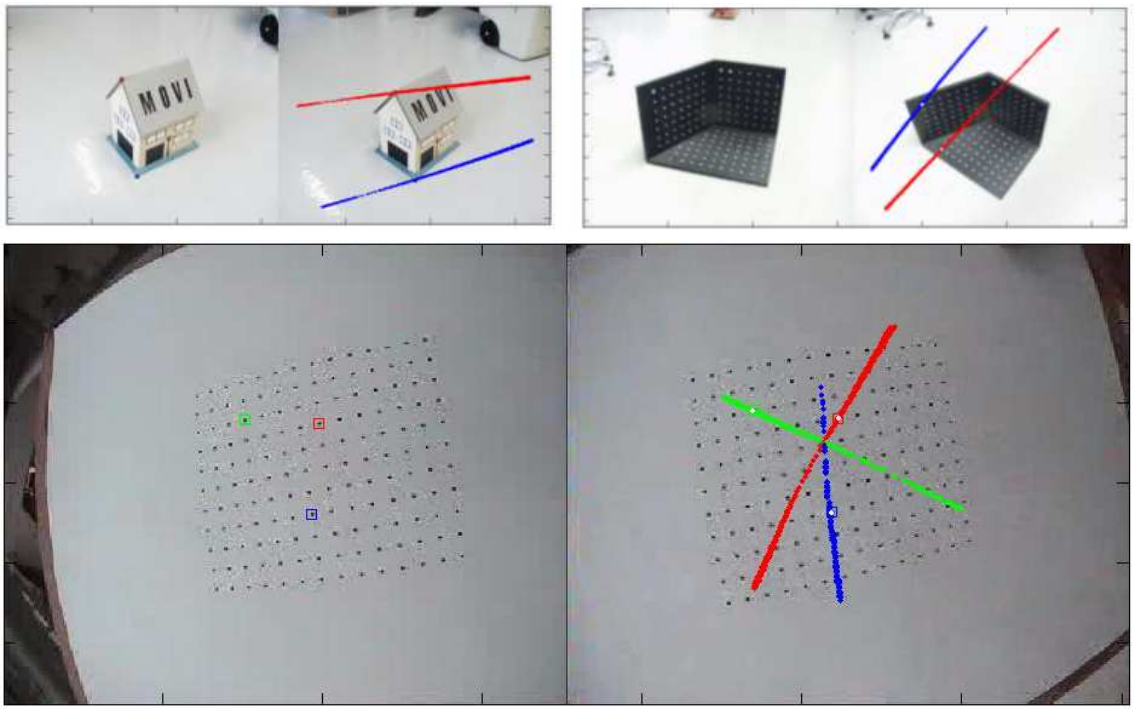
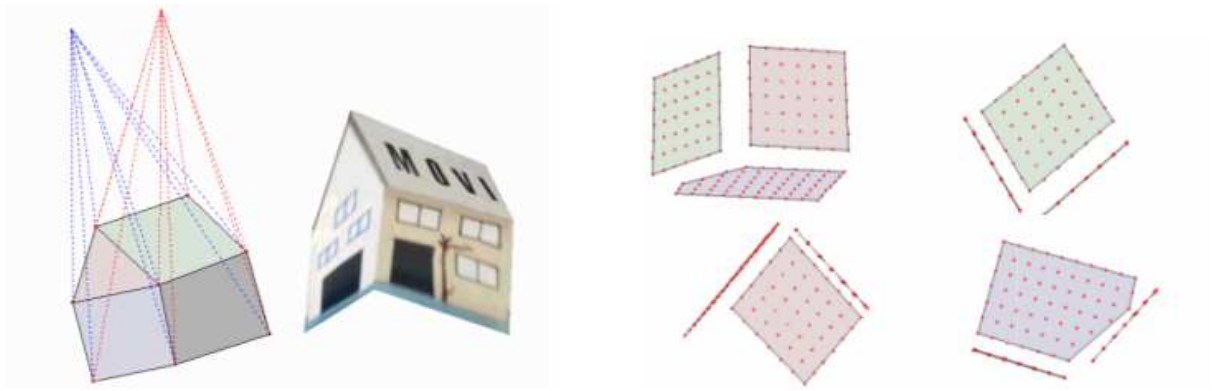


Figure 5.6: Top: Epipolar curves (for 2 points) for two scenes in pinhole images. Bottom: Epipolar curves (for 3 points) for fisheye images. These are not straight lines, but intersect in a single point (epipole), since we use a central model.



(a) (b)

Figure 5.7: We show the results of the ray intersection technique for obtaining the 3D reconstruction for two scenes, a) house and b) calibration pattern. The images used in the reconstruction are shown in Figure 5.6.

We believe in the concept's potential for calibrating cameras with "exotic" distortions – such as fish-eye lenses with hemispheric field of view or catadioptric cameras. To this end, we need to make the method stabler, e.g. by extending it such that it can use many images of calibration grids and by using bundle adjustment techniques. These issues are dealt with in chapter 7. That chapter also presents a complete set of experiments to validate our theory and success of generic calibration. Appendix A.1 lists different calibration algorithms and the nature of solutions (unique, degenerate, inconsistent, etc.), that will be obtained, by applying various algorithms for different camera models. Another interesting study would be to look at calibration algorithms in the light of partial motion. When we have partial knowledge about the motion we can develop tailor-made calibration algorithms. For example, if we know that the motion is either pure translation or pure rotation, we can have simpler calibration algorithms. The generic calibration of central cameras under restricted motion scenarios is studied in Appendix A.2. In the next chapter we consider a camera model which is slightly more general than the central one, but more restricted than the non-central one. We refer to this class of cameras as axial cameras where all projection rays intersect a single line in space.

Chapter 6

Axial 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 chapter, 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.

The axial model is a rather useful one (cf. figure 6.1(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 [2, 63, 86]. 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 [42] are another example, although they are of a more restricted class (there exist two camera axes [31]).

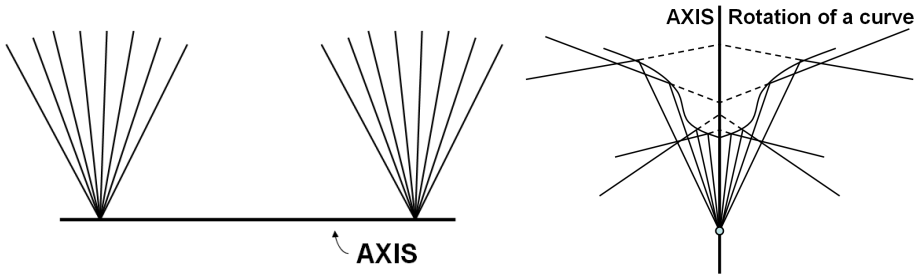
In this chapter, we propose a generic calibration approach for axial cameras, the first to our knowledge. It uses images of planar calibration grids, put in unknown positions. We show the existence of multi-view tensors that can be estimated linearly and from which the pose of the calibration grids as 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.

This chapter is organized as follows. The problem is formalized in section 6.1. In section 6.2.1, we show what can be done with two images of calibration grids. Complete calibration using three images, is described in section 6.3.2. Various types of axial catadioptric cameras are listed in section 6.5. Experimental results and conclusions are given in sections 6.6 and 6.7.

6.1 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 $[\mathbf{L}]_\times$:

$$[\mathbf{L}]_\times = \begin{pmatrix} 0 & -L_4 & L_6 & -L_2 \\ L_4 & 0 & -L_5 & -L_3 \\ -L_6 & L_5 & 0 & -L_1 \\ L_2 & L_3 & L_1 & 0 \end{pmatrix}$$



(a) (b)

Figure 6.1: 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.

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:

$$\mathbf{A} = \begin{pmatrix} L_5 \\ L_6 \\ L_4 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} L_2 \\ L_3 \\ L_1 \end{pmatrix}$$

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 thesis is to compute projection rays for all pixels of a camera, from images of 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.

6.2 Calibration of Axial Cameras with 3D Object

6.2.1 What can be done with two views of 3D 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.

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:

$$\begin{pmatrix} Q_1 \\ Q_2 \\ Q_3 \\ Q_4 \end{pmatrix}^{\mathsf{T}} \begin{pmatrix} 0 & -L_4 & L_6 & -L_2 \\ L_4 & 0 & -L_5 & -L_3 \\ -L_6 & L_5 & 0 & -L_1 \\ L_2 & L_3 & L_1 & 0 \end{pmatrix} \begin{pmatrix} \mathsf{R}' & \mathbf{t}' \\ \mathbf{0}^{\mathsf{T}} & 1 \end{pmatrix} \begin{pmatrix} Q'_1 \\ Q'_2 \\ Q'_3 \\ Q'_4 \end{pmatrix} = 0$$

We thus have the following 4×4 tensor that can be estimated linearly from point correspondences:

$$\mathsf{F} \sim \begin{pmatrix} 0 & -L_4 & L_6 & -L_2 \\ L_4 & 0 & -L_5 & -L_3 \\ -L_6 & L_5 & 0 & -L_1 \\ L_2 & L_3 & L_1 & 0 \end{pmatrix} \begin{pmatrix} \mathsf{R}' & \mathbf{t}' \\ \mathbf{0}^{\mathsf{T}} & 1 \end{pmatrix} \tag{6.1}$$

One can extract, from F, the complete camera axis $\mathbf{L}$, in the coordinate frame of each grid.
Apply, to each grid, a rotation and translation such that the camera axis becomes the z axis, i.e,

$$[\mathbf{L}]_\times = \begin{pmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

These rotations and translations are not unique only one will do. This fixes the $(\mathsf{R}', \mathbf{t}')$ of the second grid up to

- rotation about z axis (camera axis)

  $$\mathsf{R}' = \begin{pmatrix} c & -s & 0 \\ s & c & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

  with $c = cos(\theta)$ and $s = sin(\theta)$ for some angle θ.

- translation along z axis.

  $$t' = \begin{pmatrix} 0 \\ 0 \\ Z \end{pmatrix}$$

- F is now of the form

  $$\mathsf{F} = [\mathbf{L}]_X \begin{pmatrix} \mathsf{R}' & \mathbf{t}' \\ \mathbf{0}^{\mathsf{T}} & 1 \end{pmatrix} = \begin{pmatrix} -s & -c & 0 & 0 \\ c & -s & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

- Extract θ:

  1. Scale F such that $(F_{11})^2 + (F_{12})^2 = 1$. There are two solutions, defined up to sign.
  2. Compute θ. There will be two solutions for R', up to 180° rotation about Z.

     $$\begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

  3. t'_3 remains undetermined.

6.2.2 Full calibration using three views of 3D calibration grids

We use the technique described in the previous section on the image pairs $(1,2)$ and $(1,3)$. Our goal is to compute $\mathbf{t}'_3$ and $\mathbf{t}''_3$, and fix the 2-fold ambiguities for R' and R''. Since $\mathbf{Q}$, $\mathbf{Q}'$ and $\mathbf{Q}''$ are collinear we form the following matrix.

$$\mathsf{M} = \begin{pmatrix} Q_1 & \pm Q'_1 & \pm Q''_1 \\ Q_2 & \pm Q'_2 & \pm Q''_2 \\ Q_3 & \pm Q'_3 + t'_3 Q'_3 & \pm Q''_3 + t''_3 Q''_4 \\ Q_4 & \pm Q'_4 & \pm Q''_4 \end{pmatrix}$$

- For the 2-fold ambiguities in the sign We consider the following subdeterminant after removing the third row of M.

$$det \begin{pmatrix} Q_1 & \pm Q'_1 & \pm Q''_1 \\ Q_2 & \pm Q'_2 & \pm Q''_2 \\ Q_4 & \pm Q'_4 & \pm Q''_4 \end{pmatrix} = 0$$

  There are four variables $(Q'_1, Q'_2, Q''_1, Q''_2)$ with sign ambiguities. We take several point triplets $(\mathbf{Q}_i, \mathbf{Q}'_i, \mathbf{Q}''_i)$ and compute the value of the above determinant. We try all 16 combinations (two possibilities for every variable) and only one combination will give zero determinant values for all the general triplets.

- Compute t'_3 and t''_3. By removing the rows 1, 2 or 4 from the matrix M we get a constraint of the following form.

$$det \begin{pmatrix} Q_i & Q'_i & Q''_i \\ Q_j & Q'_j & Q''_j \\ Q_3 & Q'_3 + t'_3 Q'_4 & Q''_3 + t''_3 Q''_4 \end{pmatrix} = 0$$

$$\begin{pmatrix} Q'_4(Q_j Q''_i - Q_i Q''_j) & Q''_4(Q_i Q'_j - Q_j Q'_i) & Q_i Q'_j Q''_3 + Q_3 Q'_i Q''_j + Q_j Q'_3 Q''_i \end{pmatrix} \begin{pmatrix} t'_3 \\ t''_3 \\ 1 \end{pmatrix} = 0$$

  Stack all equations for all points and for removing rows 1, 2 and 4, and solve the system. Scale the 3-vector obtained to have 3rd coordinate equal to unity. Read off t'_3 and t''_3.

6.3 Calibration of Axial Cameras with 2D Objects

6.3.1 What can be done with two views of 2D 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 columns. We thus have the following 3×3 tensor that can be estimated linearly from point correspondences:

$$\mathsf{F} \sim \begin{pmatrix} 0 & -L_4 & L_6 & -L_2 \\ L_4 & 0 & -L_5 & -L_3 \\ L_2 & L_3 & L_1 & 0 \end{pmatrix} \begin{pmatrix} \bar{\mathsf{R}}' & \mathbf{t}' \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \tag{6.2}$$

F is in general of rank 2. One could give the following interpretation of F: F can be interpreted as the fundamental matrix of a pair of perspective images, as follows. Consider the two planar grids as image planes of two perspective cameras, and the camera axis of the axial camera as the baseline containing the optical centers of the two perspective cameras. The epipolar geometry of this stereo system is given by F. 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 given above). 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, see below. 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 $\mathbf{t}'$. This is not surprising since even in the more constrained case of central cameras, two planar grids are not sufficient for calibration. Thus we use three calibration grids as described below.

6.3.2 Full calibration using three views of 2D calibration grids

Let $\mathbf{Q}$, $\mathbf{Q}'$, $\mathbf{Q}''$ refer to three calibration grid points corresponding to a single pixel. 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. Every 3×3 submatrix of the following 4×3 matrix has zero determinant.

$$\begin{pmatrix} \mathbf{Q} & \begin{pmatrix} \mathsf{R}' & \mathbf{t}' \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \mathbf{Q}' & \begin{pmatrix} \mathsf{R}'' & \mathbf{t}'' \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \mathbf{Q}'' \end{pmatrix}$$

The submatrices constructed by removing the first and the second rows lead to the constraints $\sum C_i V_i = 0$ and $\sum C_i W_i = 0$ respectively (as described in Table 6.1). These are nothing but homogeneous linear systems of the form $\mathsf{A}\mathbf{X} = 0$. The unknown vector $\mathbf{X}$ is formed from the 23 variables C_i (14 each for V and W). All of these variables are coupled coefficients of the poses of the grids. The matrix A is constructed by stacking the trilinear tensors V and W, which can be computed from the coordinates of $\mathbf{Q}$, $\mathbf{Q}'$ and $\mathbf{Q}''$. The V and W are not trifocal tensors. They are unique for each triplet of corresponding points and thus do not reflect or represent the trifocal geometry. First, we explain how we formulated the Table 6.1: Table 4.4 gives two trifocal tensors for the generic imaging models. Each tensor has 14 non-zero coefficients. They share 5 coefficients, giving 23 different coefficients. One can estimate these using the equations underlying the two original tensors. However, in the case of axial cameras, the estimation is underconstrained, similarly but to a lesser degree than with central cameras (cf. chapter 5). Hence, one has to use constraints characterizing axial cameras in order to estimate the grid poses. Let us first remind that the above trifocal tensors express collinearity constraints: points seen by the same pixel have to lie on the same projection ray. In the case of axial cameras, we have further constraints: there exists a line, the camera axis, which is by definition

coplanar with all projection rays. In order to exploit this constraint, we first simplify the problem, by exploiting the information that can be extracted from the bifocal tensors introduced in section 6.3.1. This is shown next, followed by the description of how to use the coplanarity constraints associated with the camera axis to solve the calibration for axial cameras.

In the following when we refer to the rank of a linear system $\mathsf{A}\mathbf{X} = 0$, we refer to the rank of the matrix A. The rank has to be one less than the number of variables to estimate the variables uniquely up to 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 (see chapter 4). 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 need more constraints. Note that we consider only the planar calibration grids in this section.

i	Motion (C_i)	V_i	W_i
1	R'_{31}	$Q_2Q'_1Q''_4$	$Q_1Q'_1Q''_4$
2	R'_{32}	$Q_2Q'_2Q''_4$	$Q_1Q'_2Q''_4$
3	R''_{31}	$-Q_2Q'_4Q''_1$	$-Q_1Q'_4Q''_1$
4	R''_{32}	$-Q_2Q'_4Q''_2$	$-Q_1Q'_4Q''_2$
5	$t'_3 - t''_3$	$Q_2Q'_4Q''_4$	$Q_1Q'_4Q''_4$
6	$R'_{11}R''_{31} - R'_{31}R''_{11}$	0	$Q_4Q'_1Q''_1$
7	$R'_{11}R''_{32} - R'_{31}R''_{12}$	0	$Q_4Q'_1Q''_2$
8	$R'_{12}R''_{31} - R'_{32}R''_{11}$	0	$Q_4Q'_2Q''_1$
9	$R'_{12}R''_{32} - R'_{32}R''_{12}$	0	$Q_4Q'_2Q''_2$
10	$R'_{21}R''_{31} - R'_{31}R''_{21}$	$Q_4Q'_1Q''_1$	0
11	$R'_{21}R''_{32} - R'_{31}R''_{22}$	$Q_4Q'_1Q''_2$	0
12	$R'_{22}R''_{31} - R'_{32}R''_{21}$	$Q_4Q'_2Q''_1$	0
13	$R'_{22}R''_{32} - R'_{32}R''_{22}$	$Q_4Q'_2Q''_2$	0
14	$R'_{11}t''_3 - R'_{31}t''_1$	0	$Q_4Q'_1Q''_4$
15	$R'_{12}t''_3 - R'_{32}t''_1$	0	$Q_4Q'_2Q''_4$
16	$R'_{21}t''_3 - R'_{31}t''_2$	$Q_4Q'_1Q''_4$	0
17	$R'_{22}t''_3 - R'_{32}t''_2$	$Q_4Q'_2Q''_4$	0
18	$R''_{11}t'_3 - R''_{31}t'_1$	0	$-Q_4Q'_4Q''_1$
19	$R''_{12}t'_3 - R''_{32}t'_1$	0	$-Q_4Q'_4Q''_2$
20	$R''_{21}t'_3 - R''_{31}t'_2$	$-Q_4Q'_4Q''_1$	0
21	$R''_{22}t'_3 - R''_{32}t'_2$	$-Q_4Q'_4Q''_2$	0
22	$t'_1t''_3 - t'_3t''_1$	0	$Q_4Q'_4Q''_4$
23	$t'_2t''_3 - t'_3t''_2$	$Q_4Q'_4Q''_4$	0

Table 6.1: Trifocal tensors in the generic calibration of completely non-central cameras.

Intersection of axis and calibration grids

Using the technique described above we compute the intersection of the camera axis with the three grids at $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$ respectively (cf. Figure 6.2). This is done by computing the bifocal tensor F for 2 of the image pairs and extracting the intersection points from them. 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.

$$\mathbf{Q} \longleftarrow \mathbf{Q} - \mathbf{a},\ \mathbf{Q}' \longleftarrow \mathbf{Q}' - \mathbf{b},\ \mathbf{Q}'' \longleftarrow \mathbf{Q}'' - \mathbf{c},$$

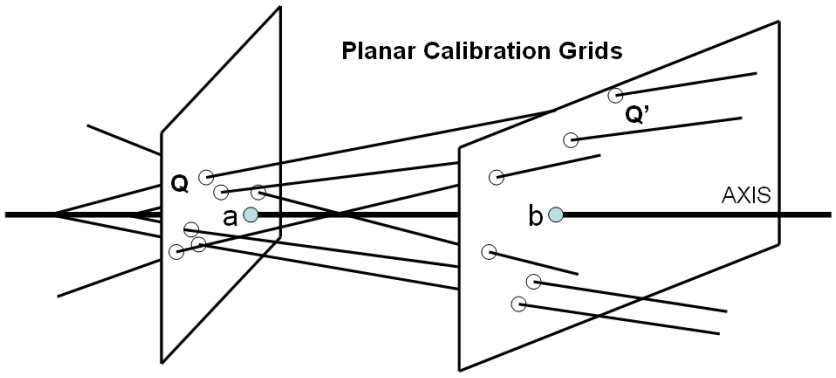


Figure 6.2: Calibration of axial cameras using calibration grids. The projection rays, camera axis and two grids are shown. The axis intersects at $\mathbf{a}$ and $\mathbf{b}$ on the first and second calibration grids respectively.

We can obtain a collinearity constraint by transforming these origins in a single coordinate system (these points all lie on the camera axis, thus are collinear). 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 $\mathbf{O}$, $\mathbf{t}'$ 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 6.1 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 $\sum C_i V_i = 0$ and $\sum C_i W_i = 0$ 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.

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.

$$\begin{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} & \begin{pmatrix} t'_1 \\ t'_2 \\ t'_3 \\ 1 \end{pmatrix} & \begin{pmatrix} \mathsf{R}' & \mathbf{t}' \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \mathbf{Q}' & \begin{pmatrix} \mathsf{R}'' & \Delta\mathbf{t}' \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \mathbf{Q}'' \end{pmatrix}$$

The corresponding constraint is a linear system $\sum \alpha_{ij} Q'_i Q''_j = 0$ (see table 6.2). Note that Δ, Q'_4 and Q''_4 are not present because of the three zeros in the first column. We can solve this linear system to compute the solutions for α_{ij}. We expand the above linear system and do some algebraic manipulation.

$$\alpha_{11} Q'_1 Q''_1 + \alpha_{12} Q'_1 Q''_2 + \alpha_{21} Q'_2 Q''_1 + \alpha_{22} Q'_2 Q''_2 = 0$$

$$Q_4(\alpha_{11}Q'_1Q''_1 + \alpha_{12}Q'_1Q''_2 + \alpha_{21}Q'_2Q''_1 + \alpha_{22}Q'_2Q''_2) = 0$$

$$Q_4Q'_2Q''_2 = -\frac{\alpha_{11}}{\alpha_{22}}Q_4Q'_1Q''_1 - \frac{\alpha_{12}}{\alpha_{22}}Q_4Q'_1Q''_2 - \frac{\alpha_{21}}{\alpha_{22}}Q_4Q'_2Q''_1$$

This will enable us to represent both W_9 and V_{13}, from Table 6.1, in terms of other variables in the tensors V and W respectively.

$$W_9 = -\frac{\alpha_{11}}{\alpha_{22}}W_6 - \frac{\alpha_{12}}{\alpha_{22}}W_7 - \frac{\alpha_{21}}{\alpha_{22}}W_8$$

$$V_{13} = -\frac{\alpha_{11}}{\alpha_{22}}V_{10} - \frac{\alpha_{12}}{\alpha_{22}}V_{11} - \frac{\alpha_{21}}{\alpha_{22}}V_{12}$$

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 6.2: Bifocal tensor from the coplanarity constraint on $\mathbf{O}$, $\mathbf{t}'$, $\mathbf{Q}'$ and $\mathbf{Q}''$.

i	Motion (A_i)	$A1_i$	$A2_i$
1	R'_{31}	$Q_2Q'_1Q''_4$	$Q_1Q'_1Q''_4$
2	R'_{32}	$Q_2Q'_2Q''_4$	$Q_1Q'_2Q''_4$
3	R''_{31}	$-Q_2Q'_4Q''_1$	$-Q_1Q'_4Q''_1$
4	R''_{32}	$-Q_2Q'_4Q''_2$	$-Q_1Q'_4Q''_2$
5	$t'_3 - t''_3$	$Q_2Q'_4Q''_4$	$Q_1Q'_4Q''_4$
6	$C_6 - \frac{\alpha_{11}}{\alpha_{22}}C_9$	0	$Q_4Q'_1Q''_1$
7	$C_7 - \frac{\alpha_{12}}{\alpha_{22}}C_9$	0	$Q_4Q'_1Q''_2$
8	$C_8 - \frac{\alpha_{21}}{\alpha_{22}}C_9$	0	$Q_4Q'_2Q''_1$
9	$C_{10} - \frac{\alpha_{11}}{\alpha_{22}}C_{13}$	$Q_4Q'_1Q''_1$	0
10	$C_{11} - \frac{\alpha_{12}}{\alpha_{22}}C_{13}$	$Q_4Q'_1Q''_2$	0
11	$C_{12} - \frac{\alpha_{21}}{\alpha_{22}}C_{13}$	$Q_4Q'_2Q''_1$	0
12	$\Delta(R'_{11}t'_3 - R'_{31}t'_1)$	0	$Q_4Q'_1Q''_4$
13	$\Delta(R'_{12}t'_3 - R'_{32}t'_1)$	0	$Q_4Q'_2Q''_4$
14	$\Delta(R'_{21}t'_3 - R'_{31}t'_2)$	$Q_4Q'_1Q''_4$	0
15	$\Delta(R'_{22}t'_3 - R'_{32}t'_2)$	$Q_4Q'_2Q''_4$	0
16	$R''_{11}t'_3 - R''_{31}t'_1$	0	$-Q_4Q'_4Q''_1$
17	$R''_{12}t'_3 - R''_{32}t'_1$	0	$-Q_4Q'_4Q''_2$
18	$R''_{21}t'_3 - R''_{31}t'_2$	$-Q_4Q'_4Q''_1$	0
19	$R''_{22}t'_3 - R''_{32}t'_2$	$-Q_4Q'_4Q''_2$	0

Table 6.3: Trifocal tensor for the generic calibration of axial cameras.

Using the above relation we obtain two new constraints given by $\sum A_i A1_i = 0$ and $\sum A_i A2_i = 0$ (cf. table 6.3). Note that each of these constraints is a homogeneous linear system with 12 nonzero coefficients. Both of them have a rank of 11 and thereby produce unique solutions for their coefficients (A_i). Once we compute A_i, we can compute C_i using table 6.3. C_i's are the same coefficients obtained in section 4.1.4. Thus we can use the same technique described in that section to extract the individual pose parameters.

6.4 Summary of the calibration algorithm

1. Take three images of the calibration grid from different viewpoints.

2. For pixels, match calibration points.

3. Compute the axis intersection points $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$ with the calibration grids and transform the grid coordinate systems as given in section 6.3.2.

4. Use the coplanarity constraint to compute the coefficients A_i (using the modified tensors given in section 6.3.2).

5. Compute the coefficients C_i from A_i. Extract the pose variables from C_i.

6. Compute the projection rays using the pose variables.

6.5 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 inconsistency (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 6.3(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 [3] (also see Table 6.4). 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. Table 6.4 gives for various mirror shapes, the conditions for the catadioptric system to be central, axial or non-central. For example, with a spherical mirror, the system is always axial unless in the useless case where the optical center is at the center of the sphere.

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	*o*=center	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*=vertex	o $\in$ MA	o $\notin$ MA	-	OA $\parallel$ MA	OA $\nparallel$ MA
planar	always	-	planar	-	-	-
mir-rot	-	always	-	-	always	-

Table 6.4: 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.

6.6 Experiments

6.6.1 Simulations

We started with perfect axial configurations for three scenarios (as shown in Figures 6.3(a), (b) and (c)) and gradually changed the configurations to make them non-central. We denote this deviation from the perfect axial configuration as disparity. For example, in Figure 6.3(a), the catadioptric system consists of the union of orthographic and perspective cameras and a spherical mirror. The disparity represents the distance between the optical center of a perspective camera and an orthographic camera axis, that passes 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 6.3(b), we have a catadioptric camera consisting of a perspective camera and hyperbolic mirror. Here the disparity represents the distance between the optical center of the perspective camera and the major axis of the hyperbolic mirror. Initially the optical center is at a distance of 5 units from the tip of the hyperbolic mirror, whose minor and major axis lengths are 5 and 10 units.

In Figure 6.3(c), we have a tristereo configuration where the optical centers of three perspective cameras lie on a straight line. Here the disparity represents the perpendicular distance between the optical center of the third camera and the line that joins the optical centers of the first and the second 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 6.3(d). Note that the mean angular error (given in radians) reaches zero only at the precise axial configuration. There is no noise in the simulations.

6.6.2 Stereo camera

We tested our axial calibration algorithm for a stereo system. We captured three images of a calibration grid. The focus is to reconstruct the projection rays for both the cameras in the same generic framework using our axial calibration algorithm. The image of the combined system is formed by concatenating the images from the two cameras. Here the camera axis is the line joining the two optical centers (see Figure 6.4).

Once we compute the poses of the grids we can compute the rays corresponding to individual cameras in the stereo system. The reconstructed rays are shown in Figure 6.4(a). We compute the axis using the constraints given in section 6.3.2. The minimum number of triplets required to compute the axis is 12 according to table 6.3. The algorithm is embedded in RANSAC. Since the axis was reconstructed using one sample, which consists of a set of 12 rays producing the least ray-point error, the axis does not pass through all the rays. The Ray-Point RMS error is 0.04 in percent with respect to the overall size of the scene. We compute the intersection points, or the points closest to the projection rays on the axis. Each of these projection rays is forced to pass through these closest points on the axis and the ray directions are recomputed using the 3D points from the calibration grids. We then clustered the rays into two sets, based on the points on the axis, corresponding to the two pinhole cameras. The centroids of these clusters correspond to the two centers of the stereo systems. Again the rays are forced to pass through their corresponding centers and the ray directions are recomputed. We used an axial bundle adjustment algorithm (see section 8.3.1). The calibration statistics for the stereo system is given in Table 6.5. The final Ray-Point RMS error is 0.07 in percent with respect to the overall size of the scene. Note that the ray-point RMS error increases after fitting the axis. This is due to the overfitting of the noise before forcing the axial constraint.

In order to evaluate the calibration, we compared results with those obtained by plane-based calibration [103, 127], that used the knowledge that the two cameras are pinholes. In both, our axial 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

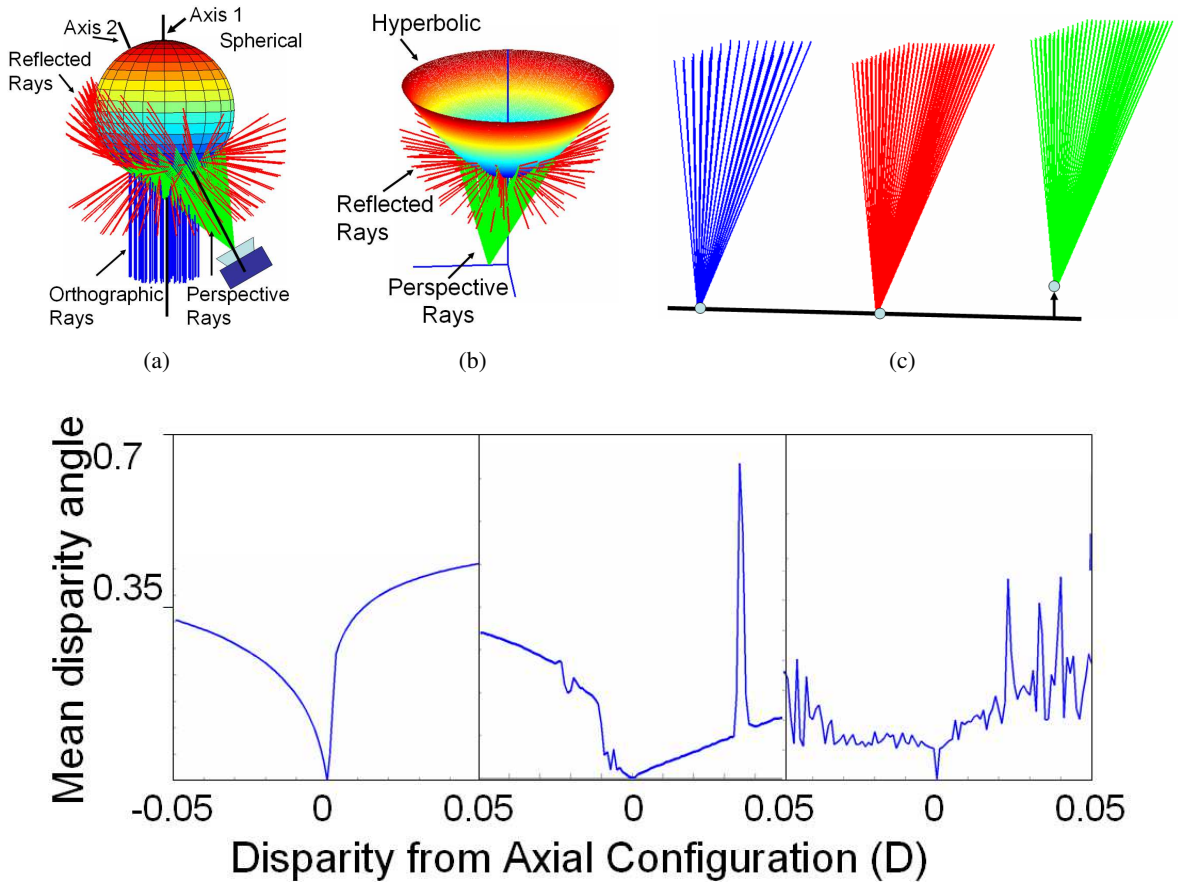
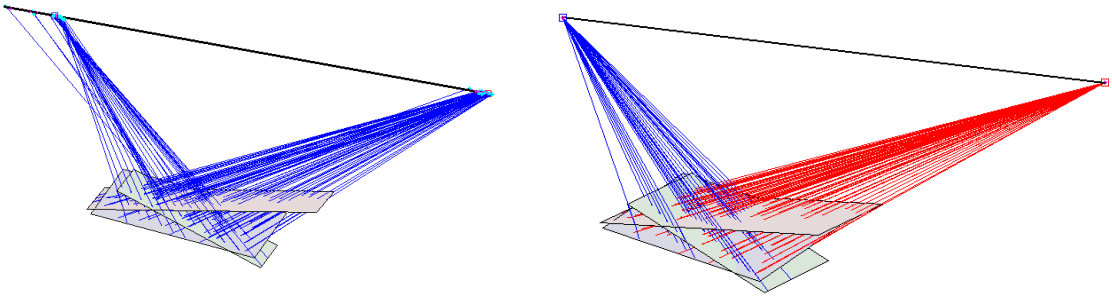


Figure 6.3: Test for axial configuration. (a) Catadioptric (spherical mirror+perspective camera+orthographic camera): becomes non-central when the two optical centers and the sphere center are not collinear (as shown).(b) Catadioptric (Hyperbolic mirror+perspective camera): becomes non-central if the optical center is not on the axis of the hyperbolic mirror (as shown). (c) Tristereo with one of the cameras 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 the left, middle, and right correspond to scenarios in (a), (b) and (c) respectively (see text for more details).

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 6.6). As for translation, we measure the distance between the estimated 3D positions of the grids' centers of gravity (columns t_i in Table 6.6) expressed in percent, relative to the scene size. Here, plane-based calibration is done separately for each camera, leading to the two rows of Table 6.6.



(a) (b)

Figure 6.4: Calibration of a stereo system using axial algorithm. (a) Reconstructed projection rays. Note that the rays corresponding to each of the pinhole cameras do not pass through a single point. This is because the rays are computed using the generic algorithm for axial cameras. (b) Clustered rays of the stereo system.

Images	Rays	Points	RMS before fitting the axis	RMS after fitting the axis
3	481	3x481	0.04	0.07

Table 6.5: Bundle adjustment statistics for the stereo camera. 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).

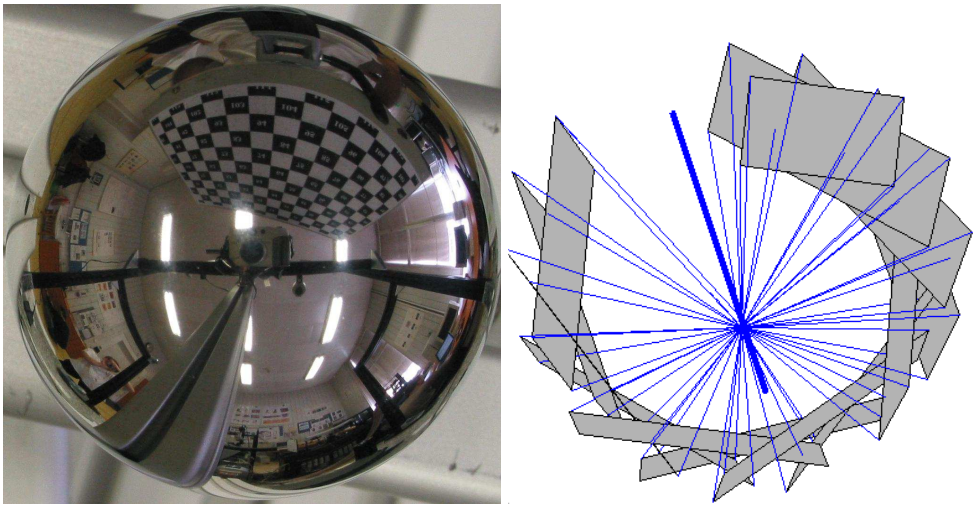
Camera	R_2	R_3	t_2	t_3	Center
1	0.0438	0.0342	0.46	0.75	3.35
2	0.0603	0.0491	0.37	0.61	1.96

Table 6.6: Evaluation of axial stereo calibration relative to the plane based calibration. The comparison is based on the pose estimation of the calibration grids and the estimation of the camera centers. See text for more details.

6.6.3 Spherical catadioptric cameras

We calibrated a spherical catadioptric camera and extracted the camera axis. First we use a central generic calibration for three calibration grids. Once the poses of the grids are estimated we compute the axis using the constraints, provided in section 6.3.2. This enables us to obtain an initial estimate for the axis and the projection rays. Now we force the projection rays to pass through the camera axis. Using this partial calibration, we use pose estimation to incrementally compute the pose of newer grids. This technique of using more than three calibration grids to completely calibrate omnidirectional images will be discussed in chapter 7. The calibration grid captured by a spherical catadioptric camera is shown in Figure 6.5(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 6.5(c). Finally we use an axial bundle adjustment (see section 8.3.1). The

spherical catadioptric model is closer to a central model than an axial one. All the points, obtained from the intersection of the projection rays and the camera axis, are distributed in a very small region on the camera axis. This can be approximated as the center of the camera. As a result of this our axial algorithm is not very robust to spherical catadioptric cameras. This is the reason for using a central approximation for initializing the bundle adjustment. However in other axial catadioptric cameras, where the intersection points are widely distributed, the axial algorithm will be more robust. We study the reasons for the instability in calibration algorithm in section 7.3.



(a) (b)

Figure 6.5: Calibration of a spherical catadioptric camera. (a) Image of a calibration grid. (b) Estimated poses of several grids along with the camera axis.

6.7 Conclusion

We studied the theory and proposed a non-iterative linear calibration algorithm for an intermediate class of cameras called axial cameras. Further investigation needs to be carried out to test the accuracy of this approach with respect to parametric and completely non-central approaches. We have considered only calibration algorithms using three calibration grids. Nevertheless three calibration grids are seldom sufficient to calibrate the complete field of view of omnidirectional cameras. In the next chapter we study the extension of the generic calibration algorithm for more than three calibration grids and more importantly the bundle adjustment formulations.

Chapter 7

Complete Generic Calibration Using Multiple Images

In this chapter we consider the generic calibration problem using multiple images. We propose two different approaches for this problem. The first approach can handle multiple calibration grids simultaneously. This allows to calibrate the image region where the projections of the calibration grids overlap. To calibrate the complete image with this method, 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. We thus also propose methods for concatenating the information contained in multiple images in case there is no global overlap (see examples in figure 7.1).

The overall proposed calibration approach works as follows. An initial calibration is performed from images of calibration grids that present a sufficient overlap, i.e. an as large region as possible where all considered grids overlap. Now we consider our second approach where we incorporate one image after the other, each time the image having 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 process is iterated until all images have been used. We also propose a bundle adjustment algorithm for the general imaging model; this can be used at any stage of the procedure, e.g. after computing the pose of a new calibration grid or of course at the end of the whole procedure.

This procedure and the underlying algorithms are developed for both, non-central and central models, although the central case is described in more detail in this chapter. Besides developing algorithms, we are also interested in the question if for certain cameras it is worth going to a full non-central model or not, cf. also [2, 63].

This chapter is organized as follows. The calibration approach is described in section 7.1 and non-central variant is proposed in section 7.2. Experimental results on a variety of cameras are presented in section 7.5, followed by conclusions in section 7.6.

7.1 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

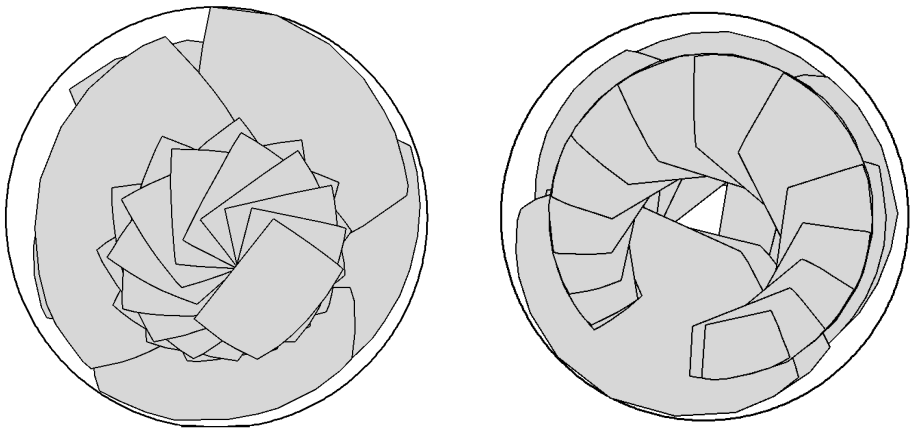


Figure 7.1: Examples of complete calibration. Left: 23 overlapping calibration grids, used in calibrating a fisheye. Right: 24 overlapping calibration grids used in calibrating a spherical catadioptric camera.

points.

For a non-central camera, 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. Naturally, a single point per pixel is sufficient at this stage.

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 section 7.2. First, we show how to use the images of multiple grids simultaneously, to compute grid pose and the optical center. As mentioned above, this only allows to use matches for pixels that lie inside the projections of all grids considered. It is then shown how to compute the pose of additional grids; this serves to iteratively calibrate the whole image region, using all grids, even if their projections have no common overlap region. Refinement of calibration after each step, through bundle adjustment, is then discussed in section 8.3.1.

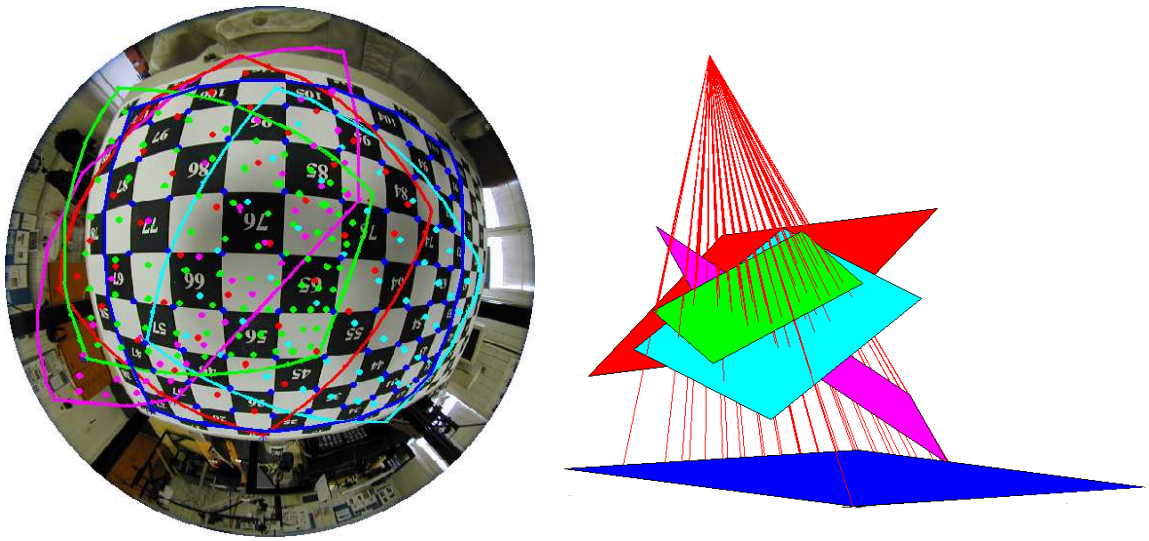
7.1.1 Calibration using multiple grids

As mentioned above, our motivation here is to devise a method for taking into account multiple calibration grids at the same time, to obtain a good initial calibration for a sub-region of the image. During this phase, we obtain the poses, that is the orientation and position of these grids w.r.t. a common coordinate system. The common coordinate system is usually the coordinate system of a reference calibration image. The reference image is usually the image where the grid occupies the largest portion of the image, or the one which has an overlap with a maximum number of other images. Let B_i denote the image region associated with the i_{th} calibration grid, for $i = 1, \cdots, n$. Let $i = 1$ represent the reference calibration image. We first calibrate the region $\cup_{i=2,\cdots,n}(B_1 \cap \cup_{i=2}^{n} B_i)$, that is the union of all pairs of intersecting grid regions formed with the reference grid and all others that have a sufficient overlap with the reference one.

Further below, we then show how to extend this calibration to the whole region $\cup_{i=0}^{n} B_i$, that is, the union of all grids.

We now outline the theory behind calibration using multiple grids. Consider one pixel and its associated grid points, with homogeneous coordinates $\mathbf{Q}^k = (Q_1^k, Q_2^k, Q_3^k, Q_4^k)^T$, for grids $k = 1, \ldots, m$. In the following, we consider planar calibration grids, and thus suppose that $Q_3^k = 0$. The use of non-planar calibration grids is actually simpler algebraically, but is less practical, so we neglect it in this chapter.

Let the unknown grid poses be represented by rotation matrices R^k and translation vectors $\mathbf{t}^k$, such that



(a) (b)

Figure 7.2: a) An omnidirectional image taken with a fisheye lens and the region of calibration grids occupied in 4 other images (shown using convex hulls of grid points). b) We show the 5 calibrated grid positions, which are used to compute the projection rays.

the point $\mathbf{Q}^k$, given in local grid coordinates, is mapped to global coordinates via

$$\begin{pmatrix} \mathsf{R}^k & \mathbf{t}^k \\ \mathbf{0}^T & 1 \end{pmatrix} \begin{pmatrix} Q_1^k \\ Q_2^k \\ 0 \\ Q_4^k \end{pmatrix} \tag{7.1}$$

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. $\mathsf{R}^1 = \mathsf{I}$ and $\mathbf{t}^1 = 0$.

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 (7.1), 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:

$$\begin{pmatrix} O_1 & Q_1^1 & R_{11}^2 Q_1^2 + R_{12}^2 Q_2^2 + t_1^2 Q_4^2 & \cdots \\ O_2 & Q_2^1 & R_{21}^2 Q_1^2 + R_{22}^2 Q_2^2 + t_2^2 Q_4^2 & \cdots \\ O_3 & 0 & R_{31}^2 Q_1^2 + R_{32}^2 Q_2^2 + t_3^2 Q_4^2 & \cdots \\ 1 & Q_4^1 & Q_4^2 & \cdots \end{pmatrix}$$

The collinearity of these points implies that this $4 \times (m+1)$ matrix must be of rank smaller than 3. This is an extension of Equation (5.1).

Consequently, the determinants of all its 3×3 submatrices must vanish.

The vanishing determinants of 3×3 submatrices give 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 $\mathbf{Q}^j$ and $\mathbf{Q}^k$ to form a submatrix, we get bilinear equations in terms of calibration point coordinates $\mathbf{Q}^j$ and $\mathbf{Q}^k$. Hence, we may write the equations in the form:

$$(Q^j)^T \mathsf{M}^{jk} \mathbf{Q}^k = 0 \tag{7.2}$$

This matrix M^{jk} (a bifocal matching tensor), depends on grid pose and optical center, in a way specific to which 3×3 submatrix is underlying the equation. Using the equation 7.2, we try to estimate such tensors from available correspondences. Since a 3×3 submatrix can be obtained by removing one row and $m-2$ columns at a time, we have $4\binom{m+1}{3}$ possible matching tensors M. However, using simulations we observed that not all of these give unique solutions, i.e. not all these tensors 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 $2(m-1)$ constraints of the type $T_{x34;12y}, (x=1,2), (y=3,\cdots,m)$ for calibration, i.e. constraints combining the optical center and the first grid, with the other grids. For these tensors, the constraint equation (7.2) takes the following form: $\sum_{i=1}^{i=9} C_i^y V_i^y = 0$ and $\sum_{i=1}^{i=9} C_i^y W_i^y = 0$ 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 7.1.

i	C_i^y	V_i^y	W_i^y
1	$Q_1Q_1^y$	0	$R_{3,1}^y$
2	$Q_1Q_2^y$	0	$R_{3,2}^y$
3	$Q_1Q_4^y$	0	$-O_3 + t_3^y$
4	$Q_2Q_1^y$	$R_{3,1}^y$	0
5	$Q_2Q_2^y$	$R_{3,2}^y$	0
6	$Q_2Q_4^y$	$-O_3 + t_3^y$	0
7	$Q_4Q_1^y$	$-O_2R_{3,1}^y + O_3R_{2,1}^y$	$-O_1R_{3,1}^y + O_3R_{1,1}^y$
8	$Q_4Q_2^y$	$-O_2R_{3,2}^y + O_3R_{2,2}^y$	$-O_1R_{3,2}^y + O_3R_{1,2}^y$
9	$Q_4Q_4^y$	$-O_2t_3^y + O_3t_2^y$	$-O_1t_3^y + O_3t_1^y$

Table 7.1: Coupled variables in tensors $T_{134;12y}$ and $T_{234;12y}$ for a central camera.

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 $(m-1)$ constraints by choosing $y = 3, \cdots, m$. We now combine all the coupled variables contained in the different tensors, to obtain the following system which links the motion variables of all the grids.

$$\begin{bmatrix} \mathsf{H}_{6\times2}^2 & \mathsf{J}_{6\times6} & \cdots & \mathsf{0}_{6\times6} \\ \vdots & \vdots & \ddots & \vdots \\ \mathsf{H}_{6\times2}^n & \mathsf{0}_{6\times6} & \cdots & \mathsf{J}_{6\times6} \end{bmatrix} \begin{bmatrix} -O_1 \\ -O_2 \\ \mathbf{X}_{6\times1}^2 \\ \vdots \\ \mathbf{X}_{6\times1}^m \end{bmatrix} = \begin{bmatrix} \mathbf{Y}_{6\times1}^2 \\ \vdots \\ \mathbf{Y}_{6\times1}^m \end{bmatrix}, \tag{7.3}$$

$$\mathsf{H}^y = \begin{bmatrix} 0 & V_4^y \\ 0 & V_5^y \\ 0 & V_6^y \\ V_4^y & 0 \\ V_5^y & 0 \\ V_6^y & 0 \end{bmatrix}, \mathsf{J} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{X}^i = \begin{bmatrix} \lambda_i O_3 R_{2,1}^i \\ \lambda_i O_3 R_{2,2}^i \\ \lambda_i O_3 (t_2^i - O_2) \\ \lambda_i O_3 R_{1,1}^i \\ \lambda_i O_3 R_{1,2}^i \\ \lambda_i O_3 (t_1^i - O_1) \end{bmatrix}, \mathbf{Y}^i = \begin{bmatrix} V_7^i \\ V_8^i \\ V_9^i \\ W_7^i \\ W_8^i \\ W_9^i \end{bmatrix}$$

We rewrite Equation (7.3) as follows:

$$\mathsf{A}_{6(m-1)\times(2+6(m-1))}\mathsf{Z}_{2+6(m-1)} = \mathbf{Y}_{6(m-1)}$$

A is of rank $(6-1)$ since the right hand side of Equation(7.3) is full rank. Since A is of rank $6(m-1)$, we obtain the Z as a linear combination of three vectors expressed with scalars l_1 and l_2. We use orthogonality constraints on rotation matrices to obtain a unique solutions for Z. Specifically, we use three quadratic equations for each grid:

$$\begin{aligned} R^y_{11}R^y_{12} + R^y_{21}R^y_{22} + R^y_{31}R^y_{32} &= 0 \\ (R^y_{11})^2 + (R^y_{21})^2 + (R^y_{31})^2 &= 1 \\ (R^y_{12})^2 + (R^y_{22})^2 + (R^y_{32})^2 &= 1 \end{aligned}$$

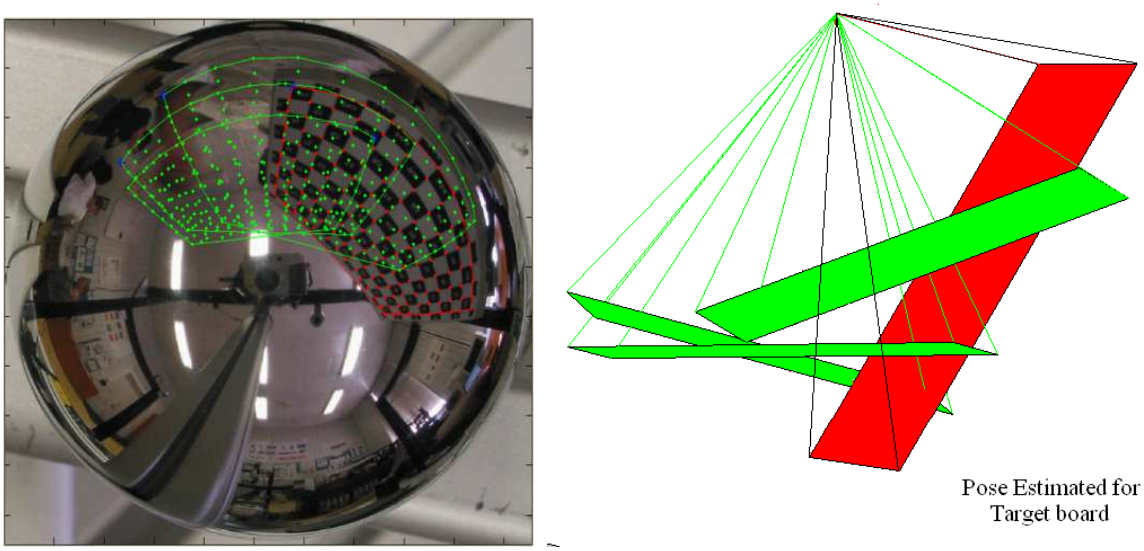
On substituting the expressions for the rotational components and solving linear systems with l_1,l_2,O,λ_y and other coupled variables, we finally obtain the solutions for l_1 and l_2. Using the formulas for Z given above, it is possible to compute the pose variables uniquely except for a sign ambiguity in n variables. This arises due to an ambiguity in the position of each of these grids leading to 2^n possible solutions. Each grid 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 one side of the center. However this constraint becomes difficult to apply for omnidirectional cameras where the grids essentially get distributed around the center. We thus apply the following procedure. The ambiguity on the grid poses can be solved without that information: for one grid, the 2-fold ambiguity is fixed arbitrarily. Then, for any additional grid, the ambiguity can be fixed as follows: consider the overlap with grids with already fixed pose. Choose among the two possible poses the one for which grid points associated with the overlap region, are closer to the points in the already fixed grids. For the central model it's even easier: choose the pose such that grid points are on the same side of the center as the corresponding points (associated with the same pixel) of already fixed grids. 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. In Figure 7.2 we show the calibration using five grids simultaneously. Figures 7.2(a) and (b) show the convex hulls of the calibration grid points and the estimated projection rays respectively.

7.1.2 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. As was mentioned, this in general only allows to calibrate a sub-region of the image (unless some grid covers the whole image). Furthermore, in the previous step, only grids were used whose projection in the image had some overlap with one of the grids ("the first grid"). In order to make the calibration complete, we now show how to include additional grids, which do not have any overlap with the first grid, but with some of the others.

Let C_k denote the region obtained by the simultaneous calibration with k grids. We now estimate the pose of the $(k+1)_{th}$ grid by taking into account the grid points falling within the region $C_k \cap B_{k+1}$. Using this estimated pose, the calibration region is then extended to $C_{k+1} = C_k \cup B_{k+1} = \cup_{i=1}^{k+1} B_i$. By estimating one grid at a time, we can iteratively calibrate the whole region occupied by the union of all grids. Figure 7.1 shows examples of the such calibrated region, for a fish-eye and a spherical catadioptric system.

Please refer to the details of the pose estimation given in section 8.2.3.



(a) (b)

Figure 7.3: Pose Estimation of a new grid using existing calibration. a) We show the new grid, whose pose has to be estimated, and the existing calibration region (using convex hulls of the grid points). b) We show the poses of 3 previously calibrated grids and the estimated pose of the new grid.

7.2 Non-central cameras

In the non-central case with planar grids, 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 section 7.1.1. For a non-central camera, we first calibrate the region $\cup_{i=3,\cdots,n}(B_1 \cap B_2 \cap B_i)$, i.e. the union of intersections of every possible triplet formed with the two reference grids and the others. This is done, as above, using all these grids simultaneously. The calibration is then extended to the whole region $\cup_{\{i,j\}}(B_i \cap B_j)$, that is the union of all pairwise intersections of grid regions (this is because the computation of projection rays requires at least 2 grid points per pixel).

We now summarize the calibration procedure, analogously to section 7.1.1. We have no optical center here, so do consider the following $4 \times n$ matrix of collinear points:

$$\begin{pmatrix} Q_1^1 & R_{11}^2 Q_1^2 + R_{12}^2 Q_2^2 + t_1^2 Q_4^2 & \cdots \\ Q_2^1 & R_{21}^2 Q_1^2 + R_{22}^2 Q_2^2 + t_2^2 Q_4^2 & \cdots \\ 0 & R_{31}^2 Q_1^2 + R_{32}^2 Q_2^2 + t_3^2 Q_4^2 & \cdots \\ Q_4^1 & Q_4^2 & \cdots \end{pmatrix}$$

Similarly to the central case we can apply the collinearity constraint by equating the determinant of every 3×3 submatrix to zero. We take the first three columns and construct three tensors as given in Table 7.2.

We have already explained this algorithm in section 4.1.4. We slightly modify this algorithm by using three tensors instead of two tensors. Then we stack the linear system with 23 coupled variables as given in Table 7.2. We observed that by stacking the constraints from all the three tensors together, we directly obtain the coupled variables from both the tensors in the same scale. This modification improved the numerical stability of the algorithm. The estimation of the motion vectors from the coupled variables is already explained in section 4.1.4.

Now let us look at the extension of this work to multiple views. In contrast to the central case, where

	Coupled motion pars	T^1	T^2	T^4
1	R'_{31}	$Q_2Q'_1Q''_4$	$Q_1Q'_1Q''_4$	0
2	R'_{32}	$Q_2Q'_2Q''_4$	$Q_1Q'_2Q''_4$	0
3	R''_{31}	$-Q_2Q'_4Q''_1$	$-Q_1Q'_4Q''_1$	0
4	R''_{32}	$-Q_2Q'_4Q''_2$	$-Q_1Q'_4Q''_2$	0
5	$t'_3 - t''_3$	$Q_2Q'_4Q''_4$	$Q_1Q'_4Q''_4$	0
6	$R'_{11}R''_{31} - R'_{31}R''_{11}$	0	$Q_4Q'_1Q''_1$	$-Q_2Q'_1Q''_1$
7	$R'_{11}R''_{32} - R'_{31}R''_{12}$	0	$Q_4Q'_1Q''_2$	$-Q_2Q'_1Q''_2$
8	$R'_{12}R''_{31} - R'_{32}R''_{11}$	0	$Q_4Q'_2Q''_1$	$-Q_2Q'_2Q''_1$
9	$R'_{12}R''_{32} - R'_{32}R''_{12}$	0	$Q_4Q'_2Q''_2$	$-Q_2Q'_2Q''_2$
10	$R'_{21}R''_{31} - R'_{31}R''_{21}$	$Q_4Q'_1Q''_1$	0	$Q_1Q'_1Q''_1$
11	$R'_{21}R''_{32} - R'_{31}R''_{22}$	$Q_4Q'_1Q''_2$	0	$Q_1Q'_1Q''_2$
12	$R'_{22}R''_{31} - R'_{32}R''_{21}$	$Q_4Q'_2Q''_1$	0	$Q_1Q'_2Q''_1$
13	$R'_{22}R''_{32} - R'_{32}R''_{22}$	$Q_4Q'_2Q''_2$	0	$Q_1Q'_2Q''_2$
14	$R'_{11}t''_3 - R'_{31}t''_1$	0	$Q_4Q'_1Q''_4$	$-Q_2Q'_1Q''_4$
15	$R'_{12}t''_3 - R'_{32}t''_1$	0	$Q_4Q'_2Q''_4$	$-Q_2Q'_2Q''_4$
16	$R'_{21}t''_3 - R'_{31}t''_2$	$Q_4Q'_1Q''_4$	0	$Q_1Q'_1Q''_4$
17	$R'_{22}t''_3 - R'_{32}t''_2$	$Q_4Q'_2Q''_4$	0	$Q_1Q'_2Q''_4$
18	$R''_{11}t'_3 - R''_{31}t'_1$	0	$-Q_4Q'_4Q''_1$	$Q_2Q'_4Q''_1$
19	$R''_{12}t'_3 - R''_{32}t'_1$	0	$-Q_4Q'_4Q''_2$	$Q_2Q'_4Q''_2$
20	$R''_{21}t'_3 - R''_{31}t'_2$	$-Q_4Q'_4Q''_1$	0	$-Q_1Q'_4Q''_1$
21	$R''_{22}t'_3 - R''_{32}t'_2$	$-Q_4Q'_4Q''_2$	0	$-Q_1Q'_4Q''_2$
22	$t'_1t''_3 - t'_3t''_1$	0	$Q_4Q'_4Q''_4$	$-Q_2Q'_4Q''_4$
23	$t'_2t''_3 - t'_3t''_2$	$Q_4Q'_4Q''_4$	0	$Q_1Q'_4Q''_4$

Table 7.2: Coupled coefficients of tensors T^1, T^2 and T^4.

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 $4 \times (n-2)$ possible tensors, represented by $T_{ijk;12y}, (\{i,j,k\} \in \{1,2,3,4\}), (i \neq j \neq k), (y = 3, \cdots, n)$. Using simulations we found, as in the central case, that not all of these tensors provide unique solutions. The computation of the pose paramters is done using the method given in section 7.1.1 and the projection rays are computed.

The next step of the calibration chain, pose estimation and computation of further projection rays, is also slightly different compared to central cameras. Here, the calibration region is extended to $C_{k+1} = \cup_{\{i,j\}}(B_i \cap B_j)$, i.e. it contains all pixels that are matched to at least 2 grid points. The pose estimation algortithm is given in section 8.2.3.

7.3 Stability of calibration algorithms

The stability or not of calibration depends on various factors, as explained in the following. When calibrating a perfectly central camera using the non-central or axial model, the tensors involved can not be computed uniquely; which tensors are picked among the ambiguous solutions is the result of a random process. One always chooses the solution corresponding to the smallest eigenvalue/singular value of the underlying equation system; with noisefree data, several eigenvalues are zero and picking any of the corresponding eigenvectors has zero probability of giving the true tensor. This implies that extraction of pose parameters and finally of projection rays, gives an incorrect result. In the presence of noise in the data (point correspondences), one picks the eigenvector corresponding to the single smallest eigenvalue. This however is completely the result of the noise, with the same result as in the noise-free case. So, with a perfectly central camera, non-central and axial models give a bad calibration. This suggests that for cameras close to being central, the calibration using a non-central or axial model, can be expected to be unstable. If it is really unstable or not depends on all the following factors:

- how close the camera is to being central (for perfectly central, calibration will always fail)
- how large is the noise in the data (with zero noise and even very slightly non-central camera, the calibration will be perfect)
- how many data are used: even in the presence of noise and even with an only slightly non-central camera, using many images (going towards infinitely many) may give a perfect calibration.

Stability depends on the combination of these factors (and furthermore on the algorithm and underlying cost function used: algorithms using linear equations only are bound to give less stable results than the bundle adjustment; of course, bundle adjustment needs an initialization).

7.4 Slightly non-central cameras

In order to obtain a stable solution from our non-central calibration algorithm, in continuation to our discussion in the earlier section, the camera must be sufficiently non-central. However, especially for omnidirectional cameras, the initial calibration step only allows to calibrate an image sub-region. The rays associated with pixels in that region, are usually not sufficiently non-central (with the exception of e.g. a multi-camera setup, which is highly non-central). As a result the calibration is very unstable.

For slightly non-central cameras like fisheye, spherical or hyperbolic catadioptric cameras, we thus start by running the central version of the initial calibration method. Typically we use four to five images simultaneously to calibrate an image region and then use pose estimation to add other images and cover the rest of the image. Then, 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.

7.4.1 Selecting the best camera model

We observed that most cameras can be calibrated using our generic calibration algorithms (central or non-central), as shown in Table 7.3. Several issues have to be discussed though. First, the non-central camera model encompasses the central one of course. However, the non-central calibration algorithm of section 7.2, 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.

Besides considering these general camera types, we may also discuss the choice between parametric and non-parametric models for a given camera. Parametric models enable faster, simpler and compact algorithms for many applications like perspective image synthesis. In any case we see a potential advantage with our non-parametric generic calibration algorithm. It 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. Generic calibration would already give a good initialization for the poses of different grids. By fitting the parametric camera model to matches between image points and 3D rays, we may get simpler expressions.

7.5 Experiments and Results

We have calibrated a wide variety of cameras (both central and non-central) as shown in Table 7.3. Results of distortion correction from fisheye images are shown in Figures 7.4 to 7.8. The correction is mostly good as is seen more clearly in Figure 7.5. In Figure 7.4(d) however, one can see some distortion remaining at the image border. This can possibly be explained by the fact that this region was covered by a single grid, cf. Figure 7.2(a). Another explanation would be that the fisheye is in reality not perfectly central, which would of course show up most strongly at the image border.

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 7.9), we applied both the central calibration and the procedure explained in section 7.4, going from central to non-central. Table 7.3 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. We show the complete distortion correction for fisheye images in Figures 7.6 to 7.8.

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. Using the calibration result, the pose of those grids is computed, using the method given in section 8.2.3. 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. The recovered grid poses are shown in Figures 7.10(a) and (b). Figure 7.10(c) shows the extension of a line in the grid's coordinate system, for the different poses. Due to the

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 7.3: 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 7.4: RMS error for circle fits to grid points, for turntable sequences (see text).

turntable motion, these lines should envelope a quadric close to a cone, which indeed is the case. The lines trace out a cone only if they cut the rotation axis of the turntable. Otherwise, they will trace out a ruled quadric, e.g. a hyperboloid. A complete quantitative analysis is difficult, but we evaluated how close the trajectories of individual grid points are to being circular (as they should be, due to the turntable motion). The least-squares circle fit for one of the grid points, from its 14 recovered positions, is shown in Figure 7.10(d). Table 7.4 shows the RMS errors of circle fits (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.

7.6 Conclusions

This chapter extends the methods of chapters 4 and 5 for the minimum number of images, to arbitrary numbers of images; thus enables calibration of whole image. 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. The optimization of complex parametric models may require a good initialization; this might be provided by the generic calibration result.

(a)

(b)

(c)

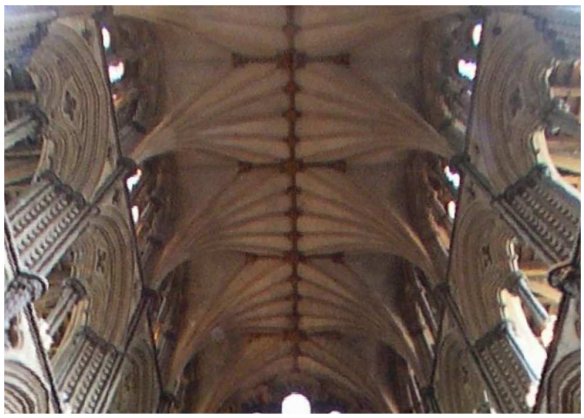
(d)

(e)

Figure 7.4: Complete distortion correction of a fisheye image of Pantheon in Paris. (a), (b), (c), (d) and (e) show the perspective synthesis of various regions of the original fisheye image (shown in Figure 5.3(a)). Since the field of view of the fisheye camera is $360^\circ \times 180^\circ$, we cannot show the perspective synthesis of the whole image at once. The distortion correction is done using the technique described in Figure 5.3.

Figure 7.5: Complete distortion correction of a fisheye image of Notre Dame in Grenoble. Note that a heavily distorted line (shown near the boundary of the image) is corrected to a perfect straight line.

(a)



(b)

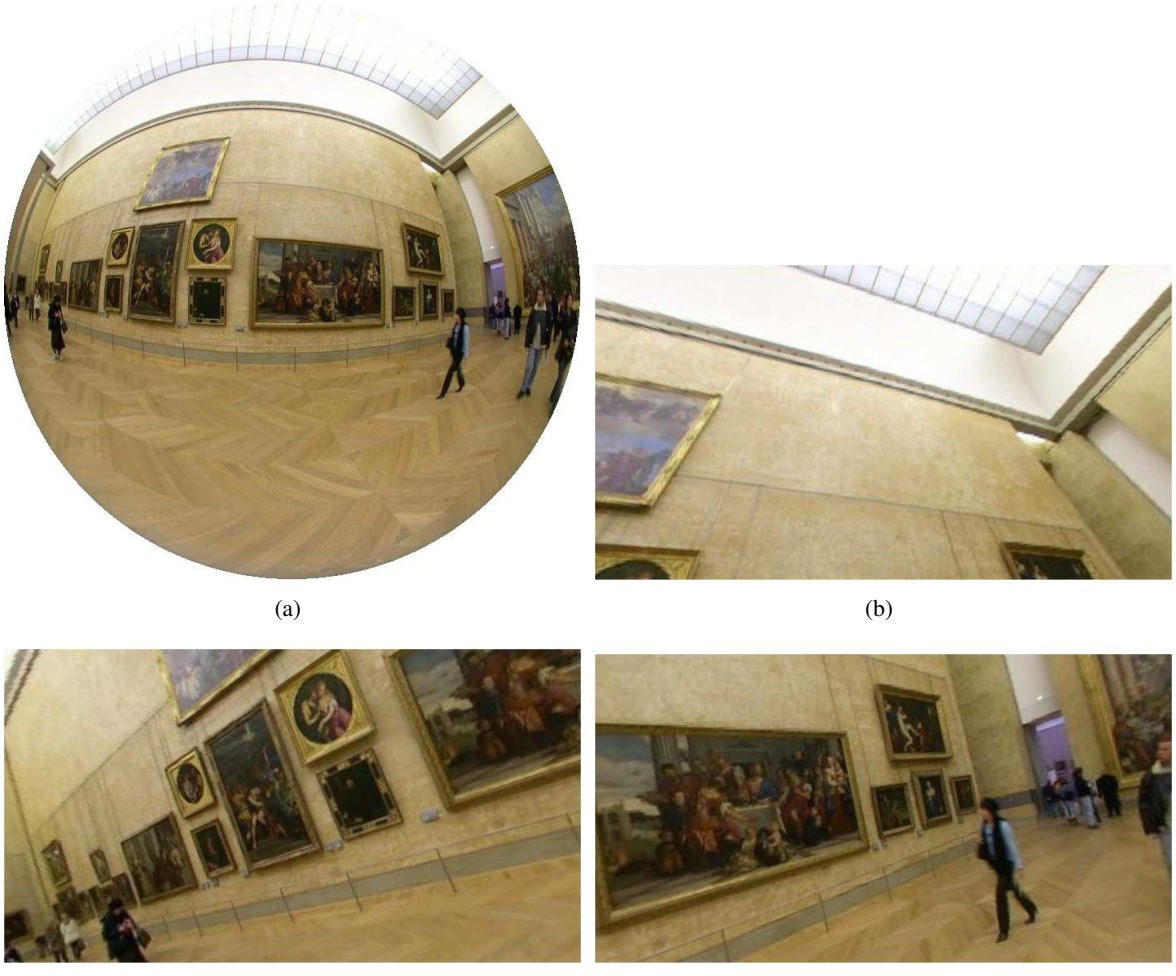
(c)

(d)

(e)

(f)

Figure 7.6: Complete distortion correction of a fisheye image of a Cathedral in Ely near Cambridge, UK.



(a) (b)

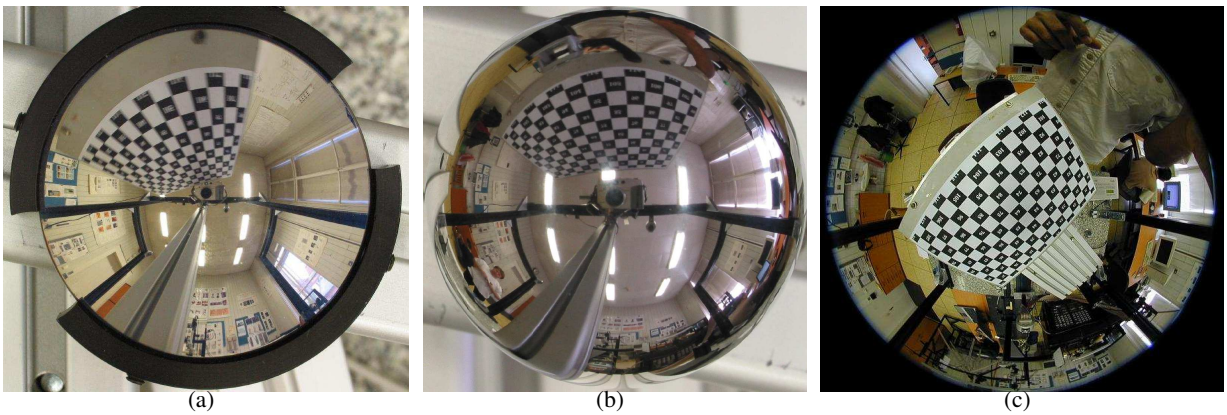
(c) (d)

Figure 7.7: Complete distortion correction of a fisheye image of the Louvre museum in Paris. In (b), the lines on the ceiling are nicely corrected.

(a) (b)

(c) (d)

Figure 7.8: Complete distortion correction of a fisheye image of the Graz Technical University in Austria.



(a) (b) (c)

Figure 7.9: sample images for a) hyperbolic catadioptric camera b) spherical catadioptric camera and c) fisheye

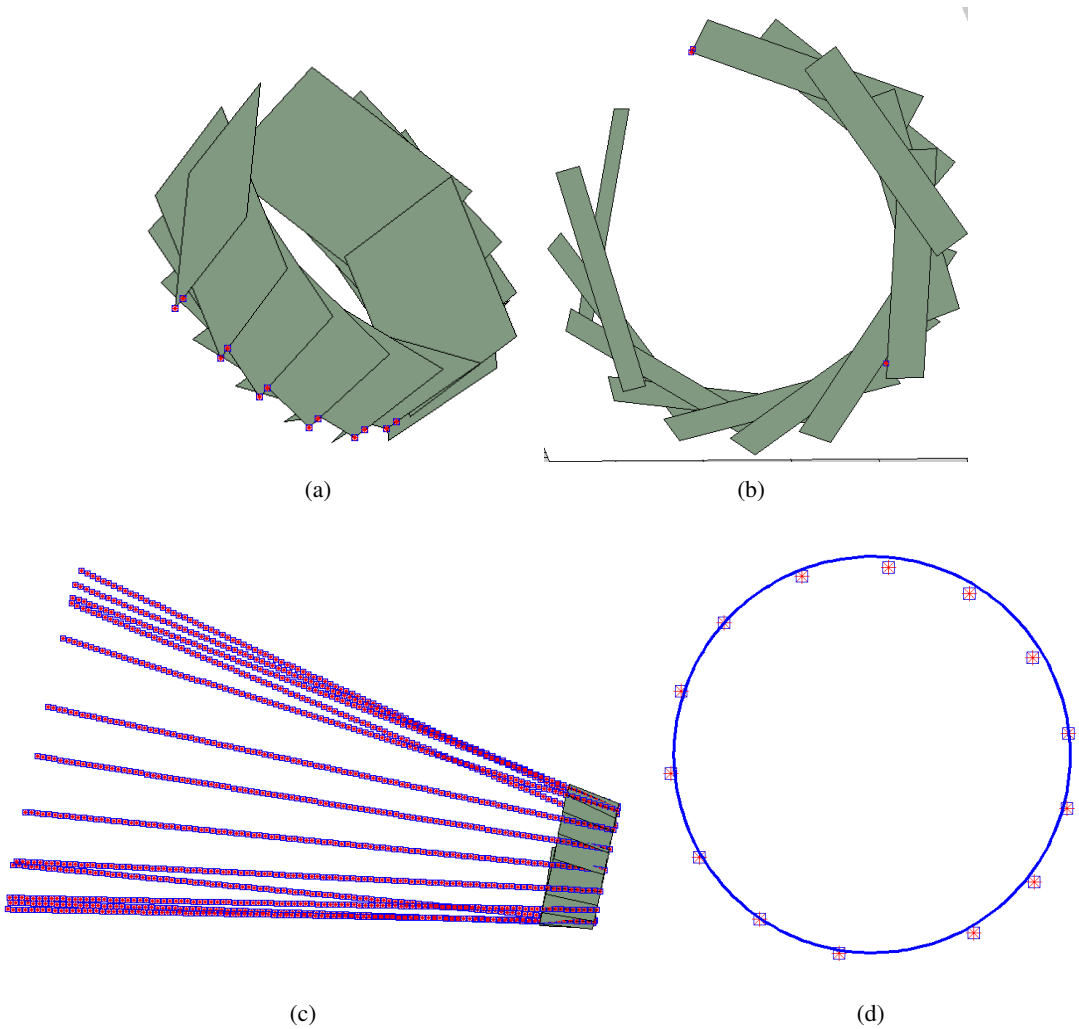


Figure 7.10: Experiment on pose estimation. (a) and (b) show the estimated poses of a calibration grid in 14 positions. (c) Extensions of a line on the calibration grid, in all 14 positions. (d) Least squares circle fit to the estimated positions of one grid point.

Chapter 8

Generic Structure-from-Motion Framework

We introduce a structure-from-motion approach for the general imaging model, that allows to reconstruct scenes from calibrated images, possibly taken by cameras of different types (cross-camera scenarios). Structure-from-motion is naturally handled via camera independent ray intersection problems, solved via linear or simple polynomial equations. We also propose two approaches for obtaining optimal solutions using bundle adjustment, where camera motion, calibration and 3D point coordinates are refined simultaneously. The proposed methods are evaluated via experiments on two cross-camera scenarios – a pinhole used together with an omni-directional camera and a stereo system used with an omni-directional camera.

8.1 Introduction and Motivation

Many different types of cameras including pinhole, stereo, catadioptric, omni-directional and non-central cameras have been used in computer vision. In Chapter 1 we have observed that some of these, especially the omni-directional class, provide more stable ego-motion estimation and larger fields of view than pinhole cameras [4, 73, 80]. Naturally, a larger field of view allows to reconstruct 3D scenes using fewer images, although the spatial resolution is lower, i.e. pinhole cameras can provide more useful texture maps. Non-central cameras, a review of which is given in [5], eliminate the scale ambiguity in motion estimation and thereby we do not need ground control points for scale computation. Thus using a variety of cameras will facilitate and enhance the 3D reconstruction in both geometry and texture. For example, we can build a surveillance system with one static omni-directional camera (which detects moving objects) and several narrow-field-of-view pan-tilt-zoom cameras that can be used to take close-up pictures of objects. Also while reconstructing complete environments, it is helpful to have a combination of omni-directional and traditional images: the traditional ones (narrow field-of-view, i.e. high spatial resolution) give good accuracy locally, whereas the omni-directional images would be good for registering images scattered throughout the environment to a single reference frame. Despite these advantages, a general, unified, structure-from-motion approach for handling different camera systems, does not exist yet.

In this chapter, we formulate the most basic structure-from-motion problems in a unified, camera independent manner, typically as ray intersection type problems. This is shown for pose and motion estimation and triangulation, in Sections 8.2.1 to 8.2.3.

The main contribution of this chapter is the description of an approach for 3D scene reconstruction from images acquired by any camera or system of cameras following the general imaging model. Its building blocks are motion estimation, triangulation and bundle adjustment algorithms, which are all basically formulated as ray intersection problems. Classical motion estimation (for pinhole cameras) and its algebraic centerpiece, the essential matrix [58], are generalized in Section 8.2.1, following [81]. As for triangulation, various algorithms have been proposed for pinhole cameras in [48]. In this work, we use the *mid-point*

approach because of its simplicity, see Section 8.2.2. Initial estimates of motion and structure estimates, obtained using these algorithms, are refined using bundle adjustment [49, 117], i.e. (non-linear in general) optimization of all unknowns. This is described in Section 8.3.

Bundle adjustment needs a good initial solution, and also depending on the cost functions the convergence rate and the optimality of the final solutions vary [48, 117]. In this work we utilize two different cost functions to design and implement two different bundle adjustment algorithms. The first cost function is based on minimizing the distance between 3D points and associated projection rays, which we refer to as the *ray-point* method. The main reason for using this cost function is that it was straightforward to use for the general camera model. The second cost function is, as usually desired, based on the re-projection error, i.e. the distance between reprojected 3D points and originally measured image points (possibly weighted using uncertainty measures on extracted image point coordinates). The main reason for using this cost function is its statistical foundation [48], and the fact that it leads to a least-squares type cost function, for which efficient optimization methods exist, such as Gauss-Newton or Levenberg-Marquardt. There is a major challenge in applying this cost function to the general imaging model used here, due to the fact that we have no analytical projection equation, and thus no analytical expression for the re-projection error based cost function and its derivatives. In order to address this challenge, we approximate the n rays of a given camera, central or non-central, by k clusters of central rays, i.e. rays that intersect in a single point. For example we have $k = 1$ for central cameras (e.g. pinhole), $k = 2$ for a stereo system, $k = n$ for oblique cameras [78], etc. Each such cluster of rays, therefore, corresponds to a single central camera. Given any 3D point we find the corresponding cluster of rays to which it belongs. The rays in every cluster are intersected by a plane to synthesize a perspective image. This allows us to formulate an analytical function that maps the 3D point to a 2D pixel on the synthesized image, and thus to drive bundle adjustment. Details are discussed in Section 8.3.2.

Experimental results with two cross-camera scenarios are given in Section 8.4: we have applied the structure-from-motion algorithm to two cross-camera scenarios – a pinhole camera used together with an omni-directional camera, and a stereo system (interpreted as a single non-central camera) used together with an omni-directional camera. We compare the performances with ground truth where available, and 3D reconstruction from pinhole images, obtained using classical techniques.

8.2 Generic Structure-from-Motion

Figure 8.1 describes the pipeline for the proposed generic structure-from-motion approach.

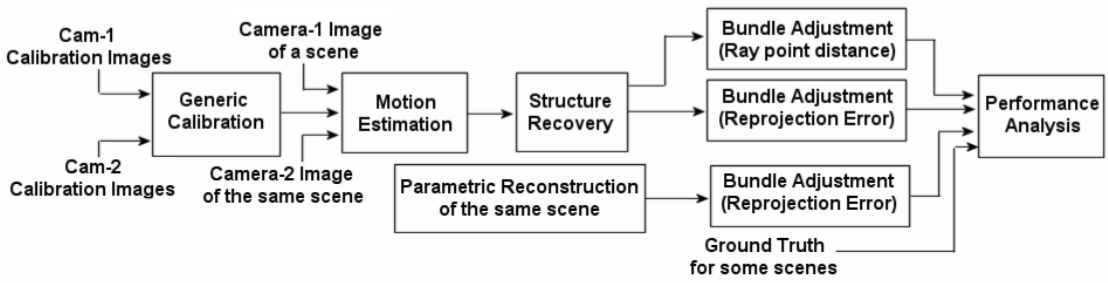


Figure 8.1: The overall pipeline of the generic structure-from-motion approach.

8.2.1 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 [58]. We briefly summarize how the classical problem is usually solved [49]. 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 $(\mathbf{x}, \mathbf{x}')$ across two views, using the epipolar constraint $\mathbf{x}'^\mathsf{T} \mathsf{E} \mathbf{x} = 0$. 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 [74]. Once the essential matrix is estimated, the motion parameters R and $\mathbf{t}$ can be extracted relatively straightforwardly [74].

In the case of our general imaging model, motion estimation is performed similarly, using pixel correspondences $(\mathbf{x}, \mathbf{x}')$. Using the calibration information, the associated projection rays can be computed. Let them be represented by their Plücker coordinates [49], i.e. 6-vectors $\mathbf{X}$ and $\mathbf{X}'$. The epipolar constraint extends naturally to rays, and manifests itself by a 6×6 essential matrix, defined as:

$$\mathcal{E} = \begin{pmatrix} \mathsf{R} & -\mathsf{E} \\ 0 & \mathsf{R} \end{pmatrix}$$

The epipolar constraint then writes: $\mathbf{X}'^\mathsf{T} \mathcal{E} \mathbf{X} = 0$ [81]. Linear estimation of $\mathcal{E}$ requires 17 correspondences. Once $\mathcal{E}$ is estimated, motion can again be extracted straightforwardly: R can simply be read off $\mathcal{E}$, as the upper left or lower right 3×3 sub-matrix, or the average of both. The obtained R will usually not obey the orthonormality constraints of a rotation matrix. We correct this by computing the orthonormal matrices that are closest to the original matrices (in the sense of the Frobenius norm). This can be done in the following way. Let the SVD of the estimated R be given by $\mathsf{R} = \mathsf{U}\Sigma\mathsf{V}^\mathsf{T}$. An orthonormal estimate for the rotation matrix R is then given by $\mathsf{U}\mathsf{V}^\mathsf{T}$, plus possibly a multiplication of the whole matrix by -1, to make its determinant equal to $+1$ (otherwise, the recovered matrix represents a reflection and not a rotation). This approximation is also reasonable because we anyway refine the rotation matrix using bundle adjustment.

The next step is the computation of the translation component $\mathbf{t}$. Note that 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). Later in section 8.4 we will observe a scenario with a stereo camera and a central omni-directional camera. Since the stereo camera (by considering it as a single general camera) models a non-central camera we automatically extract the scale information during the motion estimation. However in experiments involving only central systems, we need to use some knowledge about the scene to obtain the scale information. In any case the evaluation methods are independent of the absolute scale of the scene.

Estimation of $\mathbf{t}$ can be done as follows: $\mathcal{E}$ is usually estimated up to scale, and we first eliminate this ambiguity. Let A and B be the upper left and lower right 3×3 submatrices of $\mathcal{E}$. We estimate a scale factor λ, that minimizes the sum of the squared Frobenius norms of $\lambda\mathsf{A} - \mathsf{R}$ and $\lambda\mathsf{B} - \mathsf{R}$. This is a simple linear least squares problem. Then, multiply $\mathcal{E}$ with λ and let C be the upper right 3×3 submatrix of the product. We compute $\mathbf{t}$ as the vector that minimizes the Frobenius norm of $\mathsf{C} + [\mathbf{t}]_\times \mathsf{R}$. This is again a linear least squares problem.

Other algorithms for computing R and $\mathbf{t}$ from $\mathcal{E}$ are possible of course, but in any case, the computation may be followed by a non-linear optimization of R and $\mathbf{t}$ (by carrying out the associated sub-part of a bundle adjustment). Also note that the theoretical minimum number of required correspondences for motion estimation is 6 instead of 5 (due to the absence of the scale ambiguity), and that it might be possible, though

very involved, to derive a minimal 6-point method along the lines of [74]. More details on motion estimation are available in [101].

8.2.2 Triangulation

We now describe an algorithm for 3D reconstruction from two or more calibrated images with known relative position. Let $\mathbf{P} = (X, Y, Z)^\mathsf{T}$ 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 [48, 81], i.e. determine $\mathbf{P}$ that is closest in average to the n rays. Let us represent generic points on rays using position parameters λ_i. Then, $\mathbf{P}$ is determined by minimizing the following expression over X, Y, Z and the λ_i: $\sum_{i=1}^{n} \|\mathbf{A}_i + \lambda_i \mathbf{B}_i - \mathbf{P}\|^2$.

This is a linear least squares problem, which can be solved e.g. via the Pseudo-Inverse, leading to the following explicit equation:

$$\begin{pmatrix} \mathbf{P} \\ \lambda_1 \\ \vdots \\ \lambda_n \end{pmatrix}_{3+n} = \underbrace{\begin{pmatrix} n\mathtt{I}_3 & -\mathbf{B}_1 & \cdots & -\mathbf{B}_n \\ -^\mathsf{T}B_1 & 1 & & \\ \vdots & & \ddots & \\ -^\mathsf{T}B_n & & & 1 \end{pmatrix}^{-1}}_{\mathsf{M}_{(3+n)\times(3+n)}} \begin{pmatrix} \mathtt{I}_3 & \cdots & \mathtt{I}_3 \\ -^\mathsf{T}B_1 & & \\ & \ddots & \\ & & -^\mathsf{T}B_n \end{pmatrix}_{(3+n)\times(3n)} \begin{pmatrix} \mathbf{A}_1 \\ \vdots \\ \mathbf{A}_n \end{pmatrix}_{3n}$$

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 be performed very efficiently, as typically done in bundle adjustment for example [117]. Here, we even get a closed-form solution, based on:

$$\mathsf{M}^{-1} = \begin{pmatrix} \frac{1}{n}\left\{\mathtt{I}_3 + \mathsf{B}\mathsf{B}^\mathsf{T}\mathsf{C}^{-1}\right\} & \mathsf{C}^{-1}B \\ \mathsf{B}^\mathsf{T}\mathsf{C}^{-1} & \mathtt{I}_n + \mathsf{B}^\mathsf{T}\mathsf{C}^{-1}\mathsf{B} \end{pmatrix}$$

where $\mathsf{B} = (\mathbf{B}_1 \cdots \mathbf{B}_n)_{3\times n}$ and $\mathsf{C} = n\mathtt{I}_3 - \mathsf{B}\mathsf{B}^\mathsf{T}$.

The closed-form solution for $\mathbf{P}$ (C^{-1} can be computed in closed-form) is then:

$$\mathbf{P} = \frac{1}{n}\left\{\mathtt{I}_3 + \mathsf{B}\mathsf{B}^\mathsf{T}\mathsf{C}^{-1}\right\}\sum_{i=1}^{n}\mathbf{A}_i - \mathsf{C}^{-1}\sum_{i=1}^{n}\mathbf{B}_i\mathbf{B}_i^\mathsf{T}\mathbf{A}_i$$

This can be slightly simplified, as follows.

$$\mathbf{P} = \mathsf{C}^{-1}\sum_{i=1}^{n}(\mathsf{I} - \mathbf{B}_i\mathbf{B}_i^\mathsf{T})\mathbf{A}_i$$

To summarize, 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. A further simplification is possible, if the footpoints $\mathbf{A}_i$ are chosen such that their dot product with the $\mathbf{B}_i$ vanishes. In that case:

$$\mathbf{P} = \mathsf{C}^{-1}\sum_{i=1}^{n}\mathbf{A}_i$$

However, one may not gain any computational advantage, the simplification being counterbalanced by the necessity to compute the appropriate $\mathbf{A}_i$.

8.2.3 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 [44]. 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 via scalars λ_i, like in the previous section: $\mathbf{A}_i + \lambda_i \mathbf{B}_i$.

We know the structure of the observed object, i.e. 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 [44], a concept that is not applicable (at least in a direct way) to non-central cameras. In any case, a unique solution can be obtained using one or two additional points [44]. More details on pose estimation for non-central cameras are given in [18, 75].

8.3 Bundle Adjustment

8.3.1 Ray-point bundle adjustment

Triangulation

This technique minimizes the distance between projection rays and 3D points, over camera motion and 3D structure. We briefly describe our cost function. Let $\mathbf{C}_j = (X_j, Y_j, Z_j)^\mathsf{T}$ be the 3D coordinates of the jth point. Consider the ith image and assume that the projection ray corresponding to $\mathbf{C}_j$ is the kth ray of the camera. Let this ray be represented like above by a base point $\mathbf{A}_k$ and a direction $\mathbf{B}_k$ ($\mathbf{B}_k$ is chosen to have unit norm). Note that here, we assume these are known, since we consider calibrated cameras. Let R_i and $\mathbf{t}_i$ be the pose of the camera for the ith image. Then, points on the considered projection ray are represented by a scalar λ:

$$\mathbf{A}_k + \mathbf{t}_i + \lambda \mathsf{R}_i \mathbf{B}_k$$

We now seek to compute the (squared) distance between this ray and the point C_j. It is given by:

$$e_{ijk} = \min_{\lambda_{ijk}} \|\mathbf{A}_k + \mathbf{t}_i + \lambda_{ijk} \mathsf{R}_i \mathbf{B}_k - \mathbf{C}_j\|^2$$

It can easily be computed in closed-form; the λ_{ijk} minimizing the above expression is:

$$\lambda_{ijk} = \mathbf{B}_k^\mathsf{T} \mathsf{R}_i^\mathsf{T} (\mathbf{C}_j - \mathbf{A}_k - \mathbf{t}_i)$$

Bundle adjustment consists then in minimizing the sum of all squared distances e_{ijk} (for all available matches between points and pixels/rays), over the 3D point positions and the camera motions. This is a non-linear least squares problem, and appropriate optimization methods such as Gauss-Newton or Levenberg-Marquardt may be used for its solution.

Note that this bundle adjustment is completely generic: due to working with projection rays, it may be applied to any calibrated camera, be it central or non-central. One might include the calibration in the optimization and minimize the cost function also over projection ray coordinates (in that case, the representation using a base point and a direction may not necessarily be the best choice). This is easy to write down and implement, but one needs sufficient data to get meaningful estimates: in a fully non-central model for example, each estimated ray needs at least two associated 3D points, i.e. the pixel associated with that ray, has to correspond to actual interest points in at least two images. This can only be achieved for sufficiently many rays if a reliable *dense* matching is possible.

We can use bundle adjustment to refine the pose of all grids (except for the first one) and the projection rays during calibration. Bundle adjustment can be applied at any stage of our approach; we apply it after the initial calibration using multiple grids (see section 7.1.1), for refining the pose of each additional grid (see section 7.1.2), as well as at the end of the whole calibration.

We have developed bundle adjustment algorithms for both central, noncentral and axial scenarios as given below. Details are given in the following.

Bundle adjustment for central cameras

All the projection rays pass through the optical center. Thus any ray can be expressed using the optical center and a direction vector. The overall error, E, to be minimized is given below.

$$E = \sum_{i=1}^{m} \sum_{j=1}^{n} \|\mathbf{O} + \lambda_{ij}\mathbf{D}_i - [\mathsf{R}_j \mathbf{t}_j]\mathbf{P}_{ij}\|_2^2$$

- $\mathbf{O}$ is the camera center(since it is a central camera),
- $\mathbf{D}_i$ is the direction of the i_{th} ray starting at $\mathbf{C}$, where m is the total number of rays.
- λ_{ij} is a parameter to select specific point on the i_{th} ray, where n is the total number of grids.
- $\mathbf{P}_{ij}$ is the point for a point on the i_{th} ray,
- $[\mathsf{R}_j, \mathbf{t}_j]$ is the pose of the j_{th} grid.

Please note that the grid points from all the grids may not lie on any given ray. Since $\mathbf{D}_i$ is a direction vector it has only two degrees of freedom. We use the orthonormal representation, introduced in [8].

$$\mathbf{D}_i = \mathsf{U}_i \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

where $\mathbf{U}_i$, which is a 3×3 rotation matrix.

The bundle adjustment involves the iterative optimization of rotation matrices. Different parameterizations for rotation matrices exist. We use Euler angles, but instead of computing angles corresponding to the absolute orientation with respect to the global coordinate system, we compute the angles corresponding to the relative orientation with respect to the absolute orientation obtained in the previous iteration of the optimization. This is common practice for bundle adjustment, and allows to avoid the singularities inherent with the Euler angle representation. Concretely, let R^k be the rotation matrix obtained after the previous iteration of bundle adjustment. The rotation matrix in the current iteration is parameterized as $\mathsf{R}^{k+1} = \mathsf{R}^k\mathsf{R}(\alpha, \beta, \gamma)$, with angles α, β and γ expressing the relative orientation relative to R^k. The initial values for these angles

at the current iteration are zero. The cost function and its derivatives thus have a simple expression. Further, around the identity the Euler angle representation is free of singularities. After estimating α, β and γ, R^{k+1} is computed. At the following iteration, new update angles, relative to R^{k+1}, are computed, again with initial values being zero.

As for the update of the matrices U_i (see above), only 2 Euler angles are required.

$$\mathsf{U} \leftarrow \mathsf{U}\mathsf{R}(\theta) \; with \; \mathsf{R}(\theta) = \mathsf{R}_x(\theta_1)\mathsf{R}_y(\theta_2),$$

where $\mathsf{R}_x(\theta_1)$ and $\mathsf{R}_y(\theta_2)$ are orthonormal rotation matrices representing 3D rotations around the x and y axis respectively. When we use the orthonormal representations algebraic constraints are automatically taken into account and the system becomes well-conditioned.

Bundle adjustment for non-central cameras

There is no single optical center for noncentral cameras. We need to initialize the bundle adjustment with a separate point and a direction for every projection ray. Two grid points are sufficient to compute a ray. In the case of more than two grid points we use least squares approach to compute the optimal ray. Let $\mathbf{C}$ and $\mathbf{D}$ represent a point on the ray and its direction respectively. The ray is computed using the method given in section 5.3.

The scheme of parameterizing rotation matrices, explained in the previous section, can also be applied to parameterize and optimize individual unit vectors. Here, we use it to parameterize 3D lines. Each line is represented by two 3D points; using unconstrained points would constitute an overparameterization by 2 of the line (a 3D line has 4 degrees of freedom). We thus propose a parameterization and update procedure similar to that for rotation matrices, that at each iteration estimates exactly 4 parameters.

We can represent a ray with a minimum of four parameters. We use two parameters to represent the direction $\mathbf{D}_i$ of the ray as in the central case. Let $\mathbf{C}_i$ be the point closest to the origin on the i_{th} ray. We represent $\mathbf{C}_i$ using two parameters from $\mathbf{D}_i$ and two additional parameters (s and θ_3) as shown below. This is possible because $\mathbf{C}_i$ is perpendicular to $\mathbf{D}_i$.

$$\mathbf{C}_i = s\mathsf{V}\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

where V, which is a 3×3 rotation matrix that can be updated as given below.

$$\mathsf{V} \leftarrow \mathsf{U}\mathsf{R}(\theta) \; with \; \mathsf{R}(\theta) = \mathsf{R}_x(\theta_1)\mathsf{R}_y(\theta_2)\mathsf{R}_z(\theta_3),$$

where $\mathsf{R}_x(\theta_1)$, $\mathsf{R}_y(\theta_2)$ and $\mathsf{R}_z(\theta_3)$ are orthonormal rotation matrices representing 3D rotations around the x and y axis respectively. Also $\mathsf{R}_x(\theta_1)$ and $\mathsf{R}_y(\theta_2)$ are the same rotation matrices used in the update of $\mathbf{D}_i$. Thus we have $(\theta_1, \theta_2, \theta_3, s_i)$ as the only four parameters in the update of a ray in a noncentral camera. The cost function to be minimized is given below.

$$E = \sum_{i=1}^{m}\sum_{j}^{n} \| s_i\mathsf{V}_i \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + \lambda_{ij}\mathbf{D}_i \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} - [\mathsf{R}_j\mathbf{t}_j]\mathbf{Q}_{ij} \|_2^2$$

Bundle adjustment for axial Cameras

We give the details of a bundle adjustment which refines the estimated camera axis, projection rays and poses of the calibration grids. The bundle adjustment is done by minimizing the distance between the grid

points and the corresponding projection rays. The cost function is given below.

$$Cost = \sum_{i=1}^{n} \sum_{j=1} \|(\mathbf{A} + \lambda_i \mathbf{D} + \mu_{ji}\mathbf{D}_i - [\mathsf{R}_j \mathbf{t}_j]\mathbf{Q}_{ji})\|^2$$

- $(\mathbf{A}, \mathbf{D})$ - represents the axis (point, direction)
- $\mathbf{D}_i$ - unit direction vector of the i_{th} projection ray
- λ_i - parameter for the intersection of the i_{th} ray and the axis
- $\mathbf{Q}_{ji}$ - grid point on the j_{th} grid lying on the i_{th} ray
- μ_{ji} - parameter the point on the i_{th} ray closest to Q_{ji}
- $(\mathsf{R}_j, \mathbf{t}_j)$ - pose of the calibration grid

8.3.2 Re-projection-based bundle adjustment

We now describe some challenges in using re-projection-based bundle adjustment for the generic imaging model and our approaches to overcome these. In the generic imaging model, there is no analytical projection equation, since calibration more or less corresponds to a lookup table that gives projection ray coordinates for individual pixels (or, image points). Thus, to project a 3D point, search and interpolation are required: one searches for a certain number (could be equal to 1) among the camera's projection rays that are closest to the 3D point. The coordinates of the image point can then be computed by interpolating the coordinates of the pixels associated with these rays. Efficient optimization for re-projection-based bundle adjustment would require the computation of derivatives of this projection function; although numerical differentiation is possible and rather straightforward, it is time-consuming.

We solve this problem by considering a camera as a cluster of central cameras: given a set of rays belonging to a non-central camera, we partition them into k clusters of rays, each having its own optical center. For example, $k = 2$ for a stereo system. In addition we also impose the condition that each ray should be contained by only one cluster. In the following, we describe a simple clustering method and then, how we perform bundle adjustment over these ray clusters.

The clustering is obtained using a 3D Hough transform (mapping rays in 3D to 3D points), which we explain briefly. First we transform the "ray space", consisting of rays in space, to a discretized "point space", where we use a counter (initialized to zero) for every 3D point. Then every ray updates the counters (increase by 1) of the 3D points lying on it. Next we identify the 3D point having the largest count. This point becomes the center of the first cluster and the rays that contributed to its count, are grouped to form the cluster. The contribution of these rays to other points' count is then deleted, and the process repeated to determine other clusters. With a reasonably good resolution for the point space in the 3D Hough transform, we can obtain the correct number of clusters in simple configurations such as stereo camera and multicamera network, where the centers are distinct. However in catadioptric systems having complex caustics, the resolution of the 3D point space in the Hough transform determines the number of discrete clusters we can obtain. Each such cluster is in the following interpreted as a central camera. We synthesize a *perspective* image for each one of them, that will be used in the parameterization for the bundle adjustment. A perspective image for a cluster of rays, can be easily computed by intersecting the rays with some properly chosen plane, henceforth denoted as image plane (cf. [104]). We thus generate k perspective images, one per cluster of rays. Each of them is parameterized by the position of its optical center (the center point of the cluster), the directions of the projection rays and the position of the image plane. We have thus created a parameterization for an analytical projection equation from 3D points to 2D coordinates (instead of only a lookup table between rays

and pixels). It is used in bundle adjustment to compute and minimize the re-projection error simultaneously on all these synthesized images.

We now briefly describe how to choose an image plane for a cluster of rays. To do so, we propose to minimize the "uncertainty" in the intersection points of image plane and rays: ideally the rays should be perpendicular to the plane, and therefore we find the plane's orientation which minimizes the sum of all acute angles between the plane and rays:

$$\min_{m_1,m_2,m_3} \sum_{i=1}^{n} (m_1 l_1^i + m_2 l_2^i + m_3 l_3^i)^2$$

where (l_1^i, l_2^i, l_3^i) refers to the direction of the i_{th} ray (unit vector) and (m_1, m_2, m_3) is the normal of the image plane. The normal is given as the unit null-vector of the matrix:

$$\begin{pmatrix} \sum (l_1^i)^2 & \sum l_1^i l_2^i & \sum l_1^i l_3^i \\ \sum l_1^i l_2^i & \sum (l_2^i)^2 & \sum l_2^i l_3^i \\ \sum l_1^i l_3^i & \sum l_2^i l_3^i & \sum (l_3^i)^2 \end{pmatrix}$$

The distance between the image plane and the center of the cluster does not matter as long as we keep it the same for all clusters. Thus we place the image planes at the same distance for all the individual clusters.

It is useful to discuss what happens to our algorithm in extreme cases. The first case is when we have only one ray in a cluster. For example in a completely non-central camera, which is referred to as an oblique camera [78], where each ray belongs to a separate central cluster. In that case we consider a plane perpendicular to that ray and the center will be kept at infinity. Our re-projection-based algorithm will be exactly the same as a ray-point approach.

The next interesting case is that of a highly non-central camera, where the number of clusters is very large. We will have to generate many perspective images and if we use the above optimization criterion for computing the normal for the intersecting plane, then this algorithm tends to become a ray-point distance based bundle adjustment. Finally if the camera has just one cluster it becomes the conventional re-projection-based algorithm, if the image coordinates in the synthesized perspective image match with that of the original image. In addition to allowing the use of a re-projection based approach, our clustering technique makes a compromise between fully central (stability) and fully non-central (generality).

A possible improvement to the above approach is to identify a plane and generate a perspective view where the image coordinates are close to the original image coordinates, which would better preserve the noise model in the image. Preliminary results with this approach are promising.

In general noncentral omnidirectional cameras are constructed using mirrors and lenses. These catadioptric configurations, constructed using spherical, parabolic and hyperbolic mirrors, are either central or approximately central. The second scenario can either be approximated to a central camera or accurately modeled using a large number of clusters. On following the second option we observe the following. Firstly it is very difficult to cluster in the presence of noise. Secondly the bundle adjustment is more or less the same as the ray-point one. Thus it was not necessary for us to demonstrate the clustering for non-central omnidirectional cameras. More precisely the re-projection based approach is meaningful only to non-central configurations with distinct clusters such as stereo and multi-camera scenarios.

8.4 Results and Analysis

We consider three indoor scenarios:

- A house scene captured by an omni-directional camera and a stereo system (cf. Figure 8.4(b)).

- A house scene captured by an omni-directional and a pinhole camera (same scene as in Figure 8.4(b)).
- An objects scene, which consists of a set of objects placed in random positions as shown in Figure 8.4(a), captured by an omni-directional and a pinhole camera.

The following cameras were used: Nikon Coolpix 5400 as pinhole camera, the "Bumblebee stereo camera", and the Nikon Coolpix 5400 with an "FC-E8" fisheye converter to give omni-directional images with a field of view of $360^\circ \times 183^\circ$.

We first briefly describe how the cameras used were calibrated, and then present experiments and results with the algorithms described in this chapter.

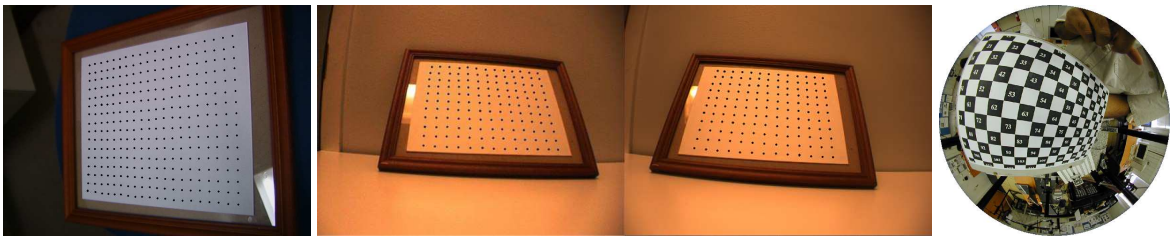
8.4.1 Calibration

We calibrate three types of cameras. They are pinhole, stereo, and omni-directional systems. Sample calibration images for these are shown in Figure 8.2 and some visual calibration information is given in Figure 8.3.

Pinhole camera. Figure 8.3(a) shows the calibration of a regular digital camera using the central model (see chapter 5).

Stereo system. Here we calibrate the left and right cameras separately as two individual central cameras. In the second step we capture images of a 3D scene and compute the motion between the two cameras using the technique described in Section 8.2.1. Finally, using the computed motion we obtain the rays of the two cameras in the same coordinate system, which thus constitutes the calibration information for this non-central system.

Omni-directional camera. We assume the camera to be central. Figure 8.3(c) shows that we have used more than three calibration grids to calibrate the camera, which is due to the fact that the minimum required number of three images is seldom sufficient to completely calibrate the whole field of view. Thus we placed a checkerboard grid, shown in Figure 8.2(c), on a turntable and captured a sequence of images to cover the entire field of view. Then we used a few overlapping images to obtain a partial initial calibration [105]. This provides, in a single coordinate system: the pose of the calibration grid for the images used, the position of the camera's optical center and the direction of projection rays for the pixels in the overlap region. The camera was calibrated using the approach of chapter 7. The calibrated image region shown in Figure 8.3(c) was obtained using 23 images.

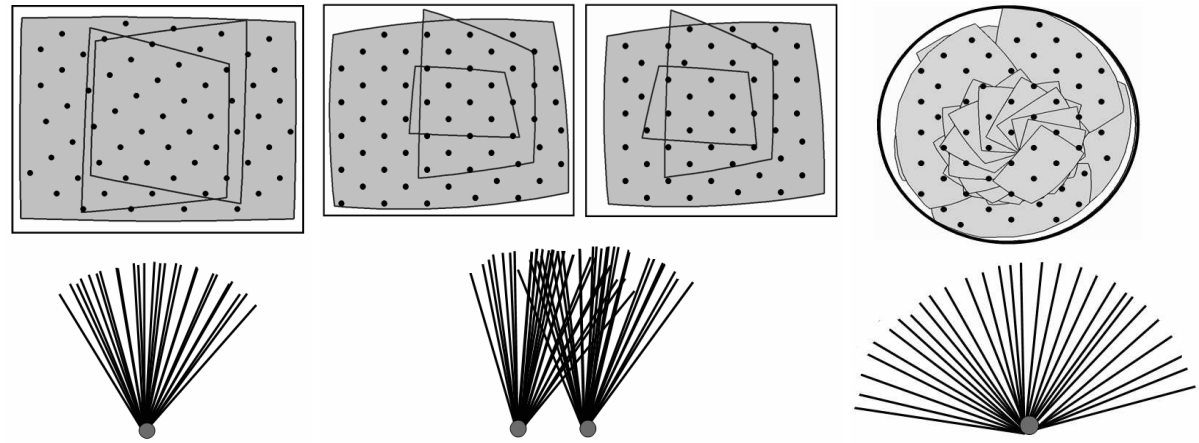


(a) (b) (c)

Figure 8.2: Sample calibration images (not necessarily the ones used for the calibration, as shown in Figure 8.3). For (a) pinhole and (b) stereo, circular calibration targets are used. For (c) omni-directional, checkerboard grids are used.

8.4.2 Motion and structure recovery

Two scenarios are considered here: combining an omni-directional camera with either a pinhole camera or a stereo system.



(a) (b) (c)

Figure 8.3: Calibration information. (a) Pinhole. (b) Stereo. (c) Omni-directional. The shading shows the calibrated regions, i.e. the regions of pixels for which projection rays were determined. The 3D rays shown on the bottom correspond to the image pixels marked in black. We also show the outlines of the calibration grids (enclosing the image pixels).

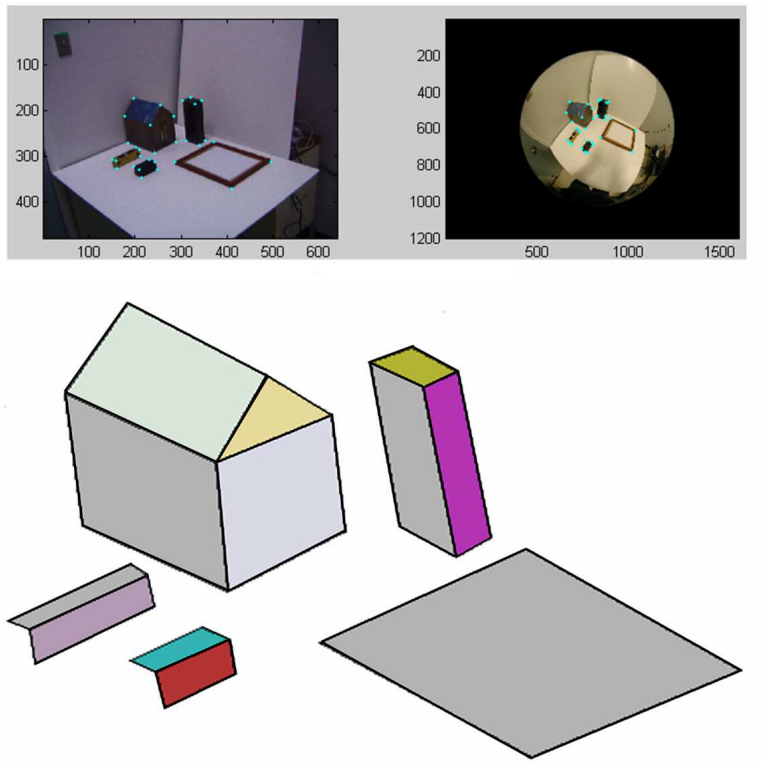
Pinhole and omni-directional. Since the omni-directional camera has a very large field of view and consequently lower resolution compared to the pinhole camera, the images taken from close viewpoints from these two cameras have different resolutions as shown in Figure 8.4(a). This poses a problem in finding correspondences between images. Operators like SIFT [59], are scale invariant, but not fully 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. From the matched points, we triangulated the 3D structure. The result (cf. Figure 8.4(a)) suggests that the algorithms used here (calibration, motion estimation, triangulation) are correct and work in practice.

Stereo system and omni-directional. Here, we treat the stereo system as a single, non-central camera; the same procedure as for the above case are applied: manual matching, motion estimation, triangulation. The only difference is that the same scene point may appear twice in the stereo camera, but this makes no difference for our algorithms. Although a simple 3D structure is used here, the result again suggests that the algorithms are correct. This experiment underlines the fact that they are generic, i.e. may be used for any camera and combination of cameras that are modeled by the generic imaging model.

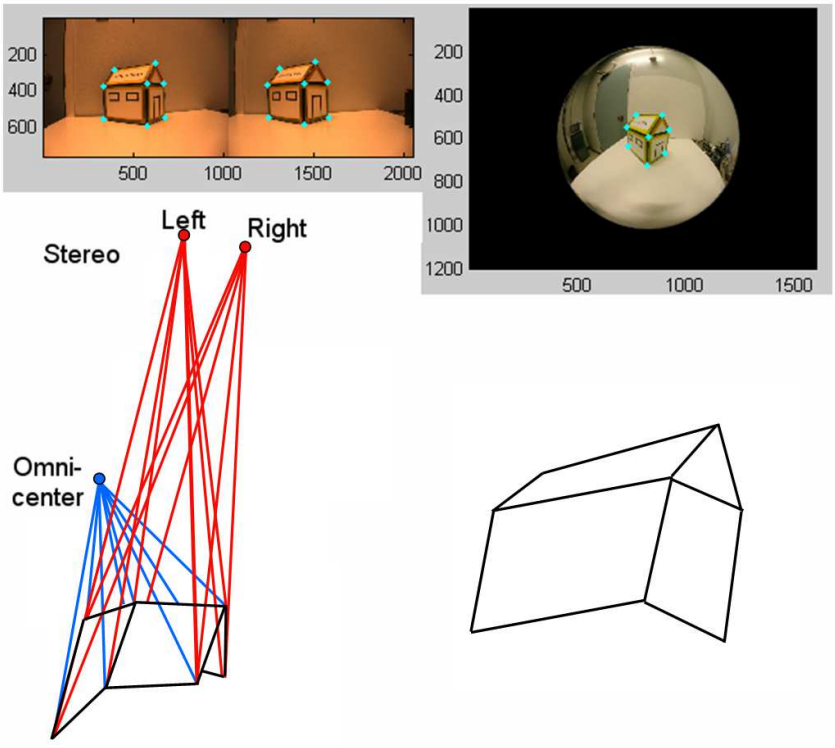
8.4.3 Bundle adjustment statistics

We discuss the convergence rate, error criteria and performance of the two bundle adjustment algorithms. Convergence rate is measured by the number of iterations. Accuracy is measured as follows: the reconstructed 3D points are first scaled such that the sum of squared distances from their centroid equals 1. This way, the accuracy measurements become relative to scene size. Then, we compute all possible pairwise distances between reconstructed 3D points. These are then compared to ground truth values if available. We also compare them to the analogous information obtained from 3D reconstruction using pinhole images only and classical structure-from-motion methods: motion estimation, triangulation and re-projection-based bundle adjustment for perspective cameras [49].

House scene. For the house scene (cf. Figure 8.4(b)), ground truth is available (manual measurement of distances). We compute the relative error between reconstructed distances d_{ij} and ground truth distances



(a)



(b)

Figure 8.4: Results of motion estimation and 3D reconstruction for cross-camera scenarios. (a) Pinhole and omni-directional. (b) Stereo and omni-directional. Shown are the reconstructed 3D points, the optical centers (computed by motion estimation) and the projection rays used for triangulation.

$\bar{d}_{ij}$ between all pairs (i, j) of 3D points:

$$\frac{|d_{ij} - \bar{d}_{ij}|}{\bar{d}_{ij}}$$

Table 8.1 shows the mean of these relative errors, given in percent. Values are shown for three camera setups: omni-directional image combined with a pinhole or a stereo system, and two pinhole images. Three methods are evaluated: classical (perspective) algorithms (called "Parametric" here), and generic algorithms, with the two different bundle adjustment methods ("Ray-Point" and "Re-projection"). Histograms giving the distribution of relative distance errors are also shown, in Figure 8.5.

Scene	Points	Camera 1	Camera 2	Parametric (it, error)	Ray-Point (it, error)	Re-projection (it, error)
House	8	Stereo	Omni	–	(26, 2.33)	(7, 1.54)
House	8	Pinhole	Omni	–	(18, 3.05)	(5, 4.13)
House	8	Pinhole	Pinhole	(8, 2.88)	–	–

Table 8.1: Statistics for the house scene. *it* refers to the number of iterations of bundle adjustment and *error* refers to the *mean relative error* on distances between 3D points, expressed in percent.

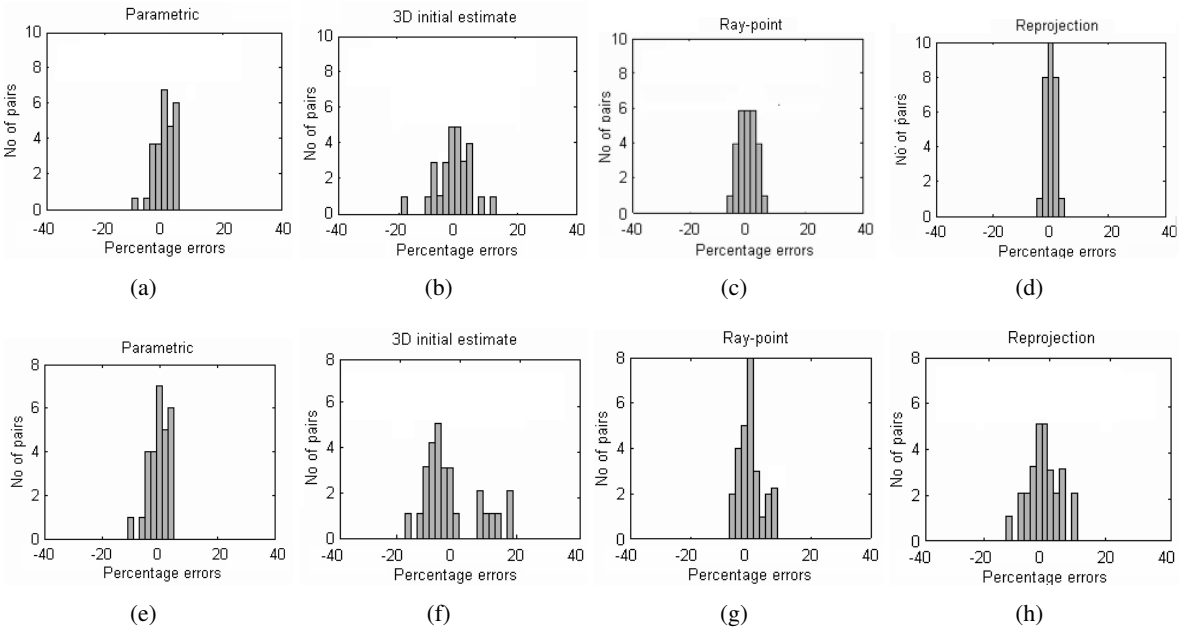


Figure 8.5: Histograms for the house scene. Top: results for the combination of stereo and an omni-directional image (besides for the left column, where two pinhole images are used). Bottom: combination of a pinhole and an omni-directional image. Please note that the different graphs are scaled differently along the y-axis.

As for the two generic bundle adjustment methods, we observe that the re-projection method converges in fewer iterations than the ray-point method. Both bundle adjustments reduce the error in the initial 3D estimate (after motion estimation and triangulation) significantly. As for the accuracy, each one of the two bundle adjustments is better than the other one in one scenario.

We also observe that the generic approaches perform better than the classical parametric one in the case they use an omni-directional camera and a stereo system; this is not surprising since one more image is used than the two pinhole images of the classical approach. Another possible reason might be the use of more parameters as compared to classical approaches. Thus they will have a good local minimum. Nevertheless, this again confirms the correctness and applicability of our generic approaches. It is no surprise either that performance is worse for the combination of a pinhole and an omni-directional image, since the spatial resolution of the omni-directional image is much lower than those of the pinhole images.

Objects scene. For this scene (cf. Figure 8.4(a)), no complete ground truth is available. We thus computed the differences between point distances obtained in reconstructions with the 3 methods. Concretely, for some methods X and Y, we compute, for all point pairs (i, j):

$$\frac{|d_{ij}^X - d_{ij}^Y|}{d_{ij}^Y}$$

where d_{ij}^X respectively d_{ij}^Y are pairwise distances obtained by using methods X and Y respectively. Figure 8.6 shows the histograms for this measure and Table 8.2 gives some details on this scene and the number of iterations for the different methods. In this scenario as well, the re-projection method converges in fewer iterations than the ray-point method.

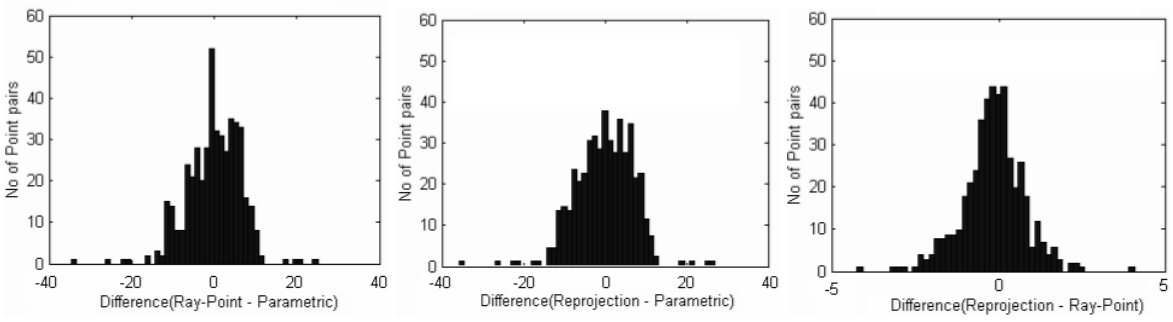
The mean values of the above measure are as follows:

X=Ray-Point	Y=Parametric	$\rightarrow$	4.96
X=Re-projection	Y=Parametric	$\rightarrow$	5.44
X=Re-projection	Y=Ray-Point	$\rightarrow$	0.69

We observe that the refinements produced by both bundle adjustments seem to be comparable to each other.

Scene	Points	Camera 1	Camera 2	Parametric	Ray-Point	Re-projection
Objects	31	Pinhole	Omni	7	25	5

Table 8.2: Details on the objects scene. The last three columns give the number of iterations of bundle adjustment for the three methods used.



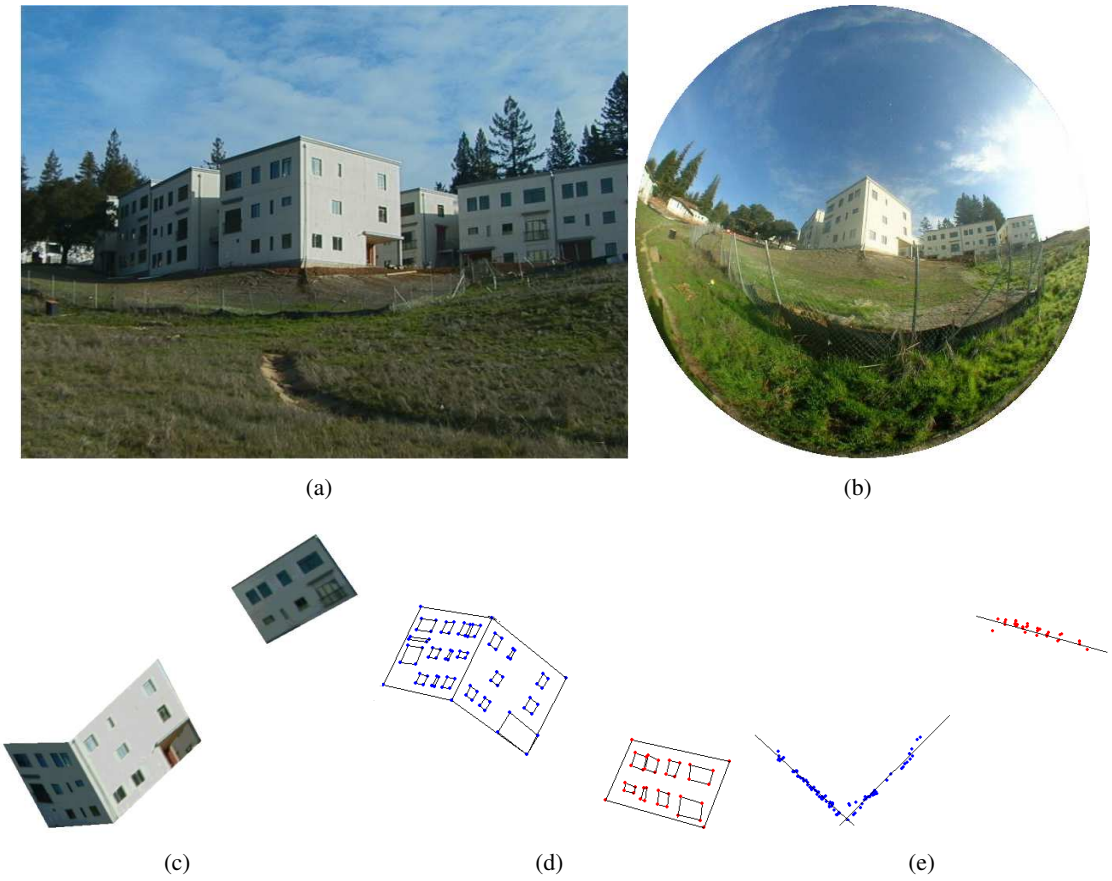
(a) (b) (c)

Figure 8.6: Histograms for the relative distance errors for the objects scene. Please note that the histograms are scaled differently along the horizontal axes.

Outdoor scene. Figure 8.7 shows results for the 3D reconstruction of an outdoor scene from two images, one omni-directional and the other pinhole. The reconstruction has 121 3D points. Figures 8.7(c)

to (e) allow a qualitative evaluation of the reconstruction, e.g. reasonable recovery of right angles (between window edges or between walls). We analyzed the reconstruction quantitatively, by measuring the deviations from right angles and from coplanarity for appropriate sets of points. To do so, we computed a least-squares plane for coplanar points and measured the residual distances. We then compute the mean distance, and express it relative to the overall size of the scene (largest distance between two points in the scene).

We also measure the angle between planes that ideally should be orthogonal, and consider the deviation from 90°. The errors are found to be low (cf. Table 8.3), considering that the images are certainly not ideal for the reconstruction task. Table 8.3 also contains these error measures for the house and objects scenes used above.



(a) (b) (c) (d) (e)

Figure 8.7: Outdoor scene. (a) Pinhole image. (b) Omni-directional image. (c) Texture-mapped model. (d) Mesh representation. (e) Top view of the points. We reconstructed 121 3D points, which lie on three walls shown in the images.

8.5 Conclusions

We have developed a generic approach for structure-from-motion, that works for any camera or mixture of cameras that fall into the generic imaging model used. Our approach includes methods for motion and pose estimation, 3D point triangulation and bundle adjustment. Promising results have been obtained for different image sets, obtained with three different cameras: pinhole, omni-directional (fisheye) and a stereo system. Using simulations and real data, we are interested in investigating our approach and the clustering issues in

Scene	Camera 1	Camera 2	ΔPlanarity (Ray-Point, Reproj)	ΔOrthogonality (Ray-point, Reproj)
House	Stereo	Omni	(0.37, 0.27)	(5.1, 3.5)
Objects	Pinhole	Omni	(0.38, 0.42)	(3.8, 4.14)
Outdoor	Pinhole	Omni	(0.59, 0.63)	(4.2, 5.4)

Table 8.3: ΔPlanarity and ΔOrthogonality refer to the mean residual for the least squares plane fit (relative to scene size and expressed in percent) and to the mean errors of deviations from right angles (see text for more details).

more exotic catadioptric cameras and multi-camera configurations.

Chapter 9

Generic Self-Calibration of Central Cameras

We consider the self-calibration problem for the generic imaging model that assigns projection rays to pixels without a parametric mapping. 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 chapter 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.

9.1 Introduction

In the earlier chapters 4 to 7, we looked at the generic calibration problem in the presence of calibration grids. In this chapter we aim at further flexibility, by addressing the problem of self-calibration. 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.

9.2 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 $\mathbf{p}$, we have determined all pixels that match $\mathbf{p}$ 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 9.1 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 $\mathbf{p}$ and $\mathbf{p}'$ refer to two matching pixels, i.e. pixels observing the same 3D point in I and I'. Let $\mathbf{p}''$ refer to the pixel that in I' matches to pixel $\mathbf{p}'$ in I. Similarly let $\mathbf{p}'''$ be the pixel that in I' matches to pixel $\mathbf{p}''$ in I, and so forth. The sequence of pixels $\mathbf{p}, \mathbf{p}', \mathbf{p}'', \mathbf{p}''', \ldots$ 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).

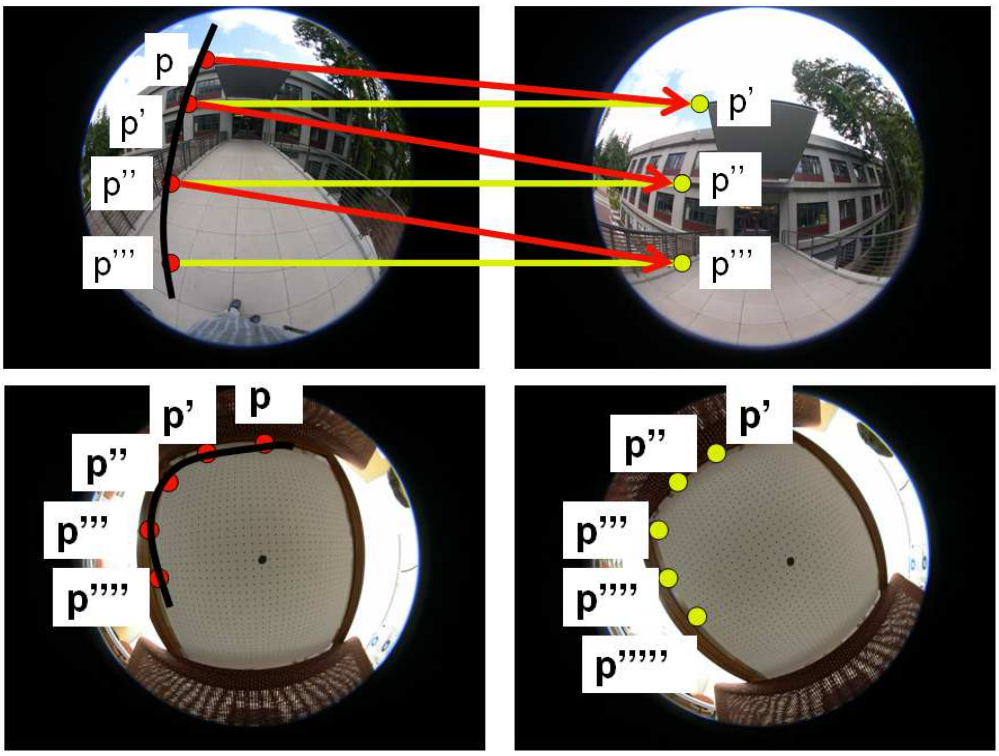


Figure 9.1: Illustration of flow curves: translational motion (top) and rotational motion (bottom).

9.3 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.

9.3.1 One translational motion

Consider two matching pixels $\mathbf{p}$ and $\mathbf{q}$, i.e. the scene point seen in pixel $\mathbf{p}$ in image 1, is seen in image 2 in pixel $\mathbf{q}$. 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.

9.3.2 One rotational motion

Let $\mathbf{L}$ be the rotation axis (going through the optical center). Consider two matching pixels $\mathbf{p}$ and $\mathbf{q}$. 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.

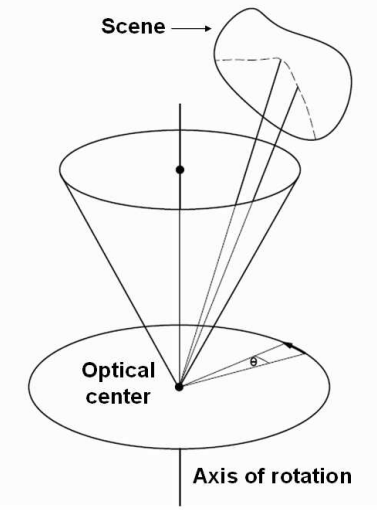


Figure 9.2: The rays of pixels in the rotation flow curve form a cone.

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 9.1. 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.

9.4 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 9.3.1). 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 [65].

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.

9.5 Self-Calibration Algorithm

We put together constraints derived in section 9.3 in order to propose a self-calibration algorithm that requires rotational and translational motions.

9.5.1 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.

9.5.2 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 $\mathbf{p}_1$ be the center pixel of the first rotation and $\mathbf{p}_2$ that of the second one. Consider the translational flow curve that contains $\mathbf{p}_1$. All pixels on one side of the flow curve starting from $\mathbf{p}_1$ will have the same ϕ_1. Similarly let ϕ_2 refer to the offset angle for pixels lying on the flow curve passing through $\mathbf{p}_2$. 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 $\mathbf{p}_2$, and $\phi_2 = 0$ for $\mathbf{p}_1$. Hence, the ray associated with $\mathbf{p}_2$ 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 $\mathbf{p}$, 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.

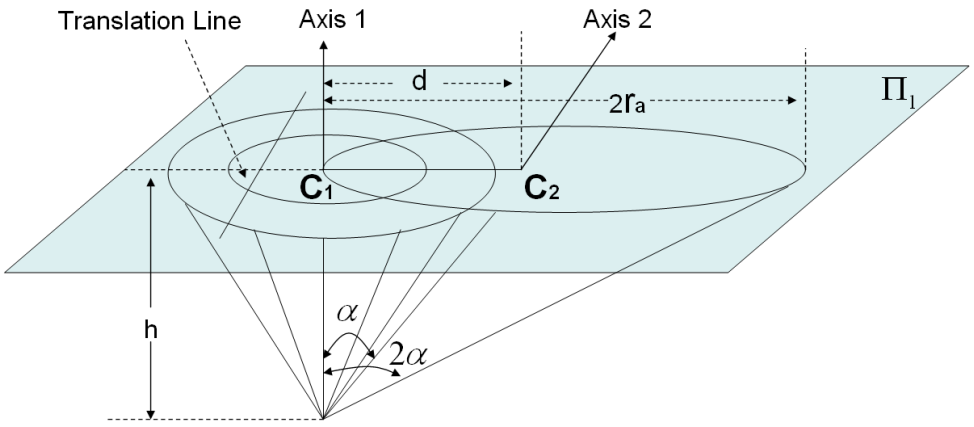
What about pixels whose rays are coplanar with the two rotation axes? This is not a problem because every computed ray direction gives the angle of the associated cone. Hence, all pixels on that cone can directly by reconstructed, by intersecting the line issuing from $\mathbf{D}_1$ or $\mathbf{D}_2$ 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 9.3(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 let us consider the intersection of the cones from the second rotation with the plane π_1. These intersections are ellipses.

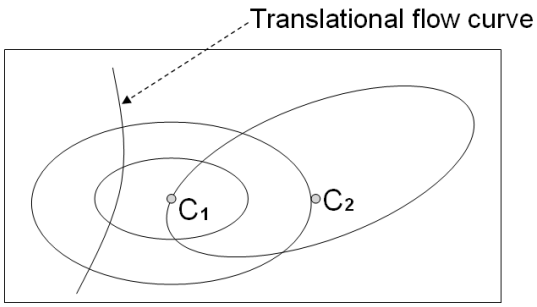
As we observed earlier translational flow curves consists of pixels whose corresponding projection rays are coplanar. The intersection of these coplanar rays and the plane π_1 is a line. We use this information to compute the relation between r_i and later the offset angles.

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.

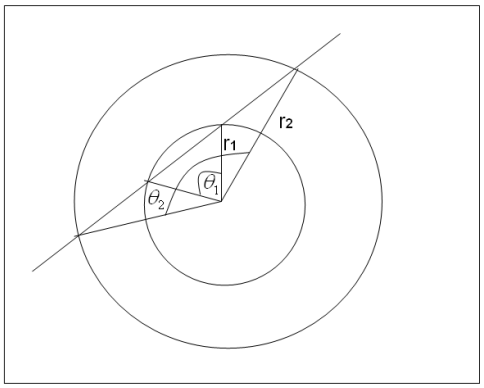
$$r_2 = \frac{cos(\frac{\theta_1}{2})}{cos(\frac{\theta_2}{2})}$$



(a)



(b)



(c)

Figure 9.3: a) We show two rotation axes and a plane π_1 orthogonal 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.

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 9.5 we show the scenario where we calibrate a fisheye camera with only few closed flow curves.

9.5.3 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.

9.6 Experiments

We tested the algorithm of section 9.5.2 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.

9.6.1 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 [90], 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 [77]. 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 9.4. We used ellipse parameterization 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

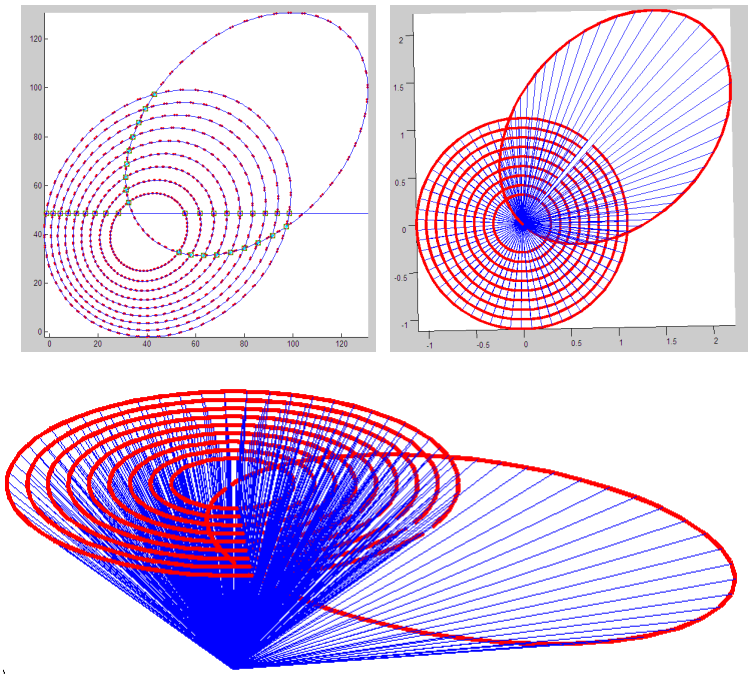


Figure 9.4: Top left: flow curves associated with a single rotation on a perspective image. We also fitted ellipses on the flow curves to analytically compute the intersections with other flow curves. Top right and bottom: projection rays after calibration in two different views.

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.

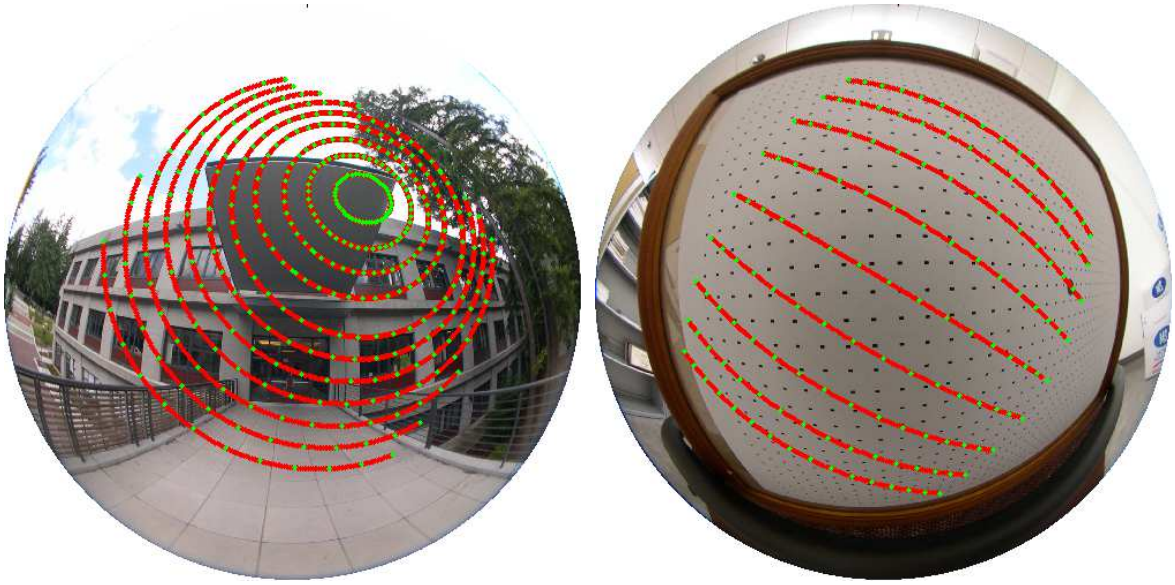
Secondly we tested our algorithm on Nikon coolpix fisheye lens, FC-E8, with a field of view of 183 degrees. In Figure 9.5 we show the translation and rotation flow curves. We fitted ellipses for both the rotational and translational flow curves.

9.6.2 Distortion correction

We show the distortion correction for few fisheye images in Figure 9.6. The technique of distortion correction is described in 5.4.2. 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.

9.7 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.



(a) (b)

Figure 9.5: a) Flow curves of pure rotation on a fisheye image. b) Translational flow curves on a fisheye image.

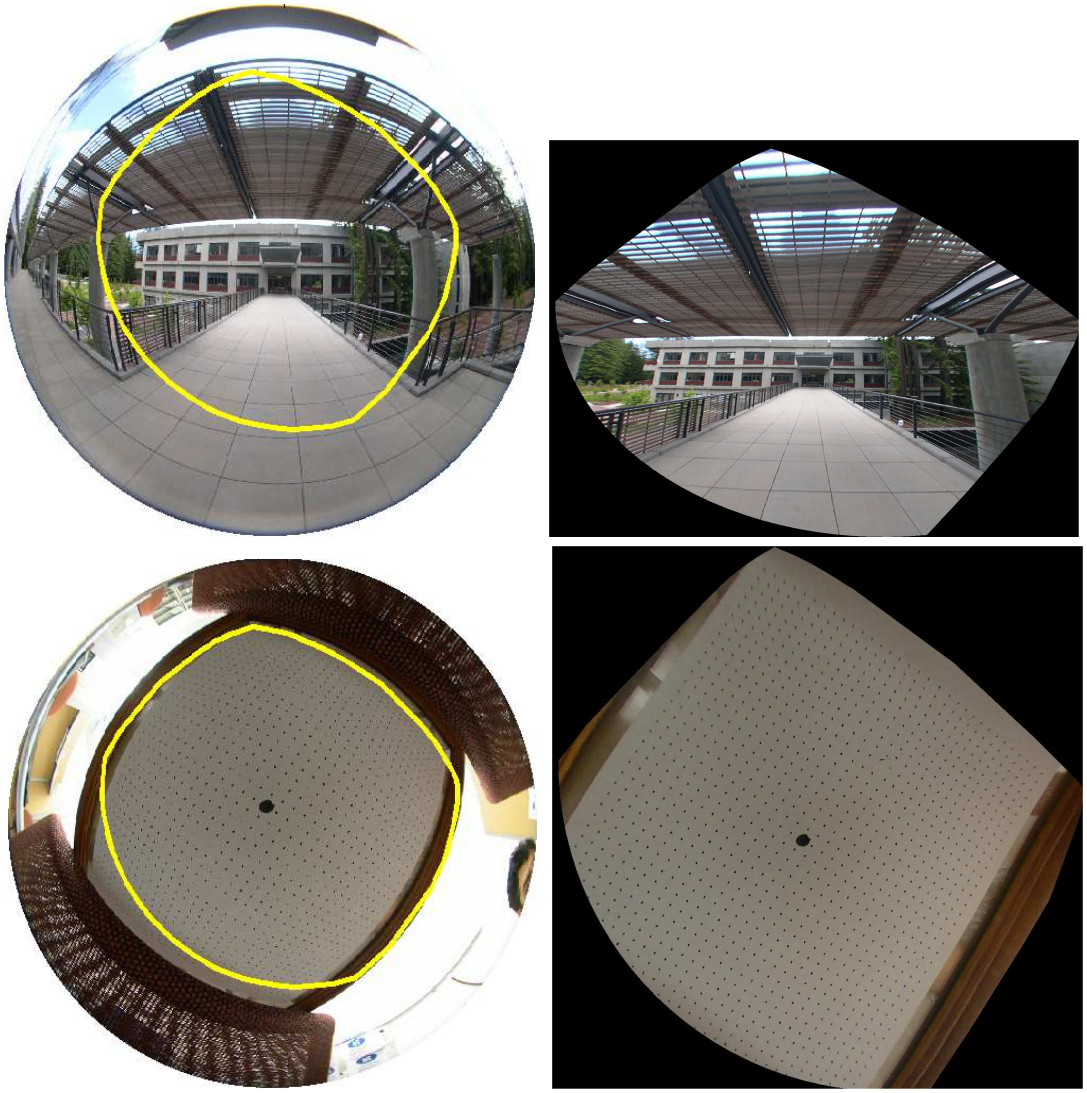


Figure 9.6: Top: original images with the boundaries showing the calibration region. Middle and bottom: generated perspective images.

Chapter 10

Generic Self-Calibration of Radially Symmetric Cameras

This chapter presents a novel approach for planar self-calibration of radially symmetric cameras. We model these camera images using notions of distortion center and concentric distortion circles around it. The rays corresponding to pixels lying on a single distortion circle form a right circular cone. Each of these cones is associated with two unknowns; optical center and focal length (opening angle). In the central case, we consider all distortion circles to have the same optical center, whereas in the non-central case they have different optical centers lying on the same optical axis. Based on this model we provide a factorization based self-calibration algorithm for planar scenes from dense image matches. Our formulation provides a rich set of constraints to validate the correctness of the distortion center. We also propose possible extensions of this algorithm in terms of non-planar scenes, non-unit aspect ratio and multi-view constraints. Experimental results are shown.

10.1 Introduction and Previous Work

In practice completely generic cameras rarely exist. Most cameras are symmetric in some way. The most common form is radial symmetry. Practically used cameras such as pinhole and fisheye are all radially symmetric (modulo a non-unit aspect ratio). Most catadioptric configurations are radially symmetric. For example, when the optical center of the pinhole camera is located on the axis of revolution of the mirror, the resulting configuration is radially symmetric.

In this chapter we propose a self-calibration algorithm for radially symmetric cameras using two or more views of unknown planar scenes. There are few works along this direction [47, 112–114]. The most closely related work is by Tardif and Sturm [113], which uses the same camera model. Thirthala and Pollefeys [114] propose a linear solution for recovering radial distortion which can also include non-central cameras. Every radial line passing through the distortion center is mapped to coplanar rays. As a consequence their algorithm involves projective reconstruction followed by metric upgradation, using the dual of the absolute conic, for self-calibration. Hartley and Kang [47] proposed a plane-based calibration algorithm for correcting radial distortion. They provide a method to compute the distortion center. However, their model is restricted to central cameras and the usage of calibration grids

Our approach maps every distortion circle around the distortion center to a cone of rays. We transform the self-calibration problem to a factorization framework requiring only a singular value decomposition using dense image matches. Dense matching is still an overhead, but in practice, it is possible to interpolate sparse matches on planar scenes. We also provide a rich set of constraints to validate or to even estimate the distortion center. There are several other works for estimating radial distortion, but these use more specific

distortion models and/or rely on the extraction of line images [13, 20, 22, 33, 63, 110].

Organization: We formally introduce our problem statement in section 10.2. Next the factorization framework for the self-calibration problem will be provided for non-central and central models. Then the possible variants such as the usage of non-planar scenes, non-unit aspect ratio and usage of more than three views are discussed. In the final section we discuss the experiments.

10.2 Problem Definition

The camera model is described in section 3.1.10. This work focuses on the planar-based self-calibration of radially symmetric cameras. Two or three views of a planar scene are captured from different camera locations. The input is the dense image matches between these views. The goal is to compute the distortion center, optical centers (d_i) and opening angles (r_i/d_i) of the cones associated with each of the distortion circles. In other words we are interested in computing the projection rays associated with every pixel in a radially symmetric image.

10.3 Algorithm: Factorization Framework

In the following we give the derivation of the self-calibration algorithm. The main idea is based on the popular triangulation constraint on the intersection of projection rays corresponding to matching pixels. In the beginning we assume that the distortion center is known. Later in section 10.3.3, we provide a technique to compute the distortion center. We define the first camera coordinate system as shown in Figure 3.11(b). Usually the optical center will be chosen for the origin. Since we have more than one optical center, corresponding to different viewing cones, we use a different coordinate system as shown in Figure 3.11. We chose the xy plane to intersect all the viewing cones and we parameterize the individual r_i on it. The optical axis of the camera is same as the z axis. The individual optical centers of the cones lie at a distance of d_i from the origin. As shown in Figure 10.1 let $(\mathsf{R}, \mathbf{t})$ be the motion of the camera. Let $\mathbf{p}_1 = (\breve{r}_1 cos(\alpha), \breve{r}_1 sin(\alpha))$ and $\mathbf{p}_2 = (\breve{r}_2 cos(\beta), \breve{r}_2 sin(\beta))$ be two matching image pixels. We only consider pixels lying on different distortion circles. The corresponding viewing cones are parameterized by (r_1, d_1) and (r_2, d_2) respectively. The projection rays can be represented by the 3D points $(r_1 cos(\alpha), r_1 sin(\alpha), 0)$ and $(r_2 cos(\beta), r_2 sin(\beta), 0)$ along with their respective optical centers in the local coordinate system of two cameras. Let the equation of the planar scene be given by: $Ax + By + Cz = 1$ in the coordinate system of the second camera.

The parameters of the plane (A, B, C), motion $(\mathsf{R}, \mathbf{t})$, two focal lengths $(d_1/r_1, d_2/r_2)$ and optical centers (d_1, d_2) are all unknown. The only information we have is the fact that the matching projection rays intersect at a point on the plane. We use this constraint to solve the unknowns and eventually calibrate the camera. The second projection ray starts from $(0, 0, d_2)$ and passes through the point $(r_2 cos(\beta), r_2 sin(\beta), 0)$. Now we compute the point of intersection of this ray with the plane, represented by the equation $Ax + By + Cz = 1$. We obtain the intersection point $\breve{\mathsf{E}}$ in the second coordinate system:

$$\breve{\mathsf{E}} = \begin{pmatrix} r_2 cos(\beta)(1 - Cd_2) \\ r_2 cos(\beta)(1 - Cd_2) \\ d_2(Ar_2 cos(\beta) + Br_2 sin(\beta) - 1) \\ Ar_2 cos(\beta) + Br_2 sin(\beta) - Cd_2 \end{pmatrix}$$

On expressing $\breve{\mathsf{E}}$ in the first coordinate frame we obtain E (cf.Figure 10.1).

$$\mathsf{E} = \begin{pmatrix} \mathsf{R} & \mathbf{t} \\ 0^T & 1 \end{pmatrix} \begin{pmatrix} r_2 cos(\beta)(1 - Cd_2) \\ r_2 cos(\beta)(1 - Cd_2) \\ d_2(Ar_2 cos(\beta) + Br_2 sin(\beta) - 1) \\ Ar_2 cos(\beta) + Br_2 sin(\beta) - Cd_2 \end{pmatrix}$$

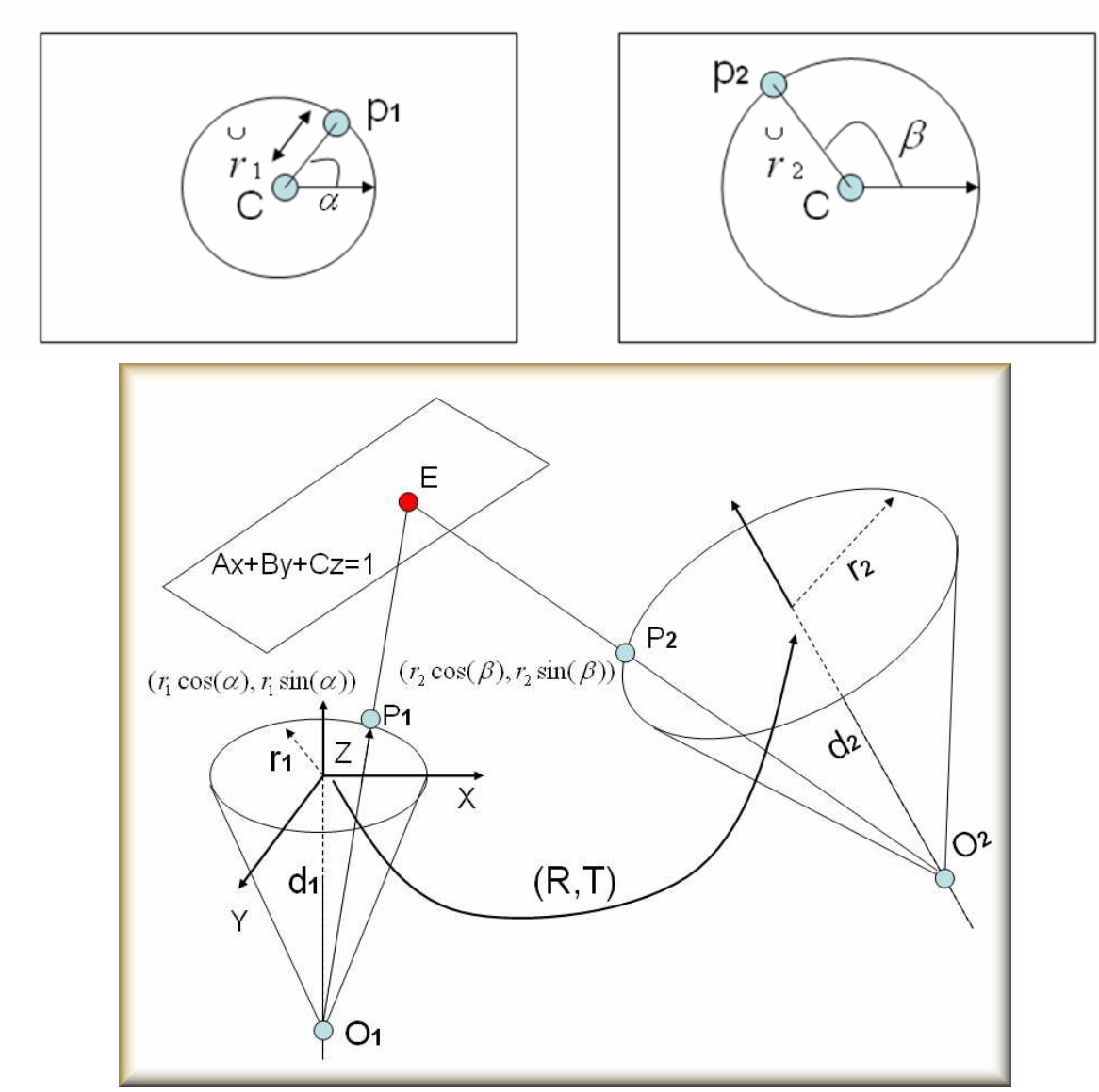


Figure 10.1: Top: Two matching pixels $\mathbf{p}_1 = (\breve{r}_1 cos(\alpha), \breve{r}_2 sin(\alpha))$ and $\mathbf{p}_2 = (\breve{r}_2 cos(\beta), \breve{r}_2 sin(\beta))$ in the images. Bottom: Triangulation of the corresponding rays $\mathbf{O}_1\mathbf{P}_1$ and $\mathbf{O}_2\mathbf{P}_2$, coming from two different cameras, intersecting at a point $\mathbf{E}$ on the plane.

We introduce a set of intermediate variables, coupling camera motion and plane coefficients:

$$\begin{aligned}
a_{1,1} &= R_{1,1} + At_1 \\
a_{1,2} &= R_{1,2} + Bt_1 \\
a_{2,1} &= R_{2,1} + At_2 \\
a_{2,2} &= R_{2,2} + Bt_2 \\
a_{3,1} &= R_{3,1} + Ct_3 \\
a_{3,2} &= R_{3,2} + Ct_3 \\
b_{1,1} &= -CR_{1,1} + AR_{1,3} \\
b_{1,2} &= -CR_{1,2} + BR_{1,3} \\
b_{1,3} &= -R_{1,3} - Ct_1 \\
b_{2,1} &= -CR_{2,1} + AR_{2,3} \\
b_{2,2} &= -CR_{2,2} + BR_{2,3} \\
b_{2,3} &= -R_{2,3} - Ct_2 \\
b_{3,1} &= -CR_{3,1} + AR_{3,3} \\
b_{3,2} &= -CR_{3,2} + BR_{3,3} \\
b_{3,3} &= -R_{3,3} - Ct_3
\end{aligned}$$

Using the above notations we denote E as follows:

$$\begin{pmatrix}
r_2cos(\beta)(a_{1,1} + d_2b_{1,1}) + r_2sin(\beta)(a_{1,2} + d_2b_{1,2}) + b_{1,3}d_2 \\
r_2cos(\beta)(a_{2,1} + d_2b_{2,1}) + r_2sin(\beta)(a_{2,2} + d_2b_{2,2}) + b_{2,3}d_2 \\
r_2cos(\beta)(a_{3,1} + d_2b_{3,1}) + r_2sin(\beta)(a_{3,2} + d_2b_{3,2}) + b_{3,3}d_2 \\
Ar_2cos(\beta) + Br_2sin(\beta) - Cd_2
\end{pmatrix} \tag{10.1}$$

This point must lie on the projection ray associated with pixel $\mathbf{p}_1$ in the first view, i.e. it must be collinear with the optical center $O_1 = (0, 0, d_1)$, and the point $\mathbf{P}_1 = (r_1cos(\alpha), r_1sin(\alpha), 0)$ (cf. figure 10.1). Collinearity of three points means that when stacking their homogeneous coordinates in a 4×3 matrix, the determinants of all four sub-matrices of size 3×3 must be zero. In our case, one of them is always zero and the other three give conditions that are algebraically dependent. One of them are thus used in this work, cf. Equation(10.2). Note that the following equation is independent of r_1 and d_1.

$$cos(\alpha)cos(\beta)((\frac{r_2}{d_2})a_{2,1} + r_2b_{2,1}) + cos(\alpha)sin(\beta)((\frac{r_2}{d_2})a_{2,2} + r_2b_{2,2}) - sin(\alpha)cos(\beta)((\frac{r_2}{d_2})a_{1,1} + r_2b_{1,1}) -$$

$$sin(\alpha)sin(\beta)((\frac{r_2}{d_2})a_{1,2} + r_2b_{1,2}) + cos(\alpha)b_{2,3} - sin(\alpha)b_{1,3} = 0 \tag{10.2}$$

Let us consider a specific distortion circle in the second camera, whose viewing cone is parameterized by (r_2, d_2). We select several pixels in this distortion circle and consider a set of matches :

$$(\breve{r}_2cos(\beta_i), \breve{r}_2sin(\beta_i)), (\breve{r}_1^i cos(\alpha_i), \breve{r}_1^i sin(\alpha_i)), i = 1..5$$

Note that all the $\breve{r}_1^i$ are different. All the pixels $(\breve{r}_2cos(\beta_i), \breve{r}_2sin(\beta_i)), i = 1..5$, correspond to the same distortion circle with fixed r_2 and d_2. Equation(10.2) does not depend on r_1 and d_1. Thus, essentially, our system contains different values of α_i and β_i, in addition to r_2,d_2,a and b. Note that α_i and β_i are already

known. Let us denote $cos(\alpha)$ by $c\alpha$ and $sin(\alpha)$ by $s\alpha$. Using this notation we obtain the following linear system from equation(10.2).

$$\Gamma = \begin{pmatrix} c\alpha_1 c\beta_1 & -c\alpha_1 s\beta_1 & -s\alpha_1 c\beta_1 & s\alpha_1 s\beta_1 & c\alpha_1 & -s\alpha_1 \\ c\alpha_2 c\beta_2 & -c\alpha_2 s\beta_2 & -s\alpha_2 c\beta_2 & s\alpha_2 s\beta_2 & c\alpha_2 & -s\alpha_2 \\ c\alpha_3 c\beta_3 & -c\alpha_3 s\beta_3 & -s\alpha_3 c\beta_3 & s\alpha_3 s\beta_3 & c\alpha_3 & -s\alpha_3 \\ c\alpha_4 c\beta_4 & -c\alpha_4 s\beta_4 & -s\alpha_4 c\beta_4 & s\alpha_4 s\beta_4 & c\alpha_4 & -s\alpha_4 \\ c\alpha_5 c\beta_5 & -c\alpha_5 s\beta_5 & -s\alpha_5 c\beta_5 & s\alpha_5 s\beta_5 & c\alpha_5 & -s\alpha_5 \end{pmatrix} \tag{10.3}$$

$$\Sigma = \begin{pmatrix} ((\frac{r_2}{d_2})a_{2,1} + r_2 b_{2,1}) \\ ((\frac{r_2}{d_2})a_{2,2} + r_2 b_{2,2}) \\ ((\frac{r_2}{d_2})a_{1,1} + r_2 b_{1,1}) \\ ((\frac{r_2}{d_2})a_{1,2} + r_2 b_{1,2}) \\ b_{2,3} \\ b_{1,3} \end{pmatrix} \tag{10.4}$$

$$\Gamma_{5\times 6} \times \Sigma_{6\times 1} = \mathsf{0} \tag{10.5}$$

As described above, the matrix $\Gamma_{5\times 6}$ is completely known because it involves α_i and β_i for the matches. On solving the above homogeneous linear system we compute Σ and thereby the values of $((\frac{r_2}{d})a_{2,1} + r_2 b_{2,1})$, $((\frac{r_2}{d})a_{2,2} + r_2 b_{2,2})$, $((\frac{r_2}{d})a_{1,1} + r_2 b_{1,1})$, $((\frac{r_2}{d})a_{1,2} + r_2 b_{1,2})$, $b_{2,3}$ and $b_{1,3}$ up to a scale. The sixth variable $b_{1,3}$, as described earlier as $(-R_{13} - Ct_1)$, depends only on pose $(\mathsf{R}, \mathbf{t})$ and the plane parameter C, i.e. $b_{1,3}$ is independent of the calibration parameters. Thus we fix this value to a constant for all the radii to obtain their corresponding equations in the same scale. We rewrite this information in the following form where the k_i are known.

$$\begin{pmatrix} (\frac{r}{d}) & r \end{pmatrix} \begin{pmatrix} a_{2,1} & a_{2,2} & a_{1,1} & a_{1,2} \\ b_{2,1} & b_{2,2} & b_{1,1} & b_{1,2} \end{pmatrix} = \begin{pmatrix} k_1 & k_2 & k_3 & k_4 \end{pmatrix}$$

We iterate the above process for different distortion circles, having different radii r_2^i, and obtain the following.

$$\begin{pmatrix} (\frac{r_2^1}{d_1}) & r_2^1 \\ (\frac{r_2^2}{d_2}) & r_2^2 \\ . & \\ (\frac{r_2^n}{d_n}) & r_2^n \end{pmatrix} \begin{pmatrix} a_{2,1} & a_{2,2} & a_{1,1} & a_{1,2} \\ b_{2,1} & b_{2,2} & b_{1,1} & b_{1,2} \end{pmatrix} = \begin{pmatrix} k_{11} & k_{12} & k_{13} & k_{14} \\ k_{21} & k_{22} & k_{23} & k_{24} \\ . & & & \\ k_{n1} & k_{n2} & k_{n3} & k_{n4} \end{pmatrix} \tag{10.6}$$

$$\mathsf{L}_{n\times 2}\mathsf{M}_{2\times 4} = \mathsf{K}_{n\times 4} \tag{10.7}$$

The matrix K is known up to a scale and both L and M are unknown. L and M are by construction both of rank 2. We use singular value decomposition to compute the factors of K.

$$\mathsf{K}_{n\times 4} = \mathsf{U}_{n\times 4}\mathsf{S}_{4\times 4}\mathsf{V}^T_{4\times 4}$$

Since K is of rank 2, S must have only two nonzero singular values. Thus S must be of the following form.

$$\mathsf{S} = \begin{pmatrix} l_1 & 0 & 0 & 0 \\ 0 & l_2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}_{4\times 4}$$

We remove the last two rows we obtain $\breve{\mathsf{U}}_{n\times 2}$, $\breve{\mathsf{S}}_{2\times 2}$ and $\breve{\mathsf{V}}_{s\times 2}$ from the matrices U, S and V respectively.

$$\mathsf{K} = \breve{\mathsf{U}}_{n\times 2}\breve{\mathsf{S}}_{2\times 2}\breve{\mathsf{V}}^{T}_{2\times 2}$$

The matrices L and M can be computed up to 4 unknowns ($\mathsf{X}_{2\times 2}$) as given below:

$$\mathsf{L} = \mathsf{U}\mathsf{X}_{2\times 2}, \quad \mathsf{M} = \mathsf{X}^{-1}_{2\times 2}\mathsf{V}$$

The framework is very similar to the one used in [115]. Without loss of generality we can fix the distance of the optical center of the first distortion circle to be 1 unit from the origin. This implies that we can reduce one degree of freedom in X. We use an approximate technique to compute the remaining three variables. This is done using two assumptions. First we assume that the ratio of the radii of very small distortion circles is equal to the ratio of the radii of their corresponding viewing cones. The second assumption is that the optical centers of the viewing cones of very small distortion circles coincide.

It is important to understand the significance of the above parameters contained in X. They represent the scale factor in the computation of the focal length and optical centers. In addition, there is a translational ambiguity in the estimation of the optical centers along the optical axis. The three unknowns can also be computed analytically using the additional images. However for practical radially symmetric non-central cameras (such as spherical catadioptric cameras) the above approximation works fine. The interesting part of this framework is that the internal and external parameters are disambiguated in the calibration process. Once we compute X we can compute L and M uniquely. The computation of L provide us the necessary calibration information. As given in equation(10.7), we obtain r_i and d_i on the computation of L. This will provide us the cone of rays for every distortion circle and eventually the calibration in the form of mapping between image pixels and projection rays.

10.3.1 Central cameras

In the case of the central cameras, all distortion circles are associated with the same optical center. This implies that there will be a single d for all viewing cones. We can fix this to be unity. The equation(10.2) simplifies itself to the following form.

$$\begin{pmatrix} r_1 \\ r_2 \\ . \\ r_n \end{pmatrix} \begin{pmatrix} a_{2,1}+b_{2,1} & a_{2,2}+b_{2,2} & a_{1,1}+b_{1,1} & a_{1,2}+b_{1,2} \end{pmatrix}$$

$$= \begin{pmatrix} k_{11} & k_{12} & k_{13} & k_{14} \\ k_{21} & k_{22} & k_{23} & k_{24} \\ . & & & \\ k_{n1} & k_{n2} & k_{n3} & k_{n4} \end{pmatrix}$$

We do a singular value decomposition of K to obtain U, S and V. The rank of K is unity. This implies that the individual radii can be computed up to a common scale. By fixing the radius of one distortion circle to unity we can compute the other radii. The overall scale can be computed using more images. We briefly explain this procedure. We capture two images of a general 3D scene. We compute a 3×3 fundamental matrix with respect to the projection rays. This is equivalent to computing the fundamental matrix between the distortion corrected images. From the fundamental matrix we can extract the focal length using an algorithm developed for perspective images [102].

10.3.2 Geometrical interpretation

The constraint given in equation(10.2) is extremely rich. For example we can consider the cases where $\alpha = \frac{\pi}{2}$. The corresponding equation is given below.

$$(a_{1,1} + b_{1,1})r_2 cos\beta + (a_{1,2} + b_{1,2})r_2 sin\beta + b_{1,3} = 0$$

This considers all possible matches for different values of r_1, r_2 and β. In the first image this constraint corresponds to considering all pixels lying on the negative y axis (see Figure 10.2). On the second image this refers to the equation of a line. This implies that in a radially symmetric central camera the mapping of a radial line passing through the distortion center is a line. Note that in Figure 10.2 the mapping in the second image is a distorted line. This is because the distortion center was not fixed correctly. Thus by imposing this constraint we can check for the correctness of the distortion center. In the next subsection we will see how to use such constraints to compute the distortion center. In the non-central case we have the following.

$$cos\beta(r_2 a_{1,1} + r_2 d_2 b_{1,1}) + sin\beta(r_2 a_{1,2} + r_2 d_2 b_{1,2}) + b_{1,3} d_2 = 0$$

The above equation again represents a line in the first image. However it is not mapped to a line in the second image. Similarly we can obtain several constraints corresponding to other scenarios ($\alpha = 0, \beta = 0, \alpha\beta = 0, \alpha\beta = 1, r_1 = r_2$, etc). We show some of these constraints which are obtained from equation(10.2) in Table 10.1 for simple cases in a central model.

Cases	Equation 10.2
$r_2 = 0$	$b_{2,3} cos\alpha + b_{1,3} sin\alpha = 0$
$\alpha = 0$	$r_2 cos\beta(a_{2,1} + b_{2,1}) + r_2 sin\beta(a_{2,2} + b_{2,2}) + b_{2,3} = 0$
$\alpha = \frac{\pi}{2}$	$(a_{1,1} + b_{1,1})r_2 cos\beta + (a_{1,2} + b_{1,2})r_2 + b_{1,3} = 0$
$\beta = 0$	$cos\alpha(r_2(a_{2,1} + b_{2,1}) + b_{2,3}) + sin\alpha(r_2(a_{1,1} + b_{1,1}) + b_{1,3}) = 0$
$\beta = \frac{\pi}{2}$	$cos\alpha(r_2(a_{2,2} + b_{2,2}) + b_{2,3}) + sin\alpha(r_2(a_{1,2} + b_{1,2}) + b_{1,3}) = 0$

Table 10.1: Some constraints obtained from Equation 10.2 for specific values of α, β and r_2

.

10.3.3 Computation of the distortion center

In this section we will briefly explain our algorithm to verify the correctness of the distortion center. We fix the distortion center at the center of the image for fisheye and pinhole images. In the case of catadioptric cameras with full image of the mirror boundary, observed as a conic, we can initialize the distortion center at the center of the conic. From the general equation 10.2 we learned how to compute r and d. We validate the correctness of the matches by substituting in the general equation(10.2). In addition by expanding another

3×3 submatrix of the matrix(10.1) we obtain the following equation.

$$\begin{aligned} cos(\alpha)cos(\beta)((\frac{r_1}{d_1})(\frac{r_2}{d_2})a_{3,1} + (\frac{r_1}{d_1})r_2 b_{3,1} + r_1(\frac{r_2}{d_2})c_{1,1}) + \\ cos(\alpha)sin(\beta)((\frac{r_1}{d_1})(\frac{r_2}{d_2})a_{3,2} + (\frac{r_1}{d_1})r_2 b_{3,2} + r_1(\frac{r_2}{d_2})c_{1,2}) + \\ cos(\alpha)((\frac{r_1}{d_1})b_{3,3} + r_1 c_{1,3}) + \\ cos(\beta)((\frac{r_2}{d_2})a_{1,1} + r_2 b_{1,1}) + \\ sin(\beta)((\frac{r_2}{d_2})a_{1,2} + r_2 b_{1,2}) + b_{1,3} = 0 \end{aligned}$$

In addition to equation(10.2) we use the above equation in checking the correctness of the distortion center. We compute the error we obtain on using the different solutions of r_1, d_1 and r_2, d_2. We compute the overall error using all the matches. The correct distortion center will be the point which minimizes this error. This technique can also be used to compute the distortion center starting from the center of the image and computing the error at various points surrounding it.

10.4 Variants

10.4.1 Non-planar scenes

Using the earlier parameterization the intersection point can be computed on the first and the second rays in the first camera coordinate system:

$$\mathbf{P}_1 = \begin{pmatrix} \lambda_1 r_1 cos\alpha \\ \lambda_1 r_1 sin\alpha \\ d1 - \lambda_1 d_1 \\ 1 \end{pmatrix} \quad \mathbf{P}_2 = \begin{pmatrix} \mathsf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{pmatrix} \begin{pmatrix} \lambda_2 r_2 cos\beta \\ \lambda_2 r_2 sin\beta \\ d2 - \lambda_2 d_2 \\ 1 \end{pmatrix}$$

λ_1 and λ_2 are two parameters used in locating the intersection point in the two rays. By matching $\mathbf{P}_1$ and $\mathbf{P}_2$ we obtain three equations. Two equations are sufficient to eliminate λ_1 and λ_2. The third equation could be used to solve for the calibration parameters.

10.4.2 Non-unit aspect ratio

In the presence of non-unit aspect ratio, γ, we can model the projection ray of every pixel $\mathbf{p}(\breve{r}cos\theta, \breve{r}sin\theta)$ to the projection ray passing through $(0, 0, d_1)$ and $(rcos\theta, \gamma r sin\theta, 0)$. Similar to the earlier scenario we can again construct a factorization framework to calibrate the camera.

$$\begin{pmatrix} (\frac{r_1}{d_1}) & r_1 \\ (\frac{r_2}{d_2}) & r_2 \\ . & \\ (\frac{r_n}{d_n}) & r_n \end{pmatrix} \begin{pmatrix} a_{2,1} & \gamma a_{2,2} & \gamma a_{1,1} & \gamma^2 a_{1,2} \\ b_{2,1} & \gamma b_{2,2} & \gamma b_{1,1} & \gamma^2 b_{1,2} \end{pmatrix} = \begin{pmatrix} k_{11} & k_{12} & k_{13} & k_{14} \\ k_{21} & k_{22} & k_{23} & k_{24} \\ . & & & \\ k_{n1} & k_{n2} & k_{n3} & k_{n4} \end{pmatrix} \tag{10.8}$$

In the case of known aspect ratio the problem is extremely simple. In the case of unknown aspect ratio we can compute it along with the pose. We look at the non-unit aspect ratio in a different way. Every pixel $\mathbf{p}(\breve{r}_1 cos(\theta), \breve{r}_1 sin(\theta))$ maps to a projection ray passing through $(0, 0, d1)$ and $(r_1 cos(\theta), \gamma r_1 sin(\theta))$. An easier way to understand this would be to think of every circle in the image to map to a cone which crushed along one axis (distorted cone with an ellipsoidal base). By doing this we avoid the aspect ratio problem in matching features in the image planes.

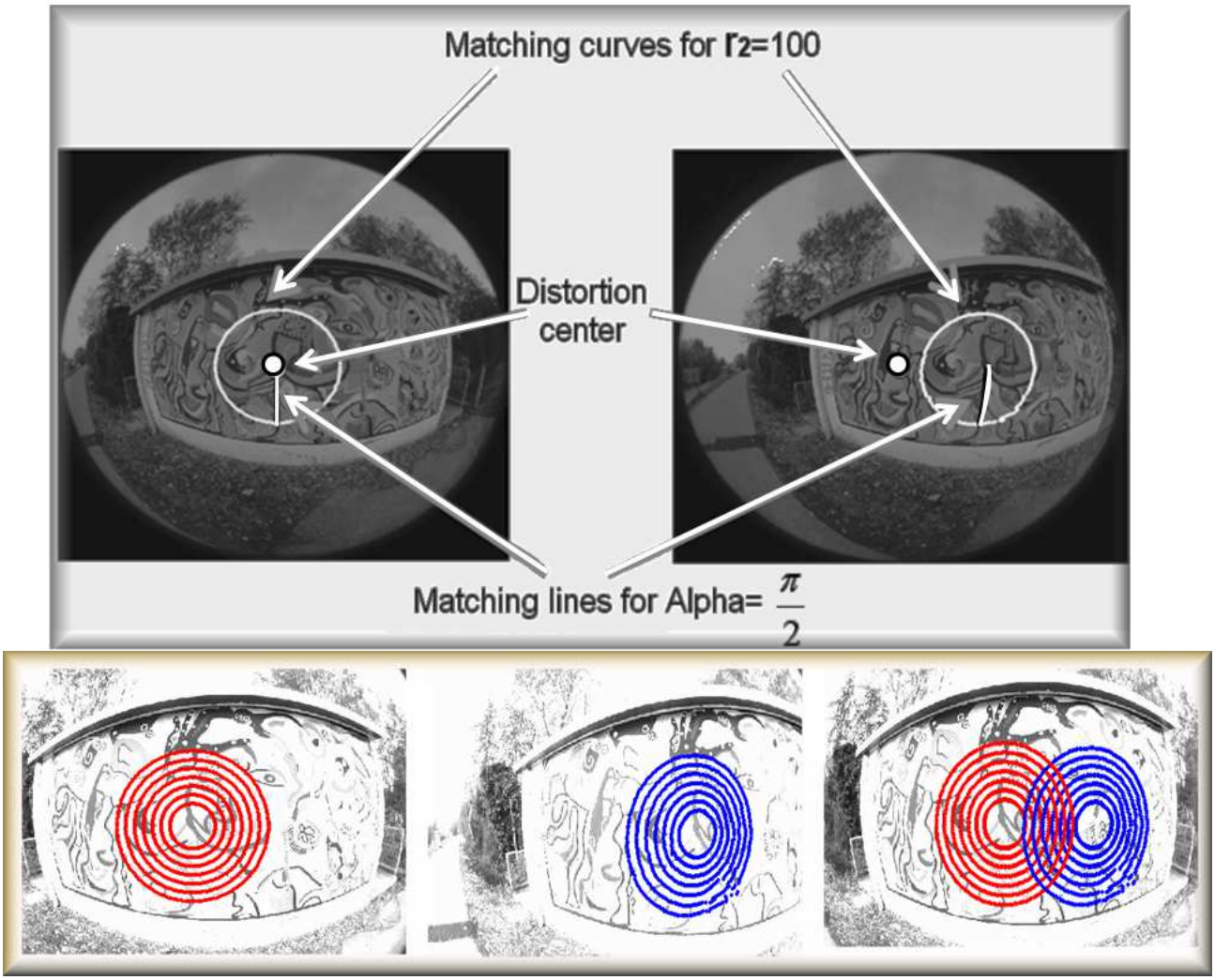


Figure 10.2: Top: Two planar scenes captured by fisheye lenses. In polar coordinates the matches are represented by $(\breve{r}_1, \alpha)$ and $(\breve{r}_2, \beta)$. We show two matching curves under the constraints of $r_2 = 100$ and $\alpha = \frac{\pi}{2}$ respectively. Bottom: left: The pixels corresponding to specific values of r. middle: The matching pixels in the second image. right: We show the pixels from both the images. Note that the curves need not intersect. Also note the matching pixels do not form straight lines (refer to their relationship in Equation(10.2).

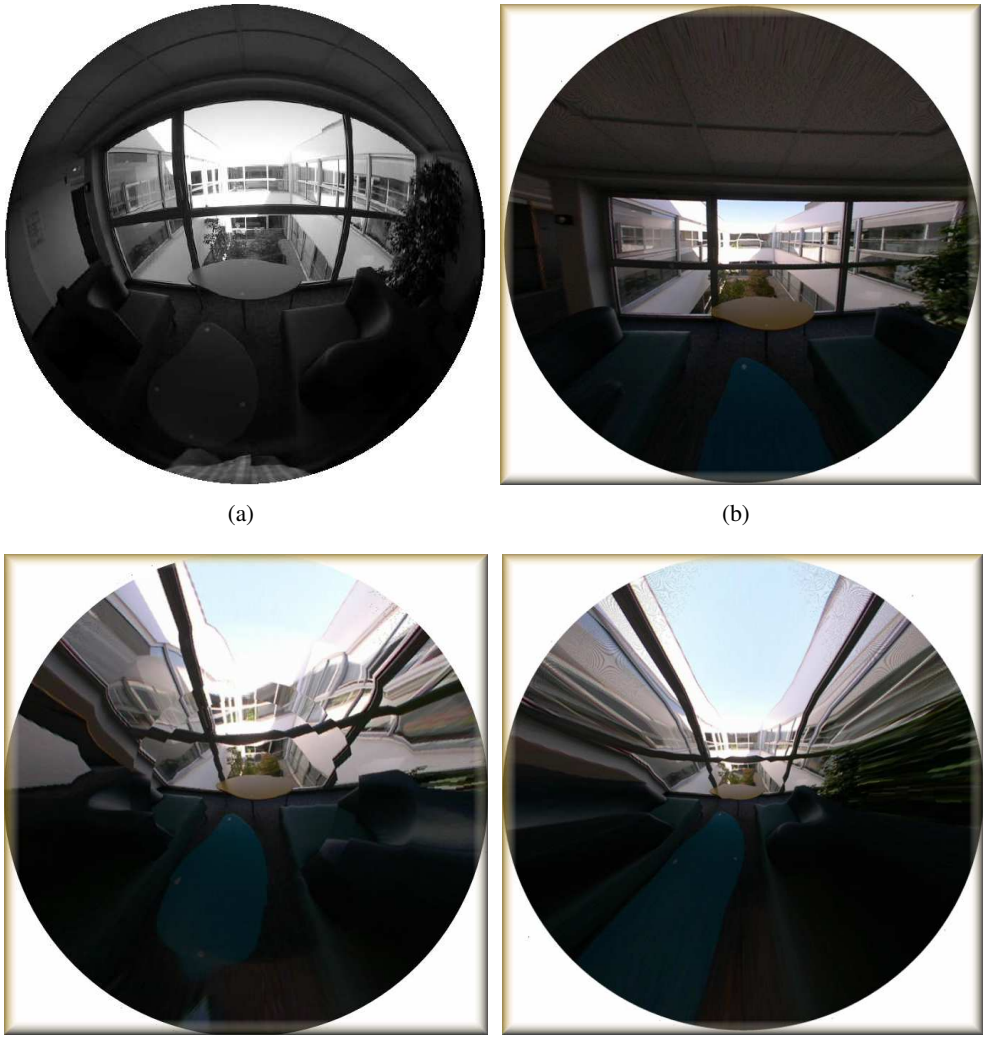
10.4.3 Multi-view relations

The factorization framework is easily expendable to multiple views. The constraints from multiple views can be used in the same framework as follows for central cameras. The extension for noncentral cameras is also straightforward.

$$\begin{pmatrix} r_1 \\ r_2 \\ . \\ r_n \end{pmatrix} \begin{pmatrix} a_{2,1}+b_{2,1} & a_{2,2}+b_{2,2} & a_{1,1}+b_{1,1} & \cdots & \\ & a_{1,2}+b_{1,2} & a'_{2,1}+b'_{2,1} & \cdots & a'_{2,2}+b'_{2,2} \end{pmatrix}$$

$$= \begin{pmatrix} k_{11} & k_{12} & k_{13} & k_{14} & k'_{11} & k'_{12} & k'_{13} & k'_{14} & .. \\ k_{21} & k_{22} & k_{23} & k_{24} & k'_{21} & k'_{22} & k'_{23} & k'_{24} & .. \\ . & & & & & & & & \\ k_{n1} & k_{n2} & k_{n3} & k_{n4} & k'_{n1} & k'_{n2} & k'_{n3} & k'_{n4} & .. \end{pmatrix}$$

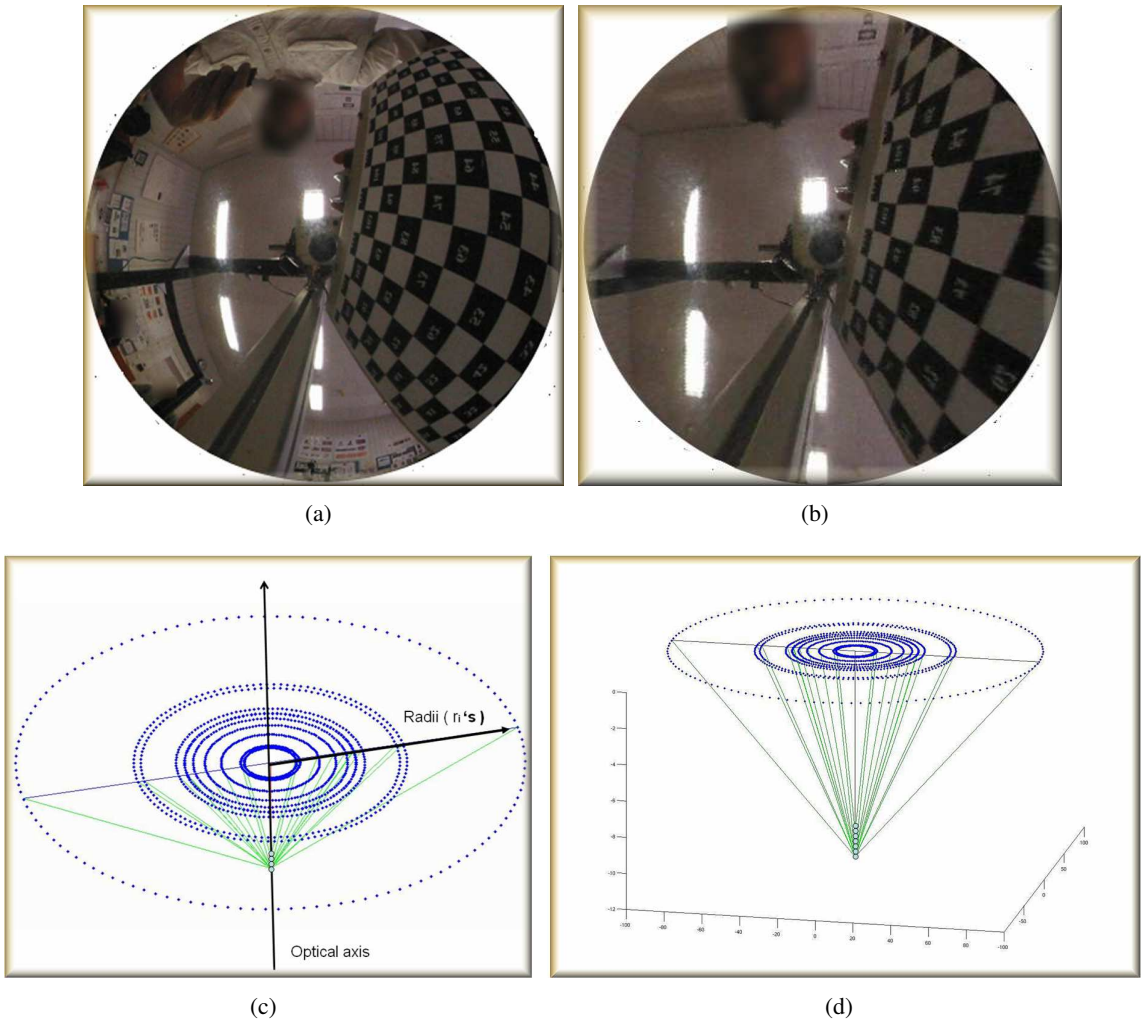
$a_{i,j}$,$b_{i,j}$, k_{ij} are associated with the first and second views. We extend this with $a'_{i,j}$,$b'_{i,j}$, k'_{ij}, which can be associated with second and third or any other two views.



(a) (b)

(c) (d)

Figure 10.3: Distortion correction (perspective view synthesis). (a) Original fisheye image (b) Using the distortion center at the correct location. (c) The distortion center is at an offset of 25 units from the correct distortion center. (d) The distortion center is at a distance of 50 units from the correct position. The image size is 1024 by 768.



(a) (b)

(c) (d)

Figure 10.4: (a) Image taken by a spherical catadioptric camera. (b) Distortion correction. Note that the camera model is non-central and exact distortion correction is not possible. We compute an approximate center close to all the projection rays and perform distortion correction. (c) and (d) show the reconstructed projection rays. Note that we do not show the distortion correction for the complete image. This is because a single image pair of the plane was not sufficient to calibrate the complete image. By using more than two images we can calibrate the whole image.

10.5 Experiments

10.5.1 Cameras

We tested our algorithms on three different cameras: Nikon coolpix 5400, E8 fisheye lens with a field of angle of 182 by 360 degrees, a non-central catadioptric camera with a spherical mirror. We modeled all three cameras as radially symmetric.

10.5.2 Dense matching for planar scenes

There are plenty of planar scenes in man made environments such as walls, posters, etc. In addition the matching process is simplified in planar scenarios. For perspective cameras points undergo a homography

transformation between two views of a planar scene. This allows us to obtain dense matching from an initial set of sparse matches. For the case of fisheye lenses we used Harris corner detection to detect features on a plane. Then image matching is done using a cross-correlation based approach. This worked well for planes which are not very close to the fisheye lens. In Figure 10.2 we detected 270 feature matches. Using them we can interpolate for other matches. We also used planar boards with black dots to simplify the matching process. These objects were used for catadioptric cameras with spherical mirrors.

10.5.3 Distortion correction

We used distortion correction to study the accuracy of our algorithm. In Figure 10.3 we show an undistorted original fisheye image and the distortion corrected images. Once we estimate the parameters of the viewing cone the process of distortion correction is very simple: every pixel with coordinates $(\check{r}cos\theta, \check{r}sin\theta)$ will be moved to $(rcos\theta, rsin\theta)$ where r is the radius of the associated viewing cone. Note that this correction can not be applied for non-central images. In the central cases the rank of matrix K (refer Equation(10.7) must be 1. However in general the matrix was found to have a higher rank. So RANSAC must be used to improve the stability of the algorithm. It is applied in selecting five correspondences in every distortion circle in the second image.

Sensitivity of the distortion center: We estimate the distortion center as explained in section 10.3.3. In Figure 10.3(c) and (d) we show the distortion correction when the distortion center was placed at a distance of 25 and 50 pixels respectively. The problems that appear in the two images are very common if the distortion center is not fixed at the correct location. In Figure 10.3(a) the relative radii are not computed correctly. This can be seen as artifacts at concentric circles. In the second case (Figure 10.3(b)) the outer radii explode. This kind of errors happen in wide angle scenarios. Both these errors don't take place on using RANSAC and the choice of correct distortion center.

10.5.4 Non-central model

We used a catadioptric camera with a spherical mirror to study the non-central model. As per the theory we obtained a K matrix of rank 2 (refer to Equation(10.7)). We did a singular value decomposition to get the first two columns of the U matrix. However as we studied earlier the solution is obtained in terms of three variables.

In Figure 10.4 (c) and (d) we show the reconstructed projection rays. Note that the camera is non-central and all the projection rays intersect at the optical axis. On using the exact algorithm to compute the re-projection rays we expect the rays to correlate well with the actual caustics of this catadioptric configuration. The optical centers of all the projection rays are already quite close, which is in fact the case for spherical catadioptric configurations. We also tested distortion correction of an image taken by a spherical catadioptric camera (see Figure 10.4). Since the model is not a central one we first compute an approximate center for all the projection rays. Considering the fact that we are using a non-central camera for distortion correction the algorithm performs reasonably well. The distortions in the calibration grid, corners of the wall, etc. are removed (see Figure 10.4(b)). These are only preliminary results in our approach. We intend to use non-linear optimization to improve the accuracy.

10.6 Conclusions

We propose a simple method to solve the self-calibration problem for radially symmetric cameras, which can be both, central or non-central. The method uses images of a planar sccen but the theoretical formulation is sufficiently general for extending to non-planar scenes and cameras with non-unit and unknown aspect

ratio. The initial experimental results are very promising. In future we plan to study more non-central configurations and non-planar scenes.

Chapter 11

Conclusions

11.1 Contributions

This thesis focuses on various theoretical and practical issues related to a generic imaging model: calibration, self-calibration and structure-from-motion analysis. The primary contributions are listed below.

11.1.1 Generic calibration

A generic imaging model is introduced and a generic calibration algorithm is proposed and shown to practically calibrate a wide variety of cameras using calibration grids. Algorithms are developed to identify the appropriate model, central or non-central, for slightly non-central cameras. The theory of an intermediate class of cameras called axial cameras, where all projection rays intersect a single line is space, has been studied and a calibration algorithm has been developed.

11.1.2 Generic Structure-from-Motion

A generic approach for structure-from-motion, that works for any camera or mixture of cameras that fall into the generic imaging model has been developed. Our approach includes methods for motion and pose estimation, 3D point triangulation and bundle adjustment. Promising results have been obtained for different image sets, obtained with three different cameras: pinhole, omni-directional (fisheye) and a stereo system.

11.1.3 Generic self-calibration

A generic self-calibration problem has been solved using 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. A simple method to solve the self-calibration problem for radially symmetric cameras, which can be both, central or non-central has been proposed. The theoretical formulation is sufficiently general for extending to non-planar scenes and cameras with non-unit and unknown aspect ratio.

11.2 Possible extensions

11.2.1 Generic Structure from Motion

The 3D reconstruction problem has been solved for pinhole cameras by several researchers and many enhancements have been proposed. Though generic SfM has more potential to provide better results a detailed investigation is necessary for the generic components of this algorithm: feature detection (especially for

omnidirectional images), image matching for different camera images, and generic motion estimation for practical applications.

11.2.2 Critical motion sequences for generic calibration

It is important to study the motion sequences which might lead to ambiguous generic calibration for generic imaging models, similar to the study on pinhole cameras [98].

11.3 Challenging open problems

11.3.1 Generic self-calibration

Very few works have been done in generic self-calibration. All these works, including our work, rely on central cameras and restricted motion sequences. For people who love challenging problems we propose the problem of generic self-calibration in general motion and non-central scenarios. We can increase the complexity further by considering the case of varying parameters.

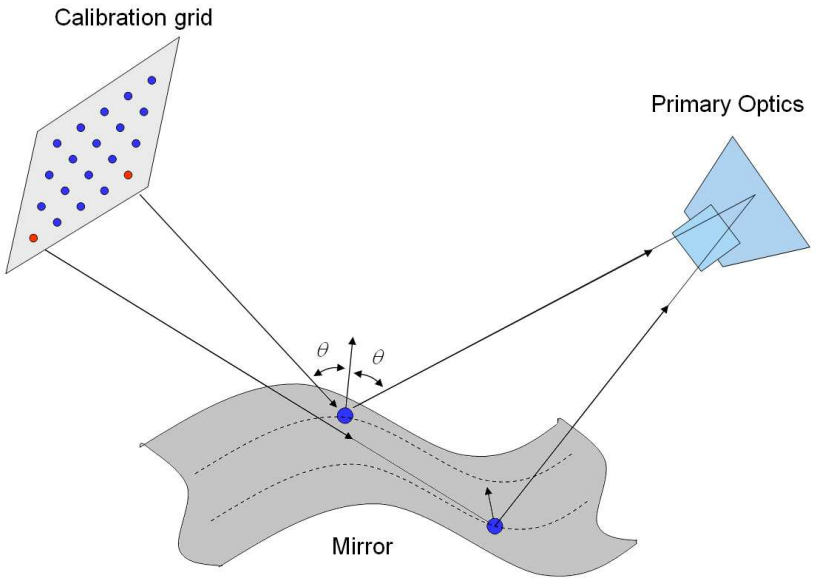


Figure 11.1: Catadioptric system with a primary optics (lens) and a mirror

In general a practical setup for a catadioptric camera is hard to realize and maintain because the camera and the mirror have to be fixed at all times. We believe that most or all of the previous works in calibrating a catadioptric camera have assumed fixed parameters. By relaxing this constraint we make the catadioptric systems more practical and enhance their capabilities. Calibration algorithms for such scenarios will also enable us to accurately model the structure of the human eye. A general catadioptric camera, as shown in Figure 11.1, consists of primary optics (lens) and a mirror. The parameters of the camera which can vary are given below.

- Internal parameters of the camera
- Pose of the mirror w.r.t camera
- Shape of the mirror (eye based catadioptric camera, micromirror array, etc)

The initial solutions for the above parameters will be computed with the help of calibration grids. Updating them seems to be extremely hard and even a theoretical solution to this problem seems to be significantly complex.

11.3.2 Generic view-planning for accurate 3D reconstruction

We are interested in designing optimal camera configurations, using different kinds of cameras, for accurate 3D reconstruction.

Basic criteria

Figure 11.2: Left: Two projection rays intersecting to reconstruct a 3D point $\mathbf{P}$. Right: Two 3D points $\mathbf{P}1$ and $\mathbf{P}2$ are lying between two consecutive projection rays.

On the left side of Figure 11.2 we observe two rays starting from $\mathbf{C}1$ and $\mathbf{C}2$. They intersect at $\mathbf{P}$. The accuracy in the computation of $\mathbf{P}$ will depend on the angle θ between the two rays. The uncertainty will be maximum when $\theta = 0$ and minimum when $\theta = 90$.

On the right side of Figure 11.2 we see two 3D points $\mathbf{P}1$ and $\mathbf{P}2$ lying between two consecutive rays of a camera. In order to reconstruct the points accurately the resolution and location of a camera should be such that no two critical 3D points lie between two consecutive rays of a camera.

Preliminary study

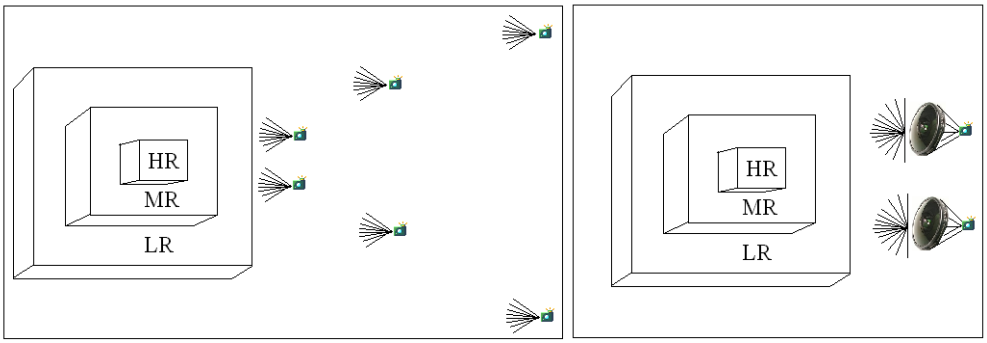


Figure 11.3: 3D scene having 3 levels of resolutions (HR, MR, LR - higher, medium and lower) with two camera configurations - 6 pinhole cameras(left), 2 fisheye cameras(right).

In a generic framework 3D reconstruction is a simple ray intersection problem. Thus the accuracy of the reconstructed 3D model essentially depends on actual structure of the scene, camera locations and the nature

of the cameras. Based on these factors we have identified three fundamental questions to clearly describe our problem.

- When the camera models and camera locations are known, what is the best possible 3D reconstruction that can be obtained.
- When the 3D structure and camera models are known, what is the camera location which will give the best 3D reconstruction.
- What is the best camera configuration, in terms of camera models and camera locations, for a known 3D scene.

To understand the importance of this problem we give the following example. Figure 11.3 shows a 3D scene having three hierarchical levels of details. The apparent complexity of the scene decreases as we go radially outwards from the center to the boundary. We have shown two camera configurations. The first setup consists of six conventional cameras, placed at varying distances, to capture the hierarchical features (similar to the setup used in [83]). In the second setup we use two omnidirectional cameras with fisheye lenses. Since the images captured by a fisheye lens have very high resolution at the center and lower resolutions at the boundaries, it will enable us to reconstruct this scene in the required manner. By comparing the reconstruction results of the two configurations we can either move the cameras to improve the reconstruction or configure a new setup, with different sets of cameras, to provide better 3D reconstruction.

Appendix A

Unconstrained cases in Generic calibration

A.1 Taxonomy of Calibration Algorithms

We have classified our generic calibration algorithms into three subclasses: non-central, axial and central. The algorithms also depend on the nature of the calibration grids. The calibration grid can be either planar or 3D. Based on these differences we can have 6 different calibration algorithms for 3D cameras. We studied that an algorithm tailormade for non-central camera produces ambiguous solutions for central cameras. On the otherhand, an algorithm tailormade for central cameras produces inconsistent solution for non-central cameras. The following table summarises the nature of the results on applying the various calibration algorithms for diffent camera models. Note that we have not developed the calibration algorithm for axial cameras using 3D calibration grids.

Alg/Model	C-Planar	C-3D	A-Planar	NC-Planar	NC-3D
C-Planar	Unique	NS	NS	NS	NS
C-3D	7/12	Unique	NS	6/12	NS
A-Planar	2/10	4/10	Unique	NS	NS
NC-Planar	5/14	8/14	2/14	Unique	NS
NC-3D	21/30	19/30	18/30	17/30	Unique

Table A.1: Nature of solutions on applying the calibration algorithms, tailormade for specific camera models, on other camera models. 'NS' means no solution. 'r/n' refers to a rank deficiency of r when the tensor dimension is n.

A.2 Generic Calibration in the Case of Restricted Motion Sequences

A.2.1 Pure translation

3D calibration grid

We take three images of the 3D calibration grid by purely translating the camera. Let $\mathbf{Q}$,$\mathbf{Q}'$ and $\mathbf{Q}''$ refer to the points on the three grids observed by the same image pixel p. Let the translation of the second and third grids with respect to the first is given by $\mathbf{t}'$ and $\mathbf{t}''$ respectively. The three points, once expressed in the same reference frame (as of the first grid in this case), are nothing but collinear points. As a result all possible

3×3 subdeterminants of the following 4×3 matrix vanish.

$$\begin{pmatrix} \begin{pmatrix} Q_1 \\ Q_2 \\ Q_3 \\ Q_4 \end{pmatrix} & \begin{pmatrix} Q'_1 + t'_1 Q'_4 \\ Q'_2 + t'_2 Q'_4 \\ Q'_3 + t'_3 Q'_4 \\ Q'_4 \end{pmatrix} & \begin{pmatrix} Q''_1 + t''_1 Q''_4 \\ Q''_2 + t''_2 Q''_4 \\ Q''_3 + t''_3 Q''_4 \\ Q''_4 \end{pmatrix} \end{pmatrix}$$

We transform each of the points in the above triplet by $-\mathbf{Q}$. This transforms the first grid point in each ray to the origin. Consequently we obtain a set of rays passing through the origin.

$$\begin{pmatrix} \begin{pmatrix} Q_1 - Q_1 \\ Q_2 - Q_2 \\ Q_3 - Q_3 \\ Q_4 \end{pmatrix} & \begin{pmatrix} (Q'_1 - Q_1) + t'_1 Q'_4 \\ (Q'_2 - Q_2) + t'_2 Q'_4 \\ (Q'_3 - Q_3) + t'_3 Q'_4 \\ Q'_4 \end{pmatrix} & \begin{pmatrix} (Q''_1 - Q_1) + t''_1 Q''_4 \\ (Q''_2 - Q_2) + t''_2 Q''_4 \\ (Q''_3 - Q_3) + t''_3 Q''_4 \\ Q''_4 \end{pmatrix} \end{pmatrix}$$

Without loss of generality we continue to use the same notations for $\mathbf{Q}'$ and $\mathbf{Q}''$ after the transformations $\mathbf{Q}' \longleftarrow \mathbf{Q}' - \mathbf{Q}$ and $\mathbf{Q}'' \longleftarrow \mathbf{Q}'' - \mathbf{Q}$.

$$\begin{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ Q_4 \end{pmatrix} & \begin{pmatrix} Q'_1 + t'_1 Q'_4 \\ Q'_2 + t'_2 Q'_4 \\ Q'_3 + t'_3 Q'_4 \\ Q'_4 \end{pmatrix} & \begin{pmatrix} Q''_1 + t''_1 Q''_4 \\ Q''_2 + t''_2 Q''_4 \\ Q''_3 + t''_3 Q''_4 \\ Q''_4 \end{pmatrix} \end{pmatrix} \tag{A.1}$$

We observe that $\mathbf{Q}' - \mathbf{t}'$ and $\mathbf{Q}'' - \mathbf{t}''$ are equivalent upto a scale. Mathematically every 2×2 subdeterminant of the following matrix vanishes.

$$\begin{pmatrix} \begin{pmatrix} Q'_1 + t'_1 Q'_4 \\ Q'_2 + t'_2 Q'_4 \\ Q'_3 + t'_3 Q'_4 \end{pmatrix} & \begin{pmatrix} Q''_1 + t''_1 Q''_4 \\ Q''_2 + t''_2 Q''_4 \\ Q''_3 + t''_3 Q''_4 \end{pmatrix} \end{pmatrix} \tag{A.2}$$

i	V_i	T_i^1	T_i^2	T_i^3
1	t'_1	0	$Q'_4 Q'_3$	$Q'_4 Q'_2$
2	t'_2	$Q'_4 Q'_3$	0	$-Q'_4 Q'_1$
3	t'_3	$-Q'_4 Q'_2$	$-Q'_4 Q'_1$	0
4	t''_1	0	$-Q'_3 Q'_4$	$-Q'_2 Q'_4$
5	t''_2	$-Q'_3 Q'_4$	0	$Q'_1 Q'_4$
6	t''_3	$Q'_2 Q'_4$	$Q'_1 Q'_4$	0
7	$t'_1 t''_2 - t'_2 t''_1$	0	0	$Q'_4 Q'_4$
8	$t'_2 t''_3 - t'_3 t''_2$	$Q'_4 Q'_4$	0	0
9	$t'_1 t''_3 - t'_3 t''_1$	0	$Q'_4 Q'_4$	0
10	-1	$Q'_3 Q'_2 - Q'_2 Q'_3$	$Q'_3 Q'_1 - Q'_1 Q'_3$	$Q'_2 Q'_1 - Q'_1 Q'_2$

Table A.2: Bilinear constraints in the case of pure translation of the camera.

As shown in the Table A.2 we have the constraints $\sum_{i=1}^{10} V_i T_i^1$, $\sum_{i=1}^{10} V_i T_i^2$ and $\sum_{i=1}^{10} V_i T_i^3$ from the three possible 2×2 matrix of the matrix A.2. On solving the three equations independently we obtain both t' and t'' uniquely. Our method gives unique solutions for axial and central cameras.

General motion approach

We adapt the algorithm developed for general motion, given in chapter 4, to pure translation scenarios. In order to do this we generate two random rotations R'_r and R''_r and apply them to all the points in second and the third grid respectively. Again without loss of generality we continue to use the same notations for $\mathbf{Q}'$ and $\mathbf{Q}''$ after the transformation.

$$\mathbf{Q}' \longleftarrow \mathsf{R}'_r\mathbf{Q}', \quad \mathbf{Q}'' \longleftarrow \mathsf{R}''_r\mathbf{Q}''$$

Now we solve the problem and extract R', R'', $\mathbf{t}'$ and $\mathbf{t}''$. We can compute the translation vectors and the rotations are just the inverses (or transposes) of R'_r and R''_r.

Planar calibration grids

In the case of planar calibration grids the z coordinate of $\mathbf{Q}$, $\mathbf{Q}'$ and $\mathbf{Q}''$ become zeros. After transforming the points in each of the triplet by $-\mathbf{Q}$ we have the following matrix.

$$\begin{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ Q_4 \end{pmatrix} & \begin{pmatrix} Q'_1 + t'_1Q'_4 \\ Q'_2 + t'_2Q'_4 \\ t'_3Q'_4 \\ Q'_4 \end{pmatrix} & \begin{pmatrix} Q''_1 + t''_1Q''_4 \\ Q''_2 + t''_2Q''_4 \\ t''_3Q''_4 \\ Q''_4 \end{pmatrix} \end{pmatrix} \tag{A.3}$$

$$\begin{pmatrix} \begin{pmatrix} Q'_1 + t'_1Q'_4 \\ Q'_2 + t'_2Q'_4 \\ t'_3Q'_4 \end{pmatrix} & \begin{pmatrix} Q''_1 + t''_1Q''_4 \\ Q''_2 + t''_2Q''_4 \\ t''_3Q''_4 \end{pmatrix} \end{pmatrix} \tag{A.4}$$

By scaling the third row in the matrix A.4 we still maintain the collinearity conditions. This also imlies that the collinearity constraints alone are not sufficient to compute $\mathbf{t}'$ and $\mathbf{t}''$ in absolute scale.

i	V_i	T_i^1	T_i^2	T_i^3
1	t'_1	0	0	$Q'_4Q'_2$
2	t'_2	0	0	$-Q'_4Q'_1$
3	t'_3	$-Q'_4Q'_2$	$-Q'_4Q'_1$	0
4	t''_1	0	0	$-Q'_2Q'_4$
5	t''_2	0	0	$Q'_1Q'_4$
6	t''_3	$Q'_2Q'_4$	$-Q'_1Q'_4$	0
7	$t'_1t''_2 - t'_2t'_1$	0	0	$Q'_4Q'_4$
8	$t'_2t''_3 - t'_3t'_2$	$Q'_4Q'_4$	0	0
9	$t'_1t''_3 - t'_3t'_1$	0	$Q'_4Q'_4$	0
10	-1	0	0	$Q'_2Q'_1 - Q'_1Q'_2$

Table A.3: Bilinear constraints in the case of pure translation of planar calibration grids.

We have a rank deficiency of 3 in the third tensor. The first two tensors provide the values of $\mathbf{t}'_2$ and $\mathbf{t}''_3$ only upto a scale. We can use this result in the third tensor to uniquely compute the other variables.

A.2.2 Pure rotation

Rotation about the optical center for central cameras

The constraints involve only two grids at a time.

$$\left(\begin{pmatrix}0\\0\\0\\Q_4\end{pmatrix} \quad \begin{pmatrix}Q_1+t_1\\Q_2+t_2\\Q_3+t_3\\Q_4\end{pmatrix} \quad \begin{pmatrix}R_{11}Q'_1+R_{12}Q'_2+R_{13}Q'_3+t'_1Q'_4\\R_{21}Q'_1+R_{22}Q'_2+R_{23}Q'_3+t'_1Q'_4\\R_{31}Q'_1+R_{32}Q'_2+R_{33}Q'_3+t'_1Q'_4\\Q'_4\end{pmatrix}\right) \tag{A.5}$$

$$\mathbf{t}' = \mathsf{R}\mathbf{t}$$

Rotation about a known 3D point in the calibration grid

Let t be the known 3D point on the calibration grids. We transform the origin to the t in the first grid's reference frame. Without loss of ambiguity we continue the same notations $\mathbf{Q}$,$\mathbf{Q}'$ and $\mathbf{Q}''$ after the transformation.

$$\left(\begin{pmatrix}Q_1\\Q_2\\Q_3\\Q_4\end{pmatrix} \quad \begin{pmatrix}R_{11}Q'_1+R_{12}Q'_2+R_{13}Q'_3\\R_{21}Q'_1+R_{22}Q'_2+R_{23}Q'_3\\R_{31}Q'_1+R_{32}Q'_2+R_{33}Q'_3\\Q'_4\end{pmatrix} \begin{pmatrix}R_{11}Q'_1+R_{12}Q'_2+R_{13}Q'_3\\R_{21}Q'_1+R_{22}Q'_2+R_{23}Q'_3\\R_{31}Q'_1+R_{32}Q'_2+R_{33}Q'_3\\Q'_4\end{pmatrix}\right) \tag{A.6}$$

Rotation about an unknown 3D point in the calibration grid

$$\left(\begin{pmatrix}Q_1+t_1\\Q_2+t_2\\Q_3+t_3\\Q_4\end{pmatrix} \quad \begin{pmatrix}R'_{11}Q'_1+R'_{12}Q'_2+R'_{13}Q'_3+t'_1Q'_4\\R'_{21}Q'_1+R'_{22}Q'_2+R'_{23}Q'_3+t'_2Q'_4\\R'_{31}Q'_1+R'_{32}Q'_2+R'_{33}Q'_3+t'_3Q'_4\\Q'_4\end{pmatrix} \begin{pmatrix}R''_{11}Q''_1+R''_{12}Q''_2+R''_{13}Q''_3+t''_1Q''_4\\R''_{21}Q''_1+R''_{22}Q''_2+R''_{23}Q''_3+t''_2Q''_4\\R''_{31}Q''_1+R''_{32}Q''_2+R''_{33}Q''_3+t''_3Q''_4\\Q''_4\end{pmatrix}\right) \tag{A.7}$$

$$\boldsymbol{t}' = \mathsf{R}'\boldsymbol{t}, \;\; \boldsymbol{t}'' = \mathsf{R}''\boldsymbol{t}$$

A.3 Analysis of Underconstrained Cases for 2D Cameras

Here, we investigate what happens when we use the general algorithm, but with data that stems from a central camera. In that case, the linear estimation of our tensors does not give a unique solution, and the subsequent calibration algorithm will fail. In the following, we analyze the degree of ambiguity of the solution. Our main motivation for doing this analysis is related to the calibration of stereo system: a stereo system consisting of central cameras, is considered here as a single non-central sensor. It turns out unhappily that a stereo system consisting of 2 central cameras, does not lead to a unique solution of the general calibration method. It is shown below that the minimum case of a stereo system that can be calibrated using the basic algorithm for non-central cameras, is a system with 3 central cameras.

To carry out our analysis, we simulate data points that are already aligned, i.e. the true solution for the motions is the identity. We then check if there exist other solutions, which would mean that calibration using the general method fails.

We first deal with the case of a single central 2D camera, for which no unique solution exists. Then, it is shown that a system with two central 2D cameras, already can be calibrated.

Then, we consider the 3D case, for one, two and three central cameras.

A.3.1 A single central 2D camera

Points are already aligned, so each triplet of correspondences can be modeled by variables X and Y (for the ray on which they lie) and scales s, s', s'', that express the positions of the points along the ray:

$$\mathbf{Q} = \begin{pmatrix} X \\ Y \\ s \end{pmatrix} \qquad \mathbf{Q}' = \begin{pmatrix} X \\ Y \\ s' \end{pmatrix} \qquad \mathbf{Q}'' = \begin{pmatrix} X \\ Y \\ s'' \end{pmatrix}$$

We plug these coordinates into equation (4.1) and the associated table 4.1 of its coefficients. This gives us table A.4.

i	C_i	V_i
1	$s''(X^2+Y^2)$	R'_{21}
2	0	R'_{22}
3	$s'(X^2+Y^2)$	$-R''_{21}$
4	0	$-R''_{22}$
5	$s(X^2+Y^2)$	$R'_{11}R''_{21} - R''_{11}R'_{21}$
6	0	$R'_{11}R''_{22} - R''_{12}R'_{21}$
7	$s's''X$	$t'_2 - t''_2$
8	$s's''Y$	$-t'_1 + t''_1$
9	$ss''X$	$R'_{11}t''_2 - R'_{21}t''_1$
10	$ss''Y$	$R'_{12}t''_2 - R'_{22}t''_1$
11	$ss'X$	$R''_{21}t'_1 - R''_{11}t'_2$
12	$ss'Y$	$R''_{22}t'_1 - R''_{12}t'_2$
13	$ss's''$	$t'_1t''_2 - t''_1t'_2$

Table A.4: Table 4.1 for data coming from a single central camera.

We observe that the tensor equation has 3 coefficients that are zero: the rank deficiency of the equation system is thus 3 at least (V_2, V_4 and V_6 are unconstrained). Hence, the linear estimation of the tensor will not give a unique solution! We should check if the rank-deficieny is even higher.

We argue that tensor coefficients associated with data coefficients that are different combinations of X, Y, s, s', s'', must be zero. Overall, we thus conclude that the linear equation system for estimating the tensor, has a rank-deficiency of 3, instead of 1 for general data.

Let us look at the solutions of the linear equation system. First, note that the true solution (identity transformations) is represented by the vector:

$$\begin{pmatrix} 0 \\ 1 \\ 0 \\ -1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Clearly, it is in the null-space of the equation matrix (the non-zero tensor coefficients are the ones that are not constrained by the linear equations).

The null-space of the equation system, when established with data from a central camera, is formed by vectors (for any a, b, c):

$$\begin{pmatrix} 0 \\ a \\ 0 \\ b \\ 0 \\ c \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

We do not need to prove for a different coordinate system because the degeneracy problem is independent of the coordinate systems (section A.5).

A.3.2 Two central 2D cameras

We now extend the previous section by taking into account a second central camera: besides rays as in the previous section, we now consider additional rays passing through a second point (optical center). Let the second point be, without loss of generality:

$$\begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}$$

Triplets of points on rays going through that point, can be modeled as:

$$\begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} + \frac{1}{s} \begin{pmatrix} X \\ Y \\ 0 \end{pmatrix} = \begin{pmatrix} X/s \\ 1 + Y/s \\ 1 \end{pmatrix} \sim \begin{pmatrix} X \\ s + Y \\ s \end{pmatrix}$$

Inserting these coordinates in table 4.1 gives coefficients on the tensor equation that are given in table A.5.

We now check which of the null-vectors of the previous section, are also null-vectors of the extended equation system. They have thus to satisfy $C_2 a + C_4 b + C_6 c = 0$:

$$X \left\{ s''(s' - s)a + s'(s'' - s)b + s(s'' - s')c \right\} = 0$$

We exclude $X = 0$, and order the other terms:

$$-ss'(b + c) + ss''(c - a) + s's''(a + b) = 0$$

This must hold for any values of s, s', s'', thus it must hold:

$$b + c = c - a = a + b = 0$$

It follows that $a = c = -b$, which corresponds, up to scale, to the true solution.

In conclusion, a system consisting of two central cameras, allows in general a unique linear solution of the calibration tensor, and can thus be calibrated with the general method.

i	C_i	V_i
1	$s''\left\{X^2+(Y+s)(Y+s')\right\}$	R'_{21}
2	$s''X(s'-s)$	R'_{22}
3	$s'\left\{X^2+(Y+s)(Y+s'')\right\}$	$-R''_{21}$
4	$s'X(s''-s)$	$-R''_{22}$
5	$s\left\{X^2+(Y+s')(Y+s'')\right\}$	$R'_{11}R''_{21}-R''_{11}R'_{21}$
6	$sX(s''-s')$	$R'_{11}R''_{22}-R''_{12}R'_{21}$
7	$s's''X$	$t'_2-t''_2$
8	$s's''(Y+s)$	$-t'_1+t''_1$
9	$ss''X$	$R'_{11}t''_2-R'_{21}t''_1$
10	$ss''(Y+s')$	$R'_{12}t''_2-R'_{22}t''_1$
11	$ss'X$	$R''_{21}t'_1-R''_{11}t'_2$
12	$ss'(Y+s'')$	$R''_{22}t'_1-R''_{12}t'_2$
13	$ss's''$	$t'_1t''_2-t''_1t'_2$

Table A.5: Table 4.1 for data coming from the second central camera.

A.4 Analysis of Underconstrained Cases for 3D Cameras

We now do a similar analysis for 3D cameras. First, we give expressions for all four tensors involved: table 4.1.3 is extended and completely given in tables A.6 and A.7.

One may observe that these tensors share many coefficients. There are exactly 69 different coefficients, that are shared among the four tensors as shown in tables A.8 and A.9.

C_i	T_i^1	T_i^2	T_i^3	T_i^4
$Q_1Q_1'Q_1''$	0	0	0	$R_{21}'R_{31}'' - R_{31}'R_{21}''$
$Q_1Q_1'Q_2''$	0	0	0	$R_{21}'R_{32}'' - R_{31}'R_{22}''$
$Q_1Q_1'Q_3''$	0	0	0	$R_{21}'R_{33}'' - R_{31}'R_{23}''$
$Q_1Q_1'Q_4''$	0	R_{31}'	R_{21}'	$R_{21}'t_3'' - R_{31}'t_2''$
$Q_1Q_2'Q_1''$	0	0	0	$R_{22}'R_{31}'' - R_{32}'R_{21}''$
$Q_1Q_2'Q_2''$	0	0	0	$R_{22}'R_{32}'' - R_{32}'R_{22}''$
$Q_1Q_2'Q_3''$	0	0	0	$R_{22}'R_{33}'' - R_{32}'R_{23}''$
$Q_1Q_2'Q_4''$	0	R_{32}'	R_{22}'	$R_{22}'t_3'' - R_{32}'t_2''$
$Q_1Q_3'Q_1''$	0	0	0	$R_{23}'R_{31}'' - R_{33}'R_{21}''$
$Q_1Q_3'Q_2''$	0	0	0	$R_{23}'R_{32}'' - R_{33}'R_{22}''$
$Q_1Q_3'Q_3''$	0	0	0	$R_{23}'R_{33}'' - R_{33}'R_{23}''$
$Q_1Q_3'Q_4''$	0	R_{33}'	R_{23}'	$R_{23}'t_3'' - R_{33}'t_2''$
$Q_1Q_4'Q_1''$	0	$-R_{31}''$	$-R_{21}''$	$t_2'R_{31}'' - t_3'R_{21}''$
$Q_1Q_4'Q_2''$	0	$-R_{32}''$	$-R_{22}''$	$t_2'R_{32}'' - t_3'R_{22}''$
$Q_1Q_4'Q_3''$	0	$-R_{33}''$	$-R_{23}''$	$t_2'R_{33}'' - t_3'R_{23}''$
$Q_1Q_4'Q_4''$	0	$t_3' - t_3''$	$t_2' - t_2''$	$t_2't_3'' - t_3't_2''$
$Q_2Q_1'Q_1''$	0	0	0	$R_{31}'R_{11}'' - R_{11}'R_{31}''$
$Q_2Q_1'Q_2''$	0	0	0	$R_{31}'R_{12}'' - R_{11}'R_{32}''$
$Q_2Q_1'Q_3''$	0	0	0	$R_{31}'R_{13}'' - R_{11}'R_{33}''$
$Q_2Q_1'Q_4''$	R_{31}'	0	$-R_{11}'$	$R_{31}'t_1'' - R_{11}'t_3''$
$Q_2Q_2'Q_1''$	0	0	0	$R_{32}'R_{11}'' - R_{12}'R_{31}''$
$Q_2Q_2'Q_2''$	0	0	0	$R_{32}'R_{12}'' - R_{12}'R_{32}''$
$Q_2Q_2'Q_3''$	0	0	0	$R_{32}'R_{13}'' - R_{12}'R_{33}''$
$Q_2Q_2'Q_4''$	R_{32}'	0	$-R_{12}'$	$R_{32}'t_1'' - R_{12}'t_3''$
$Q_2Q_3'Q_1''$	0	0	0	$R_{33}'R_{11}'' - R_{13}'R_{31}''$
$Q_2Q_3'Q_2''$	0	0	0	$R_{33}'R_{12}'' - R_{13}'R_{32}''$
$Q_2Q_3'Q_3''$	0	0	0	$R_{33}'R_{13}'' - R_{13}'R_{33}''$
$Q_2Q_3'Q_4''$	R_{33}'	0	$-R_{13}'$	$R_{33}'t_1'' - R_{13}'t_3''$
$Q_2Q_4'Q_1''$	$-R_{31}''$	0	R_{11}''	$t_3'R_{11}'' - t_1'R_{31}''$
$Q_2Q_4'Q_2''$	$-R_{32}''$	0	R_{12}''	$t_3'R_{12}'' - t_1'R_{32}''$
$Q_2Q_4'Q_3''$	$-R_{33}''$	0	R_{13}''	$t_3'R_{13}'' - t_1'R_{33}''$
$Q_2Q_4'Q_4''$	$t_3' - t_3''$	0	$t_1'' - t_1'$	$t_3't_1'' - t_1't_3''$

Table A.6: Tensor coefficients, part I.

C_i	T_i^1	T_i^2	T_i^3	T_i^4
$Q_3Q_1'Q_1''$	0	0	0	$R_{11}'R_{21}'' - R_{21}'R_{11}''$
$Q_3Q_1'Q_2''$	0	0	0	$R_{11}'R_{22}'' - R_{21}'R_{12}''$
$Q_3Q_1'Q_3''$	0	0	0	$R_{11}'R_{23}'' - R_{21}'R_{13}''$
$Q_3Q_1'Q_4''$	$-R_{21}'$	$-R_{11}'$	0	$R_{11}'t_2'' - R_{21}'t_1''$
$Q_3Q_2'Q_1''$	0	0	0	$R_{12}'R_{21}'' - R_{22}'R_{11}''$
$Q_3Q_2'Q_2''$	0	0	0	$R_{12}'R_{22}'' - R_{22}'R_{12}''$
$Q_3Q_2'Q_3''$	0	0	0	$R_{12}'R_{23}'' - R_{22}'R_{13}''$
$Q_3Q_2'Q_4''$	$-R_{22}'$	$-R_{12}'$	0	$R_{12}'t_2'' - R_{22}'t_1''$
$Q_3Q_3'Q_1''$	0	0	0	$R_{13}'R_{21}'' - R_{23}'R_{11}''$
$Q_3Q_3'Q_2''$	0	0	0	$R_{13}'R_{22}'' - R_{23}'R_{12}''$
$Q_3Q_3'Q_3''$	0	0	0	$R_{13}'R_{23}'' - R_{23}'R_{13}''$
$Q_3Q_3'Q_4''$	$-R_{23}'$	$-R_{13}'$	0	$R_{13}'t_2'' - R_{23}'t_1''$
$Q_3Q_4'Q_1''$	R_{21}''	R_{11}''	0	$t_1'R_{21}'' - t_2'R_{11}''$
$Q_3Q_4'Q_2''$	R_{22}''	R_{12}''	0	$t_1'R_{22}'' - t_2'R_{12}''$
$Q_3Q_4'Q_3''$	R_{23}''	R_{13}''	0	$t_1'R_{23}'' - t_2'R_{13}''$
$Q_3Q_4'Q_4''$	$t_2'' - t_2'$	$t_1'' - t_1'$	0	$t_1't_2'' - t_2't_1''$
$Q_4Q_1'Q_1''$	$R_{21}'R_{31}'' - R_{21}''R_{31}'$	$R_{11}'R_{31}'' - R_{11}''R_{31}'$	$R_{11}'R_{21}'' - R_{21}'R_{11}''$	0
$Q_4Q_1'Q_2''$	$R_{21}'R_{32}'' - R_{22}''R_{31}'$	$R_{11}'R_{32}'' - R_{12}''R_{31}'$	$R_{11}'R_{22}'' - R_{21}'R_{12}''$	0
$Q_4Q_1'Q_3''$	$R_{21}'R_{33}'' - R_{23}''R_{31}'$	$R_{11}'R_{33}'' - R_{13}''R_{31}'$	$R_{11}'R_{23}'' - R_{21}'R_{13}''$	0
$Q_4Q_1'Q_4''$	$R_{21}'t_3'' - R_{31}'t_2''$	$R_{11}'t_3'' - R_{31}'t_1''$	$R_{11}'t_2'' - R_{21}'t_1''$	0
$Q_4Q_2'Q_1''$	$R_{22}'R_{31}'' - R_{21}''R_{32}'$	$R_{12}'R_{31}'' - R_{11}''R_{32}'$	$R_{12}'R_{21}'' - R_{22}'R_{11}''$	0
$Q_4Q_2'Q_2''$	$R_{22}'R_{32}'' - R_{22}''R_{32}'$	$R_{12}'R_{32}'' - R_{12}''R_{32}'$	$R_{12}'R_{22}'' - R_{22}'R_{12}''$	0
$Q_4Q_2'Q_3''$	$R_{22}'R_{33}'' - R_{23}''R_{32}'$	$R_{12}'R_{33}'' - R_{13}''R_{32}'$	$R_{12}'R_{23}'' - R_{22}'R_{13}''$	0
$Q_4Q_2'Q_4''$	$R_{22}'t_3'' - R_{32}'t_2''$	$R_{12}'t_3'' - R_{32}t_1''$	$R_{12}'t_2'' - R_{22}'t_1''$	0
$Q_4Q_3'Q_1''$	$R_{23}'R_{31}'' - R_{21}''R_{33}'$	$R_{13}'R_{31}'' - R_{11}''R_{33}'$	$R_{13}'R_{21}'' - R_{23}'R_{11}''$	0
$Q_4Q_3'Q_2''$	$R_{23}'R_{32}'' - R_{22}''R_{33}'$	$R_{13}'R_{32}'' - R_{12}''R_{33}'$	$R_{13}'R_{22}'' - R_{23}'R_{12}''$	0
$Q_4Q_3'Q_3''$	$R_{23}'R_{33}'' - R_{23}''R_{33}'$	$R_{13}'R_{33}'' - R_{13}''R_{33}'$	$R_{13}'R_{23}'' - R_{23}'R_{13}''$	0
$Q_4Q_3'Q_4''$	$R_{23}'t_3'' - R_{33}'t_2''$	$R_{13}'t_3'' - R_{33}'t_1''$	$R_{13}'t_2'' - R_{23}'t_1''$	0
$Q_4Q_4'Q_1''$	$R_{31}''t_2' - R_{21}''t_3'$	$R_{31}''t_1' - R_{11}''t_3'$	$t_1'R_{21}'' - t_2'R_{11}''$	0
$Q_4Q_4'Q_2''$	$R_{32}''t_2' - R_{22}''t_3'$	$R_{32}''t_1' - R_{12}''t_3'$	$t_1'R_{22}'' - t_2'R_{12}''$	0
$Q_4Q_4'Q_3''$	$R_{33}''t_2' - R_{23}''t_3'$	$R_{33}''t_1' - R_{13}''t_3'$	$t_1'R_{23}'' - t_2'R_{13}''$	0
$Q_4Q_4'Q_4''$	$t_2't_3'' - t_3't_2''$	$t_1't_3'' - t_1''t_3'$	$t_1't_2'' - t_2't_1''$	0

Table A.7: Tensor coefficients, part II.

	Coupled motion pars	T^1	T^2	T^3	T^4
1	R'_{11}	0	$-Q_3Q'_1Q''_4$	$-Q_2Q'_1Q''_4$	0
2	R'_{12}	0	$-Q_3Q'_2Q''_4$	$-Q_2Q'_2Q''_4$	0
3	R'_{13}	0	$-Q_3Q'_3Q''_4$	$-Q_2Q'_3Q''_4$	0
4	R'_{21}	$-Q_3Q'_1Q''_4$	0	$Q_1Q'_1Q''_4$	0
5	R'_{22}	$-Q_3Q'_2Q''_4$	0	$Q_1Q'_2Q''_4$	0
6	R'_{23}	$-Q_3Q'_3Q''_4$	0	$Q_1Q'_3Q''_4$	0
7	R'_{31}	$Q_2Q'_1Q''_4$	$Q_1Q'_1Q''_4$	0	0
8	R'_{32}	$Q_2Q'_2Q''_4$	$Q_1Q'_2Q''_4$	0	0
9	R'_{33}	$Q_2Q'_3Q''_4$	$Q_1Q'_3Q''_4$	0	0
10	R''_{11}	0	$Q_3Q'_4Q''_1$	$Q_2Q'_4Q''_1$	0
11	R''_{12}	0	$Q_3Q'_4Q''_2$	$Q_2Q'_4Q''_2$	0
12	R''_{13}	0	$Q_3Q'_4Q''_3$	$Q_2Q'_4Q''_3$	0
13	R''_{21}	$Q_3Q'_4Q''_1$	0	$-Q_1Q'_4Q''_1$	0
14	R''_{22}	$Q_3Q'_4Q''_2$	0	$-Q_1Q'_4Q''_2$	0
15	R''_{23}	$Q_3Q'_4Q''_3$	0	$-Q_1Q'_4Q''_3$	0
16	R''_{31}	$-Q_2Q'_4Q''_1$	$-Q_1Q'_4Q''_1$	0	0
17	R''_{32}	$-Q_2Q'_4Q''_2$	$-Q_1Q'_4Q''_2$	0	0
18	R''_{33}	$-Q_2Q'_4Q''_3$	$-Q_1Q'_4Q''_3$	0	0
19	$t'_1 - t''_1$	0	$-Q_3Q'_4Q''_4$	$-Q_2Q'_4Q''_4$	0
20	$t'_2 - t''_2$	$-Q_3Q'_4Q''_4$	0	$Q_1Q'_4Q''_4$	0
21	$t'_3 - t''_3$	$Q_2Q'_4Q''_4$	$Q_1Q'_4Q''_4$	0	0
22	$R'_{11}R''_{21} - R'_{21}R''_{11}$	0	0	$Q_4Q'_1Q''_1$	$Q_3Q'_1Q''_1$
23	$R'_{11}R''_{22} - R'_{21}R''_{12}$	0	0	$Q_4Q'_1Q''_2$	$Q_3Q'_1Q''_2$
24	$R'_{11}R''_{23} - R'_{21}R''_{13}$	0	0	$Q_4Q'_1Q''_3$	$Q_3Q'_1Q''_3$
25	$R'_{12}R''_{21} - R'_{22}R''_{11}$	0	0	$Q_4Q'_2Q''_1$	$Q_3Q'_2Q''_1$
26	$R'_{12}R''_{22} - R'_{22}R''_{12}$	0	0	$Q_4Q'_2Q''_2$	$Q_3Q'_2Q''_2$
27	$R'_{12}R''_{23} - R'_{22}R''_{13}$	0	0	$Q_4Q'_2Q''_3$	$Q_3Q'_2Q''_3$
28	$R'_{13}R''_{21} - R'_{23}R''_{11}$	0	0	$Q_4Q'_3Q''_1$	$Q_3Q'_3Q''_1$
29	$R'_{13}R''_{22} - R'_{23}R''_{12}$	0	0	$Q_4Q'_3Q''_2$	$Q_3Q'_3Q''_2$
30	$R'_{13}R''_{23} - R'_{23}R''_{13}$	0	0	$Q_4Q'_3Q''_3$	$Q_3Q'_3Q''_3$
31	$R'_{11}R''_{31} - R'_{31}R''_{11}$	0	$Q_4Q'_1Q''_1$	0	$-Q_2Q'_1Q''_1$
32	$R'_{11}R''_{32} - R'_{31}R''_{12}$	0	$Q_4Q'_1Q''_2$	0	$-Q_2Q'_1Q''_2$
33	$R'_{11}R''_{33} - R'_{31}R''_{13}$	0	$Q_4Q'_1Q''_3$	0	$-Q_2Q'_1Q''_3$
34	$R'_{12}R''_{31} - R'_{32}R''_{11}$	0	$Q_4Q'_2Q''_1$	0	$-Q_2Q'_2Q''_1$
35	$R'_{12}R''_{32} - R'_{32}R''_{12}$	0	$Q_4Q'_2Q''_2$	0	$-Q_2Q'_2Q''_2$
36	$R'_{12}R''_{33} - R'_{32}R''_{13}$	0	$Q_4Q'_2Q''_3$	0	$-Q_2Q'_2Q''_3$
37	$R'_{13}R''_{31} - R'_{33}R''_{11}$	0	$Q_4Q'_3Q''_1$	0	$-Q_2Q'_3Q''_1$
38	$R'_{13}R''_{32} - R'_{33}R''_{12}$	0	$Q_4Q'_3Q''_2$	0	$-Q_2Q'_3Q''_2$
39	$R'_{13}R''_{33} - R'_{33}R''_{13}$	0	$Q_4Q'_3Q''_3$	0	$-Q_2Q'_3Q''_3$

Table A.8: Coupled tensor coefficients, part I.

	Coupled motion pars	T^1	T^2	T^3	T^4
40	$R'_{21}R''_{31} - R'_{31}R''_{21}$	$Q_4Q'_1Q''_1$	0	0	$Q_1Q'_1Q''_1$
41	$R'_{21}R''_{32} - R'_{31}R''_{22}$	$Q_4Q'_1Q''_2$	0	0	$Q_1Q'_1Q''_2$
42	$R'_{21}R''_{33} - R'_{31}R''_{23}$	$Q_4Q'_1Q''_3$	0	0	$Q_1Q'_1Q''_3$
43	$R'_{22}R''_{31} - R'_{32}R''_{21}$	$Q_4Q'_2Q''_1$	0	0	$Q_1Q'_2Q''_1$
44	$R'_{22}R''_{32} - R'_{32}R''_{22}$	$Q_4Q'_2Q''_2$	0	0	$Q_1Q'_2Q''_2$
45	$R'_{22}R''_{33} - R'_{32}R''_{23}$	$Q_4Q'_2Q''_3$	0	0	$Q_1Q'_2Q''_3$
46	$R'_{23}R''_{31} - R'_{33}R''_{21}$	$Q_4Q'_3Q''_1$	0	0	$Q_1Q'_3Q''_1$
47	$R'_{23}R''_{32} - R'_{33}R''_{22}$	$Q_4Q'_3Q''_2$	0	0	$Q_1Q'_3Q''_2$
48	$R'_{23}R''_{33} - R'_{33}R''_{23}$	$Q_4Q'_3Q''_3$	0	0	$Q_1Q'_3Q''_3$
49	$R'_{11}t''_2 - R'_{21}t''_1$	0	0	$Q_4Q'_1Q''_4$	$Q_3Q'_1Q''_4$
50	$R'_{12}t''_2 - R'_{22}t''_1$	0	0	$Q_4Q'_2Q''_4$	$Q_3Q'_2Q''_4$
51	$R'_{13}t''_2 - R'_{23}t''_1$	0	0	$Q_4Q'_3Q''_4$	$Q_3Q'_3Q''_4$
52	$R'_{11}t''_3 - R'_{31}t''_1$	0	$Q_4Q'_1Q''_4$	0	$-Q_2Q'_1Q''_4$
53	$R'_{12}t''_3 - R'_{32}t''_1$	0	$Q_4Q'_2Q''_4$	0	$-Q_2Q'_2Q''_4$
54	$R'_{13}t''_3 - R'_{33}t''_1$	0	$Q_4Q'_3Q''_4$	0	$-Q_2Q'_3Q''_4$
55	$R'_{21}t''_3 - R'_{31}t''_2$	$Q_4Q'_1Q''_4$	0	0	$Q_1Q'_1Q''_4$
56	$R'_{22}t''_3 - R'_{32}t''_2$	$Q_4Q'_2Q''_4$	0	0	$Q_1Q'_2Q''_4$
57	$R'_{23}t''_3 - R'_{33}t''_2$	$Q_4Q'_3Q''_4$	0	0	$Q_1Q'_3Q''_4$
58	$R''_{11}t'_2 - R''_{21}t'_1$	0	0	$-Q_4Q'_4Q''_1$	$-Q_3Q'_4Q''_1$
59	$R''_{12}t'_2 - R''_{22}t'_1$	0	0	$-Q_4Q'_4Q''_2$	$-Q_3Q'_4Q''_2$
60	$R''_{13}t'_2 - R''_{23}t'_1$	0	0	$-Q_4Q'_4Q''_3$	$-Q_3Q'_4Q''_3$
61	$R''_{11}t'_3 - R''_{31}t'_1$	0	$-Q_4Q'_4Q''_1$	0	$Q_2Q'_4Q''_1$
62	$R''_{12}t'_3 - R''_{32}t'_1$	0	$-Q_4Q'_4Q''_2$	0	$Q_2Q'_4Q''_2$
63	$R''_{13}t'_3 - R''_{33}t'_1$	0	$-Q_4Q'_4Q''_3$	0	$Q_2Q'_4Q''_3$
64	$R''_{21}t'_3 - R''_{31}t'_2$	$-Q_4Q'_4Q''_1$	0	0	$-Q_1Q'_4Q''_1$
65	$R''_{22}t'_3 - R''_{32}t'_2$	$-Q_4Q'_4Q''_2$	0	0	$-Q_1Q'_4Q''_2$
66	$R''_{23}t'_3 - R''_{33}t'_2$	$-Q_4Q'_4Q''_3$	0	0	$-Q_1Q'_4Q''_3$
67	$t'_1t''_2 - t'_2t''_1$	0	0	$Q_4Q'_4Q''_4$	$Q_3Q'_4Q''_4$
68	$t'_1t''_3 - t'_3t''_1$	0	$Q_4Q'_4Q''_4$	0	$-Q_2Q'_4Q''_4$
69	$t'_2t''_3 - t'_3t''_2$	$Q_4Q'_4Q''_4$	0	0	$Q_1Q'_4Q''_4$

Table A.9: Coupled tensor coefficients, part II.

A.4.1 A single central 3D camera

Aligned points are parameterized as:

$$\mathbf{Q} \sim \begin{pmatrix} X \\ Y \\ Z \\ s \end{pmatrix} \qquad \mathbf{Q}' \sim \begin{pmatrix} X \\ Y \\ Z \\ s' \end{pmatrix} \qquad \mathbf{Q}'' \sim \begin{pmatrix} X \\ Y \\ Z \\ s'' \end{pmatrix}$$

Inserting this in tables A.8 and A.9 gives tables A.10 and A.11. The last column of these tables indicates if tensor coefficients are estimated as being zero, and which of the four tensors allows this to be done. The underlying reasoning is as follows: consider coefficient 22, and the entry associated with the 3rd tensor: sX^2. This term appears in no other coefficient associated with that tensor, which is why coefficient 22 must be zero.

There are a total of 24 undetermined coefficients: 1, 5, 9, 10, 14, 18, 23, 24, 25, 27, 28, 29, 32, 33, 34, 36, 37, 38, 41, 42, 43, 45, 46, 47.

For the four tensors, we now list available constraints on these:

1 (5-9=0) (14-18=0) (41+43=0) (42+46=0) (45+47=0)

2 (1-9=0) (10-18=0) (32+34=0) (33+37=0) (36+38=0)

3 (1-5=0) (10-14=0) (23+25=0) (24+28=0) (27+29=0)

4 (42+46=0) (23+25-33-37+45+47=0) (24+28=0) (36+38=0) (27+29=0) (41+43=0) (32+34=0)

Note that some equations are redundant.

	Coupled motion pars	T^1	T^2	T^3	T^4	ZERO DUE TO
1	R'_{11}	0	$-s''XZ$	$-s''XY$	0	
2	R'_{12}	0	$-s''YZ$	$-s''Y^2$	0	2, 3
3	R'_{13}	0	$-s''Z^2$	$-s''YZ$	0	2, 3
4	R'_{21}	$-s''XZ$	0	$s''X^2$	0	1, 3
5	R'_{22}	$-s''YZ$	0	$s''XY$	0	
6	R'_{23}	$-s''Z^2$	0	$s''XZ$	0	1, 3
7	R'_{31}	$s''XY$	$s''X^2$	0	0	1, 2
8	R'_{32}	$s''Y^2$	$s''XY$	0	0	1, 2
9	R'_{33}	$s''YZ$	$s''XZ$	0	0	
10	R''_{11}	0	$s'XZ$	$s'XY$	0	
11	R''_{12}	0	$s'YZ$	$s'Y^2$	0	2, 3
12	R''_{13}	0	$s'Z^2$	$s'YZ$	0	2, 3
13	R''_{21}	$s'XZ$	0	$-s'X^2$	0	1, 3
14	R''_{22}	$s'YZ$	0	$-s'XY$	0	
15	R''_{23}	$s'Z^2$	0	$-s'XZ$	0	1, 3
16	R''_{31}	$-s'XY$	$-s'X^2$	0	0	1, 2
17	R''_{32}	$-s'Y^2$	$-s'XY$	0	0	1, 2
18	R''_{33}	$-s'YZ$	$-s'XZ$	0	0	
19	$t'_1 - t''_1$	0	$-s's''Z$	$-s's''Z$	0	2, 3
20	$t'_2 - t''_2$	$-s's''Z$	0	$s's''X$	0	1, 3
21	$t'_3 - t''_3$	$s's''Y$	$s's''X$	0	0	1, 2
22	$R'_{11}R''_{21} - R'_{21}R''_{11}$	0	0	sX^2	X^2Z	3
23	$R'_{11}R''_{22} - R'_{21}R''_{12}$	0	0	sXY	XYZ	
24	$R'_{11}R''_{23} - R'_{21}R''_{13}$	0	0	sXZ	XZ^2	
25	$R'_{12}R''_{21} - R'_{22}R''_{11}$	0	0	sXY	XYZ	
26	$R'_{12}R''_{22} - R'_{22}R''_{12}$	0	0	sY^2	Y^2Z	3
27	$R'_{12}R''_{23} - R'_{22}R''_{13}$	0	0	sYZ	YZ^2	
28	$R'_{13}R''_{21} - R'_{23}R''_{11}$	0	0	sXZ	XZ^2	
29	$R'_{13}R''_{22} - R'_{23}R''_{12}$	0	0	sYZ	YZ^2	
30	$R'_{13}R''_{23} - R'_{23}R''_{13}$	0	0	sZ^2	Z^3	3, 4
31	$R'_{11}R''_{31} - R'_{31}R''_{11}$	0	sX^2	0	$-X^2Y$	2
32	$R'_{11}R''_{32} - R'_{31}R''_{12}$	0	sXY	0	$-XY^2$	
33	$R'_{11}R''_{33} - R'_{31}R''_{13}$	0	sXZ	0	$-XYZ$	
34	$R'_{12}R''_{31} - R'_{32}R''_{11}$	0	sXY	0	$-XY^2$	
35	$R'_{12}R''_{32} - R'_{32}R''_{12}$	0	sY^2	0	$-Y^3$	2, 4
36	$R'_{12}R''_{33} - R'_{32}R''_{13}$	0	sYZ	0	$-Y^2Z$	
37	$R'_{13}R''_{31} - R'_{33}R''_{11}$	0	sXZ	0	$-XYZ$	
38	$R'_{13}R''_{32} - R'_{33}R''_{12}$	0	sYZ	0	$-Y^2Z$	
39	$R'_{13}R''_{33} - R'_{33}R''_{13}$	0	sZ^2	0	$-YZ^2$	2

Table A.10: Coupled tensor coefficients, for data coming from a single central camera, part I.

	Coupled motion pars	T^1	T^2	T^3	T^4	ZERO DUE TO
40	$R'_{21}R''_{31} - R'_{31}R''_{21}$	sX^2	0	0	X^3	1, 4
41	$R'_{21}R''_{32} - R'_{31}R''_{22}$	sXY	0	0	X^2Y	
42	$R'_{21}R''_{33} - R'_{31}R''_{23}$	sXZ	0	0	X^2Z	
43	$R'_{22}R''_{31} - R'_{32}R''_{21}$	sXY	0	0	X^2Y	
44	$R'_{22}R''_{32} - R'_{32}R''_{22}$	sY^2	0	0	XY^2	1
45	$R'_{22}R''_{33} - R'_{32}R''_{23}$	sYZ	0	0	XYZ	
46	$R'_{23}R''_{31} - R'_{33}R''_{21}$	sXZ	0	0	X^2Z	
47	$R'_{23}R''_{32} - R'_{33}R''_{22}$	sYZ	0	0	XYZ	
48	$R'_{23}R''_{33} - R'_{33}R''_{23}$	sZ^2	0	0	XZ^2	1
49	$R'_{11}t''_2 - R'_{21}t''_1$	0	0	$ss''X$	$s''XZ$	3
50	$R'_{12}t''_2 - R'_{22}t''_1$	0	0	$ss''Y$	$s''YZ$	3
51	$R'_{13}t''_2 - R'_{23}t''_1$	0	0	$ss''Z$	$s''Z^2$	3, 4
52	$R'_{11}t''_3 - R'_{31}t''_1$	0	$ss''X$	0	$-s''XY$	2
53	$R'_{12}t''_3 - R'_{32}t''_1$	0	$ss''Y$	0	$-s''Y^2$	2, 4
54	$R'_{13}t''_3 - R'_{33}t''_1$	0	$ss''Z$	0	$-s''YZ$	2
55	$R'_{21}t''_3 - R'_{31}t''_2$	$ss''X$	0	0	$s''X^2$	1, 4
56	$R'_{22}t''_3 - R'_{32}t''_2$	$ss''Y$	0	0	$s''XY$	1
57	$R'_{23}t''_3 - R'_{33}t''_2$	$ss''Z$	0	0	$s''XZ$	1
58	$R''_{11}t'_2 - R''_{21}t'_1$	0	0	$-ss'X$	$-s'XZ$	3
59	$R''_{12}t'_2 - R''_{22}t'_1$	0	0	$-ss'Y$	$-s'YZ$	3
60	$R''_{13}t'_2 - R''_{23}t'_1$	0	0	$-ss'Z$	$-s'Z^2$	3, 4
61	$R''_{11}t'_3 - R''_{31}t'_1$	0	$-ss'X$	0	$s'XY$	2
62	$R''_{12}t'_3 - R''_{32}t'_1$	0	$-ss'Y$	0	$s'Y^2$	2, 4
63	$R''_{13}t'_3 - R''_{33}t'_1$	0	$-ss'Z$	0	$s'YZ$	2
64	$R''_{21}t'_3 - R''_{31}t'_2$	$-ss'X$	0	0	$-s'X^2$	1, 4
65	$R''_{22}t'_3 - R''_{32}t'_2$	$-ss'Y$	0	0	$-s'XY$	1
66	$R''_{23}t'_3 - R''_{33}t'_2$	$-ss'Z$	0	0	$-s'XZ$	1
67	$t'_1t''_2 - t'_2t''_1$	0	0	$ss's''$	$s's''Z$	3, 4
68	$t'_1t''_3 - t'_3t''_1$	0	$ss's''$	0	$-s's''Y$	2, 4
69	$t'_2t''_3 - t'_3t''_2$	$ss's''$	0	0	$s's''X$	1, 4

Table A.11: Coupled tensor coefficients, for data coming from a single central camera, part II.

We may regroup the equations and split them in connected parts:

$$\begin{pmatrix} & 1 & -1 \\ 1 & & -1 \\ 1 & -1 & \end{pmatrix} \begin{pmatrix} 1 \\ 5 \\ 9 \end{pmatrix} = \mathbf{0}$$

$$\begin{pmatrix} & 1 & -1 \\ 1 & & -1 \\ 1 & -1 & \end{pmatrix} \begin{pmatrix} 10 \\ 14 \\ 18 \end{pmatrix} = \mathbf{0}$$

$$\begin{pmatrix} 1 & & 1 & & \\ & 1 & & & 1 \\ & & & 1 & \end{pmatrix} \begin{pmatrix} 23 \\ 24 \\ 25 \\ 27 \\ 28 \\ 29 \end{pmatrix} = \mathbf{0}$$

$$\begin{pmatrix} 1 & & 1 & & \\ & 1 & & & 1 \\ & & & 1 & \end{pmatrix} \begin{pmatrix} 32 \\ 33 \\ 34 \\ 36 \\ 37 \\ 38 \end{pmatrix} = \mathbf{0}$$

$$\begin{pmatrix} 1 & & 1 & & \\ & 1 & & & 1 \\ & & & 1 & \end{pmatrix} \begin{pmatrix} 41 \\ 42 \\ 43 \\ 45 \\ 46 \\ 47 \end{pmatrix} = \mathbf{0}$$

We conclude the following: the unknowns are determined up to $24 - 2 \times 2 - 3 \times 3 - 1 = 10$ degrees of freedom (1 for freedom of scale). We now consider the use of additional data, corresponding to rays passing through a second optical center.

A.4.2 Two central 3D cameras

For simplicity, we only consider tensor coefficients that were undetermined in the previous section.

We again consider already aligned points. Besides rays as in the previous section, we now consider additional rays passing through a second point. Let the second point be, without loss of generality:

$$\begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}$$

Triplets of points on rays going through that point, can be modeled as:

$$\begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} + \frac{1}{s} \begin{pmatrix} X \\ Y \\ Z \\ 0 \end{pmatrix} = \begin{pmatrix} X/s \\ Y/s \\ 1 + Z/s \\ 1 \end{pmatrix} \sim \begin{pmatrix} X \\ Y \\ s + Z \\ s \end{pmatrix}$$

We insert this in tables A.10 and A.11 (only for the coefficients that remained undetermined), which gives table A.12.

	T^1	T^2	T^3	T^4
1	0	$-ss''X - s''XZ$	$-s''XY$	0
5	$-ss''Y - s''YZ$	0	$s''XY$	0
9	$s's''Y + s''YZ$	$s's''X + s''XZ$	0	0
10	0	$ss'X + s'XZ$	$s'XY$	0
14	$ss'Y + s'YZ$	0	$-s'XY$	0
18	$-s's''Y - s'YZ$	$-s's''X - s'XZ$	0	0
23	0	0	sXY	$sXY + XYZ$
24	0	0	$ss''X + sXZ$	$ss''X + sXZ + s''XZ + XZ^2$
25	0	0	sXY	$sXY + XYZ$
27	0	0	$ss''Y + sYZ$	$ss''Y + sYZ + s''YZ + YZ^2$
28	0	0	$ss'X + sXZ$	$ss'X + sXZ + s'XZ + XZ^2$
29	0	0	$ss'Y + sYZ$	$ss'Y + sYZ + s'YZ + YZ^2$
32	0	sXY	0	$-XY^2$
33	0	$ss''X + sXZ$	0	$-s''XY - XYZ$
34	0	sXY	0	$-XY^2$
36	0	$ss''Y + sYZ$	0	$-s''Y^2 - Y^2Z$
37	0	$ss'X + sXZ$	0	$-s'XY - XYZ$
38	0	$ss'Y + sYZ$	0	$-s'Y^2 - Y^2Z$
41	sXY	0	0	X^2Y
42	$ss''X + sXZ$	0	0	$s''X^2 + X^2Z$
43	sXY	0	0	X^2Y
45	$ss''Y + sYZ$	0	0	$s''XY + XYZ$
46	$ss'X + sXZ$	0	0	$s'X^2 + X^2Z$
47	$ss'Y + sYZ$	0	0	$s'XY + XYZ$

Table A.12: Coupled tensor coefficients, for data coming from a second central camera.

Let us group terms with equal monomials in (X, Y, Z, s, s', s''), per tensor:

All coefficients that appear alone, must be zero: 24, 27, 28, 29, 36, 38, 42, 46

We remain with the following 16 coefficients: 1, 5, 9, 10, 14, 18, 23, 25, 32, 33, 34, 37, 41, 43, 45, 47.

We retain the equations given at the end of the previous section, and add the ones given in the tables above:

$$\left(\begin{array}{cccccccccccccccc}
 & 1 & -1 & & & & & & & & & & & & & \\
1 & & -1 & & & & & & & & & & & & & \\
1 & -1 & & & & & & & & & & & & & & \\
 & & & & 1 & -1 & & & & & & & & & & \\
 & & & 1 & & -1 & & & & & & & & & & \\
 & & & 1 & -1 & & & & & & & & & & & \\
 & & & & & & 1 & 1 & & & & & & & & \\
 & & & & & & & & 1 & & 1 & & & & & \\
 & & & & & & & & & 1 & & 1 & & & & \\
 & & & & & & & & & & & & 1 & 1 & & \\
 & & & & & & & & & & & & & & 1 & 1 \\
\hline
 & & & & 1 & & & & & & & & & & & 1 \\
 & -1 & & & & & & & & & & & & & 1 & \\
 & & 1 & & & -1 & & & & & & & & & & \\
 & & & 1 & & & & & & & 1 & & & & & \\
-1 & & & & & & & & 1 & & & & & & & \\
 & & & & & & & & & -1 & & & & & 1 & \\
 & & & & & & & & -1 & & & & & 1 & &
\end{array}\right)
\left(\begin{array}{c}
1 \\ 5 \\ 9 \\ 10 \\ 14 \\ 18 \\ 23 \\ 25 \\ 32 \\ 33 \\ 34 \\ 37 \\ 41 \\ 43 \\ 45 \\ 47
\end{array}\right) = \mathbf{0}$$

This is seen to be equivalent to:

$$\begin{aligned}
1 = 5 = 9 = 10 = 14 = 18 = 33 = -37 = -45 = -47 \\
23 = -25 \\
32 = 34 \\
41 = 43
\end{aligned}$$

We thus still have a 3-dof ambiguity (4 minus 1 for scale) on the remaining 16 unknown coefficients.

A.4.3 Three central 3D cameras

By repeating the same analysis with an additional, third central camera (not shown here), we find that the ambiguity disappears. In conclusion, a stereo system consisting of 3 or more central cameras, can be calibrated with the general algorithm.

A.5 Independence of Coordinate System

The degeneracy problems arise because of the nature of cameras and rays. An important observation is that the degeneracy problems are independent of the motion between the boards. Let $\mathbf{Q}$, $\mathbf{Q}'$ and $\mathbf{Q}''$ represent the 3D points, which lie on the same ray, in different coordinate systems. One important requirement is that the data must be completely general. For example if we consider a noncentral camera, then we should have at least three nonintersecting rays passing in different directions (to distinguish itself from parallel). Let

T^1	
$ss'X$	46
$ss'Y$	14+47
$ss''X$	42
$ss''Y$	45-5
sXY	41+43
sXZ	42+46
sYZ	45+47
$s's''Y$	9-18
$s'YZ$	14-18
$s''YZ$	9-5

T^2	
$ss'X$	10+37
$ss'Y$	38
$ss''X$	33-1
$ss''Y$	36
sXY	32+34
sXZ	33+37
sYZ	36+38
$s's''X$	9-18
$s'XZ$	10-18
$s''XZ$	9-1

T^3	
$ss'X$	28
$ss'Y$	29
$ss''X$	24
$ss''Y$	27
sXY	23+25
sXZ	24+28
sYZ	27+29
$s'XY$	10-14
$s''XY$	5-1

T^4	
$ss'X$	28
$ss'Y$	29
$ss''X$	24
$ss''Y$	27
sXY	23+25
sXZ	24+28
sYZ	27+29
$s'X^2$	46
$s'XY$	47-37
$s'XZ$	28
$s'Y^2$	-38
$s'YZ$	29
$s''X^2$	42
$s''XY$	45-33
$s''XZ$	24
$s''Y^2$	-36
$s''YZ$	27
X^2Y	41+43
X^2Z	42+46
XY^2	-32-34
XYZ	23+25-33-37+45+47
XZ^2	24+28
Y^2Z	-36-38
YZ^2	27+29

Figure A.1: Grouped coefficients of table A.12

.

$(\mathsf{R}', \mathbf{t}')$ be the motion between the first and second grids and let $(\mathsf{R}'', \mathbf{t}'')$ be the motion between the first and third grids. On applying the collinearity constraint we obtain the following equation.

$$\sum_{i,j,k=1}^{4} Q_i Q'_j Q''_k V_{ijk} = 0$$

$$\begin{bmatrix} Q_1 Q'_1 Q''_1 & . & . & Q_4 Q'_4 Q''_4 \\ . & . & . & . \end{bmatrix} \begin{bmatrix} V_{111} \\ . \\ . \\ V_{444} \end{bmatrix} = \mathbf{0}$$

$$\mathsf{A}\mathbf{V} = \mathbf{0}$$

where A is a matrix of size $n \times 64$ (where n is the number of rays and is much larger than 64) and V is a vector of length 64. From [104] we know that some of the coupled variables V_{ijk} are zeros. Let us assume that we have z zeros in the vector $\mathbf{V}$. We know that the rank of a matrix is the size of the largest nonsingular submatrix. Since the vector $\mathbf{V}$ is one of the solutions, the rank of the matrix A must be less than or equal to $64 - z - 1$. Now let us transform $\mathbf{Q}$, $\mathbf{Q}'$ and $\mathbf{Q}''$ individually to different coordinate systems $\bar{\mathbf{Q}}$, $\bar{\mathbf{Q}}'$ and $\bar{\mathbf{Q}}''$ respectively. Let the transformation parameters be given by $(\breve{\mathsf{R}}, \breve{\mathbf{t}})$, $(\breve{\mathsf{R}}', \breve{\mathbf{t}}')$ and $(\breve{\mathsf{R}}'', \breve{\mathbf{t}}'')$.

$$\bar{\mathbf{Q}} = [\breve{\mathsf{R}}\breve{\mathbf{t}}]\mathbf{Q}$$

$$\bar{\mathbf{Q}}' = [\breve{\mathsf{R}}'\breve{\mathbf{t}}']\mathbf{Q}'$$

$$\bar{\mathbf{Q}}'' = [\breve{\mathsf{R}}''\breve{\mathbf{t}}'']\mathbf{Q}''$$

Expressing in the transformed coordinate systems we have the following.

$$\bar{\mathsf{A}}\bar{\mathbf{V}} = 0$$

$$\begin{bmatrix} \bar{\mathbf{Q}}_1 \bar{\mathbf{Q}}'_1 \bar{\mathbf{Q}}''_1 & . & . & \bar{\mathbf{Q}}_4 \bar{\mathbf{Q}}'_4 \bar{\mathbf{Q}}''_4 \\ . & . & . & . \end{bmatrix} \begin{bmatrix} \bar{V}_{111} \\ . \\ . \\ \bar{V}_{444} \end{bmatrix} = 0$$

Similarly $\bar{\mathsf{A}}$ is a matrix of size $n \times 64$ and the rank is less than or equal to $64 - z - 1$.

$$\begin{bmatrix} (\breve{R}_{11}Q_1 + \breve{R}_{12}Q_2 + \breve{R}_{13}Q_3 + \breve{t}_1 Q_4)(\breve{R}'_{21}Q'_1 + \breve{R}'_{22}Q'_2 + \breve{R}'_{23}Q'_3 + \breve{t}'_1 Q'_4)(\breve{R}''_{31}Q''_1 + \breve{R}''_{32}Q''_2 + \breve{R}''_{33}Q''_3 + \breve{t}''_1 Q''_4 & . & . & Q_4 Q'_4 Q''_4 \\ . & & & \end{bmatrix}$$

By column operations we can transform the matrix A to $\bar{\mathsf{A}}$. If the columns are multiplied by nonzero scalars and added with other columns, the rank of the matrix does not get affected. However we are not sure whether the scalars (individual elements in $(\breve{\mathsf{R}}, \breve{\mathbf{t}})$, $(\breve{\mathsf{R}}', \breve{\mathbf{t}}')$ and $(\breve{\mathsf{R}}'', \breve{\mathbf{t}}'')$) are all nonzeros. So the rank of $\bar{\mathsf{A}}$ must be less than or equal to A. The coordinate transformations are always reversible when we have atleast three noncollinear points. Thus by scalar multiplication and addition we can also transform $\bar{\mathsf{A}}$ to A. By applying the previous argument we can say that the rank of A must be less than or equal to the rank of $\bar{\mathsf{A}}$. This can be possible only if the rank of A and $\bar{\mathsf{A}}$ are same.

Now let us look at the technique we use to extract motion from the constraint $\mathsf{A}\mathbf{V} = \mathbf{0}$. We remove the columns in A corresponding to zero entries in $\mathbf{V}$. We solve the system for the rest of the nonzero coupled variables. Finally we extract individual rotation and translation elements as given in [104]. We can see that these columns do not contribute to the rank of a matrix. Both $\mathbf{V}$ and $\bar{\mathbf{V}}$ have the same number of zero entries in the same locations in the coupled variables. Hence by removing the corresponding columns in $\mathbf{A}$ and $\bar{\mathbf{A}}$ we maintain the rank equality of the two matrices. This in other words proves that the rank deficiency issues are unaffected by the reversible coordinate transformations.

Bibliography

[1] E.H. Adelson and J.R. Bergen. The plenoptic function and the elements of early vision. *Computational Models of Visual Processing*, pages 3–20, 1991.

[2] D.G. Aliaga. Accurate catadioptric calibration for real-time pose estimation in room-size environments. In *International Conference on Computer Vision (ICCV)*, pages 127–134, 2001.

[3] S. Baker and S. Nayar. A theory of catadioptric image formation. In *International Conference on Computer Vision (ICCV)*, pages 35–42, 1998.

[4] H. Bakstein. Non-central cameras for 3d reconstruction. In *Research Report CTU-CMP-2001-21*, 2001.

[5] H. Bakstein and T. Pajdla. An overview of non-central cameras. In *Computer Vision Winter Workshop*, pages 223–233, 2001.

[6] H. Bakstein and T. Pajdla. Panoramic mosaicing with a 180 field of view lens. In *IEEE Workshop on omnidirectional vision*, 2002.

[7] J.P. Barreto and H. Araujo. Geometric properties of central catadioptric line images. In *European Conference on Computer Vision (ECCV)*, volume 2353, pages 237–251. Lecture notes in computer science, 2002.

[8] A. Bartoli. On the non-linear optimization of projective motion using minimal parameters. In ECCV, 2002.

[9] S.S. Beauchemin, R. Bajcsy, and G. Givaty. A unified procedure for calibrating intrinsic parameters of fish-eye lenses. In *Vision Interface*, pages 272–279, 79.

[10] S. Bogner. Introduction to panoramic imaging. In *IEEE SMC*, volume 54, pages 3100–3106, 1995.

[11] M. Born and E. Wolf. *Principles of Optics*. Pergamon Press, 1965.

[12] T. Brodsky, C. Fermüller, and Y. Aloimonos. Directions of motion fields are hardly ever ambiquous. In *European Conference on Computer Vision (ECCV)*, volume 2, pages 110–128, 1996.

[13] D.C. Brown. Close-range camera calibration. In *Photogrammetric Engineering*, volume 37(8), pages 855–866, 1971.

[14] J. W. Bruce, P. J. Giblin, and C. G. Gibson. On caustics of plane curves. *American Mathematical Monthly*, 88:651–667, 1981.

[15] D.G. Burkhard and D.L. Shealy. Flux density for ray propagation in geometrical optics. *Journal of the Optical Society of America*, 63(2):299–304, 1973.

[16] P. Chang and M. Hébert. Omni-directional structure from motion. In *IEEE Workshop on Omnidirectional vision*, pages 127–133, 2000.

[17] J. Charles, R. Reeves, and C. Schur. How to build and use an all-sky camera. In *Astronomy magazine*, 1987.

[18] C.S. Chen and W.Y. Chang. On pose recovery for generalized visual sensors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 26(7):848–861, 2004.

[19] S.E. Chen. Quicktime VR - an image based approach to virtual environment navigation. In *SIGGRAPH*, pages 29–38, 1995.

[20] D. Claus and A. Fitzgibbon. A rational function lens distortion model for general cameras. In *International Conference on Computer Vision*, volume 1, pages 213 – 219, 2005.

[21] R. Descartes and D. Smith. *The geometry of René Descartes.* Dover Publ.: New York. Originally published in Discours de la Méthode, 1637.

[22] F. Devernay and O. Faugeras. Straight lines have to be straight. *Machine Vision and Applications*, 13:14–24, 2001.

[23] P. Doubek and T. Svoboda. Reliable 3d reconstruction from a few catadioptric images. In *OMNIVIS*, pages 71–78, 2002.

[24] A. Dürer. *Underweysung der Messung (Instruction in measurement), Book with more than 150 woodcuts*. 1525.

[25] J. Fabrizio, J.P. Tarel, and R. Benosman. Calibration of panoramic catadioptric sensors made easier. In *Workshop on Omnidirectional vision*, pages 45–52, 2002.

[26] H. Farid and A.C. Popescu. Blind removal of image non-linearities. In *International Conference on Computer Vision (ICCV)*, volume 1, pages 76–81, 2001.

[27] O. Faugeras. What can be seen in three dimensions with an uncalibrated stereo rig. In *European Conference on Computer Vision (ECCV)*, pages 568–578. Springer-Verlag, 1992.

[28] O. Faugeras and Q.T. Luong. *The Geometry of Multiple Images*. The MIT Press, 2001.

[29] O. Faugeras and B. Mourrain. On the geometry and algebra of the point and line correspondences between n images. In *International conference on computer vision*, pages 951–6, 1995.

[30] O. Faugeras and G. Toscani. The calibration problem for stereo. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15–20, 1986.

[31] D. Feldman, T. Pajdla, and D. Weinshall. On the epipolar geometry of the crossed-slits projection. In *International Conference on Computer Vision (ICCV)*, 2003.

[32] S. Finsterwalder. Die geometrischen Grundlagen der Photogrammetrie. *Jahresbericht Deutscher Mathematik*, 6:1–44, 1899.

[33] A.W. Fitzgibbon. Simultaneous linear estimation of multiple view geometry and lens distortion. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001.

[34] M.M. Fleck. The wrong imaging model. *Technical Report TR 95-01, University of Iowa*, 1995.

[35] C. Geyer and K. Daniilidis. Catadioptric camera calibration. In *International Conference on Computer Vision (ICCV)*, pages 398–404, 1999.

[36] C. Geyer and K. Daniilidis. A unifying theory of central panoramic systems and practical implications. In *European Conference on Computer Vision (ECCV)*, pages 159–179, 2000.

[37] C. Geyer and K. Daniilidis. Structure and motion from uncalibrated catadioptric views. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 279–286, 2001.

[38] C. Geyer and K. Daniilidis. Paracatadioptric camera calibration. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, volume 24, pages 687–695, 2002.

[39] C. Geyer and K. Daniilidis. Mirrors in motion: Epipolar geometry and motion estimation. In *International Conference on Computer Vision (ICCV)*, pages 766–773, 2003.

[40] S.J. Gortler, R. Grzeszczuk, R. Szeliski, and M.F. Cohen. The Lumigraph. In *SIGGRAPH*, pages 43–54, 1996.

[41] M.D. Grossberg and S.K. Nayar. A general imaging model and a method for finding its parameters. In *International Conference on Computer Vision (ICCV)*, volume 2, pages 108–115, 2001.

[42] R. Gupta and R.I. Hartley. Linear pushbroom cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 1997.

[43] M.A. Halstead, B.A. Barsky, S.A. Klein, and R.B. Mandell. Reconstructing curved surfaces from specular reflection patterns using spline surface fitting of normals. In *ACM SIGGRAPH*, pages 335–342, 1996.

[44] 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 (IJCV)*, 13(3):331–356, 1994.

[45] C. Harris and M. Stephens. A combined corner and edge detector. In *Fourth Alvey Vision Conference*, pages 147–151, 1988.

[46] R. Hartley, R. Gupta, and T. Chang. Stereo from uncalibrated cameras. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 761–764, 1992.

[47] R. Hartley and S.B. Kang. A parameter free method for estimating radial distortion. In *International Conference on Computer Vision (ICCV)*, 2005.

[48] R.I. Hartley and P. Sturm. Triangulation. *CVIU*, 68(2):146–157, 1997.

[49] R.I. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, 2000.

[50] R.A. Hicks and R. Bajcsy. Catadioptric sensors that approximate wide-angle perspective projections. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 545–551, 2000.

[51] J. Hong. Image based homing. In *International Conference on Robotics and Automation*, 1991.

[52] H. Ishiguro, M. Yamamoto, and S. Tsuji. Omni-directional stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 14(2):257–262, 1992.

[53] S.B. Kang. Catadioptric self-calibration. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 201–207, 2000.

[54] S.B. Kang and R. Szeliski. 3-d scene data recovery using omnidirectional multibaseline stereo. *International Journal of Computer Vision (IJCV)*, 25(2), 1997.

[55] A. Krishnan and N. Ahuja. Panoramic image acquisition. In *Computer Vision and Pattern Recognition (CVPR)*, pages 379–384, 1996.

[56] M. Levoy and P. Hanrahan. Light field rendering. In *SIGGRAPH*, pages 31–42, 1996.

[57] S.S. Lin and R. Bajczy. True single view cone mirror omnidirectional catadioptric system. In *International conference on Computer Vision (ICCV)*, volume 2, pages 102–107, 2001.

[58] H.C. Longuet-Higgins. A computer program for reconstructing a scene from two projections. *Nature*, pages 133–135, 1981.

[59] D.G. Lowe. Object recognition from local scale-invariant features. In *International Conference on Computer Vision (ICCV)*, pages 1150–1157, 1999.

[60] L. McMillan and G. Bishop. Plenoptic modeling: An image-based rendering system. In *SIGGRAPH*, pages 39–46, 1995.

[61] J. Mellor. Geometry and texture from thousands of images. *International Journal of Computer Vision (IJCV)*, 51(1), 2003.

[62] B. Micusik and T. Pajdla. Estimation of omnidirectional camera model from epipolar geometry. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 485–490, 2003.

[63] B. Micusik and T. Pajdla. Autocalibration and 3d reconstruction with non-central catadioptric cameras. In *Computer Vision and Pattern Recognition*, pages 748–753, 2004.

[64] K. Miyamoto. Fish eye lens. *Journal of Optical Society of America*, 54(8):1060–1061, 1964.

[65] T. Moons, L. Van Gool, M. van Diest, and E. Pauwels. Affine reconstruction from perspective image pairs. In *Workshop on Applications of Invariants in Computer Vision, Azores*, pages 249–266, 1993.

[66] J. R. Murphy. Aplication of panoramic imaging to a teleoperated lunar rover. In *IEEE SMC Conference*, pages 3117–3121, 1995.

[67] V. Nalwa. A true omnidirectional viewer. In *Technical report, Bell Laboratories, Holmdel, NJ, USA*, 1996.

[68] S.G. Narasimhan, S.K. Nayar, B. Sun, and S.J. Koppal. Structured light in scattering media. In *IEEE International Conference on Computer Vision (ICCV)*, volume 1, pages 420–427, 2005.

[69] S.K. Nayar. Catadioptric omnidirectional camera. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 482–488, 1997.

[70] S.K. Nayar. Catadioptric omnidirectional camera. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 482–488, 1997.

[71] S.K. Nayar, V. Branzoi, and T.E. Boult. Programmable imaging using a digital micromirror array. In *International Conference on Computer Vision and Pattern Recognition*, pages 436–443, 2004.

[72] S.A. Nene and S.K. Nayar. Stereo with mirrors. In *International Conference on Computer Vision (ICCV)*, volume 2, pages 1087–1094, 1997.

[73] J. Neumann, C. Fermüller, and Y. Aloimonos. Polydioptric camera design and 3d motion estimation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 294–301, 2003.

[74] D. Nistér. An efficient solution to the five-point relative pose problem. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 26, pages 756–770, 2003.

[75] D. Nistér. A minimal solution to the generalized 3-point pose problem. In Conference on Computer Vision and Pattern Recognition, Washington, USA, pages 560–567, 2004.

[76] D. Nistér, H. Stewenius, and E. Grossmann. Non-parametric self-calibration. In *International Conference on Computer Vision (ICCV)*, 2005.

[77] Opencv (open source computer vision library). Intel, www.intel.com/research/mrl/research/opencv/.

[78] T. Pajdla. Stereo with oblique cameras. In *International Journal of Computer Vision (IJCV)*, volume 47(1), 2002.

[79] S. Peleg, M. Ben-Ezra, and Y. Pritch. Omnistereo: Panoramic stereo imaging. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, pages 279–290, 2001.

[80] S. Peleg, Y.Pritch, and M. Ben-Ezra. Cameras for stereo panoramic imaging. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 208–214, 2000.

[81] R. Pless. Using many cameras as one. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 587–594, 2003.

[82] M. Pollefeys and L. Van Gool. Some issues on self-calibration and critical motion sequences. *ACCV*, pages 893–898, 2000.

[83] S. Ramalingam and S.K. Lodha. Adaptive enhancement of 3D scenes using hierarchical registration of texture mapped models. In *3DIM*, 2003.

[84] S. Ramalingam, P. Sturm, and E. Boyer. A factorization based self-calibration for radially symmetric cameras. In *Third International Symposium on 3D Data Processing, Visualization and Transmission*, 2006.

[85] S. Ramalingam, P. Sturm, and S.K. Lodha. Generic calibration of axial cameras. *INRIA Research Report*, 2005.

[86] S. Ramalingam, P. Sturm, and S.K. Lodha. Towards complete generic camera calibration. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[87] S. Ramalingam, P. Sturm, and S.K. Lodha. Towards generic self-calibration of central cameras. In *OMNIVIS*, 2005.

[88] S. Ramalingam, P. Sturm, and S.K. Lodha. Theory and calibration algorithms for axial cameras. In *ACCV*, 2006.

[89] D.W. Rees. Panoramic television viewing system. In *United States Patent (3,505,465)*, 1970.

[90] J. Salvi, J. Pages, and J. Batlle. Pattern codification strategies in structured light systems. In *Pattern Recognition*, volume 34(7), pages 827–849, 2004.

[91] S. Seitz. The space of all stereo images. In *International Conference on Computer Vision (ICCV)*, volume 1, pages 26–33, 2001.

[92] S. Shah and 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.

[93] A. Shashua. Projective functions for recognition. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 1995.

[94] A. Shashua and L. Wolf. Homography tensors: On algebraic entities that represent three views of static or moving planar points. In *European Conference on Computer Vision (ECCV)*, 2000.

[95] H.Y. Shum, A. Kalai, and S.M. Seitz. Omnivergent stereo. In *International Conference on Computer Vision (ICCV)*, pages 22–29, 1999.

[96] J. Slater. Photography with the whole sky lens. *American Photographer*, 1932.

[97] P. Sturm. Critical motion sequences for monocular self-calibration and uncalibrated euclidean reconstruction. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1100–1105, 1997.

[98] P. Sturm. Vision 3d non-calibrée : contributions à la reconstruction projective et étude des mouvements critiques pour l'auto-calibrage. *Ph.D. Thesis, INP de Grenoble, France*, 1997.

[99] P. Sturm. A method for 3d reconstruction of piecewise planar objects from single panoramic images. In *IEEE workshop on omnidirectional vision*, 2000.

[100] P. Sturm. Mixing catadioptric and perspective cameras. In *OMNIVIS*, pages 60–67, 2002.

[101] P. Sturm. Multi-view geometry for general camera models. In *Computer Vision and Pattern Recogntion (CVPR)*, 2005.

[102] P. Sturm, Z. Cheng, P.C.Y. Chen, and A.N. Poo. Focal length calibration from two views: method and analysis of singular cases. In *CVIU*, 2005.

[103] P. Sturm and S. Maybank. On plane-based camera calibration. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 432–437, 1999.

[104] P. Sturm and S. Ramalingam. A generic calibration concept: Theory and algorithms. Technical Report 5058, INRIA, 2003.

[105] P. Sturm and S. Ramalingam. A generic concept for camera calibration. In *European Conference on Computer Vision (ECCV)*, volume 2, pages 1–13, 2004.

[106] P. Sturm and B. Triggs. A factorization based algorithm for multi-image projective structure and motion. In *European Conference on Computer Vision (ECCV)*, pages 709–720, 1996.

[107] Tomas Svoboda and Tomas Pajdla. Epipolar geometry for central catadioptric cameras. *International Journal of Computer Vision*, 49(1):23–27, 2002.

[108] R. Swaminathan, M.D. Grossberg, and S.K. Nayar. A perspective on distortions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, page 594, 2003.

[109] R. Swaminathan, M.D. Grossberg, and S.K. Nayar. Designing mirrors for catadioptric systems that minimize image errors. In *IEEE workshop on Omnivis*, 2004.

[110] R. Swaminathan and S. Nayar. Non-metric calibration of wide-angle lenses and polycameras. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 1999.

[111] R. Swaminathan and S.K. Nayar. Non-metric calibration of wide-angle lenses and polycameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 22(10):1172–1178, 2000.

[112] J.P. Tardif and P. Sturm. Calibration of cameras with radially symmetric distortion. In *OMNIVIS*, 2005.

[113] J.P. Tardif, P. Sturm, and S. Roy. Self-calibration of a general radially symmetric distortion model. In *European Conference on Computer Vision*, 2006.

[114] S. Thirthala and M. Pollefeys. 1D radial cameras and its application to omnidirectional geometry. In *International Conference on Computer Vision (ICCV)*, 2005.

[115] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. In *International Journal of Computer Vision (IJCV)*, volume 9, pages 137–154, 1992.

[116] B. Triggs. Matching constraints and the joint image. In *International conference on computer vision*, pages 338–43, 1995.

[117] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment – A modern synthesis. *Workshop on Vision Algorithms: Theory and Practice*, pages 298–375, 2000.

[118] R. Tsai. An efficient and accurate camera calibration technique for 3d machine vision. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 368–374, june 1986.

[119] Y. Wexler, A.W. Fitzgibbon, and A. Zisserman. Learning epipolar geometry from image sequences. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 209–216, 2003.

[120] D. Wood, A. Finkelstein, J.F. Hughes, C.E. Thayer, and D.H. Salesin. Multiperspective panoramas for cell animation. In *SIGGRAPH*, pages 243–250, 1997.

[121] R.W. Wood. Fish-eye view, and vision under water. *Philosophical Magazine*, 12:159–162, 1902.

[122] Y. Xiong and K. Turkowski. Creating image-based VR using a self-calibrating fisheye lens. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 237–243, 1997.

[123] Y. Yagi and S. Kawato. Panoramic scene analysis with conic projection. In *International Conference on Robots and Systems (IROS)*, 1990.

[124] K. Yamazawa, Y. Yagi, and M. Yachida. Obstacle avoidance with omnidirectional image sensor hyperomni vision. In *International Conference on Robotics and Automation*, pages 1062–1067, 1995.

[125] X. Ying and Z. Hu. Can we consider central catadioptric cameras and fisheye cameras within a unified imaging model. In *European Conference on Computer Vision (ECCV)*, pages 442–355, 2004.

[126] Z. Zhang. On the epipolar geometry between two images with lens distortion. In *International conference on Pattern Recognition (ICPR)*, volume 1, pages 407–411, 1996.

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

[128] J. Y. Zheng and S. Tsuji. Panoramic representation of scenes for route understanding. In *International Conference on Pattern Recognition*, volume 1, pages 161–167, 1990.