UNIVERSITÉ BLAISE PASCAL – CLERMONT II

Ecole Doctorale Sciences Pour l'Ingénieur

HABILITATION À DIRIGER DES RECHERCHES

Préparée au LASMEA, UMR 6602 CNRS / Université Blaise Pascal
(Laboratoire des Sciences des Matériaux pour l'Electronique, et d'Automatique)

Spécialité : Vision par Ordinateur

Présentée et soutenue publiquement par

Adrien Bartoli

le 9 juin 2008

Contributions au recalage d'images et à la reconstruction 3D de scènes rigides et déformables

—— Recueil d'articles publiés sur 2004 – 2007 ——

Contributions to Image Registration and to the 3D Reconstruction of Rigid and Deformable Scenes

—— Collected Publications over 2004 – 2007 ——

Devant le jury composé de

Président:	Michel Dhome	CNRS (Centre National de la Recherche Scientifique) – LASMEA
Rapporteurs externes:	Pascal Fua	EPFL (Ecole Polytechnique Fédérale de Lausanne)
	Richard Hartley	ANU (the Australian National University)
	Nikos Paragios	ECP (Ecole Centrale de Paris)
Rapporteur interne:	Jean-Marc Lavest	UdA (Université d'Auvergne) – LASMEA
Examinateurs:	Mads Nielsen	DIKU (Datalogisk Institut, Københavns Universitet)
	Marc Pollefeys	ETHZ (Eldgenössische Technische Hochschule Zürich) et UNC
	Jean Ponce	ENS (Ecole Normale Supérieure) et UIUC

Avant-propos

Ce document est un recueil d'articles que j'ai écrits ou co-écrits sur la période 2004 – 2007. Il accompagne un document de synthèse de mes travaux et activités scientifiques sur cette même période. Sa structure est identique à celle des chapitres 6 à 9 de ce document de synthèse : il y a une correspondance stricte entre les chapitres, sections et sous-sections des deux documents. J'indique au début de chaque section la liste des articles inclus, et précise pour chacun la sous-section qui lui est consacrée.

Forewords

This document is a collection of the most significant articles I have authored or co-authored between the years 2004 – 2007. It comes as a companion to my habilitation thesis, which summarises my scientific work and activity over this period. Its structure is similar to that of chapters 6 to 9 in the thesis, where the chapters, sections and sub-sections follow an identical format. The list of included articles is given at the beginning of each section, with the sub-section corresponding to each article.

Contents

CHAPTER

6

IMAGE REGISTRATION

6.1 Photometry in Direct Image Registration

V01 **Direct Image Registration With Gain and Bias** §6.1.1
A. Bartoli
Topics in Automatic 3D Modeling and Processing Workshop, Verona, Italy, March 2006

J12 **Groupwise Geometric and Photometric Direct Image Registration** §6.1.2
A. Bartoli
IEEE *Transactions on Pattern Analysis and Machine Intelligence*, accepted December 2007
Previous version: [I28]
Related paper: [I44]

I31 **Shadow Resistant Direct Image Registration** §6.1.3
D. Pizarro and A. Bartoli
SCIA'07 - *Scandinavian Conf. on Image Analysis*, Aalborg, Denmark, June 2007

6.1.1 Paper (LIMA3D'06) – *Direct Image Registration With Gain and Bias*

Direct Image Registration With Gain and Bias

Adrien Bartoli ◇ Adrien.Bartoli@gmail.com
CNRS – LASMEA ◇ Clermont-Ferrand, France

Abstract

Image registration consists in estimating geometric and photometric transformations that align a template and an image as best as possible. The direct approach consists in minimizing the intensity discrepancy between the aligned template and image. The inverse compositional algorithm has been recently proposed for the direct estimation of groupwise geometric transformations. It is efficient in that it performs most computationally expensive calculations at the pre-computation phase.

We propose the gain and bias inverse compositional algorithm which estimates, along with the geometric transformation, a photometric one modeling for example global lighting change. Our algorithm preserves the efficient pre-computation-based design of the original inverse compositional one. Previous attempts at incorporating appearance variations to the inverse compositional algorithm spoils this property.

We report experimental results on simulated and real data, showing the improvement in computational efficiency of our algorithm compared to previous ones.

1. Introduction

Image registration is the task of applying some transformations to a template and / or an image so that they match as best as possible. This can be seen as the computation of some geometric transformation, for example an homography, used to deform the image to model camera pose, and some photometric transformation, applied to the pixel intensities, for example gain and bias to model global lighting.

Image registration has been an important research topic for the past decades. It is central to many tasks in computer vision, medical imaging, augmented reality and robotics.

Broadly speaking, two approaches have been proposed: the feature-based and the direct approaches. The feature-based approach, see *e.g.* [8], relies on abstracting the input images by the geometric location of a set of carefully chosen, salient features. The direct approach, see *e.g.* [6], uses the intensity of all pixels in the region of interest.

This paper focuses on the direct approach, and brings as its main contribution a computationally efficient registration algorithm dealing with gain and bias, based on the inverse compositional principle of Baker *et al.* [2].

The geometric registration problem is the minimization of a nonlinear least squares error function, given by the discrepancy in pixel intensities, between the template $\mathcal{T}$ and the image $\mathcal{I}$, warped onto the template by the geometric transformation to be estimated. The geometric transformation, denoted $\mathcal{G}$, maps a pixel $\mathbf{q}$ in the region of interest $\mathcal{R}$ defined in the template to the corresponding pixel $\mathcal{G}(\mathbf{q};\mathbf{g})$ in the image. Vector $\mathbf{g}$ encapsulates its parameters. We expect that given an 'appropriate' parameter vector $\mathbf{g}$, $\mathcal{T}[\mathbf{q}]$ is 'close to' $\mathcal{I}[\mathcal{G}(\mathbf{q};\mathbf{g})]$, for all $\mathbf{q} \in \mathcal{R}$. The direct image registration problem is thus formally posed as:

$$\min_{\mathbf{g}} \sum_{\mathbf{q}\in\mathcal{R}} (\mathcal{T}[\mathbf{q}] - \mathcal{I}[\mathcal{G}(\mathbf{q};\mathbf{g})])^2. \tag{1}$$

Note that other error functions can be used, to deal for example with outliers, see *e.g.* [4]. Most algorithms linearize each term in the transformation parameters $\mathbf{g}$, and iteratively update an initial guess by solving linear least squares problems. The popular Lucas-Kanade algorithm [7] and work by Bergen *et al.* [3] fall into this category. Baker *et al.* [2] have recently proposed an efficient algorithm for solving problem (1), the *inverse compositional algorithm*, using a Gauss-Newton, local approximation to the error function. The efficiency stems from the fact that the Hessian matrix[1] involved in the normal equations is constant. Its inverse can thus be pre-computed.

The above-derived formulation (1) suffers from the fact that it does not take into account photometric changes, *i.e.* changes in the intensity of the pixels. These changes occur for example when the lighting changes between acquisition of the template and the image. They are modeled by a transformation $\mathcal{P}$ with parameter vector $\mathbf{p}$, and give rise to the following minimization problem:

$$\min_{\mathbf{g},\mathbf{p}} \sum_{\mathbf{q}\in\mathcal{R}} (\mathcal{P}(\mathcal{T}[\mathbf{q}];\mathbf{p}) - \mathcal{I}[\mathcal{G}(\mathbf{q};\mathbf{g})])^2. \tag{2}$$

[1] We use the expression 'Hessian matrix' for the Gauss-Newton approximation to the true Hessian matrix.

The photometric transformation is typically chosen as an affine transformation modeling gain and bias, and accounting for global intensity changes between the template and the image:

$$\mathcal{P}(v;\mathbf{p}) = av + b \qquad \text{with} \qquad \mathbf{p}^\mathsf{T} = (a \;\; b). \tag{3}$$

The contribution of this paper is an efficient method for solving problem (2), the registration problem with gain and bias. The proposed method is dubbed the *gain and bias inverse compositional algorithm*. It is based on the inverse compositional approach of Baker *et al.* [2] and is thus applicable to the registration of images related by groupwise geometric transformations such as homographies.

Estimating gain and bias jointly with geometric registration parameters makes the Hessian matrix vary accross the iterations. Previous work thus re-estimate and invert it at each iteration: this is the *simultaneous inverse compositional algorithm* of Baker *et al.* [1], which not only deals with gain and bias but also with general linear appearance variations.

We show that the Hessian matrix has a strong block structure with blocks constant up to some scale factors, depending on the gain. From this analysis, we derive an algorithm allowing us to pre-compute a block-wise inverse of the Hessian matrix. The normal equations are then solved by simply multiplying the right hand side by some constant, appropriately rescaled matrices, which is very efficient in terms of computational cost. We underline that our algorithm performs exactly the same calculations as the simultaneous inverse compositional algorithm does. Experimental results show that the computational cost is reduced by factors of at least 2.

Paper organization. We introduce background material, namely the inverse compositional and the simultaneous inverse compositional algorithms of Baker *et al.* in §2. We present our gain and bias inverse compositional algorithm in §3. We report experimental results on simulated and real data in §4. A discussion is provided in §5. The parameterization of homographic warps is detailled in §A.

Notation. Vectors are denoted using bold fonts, *e.g.* $\mathbf{q}$, matrices using sans-serif fonts, *e.g.* E, and scalars in italics, *e.g.* a. We deal with grey-level images only: the template and image, respectively denoted $\mathcal{T}$ and $\mathcal{I}$, are seen as functions from $\mathbb{R}^2$ to $\mathbb{R}$. For instance, $\mathcal{T}[\mathbf{q}]$ is the intensity at location $\mathbf{q} \in \mathbb{R}^2$. Bilinear interpolation is used for subpixel coordinates. The geometric and photometric transformations are respectively denoted $\mathcal{G}$ and $\mathcal{P}$, with respective parameter vectors $\mathbf{g}$ and $\mathbf{p}$. The geometric transformation is also called the warp.

2. The Inverse Compositional Algorithm

Baker *et al.* have recently published a series of five papers on direct image registration. In the first one [2], they propose the efficient *inverse compositional algorithm*. Baker *et al.* show in [1] that the efficiency is lost if appearance variations, in particular gain and bias transformations, are incorporated in the general algorithm, making it much more computationally expensive. Below, we describe this algorithm in details since it forms the basis for the one we propose.

2.1. Principle

The inverse compositional algorithm is an iterative procedure with, as is often the case for registration algorithms, three main steps in its inner loop:

1. **Image warping.** Warp the image on the template using the current warp parameters $\mathbf{g}$.
2. **Local registration.** Compute the local warp parameters $\boldsymbol{\delta}_g$ between the warped image and the template.
3. **Warp updating.** Update the current warp parameters by composing the current warp with the inverse of the local warp.

The main advantages of this method is that it converges rapidly, and is computationally cheap since computationally demanding calculations are pre-computed.

2.2. Geometric Registration

The algorithm is summarized in table 1. Let $\tilde{\mathcal{I}}$ be the warped image, *i.e.* $\tilde{\mathcal{I}}[\mathbf{q}] = \mathcal{I}[\mathcal{G}(\mathbf{q};\mathbf{g})]$. The geometric registration problem (1) is rewritten as:

$$\min_{\boldsymbol{\delta}_g} \sum_{\mathbf{q}\in\mathcal{R}} (\mathcal{T}[\mathcal{G}(\mathbf{q};\boldsymbol{\delta}_g] - \tilde{\mathcal{I}}[\mathbf{q}])^2. \tag{4}$$

Vector $\boldsymbol{\delta}_g$ represents the parameters of the local geometric transformation. The error function in problem (4) is linearized by first order Taylor expansion in $\boldsymbol{\delta}_g$ to form a Gauss-Newton approximation, giving, using the chain rule:

$$\min_{\boldsymbol{\delta}_g} \sum_{\mathbf{q}\in\mathcal{R}} (\mathcal{T}[\mathbf{q}] + \nabla\mathcal{T}[\mathbf{q}]^\mathsf{T} \left.\frac{\partial\mathcal{G}}{\partial\mathbf{g}}\right|_{\mathbf{q};\tilde{\mathbf{g}}} \boldsymbol{\delta}_g - \tilde{\mathcal{I}}[\mathbf{q}])^2,$$

where $\nabla\mathcal{T}[\mathbf{q}]$ is the (2×1) template gradient at $\mathbf{q}$ and $\left.\frac{\partial\mathcal{G}}{\partial\mathbf{g}}\right|_{\mathbf{q};\tilde{\mathbf{g}}}$ is the Jacobian of the warp, evaluated at $\mathbf{q}$ and at warp parameters $\tilde{\mathbf{g}}$, representing the identity warp[2]. The advantage of this formulation is that the partial derivatives

[2]The warp is generally parameterized such that $\tilde{\mathbf{g}} = \mathbf{0}$.

of the error function are constant, and so are the gradient vectors:

$$\ell_{\mathbf{q}}^{\mathsf{T}} = \nabla \mathcal{T}[\mathbf{q}]^{\mathsf{T}} \left. \frac{\partial \mathcal{G}}{\partial \mathbf{g}} \right|_{\mathbf{q};\tilde{\mathbf{g}}}. \quad (5)$$

The normal equations induced by the linear least squares minimization problem (4) are:

$$\underbrace{\left(\sum_{\mathbf{q} \in \mathcal{R}} \ell_{\mathbf{q}} \ell_{\mathbf{q}}^{\mathsf{T}} \right)}_{\mathsf{E}_g} \boldsymbol{\delta}_g = \underbrace{\left(\sum_{\mathbf{q} \in \mathcal{R}} \ell_{\mathbf{q}} (\tilde{\mathcal{I}}[\mathbf{q}] - \mathcal{T}[\mathbf{q}]) \right)}_{\mathbf{b}_g}.$$

The solution $\boldsymbol{\delta}_g = \mathsf{E}_g^{-1} \mathbf{b}_g$ for the local warp parameters can thus be computed very efficiently since the inverse of the constant Hessian matrix E_g can be pre-computed, as well as the gradient vectors $\ell_{\mathbf{q}}$.

Once $\boldsymbol{\delta}_g$ has been computed, the current warp parameters $\mathbf{g}$ are updated by composing the current warp with the inverse of the local warp. If one uses an homographic warp for example, then the updated parameters are given by multiplying the current homography by the inverse of the local one as detailled in §A. We write the warp update rule as:

$$\mathbf{g} \leftarrow \mathcal{U}_g(\mathbf{g}, \boldsymbol{\delta}_g).$$

The process is iterated until convergence, determined, in our experiments, by thresholding the two-norm of $\boldsymbol{\delta}_g$ by $\varepsilon = 10\mathrm{e} - 8$, or when the update increases the error. In the latter case, the last update is cancelled before stopping the iterations.

2.3. Incorporating Gain and Bias

We incorporate the global, affine illumination variation model (3), referred to as gain and bias. The registration algorithm is summarized in table 2. Applying the photometric transformation to the template or to the image does not lead to exactly the same error function. They are equivalent, up to resampling issues, only when the geometric alignment is the correct one. In both cases, the Hessian matrix is not constant, thus spoiling the main advantage of the inverse compositional approach. We apply the photometric transformation to the template since it leads us to an efficient minimization algorithm in §3.

We rewrite problem (2) as:

$$\min_{\mathbf{g},\mathbf{p}} \sum_{\mathbf{q} \in \mathcal{R}} (a\mathcal{T}[\mathbf{q}] + b - \mathcal{I}[\mathcal{G}(\mathbf{q};\mathbf{g})])^2. \quad (6)$$

Using an additive update rule for the photometric parameters, *i.e.* $\mathbf{p} \leftarrow \mathbf{p} + \boldsymbol{\delta}_p$, and the inverse compositional trick for the geometric parameters $\mathbf{g}$, we transform problem (6) to:

$$\min_{\boldsymbol{\delta}_g,\boldsymbol{\delta}_p} \sum_{\mathbf{q} \in \mathcal{R}} ((a + \delta_a)\mathcal{T}[\mathcal{G}(\mathbf{q};\boldsymbol{\delta}_g)] + b + \delta_b - \mathcal{I}[\mathcal{G}(\mathbf{q};\mathbf{g})])^2.$$

OBJECTIVE

Register an image $\mathcal{I}$ to a template $\mathcal{T}$ by computing the parameters $\mathbf{g}$ of a warp $\mathcal{G}(\mathbf{q};\mathbf{g})$ by minimizing the intensity error. Other inputs are the region of interest $\mathcal{R}$ in the template and an initial value for $\mathbf{g}$.

ALGORITHM

Pre-computations

1. Compute the gradient vectors $\ell_{\mathbf{q}}$ from (5) for $\mathbf{q} \in \mathcal{R}$
2. Compute the Hessian $\mathsf{E}_g = \sum_{\mathbf{q} \in \mathcal{R}} \ell_{\mathbf{q}} \ell_{\mathbf{q}}^{\mathsf{T}}$ and its inverse

Iterations

1. Warp the image $\mathcal{I}$ to $\tilde{\mathcal{I}}$ using the warp parameters $\mathbf{g}$
   - Compute the right hand side of the normal equations $\mathbf{b}_g = \sum_{\mathbf{q} \in \mathcal{R}} \ell_{\mathbf{q}} (\tilde{\mathcal{I}}[\mathbf{q}] - \mathcal{T}[\mathbf{q}])$
   - Solve for the update $\boldsymbol{\delta}_g = \mathsf{E}_g^{-1} \mathbf{b}_g$
2. Update the warp parameters: $\mathbf{g} \leftarrow \mathcal{U}_g(\mathbf{g}, \boldsymbol{\delta}_g)$

Table 1. The inverse compositional algorithm of Baker *et al.* [2] for estimating a groupwise geometric registration.

First order taylor expansion in $\boldsymbol{\delta}_g$ yields:

$$\min_{\boldsymbol{\delta}_g,\boldsymbol{\delta}_p} \sum_{\mathbf{q} \in \mathcal{R}} ((a + \delta_a)(\mathcal{T}[\mathbf{q}] + \ell_{\mathbf{q}}^{\mathsf{T}} \boldsymbol{\delta}_g) + b + \delta_b - \mathcal{I}[\mathcal{G}(\mathbf{q};\mathbf{g})])^2.$$

Expanding and neglecting second-order terms gives:

$$\min_{\boldsymbol{\delta}_g,\boldsymbol{\delta}_p} \sum_{\mathbf{q} \in \mathcal{R}} (a\ell_{\mathbf{q}}^{\mathsf{T}} \boldsymbol{\delta}_g + \delta_a \mathcal{T}[\mathbf{q}] + \delta_b + a\mathcal{T}[\mathbf{q}] + b - \tilde{\mathcal{I}}[\mathbf{q}])^2. \quad (7)$$

Directly solving this linear least squares problem leads to the *simultaneous inverse compositional algorithm* of Baker *et al.* [1].

Defining the complete unknown parameter update vector by $\boldsymbol{\delta}_{gp}^{\mathsf{T}} = (\boldsymbol{\delta}_g^{\mathsf{T}} \;\; \boldsymbol{\delta}_p^{\mathsf{T}})$, the normal equations are given by:

$$\underbrace{\left(\sum_{\mathbf{q} \in \mathcal{R}} \begin{pmatrix} a\ell_{\mathbf{q}} \\ \mathcal{T}[\mathbf{q}] \\ 1 \end{pmatrix} \begin{pmatrix} a\ell_{\mathbf{q}} & \mathcal{T}[\mathbf{q}] & 1 \end{pmatrix} \right)}_{\mathsf{E}_{gp}} \boldsymbol{\delta}_{gp} \quad (8)$$

$$= \underbrace{\left(\sum_{\mathbf{q} \in \mathcal{R}} \begin{pmatrix} a\ell_{\mathbf{q}} \\ \mathcal{T}[\mathbf{q}] \\ 1 \end{pmatrix} (\tilde{\mathcal{I}}[\mathbf{q}] - a\mathcal{T}[\mathbf{q}] - b) \right)}_{\mathbf{d}_{gp}}. \quad (9)$$

The Hessian matrix E_{gp} is clearly not constant, thus spoiling the main advantage of the inverse compositional approach. Baker *et al.* [1] propose several approximations to reduce

the computational cost: the 'project out inverse compositional algorithm' and the 'normalization inverse compositional algorithm'. They show that these approximations do not perform well for high gain values, see [1].

OBJECTIVE

Register an image $\mathcal{I}$ to a template $\mathcal{T}$ by computing the parameters $\mathbf{g}$ of a warp $\mathcal{G}(\mathbf{q};\mathbf{g})$ and gain and bias parameters $\mathbf{p}$ by minimizing the intensity error. Other inputs are the region of interest $\mathcal{R}$ in the template and an initial value for $\mathbf{g}$ and $\mathbf{p}$.

ALGORITHM

Pre-computations

1. Compute the gradient vectors $\ell_{\mathbf{q}}$ from (5) for $\mathbf{q} \in \mathcal{R}$

Iterations

1. Warp the image $\mathcal{I}$ to $\tilde{\mathcal{I}}$ using the warp parameters $\mathbf{g}$
   - Compute and invert the Hessian matrix E_{gp} from (8)
   - Compute the right hand side $\mathbf{d}_{gp}$ of the normal equations from (9)
   - Solve for the update $\boldsymbol{\delta}_{gp} = \mathsf{E}_{gp}^{-1}\mathbf{d}_{gp}$
2. Update the warp parameters: $\mathbf{g} \leftarrow \mathcal{U}_g(\mathbf{g}, \boldsymbol{\delta}_g)$ and the photometric parameters: $\mathbf{p} \leftarrow \mathbf{p} + \boldsymbol{\delta}_p$

Table 2. The simultaneous inverse compositional algorithm of Baker *et al.* [1] for estimating a groupwise geometric registration and gain and bias parameters. Note that the original algorithm handles general linear appearance variations.

3. The Gain and Bias Inverse Compositional Algorithm

We propose an algorithm which performs exactly the same calculations as the simultaneous inverse compositional algorithm of Baker *et al.*, but which do not require one to re-compute the Hessian matrix at each iteration, thus preserving the computational advantage of the original inverse compositional algorithm. The proposed algorithm is summarized in table 3.

3.1. The Structure of the Hessian Matrix

We expand the Hessian matrix E_{gp} from equation (8):

$$\mathsf{E}_{gp} = \begin{pmatrix} a^2\mathsf{E}_g & a\mathsf{E}_c \\ a\mathsf{E}_c^\mathsf{T} & \mathsf{E}_p \end{pmatrix},$$

with E_g, E_c and E_p depending only on the template and thus constant matrices, given by:

$$\mathsf{E}_g = \sum_{\mathbf{q}\in\mathcal{R}} \ell_{\mathbf{q}}\ell_{\mathbf{q}}^\mathsf{T} \qquad \mathsf{E}_c = \sum_{\mathbf{q}\in\mathcal{R}} \ell_{\mathbf{q}}(\mathcal{T}[\mathbf{q}] \;\; 1),$$

$$\text{and} \qquad \mathsf{E}_p = \sum_{\mathbf{q}\in\mathcal{R}} \begin{pmatrix} \mathcal{T}[\mathbf{q}]^2 & \mathcal{T}[\mathbf{q}] \\ \mathcal{T}[\mathbf{q}] & 1 \end{pmatrix}.$$

We observe that the Hessian matrix has thus a strong block structure. More precisely, all the blocks are constant up to some scale factors, depending on the gain a.

Similarly, the right hand side of the normal equations, defined in equation (9), is:

$$\mathbf{d}_{gp} = \begin{pmatrix} a\mathbf{d}_g \\ \mathbf{d}_p \end{pmatrix},$$

with:

$$\begin{aligned} \mathbf{d}_g &= \sum_{\mathbf{q}\in\mathcal{R}} \ell_{\mathbf{q}}(\tilde{\mathcal{I}}[\mathbf{q}] - a\mathcal{T}[\mathbf{q}] - b) \\ \mathbf{d}_p &= \sum_{\mathbf{q}\in\mathcal{R}} \begin{pmatrix} \mathcal{T}[\mathbf{q}] \\ 1 \end{pmatrix} (\tilde{\mathcal{I}}[\mathbf{q}] - a\mathcal{T}[\mathbf{q}] - b). \end{aligned}$$

3.2. Solving the Normal Equations

We propose a way to solve the normal equations allowing us to pre-compute some of the expensive steps. The solution is obtained by simple multiplication of the right hand side by rescaled constant matrices. The normal equations we want to solve are $\mathsf{E}_{gp}\boldsymbol{\delta}_{gp} = \mathbf{d}_{gp}$, or:

$$\begin{pmatrix} a^2\mathsf{E}_g & a\mathsf{E}_c \\ a\mathsf{E}_c^\mathsf{T} & \mathsf{E}_p \end{pmatrix} \begin{pmatrix} \boldsymbol{\delta}_g \\ \boldsymbol{\delta}_p \end{pmatrix} = \begin{pmatrix} a\mathbf{d}_g \\ \mathbf{d}_p \end{pmatrix}.$$

Borrowing from the standard photogrammetric block bundle adjustment technique, see *e.g.* [5], we multiply to the left by a full-rank matrix, as follows:

$$\begin{pmatrix} \mathrm{I} & 0 \\ -a\mathsf{E}_c^\mathsf{T}(a^2\mathsf{E}_g)^{-1} & \mathrm{I} \end{pmatrix} \begin{pmatrix} a^2\mathsf{E}_g & a\mathsf{E}_c \\ a\mathsf{E}_c^\mathsf{T} & \mathsf{E}_p \end{pmatrix} \begin{pmatrix} \boldsymbol{\delta}_g \\ \boldsymbol{\delta}_p \end{pmatrix}$$

$$= \begin{pmatrix} \mathrm{I} & 0 \\ -a\mathsf{E}_c^\mathsf{T}(a^2\mathsf{E}_g)^{-1} & \mathrm{I} \end{pmatrix} \begin{pmatrix} a\mathbf{d}_g \\ \mathbf{d}_p \end{pmatrix},$$

giving:

$$\begin{pmatrix} a^2\mathsf{E}_g & a\mathsf{E}_c \\ 0 & \mathsf{E}_p - a\mathsf{E}_c^\mathsf{T}(a^2\mathsf{E}_g)^{-1}a\mathsf{E}_c \end{pmatrix} \begin{pmatrix} \boldsymbol{\delta}_g \\ \boldsymbol{\delta}_p \end{pmatrix}$$

$$= \begin{pmatrix} a\mathbf{d}_g \\ \mathbf{d}_p - a^2\mathsf{E}_c^\mathsf{T}(a^2\mathsf{E}_g)^{-1}\mathbf{d}_g \end{pmatrix}.$$

OBJECTIVE

Register an image $\mathcal{I}$ to a template $\mathcal{T}$ by computing the parameters $\mathbf{g}$ of a warp $\mathcal{G}(\mathbf{q};\mathbf{g})$ and gain and bias parameters $\mathbf{p}$ by minimizing the intensity error. Other inputs are the region of interest $\mathcal{R}$ in the template and an initial value for $\mathbf{g}$ and $\mathbf{p}$.

ALGORITHM

Pre-computations

1. Compute the gradient vectors $\ell_{\mathbf{q}} = \nabla\mathcal{T}[\mathbf{q}]^{\mathsf{T}} \left.\frac{\partial\mathcal{G}}{\partial\mathbf{g}}\right|_{\mathbf{q};\tilde{\mathbf{g}}}$ for $\mathbf{q} \in \mathcal{R}$

2. Compute the three blocks forming the Hessian matrix:

$$\mathsf{E}_g = \sum_{\mathbf{q}\in\mathcal{R}} \ell_{\mathbf{q}}\ell_{\mathbf{q}}^{\mathsf{T}} \qquad \mathsf{E}_c = \sum_{\mathbf{q}\in\mathcal{R}} \ell_{\mathbf{q}}(\mathcal{T}[\mathbf{q}] \;\; 1) \qquad \mathsf{E}_p = \sum_{\mathbf{q}\in\mathcal{R}} \begin{pmatrix} \mathcal{T}[\mathbf{q}]^2 & \mathcal{T}[\mathbf{q}] \\ \mathcal{T}[\mathbf{q}] & 1 \end{pmatrix}$$

3. Compute matrices E_p^{-1}, Z and Y:

$$\mathsf{Z} = (\mathsf{E}_p - \mathsf{E}_c^{\mathsf{T}}\mathsf{E}_g^{-1}\mathsf{E}_c)^{-1} \qquad \mathsf{Y} = -\mathsf{Z}\mathsf{E}_c^{\mathsf{T}}\mathsf{E}_g^{-1}$$

Iterations

1. Warp the image $\mathcal{I}$ to $\tilde{\mathcal{I}}$ using the warp parameters $\mathbf{g}$

   - Compute the right hand side of the normal equations:

$$\mathbf{d}_g = \sum_{\mathbf{q}\in\mathcal{R}} a\ell_{\mathbf{q}}(\tilde{\mathcal{I}}[\mathbf{q}] - a\mathcal{T}[\mathbf{q}] - b) \qquad \mathbf{d}_p = \sum_{\mathbf{q}\in\mathcal{R}} \begin{pmatrix} \mathcal{T}[\mathbf{q}] \\ 1 \end{pmatrix} (\tilde{\mathcal{I}}[\mathbf{q}] - a\mathcal{T}[\mathbf{q}] - b)$$

   - Solve for the update:

$$\boldsymbol{\delta}_p = \mathsf{Z}\mathbf{d}_p + \mathsf{Y}\mathbf{d}_g \qquad \boldsymbol{\delta}_g = \frac{1}{a}\mathsf{E}_p^{-1}(\mathbf{d}_g - \mathsf{E}_c\boldsymbol{\delta}_p)$$

2. Update the warp and photometric parameters:

$$\mathbf{g} \leftarrow \mathcal{U}_g(\mathbf{g}, \boldsymbol{\delta}_g) \qquad \mathbf{p} \leftarrow \mathbf{p} + \boldsymbol{\delta}_p$$

Table 3. The proposed gain and bias inverse compositional algorithm for estimating a groupwise geometric registration and gain and bias parameters.

This equation simplifies to:

$$\begin{pmatrix} a^2\mathsf{E}_g & a\mathsf{E}_c \\ 0 & \mathsf{E}_p - \mathsf{E}_c^\mathsf{T}\mathsf{E}_g^{-1}\mathsf{E}_c \end{pmatrix} \begin{pmatrix} \boldsymbol{\delta}_g \\ \boldsymbol{\delta}_p \end{pmatrix} = \begin{pmatrix} a\mathbf{d}_g \\ \mathbf{d}_p - \mathsf{E}_c^\mathsf{T}\mathsf{E}_g^{-1}\mathbf{d}_g \end{pmatrix}.$$

The solution for the photometric parameters $\boldsymbol{\delta}_p$ is obtained directly from the second set of equations as:

$$\boldsymbol{\delta}_p = \mathsf{Z}\mathbf{d}_p + \mathsf{Y}\mathbf{d}_g,$$

where Z and Y are constant matrices, given by:

$$\begin{aligned} \mathsf{Z} &= (\mathsf{E}_p - \mathsf{E}_c^\mathsf{T}\mathsf{E}_g^{-1}\mathsf{E}_c)^{-1} \\ \mathsf{Y} &= -\mathsf{Z}\mathsf{E}_c^\mathsf{T}\mathsf{E}_g^{-1}. \end{aligned}$$

The solution for the geometric parameters $\boldsymbol{\delta}_g$ is given, from the first set of equations, by:

$$\begin{aligned} a^2\mathsf{E}_g\boldsymbol{\delta}_g &= a\mathbf{d}_g - a\mathsf{E}_c\boldsymbol{\delta}_p \\ \boldsymbol{\delta}_g &= \frac{1}{a}\mathsf{E}_p^{-1}(\mathbf{d}_g - \mathsf{E}_c\boldsymbol{\delta}_p). \end{aligned}$$

In this equation, matrix E_p^{-1} is constant and can be precomputed.

4. Experimental Results

We compared the simultaneous inverse compositional algorithm of Baker *et al.* and the gain and bias inverse compositional algorithm we propose, as described in tables 2 and 3 respectively, in the case of homographic warps, see §A. Note that both algorithms produce *exactly* the same results but with different computation times. Our experiments are designed to assess to which extent these differences are significant. We refer the reader to [1, 2] for a thorough set of experiments on the behaviour of a great variety of different algorithms. The cost of an iteration is constant for each algorithm. We used our own, fairly optimized implementation in MATLAB.

4.1. Simulated Data

Figure 1 shows the computational time of an iteration when varying image side length. The ranges of image side lengths are 10 to 100 and 100 to 1000 respectively for the left and right hand side graphs. We observed that their is a factor of at least 2 between the two algorithms, in favor of the gain and bias inverse compositional algorithm, in all cases.

4.2. Real Data

We compared the algorithms on several sets of images. We one of them, we show results. The template and the image are both 600×800. They are shown on figure 3, together with the region of interest. The region of interest contains 255,210 pixels. Figure 2 shows the error in intensity through the 28 iterations that were necessary to register the images. Figure 4 shows the error image at different iterations. The photometric parameters that were computed are $a = 0.98$ and $b = 3.77$, and the final RMS intensity error is 4.13. The computational time needed by the simultaneous inverse compositional algorithm was 76.91 seconds, while the gain and bias inverse compositional algorithm took 38.40 seconds.

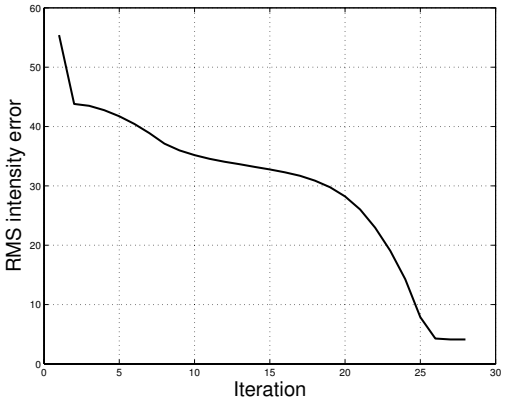


Figure 2. Error in intensity through the iterations for the images shown in figure 3.

5. Discussion

The proposed algorithm efficiently extends the inverse compositional algorithm of Baker *et al.* [2] to handle gain and bias. The computational time is reduced by a factor of at least 2 compared to the general linear appearance variations algorithm of Baker *et al.* [1].

There are several important issues that need to be investigated. The first one is about numerical conditioning: the elements of the Hessian matrices have different orders of magnitude, from 1 to $\mathcal{O}(k^2 s)$, where k is the image side length (in pixels) and s the maximum image intensity (in practice we expect s to be close to 255). Similarly to the normalization used to improve the conditioning in the eight point algorithm [5], we naturally wonder if normalizing the image coordinates and intensity can improve the numerical stability and thus the convergence properties of the algorithms.

The second issue is about using color images, *i.e.* determine if the proposed algorithm extends to deal with individual gain and bias for each color channel, or even for full

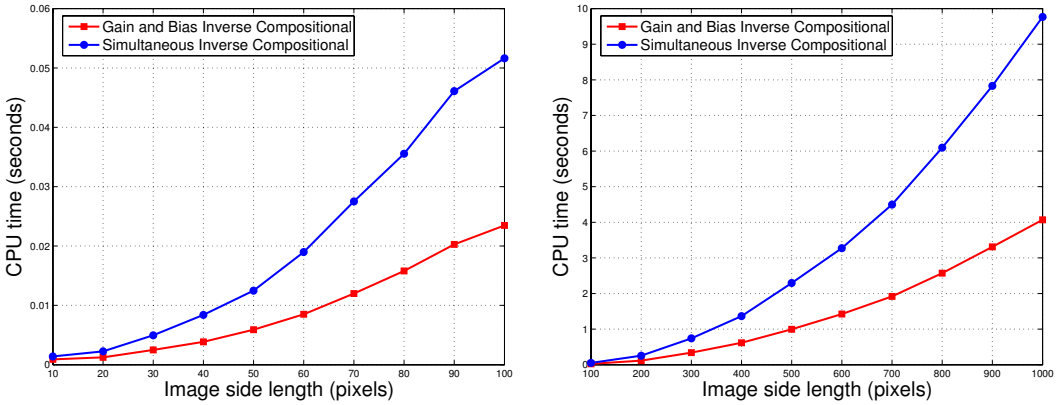
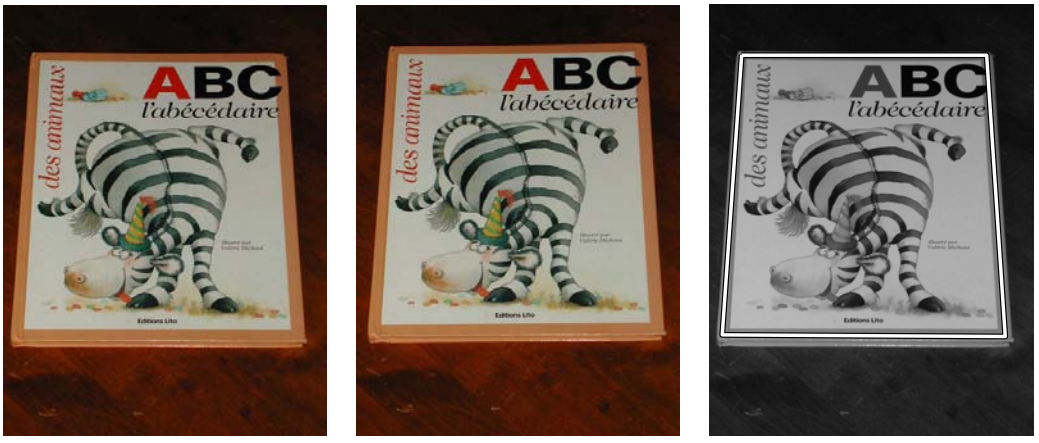


Figure 1. Computational time of an iteration versus image side length.



Template Image Region of interest

Figure 3. Real images used in the experiments.

Iteration 1
RMS error: 55.41

Iteration 10
RMS error: 34.56

Iteration 19
RMS error: 28.22

Iteration 28
RMS error: 4.13

Figure 4. The error image at different iterations and corresponding RMS error on the intensity.

linear combinations of the color channels.

The MATLAB code used to produce the experimental results in this paper is available for download on the web homepage of the author.

References

[1] S. Baker, R. Gross, and I. Matthews. Lucas-Kanade 20 years on: A unifying framework: Part 3. Technical Report CMU-RI-TR-03-35, Robotics Institute, Carnegie Mellon University, November 2003.

[2] S. Baker and I. Matthews. Lucas-Kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221–255, February 2004.

[3] J. R. Bergen, P. Anandan, K. J. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. In *Proceedings of the European Conference on Computer Vision*, 1992.

[4] G. D. Hager and P. N. Belhumeur. Efficient region tracking with parametric models of geometry and illumination. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 20(10):1025–1039, 1998.

[5] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003. Second Edition.

[6] M. Irani and P. Anandan. About direct methods. In *Workshop on Vision Algorithms: Theory and Practice*, 1999.

[7] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, 1981.

[8] P. H. S. Torr and A. Zisserman. Feature based methods for structure and motion estimation. In *Workshop on Vision Algorithms: Theory and Practice*, 1999.

A. Parameterizing Homographic Warps

The homographic warp, denoted $\mathcal{H}$, has 9 parameters defined up to scale. In homogenous coordinates, it is represented by a (3×3) homography matrix H. The representation of the warp by an homography matrix makes it easy to invert a warp or compose two warps, as required by the inverse compositional algorithm, respectively by inverting the homography matrix and by multiplying the two homography matrices.

Following [2], the local homography matrix is parameterized by an 8-vector $\boldsymbol{\delta}_h$ as:

$$\Delta\mathsf{H} \sim \mathrm{I} + \begin{pmatrix} \delta_{h,1} & \delta_{h,2} & \delta_{h,3} \\ \delta_{h,4} & \delta_{h,5} & \delta_{h,6} \\ \delta_{h,7} & \delta_{h,8} & 0 \end{pmatrix}. \tag{10}$$

The corresponding warp is:

$$\mathcal{H}(\mathbf{q}; \boldsymbol{\delta}_h) = \frac{1}{\delta_{h,7}q_1 + \delta_{h,8}q_2} \begin{pmatrix} (1+\delta_{h,1})q_1 + \delta_{h,2}q_2 + \delta_{h,3} \\ \delta_{h,4}q_1 + (1+\delta_{h,5})q_2 + \delta_{h,6} \end{pmatrix}$$

Note that $\boldsymbol{\delta}_h = \mathbf{0}$ corresponds to the identity warp since $\mathcal{H}(\mathbf{q}; \mathbf{0}) = \mathbf{q}$.

In practice, we represent the warp by the (3×3) homography matrix H, and implement the update rule as:

$$\mathcal{U}_g(\mathsf{H}, \boldsymbol{\delta}_h) = \mathsf{H} \cdot \Delta\mathsf{H},$$

where $\Delta\mathsf{H}$ is given by equation (10).

The Jacobian of the warp, evaluated around the identity warp, is given by:

$$\left.\frac{\partial \mathcal{H}}{\partial \boldsymbol{\delta}_h}\right|_{\mathbf{q};\mathbf{0}} = \begin{pmatrix} q_1 & q_2 & 1 & 0 & 0 & 0 & -q_1^2 & -q_1 q_2 \\ 0 & 0 & 0 & q_1 & q_2 & 1 & -q_1 q_2 & -q_2^2 \end{pmatrix}.$$

6.1.2 Paper (PAMI'08) – *Groupwise Geometric and Photometric Direct Image Registration*

Groupwise Geometric and Photometric Direct Image Registration

Adrien Bartoli
LASMEA (CNRS / Université Blaise Pascal), Clermont-Ferrand, France
Adrien.Bartoli@gmail.com

***Abstract*— Image registration consists in estimating geometric and photometric transformations that align two images as best as possible. The direct approach consists in minimizing the discrepancy in the intensity or color of the pixels. The inverse compositional algorithm has been recently proposed by Baker *et al.* for the direct estimation of groupwise geometric transformations. It is efficient in that it performs several computationally expensive calculations at a pre-computation phase.**

Photometric transformations act on the value of the pixels. They account for effects such as lighting change. Jointly estimating geometric and photometric transformations is thus important for many tasks such as image mosaicing. We propose an algorithm to jointly estimate groupwise geometric and photometric transformations while preserving the efficient pre-computation based design of the original inverse compositional algorithm. It is called the dual inverse compositional algorithm. It uses different approximations than the simultaneous inverse compositional algorithm and handles groupwise geometric and global photometric transformations. Its name stems from the fact that it uses an inverse compositional update rule for both the geometric and the photometric transformations.

We demonstrate the proposed algorithm and compare it to previous ones on simulated and real data. This shows clear improvements in computational efficiency and in terms of convergence.

***Index Terms*— image registation, geometric warp, photometric transformation, inverse composition.**

I. INTRODUCTION

Image registration is the task of applying some transformations to two images so that they match as best as possible. This can be seen as the computation of some geometric transformation, for example an homography, used to deform one of the images to model camera pose, and some photometric transformation, applied to the intensity or color of the pixels, to account *e.g.* for lighting change.

Image registration has been an important research topic for the past decades. It is central to many tasks in computer vision, medical imaging, augmented reality and robotics. Applications include image mosaicing [8], object and feature tracking [5], [7], [9], [11], superresolution [6] and visual servoing.

Broadly speaking, two approaches have been proposed: The feature-based and the direct approaches. The feature-based approach, see *e.g.* [12], relies on abstracting the input images by the geometric location of a set of carefully chosen, salient features. The direct approach, see *e.g.* [8], uses the value, *i.e.* the intensity or color, of the pixels of interest. The *inverse compositional algorithm* of Baker *et al.* [2] estimates groupwise geometric transformations such as homographies[1]. It has been shown to be one of the most reliable and computationally efficient registration methods. The efficiency stems from the so-called *inverse compositional trick*, making the Hessian matrix[2] constant (it is the design matrix involved in the linear least squares problem to be solved at each iteration). This makes it possible to pre-compute its inverse.

This paper is about the registration of two images related by a geometric and a photometric transformation. An example of photometric transformation is 'gain and bias' which rescales and offsets the value of the pixels. The *simultaneous inverse compositional algorithm* proposed in [1] by Baker *et al.* estimates such transformations but at the expense of recomputing and inverting the Hessian matrix at each iteration. An efficient variant called the *project out inverse compositional algorithm* is proposed in [1]. Due to an approximation of the photometric error function, it performs worse than the simultaneous inverse compositional algorithm, in terms of convergence frequency and number of iterations.

We propose the *dual inverse compositional algorithm*, which uses the inverse compositional trick for both the geometric and photometric counterparts of the registration, thereby preserving the possibility of pre-computing the inverse of the Hessian matrix. We originally proposed this method in [3]. It deals with grey-level and color images and groupwise photometric transformations. The dual inverse compositional algorithm takes different steps to converge compared to the simultaneous inverse compositional algorithm of Baker *et al*. Thorough experiments show similar convergence properties for these algorithms, with a significantly lower computational cost in favor of the proposed algorithm, and an improved stability compared to the project out inverse compositional algorithm. The dual inverse compositional algorithm is based on the assumption that the geometric and photometric transformations commute. This assumption prevents its use for estimating non global photometric transformations such as Linear Appearance Variations. It is thus useful mainly for global changes such as lighting and camera settings changes which can mix the different color channels.

a) Paper organization.: We formally state the problem and review previous work in §II. We present as background material the inverse compositional, the simultaneous inverse compositional and the project out inverse compositional algorithms of Baker *et al.* in §III. We propose the dual inverse compositional algorithm in §IV. Some groupwise color photometric transformations are

[1]To be precise, transformations parameterized such that there is a group structure on the parameter vector.

[2]We use the expression 'Hessian matrix' for the Gauss-Newton approximation to the 'true Hessian matrix'.

Digital Object Indentifier 10.1109/TPAMI.2008.22 0162-8828/$25.00 © 2008 IEEE

presented in §V. We report experimental results on simulated data in §VI. A conclusion is provided in §VII. The parameterization of homographic warps is detailed in appendix A and a proof showing that non global transformations can not be used with the dual inverse compositional algorithm is reported in appendix B. We report experimental results on real data in appendix C. The proposed dual inverse compositional algorithm is finally summarized.

b) Notation.: Vectors are denoted using bold fonts, *e.g.* $\mathbf{q}$, matrices using sans-serif fonts, *e.g.* E, and scalars in italics, *e.g.* a. The entries of a vector or matrix are written as in $\mathbf{x}^\mathsf{T} = (x_1 \ \cdots \ x_n)$, where $\mathbf{x}$ is transposed in this equation. The two-norm of a vector $\mathbf{r}$ is written $\|\mathbf{r}\|$. The gradient of a scalar-valued function f, in other words, its partial derivative vector, with respect to vector $\mathbf{x}$, is denoted $\nabla_\mathbf{x} f$. It is evaluated at $\mathbf{0}$, *i.e.* the zero vector, unless specified as in $(\nabla_\mathbf{x} f)(\mathbf{x}_0)$:

$$\nabla_\mathbf{x} f = (\nabla_\mathbf{x} f)(\mathbf{0}) = \left(\left(\frac{\partial f}{\partial x_1} \right)(\mathbf{0}) \ \cdots \ \left(\frac{\partial f}{\partial x_n} \right)(\mathbf{0}) \right)^\mathsf{T}.$$

Note that for vector-valued functions, ∇ gives the Jacobian matrix, *i.e.* the matrix containing all the partial derivatives of the function. Columnwise matrix vectorization is written vect.

The source and target images to be registered are denoted $\mathcal{S}$ and $\mathcal{T}$ respectively. They are seen as functions from $\mathbb{R}^2$ to $\mathbb{R}^c$ where c is the number of channels, *i.e.* $c = 1$ in the grey-level case and $c = 3$ in the color case. For instance, $\mathcal{T}(\mathbf{q})$ is the image value at pixel $\mathbf{q} \in \mathbb{R}^2$. Bilinear interpolation is used for sub-pixel coordinates. The unit column vector is denoted $\mathbf{1}$ with length given by the context. The geometric and photometric transformations are respectively denoted $\mathcal{G}$ and $\mathcal{P}$, with respective parameter vectors to be estimated denoted $\mathbf{g}$ and $\mathbf{p}$. We make the distinction between the target to source image photometric parameters and the reverse one, respectively denoted $\mathbf{p}$ and $\tilde{\mathbf{p}}$. Note that $\mathcal{G}$ and $\mathcal{P}$ refer to global parameterizations, as opposed to local, minimal parameterizations written $\bar{\mathcal{G}}$ and $\bar{\mathcal{P}}$ with parameters $\boldsymbol{\delta}_g$ and $\boldsymbol{\delta}_p$ or $\boldsymbol{\delta}_{\tilde{p}}$ respectively, defined such that the zero vector induces the identity transformation. The geometric transformation is also called the warp.

II. Problem Statement and Previous Work

The geometric registration problem is the minimization of a nonlinear least squares error function, given by the discrepancy in the value of the pixels, between the source image $\mathcal{S}$ and the target image $\mathcal{T}$ warped onto the source one by the unknown warp. The warp maps a pixel $\mathbf{q}$ in the region of interest $\mathcal{R}$ defined in the source image to the corresponding pixel $\mathcal{G}(\mathbf{q};\mathbf{g})$ in the target image. We expect that given an 'appropriate' parameter vector $\mathbf{g}$, $\mathcal{S}(\mathbf{q})$ is 'close to' $\mathcal{T}(\mathcal{G}(\mathbf{q};\mathbf{g}))$, for all $\mathbf{q} \in \mathcal{R}$: This is the brightness constancy assumption. The direct image registration problem is thus formally posed as the minimization of the *photometric error*:

$$\min_\mathbf{g} \sum_{\mathbf{q}\in\mathcal{R}} \|\mathcal{S}(\mathbf{q}) - \mathcal{T}(\mathcal{G}(\mathbf{q};\mathbf{g}))\|^2. \tag{1}$$

Note that other error functions can be used, to deal for example with occlusions. Most algorithms linearize each term in the transformation parameters $\mathbf{g}$, and iteratively update an initial guess by solving linear least squares problems. Hardie *et al.* [6] register several images at once while computing a superresolved one. Baker *et al.* [2] propose the efficient *inverse compositional algorithm* for solving problem (1). More details are given in the next section. It uses a Gauss-Newton, local approximation to the error function. The efficiency stems from the fact that the Hessian matrix involved in the normal equations to be solved at each iteration is constant. Its inverse is thus pre-computed.

Problem (1) does not take into account photometric changes, *i.e.* changes in the pixel values. These changes occur for example when lighting changes between the acquisition of the two images or when two different cameras are used. They are modeled by a transformation $\mathcal{P}$ with parameter vector $\mathbf{p}$, and give rise to the following minimization problem:

$$\min_{\mathbf{g},\mathbf{p}} \sum_{\mathbf{q}\in\mathcal{R}} \|\mathcal{S}(\mathbf{q}) - \mathcal{P}(\mathcal{T}(\mathcal{G}(\mathbf{q};\mathbf{g}));\mathbf{p})\|^2. \tag{2}$$

A commonly employed photometric model $\mathcal{P}$ is an affine transformation modeling gain and bias (or contrast and brightness). More complex transformations are reviewed in §V.

Jin *et al.* [9] use this model for feature tracking in grey-level images, in contrast to [11] which normalizes the image patches by using the mean and standard deviation of the pixel values. Heigl *et al.* [7] track points in color images by summing the error over the three channels. Lai and Fang [10] register images with low-order polynomials for modeling spatially varying gain and bias. Vemuri *et al.* [13] use a forward compositional framework for estimating spline-based Free-Form Deformations. Their modified Newton optimization scheme uses a constant approximation of the Hessian matrix evaluated at the optimum.

Baker *et al.* extend the inverse compositional algorithm in [1] to deal with linear appearance variations of the source image. In their framework, the photometric transformation is applied to the source image. As will be seen in §III-B, this makes the Hessian matrix varies accross the iterations, thereby spoiling the computational efficiency of the inverse compositional algorithm: The simultaneous inverse compositional algorithm estimates and inverts the Hessian matrix at each iteration. Baker *et al.* propose several approximations to reduce the computational cost, namely the 'project out inverse compositional algorithm' and the 'normalization inverse compositional algorithm'. They show that these approximations do not behave well for high gain values.

One reason for Baker *et al.* to apply the photometric transformation to the source image is to handle general linear appearance variations, *i.e.* based on linear combinations of 'eigenimages'. This is used in conjunction with 3D Morphable Models of *e.g.* faces which do not form a group. In the case of *e.g.* homography estimation, there is however no practical reason to handle such transformations.

III. Background

This section is devoted to the description of the inverse compositional and simultaneous inverse compositional algorithms of Baker *et al.* proposed in [2] and [1] respectively.

A. The Inverse Compositional Algorithm

The inverse compositional algorithm, or IC for short, forms the basis for our dual inverse compositional algorithm, presented in the next section. Its advantages are two-fold. First, it converges rapidly compared to other optimization schemes. Second, as already mentioned, each iteration is performed efficiently.

The inverse compositional algorithm iteratively updates an initial guess of the sought-after transformation. The key idea is to express the updated transformation as the composition of the

current transformation $\mathcal{G}(\cdot;\mathbf{g})$ and the inverse of an incremental transformation $\bar{\mathcal{G}}^{-1}(\cdot;\boldsymbol{\delta}_g)$: This is the inverse compositional update rule: $\mathcal{G}(\cdot;\mathbf{g}) \leftarrow \mathcal{G}(\bar{\mathcal{G}}^{-1}(\cdot;\boldsymbol{\delta});\mathbf{g})$. Notation $\bar{\mathcal{G}}$ refers to a local parameterization of the warp, as opposed to the global parameterization denoted $\mathcal{G}$. This update rule is written $\mathbf{g} \leftarrow \mathcal{U}_g(\mathbf{g},\boldsymbol{\delta}_g)$. Details on the global and local parameterizations and the update rule are given in appendix A for homographic warps.

The optimization is performed over $\boldsymbol{\delta}_g$, the parameter vector of the incremental warp, instead of $\mathbf{g}$. The geometric registration problem (1) is rewritten $\min_{\boldsymbol{\delta}_g} \sum_{\mathbf{q}\in\mathcal{R}} \|\mathcal{S}(\mathbf{q}) - \mathcal{T}(\mathcal{G}(\bar{\mathcal{G}}^{-1}(\mathbf{q};\boldsymbol{\delta}_g);\mathbf{g}))\|^2$. Let $\mathcal{W}$ be the warped target image, *i.e.* $\mathcal{W}(\mathbf{q}) = \mathcal{T}(\mathcal{G}(\mathbf{q};\mathbf{g}))$. The incremental transformation is then applied to the source image, instead of the target one, leading to $\min_{\boldsymbol{\delta}_g} \sum_{\mathbf{q}\in\mathcal{R}} \|\mathcal{S}(\bar{\mathcal{G}}(\mathbf{q};\boldsymbol{\delta}_g)) - \mathcal{W}(\mathbf{q})\|^2$. This is the *inverse compositional trick*. Note that this only approximates the original problem (1) since the error function is expressed within the warped and not within the source image. The error function is linearized by first order Taylor expansion in $\boldsymbol{\delta}_g$, forming a Gauss-Newton approximation: $\min_{\boldsymbol{\delta}_g} \sum_{\mathbf{q}\in\mathcal{R}} \|\mathcal{S}(\mathbf{q}) + \mathsf{L}_g^\mathsf{T}(\mathbf{q})\boldsymbol{\delta}_g - \mathcal{W}(\mathbf{q})\|^2$. This is a linear least squares problem, which is solved via its normal equations. We define $(\nabla\mathcal{S})(\mathbf{q})$ to be the $(2\times c)$ Jacobian matrix of the source image at $\mathbf{q}$ and $\left(\nabla_\mathbf{g}\bar{\mathcal{G}}\right)(\mathbf{q};\mathbf{0})$ the Jacobian matrix of the local warp, evaluated at $\mathbf{q}$ and at warp parameters $\mathbf{0}$. It is assumed for simplicity that $\mathbf{0}$ represents the identity warp, as discussed in appendix A. The Jacobian matrices $\mathsf{L}_g(\mathbf{q})$ are thus obtained, using the chain rule, as $\mathsf{L}_g^\mathsf{T}(\mathbf{q}) = (\nabla\mathcal{S})(\mathbf{q})^\mathsf{T}\left(\nabla_\mathbf{g}\bar{\mathcal{G}}\right)(\mathbf{q};\mathbf{0})$. They only depend on the source image at the pixels of interest, and are thus constant over the iterations. Let $\mathcal{D}(\mathbf{q}) = \mathcal{W}(\mathbf{q}) - \mathcal{S}(\mathbf{q})$ be the error image, the normal equations are $\mathsf{E}_g\boldsymbol{\delta}_g = \mathbf{b}_g$, where the Hessian matrix E_g and the right hand side $\mathbf{b}_g$ are $\mathsf{E}_g = \sum_{\mathbf{q}\in\mathcal{R}} \mathsf{L}_g(\mathbf{q})\mathsf{L}_g^\mathsf{T}(\mathbf{q})$ and $\mathbf{b}_g = \sum_{\mathbf{q}\in\mathcal{R}} \mathsf{L}_g(\mathbf{q})\mathcal{D}(\mathbf{q})$. The solution $\boldsymbol{\delta}_g = \mathsf{E}_g^{-1}\mathbf{b}_g$ for the local warp parameters is thus computed very efficiently since the Jacobian matrices $\mathsf{L}_g(\mathbf{q})$ as well as the inverse E_g^{-1} of the Hessian matrix are pre-computed.

Once $\boldsymbol{\delta}_g$ has been computed, parameters $\mathbf{g}$ are updated by composing the current warp with the incremental warp with the update rule $\mathbf{g} \leftarrow \mathcal{U}_g(\mathbf{g},\boldsymbol{\delta}_g)$. If one uses an homographic warp for example, then the updated parameters are given by multiplying the current homography by the inverse of the local one. The process is iterated until convergence, determined in our experiments, by thresholding $\|\boldsymbol{\delta}_g\|$ by $\varepsilon = 10\text{e}-8$.

B. The Simultaneous Inverse Compositional Algorithm

The simultaneous inverse compositional algorithm, or SIC for short, aims at registering two images by computing both a warp and a parametric photometric transformation $\mathcal{P}$ acting on the pixel values, *i.e.* their intensity of color.

Let $\tilde{\mathbf{p}}$ denotes the parameter vector for the photometric transformation from the source to the target image. Baker *et al.* [1] pose the registration problem as:

$$\min_{\mathbf{g},\tilde{\mathbf{p}}} \sum_{\mathbf{q}\in\mathcal{R}} \|\mathcal{P}(\mathcal{S}(\mathbf{q});\tilde{\mathbf{p}}) - \mathcal{T}(\mathcal{G}(\mathbf{q};\mathbf{g}))\|^2. \tag{3}$$

We note that the minimization takes place in the geometric space of the source image but in the photometric space of the target image.

Baker *et al.* use an inverse compositional update rule for the warp, and a forward additive update rule for the photometric transformation. Applying the inverse compositional trick as in the previous section, with $\mathcal{W}(\mathbf{q}) = \mathcal{T}(\mathcal{G}(\mathbf{q};\mathbf{g}))$ the warped image, yields $\min_{\boldsymbol{\delta}_g,\boldsymbol{\delta}_{\tilde{p}}} \sum_{\mathbf{q}\in\mathcal{R}} \|\mathcal{P}(\mathcal{S}(\bar{\mathcal{G}}(\mathbf{q};\boldsymbol{\delta}_g));\tilde{\mathbf{p}}+\boldsymbol{\delta}_{\tilde{p}}) - \mathcal{W}(\mathbf{q})\|^2$, where we switched the warp and the photometric transformation, *i.e.* we used $(\mathcal{P}(\mathcal{S};\tilde{\mathbf{p}}+\boldsymbol{\delta}_{\tilde{p}}))(\bar{\mathcal{G}}(\mathbf{q};\boldsymbol{\delta}_g)) = \mathcal{P}(\mathcal{S}(\bar{\mathcal{G}}(\mathbf{q};\boldsymbol{\delta}_g));\tilde{\mathbf{p}}+\boldsymbol{\delta}_{\tilde{p}})$. First order Taylor expansion in $\boldsymbol{\delta}_g$ and in $\boldsymbol{\delta}_{\tilde{p}}$ gives:

$$\min_{\boldsymbol{\delta}_g,\boldsymbol{\delta}_{\tilde{p}}} \sum_{\mathbf{q}\in\mathcal{R}} \Big\| \mathcal{P}(\mathcal{S}(\mathbf{q}) + \mathsf{L}_g^\mathsf{T}(\mathbf{q})\boldsymbol{\delta}_g;\tilde{\mathbf{p}}) + (\nabla_\mathbf{p}\mathcal{P})^\mathsf{T}(\mathcal{S}(\mathbf{q}) + \mathsf{L}_g^\mathsf{T}(\mathbf{q})\boldsymbol{\delta}_g;\tilde{\mathbf{p}})\boldsymbol{\delta}_{\tilde{p}} - \mathcal{W}(\mathbf{q}) \Big\|^2.$$

Further expansion is achieved using the assumption that $\mathcal{P}$ is an affine transformation for its parameters $\mathbf{p}$, which includes many different photometric transformations and linear appearance variations. We define $\breve{\mathbf{p}}$ as the linear counterpart of the parameters, *i.e.* where the intercept vanishes, in other words, without its affine counterpart. For example, if $\mathcal{P}(v;\tilde{\mathbf{p}}) = \tilde{p}_1 v + \tilde{p}_2$, *i.e.* the gain and bias photometric transformation, then $\breve{\mathbf{p}}^\mathsf{T} = (\tilde{p}_1\ 0)$. This allows us to simplify the first term as $\mathcal{P}(\mathcal{S}(\mathbf{q}) + \mathsf{L}_g^\mathsf{T}(\mathbf{q})\boldsymbol{\delta}_g;\tilde{\mathbf{p}}) = \mathcal{P}(\mathcal{S}(\mathbf{q});\tilde{\mathbf{p}}) + \mathcal{P}(\mathsf{L}_g^\mathsf{T}(\mathbf{q});\breve{\mathbf{p}})\boldsymbol{\delta}_g$. Neglecting the second order terms, we approximate the second term as $(\nabla_\mathbf{p}\mathcal{P})^\mathsf{T}(\mathcal{S}(\mathbf{q}) + \mathsf{L}_g^\mathsf{T}(\mathbf{q})\boldsymbol{\delta}_g;\tilde{\mathbf{p}})\boldsymbol{\delta}_{\tilde{p}} \approx \mathsf{L}_p^\mathsf{T}(\mathbf{q})\boldsymbol{\delta}_{\tilde{p}}$, with $\mathsf{L}_p(\mathbf{q}) = (\nabla_\mathbf{p}\mathcal{P})(\mathcal{S}(\mathbf{q});\tilde{\mathbf{p}})$. Note that $\mathsf{L}_p(\mathbf{q})$ is independent of $\mathbf{p}$ due to the affine nature of $\mathcal{P}$. It is thus pre-computed. Let the error image be $\mathcal{D}(\mathbf{q}) = \mathcal{W}(\mathbf{q}) - \mathcal{P}(\mathcal{S}(\mathbf{q});\tilde{\mathbf{p}})$, we get $\min_{\boldsymbol{\delta}_g,\boldsymbol{\delta}_{\tilde{p}}} \sum_{\mathbf{q}\in\mathcal{R}} \left\|\mathcal{P}(\mathsf{L}_g^\mathsf{T}(\mathbf{q});\breve{\mathbf{p}})\boldsymbol{\delta}_g + \mathsf{L}_p^\mathsf{T}(\mathbf{q})\boldsymbol{\delta}_{\tilde{p}} - \mathcal{D}(\mathbf{q})\right\|^2$. We see in this linear least squares problem that the associated Jacobian matrix has a constant part associated to the photometric parameters $\boldsymbol{\delta}_{\tilde{p}}$, and a non-constant part associated to the warp parameters $\boldsymbol{\delta}_g$. This makes non-constant the Hessian matrix of the normal equations since the current photometric transformation has to be applied to the Steepest Descent images (the columns of the Jacobian matrix). The optimization problem is rewritten $\min_{\boldsymbol{\delta}_{g\tilde{p}}} \sum_{\mathbf{q}\in\mathcal{R}} \left\|\mathsf{K}_{g\tilde{p}}^\mathsf{T}(\mathbf{q};\tilde{\mathbf{p}})\boldsymbol{\delta}_{g\tilde{p}} - \mathcal{D}(\mathbf{q})\right\|^2$, with $\boldsymbol{\delta}_{g\tilde{p}}^\mathsf{T} = (\boldsymbol{\delta}_g^\mathsf{T}\ \boldsymbol{\delta}_{\tilde{p}}^\mathsf{T})$ the joint incremental parameter vector and $\mathsf{K}_{g\tilde{p}}$ the joint Jacobian matrices given by $\mathsf{K}_{g\tilde{p}}^\mathsf{T}(\mathbf{q};\tilde{\mathbf{p}}) = (\mathcal{P}(\mathsf{L}_g^\mathsf{T}(\mathbf{q});\breve{\mathbf{p}})\ \mathsf{L}_p^\mathsf{T}(\mathbf{q}))$. The normal equations are $\mathsf{E}_{g\tilde{p}}\boldsymbol{\delta}_{g\tilde{p}} = \mathbf{b}_{g\tilde{p}}$, with the Hessian matrix $\mathsf{E}_{g\tilde{p}}$ and the right hand side $\mathbf{b}_{g\tilde{p}}$ given by $\mathsf{E}_{g\tilde{p}} = \sum_{\mathbf{q}\in\mathcal{R}} \mathsf{K}_{g\tilde{p}}(\mathbf{q};\tilde{\mathbf{p}})\mathsf{K}_{g\tilde{p}}^\mathsf{T}(\mathbf{q};\tilde{\mathbf{p}})$ and $\mathbf{b}_{g\tilde{p}} = \sum_{\mathbf{q}\in\mathcal{R}} \mathsf{K}_{g\tilde{p}}(\mathbf{q};\tilde{\mathbf{p}})\mathcal{D}(\mathbf{q})$.

C. The Project Out Inverse Compositional Algorithm

The project out inverse compositional algorithm, or PO for short, proposed by Baker *et al.* in [1], aims at reducing the computational cost required by each iteration of the simultaneous inverse computational algorithm. We start its derivation from equation (3) and rewrite the error function in matrix form using the $\mathcal{L}_2$-norm:

$$\min_{\mathbf{g},\tilde{\mathbf{p}}} \left\| \begin{pmatrix} \vdots \\ \mathcal{P}(\mathcal{S}(\mathbf{q});\tilde{\mathbf{p}}) - \mathcal{T}(\mathcal{G}(\mathbf{q};\mathbf{g})) \\ \vdots \end{pmatrix} \right\|^2.$$

The error vector is projected into a linear subspace $\mathcal{B}$ chosen as a basis for the affine photometric models we use, see §V, and its orthogonal complement $\mathcal{B}^\perp$, giving:

$$\min_{\mathbf{g},\tilde{\mathbf{p}}} \left\| \begin{pmatrix} \vdots \\ \mathcal{P}(\mathcal{S}(\mathbf{q});\tilde{\mathbf{p}}) - \mathcal{T}(\mathcal{G}(\mathbf{q};\mathbf{g})) \\ \vdots \end{pmatrix} \right\|_\mathcal{B}^2 + \left\| \begin{pmatrix} \vdots \\ \mathcal{S}(\mathbf{q}) - \mathcal{T}(\mathcal{G}(\mathbf{q};\mathbf{g})) \\ \vdots \end{pmatrix} \right\|$$

We observe that the second term is independent of the photometric transformation. The project out inverse compositional algorithm consists in minimizing the second term with respect to the geometric parameters $\mathbf{g}$. Minimizing the first term with respect to $\mathbf{p}$ subsequently gives the photometric parameters through a closed-form solution. Minimizing the second term is performed by using a weighted $\mathcal{L}_2$-norm, see [1] for the details, and is implemented with a weighted inverse compositional algorithm.

It is shown that this algorithm has problems with determining the optimal magnitude of the update vector. This arises when a gain is estimated in the photometric transformation, which is the case for all the transformations we show in §V. In the absence of noise, it is shown in [1] that the computed update vector $\boldsymbol{\delta}_g$ is a gain-weighted version of the ideal one. Two solutions are then suggested. The first one is to use Levenberg-Marquardt instead of Gauss-Newton as a local optimization engine, since it dynamically adjusts the step size. The second solution is a step size correction scheme based on dividing $\boldsymbol{\delta}_g$ by the current estimate of the gain. We implemented both solutions. Both work well but, as reported in [1], they sometimes oscillates around the sought after solution and may even diverge. Assessing convergence is thus difficult and one has to define a fixed number of iterations, and select the best estimate computed so far.

As reported in [1], the project out inverse compositional algorithm has poorer performances than the simultaneous one. The reason is that the algorithm is based on the fact that the error projected in the $\mathcal{B}^\perp$ subspace does not depend on the photometric transformation parameters. This is however true only when the alignment is correct. Therefore in practice, parameter updates are subject to unexpected perturbations which may prevent convergence.

IV. THE DUAL INVERSE COMPOSITIONAL ALGORITHM

We extend the inverse compositional algorithm to estimate a groupwise photometric transformation along with the warp, as stated in problem (2). The algorithm is summarized in table I and illustrated in figure 1. It is dubbed DIC for short.

The main difference compared to the simultaneous inverse compositional algorithm resides in the problem formulation: One of the images, namely the target one, is taken as a 'generator' for the other one, namely the source image. In other words, while the minimization takes place in the geometric frame of the source image and in the photometric frame of the target image in the formulation of Baker *et al.*, it takes place in the frame of the source image for both photometry and geometry in our formulation. The algorithms also differ in the update rule they employ for the photometric parameters. Baker *et al.* use a forward additive update rule, while we use an inverse compositional update rule. This allows us to apply the inverse compositional trick for both the geometric and photometric transformations, under the assumption that these transformations commute.

Consider problem (2), and plug in an inverse compositional update rule for both the geometric and photometric transformations, *i.e.* $\mathcal{G}(\cdot;\mathbf{g}) \leftarrow \mathcal{G}(\bar{\mathcal{G}}^{-1}(\cdot;\boldsymbol{\delta}_g);\mathbf{g})$ and $\mathcal{P}(\cdot;\mathbf{p}) \leftarrow \bar{\mathcal{P}}^{-1}(\mathcal{P}(\cdot;\mathbf{p});\boldsymbol{\delta}_p)$. Note that the incremental photometric transformation is composed to the left and not to the right of the current transformation, contrarily to the case of the warp. For more details, see §V. This gives:

$$\min_{\boldsymbol{\delta}_g,\boldsymbol{\delta}_p} \sum_{\mathbf{q}\in\mathcal{R}} \|\mathcal{S}(\mathbf{q}) - \bar{\mathcal{P}}^{-1}(\mathcal{P}(\mathcal{T}(\mathcal{G}(\bar{\mathcal{G}}^{-1}(\mathbf{q};\boldsymbol{\delta}_g);\mathbf{g}));\mathbf{p});\boldsymbol{\delta}_p)\|^2. \quad (4)$$

The optimization is now to be performed on the incremental parameters $\boldsymbol{\delta}_g$ and $\boldsymbol{\delta}_p$, the latter accounting for the incremental photometric transformation. Using the inverse compositional trick on the photometric transformation, *i.e.* applying the incremental photometric transformation to the source image instead of the target image gives:

$$\min_{\boldsymbol{\delta}_g,\boldsymbol{\delta}_p} \sum_{\mathbf{q}\in\mathcal{R}} \|\bar{\mathcal{P}}(\mathcal{S}(\mathbf{q});\boldsymbol{\delta}_p) - \mathcal{P}(\mathcal{T}(\mathcal{G}(\bar{\mathcal{G}}^{-1}(\mathbf{q};\boldsymbol{\delta}_g);\mathbf{g}));\mathbf{p})\|^2. \quad (5)$$

We now use the assumption that the order used to apply the warp and the photometric transformation to an image does not matter, *i.e.* that $\mathcal{P}(\mathcal{T}(\mathcal{G}(\cdot;\mathbf{g}));\mathbf{p}) = (\mathcal{P}(\mathcal{T};\mathbf{p}))(\mathcal{G}(\cdot;\mathbf{g}))$. This assumption allows us to switch the photometric transformation and the warps in the second term of equation (5). Applying the inverse compositional trick once again, on the incremental warp gives $\min_{\boldsymbol{\delta}_g,\boldsymbol{\delta}_p} \sum_{\mathbf{q}\in\mathcal{R}} \|\bar{\mathcal{P}}(\mathcal{S}(\bar{\mathcal{G}}(\mathbf{q};\mathbf{g}));\boldsymbol{\delta}_p) - \mathcal{P}(\mathcal{T};\mathbf{p})(\mathcal{G}(\mathbf{q};\mathbf{g}))\|^2$. Switching the warp and photometric transformation again, on the second term, and letting $\mathcal{W}$ be the warped and photometrically transformed image, *i.e.* $\mathcal{W}(\mathbf{q}) = \mathcal{P}(\mathcal{T}(\mathcal{G}(\mathbf{q};\mathbf{g}));\mathbf{p})$, yields:

$$\min_{\boldsymbol{\delta}_g,\boldsymbol{\delta}_p} \sum_{\mathbf{q}\in\mathcal{R}} \|\bar{\mathcal{P}}(\mathcal{S}(\bar{\mathcal{G}}(\mathbf{q};\boldsymbol{\delta}_g));\boldsymbol{\delta}_p) - \mathcal{W}(\mathbf{q})\|^2. \quad (6)$$

We show below that the normal equations induced by the Gauss-Newton approximation to problem (6) have a constant Hessian matrix. Similarly to the simultaneous inverse compositional algorithm, we use first order Taylor expansion in $\boldsymbol{\delta}_g$ and $\boldsymbol{\delta}_p$, giving:

$$\min_{\boldsymbol{\delta}_g,\boldsymbol{\delta}_p} \sum_{\mathbf{q}\in\mathcal{R}} \|\mathcal{S}(\mathbf{q}) + \mathsf{L}_g^\mathsf{T}(\mathbf{q})\boldsymbol{\delta}_g + (\nabla_\mathbf{p}\bar{\mathcal{P}})(\mathcal{S}(\mathbf{q}) + \mathsf{L}_g^\mathsf{T}(\mathbf{q})\boldsymbol{\delta}_g;\mathbf{0})\boldsymbol{\delta}_p - \mathcal{W}(\mathbf{q})$$

Using the assumption that $\mathcal{P}$ is an affine transformation and neglecting the second order terms yield:

$$\min_{\boldsymbol{\delta}_g,\boldsymbol{\delta}_p} \sum_{\mathbf{q}\in\mathcal{R}} \|(\mathsf{L}_g^\mathsf{T}(\mathbf{q}) \;\; \mathsf{L}_p^\mathsf{T}(\mathbf{q}))\boldsymbol{\delta}_{gp} - \mathcal{D}(\mathbf{q})\|^2,$$

where $\mathcal{D}$ is the error image, *i.e.* $\mathcal{D}(\mathbf{q}) = \mathcal{W}(\mathbf{q}) - \mathcal{S}(\mathbf{q})$. We denote the joint incremental parameter vector $\boldsymbol{\delta}_{gp}^\mathsf{T} = (\boldsymbol{\delta}_g^\mathsf{T} \;\; \boldsymbol{\delta}_p^\mathsf{T})$ and define the joint Jacobian matrices $\mathsf{L}_{gp}(\mathbf{q})$ by:

$$\mathsf{L}_{gp}^\mathsf{T}(\mathbf{q}) = (\mathsf{L}_g^\mathsf{T}(\mathbf{q}) \;\; \mathsf{L}_p^\mathsf{T}(\mathbf{q})). \quad (7)$$

The Jacobian matrices $\mathsf{L}_p(\mathbf{q})$ for the photometric parameters are $\mathsf{L}_p(\mathbf{q}) = \left(\nabla_\mathbf{p}\bar{\mathcal{P}}\right)(\mathcal{S}(\mathbf{q});\mathbf{0})$. As in the original inverse compositional algorithm, the Jacobian matrices only depend on the source image at the pixels of interest. They are thus constant as well as the Hessian matrix E_{gp} of the normal equations $\mathsf{E}_{gp}\boldsymbol{\delta}_{gp} = \mathbf{b}_{gp}$ with:

$$\mathsf{E}_{gp} = \sum_{\mathbf{q}\in\mathcal{R}} \mathsf{L}_{gp}(\mathbf{q})\mathsf{L}_{gp}^\mathsf{T}(\mathbf{q}) \quad (8)$$

$$\mathbf{b}_{gp} = \sum_{\mathbf{q}\in\mathcal{R}} \mathsf{L}_{gp}(\mathbf{q})\mathcal{D}(\mathbf{q}). \quad (9)$$

The warp is updated as in the inverse compositional algorithm: $\mathbf{g} \leftarrow \mathcal{U}_g(\mathbf{g},\boldsymbol{\delta}_g)$, and the photometric transformation update rule is written $\mathbf{p} \leftarrow \mathcal{U}_p(\mathbf{p},\boldsymbol{\delta}_p)$. The proposed dual inverse compositional algorithm handles most groupwise photometric transformation such as those given in the next section.

V. SOME GROUPWISE COLOR PHOTOMETRIC TRANSFORMATIONS

We mention some photometric transformations that can be employed within our framework. The dual inverse compositional

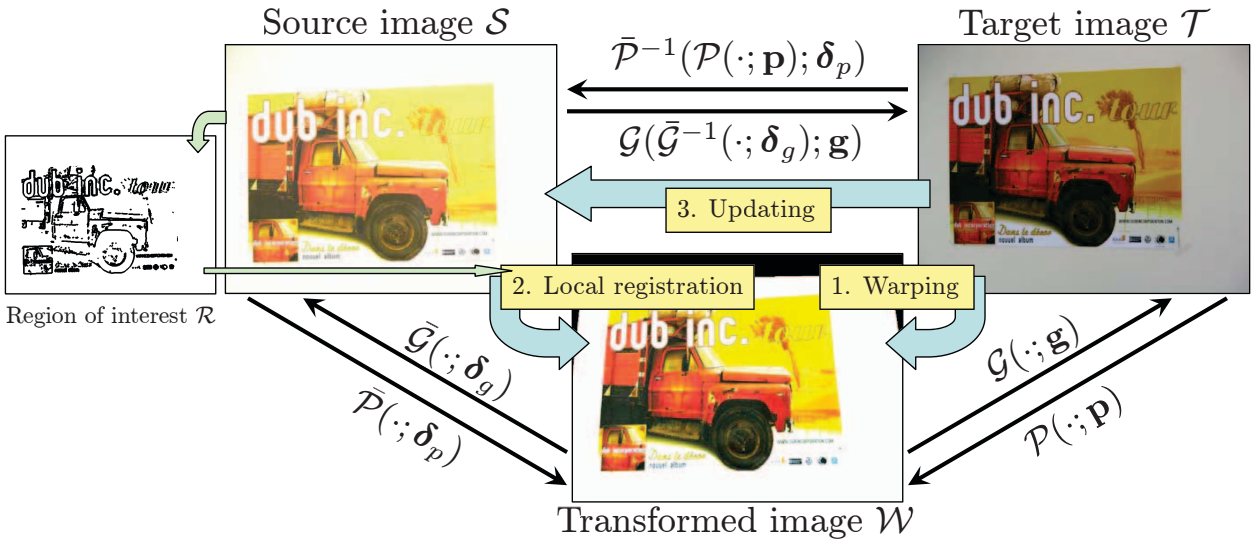


Fig. 1. DIC. The proposed *dual inverse compositional algorithm* extends the inverse compositional algorithm to jointly compute a geometric and a photometric registration, $\mathcal{G}(\cdot;\mathbf{g})$ and $\mathcal{P}(\cdot;\mathbf{p})$, by iterating the three main steps mentioned in the schema. One of the strengths of this approach is that, as in the inverse compositional algorithm, the Hessian matrix involved in the normal equations in step 2 is constant.

algorithm is based on the assumption that the geometric and photometric transformation commute. We show in appendix B that this prevents the use of non global photometric transformations. Spatially varying transformations, such as the polynomial one in [10], are usually non global. However, transformations such as the one in [5] which approximates specular highlights to first order, can be employed. It is spatially varying but has a constant weight for each pixel, depending only on its distance with some point of interest, making its parameters global.

The most common photometric transformation for grey-level images is the aforementioned gain and bias. In the color image case, we use affine transformations, *i.e.* transformations that can be written as $\mathsf{A}\mathbf{v}+\mathbf{b}$, where A is a (3×3) matrix combining the three color channels, and $\mathbf{b}$ is a 3-vector, modeling a per-channel bias. Finlayson *et al.* [4] show that linear transformations are well adapted for color constancy in practice. We have tried several variants, summarized below. Note that similarly to the warp, we use a global and a local parameterization for each transformation. The local parameterizations all guarantee $\bar{\mathcal{P}}(\mathbf{v};\mathbf{0})=\mathbf{v}$.

The update rules are obtained by writing the transformation in matrix form, by reshaping vector $\boldsymbol{\delta}_p$ to a matrix A and vector $\mathbf{b}$. Inversion and composition are then performed and the resulting set of parameters is vectorized to get the updated vector $\mathbf{p}$.

c) Single gain and bias.: This is the direct transposition of the gain and bias transformation of grey-level images to color images. It is due to account for global and uniform lighting change. The global and local transformations are:

$$\mathcal{P}(\mathbf{v};\mathbf{p})=p_1\mathbf{v}+p_2\mathbf{1} \qquad \text{and} \qquad \bar{\mathcal{P}}(\mathbf{v};\boldsymbol{\delta}_p)=\mathcal{P}\left(\mathbf{v};\begin{pmatrix}1\\0\end{pmatrix}+\boldsymbol{\delta}_p\right).$$

The (3×2) Jacobian matrix is $\mathsf{L}_p^{\mathsf{T}}(\mathbf{q}) = (\mathcal{S}(\mathbf{q})\ \mathbf{1})$, and the inverse compositional update rule is $\mathcal{U}_p(\mathbf{p},\boldsymbol{\delta}_p) = \frac{1}{1+\delta_{p,1}}\begin{pmatrix}p_1 & p_2-\delta_{p,2}\end{pmatrix}^{\mathsf{T}}$.

d) Multiple gain and bias.: This is a generalization of the gain and bias transformation to color images, with independent gain and bias applied to each color channel. This models global lighting change and the fact that each color channel may have a different behavior when lighting changes. Finlayson *et al.* [4] show that this model is effective for color constancy. The global and local transformations are:

$$\begin{aligned}\mathcal{P}(\mathbf{v};\mathbf{p}) &= \begin{pmatrix}p_1 & & \\ & p_2 & \\ & & p_3\end{pmatrix}\mathbf{v}+\begin{pmatrix}p_4\\p_5\\p_6\end{pmatrix}\\ \bar{\mathcal{P}}(\mathbf{v};\boldsymbol{\delta}_p) &= \mathcal{P}\left(\mathbf{v};\begin{pmatrix}\mathbf{1}\\\mathbf{0}\end{pmatrix}+\boldsymbol{\delta}_p\right),\end{aligned}$$

where $\mathbf{1}$ and $\mathbf{0}$ are (3×1) vectors. The (3×6) Jacobian matrix is $\mathsf{L}_p^{\mathsf{T}}(\mathbf{q}) = (\mathrm{diag}(\mathcal{S}(\mathbf{q}))\ \mathrm{I})$, and the inverse compositional update rule is $\mathcal{U}_p(\mathbf{p},\boldsymbol{\delta}_p) = \begin{pmatrix}\frac{p_1}{1+\delta_{p,1}} & \frac{p_2}{1+\delta_{p,2}} & \frac{p_3}{1+\delta_{p,3}} & \frac{p_4-\delta_{p,4}}{1+\delta_{p,1}} & \frac{p_5-\delta_{p,5}}{1+\delta_{p,2}} & \frac{p_6-\delta_{p,6}}{1+\delta_{p,3}}\end{pmatrix}^{\mathsf{T}}$.

e) Full affine channel mixing.: This generalizes the other photometric transformations by mixing the different color channels and applying a per-channel bias. This is mainly useful for images taken by different cameras or under different lighting colors. The global and local transformations are:

$$\begin{aligned}\mathcal{P}(\mathbf{v};\mathbf{p}) &= \begin{pmatrix}p_1 & p_2 & p_3\\ p_4 & p_5 & p_6\\ p_7 & p_8 & p_9\end{pmatrix}\mathbf{v}+\begin{pmatrix}p_{10}\\p_{11}\\p_{12}\end{pmatrix}\\ \bar{\mathcal{P}}(\mathbf{v};\boldsymbol{\delta}_p) &= \mathcal{P}(\mathbf{v};\mathbf{u}+\boldsymbol{\delta}_p),\end{aligned}$$

with $\mathbf{u}^{\mathsf{T}} = \mathrm{vect}(\mathrm{I}_{(3\times 3)}) = (1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0)$. The (3×12) Jacobian matrix is:

$$\mathsf{L}_p^{\mathsf{T}}(\mathbf{q}) = \left(\mathrm{diag}\left(\mathcal{S}(\mathbf{q})^{\mathsf{T}},\mathcal{S}(\mathbf{q})^{\mathsf{T}},\mathcal{S}(\mathbf{q})^{\mathsf{T}}\right)\quad \mathrm{I}\right),$$

and the inverse compositional update rule is:

$$\mathcal{U}_p(\mathbf{p},\boldsymbol{\delta}_p) = \begin{pmatrix}\mathrm{vect}\left(\begin{pmatrix}1+\delta_{p,1} & \delta_{p,2} & \delta_{p,3}\\ \delta_{p,4} & 1+\delta_{p,5} & \delta_{p,6}\\ \delta_{p,7} & \delta_{p,8} & 1+\delta_{p,9}\end{pmatrix}^{-1}\begin{pmatrix}p_1 & p_2 & p_3\\ p_4 & p_5 & p_6\\ p_7 & p_8 & p_9\end{pmatrix}\right)\\ \begin{pmatrix}1+\delta_{p,1} & \delta_{p,2} & \delta_{p,3}\\ \delta_{p,4} & 1+\delta_{p,5} & \delta_{p,6}\\ \delta_{p,7} & \delta_{p,8} & 1+\delta_{p,9}\end{pmatrix}^{-1}\begin{pmatrix}p_{10}-\delta_{p,10}\\ p_{11}-\delta_{p,11}\\ p_{12}-\delta_{p,12}\end{pmatrix}\end{pmatrix}.$$

VI. EXPERIMENTAL RESULTS ON SIMULATED DATA

Our experiments are designed to compare the converge properties and the computational cost of the proposed dual inverse compositional algorithm to other algorithms in various conditions, as well as different photometric transformations as described in §V. All comparisons are done by estimating homographies. Our implementation uses MATLAB with a number of routines written in C through Mex files (*e.g.* bilinear image warping and error computation). Timing is measured on a PC equipped with a Pentium M at 1.6 GHz. We report results for SIC (see §III-B), PO (see §III-C) and the proposed DIC (see §IV).

f) Simulation setup.: Given a texture image, we simulate a 2D homography by displacing four points in random directions by some magnitude γ with default value $\gamma = 5$ pixels. This is used to generate the target image in conjunction with a gain α and a bias β with default values $\alpha = 1.2$ and $\beta = 15$. For the multiple gains and biases and full affine channel mixing photometric models, we use α with 10% Gaussian perturbation for the diagonal entries, some random values for the off-diagonal entries and β with 10% Gaussian perturbations for the biases. Finally, centred Gaussian noise with variance σ is added to the pixel values in the source and target images, with default value $\sigma = 25.5$, *i.e.* 10% of 255 (the maximum value of intensity and of each color channel). Finally, the pixel values are clamped between 0 and 255 in order to simulate sensor saturation. The source image is 600×800, and 25,392 pixels of interest are used.

We vary some parameters of this setup independently, namely the noise variance σ from 0% to 20% (*i.e.* 0 to 51 pixel intensity or color units), the geometric magnitude γ from 0 to 20 pixels and the gain from 0.2 to 3. The results are average values over 100 trials.

The algorithms are run for 20 iterations. Results are shown for the grey-level gain and bias model (2 parameters) and the full affine channel mixing model (12 parameters).

g) Computational time.: We can see on figure 2 (a) that the overall computational time needed by DIC and PO is much lower than the one needed by SIC. This also holds against the geometric transformation magnitude and the gain value. Figure 2 (b) shows that DIC needs slightly more computational time than PO, at worse 1.69 times more. The next table shows detailed timing results in seconds for the parameter update step for a single iteration:

	G&B	Single G&B	Multiple G&B	Full Affine
SIC	0.0887	0.2927	0.5070	0.8723
PO	0.0023	0.0065	0.0065	0.0065
DIC	0.0033	0.0098	0.0117	0.0146

The image warping step respectively takes 0.0270 seconds and 0.0312 seconds in the grey-level and color cases respectively. We observe that DIC and PO have computational times of the same order of magnitude, while SIC is two orders of magnitude more expensive.

h) Geometric error.: This is measure by comparing the estimated transformation at convergence to the true one and reflects the accuracy of the algorithms. As can be seen on figure 2 (c), the geometric error is almost the same for all algorithms. This is also the case for the other experiments we performed, and means that all algorithms reach the same accuracy on the estimated transformation when the converge to the sought after solution.

i) Number of iterations.: Figure 3 (a) shows that the proposed DIC needs about the same number of iterations as SIC, while PO needs more iterations and usually reaches the maximum 20 iterations. Given that the computational cost of an iteration is lower for PO and DIC than for SIC, this explains why the computational time needed by SIC is much larger than the one needed by DIC and PO.

j) Photometric error.: The photometric error (not shown here) reached by all algorithms in all our experiments are identical, meaning that they all are able to minimize the error function to the same extent, and thus that they all converge to the right solution.

k) Convergence frequency.: We observe on figure 3 (b) that beyond a noise variance of 50, the converge frequency drops from 100%. Figure 3 (c) shows that the convergence frequency also decreases when the geometric transformation magnitude γ increases. The decrease starts at about 6 pixels for PO and 8 pixels for SIC and DIC, and is slightly faster for PO than for the two other methods. Overall, PO has a clearly lower convergence frequency than SIC and DIC. Varying the gain do not affect convergence. The convergence frequencies for DIC and SIC are equivalent.

l) Overall.: All three methods are equivalently accurate. PO requires more iterations than SIC and DIC. SIC is much more expensive than PO and DIC, and DIC is slightly more expensive than PO. SIC and DIC have equivalent convergence frequencies, significantly higher than the convergence frequency of PO. This means that DIC combines both the stability of SIC with the efficiency of PO and is thus the algorithm we recommend for this kind of groupwise image registration problem.

VII. CONCLUSION

An algorithm is proposed for the direct registration of two images: the dual inverse compositional algorithm. It is original in that it considers one of the images to be registered as a 'generator' for the other one. Its main advantage compared to the simultaneous inverse compositional algorithm is that the Hessian matrix involved in each iteration is constant, making the algorithm very efficient in terms of computational cost. This stems from the fact that, as for the geometric transformation, the photometric one is dealt with using the inverse compositional trick. It is more stable than the project out inverse compositional algorithm. The proposed dual inverse compositional algorithm handles any global groupwise geometric and photometric transformation.

REFERENCES

[1] S. Baker, R. Gross, and I. Matthews. Lucas-Kanade 20 years on: A unifying framework: Part 3. Technical Report CMU-RI-TR-03-35, Robotics Institute, Carnegie Mellon University, November 2003.

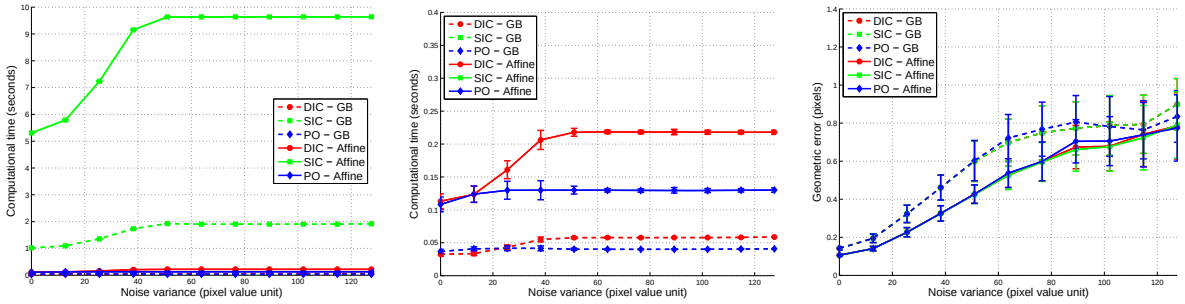
[2] S. Baker and I. Matthews. Lucas-Kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221–255, February 2004.

[3] A. Bartoli. Groupwise geometric and photometric direct image registration. In *Proceedings of the British Machine Vision Conference*, 2006.

[4] G. D. Finlayson, M. S. Drew, and B. Funt. Color constancy: Generalized diagonal transforms suffice. *Journal of the Optical Society of America A*, 11(11):3011–3019, November 1994.

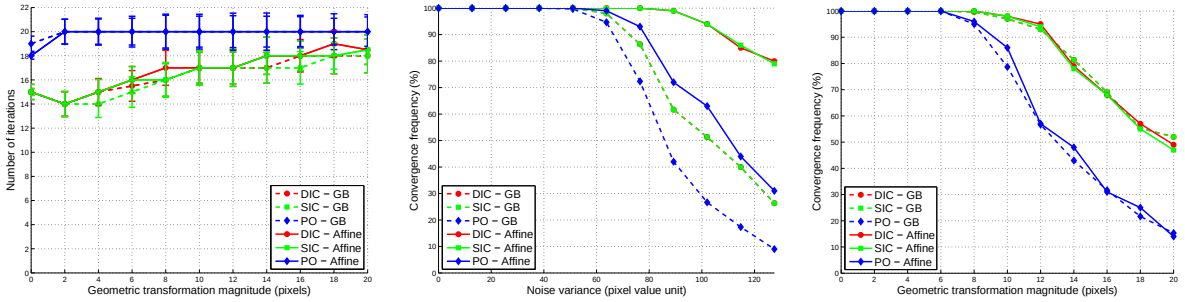
[5] M. Gouiffes, C. Collewet, C. Fernandez-Maloigne, and A. Trémeau. Feature point tracking: Robustness to specular highlights and lighting changes. In *Proceedings of the European Conference on Computer Vision*, 2006.

[6] R. C. Hardie, K. J. Barnard, and E. E. Armstrong. Joint MAP registration and high-resolution image estimation using a sequence of undersampled images. IEEE *Transactions on Image Processing*, 6(12):1621–1633, December 1997.



(a) (b) (c)

Fig. 2. (a) Computational time versus the variance of pixel noise. (b) Zoom on the graph in (a). (c) Geometric error versus the variance of pixel noise.



(a) (b) (c)

Fig. 3. (a) Number of iterations versus the magnitude of the geometric transformation. (c) Convergence frequency versus the variance of pixel noise. (c) Convergence frequency versus the magnitude of the geometric transformation.

[7] B. Heigl, D. Paulus, and H. Niemann. Tracking points in sequences of color images. In *Pattern Recognition and Image Understanding*, 1999.

[8] M. Irani and P. Anandan. About direct methods. In *Workshop on Vision Algorithms: Theory and Practice*, 1999.

[9] H. Jin, P. Favaro, and S. Soatto. Real-time feature tracking and outlier rejection with changes in illumination. In *Proceedings of the International Conference on Computer Vision*, 2001.

[10] S.-H. Lai and M. Fang. Robust and efficient image alignment with spatially varying illumination models. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, 1999.

[11] T. Tommasini, A. Fusiello, E. Trucco, and V. Roberto. Improving feature tracking with robust statistics. *Pattern Analysis and Applications*, 2(4):312–320, 1999.

[12] P. H. S. Torr and A. Zisserman. Feature based methods for structure and motion estimation. In *Workshop on Vision Algorithms: Theory and Practice*, 1999.

[13] B. C. Vemuri, S. Huang, S. Sahni, C. M. Leonard, C. Mohr, R. Gilmore, and J. Fitzsimmons. An efficient motion estimator with application to medical image registration. *Medical Image Analysis*, 2(1):79–98, 1998.

APPENDIX

A. Parameterizing Homographies

Groupwise geometric transformations include translations, rotations, affinities and homographies. We describe the case of homographies. They have 8 degrees of freedom, and can be represented by (3×3) homogeneous matrices (*i.e.* defined up to scale). The representation of the warp by an homography matrix makes it easy to invert a warp or compose two warps, as required by the inverse composition trick, respectively by inverting the homography matrix and by multiplying the two homography matrices. We use an homography matrix H for the parameterization of the global warp $\mathcal{G}$:

$$\mathcal{G}(\mathbf{q};\mathsf{H}) = \frac{1}{H_{31}q_1 + H_{32}q_2 + H_{33}} \begin{pmatrix} H_{11}q_1 + H_{12}q_2 + H_{13} \\ H_{21}q_1 + H_{22}q_2 + H_{23} \end{pmatrix}.$$

We constrain H to have unit two-norm. This is enforced each time the update rule is applied by simply dividing H by its two-norm.

Following [2], the local warp $\bar{\mathcal{G}}$ is parameterized by an 8-vector $\boldsymbol{\delta}_h$ as:

$$\bar{\mathcal{G}}(\mathbf{q};\boldsymbol{\delta}_h) = \mathcal{G}\left(\mathbf{q};\mathrm{I} + \begin{pmatrix} \delta_{h,1} & \delta_{h,2} & \delta_{h,3} \\ \delta_{h,4} & \delta_{h,5} & \delta_{h,6} \\ \delta_{h,7} & \delta_{h,8} & 0 \end{pmatrix}\right).$$

This parameterization is such that, as required for deriving the registration algorithms, the identity local warp is obtained for $\boldsymbol{\delta}_h = \mathbf{0}$:

$$\bar{\mathcal{G}}(\mathbf{q};\mathbf{0}) = \mathcal{G}(\mathbf{q};\mathrm{I}) = \mathbf{q}.$$

A short calculation shows that the Jacobian of the local warp is:

$$(\nabla_{\boldsymbol{\delta}_h}\bar{\mathcal{G}})(\mathbf{q};\mathbf{0}) = \begin{pmatrix} q_1 & q_2 & 1 & 0 & 0 & 0 & -q_1^2 & -q_1q_2 \\ 0 & 0 & 0 & q_1 & q_2 & 1 & -q_1q_2 & -q_2^2 \end{pmatrix}.$$

Inverse composition is performed by multiplying the current homography matrix to the right by the inverse of the incremental one:

$$\mathcal{U}_g(\mathsf{H},\boldsymbol{\delta}_h) = \mathsf{H}\left(\mathrm{I} + \begin{pmatrix} \delta_{h,1} & \delta_{h,2} & \delta_{h,3} \\ \delta_{h,4} & \delta_{h,5} & \delta_{h,6} \\ \delta_{h,7} & \delta_{h,8} & 0 \end{pmatrix}\right)^{-1}.$$

OBJECTIVE

Register a target image $\mathcal{T}$ to a source image $\mathcal{S}$ by computing the parameters $\mathbf{g}$ of a geometric registration $\mathcal{G}(\cdot;\mathbf{g})$ and the parameters $\mathbf{p}$ of a photometric registration $\mathcal{P}(\cdot;\mathbf{p})$. Other inputs are the region of interest $\mathcal{R}$ in the source image and an initial value for $\mathbf{g}$ and $\mathbf{p}$. Upon convergence, the photometric error $\sum_{\mathbf{q}\in\mathcal{R}} \|\mathcal{S}(\mathbf{q}) - \mathcal{P}(\mathcal{T}(\mathcal{G}(\mathbf{q};\mathbf{g}));\mathbf{p})\|^2$ is minimized.

ASSUMPTIONS

- Group structure on the warp parameters $\mathbf{g}$
- Group structure on the global photometric parameters $\mathbf{p}$
- Commutativity of the warp and the photometric transformation

ALGORITHM

Pre-computations

1) Compute the joint Jacobian matrices $\mathsf{L}_{gp}(\mathbf{q})$ for $\mathbf{q}\in\mathcal{R}$ from equation (7):

$$\mathsf{L}_{gp}(\mathbf{q}) = \left((\nabla\mathcal{S})(\mathbf{q})^{\mathsf{T}} \left(\nabla_{\mathbf{g}}\bar{\mathcal{G}}\right)(\mathbf{q};\mathbf{0}) \quad (\nabla_{\mathbf{p}}\mathcal{P})(\mathcal{S}(\mathbf{q});\mathbf{0})\right)$$

2) Compute the Hessian matrix E_{gp} and its inverse from equation (8):

$$\mathsf{E}_{gp} = \sum_{\mathbf{q}\in\mathcal{R}} \mathsf{L}_{gp}(\mathbf{q})\mathsf{L}_{gp}^{\mathsf{T}}(\mathbf{q})$$

Iterations

1) Warp and photometrically transform the target image $\mathcal{T}$ to $\mathcal{W}$ using the warp $\mathbf{g}$ and the photometric parameters $\mathbf{p}$:

$$\mathcal{W}(\mathbf{q}) = \mathcal{P}(\mathcal{T}(\mathcal{G}(\mathbf{q};\mathbf{g}));\mathbf{p})$$

2) Compute the incremental warp and photometric parameters $\boldsymbol{\delta}_g$ and $\boldsymbol{\delta}_p$:
   - Compute the error image $\mathcal{D}$:
   $$\mathcal{D}(\mathbf{q}) = \mathcal{W}(\mathbf{q}) - \mathcal{S}(\mathbf{q})$$
   - Compute the right hand side of the normal equations $\mathbf{b}_{gp}$ from equation (9):
   $$\mathbf{b}_{gp} = \sum_{\mathbf{q}\in\mathcal{R}} \mathsf{L}_{gp}(\mathbf{q})\mathcal{D}(\mathbf{q})$$
   - Solve for the joint incremental parameters $\boldsymbol{\delta}_{gp}$:
   $$\boldsymbol{\delta}_{gp} = \mathsf{E}_{gp}^{-1}\mathbf{b}_{gp}$$

3) Update the warp $\mathbf{g}$ and photometric parameters $\mathbf{p}$:

$$\mathbf{g} \leftarrow \mathcal{U}_g(\mathbf{g},\boldsymbol{\delta}_g) \qquad \mathbf{p} \leftarrow \mathcal{U}_p(\mathbf{p},\boldsymbol{\delta}_p)$$

TABLE I

DIC. THE PROPOSED *dual inverse compositional algorithm* FOR GROUPWISE GEOMETRIC AND PHOTOMETRIC REGISTRATION OF GREY-LEVEL OR COLOR IMAGES. THE HESSIAN MATRIX IS CONSTANT THROUGHOUT THE ITERATIONS.

B. The Dual Inverse Composition Algorithms and Non Global Photometric Transformations

We show that the dual inverse compositional algorithm does not handle non global varying photometric models such as the Linear Appearance Variations used in Active Appearance Models. We note that the photometric transformation $\mathcal{P}$ now depends on the pixel location $\mathbf{q}$:

$$\mathcal{P}(\mathbf{v};\mathbf{p}) \Rightarrow \mathcal{P}(\mathbf{v};\mathbf{p};\mathbf{q}).$$

We derive the proof for the Linear Appearance Variation model, but the reasoning holds for general non-global photometric models. The Linear Appearance Variation model combines basis images $\mathcal{A}_k$ as:

$$\mathcal{P}(\mathbf{v};\mathbf{p};\mathbf{q}) = \mathbf{v} + \sum_{k=1}^{l} p_k\mathcal{A}_k(\mathbf{q}).$$

For the case where the basis images are aligned with the source image, the following property holds:

$$\mathcal{P}(\mathcal{S}(\mathbf{q});\mathbf{p};\mathbf{q})) \quad = \quad (\mathcal{P}(\mathcal{S};\mathbf{p}))(\mathbf{q}). \tag{10}$$

When they are aligned with the target image, this transforms as:

$$\mathcal{P}(\mathcal{T}(\mathbf{q});\mathbf{p};\mathbf{q})) \quad = \quad (\mathcal{P}(\mathcal{T};\mathbf{p}))(\mathbf{q}). \tag{11}$$

We examine the case where the basis images are aligned with the source image and then when they are aligned with the target image.

1) Basis Images Aligned With the Source Image:

a) The error function.: Each term of the nonlinear least squares error in (4) is:

$$e(\mathbf{q}) = \mathcal{S}(\mathbf{q}) - \bar{\mathcal{P}}^{-1}(\mathcal{P}(\mathcal{T}(\mathcal{G}(\bar{\mathcal{G}}^{-1}(\mathbf{q};\boldsymbol{\delta}_g);\mathbf{g}));\mathbf{p};\mathbf{q});\boldsymbol{\delta}_p;\mathbf{q}).$$

We use the inverse compositional trick on the photometric transformation, giving:

$$e(\mathbf{q}) \approx \bar{\mathcal{P}}(\mathcal{S}(\mathbf{q});\boldsymbol{\delta}_p;\mathbf{q}) - \mathcal{P}(\mathcal{T}(\mathcal{G}(\bar{\mathcal{G}}^{-1}(\mathbf{q};\boldsymbol{\delta}_g);\mathbf{g}));\mathbf{p};\mathbf{q}).$$

We then switch the incremental geometric transformation ($\boldsymbol{\delta}_g$) and the current photometric transformation ($\mathbf{p}$) and, to avoid notational burden, we apply the inverse compositional trick on the geometric transformation at the same time, giving:

$$e(\mathbf{q}) \approx \bar{\mathcal{P}}(\mathcal{S}(\bar{\mathcal{G}}(\mathbf{q};\boldsymbol{\delta}_g));\boldsymbol{\delta}_p;\bar{\mathcal{G}}(\mathbf{q};\boldsymbol{\delta}_g)) - \mathcal{P}(\mathcal{T}(\mathcal{G}(\mathbf{q};\mathbf{g}));\mathbf{p};\bar{\mathcal{G}}(\mathbf{q};\boldsymbol{\delta}_g))$$

b) First term.: By switching the order of the photometric and geometric transformations, the leading term rewrites as:

$$e_1(\mathbf{q}) = \bar{\mathcal{P}}(\mathcal{S}(\bar{\mathcal{G}}(\mathbf{q};\boldsymbol{\delta}_g));\boldsymbol{\delta}_p;\bar{\mathcal{G}}(\mathbf{q};\boldsymbol{\delta}_g)) = \left(\bar{\mathcal{P}}(\mathcal{S};\boldsymbol{\delta}_p)\right)(\bar{\mathcal{G}}(\mathbf{q};\boldsymbol{\delta}_g)),$$

and thus has a constant Jacobian matrix since the photometric transformation does not depend on the pixel location. We rewrite it as:

$$e_1(\mathbf{q}) = \left(\bar{\mathcal{P}}(\mathcal{S};\boldsymbol{\delta}_p)\right)(\bar{\mathcal{G}}(\mathbf{q};\boldsymbol{\delta}_g)) = \left(\mathcal{S} + \sum_{k=1}^{l} \delta_p^k\mathcal{A}_k\right)(\bar{\mathcal{G}}(\mathbf{q};\boldsymbol{\delta}_g)).$$

$$\begin{aligned}
\frac{\partial e_1}{\partial \boldsymbol{\delta}_p^k}(\mathbf{q};\boldsymbol{\delta}_p=\mathbf{0};\boldsymbol{\delta}_g=\mathbf{0}) &= \mathcal{A}_k(\mathbf{q}) \\
\frac{\partial e_1}{\partial \boldsymbol{\delta}_g}(\mathbf{q};\boldsymbol{\delta}_p=\mathbf{0};\boldsymbol{\delta}_g=\mathbf{0}) &= (\nabla\mathcal{S})(\mathbf{q})^{\mathsf{T}}(\nabla_{\mathbf{g}}\bar{\mathcal{G}})(\mathbf{q};\mathbf{0}).
\end{aligned}$$

c) Second term.: The second term has a varying Jacobian matrix. Define:

$$e_2(\mathbf{q}) = \mathcal{P}(\mathcal{T}(\mathcal{G}(\mathbf{q};\mathbf{g}));\mathbf{p};\bar{\mathcal{G}}(\mathbf{q};\boldsymbol{\delta}_g)).$$

We expand it as:

$$e_2(\mathbf{q}) = \mathcal{T}(\mathcal{G}(\mathbf{q};\mathbf{g})) + \sum_{k=1}^{l} p^k \mathcal{A}_k(\bar{\mathcal{G}}(\mathbf{q};\boldsymbol{\delta}_g)).$$

We get:

$$\begin{aligned}\frac{\partial e_2}{\partial \delta_p^k}(\mathbf{q};\boldsymbol{\delta}_p=\mathbf{0};\boldsymbol{\delta}_g=\mathbf{0}) &= \mathbf{0}\\ \frac{\partial e_2}{\partial \boldsymbol{\delta}_g}(\mathbf{q};\boldsymbol{\delta}_p=\mathbf{0};\boldsymbol{\delta}_g=\mathbf{0}) &= \sum_{k=1}^{l} p^k (\nabla \mathcal{A}_k)(\mathbf{q})^\mathsf{T} (\nabla_\mathbf{g}\bar{\mathcal{G}})(\mathbf{q};\mathbf{0}).\end{aligned}$$

These partial derivatives are not constant since they depend on $\mathbf{p}$ which is updated every iteration.

2) Basis Images Aligned With the Target Image:

a) The error function.: We replace the argument $\mathbf{q}$ of the photometric transformation by its corresponding point in the target image, that we dub $\boldsymbol{\gamma} = \mathcal{G}(\bar{\mathcal{G}}^{-1}(\mathbf{q};\boldsymbol{\delta}_g);\mathbf{g})$. Each term of the error in (4) is thus:

$$f(\mathbf{q}) = \mathcal{S}(\mathbf{q}) - \bar{\mathcal{P}}^{-1}(\mathcal{P}(\mathcal{T}(\boldsymbol{\gamma});\mathbf{p};\boldsymbol{\gamma});\boldsymbol{\delta}_p;\boldsymbol{\gamma}).$$

Using the inverse compositional trick on the photometric transformation gives:

$$f(\mathbf{q}) \approx \bar{\mathcal{P}}(\mathcal{S}(\mathbf{q});\boldsymbol{\delta}_p;\boldsymbol{\gamma}) - \mathcal{P}(\mathcal{T}(\boldsymbol{\gamma});\mathbf{p};\boldsymbol{\gamma}).$$

Switching the incremental photometric and geometric transformations and using the inverse compositional trick on the geometric transformation gives:

$$f(\mathbf{q}) \approx \bar{\mathcal{P}}(\mathcal{S}(\bar{\mathcal{G}}(\mathbf{q};\boldsymbol{\delta}_g));\boldsymbol{\delta}_p;\mathcal{G}(\mathbf{q};\mathbf{g})) - \mathcal{P}(\mathcal{T}(\mathcal{G}(\mathbf{q};\mathbf{g}));\mathbf{p};\mathcal{G}(\mathbf{q};\mathbf{g})).$$

b) Second term.: The second term $f_2 = \mathcal{P}(\mathcal{T}(\mathcal{G}(\mathbf{q};\mathbf{g}));\mathbf{p};\mathcal{G}(\mathbf{q};\mathbf{g}))$ rewrites as:

$$f_2(\mathbf{q}) = \mathcal{P}(\mathcal{T}(\mathcal{G}(\mathbf{q};\mathbf{g}));\mathbf{p};\mathcal{G}(\mathbf{q};\mathbf{g})) = (\mathcal{P}(\mathcal{T};\mathbf{p}))(\mathcal{G}(\mathbf{q};\mathbf{g})).$$

This is the warped image $\mathcal{W}$, which does not depend on the unknown incremental parameters $\boldsymbol{\delta}_g$ and $\boldsymbol{\delta}_p$.

c) First term.: The first term has a varying Jacobian matrix. Define:

$$f_1(\mathbf{q}) = \bar{\mathcal{P}}(\mathcal{S}(\bar{\mathcal{G}}(\mathbf{q};\boldsymbol{\delta}_g));\boldsymbol{\delta}_p;\mathcal{G}(\mathbf{q};\mathbf{g})).$$

We expand it as:

$$f_1(\mathbf{q}) = \mathcal{S}(\bar{\mathcal{G}}(\mathbf{q};\boldsymbol{\delta}_g)) + \sum_{k=1}^{l} \delta_p^k \mathcal{A}_k(\mathcal{G}(\mathbf{q};\mathbf{g})).$$

We get:

$$\begin{aligned}\frac{\partial f_1}{\partial \boldsymbol{\delta}_g}(\mathbf{q};\boldsymbol{\delta}_p=\mathbf{0};\boldsymbol{\delta}_g=\mathbf{0}) &= (\nabla\mathcal{S})(\mathbf{q})^\mathsf{T}(\nabla_\mathbf{g}\bar{\mathcal{G}})(\mathbf{q};\mathbf{0})\\ \frac{\partial f_1}{\partial \delta_p^k}(\mathbf{q};\boldsymbol{\delta}_p=\mathbf{0};\boldsymbol{\delta}_g=\mathbf{0}) &= \bar{\mathcal{A}}_k(\bar{\mathcal{G}}(\mathbf{q};\mathbf{g})).\end{aligned}$$

So the partial derivatives with respect to $\boldsymbol{\delta}_g$ are constant but those with respect to the δ_p^k are varying – they are the warped basis images, depending on the current geometric parameters which are updated every iteration.

C. Experimental Results on Real Data

Our goal is to validate our algorithms and compare them to different algorithms on real data sets, as well as to evaluate the photometric transformations presented in §V, and to compare the results obtained when using grey-level and color images. We show results for the pair of images shown on figure 1. They have been used in figures 1 and 1 to illustrate the inverse compositional and the dual inverse compositional algorithms. They were acquired by the same camera under different lighting conditions, namely natural daylight and electric light. The image resolution is 640×480. The initial warp was chosen as the identity since the images are close enough to enable convergence to the sought after solution for all methods. The number of pixels of interest chosen near the image edges is 53,889.

We launched the simultaneous inverse compositional, the project out and the dual inverse compositional algorithms with the images converted to grey-level for estimating gain and bias and with the color images and different photometric models (single gain and bias, multiple gains and biases and full affine channel mixing). We show the results in figure 4 for the grey-level gain and bias (2 parameters) and the color full affine cases (12 parameters). Note that for PO, we compute the best photometric transformation at each iteration to measure the photometric error, but this is not counted into the measured computational time.

In the grey-level, gain and bias case, PO converges to the solution in 53 iterations, while DIC and SIC both requires 79 iterations and behave very similarly. All three of them converge to the same solution.

In the color, affine case, SIC converges first, in 69 iterations, followed by DIC which requires 95 iterations. Finally, PO uses 204 iterations to converge. All three algorithms converge to the same solution.

We observe that the first iteration increases the photometric error for both SIC and DIC. The magnitude of error variation, both at the increasing and decreasing phases, is strongly related to the number of parameters in the photometric model. In other words, the more flexible the photometric model, the steepest the error variation. This behavior is discussed below.

All three algorithms diverge when no photometric model is used. Figure 5 shows the error image at convergence for the different photometric model.

Finally, we report the total computational time in seconds for each algorithm and each photometric model:

	G&B	Single G&B	Multiple G&B	Full Affine
SIC	20.09	50.29	54.72	89.02
PO	3.88	8.57	12.81	15.24
DIC	5.95	7.03	7.45	7.81

It is clear that SIC is the most expensive algorithm in all cases, being 4 to more than 10 times slower than DIC, while PO is faster in the grey-level gain and bias case, but slower in all the three color cases.

We observed in our experiments that the photometric error measured throughout the iterations is often increased by the first iteration and, very rarely, by the second iteration for SIC and DIC. This phenomenon is thus not related to the form of the update rule. We have the following hints to understand this behavior:

- The higher the number of parameters in the photometric transformation, the steepest the photometric error curve, both at the increasing and decreasing phases.

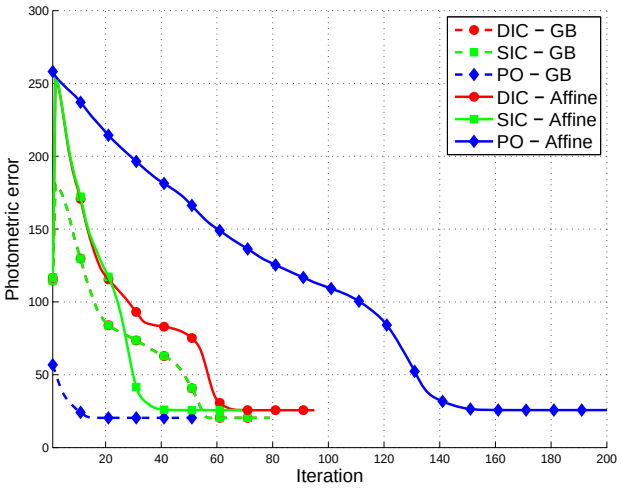


Fig. 4. Color photometric error through the iterations for the image pair shown in figure 1 for different photometric transformations and the dual inverse compositional algorithm.

Fig. 5. The error images for different color photometric transformations applied to the image pair in figure 1 with the dual inverse compositional algorithm.

- Using edge pixels only make stronger this behavior compared to using the whole region of interest.

The geometric error reflects the closeness of the estimated warp to the true one. Using simulated data, we assessed the change in the geometric error caused by the first iteration. As already observed by Baker *et al.* [1], it always decreases. This holds true even if the photometric error increases. In other words, the first iteration brings the warp closer to the sought after solution, while it takes the photometric transformation away from it, since the value of the error function, namely the photometric error, increases.

The reasons are as follows. The initial warp causes a geometric misalignment of the images. The intensity or color correspondences from which the photometric transformation is estimated are thus erroneous. All tested algorithms are based on a Gauss-Newton approximation of the error function, which is not fully second-order, and thus do not fully preserve the tight coupling between the incremental warp and photometric transformation. The badly estimated incremental photometric transformation, along with the incremental warp, can thus make the photometric error grow, as we observed. This phenomenon is amplified by the fact that the error function is an affine function of the photometric transformation parameters that thus exactly fits the intensity or color correspondences conditioned on the current warp estimate.

6.1.3 Paper (SCIA'07) – *Shadow Resistant Direct Image Registration*

Shadow Resistant Direct Image Registration

Daniel Pizarro[1] and Adrien Bartoli[2]

[1] Department of Electronics, University of Alcala
Alcala de Henares, Spain
pizarro@depeca.uah.es
[2] LASMEA – CNRS / Université Blaise Pascal
Clermont-Ferrand, France
adrien.bartoli@univ-bpclermont.fr

Abstract. Direct image registration methods usually treat shadows as outliers. We propose a method which registers images in a 1D shadow invariant space. Shadow invariant image formation is possible by projecting color images, expressed in a log-chromaticity space, onto an 'intrinsic line'. The slope of the line is a camera dependent parameter, usually obtained in a prior calibration step. In this paper, calibration is avoided by jointly determining the 'invariant slope' with the registration parameters. The method deals with images taken by different cameras by using a different slope for each image and compensating for photometric variations. Prior information about the camera is, thus, not required. The method is assessed on synthetic and real data.

Key words: Direct Registration, Shadow Invariant, Photometric Camera Calibration

1 Introduction

The registration of image pairs consists in finding the transformation that best fits two images. That has been a key issue in computer vision, robotics, augmented reality and medical imagery. Although it was thoroughly studied in the past decades, there remain several open problems. Roughly speaking, there are two kinds of approaches: direct and feature based methods. The formers rely on fiducial points described by local properties, which allows matching despite geometric and photometric transformations. The geometric registration is thus formed by minimizing an error between the fiducials position expressed in pixels. As opposed to the local approach, direct methods use pixel discrepancy as a registration error measure. The brightness constancy assumption states that pixel values are equivalent under the sought after transformation. The warp relating two images consists of some geometrical transformation (*e.g.* an homography or an affine transformation) and some photometric model (*e.g.* channel intensity bias and gain or full affine channel mixing).

One of the main problems that arises in direct methods is the existence of partial illumination or shadow changes in the scene to register. In such cases,

the brightness constancy assumption is violated. This paper addresses the problem of directly registering such kind of images. Our proposal is based on expressing the error in a transformed space different from the usual one based on image intensities. In this space which is onedimensional the change of illumination or shadows are removed. This invariant space is governed by a single parameter which is camera-dependent and defines a transformation between the log-chromaticity values of the original RGB image and the invariant image. We propose a method for jointly computing the sought after geometric registration and the parameters defining the shadow invariant space for each image.

Paper Organization

We review previous work and give some background in §2. In §3 we state our error function and give an algorithm for effectively registering images in §4. Results on synthetic and real images are presented in §5. Finally, conclusions are presented in §6.

2 Previous Work

The content of this section is divided into two major parts. First, some previous work about direct image registration is briefly described. The general approach and the most common problems are described. Secondly, some background on color image formation is presented, necessary for describing the process of shadow invariant image formation, which is finally stated.

2.1 Direct Image Registration

The registration of two images is a function $\mathcal{P}$, which models the transformation between a source image, $\mathcal{S}$ and a target image $\mathcal{T}$ over a region of interest $\mathcal{R}$. Function $\mathcal{P}(\mathcal{T}(q), q; \phi)$ is parametrized by a vector ϕ composed of geometric and photometric parameters in the general case.

The error function to be minimized make is the sum of square differences of intensity values, over the parameter vector ϕ.

The problem is formally stated as:

$$\min_{\phi} \sum_{q \in \mathcal{R}} \|\mathcal{S}(q) - \mathcal{P}(\mathcal{T}(q), q; \phi)\|^2. \tag{1}$$

A linearization of each residual, which allows to solve it in an iterative Linear Least Square fashion, was popularized by the Lucas-Kanade algorithm [1]. There exist remarkably fast approaches for warps functions forming groups. It is known as the Inverse Compositional algorithm [2] and it has been successfully applied with geometric transformations and in [3] an affine photometric model is also included.

The presence of shadows or illumination changes affect the applicability of equation (1), producing registration errors or divergence in the algorithm. There

exist plenty of proposals in the literature to extend the direct registration with a certain grade of immunity against perturbations. The most common approach is to mark shadow areas as outliers. The use of robust kernels inside the minimization process [4] allows the algorithm to reach a solution. Other approaches try to model the shadows and changes of illumination. In [5] a learning approach is used to tackle illumination changes by using a linear appearance basis.

2.2 Background on Color Image Formation

We present the physical model used to describe the image formation process. The theory of invariant images is described later in terms and under the assumptions stated below.

We consider that all the surfaces are lambertian, that the lights follow a planckian model and that the camera sensor is narrow-band. The RGB color obtained at a pixel is modeled by the following physical model:

$$\rho_k = \sigma S(\lambda_k) E(\lambda_k, T) Q_k \delta(\lambda - \lambda_k) \quad k = 1, 2, 3, \tag{2}$$

where $\sigma S(\lambda_k)$ represents the surface spectral reflectance functions times the lambertian factor. The term $Q_k \delta(\lambda - \lambda_k)$ represents the sensor spectral response function for each color channel k centered at wavelength λ_k. $E(\lambda_k, T)$ is the spectral power distribution of the light in the planckian model. This is modeled by the following expression:

$$E(\lambda, T) = I c_1 \lambda^{(-5)} e^{\left(\frac{-c_2}{T\lambda}\right)} \tag{3}$$

This model holds for a high rank of color temperatures $T = [2500^o, 10000^o]$. The term I is a global light intensity and the constants c_1 and c_2 are fixed.

According to this model, the value obtained by the camera at any pixel ρ_k is directly obtained by:

$$\rho_k = \sigma I c_1 (\lambda_k)^{-5} e^{\left(\frac{-c_2}{T\lambda_k}\right)} S(\lambda_k) Q_k. \tag{4}$$

2.3 Shadow Invariant Image Theory

The transformation which allows invariant image formation is based on the original work of [6] in which a method for obtaining an illumination invariant, intrinsic image from an input color image is developed . The method relies on the above presented image formation model, based on the assumption of lambertian surfaces, narrow-band sensors and planckian illuminants.

Given the three channel color components $\rho = (\rho_1, \rho_2, \rho_3)$ described in (4), the logarithm of chromaticity ratios are formed.

$$\begin{aligned}\mathcal{X}_1 &= \log\left(\frac{\rho_1}{\rho_3}\right) = \log(s_1/s_3) + (e_1 - e_3)/T \\ \mathcal{X}_2 &= \log\left(\frac{\rho_2}{\rho_3}\right) = \log(s_2/s_3) + (e_2 - e_3)/T,\end{aligned} \tag{5}$$

where $e_k = -c_2/\lambda_k$ only depends on camera spectral response and not on the surface and $s_k = c_1\lambda_k^{(-5)}S(\lambda_k)Q_k$ does not depend on color temperature T.

The pair of values $\mathcal{X}_1$ and $\mathcal{X}_2$ lie on a line with direction vector $\bar{e} = (e_1 - e_3, e_2 - e_3)$. Across different illumination temperature T, vector $\mathcal{X} = (\mathcal{X}_1, \mathcal{X}_2)$ moves along the line.

An illumination invariant quantity can be formed by projecting any vector $\mathcal{X}$ onto the orthogonal line defined by $\bar{e}^{\perp} = (\cos(\theta), \sin(\theta))$. Therefore, two pixels from the same surface viewed under different illuminations get projected at the same place.

To reduce the arbitrary election of the chromaticity ratios, in [6], is proposed a method to use the geometrical mean $(\rho_1\rho_2\rho_3)^{(1/3)}$ of the three channel values as denominator. A vector of three linearly dependent coordinates is obtained. By choosing a proper decomposition, a twodimensional equivalent vector $\mathcal{X}$ is obtained that preserve the essential properties of equation (5).

The transformation $\mathcal{L}$ is simply obtained by projecting vector $\mathcal{X}$ onto the invariant line parametrized by its slope angle θ:

$$\mathcal{L}(\rho, \theta) = \mathcal{X}_1(\rho)\cos(\theta) + \mathcal{X}_2(\rho)\sin(\theta) \tag{6}$$

This transformation, as it has been previously stated, represents the mapping between a color image and its corresponding shadow invariant representation. By explicitly describing the whole color image $\mathcal{S}$ as an input in (6), the result of $\mathcal{L}(\mathcal{S}, \theta)$ is a 1D shadow invariant image. The transformation is therefore global so it does not depend on pixel position $q \in \mathbb{R}^2$, but only on its color value.

The slope angle θ of the invariant line only depends on camera spectral properties, so it varies across different cameras. In [6] it is presented a method to obtain the slope by a calibration step using a color pattern or by a set of preregistered images from the same camera under illumination changes. In [7] an autocalibration approach is presented by finding the slope for which the entropy of the invariant image is minimum. The later method is proved to be capable to find the correct slope with only one image.

The entropy based method unless simple and powerful requires images in which remarkable shadow areas are present. In the case of images in which the change of illumination is global and no shadow is present, the method is not able to produce the correct slope.

3 Joint Image Registration and Photometric Camera Calibration

Image registration is proposed under invariant transformation $\mathcal{L}(\mathcal{S}(q), \theta)$, applied to both the target and the source image.

The new cost function, is expressed as follows:

$$\min_{\phi,\theta_1,\theta_2} \sum_{q\in\mathcal{R}} \|\mathcal{L}(\mathcal{S}(q), \theta_1) - \mathcal{L}(\mathcal{P}(\mathcal{T}, q; \phi), \theta_2)\|^2 \tag{7}$$

As previously stated, the slope parameter θ is, in general, different in both images (θ_1 and θ_2). According to the camera model, any change of illumination intensity and temperature will be discarded in the invariant image once parameter θ is obtained. Therefore, in theory, $\mathcal{P}$ can only be geometrical.

The validity of (7) is based on the assumption that in different cameras, intrinsic images are comparable. However as is stated in this section, in general such an hypothesis does not hold due to differences in camera response functions. A photometric model is proposed for compensating such differences.

3.1 Camera Response Dependent Parameters

In this section it will be shown that besides the slope, between two cameras it is of importance the inclusion of photometric parameters over RGB space so that the invariant space of two images is directly comparable. Such parameters will not try to compensate for global illumination as in previous attempts [3], but instead they will represent a compensation between different camera responses.

Multiple Gain Compensation Assuming that each camera has similar spectral response, so that the values of λ_k are similar, the slope and surface reflectance will produce similar values. However for different channel gains Q_k the log-chromaticity values are affected.

It is thus reasonable to include multiple gains compensation a_k per channel for the target image before computing its log-chromaticity values:

$$\mathcal{X}_k = \log\left(\frac{a_k\rho_k}{a_3\rho_3}\right) = \log(a_k/a_3) + \log(\rho_k/\rho_3) \tag{8}$$

According to (6), the projection reduces the photometric compensation into a one dimensional offset $d_{\mathcal{L}}$.

$$\mathcal{L}(\rho, \theta) = \log\left(\frac{\rho_1}{\rho_3}\right)\cos(\theta) + \log\left(\frac{\rho_2}{\rho_3}\right)\sin(\theta) + d_{\mathcal{L}}, \tag{9}$$

where $d_{\mathcal{L}} = \log(a_1/a_3)\cos(\theta) + \log(a_2/a_3)\sin(\theta)$.

In the case where both cameras where different by only constant gains, it is still enough as a way to compensate camera responses, to compute a single offset.

Multiple gain and bias for each channel compensation As stated in [8], the real response for most digital cameras is not linear. Under certain range of values we can consider that the camera response can be approximated by a gain and bias function. The presence of bias over RGB represents a problem since the assumption of the invariant line is no longer valid.

Adding bias and gain over RGB results in the following invariant representation:

$$\mathcal{L}(\rho,\theta) = \log\left(\frac{\rho_1 + b_1}{\rho_3 + b_3}\right)\cos(\theta) + \log\left(\frac{\rho_2 + b_2}{\rho_3 + b_3}\right)\sin(\theta) + d_{\mathcal{L}}. \tag{10}$$

Where $d_{\mathcal{L}}$ is the same commented in the multiple gain model, and b_k are biases added to color values.

The total number of required photometric parameters is four under the assumption that only one of the cameras suffer from the bias problem. In the case that both images are suitable to be affected, an extra bias model is introduced for the source image. For such critical case the number of parameters is increased to seven.

The new cost function, which includes photometric parameters (ϕ_p^1, ϕ_p^2) in source and target images, is presented:

$$\min_{\phi,\theta_1,\theta_2,\phi_p^1,\phi_p^2} \sum_{q\in\mathcal{R}} \|\mathcal{L}(\mathcal{S}(q),\theta_1,\phi_p^1) - \mathcal{L}(\mathcal{P}(\mathcal{T},q;\phi),\theta_2,\phi_p^2)\|^2 \tag{11}$$

Besides the commented models, specially amateur cameras suffer from many artificial perturbations which includes saturation boosting, channel mixing and digital filters applied to the raw image sensed.

4 Minimizing the Error Function

In this section, the optimization process involved in obtaining image registration and invariant space parameters is presented in details.

Given the more general expression (11), which includes a photometric model, a Gauss-Newton approach is derived as an optimization method.

A first order approximation of the warped image around current estimation of parameters $\Phi = (\phi, \theta_1, \theta_2, \phi_S, \phi_{\mathcal{T}})$ is obtained. The residue in the error function is also renamed by using two different functions $\mathcal{W}_1$ and $\mathcal{W}_2$ depending on vector Φ.

The renamed cost function becomes:

$$\min_{\Phi} \sum_{q\in\mathcal{R}} \|\mathcal{W}_1(\mathcal{S}(q),\Phi) - \mathcal{W}_2(\mathcal{T}(q),q,\Phi)\|^2. \tag{12}$$

The Gauss-Newton approximation is:

$$\epsilon^2 \approx \sum_{q\in\mathcal{R}} \|\mathcal{W}_1(\mathcal{S}(q),\Phi) - \mathcal{W}_2(\mathcal{T}(q),q,\Phi) + (L_{\mathcal{W}_1}(q) + L_{\mathcal{W}_2}(q))\Delta\Phi\|^2, \tag{13}$$

where $L_{\mathcal{W}_1}(q)$ and $L_{\mathcal{W}_2}(q)$ represents respectively the first derivatives of functions $\mathcal{W}_1$ and $\mathcal{W}_2$. ϵ^2 is the residual error from the cost function to minimize.

The parameter increment $\Delta\Phi$ is given by solving the following linear system:

$$E_\Phi \Delta_\Phi = b_\Phi, \tag{14}$$

where E_Φ represents the approximated Hessian of the error function:

$$E_\Phi = \sum_{q \in \mathcal{R}} (L_{\mathcal{W}_1}(q) + L_{\mathcal{W}_2}(q))(L_{\mathcal{W}_1}(q) + L_{\mathcal{W}_2}(q))^T. \tag{15}$$

The right hand side of the linear system b_Φ includes the error image:

$$b_\Phi = \sum_{q \in \mathcal{R}} (L_{\mathcal{W}_1}(q) + L_{\mathcal{W}_2}(q))(\mathcal{W}_1(\mathcal{S}(q), \Phi) - \mathcal{W}_2(\mathcal{T}(q), q, \Phi)). \tag{16}$$

Once the increment $\Delta\Phi$ is obtained Φ is updated accordingly with each model.

4.1 Using an Homography for the Geometric Model

The used geometric model consists of an homography transformation. Homographies are fully representative as global geometrical models. They are suitable for registering planar scenes or under camera rotation. It is a groupwise homogeneous transformation represented by a full rank 3×3 matrix H with eight degrees of freedom. The homography is applied to the homogeneous coordinates q in the target image for composing the warp. It is assumed that the first eight coordinates of vector Φ represent the values of ΔH at each iteration.

5 Experimental Results

In this section some of the results are presented in order to validate the proposal. The experiments are designed to compare the convergence properties of our algorithm and to test the photometric models we proposed.

5.1 Synthetic Image Registration

A set of synthetic images is generated according to the model presented in §2. Each image consists of a set of quadrangular color patches under different illuminations, covering a range of color temperatures from 2500^o to 10000^o. By modifying camera response parameters we are able to simulate images taken by different cameras. Two models are considered for the experiment.

- *Multiple gains*: described by only one bias parameter $d_\mathcal{L}$ in the invariant space.
- *Multiple gains and biases*: Considering the complete case, the model has five parameters for the target image $\phi_{p\mathcal{T}}$ and three parameters $\phi_{p\mathcal{S}}$ for the source image.

compared Algorithms In all the experiments the following algorithms are compared:

- DRSI-NP: Direct Registration in Shadow Invariant space with No Photometric model to compensate between cameras.
- DRSI-MG: Direct Registration in Shadow Invariant space with Multiple Gains as a photometric model to compensate between cameras.
- DRSI-MGB: Direct Registration in Shadow Invariant with Multiple Gains and Biases in target image and only bias in source image.
- DR: Direct Registration over greylevel values

Simulation Setup Given two differently illuminated sequences of patches, we simulate a 2D homography by displacing the corners in the target image in random directions by some value γ with default value of 5 pixels. The target image is contaminated by gaussian noise with variance σ and a default value of 25.5. The value of photometric parameters for target and source image has been chosen fixed for the simulations: $\phi_{\mathcal{T}} = (b_1 = 3.2, b_2 = 2.1, b_3 = 1)$ and $\phi_{\mathcal{S}} = (b_1 = 0.8, b_2 = 4, b_3 = 3.5)$. Both target and source image has a slope parameter of $\theta_1 = \theta_2 = 169.23^o$. Interest area $\mathcal{R}$ is obtained by using strong edges in greylevel image and dilating them by a factor of 8..

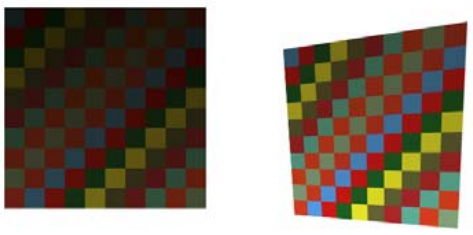
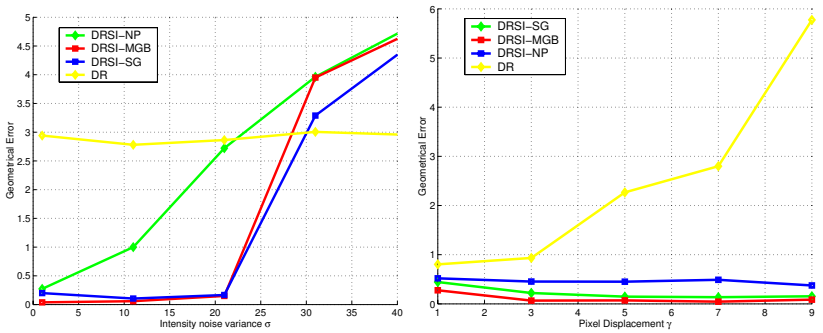


Fig. 1. Pair of synthetic images

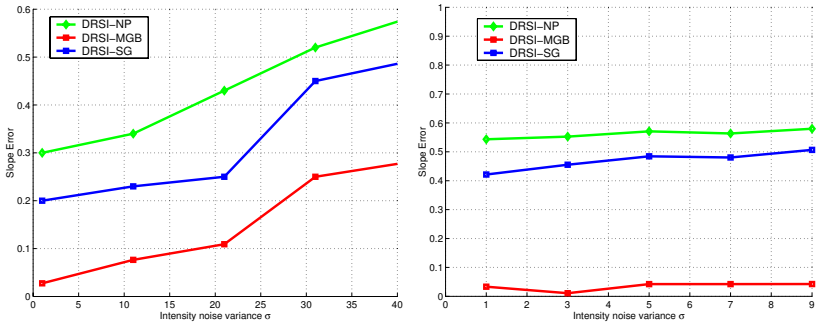
Results In Figure 2.a and 2.b the geometric error is presented against noise variance σ and initial pixel displacement γ. In Figure 2.c and 2.d the slope angle error is presented against noise variance σ and initial pixel displacement γ.

5.2 Real Image Registration

For testing the presented proposal with real images, the same planar surface is acquired with two different low-cost commercial cameras. By manually clicking in the four corners of the planar shape, an interest area $\mathcal{R}$ is obtained. If the two regions are far from 10 pixels of displacement a pre-registration is used using manual clicked points. The results presented show the error measured between a pair of images transformed into its respective invariant spaces across the iterations.

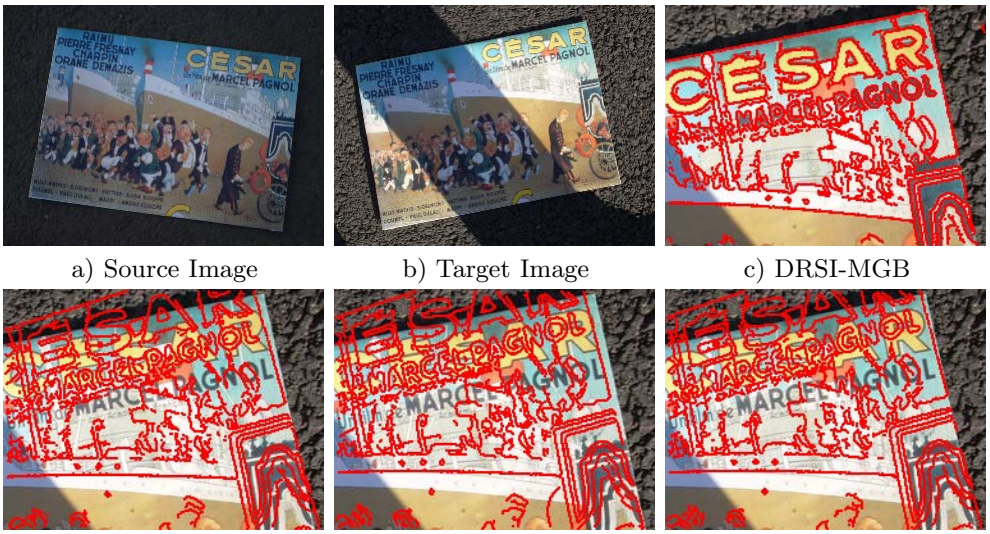


a)Geometric vs σ b) Geometric vs γ



c) Slope error vs σ d) Slope error vs γ

Fig. 2. Residual Error vs σ and Geometrical Error vs γ



a) Source Image b) Target Image c) DRSI-MGB

a) DRSI-SGB b) DRSI-NP c) DR2

Fig. 3. Real images and its resulting registration

6 Conclusions

A new method to achieve direct image registration in the presence of shadows is proposed. The approach is based on minimizing the registration error directly in a transformed space from RGB space. The new space is parametrized by a single camera dependent parameter, the invariant line slope. Such parameter is in general different from each camera, so it is included in the optimization stage. Solving registration parameters in the invariant space from images taken by different cameras offers difficulties due to the response function of each camera. In this paper, two models are proposed to compensate such differences: Multiple Gain compensation and Multiple Gain and Bias. Results on synthetic data show that the last one obtains better registration performance against pixel displacement and noise. In real images, under some conditions the use of the multiple gains and biases model is crucial to achieve registration. If both cameras are of similar response, the simple algorithm which avoid photometric model calculations is the best choice. The use of invariant space in direct methods allows to avoid shadows directly without the need of using complex methods which use illumination modeling or robust kernel optimization.

References

[1] B. Lucas, T.Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 1981.

[2] S. Baker, I.Matthews. Lucas-Kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221–255, 2004.

[3] A. Bartoli. Groupwise Geometric and Photometric Direct Image Registration. BMVC'06 *Proceedings of the Seventeenth British Machine Vision Conference*, Edinburgh, UK, pp.11–20, September 2006

[4] S. Baker, R. Gross, I. Matthews, T. Ishikawa, Lucas-Kanade 20 Years ON: A Unifying Framework: Part 2, tech. report CMU-RI-TR-03-01, Robotics Institute, Carnegie Mellon University, February, 2003

[5] S. Baker, R. Gross, I. Matthews, T. Ishikawa, Face Recognition Across Pose and Illumination, *Handbook of Face Recognition*, Stan Z. Li and Anil K. Jain, ed., Springer-Verlag, June, 2004

[6] G.D. Finlayson, S.D. Hordley, and M.S. Drew. Removing shadows from images. In ECCV 2002: European Conference on Computer Vision, pages 4:823–836, 2002

[7] G. Finlayson, M. Drew, C. Lu, Intrinsic Images by Entropy Minimization, *Proceedings of 8th European Conference on Computer Vision, Praque, pp 582–595, 2004*

[8] Kobus Barnard, Brian Funt, Camera characterization for color research, Color Research and Application, Vol. 27, No. 3, pp. 153–164, 2002

6.2 Estimation of Deformable Image Warps

I34 **Generalized Thin-Plate Spline Warps** §6.2.1
A. Bartoli, M. Perriollat and S. Chambon
CVPR'07 - IEEE *Int'l Conf. on Computer Vision and Pattern Recognition*, Minneapolis, USA, June 2007

J11 **Maximizing the Predictivity of Smooth Deformable Image Warps Through Cross-Validation** §6.2.2
A. Bartoli
Journal of Mathematical Imaging and Vision, special issue: tribute to Peter Johansen, accepted December 2007

I18 **Direct Estimation of Non-Rigid Registrations** §6.2.3
A. Bartoli and A. Zisserman
BMVC'04 - *British Machine Vision Conf.*, London, UK, p. 899-908, vol. II, September 2004

I40 **Feature-Driven Direct Non-Rigid Image Registration** §6.2.4
V. Gay-Bellile, A. Bartoli and P. Sayd
BMVC'07 - *British Machine Vision Conf.*, Warwick, UK, September 2007
Version in French: [N12]

I46 **Direct Estimation of Non-Rigid Registrations with Image-Based Self-Occlusion Reasoning** §6.2.5
V. Gay-Bellile, A. Bartoli and P. Sayd
ICCV'07 - IEEE *Int'l Conf. on Computer Vision, Rio de Janeiro, Brazil*, October 2007
Version in French: [N14]
Related papers: [I47,N13]

6.2.1 Paper (CVPR'07) – *Generalized Thin-Plate Spline Warps*

Generalized Thin-Plate Spline Warps

Adrien Bartoli
LASMEA (CNRS / UBP)
Clermont-Ferrand, France
Adrien.Bartoli@gmail.com

Mathieu Perriollat
LASMEA (CNRS / UBP)
Clermont-Ferrand, France
Mathieu.Perriollat@gmail.com

Sylvie Chambon
LTCI (CNRS / ENST) – Paris, France
IRIT (CNRS / UPS) – Toulouse, France
schambon@enst.fr

Abstract

Thin-Plate Spline warps have been shown to be very effective as a parameterized model of the optic flow field between images of various deforming surfaces. Examples include a sheet of paper being manually handled. Recent work has used such warps for images of smooth rigid surfaces. Standard Thin-Plate Spline warps are not rigid, in the sense that they do not satisfy the epipolar geometry constraint, and are intrinsically affine, in the sense of the affine camera model.

We propose three types of warps based on the Thin-Plate Spline. The first one is a flexible rigid warp. It describes the optic flow field induced by a smooth rigid surface, and satisfies the affine epipolar geometry constraint. The second and third ones extend the standard Thin-Plate Spline and the proposed rigid flexible warp to the perspective camera model. The properties of these warps are studied in details, and a hierarchy is defined. Experimental results on simulated and real data are reported, showing that the proposed warps outperform the standard one in several cases of interest.

1. Introduction

Given two images of some scene surface, there exists an $\mathbb{R}^2 \rightarrow \mathbb{R}^2$ function, called a warp, mapping a point from the first image to the corresponding point in the second image. For instance, two images of a rigid planar surface taken by two perspective cameras are related by an homographic warp. For a non-planar 3D scene, the warp is more complex since it depends on the surface depth. When the observed surface deforms, the warp is even more involved. Examples of rigid scene models include piecewise or nearly planar structures. Examples of deformable scene models include the flexible low-rank shape and face models.

Representing the warp using a parametric function requires prior assumptions about the observed scene structure. One common, fairly generic assumption is that a smooth surface is observed. This naturally leads to using the Thin-Plate Spline (TPS) as a building block for the warps. TPS are smooth, compact and convenient, $\mathbb{R}^2 \rightarrow \mathbb{R}$ functions. Standard TPS warps, built by 'stacking' a pair of TPS, are very flexible in that they are controlled by centres that may be placed anywhere in the images. They are known to be effective approximations to many types of deformations, see *e.g.* [1]. Standard TPS warps have recently been used as simple parametric warps for images of rigid 3D surfaces by Wills and Belongie [8] and Masson *et al.* [5], for respectively wide-baseline matching and object tracking.

There are, however, two main issues with the use of standard TPS warps in this context, that have not been dealt with in the literature. *(i)* **Standard TPS warps overfit affine images of rigid surfaces.** They do not in general satisfy the rigidity constraint modeled by the affine epipolar geometry. In that sense they are 'too flexible' in affine imaging conditions. *(ii)* **Standard TPS warps do not model perspective projection.** They are intrinsically affine, in the sense of the affine camera model, since their formulation does not include a division. For instance, as mentioned in [8], they are not able to 'reproduce' simple homographic warps with a finite number of centres. Henceforth, we call *DA-Warps* the standard TPS warps (for 'Deformable Affine').

This paper addresses the two above-mentioned issues. First, the DA-Warps are specialized to rigid surfaces in §4. These warps are called *RA-Warps* (for 'Rigid Affine') and are very similar to DA-Warps with an epipolar constraint on the warp coefficients. This solves the first issue. Second, the RA-Warps are extended to the perspective camera model in §5. These warps, dubbed *RP-Warps* (for 'Rigid Perspective'), naturally include FP-Warps ('Flat Perspective') similarly to the RA-Warps including FA-Warps[1]. This solves the second issue for the case of rigid surfaces. Third, we introduce the *DP-Warps* (for 'Deformable Perspective') which are shown to be the perspective analogue of the DA-Warps. This solves the second issue for the case of deformable surfaces. The derivation of these warps is made possible by a feature-driven parameterization of the Thin-Plate Spline

[1]The FP-Warps are 2D homographic warps with 8 parameters. The FA-Warps (for 'Flat Affine') are 2D affine warps with 6 parameters.

we propose in §3. The hierarchy and dependencies between the six types of warps mentioned so far is studied in details in §6. In order to derive warps independent of the intrinsic camera parameters, we consider uncalibrated cameras.

The second line of contributions in this paper, presented in §7, is a set of algorithms for estimating the proposed warps from image point correspondences. Experimental results are reported in §8 and our contributions discussed in §9. Most proofs of our statement will appear in an extended version of the paper.

2. Preliminaries

Previous work. DA-Warps, *i.e.* standard TPS warps, are used in many different contexts. While there is a great body of work on defining alternative warps, such as FFD (Free-Form Deformations) [6] or more recently diffeomorphic warps, DA-Warps are usually used in their original form. Since the seminal paper by Bookstein [1], the literature is mostly focused on estimation methods. Bookstein proposed a method relying on point landmark correspondences, that are chosen as centres for the DA-Warp. The DA-Warp and point matching are simultaneously estimated using the soft-assign in [2]. Algorithms to make faster the computation of DA-Warps from point correspondences are proposed in [3]. Several papers use the integral bending energy for 3D surface reconstruction, for instance, as one of the terms in an energy functional, see *e.g.* [7]. In contrast, we build on the existing DA-Warps to derive new warps by taking into account the possible rigidity of the observed surface and the perspective camera model.

Notation. Scalars are in italics, *e.g.* x, vectors in bold right fonts, *e.g.* $\mathbf{q}$, and matrices in sans-serif and calligraphic fonts, *e.g.* P and $\mathcal{E}$. The elements of a vector are written as in $\mathbf{a}^\mathsf{T} = (a_1 \;\; a_2 \;\; a_3)$ where $^\mathsf{T}$ is vector and matrix transpose. We do not make a difference between coordinate vectors and physical entities. The coordinates of a point in the first image are written with a 2-vector $\mathbf{q}^\mathsf{T} = (x \;\; y)$. $\mathbb{R}^r$ and $\mathbb{P}^r$ designate respectively the Euclidean and projective spaces of dimension r. We write $d^2(\mathbf{q}, \mathbf{q}') = \|\mathbf{q} - \mathbf{q}'\|^2$ the Euclidean distance between two points $\mathbf{q}$ and $\mathbf{q}'$ with $\|\cdot\|^2$ the vector two-norm and matrix Frobenius norm. Homogeneous coordinates are written $\check{\mathbf{q}}^\mathsf{T} \sim (\mathbf{q}^\mathsf{T} \;\; 1)$, where $\sim$ means equality up to scale. Scaled homogeneous coordinates are written $\tilde{\mathbf{q}}^\mathsf{T} = (\mathbf{q}^\mathsf{T} \;\; 1)$. The homogeneous to affine coordinate function ψ is defined by $\mathbf{q} = \psi(\check{\mathbf{q}})$. The skew-symmetric (3×3) cross-product matrix $[\check{\mathbf{q}}]_\times$ is defined such that $[\check{\mathbf{q}}]_\times \check{\mathbf{q}}' = \check{\mathbf{q}} \times \check{\mathbf{q}}'$. Full column rank portrait matrix pseudo-inverse is defined by $\mathsf{X}^\dagger = \left(\mathsf{X}^\mathsf{T}\mathsf{X}\right)^{-1}\mathsf{X}^\mathsf{T}$.

We consider l centres with coordinates $\mathbf{c}_k$ in the first image, with $k = 1, \ldots, l$. They are gathered in an $(l \times 2)$ matrix P containing their x and y coordinates on its columns, and an $(l \times 3)$ matrix $\tilde{\mathsf{P}}$ with a third column of ones, *i.e.* $\tilde{\mathsf{P}} = (\mathsf{P} \;\; \mathbf{1})$. Matrix $\check{\mathsf{P}}$ equals matrix $\tilde{\mathsf{P}}$ with each row rescaled by some scalar factor, *i.e.* $\check{\mathsf{P}} = \mathrm{diag}(\mathbf{d})\tilde{\mathsf{P}}$. The centres in the second image are written $\mathbf{c}'_k$. Matrices P', $\tilde{\mathsf{P}}'$ and $\check{\mathsf{P}}'$ are defined similarly as for the first image.

Warps in affine coordinates are written $\mathcal{W}$, while $\tilde{\mathcal{W}}$ and $\check{\mathcal{W}}$ are used for scaled homogeneous and homogeneous coordinates respectively. The sets of flat affine warps and flat perspective warps (*i.e.* homographic warps) are respectively denoted $\mathbb{S}_{\text{FA}}$ and $\mathbb{S}_{\text{FP}}$. For a (2×3) flat affine warp matrix A and a (3×3) flat perspective warp matrix H, we have:

$$\mathcal{W}_{\text{FA}}(\mathbf{q}; \mathsf{A}) \stackrel{\text{def}}{=} \mathsf{A}\tilde{\mathbf{q}} \qquad \text{and} \qquad \check{\mathcal{W}}_{\text{FP}}(\mathbf{q}; \mathsf{H}) \stackrel{\text{def}}{\sim} \mathsf{H}\tilde{\mathbf{q}}.$$

Thin-Plate Splines. The TPS is an $\mathbb{R}^2 \to \mathbb{R}$ function driven by assigning target values α_k to the l 2D centres $\mathbf{c}_k$ and enforcing several conditions: the TPS is the Radial Basis Function (RBF) that minimizes the integral bending energy. It is usually parameterized by an $l+3$ coefficient vector $\mathbf{h}_{\boldsymbol{\alpha},\lambda}^\mathsf{T} = (\;\mathbf{w}^\mathsf{T} \quad \mathbf{a}^\mathsf{T})$ computed from the target vector $\boldsymbol{\alpha}$ and a regularization parameter $\lambda \in \mathbb{R}^+$. The coefficients in $\mathbf{w}$ must satisfie $\tilde{\mathsf{P}}^\mathsf{T}\mathbf{w} = \mathbf{0}$. These three 'side-conditions' ensure that the TPS has square integrable second derivatives. The TPS is defined by:

$$\omega(\mathbf{q}, \mathbf{h}_{\boldsymbol{\alpha},\lambda}) \stackrel{\text{def}}{=} \boldsymbol{\ell}_\mathbf{q}^\mathsf{T}\mathbf{h}_{\boldsymbol{\alpha},\lambda}, \tag{1}$$

with $\boldsymbol{\ell}_\mathbf{q}^\mathsf{T} \stackrel{\text{def}}{=} (\rho(d^2(\mathbf{q}, \mathbf{c}_1)) \;\cdots\; \rho(d^2(\mathbf{q}, \mathbf{c}_l)) \;\; \tilde{\mathbf{q}}^\mathsf{T})$ and $\rho(d) \stackrel{\text{def}}{=} d\log(d)$ is the TPS kernel function for the squared distance. Combining the equations obtained for all the l centres $\mathbf{c}_r$ with target values α_r in a single matrix equation gives:

$$\mathsf{K}_\lambda\mathbf{w} + \tilde{\mathsf{P}}\mathbf{a} = \boldsymbol{\alpha}, \;\; K_{r,k} = \begin{cases} \lambda & r = k \\ \rho(d^2(\mathbf{c}_r, \mathbf{c}_k)) & r \neq k. \end{cases} \tag{2}$$

Adding $\lambda\mathrm{I}$ acts as a regularizer. Solving for $\mathbf{h}_{\boldsymbol{\alpha},\lambda}$ using the above equation and the side-conditions is the classical linear method for estimating the TPS coefficients due to Bookstein [1]. The coefficient vector $\mathbf{h}_{\boldsymbol{\alpha},\lambda}$ is a nonlinear function of the regularization parameter λ and a linear function of the target vector $\boldsymbol{\alpha}$.

Rigid surfaces, fundamental and projection matrices. The rigidity of the observed scene is modeled by the fundamental matrix that we write $\mathcal{F}$ or $\mathcal{A}$ for the perspective and affine camera models respectively. In both cases, the rigidity constraint is $\tilde{\mathbf{q}}'^\mathsf{T}\mathcal{F}\tilde{\mathbf{q}} = 0$. A warp $\mathcal{W}$ is rigid if and only if:

$$\check{\mathcal{W}}(\mathbf{q})^\mathsf{T}\mathcal{F}\tilde{\mathbf{q}} = 0 \qquad \forall \mathbf{q} \in \mathbb{R}^2. \tag{3}$$

Parameterizing the affine fundamental matrix as:

$$\mathcal{A} \stackrel{\text{def}}{\sim} \begin{pmatrix} 0 & 0 & \\ 0 & 0 & \boldsymbol{\imath} \\ & \boldsymbol{\jmath}^\mathsf{T} & \end{pmatrix} \text{ with } \begin{cases} \boldsymbol{\jmath}^\mathsf{T} \stackrel{\text{def}}{=} (c \;\; d \;\; e) \\ \boldsymbol{\imath}^\mathsf{T} \stackrel{\text{def}}{=} (a \;\; b), \end{cases} \tag{4}$$

we rewrite the definition (3) of a rigid affine warp as:

$$\mathcal{W}(\mathbf{q})^{\mathsf{T}}\boldsymbol{\imath} + \tilde{\mathbf{q}}^{\mathsf{T}}\boldsymbol{\jmath} = 0 \qquad \forall \mathbf{q} \in \mathbb{R}^2. \tag{5}$$

The (perspective) fundamental matrix has 7 degrees of freedom and lies on a nontrivial algebraic variety in $\mathbb{R}^9$, written $\mathbb{F}$. The affine fundamental matrix has 4 degrees of freedom and is a point in $\mathbb{P}^4$, see *e.g.* [4, §9.2].

The fundamental matrix is an implicit reconstruction of the two cameras. Canonical cameras are obtained by setting the first (3×4) camera matrix to $(\mathrm{I} \;\; \mathbf{0})\mathcal{M}$ and the second one to $\mathcal{G}_{\mathcal{F}} = ([\check{\mathbf{e}}']_{\times}\mathcal{F} \;\; \check{\mathbf{e}}')\mathcal{M}$, where the second epipole $\mathbf{e}'$ is defined by $\mathcal{F}^{\mathsf{T}}\check{\mathbf{e}}' = \mathbf{0}$. Matrix $\mathcal{M}$ simply swaps the third and fourth coordinates, making affine the first camera, even in the perspective case. Note that $\mathcal{M}\mathcal{M} \sim \mathrm{I}$. In the affine case, we write $\mathcal{S}_{\mathcal{A}}$ the first two rows of $\mathcal{G}_{\mathcal{F}}$, the third row being $(0 \;\; 0 \;\; 0 \;\; 1)$. Within this canonical reconstruction basis, a 3D point with depth δ can be written[2] $\tilde{\mathbf{Q}}^{\mathsf{T}} \sim (\mathbf{q}^{\mathsf{T}} \;\; \delta \;\; 1)$. Reprojecting a 3D point in the second camera $\mathcal{G}_{\mathcal{F}}$ gives the transfer equation $\tilde{\mathbf{q}}' \sim \bar{\mathcal{G}}_{\mathcal{F}}\tilde{\mathbf{q}} + \mathbf{g}_{\mathcal{F}}\delta$ with $\bar{\mathcal{G}}_{\mathcal{F}}$ the first, second and fourth columns of $\mathcal{G}_{\mathcal{F}}$ and $\mathbf{g}_{\mathcal{F}}$ the third one. In the affine case, the second camera matrix $\mathcal{S}_{\mathcal{A}}$ is (2×4). We Define $\bar{\mathcal{S}}_{\mathcal{A}}$ and $\mathbf{s}_{\mathcal{A}}$ similarly to $\bar{\mathcal{G}}_{\mathcal{F}}$ and $\mathbf{g}_{\mathcal{F}}$.

3. Feature-Driven Parameterization of the TPS

We write $\mathbf{h}_{\boldsymbol{\alpha},\lambda} = \mathcal{E}_\lambda\boldsymbol{\alpha}$, *i.e.* as a linear 'back-projection' of the target vector $\boldsymbol{\alpha}$. Matrix $\mathcal{E}_\lambda$ nonlinearly depends on λ. It is given from equation (2) as a function of K_λ and $\tilde{\mathsf{P}}$ by:

$$\mathcal{E}_\lambda \stackrel{\text{def}}{=} \begin{pmatrix} \mathsf{K}_\lambda^{-1}\left(\mathrm{I} - \tilde{\mathsf{P}}\left(\tilde{\mathsf{P}}^{\mathsf{T}}\mathsf{K}_\lambda^{-1}\tilde{\mathsf{P}}\right)^{-1}\tilde{\mathsf{P}}^{\mathsf{T}}\mathsf{K}_\lambda^{-1}\right) \\ \left(\tilde{\mathsf{P}}^{\mathsf{T}}\mathsf{K}_\lambda^{-1}\tilde{\mathsf{P}}\right)^{-1}\tilde{\mathsf{P}}^{\mathsf{T}}\mathsf{K}_\lambda^{-1} \end{pmatrix}.$$

This parameterization has the advantages to separate λ and $\boldsymbol{\alpha}$ and introduces units[3]. The side-conditions are naturally enforced by this parameterization.

Incorporating this parameterization into the TPS (1) we obtain what we call the *feature-driven* parameterization $\tau(\mathbf{q};\boldsymbol{\alpha},\lambda) = \omega(\mathbf{q};\mathbf{h}_{\boldsymbol{\alpha},\lambda})$ for the TPS:

$$\boxed{\tau(\mathbf{q};\boldsymbol{\alpha},\lambda) \stackrel{\text{def}}{=} \boldsymbol{\ell}_{\mathbf{q}}^{\mathsf{T}}\mathcal{E}_\lambda\boldsymbol{\alpha}.} \tag{6}$$

This is a feature-driven parameterization since $\boldsymbol{\alpha}$ contains the coordinates of the target centres. In practice, these are image points. The following important properties hold:

$$\boldsymbol{\ell}_{\mathbf{q}}^{\mathsf{T}}\mathcal{E}_\lambda\mathbf{1} = 1 \text{ and } \tilde{\mathbf{q}}^{\mathsf{T}}\boldsymbol{\theta} = \boldsymbol{\ell}_{\mathbf{q}}^{\mathsf{T}}\mathcal{E}_\lambda\tilde{\mathsf{P}}\boldsymbol{\theta} \quad \forall \mathbf{q} \in \mathbb{R}^2 \quad \forall \boldsymbol{\theta} \in \mathbb{R}^3. \tag{7}$$

[2] δ is actually the inverse of the depth relative to the first camera. If the camera is calibrated this is the 'true' inverse depth, otherwise this is the inverse projective depth. The advantages of this 3D point parameterization are that it is minimal (*i.e.* it has 3 effective parameters) and handles points at infinity.

[3] While $\mathbf{h}_{\boldsymbol{\alpha},\lambda}$ has no obvious unit, $\boldsymbol{\alpha}$ in general has (*e.g.* pixels, meters). This stems from $\mathcal{E}_\lambda\tilde{\mathsf{P}} = (0 \;\; \mathrm{I})^{\mathsf{T}}$. The asymptotic regularization behaviour of the TPS is an affine transformation:

$$\lim_{\lambda\to+\infty} \tau(\mathbf{q};\boldsymbol{\alpha},\lambda) = \boldsymbol{\zeta}^{\mathsf{T}}\tilde{\mathbf{q}}, \;\; \boldsymbol{\zeta} \stackrel{\text{def}}{=} \tilde{\mathsf{P}}^{\dagger}\boldsymbol{\alpha}.$$

4. Warps with the Affine Camera Model

4.1. DA-Warps – Standard TPS-Warps

Derivation and properties. Standard $\mathbb{R}^2 \to \mathbb{R}^2$ TPS-Warps are obtained by stacking two $\mathbb{R}^2 \to \mathbb{R}$ TPS sharing their centres and regularization parameter. Using (6), this gives $(\tau(\mathbf{q};\mathbf{x},\lambda) \;\; \tau(\mathbf{q};\mathbf{y},\lambda)) = \boldsymbol{\ell}_{\mathbf{q}}^{\mathsf{T}}\mathcal{E}_\lambda\begin{pmatrix}\mathbf{x} & \mathbf{y}\end{pmatrix}$. DA-Warps are thus defined as:

$$\boxed{\mathcal{W}_{\text{DA}}(\mathbf{q};\mathsf{P}',\lambda) \stackrel{\text{def}}{=} \mathcal{M}_{\text{DA}}\boldsymbol{\ell}_{\mathbf{q}}, \;\; \mathcal{M}_{\text{DA}}^{\mathsf{T}} \stackrel{\text{def}}{=} \mathcal{E}_\lambda\mathsf{P}'.} \tag{8}$$

We call this a Deformable Affine Thin-Plate Spline Warp, or 'DA-Warp' since it models images of deformable surfaces and corresponds to an affine camera model. Thanks to property (7), we write homogeneous DA-Warps as $\tilde{\mathcal{W}}_{\text{DA}}(\mathbf{q};\mathsf{P}',\lambda) \stackrel{\text{def}}{=} \tilde{\mathcal{M}}_{\text{DA}}\boldsymbol{\ell}_{\mathbf{q}}$ with $\tilde{\mathcal{M}}_{\text{DA}}^{\mathsf{T}} \stackrel{\text{def}}{=} \mathcal{E}_\lambda\tilde{\mathsf{P}}'$.

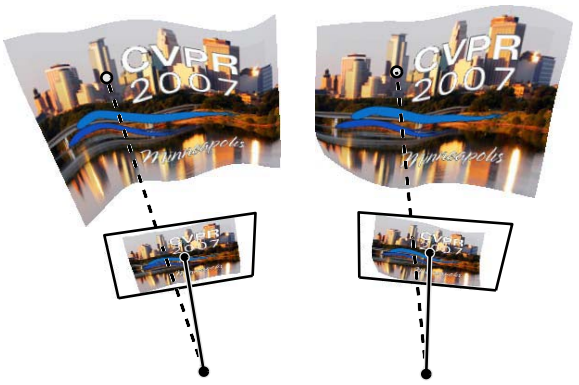


Figure 1. The DA-Warps are interpreted as relating the projection of a deforming 3D surface by two affine cameras.

The set of DA-Warps is written $\mathbb{S}_{\text{DA}}$. They have $2l$ degrees of freedom through $\mathsf{P}' \in \mathbb{R}^{2l}$. The asymptotic regularization behaviour is as follows:

$$\lim_{\lambda\to+\infty} \mathcal{W}_{\text{DA}}(\mathbf{q};\mathsf{P}',\lambda) = \mathcal{L}_{\text{DA}}\tilde{\mathbf{q}}', \;\; \mathcal{L}_{\text{DA}}^{\mathsf{T}} \stackrel{\text{def}}{=} \tilde{\mathsf{P}}^{\dagger}\mathsf{P}'.$$

In other words, a DA-Warp tends to a flat affine warp represented by the (2×3) matrix $\mathcal{L}_{\text{DA}}$. It can be shown that $\mathcal{L}_{\text{DA}}$ minimizes the transfer error[4].

[4] The transfer error is the discrepancy between the data points in the second image and the points transferred by the warp from the first image, see §7 for more details.

A projected deformable surface interpretation. We propose a geometrical interpretation of the DA-Warps as the warps induced by the observation of a deforming surface with two affine cameras, as illustrated in figure 1. In order to model the surface depth, its motion and deformation, we introduce an $\mathbb{R}^2 \rightarrow \mathbb{R}^3$ map, parameterizing the surface in a concise manner by the 3D coordinates $\mathbf{C}'_k$ of the l centres. This map is built by stacking three TPS sharing their centres and regularization parameter. More formally, gathering the '3D centres' $\mathbf{C}'_k$ in a single $(l \times 3)$ matrix $\mathsf{Z}^\mathsf{T} = (\mathbf{C}'_1 \ \cdots \ \mathbf{C}'_l)$, the map is written:

$$\mathcal{R}(\mathbf{q}; \mathsf{Z}, \lambda) \stackrel{\text{def}}{=} \mathcal{M}_{3\text{D}} \boldsymbol{\ell}_\mathbf{q}, \quad \mathcal{M}_{3\text{D}}^\mathsf{T} \stackrel{\text{def}}{=} \mathcal{E}_\lambda \mathsf{Z}. \tag{9}$$

Reprojecting a 3D surface point gives $\mathbf{q}' = \bar{\mathcal{S}}_\mathcal{A} \mathsf{Z}^\mathsf{T} \mathcal{E}_\lambda^\mathsf{T} \boldsymbol{\ell}_\mathbf{q} + \mathbf{s}_\mathcal{A}$. Using property (7), we get $\mathbf{q}' = \mathcal{S}_\mathcal{A} \tilde{\mathsf{Z}}^\mathsf{T} \mathcal{E}_\lambda^\mathsf{T} \boldsymbol{\ell}_\mathbf{q}$, that we identify with a DA-Warp (8), giving:

$$\mathbf{q}' = \mathcal{W}_{\text{DA}}(\mathbf{q}; \tilde{\mathsf{Z}} \mathcal{S}_\mathcal{A}^\mathsf{T}, \lambda).$$

This shows that the 3D centres $\mathbf{C}'_k$ can be replaced by the 2D centres $\mathbf{c}'_k$ in the second image since $\mathsf{P}' = \tilde{\mathsf{Z}} \mathcal{S}_\mathcal{A}^\mathsf{T}$.

This geometric interpretation does not only provide a strong intuition on the fact that the DA-Warps are intrinsically affine, but also a setting for naturally deriving the DP-Warps, their perspective projection extension, in §5.2.

4.2. RA-Warps – Rigid Affine Warps

Derivation and properties. Applying the rigid affine warp definition (5) to a DA-Warp (8) gives:

$$\boldsymbol{\ell}_\mathbf{q}^\mathsf{T} \mathcal{E}_\lambda \mathsf{P}' \boldsymbol{\imath} + \tilde{\mathbf{q}}^\mathsf{T} \boldsymbol{\jmath} = 0 \qquad \forall \mathbf{q} \in \mathbb{R}^2.$$

Using property (7) gives $\boldsymbol{\ell}_\mathbf{q}^\mathsf{T} \mathcal{E}_\lambda \left(\mathsf{P}' \boldsymbol{\imath} + \tilde{\mathsf{P}} \boldsymbol{\jmath} \right) = 0, \forall \mathbf{q} \in \mathbb{R}^2$. This implies $\mathsf{P}' \boldsymbol{\imath} + \tilde{\mathsf{P}} \boldsymbol{\jmath} = \mathbf{0}_{(l \times 1)}$, which is the epipolar constraint for all pairs of centres. We call it the rigidity consistency constraint for DA-Warps. This means that if the warp satifies the epipolar geometry, then the centres also have to.

Each pair of centres $\mathbf{c}_k \leftrightarrow \mathbf{c}'_k$ satisfies the epipolar constraint and thus is the projection of a 3D point $\mathbf{C}_k^\mathsf{T} = (\mathbf{c}_k^\mathsf{T} \ \ \delta_k)$ in the canonical basis. Reprojecting all centres in the second image gives $\mathsf{P}' = (\mathsf{P} \ \boldsymbol{\delta} \ \mathbf{1}) \mathcal{S}_\mathcal{A}^\mathsf{T}$. Substituting into the DA-Warp formulation (8) gives:

$$\mathcal{W}_{\text{DA}}(\mathbf{q}; \mathsf{P}', \lambda) = \mathcal{S}_\mathcal{A} (\mathsf{P} \ \boldsymbol{\delta} \ \mathbf{1})^\mathsf{T} \mathcal{E}_\lambda^\mathsf{T} \boldsymbol{\ell}_\mathbf{q},$$

which can be seen as the projection of some 3D point by the second camera, thereby satisfying the rigidity constraint, completing the proof. We thus define:

$$\boxed{\mathcal{W}_{\text{RA}}(\mathbf{q}; \boldsymbol{\delta}, \mathcal{A}, \lambda) \stackrel{\text{def}}{=} \mathcal{M}_{\text{RA}} \boldsymbol{\ell}_\mathbf{q}, \quad \mathcal{M}_{\text{RA}}^\mathsf{T} \stackrel{\text{def}}{=} \mathcal{E}_\lambda (\mathsf{P} \ \boldsymbol{\delta} \ \mathbf{1}) \mathcal{S}_\mathcal{A}^\mathsf{T}.} \tag{10}$$

This definition of RA-Warps can be made homogeneous by replacing the (2×4) camera $\mathcal{S}_\mathcal{A}$ by its (3×4) equivalent $\mathcal{G}_\mathcal{A}$ in the above equations, giving $\tilde{\mathcal{W}}_{\text{RA}}(\mathbf{q}; \boldsymbol{\delta}, \mathcal{A}, \lambda) \stackrel{\text{def}}{=} \tilde{\mathcal{M}}_{\text{RA}} \boldsymbol{\ell}_\mathbf{q}$ with $\tilde{\mathcal{M}}_{\text{RA}}^\mathsf{T} \stackrel{\text{def}}{=} \mathcal{E}_\lambda (\mathsf{P} \ \boldsymbol{\delta} \ \mathbf{1}) \mathcal{G}_\mathcal{A}^\mathsf{T}$.

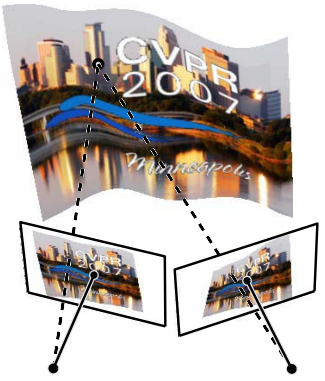


Figure 2. The RA-Warps are interpreted as relating the projection of a rigid smooth 3D surface by two affine cameras.

From the above derivation follows that the set of RA-Warps, denoted $\mathbb{S}_{\text{RA}}$, is a subset of $\mathbb{S}_{\text{DA}}$. In can be shown that the set of flat affine warps $\mathbb{S}_{\text{FA}}$ is included into $\mathbb{S}_{\text{RA}}$. The RA-Warps have $l + 4$ degrees of freedom through $(\boldsymbol{\delta}, \mathcal{A}) \in \mathbb{R}^l \times \mathbb{P}^4$. Parameters $\boldsymbol{\delta}$ are the depth of the centres with respect to the first camera.

The asymptotic regularization behaviour of the RA-Warps is derived directly from the one for the DA-Warps (8):

$$\lim_{\lambda \rightarrow +\infty} \mathcal{W}_{\text{RA}}(\mathbf{q}; \boldsymbol{\delta}, \mathcal{A}, \lambda) = \mathcal{L}_{\text{RA}} \tilde{\mathbf{q}}, \quad \mathcal{L}_{\text{RA}}^\mathsf{T} \stackrel{\text{def}}{=} \tilde{\mathsf{P}}^\dagger (\mathsf{P} \ \boldsymbol{\delta} \ \mathbf{1}) \mathcal{S}_\mathcal{A}^\mathsf{T}.$$

In other words, an RA-Warp tends to a flat affine warp represented by the (2×3) matrix $\mathcal{L}_{\text{RA}}$. It can be shown to be rigid and can be written $\mathcal{L}_{\text{RA}} = \mathcal{S}_\mathcal{A} + \mathbf{s}_\mathcal{A} \boldsymbol{\delta}^\mathsf{T} \tilde{\mathsf{P}}^\dagger$, a plane-induced affine warp minimizing the transfer error, under the assumption that the point correspondences satisfy the rigidity constraint.

A projected rigid surface interpretation. A geometric interpretation of the RA-Warps, illustrated in figure 2, directly stems from their definition (10). The RA-Warps are induced by a surface defined as a Monge patch parameterized by a TPS mapping points from the first image to their depths. This $\mathbb{R}^2 \rightarrow \mathbb{R}$ TPS is of the form (6) and has the same centres as the RA-Warp. This is derived by expanding the formulation (10) of the RA-Warps and property (7):

$$\mathcal{W}_{\text{RA}}(\mathbf{q}; \boldsymbol{\delta}, \mathcal{A}, \lambda) = \bar{\mathcal{S}}_\mathcal{A} \tilde{\mathbf{q}} + \mathbf{s}_\mathcal{A} \tau(\mathbf{q}; \boldsymbol{\delta}, \lambda). \tag{11}$$

5. Warps with the Perspective Camera Model

5.1. RP-Warps – Rigid Perspective Warps

We derive the RP-Warps by introducing perspective projection in the RA-Warps. Following §4.2, we pick up a 3D point $\mathbf{Q}$ on the scene surface, defined by an $\mathbb{R}^2 \rightarrow \mathbb{R}$ TPS parameterized Monge patch, and reproject it in the second image, giving from equation (11):

$$\check{\mathcal{W}}_{\text{RP}}(\mathbf{q}; \boldsymbol{\delta}, \mathcal{F}, \lambda) \sim \bar{\mathcal{G}}_\mathcal{F} \tilde{\mathbf{q}} + \mathbf{g}_\mathcal{F} \tau(\mathbf{q}; \boldsymbol{\delta}, \lambda).$$

Replacing τ by its expression (6), and applying property (7) to each of the three rows of $\bar{\mathcal{G}}_\mathcal{F}$, we get $\check{\mathcal{W}}_{\text{RP}}(\mathbf{q}; \boldsymbol{\delta}, \mathcal{F}, \lambda) \sim$

$(\bar{\mathcal{G}}_{\mathcal{F}}\tilde{\mathsf{P}}^{\mathsf{T}} + \mathbf{g}_{\mathcal{F}}\boldsymbol{\delta}^{\mathsf{T}})\mathcal{E}_{\lambda}^{\mathsf{T}}\boldsymbol{\ell}_{\mathbf{q}}$ and thus:

$$\boxed{\check{\mathcal{W}}_{\text{RP}}(\mathbf{q};\boldsymbol{\delta},\mathcal{F},\lambda) \overset{\text{def}}{\sim} \check{\mathcal{M}}_{\text{RP}}\boldsymbol{\ell}_{\mathbf{q}}, \quad \check{\mathcal{M}}_{\text{RP}}^{\mathsf{T}} \overset{\text{def}}{\sim} \mathcal{E}_{\lambda}(\mathsf{P}\ \ \boldsymbol{\delta}\ \ \mathbf{1})\mathcal{G}_{\mathcal{F}}^{\mathsf{T}}.} \quad (12)$$

This is the homogeneous Rigid Perspective Thin-Plate Spline Warp. The homogeneous coordinates of the transferred point are linear functions of $\boldsymbol{\ell}_{\mathbf{q}}$. The affine coordinates are obtained as ratios of linear functions through $\mathcal{W}_{\text{RP}}(\mathbf{q};\boldsymbol{\delta},\mathcal{F},\lambda) \overset{\text{def}}{=} \psi(\check{\mathcal{W}}_{\text{RP}}(\mathbf{q};\boldsymbol{\delta},\mathcal{F},\lambda))$.

The set of RP-Warps, denoted $\mathbb{S}_{\text{RP}}$ is a superset of $\mathbb{S}_{\text{RA}}$. This is shown easily by choosing for $\mathcal{F}$ an affine fundamental matrix. It can be shown that $\mathbb{S}_{\text{RP}}$ is also a superset of $\mathbb{S}_{\text{FP}}$. An RP-Warp is guaranteed to be rigid since it implicitly projects 3D points, giving image points satisfying the epipolar geometry constraint. It has $l+7$ degrees of freedom through $(\boldsymbol{\delta},\mathcal{F}) \in \mathbb{R}^l \times \mathbb{F}$.

The asymptotic regularization behaviour is:

$$\lim_{\lambda\to+\infty} \check{\mathcal{W}}_{\text{RP}}(\mathbf{q};\boldsymbol{\delta},\mathcal{F},\lambda) \sim \check{\mathcal{L}}_{\text{RP}}\tilde{\mathbf{q}}, \quad \check{\mathcal{L}}_{\text{RP}}^{\mathsf{T}} \overset{\text{def}}{\sim} \tilde{\mathsf{P}}^{\dagger}(\mathsf{P}\ \ \boldsymbol{\delta}\ \ \mathbf{1})\mathcal{G}_{\mathcal{F}}^{\mathsf{T}}.$$

An RP-Warp thus tends to a flat perspective warp represented by the (3×3) homogeneous homography matrix $\check{\mathcal{L}}_{\text{RP}}$. It can be shown to be the plane-induced rigid warp $\check{\mathcal{L}}_{\text{RP}} \sim \mathcal{G}_{\mathcal{F}} + \mathbf{g}_{\mathcal{F}}\boldsymbol{\delta}^{\mathsf{T}}\tilde{\mathsf{P}}^{\dagger}$ minimizing an algebraic transfer error, under the assumption that the point correspondences satisfy the rigidity constraint.

5.2. DP-Warps – Deformable Perspective Warps

The DP-Warps form a superset of all other warps derived so far in this paper, including the standard DA-Warps. The RP-Warps are derived by introducing perspective projection in the RA-Warps. We derive the DP-Warps from the DA-Warps in the same spirit. Consider the deformable surface geometric interpretation shown in figure 1. The surface seen by the second camera is defined by an $\mathbb{R}^2 \to \mathbb{R}^3$ map $\mathcal{R}(\mathbf{q};\mathsf{Z},\lambda)$. Projecting a point on this surface to the second image gives $\check{\mathbf{q}}' \sim \bar{\mathcal{G}}_{\mathcal{F}}\mathcal{R}(\mathbf{q};\mathsf{Z},\lambda) + \mathbf{g}_{\mathcal{F}}$. Substituting the map (9) defining the 3D surface gives:

$$\check{\mathbf{q}}' \sim \bar{\mathcal{G}}_{\mathcal{F}}\mathsf{Z}^{\mathsf{T}}\mathcal{E}_{\lambda}^{\mathsf{T}}\boldsymbol{\ell}_{\mathbf{q}} + \mathbf{g}_{\mathcal{F}}.$$

Using property (7), we get $\check{\mathbf{q}}' \sim \bar{\mathcal{G}}_{\mathcal{F}}\check{\mathsf{Z}}^{\mathsf{T}}\mathcal{E}_{\lambda}^{\mathsf{T}}\boldsymbol{\ell}_{\mathbf{q}}$. The centres in the second image are the reprojection of the '3D centres' in matrix Z. The weights of the homogeneous coordinates in $\check{\mathsf{P}}'$ are important: they model the perspective part of the DP-Warps. We thus define the DP-Warps as:

$$\boxed{\check{\mathcal{W}}_{\text{DP}}(\mathbf{q};\check{\mathsf{P}}',\lambda) \overset{\text{def}}{\sim} \check{\mathcal{M}}_{\text{DP}}\boldsymbol{\ell}_{\mathbf{q}}, \quad \check{\mathcal{M}}_{\text{DP}}^{\mathsf{T}} \overset{\text{def}}{\sim} \mathcal{E}_{\lambda}\check{\mathsf{P}}'.} \quad (13)$$

The affine coordinates are obtained as ratios of linear functions through $\mathcal{W}_{\text{DP}}(\mathbf{q};\check{\mathsf{P}}',\lambda) \overset{\text{def}}{=} \psi(\check{\mathcal{W}}_{\text{DP}}(\mathbf{q};\check{\mathsf{P}}',\lambda))$.

The set of DP-Warps, denoted $\mathbb{S}_{\text{DP}}$, forms a superset of $\mathbb{S}_{\text{RP}}$ and a superset of $\mathbb{S}_{\text{DA}}$. The DP-Warps have parameters $\check{\mathsf{P}}'$ and thus $3l-1$ degrees of freedom. Consequently, they can not be estimated by choosing as centres all data points: each point correspondence giving two contraints, we end up with $2l$ contraints, which is less than the $3l-1$ unknowns. Methods for estimating DP-Warps are reported in §7.

The asymptotic regularization behaviour is formulated below for all data points chosen as centres. A consequence is that the limiting warp we get is undetermined, *i.e.* has some free parameters. Unsurprisingly, it actually has $(3l-1)-2l = l-1$ free parameters, *i.e.* the difference between the number of free parameters of the DP-Warps and the number of constraints given by interpolating the l centres:

$$\lim_{\lambda\to+\infty} \check{\mathcal{W}}_{\text{DP}}(\mathbf{q};\check{\mathsf{P}}',\lambda) \sim \check{\mathcal{L}}_{\text{DP}}\tilde{\mathbf{q}}, \quad \check{\mathcal{L}}_{\text{DP}}^{\mathsf{T}} \overset{\text{def}}{\sim} \tilde{\mathsf{P}}^{\dagger}\text{diag}(\mathbf{d})\tilde{\mathsf{P}}'.$$

The $(l\times 1)$ vector $\mathbf{d}$, defined up to scale, represents the $l-1$ free parameters of the limiting flat perspective warp, represented by matrix $\check{\mathcal{L}}_{\text{DP}}$, minimizing an algebraic transfer error, different from the one we use in §7.

6. A Hierarchy of Warps

The aim of this section is to define a hierarchy between the sets of standard DA-Warps $\mathbb{S}_{\text{DA}}$, of flat affine and perspective warps $\mathbb{S}_{\text{FA}}$ and $\mathbb{S}_{\text{FP}}$, and of the three types of warps we introduced, $\mathbb{S}_{\text{RA}}$, $\mathbb{S}_{\text{RP}}$ and $\mathbb{S}_{\text{DP}}$. The whole hierarchy is illustrated in figure 3. So far, we have established $\mathbb{S}_{\text{RA}} \subset \mathbb{S}_{\text{DA}}$ and $\mathbb{S}_{\text{RA}} \subset \mathbb{S}_{\text{RP}}$. Intuitively, the common warps to $\mathbb{S}_{\text{DA}}$ and $\mathbb{S}_{\text{RP}}$ must be rigid and affine. More precisely, we have $\mathbb{S}_{\text{RA}} = \mathbb{S}_{\text{DA}} \cap \mathbb{S}_{\text{RP}}$. We also established that $\mathbb{S}_{\text{DP}}$ is a superset of all the other warps, *i.e.* $\mathbb{S}_{\text{RP}} \subset \mathbb{S}_{\text{DP}}$ and $\mathbb{S}_{\text{DA}} \subset \mathbb{S}_{\text{DP}}$, and thus $\mathbb{S}_{\text{RA}} \subset \mathbb{S}_{\text{DP}}$. The set of DA-Warps $\mathbb{S}_{\text{DA}}$ does not contain any flat perspective warp. More formally, $(\mathbb{S}_{\text{FP}} - \mathbb{S}_{\text{FA}}) \cap \mathbb{S}_{\text{DA}} = \emptyset$, implying $(\mathbb{S}_{\text{FP}} - \mathbb{S}_{\text{FA}}) \cap \mathbb{S}_{\text{RA}} = \emptyset$.

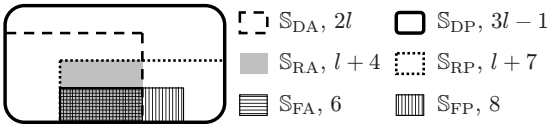


Figure 3. Hierarchical representation for the three proposed types of warps – RA, RP and DP – along with the standard TPS warps, dubbed DA-Warps, and the flat warps FA-Warps and FP-Warps. D stands for Deformable, R for rigid, F for flat, A for Affine and P for Perspective. The number of degrees of freedom for l centres is indicated for each set of warps.

7. Estimation of the Warps

We propose warp estimation methods from m point correspondences $\mathbf{q}_j \leftrightarrow \mathbf{q}'_j$. We examine two cases for the centres in the first image. *(i): 'centre-on-data'* – the centres in the first image coincide with the data points, *i.e.* $m = l$.

(ii): 'arbitrary-centres' – the centres in the first image may not coincide with the data points. They are typically chosen on a regular grid or as interest points. We assume $m \geq l$, *i.e.* we have sufficiently many point correspondences to estimate the warp without having to regularize it.

All warps are estimated by minimizing the transfer error, *i.e.* the discrepancy, measured by the Euclidean distance, between the data points in the second image, and the corresponding points transferred by the sought after warp from the first image. Criteria based on a 3D depth error for the rigid warps are avoided since they are not physically meaningful for uncalibrated cameras. For most warps, the transfer error is nonlinear. Two-step methods and algebraic approximations are used to get an initial estimate through Linear Least Squares minimization (LLS), solved using the pseudo-inverse technique or Singular Value Decomposition (SVD) of the design matrix, if the system is homogeneous, enforcing unit two-norm on the unknown vector. The initial estimate is refined by iteratively minimizing the transfer error through Nonlinear Least Squares minimization (NLS) with the Levenberg-Marquardt algorithm, see *e.g.* [4, §A]. For a warp $\mathcal{W}$ with parameters $\mathcal{U}$, the minimization problem is generically written:

$$\vartheta(\mathcal{W},\mathcal{U}) \stackrel{\text{def}}{=} \min_{\mathcal{U}} \sum_{j=1}^{n} \|\mathcal{W}(\mathbf{q}_j;\mathcal{U},\lambda) - \mathbf{q}'_j\|^2.$$

DA-Warps. The minimal number of point correspondences is $m \geq 3$. In the centre-on-data case, this is the classical problem solved by Bookstein [1]. With our feature-driven parameterization (8), the transformation is readily expressed in terms of the centre coordinates P'. Note that the data points are interpolated, which nullifies the transfer error. In the arbitrary-centre case, we solve $\vartheta(\mathcal{W}_{\text{DA}}, \mathsf{P}')$. Writing $\boldsymbol{\ell}_{\mathbf{q}_j}$ as $\boldsymbol{\ell}_j$, and replacing $\mathcal{W}_{\text{DA}}$ by its expression (8), we get an LLS problem:

$$\min_{\mathsf{P}'} \sum_{j=1}^{m} \left\| \boldsymbol{\ell}_j^\mathsf{T} \mathcal{E}_\lambda \mathsf{P}' - \mathbf{q}_j'^\mathsf{T} \right\|^2.$$

RA-Warps. A single algorithm solves both the centre-on-data and arbitrary-centre cases for $m \geq 4$ point correspondences. RA-Warps depend on the affine fundamental matrix $\mathcal{A}$ and a depth vector $\boldsymbol{\delta}$. Contrarily to the DA-Warp case, $\vartheta(\mathcal{W}_{\text{RA}}, \{\boldsymbol{\delta}, \mathcal{A}\})$ is an NLS problem due to the coupling between $\boldsymbol{\delta}$ and $\mathcal{A}$ in the expression (10) of the warp. In order to find an initial estimate, we use a two-step procedure. We compute $\mathcal{A}$ using *e.g.* the Gold Standard algorithm in [4, §14.3]. Given $\mathcal{A}$, finding $\boldsymbol{\delta}$ by minimizing the transfer error, *i.e.* solving $\vartheta(\mathcal{W}_{\text{RA}}, \boldsymbol{\delta})$ turns out to be an LLS problem:

$$\min_{\boldsymbol{\delta}} \sum_{j=1}^{m} \left\| \mathbf{s}_\mathcal{A} \boldsymbol{\ell}_j^\mathsf{T} \mathcal{E}_\lambda^\mathsf{T} \boldsymbol{\delta} + \boldsymbol{\ell}_j^\mathsf{T} \mathcal{E}_\lambda \tilde{\mathsf{P}} \mathcal{S}_\mathcal{A}^\mathsf{T} - \mathbf{q}_j'^\mathsf{T} \right\|^2.$$

RP-Warps. For both the centre-on-data and the arbitrary-centre cases, the minimal number of point correspondences is $m \geq 7$. RP-Warps depend on the fundamental matrix $\mathcal{F}$ and a depth vector $\boldsymbol{\delta}$. $\vartheta(\mathcal{W}_{\text{RP}}, \{\boldsymbol{\delta}, \mathcal{F}\})$ is an NLS problem, for several reasons: (i) $\boldsymbol{\delta}$ and $\mathcal{F}$ are coupled in the homogeneous expression (12) of the warp, (ii) finding the affine coordinates of the transferred point requires a division and (iii) the fundamental matrix must fulfill a nonlinear rank-deficiency constraint[5]. Similarly to the algorithm for the RA-Warps, we use a two-step initialization procedure. We compute $\mathcal{F}$ using *e.g.* the Gold Standard algorithm in [4, §11.4]. Given $\mathcal{F}$, we estimate $\boldsymbol{\delta}$ by minimizing an algebraic approximation to the transfer error:

$$\min_{\boldsymbol{\delta}} \sum_{j=1}^{m} d_a^2(\check{\mathcal{W}}_{\text{RP}}(\mathbf{q}_j;\boldsymbol{\delta},\mathcal{F},\lambda), \tilde{\mathbf{q}}'_j),$$

with $d_a^2(\check{\mathbf{q}}, \check{\mathbf{q}}') = \|\mathsf{S}[\check{\mathbf{q}}]_\times \check{\mathbf{q}}'\|^2$ an algebraic distance between points $\mathbf{q}$ and $\mathbf{q}'$, and $\mathsf{S} = (\mathrm{I}\ \mathbf{0})$ simply selects the first two rows of the cross-product. The algebraic approximation yields an LLS minimization problem since the algebraic distance directly compares homogeneous coordinate vectors, thereby avoiding the need for the perspective division. Normalizing the image coordinates is crucial to make the algebraic distance 'similar to' the Euclidean one [4]. Substituting d_a by its expression, and the RP-Warp by its homogeneous formulation (12), we get $\min_{\boldsymbol{\delta}} \sum_{j=1}^{m} \|\mathsf{S}[\tilde{\mathbf{q}}'_j]_\times \mathcal{G}_\mathcal{F} (\mathsf{P}\ \boldsymbol{\delta}\ \mathbf{1})^\mathsf{T} \mathcal{E}_\lambda^\mathsf{T} \boldsymbol{\ell}_j\|^2$ and as sought, after minor algebraic manipulations, an LLS problem:

$$\min_{\boldsymbol{\delta}} \sum_{j=1}^{m} \left\| \mathsf{S}[\tilde{\mathbf{q}}'_j]_\times \mathbf{g}_\mathcal{F} \boldsymbol{\ell}_j^\mathsf{T} \mathcal{E}_\lambda \boldsymbol{\delta} + \mathsf{S}[\tilde{\mathbf{q}}'_j]_\times \bar{\mathcal{G}}_\mathcal{F} \tilde{\mathsf{P}}^\mathsf{T} \mathcal{E}_\lambda^\mathsf{T} \boldsymbol{\ell}_j \right\|^2.$$

DP-Warps. The minimum number of data points is $m \geq 4$. In the *arbitrary-centre* case, $\vartheta(\mathcal{W}_{\text{DP}}, \check{\mathsf{P}}')$ is an NLS problem, due to the division required for finding the affine coordinates of the transferred point. An initial estimate is found, as for the RP-Warps, by minimizing an algebraic approximation to the transfer error, *i.e.* $\min_{\check{\mathsf{P}}'} \sum_{j=1}^{m} d_a^2(\check{\mathcal{W}}_{\text{DP}}(\mathbf{q}_j;\check{\mathsf{P}}',\lambda), \tilde{\mathbf{q}}'_j)$. The arbitrary scale of $\check{\mathsf{P}}'$ is fixed by enforcing its norm to unity, *i.e.* $\|\check{\mathsf{P}}'\| = 1$. Replacing d_a by its expression, and $\check{\mathcal{W}}_{\text{DP}}$ by its homogeneous expression (13), we obtain, after some minor algebraic manipulations, an LLS problem:

$$\min_{\check{\mathsf{P}}', \|\check{\mathsf{P}}'\|=1} \sum_{j=1}^{m} \left\| \mathsf{S}[\tilde{\mathbf{q}}'_j]_\times \mathrm{diag}_3(\boldsymbol{\ell}_j^\mathsf{T} \mathcal{E}_\lambda) \mathrm{vect}(\check{\mathsf{P}}') \right\|^2,$$

with $\mathrm{diag}_r(\mathbf{x})$ an r block diagonal matrix with $\mathbf{x}$ the repeated block, and with vect the row-wise matrix vectorization.

[5] In practice, we directly optimize over the 12 entries of the second projection matrix to avoid parameterizing the nontrivial variety of fundamental matrices.

The simple centre-on-data setting in not possible for this type of warps. Indeed, as already discussed, the $2l$ constraints provided in general by l centre correspondences are not enough to constrain the $3l-1$ degrees of freedom of the warp. In other words, there is no solution to estimate the warp parameters by taking as centres the whole set of point correspondences: $l-1$ other point correspondences are needed. We thus consider a *weak centre-on-data* case: we pick a subset of l out of the m data points as centres, with $l \leq \lfloor \frac{2m+1}{3} \rfloor$. In case where $3l = 2m+1$ holds, there is a unique solution, which obviously depends on which data points are chosen as centres. The general minimal case is $m = 3k+1$ and $l = 2k+1$ with $k \in \mathbb{R}^{+*}$. Both the minimal and redundent cases are solved by writing an algebraic approximation to the transfer error. Given the centre coordinates P and P' in both images, we are looking for the parameters $\tilde{\mathsf{P}}'$ of the DP-Warp. A simple way of parameterizing the problem is to use $\tilde{\mathsf{P}}' = \mathrm{diag}(\mathbf{d})\tilde{\mathsf{P}}'$, and compute the 'scale vector' $\mathbf{d}$ only. This enforces interpolation of the centres and leaves only the remaining $l-1$ unknowns since $\mathbf{d}$ is an l-vector defined up to scale, leading to $\min_{\mathbf{d}} \sum_{j=1}^{m} d_a^2(\tilde{\mathcal{W}}_{\mathrm{DP}}(\mathbf{q}_j; \mathrm{diag}(\mathbf{d})\tilde{\mathsf{P}}', \lambda), \tilde{\mathbf{q}}'_j)$ such that $\|\mathbf{d}\| = 1$. Substituting d_a by its expression, as well as the DP-Warp from equation (13), gives an LLS problem:

$$\min_{\mathbf{d}, \|\mathbf{d}\|=1} \sum_{j=1}^{m} \left\| \mathsf{S}[\tilde{\mathbf{q}}'_j]_\times \tilde{\mathsf{P}}'^{\mathsf{T}} \mathrm{diag}(\mathcal{E}_\lambda^{\mathsf{T}} \boldsymbol{\ell}_j)\mathbf{d} \right\|^2.$$

8. Experimental Evaluation

8.1. Simulated Data

We synthetically generated training and test data sets by projecting 3D points into two cameras with focal length f. The 3D points lie on paper-like 3D surfaces, *i.e.* with vanishing Gaussian curvature. We believe that it is representative of the kind of real images one may use the proposed warps on. The 50, Gaussian noise corrupted training data points are used as centres to estimate the warps. The errors reported below are means over 500 trials of the transfer error estimated over the 200 points of the test set, so as to reflect the quality of the estimated warps.

Figure 4 (left) shows the results we obtained. We must not compare the affine and perspective warps with these results, since in order to assess the influence of noise only, we use affine cameras for the affine warps, and perspective cameras with $f = 300$ pixels for the perspective warps. We observe that the quality of the warps linearly degrades with the noise level, and is actually around twice the noise standard deviation. This is a satisfying behaviour, holding true for all warps.

We gradually increase the focal length of the cameras, which has the effect of making the images more affine and keep the scene rigid. In order to preserve the transfer error scale, we keep invariant the size of the imaged object by translating each camera along its optical axis. Figure 4 (right) shows the result, with a 2 pixel noise level. As expected, the transfer error for the RP-Warps appears to be invariant to the focal length (the slight variations are due to the image scale). We make the same observation for the DA-Warps. This is due to the fact that the DA-Warps have extra degree of freedom compared to the rigid warps approximating the perspective projection of the rigid surface. The RA-Warps have the highest error for short focal lengths, which dramatically decreases as the focal length increases. It eventually converges to the same error as for the RP-Warps when the affine camera model becomes numerically equivalent to the perspective one. We see that beyond $f_{\mathrm{DA}} = 2000$ pixels, the RA-Warps do better than the DA-Warps. This is the breakdown focal length for the DA-Warps, above which the lack of rigidity, and the fact that the affine approximation is better fulfilled, makes the RA-Warps better capture the underlying true warp. We note that 2000 pixels is the order of magnitude one may have with real images. We experimentally measured $f_{\mathrm{DA}} \approx \{\infty, 4000, 2000, 500, 300\}$ pixels for $\{0, 1, 2, 5, 10\}$ pixel noise levels. This shows that enforcing the rigidity constraint is very important to capture a warp as close as possible to the true one from limited image measurements. Another conclusion is due to the significant difference between the RA-Warps and the RP-Warps. The latter achieves consistently lower test transfer errors, much lower for short to medium focal lengths. This shows that much more accurate warps are captured by modeling perspective projection. Experimental results with a deforming surface, not shown here, allow us to draw similar conclusions for the DA-Warps and the DP-Warps. It also shows that the DP-Warps overfit the data and usually have large variance. Another experiment shows that gradually merging the test and the training sets decreases the error to zero for the deformable warps and to the epipolar geometry fitting error for the rigid warps, as expected.

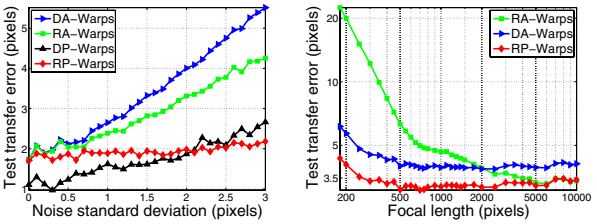


Figure 4. Influence of the noise level (left) and the focal length (right) onto the quality of the warps.

8.2. Real Data

We took a set of images of a manually handled poster with short and long focal lengths and various deformations.

Figure 5. One of the perspective images (left) and closeup on transferred points in another image for the deformable warps (right).

Figure 6. Negative difference image for an RA-Warp (left) and an RP-Warp (right). Bright colors indicate low discrepancy. The RMS errors are respectively 33.66 and 19.01.

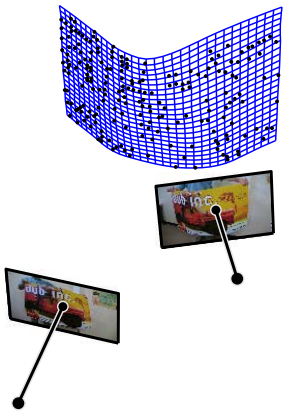


Figure 7. The cameras and 3D surface reconstructed through an RP-Warp.

We then manually clicked 206 point correspondences into all the images and estimated the warps. One of the experiments consisted in comparing the RA-Warps and the RP-Warps in the presence of significant perspective projection effects, see one of the images, overlaid with the point correspondences, on figure 5 (left) but *without surface deformations*. We selected 52 points to serve as centres (approximately 25% of the data points), and minimized the transfer error over all data points to fit the warps. The transfer error we obtained is 19.01 pixels and 12.53 pixels for the RA-Warp and the RP-Warp respectively. We used the two computed warps to warp the second image onto the first one, and computed their difference, shown on figure 6. These ideally are black, zero value images. Non-zero values are to be interpreted as unmodeled physical phenomena, mainly the extent to which the warp models the image deformation. The difference image clearly reflects the quality of the warp. We see that the RP-Warps do much better than the RA-Warps. The mean color alignment error in pixel value units is 33.66 and 19.01 for the RA-Warp and the RP-Warp respectively. We observed in particular that there is no data points in the top-right hand corner of the poster. This is were the difference is the highest for the RA-Warp, showing that the deformation, and thus the surface, is not very well captured by this model. The 3D surface reconstructed by the RP-Warp is shown in figure 7.

Similar comparison results for the DA-Warps and the DP-Warps were obtained by using two images of the poster with different deformations, giving a transfer error of 6.66 and 5.83 pixels respectively. Figure 5 (right) shows a representative closeup on transferred points. We observe that the data points are better predicted by the DP-Warp, meaning that it effectively models perspective projection but has however a high variance, *i.e.* is very dependent on which data points are used.

9. Discussion

Three types of $\mathbb{R}^2 \rightarrow \mathbb{R}^2$ image warps are proposed, using the $\mathbb{R}^2 \rightarrow \mathbb{R}$ Thin-Plate Spline as a building block. They are designed to overcome some limitations of the standard Thin-Plate Spline warps, and derived based on a feature-driven parameterization we introduce. These warps have direct practical impacts since they better model image deformations than the standard DA-Warps in several cases, *e.g.* for rigid smooth surfaces and images with perspective projection effects. The DP-Warps are unstable because they overfit the data and tend to have high variance since they actually depend on the depth of the centres. One remedy may be to regularize their denominator. The warps essentially use two view visual geometry and the Thin-Plate Spline. Since they are given geometric interpretations, they can probably be extended to multiple views.

References

[1] F. L. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. *PAMI*, 1989. 1, 2, 6

[2] H. Chui and A. Rangarajan. A new point matching algorithm for non-rigid registration. *CVIU*, 2003. 2

[3] G. Donato and S. Belongie. Approximate thin-plate spline mappings. *ECCV*, 2002. 2

[4] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003. 3, 6

[5] L. Masson, M. Dhome, and F. Jurie. Tracking 3D objects using flexible models. *BMVC*, 2005. 1

[6] T. W. Sederberg and S. R. Parry. Free-form deformation of solid geometric models. SIGGRAPH, 1986. 2

[7] D. Terzopoulos. Multilevel computational processes for visual surface reconstruction. *CVGIP*, 1983. 2

[8] J. Wills and S. Belongie. A feature-based approach for determining dense long range correspondences. In *ECCV*, 2004. 1

6.2.2 Paper (JMIV'08) – *Maximizing the Predictivity of Smooth Deformable Image Warps Through Cross-Validation*

J Math Imaging Vis
DOI 10.1007/s10851-007-0062-1

Maximizing the Predictivity of Smooth Deformable Image Warps through Cross-Validation

Adrien Bartoli

© Springer Science+Business Media, LLC 2008

Abstract Estimating smooth image warps from landmarks is an important problem in computer vision and medical image analysis. The standard paradigm is to find the model parameters by minimizing a compound energy including a data term and a smoother, balanced by a 'smoothing parameter' that is usually fixed by trial and error.

We point out that warp estimation is an instance of the general supervised machine learning problem of fitting a flexible model to data, and propose to learn the smoothing parameter while estimating the warp. The leading idea is to depart from the usual paradigm of minimizing the energy to the one of maximizing the predictivity of the warp, *i.e.* its ability to do well on the entire image, rather than only on the given landmarks. We use cross-validation to measure predictivity, and propose a complete framework to solve for the desired warp. We point out that the well-known non-iterative closed-form for the leave-one-out cross-validation score is actually a good approximation to the true score and show that it extends to the warp estimation problem by replacing the usual vector two-norm by the matrix Frobenius norm. Experimental results on real data show that the procedure selects sensible smoothing parameters, very close to user selected ones.

Keywords Image registration · Landmarks · Warps · Cross-validation

A. Bartoli (✉)
LASMEA, CNRS/Université Blaise Pascal, Clermont-Ferrand, France
e-mail: Adrien.Bartoli@gmail.com

A. Bartoli
DIKU, University of Copenhagen, Copenhagen, Denmark

1 Introduction

The image registration problem is important since it directly relates to numerous applications, for instance deformable surface augmentation in computer vision, see *e.g.* [21], or multimodal image fusion in medical imaging, see *e.g.* [15].

The problem has been tackled in several different ways. A commonly agreed paradigm is to minimize some compound energy including a data term and a smoother [18]. The latter is weighted so that the estimated warp is smooth but still close to interpolating the landmarks. Most of the work uses trial and error to manually set an acceptable value for this weight, called the *smoothing parameter*. The energy can obviously not be minimized over the smoothing parameter since the result would always be zero.

The purpose of this paper is to bring a simple method that jointly learns the warp and the smoothing parameter. The key idea is to make the warp as general as possible in the sense of making it able to explain the deformation of the entire image, given a restricted set of landmarks. This is different from the classical approach that makes the warp interpolate the landmarks as best as possible, given some smoothing parameter. This is strongly inspired by the machine learning paradigm of supervised learning from examples: the source image landmarks are the inputs and the target image landmarks are the corresponding outputs. In this setting, the classical approach is an empirical risk minimization algorithm. The smoothing parameter controls the model complexity since increasing smoothness decreases the number of effective model parameters.

Determining smoothing weights and other parameters such as kernel widths is a common machine learning problem. A successful approach is to consider the expected prediction error, also termed test or generalization error, which, as the smoothing weight varies, measures the bias-variance

trade-off, see *e.g.* [22]. For the warp estimation problem, the generalization error can not be computed exactly since the number of landmarks is usually low and their distribution is unknown. There are however several ways to approximate the generalization error. The so-called model selection criteria such as BIC, AIC and GRIC have been successfully applied to pick up the best model in a discrete set of possible models. For instance, given two images of a rigid scene, one must choose between, say a homography and the fundamental matrix, see [14, 26]. Determining the smoothing parameter is however not a model selection problem since it does not change the actual warp model, but the estimation method.[1] A related approach is MDL, that has been used in medical image registration to register sets of multiple images, see *e.g.* [16], and for the Structure-from-Motion problem in [17].

The approach we follow is to split the data points in a training and a test set, and select the smoothing parameter for which the trained model minimizes the test error. Since the number of landmarks is usually small, we follow the approach of recycling the test set, in a leave-one-out cross-validation (LOOCV) manner. This technique was introduced in [1, 28]. It is related to the Jackknife and bootstrap techniques of sampling the dataset so that statistics can be drawn from it, and has been widely applied in machine learning, see *e.g.* [2]. For linear least squares (LLS) problems, there exists a non-iterative closed-form giving the LOOCV score. It is very close to the prediction sum of squares (PRESS) statistic and the studentized residuals.[2] We show that, while exact for the PRESS, this closed-form is actually an approximation of the LOOCV score, which turns out to be a very good approximation for typical parameter values. For the warp estimation problem, each landmark brings two equations through its two-dimensional coordinates. These two equations are said to be linked since they must be handled jointly (it would be meaningless to select one coordinate of a landmark for the training set and the other one for the test set). We show that the existing LOOCV closed-form extends to the linked measurement case by replacing the usual vector two-norm by the matrix Frobenius norm, and that this holds true for any dimension of the target space.

We point out that cross-validation is very different from the Random Sample Consensus (RANSAC) paradigm [9]. The latter trains the model using randomly sampled sets of minimal data, test on the rest of the data, and keeps the model with the largest 'consensus set'. It is meant to robustly estimate the model parameters, while cross-validation aims at quantifying the predictivity of the model. It is not obvious how RANSAC could be used to estimate image warps since there is not a clear definition of what a minimal data set is in this case. The proposed method using cross-validation is not robust, in the sense that it does not cope with mismatched landmarks.

We implement the idea of using cross-validation to register images through a parametric registration framework based on landmarks. The warp is assumed to be linear for some nonlinearly lifted source landmark coordinates. This includes warps such as Free-Form Deformations (FFD) based on tensor products [24, 25] and Radial Basis Functions (RBF), see *e.g.* [3, 10]. Experimental results are reported for Thin-Plate Spline (TPS) warps [3] which are the bending energy minimizing RBF.

Paper Organization Section 2 reviews the standard LLS estimation of warps from landmarks. Section 3 derives our approximate non-iterative closed-form to the LOOCV score and shows how it relates to the generalized cross-validation (GCV) score. Section 4 reports experimental results and Sect. 5 concludes. Finally, Appendix 1 reviews the TPS and derives our feature-driven parameterization, Appendix 2 brings a proof of the LOOCV lemma, and Appendix 3 an experimental evaluation of the closed-form LOOCV formula.

Notation Vectors are in bold fonts, *e.g.* $\mathbf{p}$, and matrices in sans-serif, *e.g.* A. Matrix, vector transpose and matrix inverse are written as in $\mathbf{p}^\mathsf{T}$, A^T and A^{-1}. Vector two-norm is denoted as in $\|\mathbf{x}\|_2 = \sqrt{\mathbf{x}^\mathsf{T}\mathbf{x}}$ and matrix Frobenius norm as in $\|\mathsf{A}\|_\mathcal{F} = \sqrt{\mathrm{tr}(\mathsf{A}^\mathsf{T}\mathsf{A})}$, where tr is the matrix trace operator. We stress that $\|\mathsf{A}\|_\mathcal{F}^2 = \|\mathbf{a}_1\|_2^2 + \|\mathbf{a}_2\|_2^2 + \cdots$, where $\mathbf{a}_1, \mathbf{a}_2, \ldots$ are the columns of matrix A. The real and projective spaces of dimension n are respectively written $\mathbb{R}^n$ and $\mathbb{P}^n$.

2 Landmark-Based Warp Estimation

Let $\mathbf{p} \in \mathbb{R}^2$ be a landmark coordinate vector in the source image. The warp $\mathcal{W} : \mathbb{R}^2 \times \mathbb{R}^{l\times 2} \mapsto \mathbb{R}^2$ maps a point from the source to the target image and depends on a set of parameters (often a set of l control points) in matrix $\mathsf{L} \in \mathbb{R}^{l\times 2}$ as:

$$\mathcal{W}(\mathbf{p}; \mathsf{L}) \stackrel{\text{def}}{=} \mathsf{L}^\mathsf{T} \nu(\mathbf{p}), \tag{1}$$

with $\nu : \mathbb{R}^2 \mapsto \mathbb{R}^l$ some nonlinear lifting function, which outputs an l-vector representing the lifted coordinates of a landmark. The lifted coordinates are linearly projected to $\mathbb{R}^2$ to give the predicted point in the target image. This general model encompasses FFDs and general RBFs. As an example, the lifting function for TPS warps is derived in Appendix 1.

[1] Another reason is that most of the model selection criteria requires that the distribution of the data point to model residuals has a known parametric form, which is clearly not the case in general for empirical smooth deformable warps.

[2] The PRESS statistic is similar to the LOOCV score but for a cost function with a data term only.

Let $\mathbf{p}_j \leftrightarrow \mathbf{q}_j$, $j = 1, \ldots, m$ be m landmark correspondences between the two images. Let ϵ_j be some random variable representing the noise and the deviation between the physics and the warp model, *i.e.* $\mathbf{q}_j = \mathcal{W}(\mathbf{p}_j; \mathsf{L}) + \epsilon_j$, from which the mean sum of squared residuals (MSR) is:

$$\mathcal{E}_d^2(\mathsf{L}) \stackrel{\text{def}}{=} \frac{1}{m} \sum_{j=1}^{m} \|\mathcal{W}(\mathbf{p}_j; \mathsf{L}) - \mathbf{q}_j\|_2^2.$$

It plays the role of a *data term* as it measures the transfer error, *i.e.* the discrepancy between the predicted and the measured target landmarks. It is used in conjunction with a smoother $\mathcal{E}_s$ in a compound cost function:

$$\mathcal{E}^2(\mathsf{L}; \mu) \stackrel{\text{def}}{=} \mathcal{E}_d^2(\mathsf{L}) + \mu^2 \mathcal{E}_s^2(\mathsf{L}),$$

with μ the smoothing parameter. The smoothing term is usually based on partial derivatives of the warp, such as the second derivatives:

$$\mathcal{B}^2(\mathsf{L}) \stackrel{\text{def}}{=} \int_{\mathbb{R}^2} \left\| \frac{\partial^2 \mathcal{W}}{\partial \mathbf{p}^2}(\mathbf{p}; \mathsf{L}) \right\|_{\mathcal{F}}^2 d\mathbf{p}. \tag{2}$$

Other examples are elastic registration which uses spring terms [6] and fluid registration which uses viscosity [4]. A different way of controlling the smoothness is to directly change the number of warp parameters, such as the number of control points in FFD-based registration [24]. Brownian warps are proposed in [20] along with a smoother constraining the estimated warp to be invertible [19]. Depending on the warp being used, the integral in (2) needs to be discretized. Note that using TPS warps allows to solve the integral in closed-form, as is shown in Appendix 1. We assume that it can anyway be fairly approximated by a discrete differential operator or any other matrix operator, and define:

$$\mathcal{E}_s^2(\mathsf{L}) \stackrel{\text{def}}{=} \|\mathsf{Z}\mathsf{L}\|_{\mathcal{F}}^2 \approx \mathcal{B}^2(\mathsf{L}). \tag{3}$$

The compound cost function thus writes as:

$$\mathcal{E}^2(\mathsf{L}; \mu) = \frac{1}{m} \|\mathsf{N}\mathsf{L} - \Xi\|_{\mathcal{F}}^2 + \mu^2 \|\mathsf{Z}\mathsf{L}\|_{\mathcal{F}}^2$$

with

$$\mathsf{N}^\mathsf{T} \stackrel{\text{def}}{=} \begin{pmatrix} \nu(\mathbf{p}_1) & \cdots & \nu(\mathbf{p}_m) \end{pmatrix}$$

and

$$\Xi^\mathsf{T} \stackrel{\text{def}}{=} \begin{pmatrix} \mathbf{q}_1 & \cdots & \mathbf{q}_m \end{pmatrix}.$$

Using the matrix Frobenius norm is a natural choice since, as the vector two-norm, it is based on summing squared matrix or vector elements. Given the smoothing parameter μ, the warp parameters $\hat{\mathsf{L}}(\mu)$ are solved for through:

$$\begin{aligned} \hat{\mathsf{L}}(\mu) &= \arg\min_{\mathsf{L}} \mathcal{E}^2(\mathsf{L}; \mu) && (4) \\ &= \arg\min_{\mathsf{L}} \left\| \begin{pmatrix} \mathsf{N} \\ \sqrt{m}\mu\mathsf{Z} \end{pmatrix} \mathsf{L} - \begin{pmatrix} \Xi \\ 0 \end{pmatrix} \right\|_{\mathcal{F}}^2 \\ &= \underbrace{(\mathsf{N}^\mathsf{T}\mathsf{N} + m\mu^2 \mathsf{Z}^\mathsf{T}\mathsf{Z})^{-1} \mathsf{N}^\mathsf{T}}_{\mathsf{T}(m\mu^2)} \Xi. && (5) \end{aligned}$$

The *influence matrix* T maps the target landmark coordinates in Ξ to the warp coefficients $\hat{\mathsf{L}}$ and plays an important role in the cross-validation technique given in the next section. We note that the matrix Frobenius norm naturally allows handling the linked equations induced by the two dimensions of landmark coordinates.

3 Maximizing Predictivity by Cross-Validation

The idea of cross-validation is to approximate the generalization error by splitting the data in a training and a test set, and average the test error over several such partitionings. There are different kinds of cross-validations, including leave-one-out (LOOCV), v-fold and generalized cross-validation (GCV). The two latter ones are usually preferred for computation efficiency. We use LOOCV and show that it can be very efficiently approximated in closed-form, for models in the form (1). The formula for LOOCV is the same as for the PRESS [1], except that the hat matrix is replaced by the *influence matrix*, incorporating the smoother, and that an approximation needs to be made in the derivation.

The LOOCV score is defined as a function of the smoothing parameter μ:

$$\mathcal{E}_g^2(\mu) \stackrel{\text{def}}{=} \frac{1}{m} \sum_{j=1}^{m} \|\mathcal{W}(\mathbf{p}_j, \hat{\mathsf{L}}_{(j)}(\mu)) - \mathbf{q}_j\|_2^2, \tag{6}$$

where $\hat{\mathsf{L}}_{(j)}(\mu)$ are the model parameters estimated with all but the j-th landmark:

$$\hat{\mathsf{L}}_{(j)}(\mu) \stackrel{\text{def}}{=} \arg\min_{\mathsf{L}} \mathcal{E}_{(j)}^2(\mathsf{L}; \mu). \tag{7}$$

We therefore have to solve the following nested optimization problem to get the most predictive solution $\hat{\mathsf{L}}$, obtained by plugging the optimal $\hat{\mu}$ in (4), giving:

$$\hat{\mathsf{L}} \stackrel{\text{def}}{=} \arg\min_{\mathsf{L}} \mathcal{E}^2(\mathsf{L}; \arg\min_{\mu} \mathcal{E}_g^2(\mu)).$$

At first glance, the LOOCV score seems computationally expensive, making its minimization over μ extremely costly if not infeasible in a reasonable amount of time on a standard computer. It turns out that there actually is a non-iterative closed-form for the LOOCV score which does not require solving the system m times as a trivial, greedy application of (6) requires.

The closed-form is based on the so-called LOOCV lemma, demonstrated in Appendix 2. Consider an LLS problem, and a reduced problem with only a subset of the measurements. The lemma says that replacing in the full dataset problem the measurements by their prediction with the solution to the reduced problem makes the solution to this modified problem the same as for the reduced one. In other words, define $\tilde{\Xi}^j$ as Ξ except that the j-th row, corresponding to the j-th landmark, is replaced by the prediction $\mathcal{W}(\mathbf{p}_j;\hat{\mathsf{L}}_{(j)}(\mu))^{\mathsf{T}}$, *i.e.*:

$$\tilde{\Xi}^j \stackrel{\text{def}}{=} \Xi - \mathbf{e}_j(\mathbf{q}_j - \mathcal{W}(\mathbf{p}_j;\hat{\mathsf{L}}_{(j)}(\mu)))^{\mathsf{T}}, \tag{8}$$

with $\mathbf{e}_j$ a zero vector with one at the j-th entry and $\hat{\mathsf{L}}_{(j)}(\mu)$ the solution to the reduced problem. The lemma states:

$$\hat{\mathsf{L}}_{(j)}(\mu) = \mathsf{T}\big((m-1)\mu^2\big)\tilde{\Xi}^j. \tag{9}$$

In other words, the solution is a constant linear function of a slightly modified right-hand side matrix. Although it could seem weird that the unknown estimate $\hat{\mathsf{L}}_{(j)}(\mu)$ is used to make a prediction in order to artificially create a problem to solve for this estimate, it actually is essential for deriving the non-iterative closed-form we are aiming at, as is clearly shown below.

Recall that matrix T maps the target landmarks to the model parameters while matrix N maps the model parameters to the predicted landmarks. We therefore construct the influence matrix H which maps the target landmarks to the predicted ones as:

$$\mathsf{H}(\gamma) \stackrel{\text{def}}{=} \mathsf{NT}(\gamma) = \mathsf{N}(\mathsf{N}^{\mathsf{T}}\mathsf{N} + \gamma\mathsf{Z}^{\mathsf{T}}\mathsf{Z})^{-1}\mathsf{N}^{\mathsf{T}}.$$

Matrix H has size $(m \times m)$, *i.e.* it has as many rows and columns as there are landmark correspondences, and is symmetric. We write $\mathbf{h}_j(\gamma)$, $j = 1,\ldots,m$ the columns (or rows) of $\mathsf{H}(\gamma)$. This allows us to write:

$$\mathcal{W}(\mathbf{p}_j;\hat{\mathsf{L}}(\mu)) = \Xi^{\mathsf{T}}\mathbf{h}_j(m\mu^2)$$

$$\mathcal{W}(\mathbf{p}_j;\hat{\mathsf{L}}_{(j)}(\mu)) = (\tilde{\Xi}^j)^{\mathsf{T}}\mathbf{h}_j((m-1)\mu^2).$$

Taking the difference between the two equations and approximating $\mathbf{h}_j \stackrel{\text{def}}{=} \mathbf{h}_j(m\mu^2) \approx \mathbf{h}_j((m-1)\mu^2)$ gives:

$$\mathcal{W}(\mathbf{p}_j;\hat{\mathsf{L}}(\mu)) - \mathcal{W}(\mathbf{p}_j;\hat{\mathsf{L}}_{(j)}(\mu)) \approx (\Xi - \tilde{\Xi}^j)^{\mathsf{T}}\mathbf{h}_j.$$

Substituting the definition (8) of $\tilde{\Xi}^j$ gives:

$$\begin{aligned}&\mathcal{W}(\mathbf{p}_j;\hat{\mathsf{L}}(\mu)) - \mathcal{W}(\mathbf{p}_j;\hat{\mathsf{L}}_{(j)}(\mu))\\ &\quad\approx \mathbf{h}_j^{\mathsf{T}}\mathbf{e}_j(\mathbf{q}_j - \mathcal{W}(\mathbf{p}_j;\hat{\mathsf{L}}_{(j)}(\mu))).\end{aligned}$$

Writing $h_{j,j} \stackrel{\text{def}}{=} \mathbf{h}_j^{\mathsf{T}}\mathbf{e}_j$ the diagonal elements of $\mathsf{H}(m\mu^2)$, we get:

$$\begin{aligned}&\mathcal{W}(\mathbf{p}_j;\hat{\mathsf{L}}(\mu)) - \mathcal{W}(\mathbf{p}_j;\hat{\mathsf{L}}_{(j)}(\mu))\\ &\quad\approx h_{j,j}(\mathbf{q}_j - \mathcal{W}(\mathbf{p}_j;\hat{\mathsf{L}}_{(j)}(\mu))),\end{aligned}$$

that we rearrange to:

$$h_{j,j}\mathbf{q}_j + (1-h_{j,j})\mathcal{W}(\mathbf{p}_j;\hat{\mathsf{L}}_{(j)}(\mu)) \approx \mathcal{W}(\mathbf{p}_j;\hat{\mathsf{L}}(\mu)).$$

Subtracting $\mathbf{q}_j$ on each side gives:

$$\begin{aligned}&h_{j,j}\mathbf{q}_j + (1-h_{j,j})\mathcal{W}(\mathbf{p}_j;\hat{\mathsf{L}}_{(j)}(\mu)) - \mathbf{q}_j\\ &\quad\approx \mathcal{W}(\mathbf{p}_j;\hat{\mathsf{L}}(\mu)) - \mathbf{q}_j,\\ &(1-h_{j,j})(\mathcal{W}(\mathbf{p}_j;\hat{\mathsf{L}}_{(j)}(\mu)) - \mathbf{q}_j) \approx \mathcal{W}(\mathbf{p}_j;\hat{\mathsf{L}}(\mu)) - \mathbf{q}_j,\\ &\mathcal{W}(\mathbf{p}_j;\hat{\mathsf{L}}_{(j)}(\mu)) - \mathbf{q}_j \approx \frac{1}{1-h_{j,j}}(\mathcal{W}(\mathbf{p}_j;\hat{\mathsf{L}}(\mu)) - \mathbf{q}_j).\end{aligned}$$

We thus have obtained an analytical, non-iterative expression giving each term in the sum for the LOOCV score (6), that we can rewrite as:

$$\mathcal{E}_g^2(\mu) \approx \frac{1}{m}\left\|\operatorname{diag}\left(\frac{1}{\mathbf{1}-\operatorname{diag}(\mathsf{H}(m\mu^2))}\right)(\mathsf{N}\hat{\mathsf{L}}(\mu) - \Xi)\right\|_{\mathcal{F}}^2, \tag{10}$$

where $\operatorname{diag}(\mathsf{M})$ is a vector containing the diagonal entries of matrix M and $\operatorname{diag}(\mathbf{v})$ is a diagonal matrix with as diagonal entries the elements of vector $\mathbf{v}$, and $\mathbf{1}$ is a vector of ones.

Minimizing the LOOCV score is done through the closed-form (10). Most of the methods in the literature are specific to the GCV score, which uses the approximation $\operatorname{diag}(\mathsf{H}(m\mu^2)) \approx \operatorname{tr}(\mathsf{H}(m\mu^2))\mathrm{I}$, with I the identity matrix, which allows simplifying the closed-form $\mathcal{E}_g$ further, see [27]. The minimization problem however remains nonlinear, and most of the methods for the GCV score can be applied to the LOOCV score, eventhough it is often neglected in the literature. Possible methods range from golden section search [5] and sampling (with optional local polynomial interpolation), *e.g.* [12, 13]. We tried several different methods. Most of them find the correct minimum in all cases. The fastest one is downhill simplex, which has typical computation times of less than half a second for $m \approx 50$ landmarks and $l \approx 25$ deformation centres on a standard PC running our MATLAB implementation.[3] This computation time, although not prohibitive, is much higher than that of a straightforward fitting of the warp, given the smoothing parameter.

The approximation based on $\mathbf{h}_j = \mathbf{h}_j(m\mu^2)$ in the above derivation allows to derive the closed-form (10). We call it the m-approximation. We compared its value against the direct evaluation of (6), giving the 'true' LOOCV score, on a bunch of typical values, and with another candidate approximation using $\mathbf{h}_j = \mathbf{h}_j((m-1)\mu^2)$, called the

[3]The downhill simplex or Nelder-Mead algorithm is implemented within the `fminsearch` MATLAB function.

Fig. 1 (*left*, *middle*) The source and target images in the dishcloth dataset overlaid with the 130 manually clicked point correspondences. (*right*) The ancillary image, showing the dishcloth flat, which is used to create a warp visualization grid, as shown on Fig. 2

Fig. 2 (*left*) The ancillary image showing the warp visualization grid over the region of interest. (*middle*) The warp visualization grid transferred from the ancillary image to the source image. (*right*) The 10×10 deformation centre grid in the source image

$(m-1)$-approximation. The results are reported in Appendix 3. Our conclusions are that there is no significant difference between the two approximations, albeit that the m-approximation has a better behavior than the $(m-1)$-approximation in the sense that its minimum is located closer to the true one, and has the same value as the true minimum LOOCV score. We also tried approximations based on $\mathbf{h}_j = \mathbf{h}_j((\eta m + (1-\eta)(m-1))\mu^2) = \mathbf{h}_j((m-1+\eta)\mu^2)$ with $\eta \in [0,1]$ – none of them did better than the m-approximation.

4 Experimental Results

We evaluated our algorithm on several datasets. For three of them we show results. Most of the other methods in the literature assume that the smoothing parameter is given and estimate the warp parameters, whereas the proposed algorithm estimate both the warp and smoothing parameters.

4.1 The Dishcloth Dataset

This dataset has three images of a dishcloth for which 130 corresponding points were manually marked, see Fig. 1. The two images that we use for testing our algorithm show the dishcloth with the same deformation but from a different viewpoint. The point correspondences cover the entire dishcloth, which remains entirely visible. This dataset is thus easy in the sense that many point correspondences are available and that the two images are quite similar.

The third image in this dataset shows the dishcloth flatten on a table, and is called the ancillary image. It is used to create a warp visualization grid in the source image, as shown on Fig. 2 and explained below. First, we mark the four corners of the region of interest in the ancillary image, and use them to create a homography of $\mathbb{P}^2$ mapping the canonical unit square to these four corners. This is used to transfer a regular grid from the canonical unit square to the ancillary image. Second, we use the point correspondences to compute a deformable warp from the ancillary to the source image, and use it to transfer the visualization grid to the source image. This visualization grid, although similar to the data points, is very useful to visualize the behavior of the warp independently of the actual data points.

We proceed to register the images and use a regular grid of 10×10 deformation centres, as shown in Fig. 2. Figure 3 shows the LOOCV score and the RMSR as functions of the smoothing parameter μ. We observe that they both asymptotically tend to respectively the PRESS statistic and RMSR of a two-dimensional affine image transformation, which we measured to be respectively 3.14 pixels and 3.06 pixels. The minimization finds the LOOCV optimal smoothing parameter $\hat{\mu} \approx 0.55$. Zooming onto the graph shows that

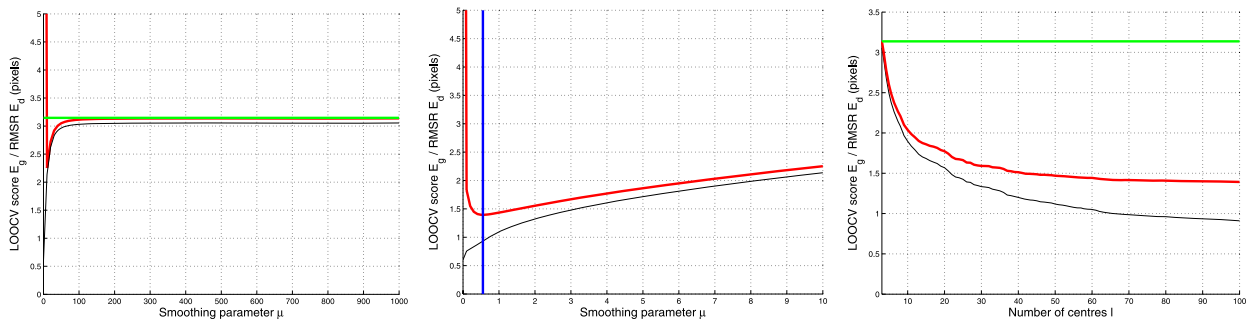


Fig. 3 (*left*) The LOOCV score (*thick*, *red curve*) and RMSR (*thin*, *black curve*) as functions of the smoothing parameter μ—the *green horizontal line* is the PRESS for an affinity. (*middle*) Zoom onto the left graph—the *blue vertical line* shows the selected optimal smoothing parameter $\hat{\mu}$. (*right*) The mean LOOCV score (*thick*, *red curve*) and RMSR (*thin*, *black curve*) as functions of the number of deformation centres l

LOOCV optimal solution
$\mu = \hat{\mu} \approx 0.55$
RMSR ≈ 0.93 pixels
LOOCV ≈ 1.40 pixels

Over-smoothed solution
$\mu = 10\hat{\mu} \approx 5.53$
RMSR ≈ 1.77 pixels
LOOCV ≈ 1.91 pixels

Extremely over-smoothed solution
$\mu = 10,000$
RMSR ≈ 3.06 pixels
LOOCV ≈ 3.14 pixels

Fig. 4 The visualization grid predicted by the warp for (*left*) the LOOCV optimal solution parameter, (*middle*) an exaggerated smoothing parameters and (*right*) an extreme smoothing parameter corresponding to the asymptotically affine behavior of the warp

it actually has a shallow minimum. This is explained by the fact that this dataset is 'simple', in the sense that the image deformation is limited. Once a sufficient amount of smoothness is reached, it is not that critical to oversmooth. As expected, the RMSR is a monotonic function of μ: the smoother the warp, the lower the effective number of parameters and so the higher the training RMSR error. Figure 3 also shows the LOOCV score and the RMSR as functions of the number of centres l. These curves were obtained by randomly sampling centres in the convex hull of the source landmarks, and for each set of centres, finding the smoothing parameter minimizing the LOOCV score. It can be seen that both the LOOCV score and the RMSR are decreasing functions of m. This is explained by the fact that since an adaptive smoothing parameter is used, adding more parameters can not degrade the quality of the warp, since the extra parameters just get smoothed out. This means that without any prior information, the number of deformation centres should be chosen large. On this particular example, it is clear that choosing more than 40 deformation centres, say, does not bring a significant improvement to the quality of the warp.

Finally, Fig. 4 shows the visualization grid transferred to the target image, for different smoothing parameters. As was expected from the shape of the LOOCV score in Fig. 3, oversmoothing has a limited effect on the estimated warp. Note however that the LOOCV score grows by more than a third, from 1.40 pixels to 1.91 pixels, when 10 times the optimal smoothing parameter is used.

4.2 The Paper Sheet Dataset

This dataset has two images of a paper sheet shown in Fig. 5. One of the images shows the paper sheet flat, with substantial radial distortion. The other image shows the paper smoothly bent in such a way that a self-occlusion shows up, *i.e.* part of the surface is being occluded by itself. We manually clicked 53 points on both images as shown in Fig. 5. We clicked the four corners or the paper sheet in the flat paper image, and, as for the dishcloth ancillary image, created a regular visualization grid. It is used to visually assess the quality of an estimated warp.

Fig. 5 (*left*, *middle*) The source and target images of the self-occluding paper sheet dataset overlaid with the 53 manually clicked point correspondences. (*right*) The warp visualization grid covering the region of interest

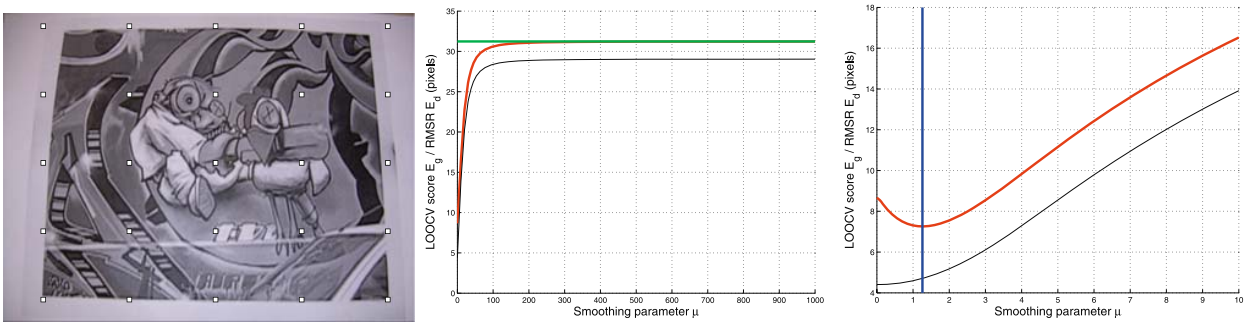


Fig. 6 (*left*) The 5×5 deformation centre grid. (*middle*) The LOOCV score (*thick*, *red curve*) and RMSR (*thin*, *black curve*) as functions of the smoothing parameter μ—the *green horizontal line* is the PRESS for an affinity. (*right*) Zoom onto the middle graph—the *blue vertical line* shows the selected optimal smoothing parameter $\hat{\mu}$

This dataset is much more difficult than the dishcloth dataset, in the sense that due to surface self-occlusion in the target image, a large part of the region of interest visible in the source image disappears in the target image.

Figure 6 shows the 5×5 grid of deformation centres we selected over the source image. This figure also shows the LOOCV score as a function of the smoothing parameter μ. We observe that it asymptotically converges to the PRESS score for an affine image transformation, which is 31.24 pixels. The RMSR has the same behavior in that it converges to the RMSR for the affine transformation (not shown on the graph), which is 29.05 pixels. Zooming onto the beginning of the LOOCV curve shows that it actually has a well defined minimum, and that this is what our algorithm selects as optimal smoothing parameter $\hat{\mu}$.

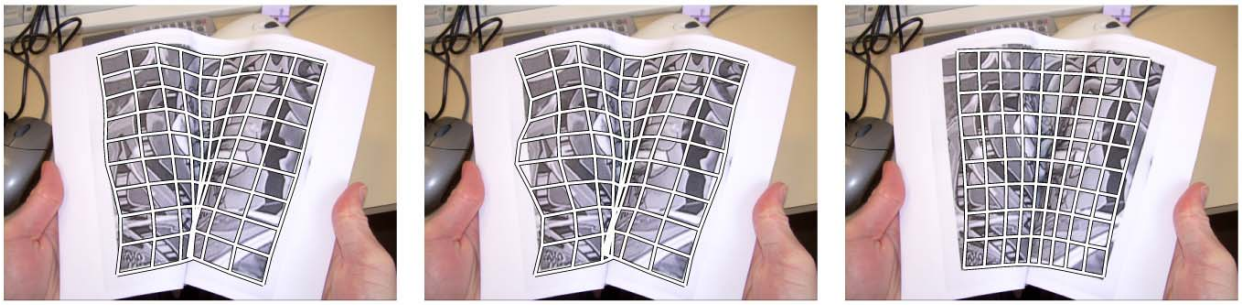
The LOOCV optimal warp we obtain is shown on Fig. 7. An under- and an over-smoothed solutions are also shown for comparison. The selected $\hat{\mu}$ clearly corresponds to what one would have chosen by tweaking, since it is visually very satisfying.

We observed that the LOOCV score and the RMSR are, as for the dishcloth example, monotonic decreasing functions of the number of deformation centres l (this graph is not shown). Choosing $l = 25$ for this particular example is a sensible choice.

4.3 The Spine Dataset

This dataset is extracted from the one used in [7]. It consists of lateral, lumbar spine X-ray images, similar to the pair of example images shown in Fig. 8 for two different patients. Each image has been annotated by experienced radiologists who placed 6 points on the corners and in the middle of the vertebra endplates. This provides a total of 36 landmarks since L1 to L4, and the 2 neighboring vertebrae are used in every image. They also manually marked the outlines of the L1 to L4 vertebrae in each image. As can be seen from the outlines, the vertebrae show different degrees of fracture at follow up. On the source image, L3 and L4 show a moderate biconcave deformity, while on the target image, L1 shows a severe wedge deformity.

We estimated a warp with a grid of 3×3 deformation centres, as show on Fig. 8. The LOOCV score we obtained is 12.60 pixels and the RMSR is 9.71 pixels. This is quite high, as the noticeable discrepancy between the target and predicted landmarks shows.



LOOCV optimal solution
$\mu = \hat{\mu} \approx 1.26$
RMSR ≈ 4.72 pixels
LOOCV ≈ 7.27 pixels

Under-smoothed solution
$\mu = 0$
RMSR ≈ 4.41 pixels
LOOCV ≈ 8.67 pixels

Over-smoothed solution
$\mu = 10\hat{\mu} \approx 12.58$
RMSR ≈ 15.97 pixels
LOOCV ≈ 18.53 pixels

Fig. 7 The visualization grid predicted by the warp for (*left*) the LOOCV optimal smoothing parameter, (*middle*) no smoothing at all and (*right*) an exaggerated smoothing parameter

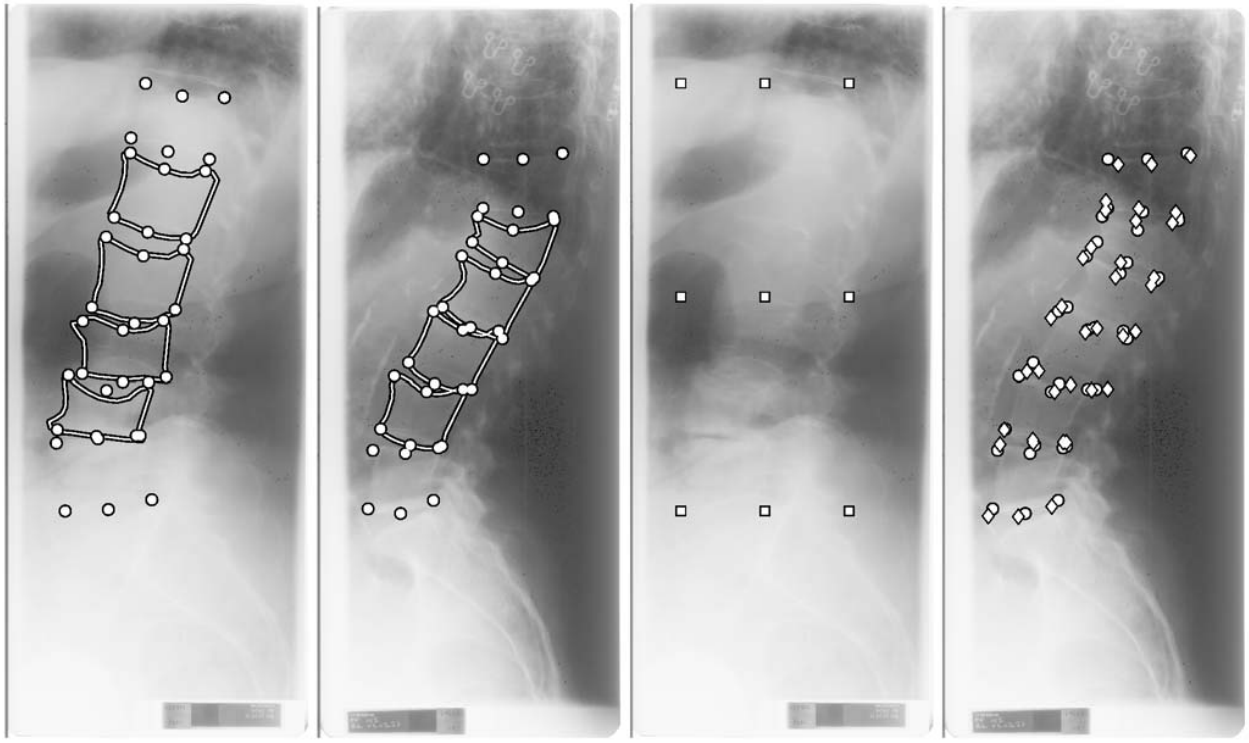


Fig. 8 (from *left to right*) The source and target images overlaid with the 36 landmarks (*circles*) and the vertebrae boundaries (see main text for details), the 3×3 grid of deformation centres we use in the source image, and the landmarks predicted by the LOOCV optimal warp (*diamonds*)

Figure 9 shows the visualization grid for different values of the smoothing parameter. We observe that the LOOCV optimal solution has a nice visual behavior. The under-smoothed one almost folds on itself, while the over-smoothed one is very rigid.

Figure 10 shows the LOOCV score and RMSR as functions of the smoothing parameter. The LOOCV has a well-defined minimum $\hat{\mu} \approx 2.17$, and the curves have the same shape as for the two previous datasets. These two graphs however show a novel curve, representing the target to transferred boundary distance. This is computed as follow. Given a warp estimate, we transfer each point on the source boundary to the target image, and measure the distance to the closed point onto the target boundary. Averaging over the source boundary points gives the 'boundary error'. What we observe is that this boundary error has a minimum,

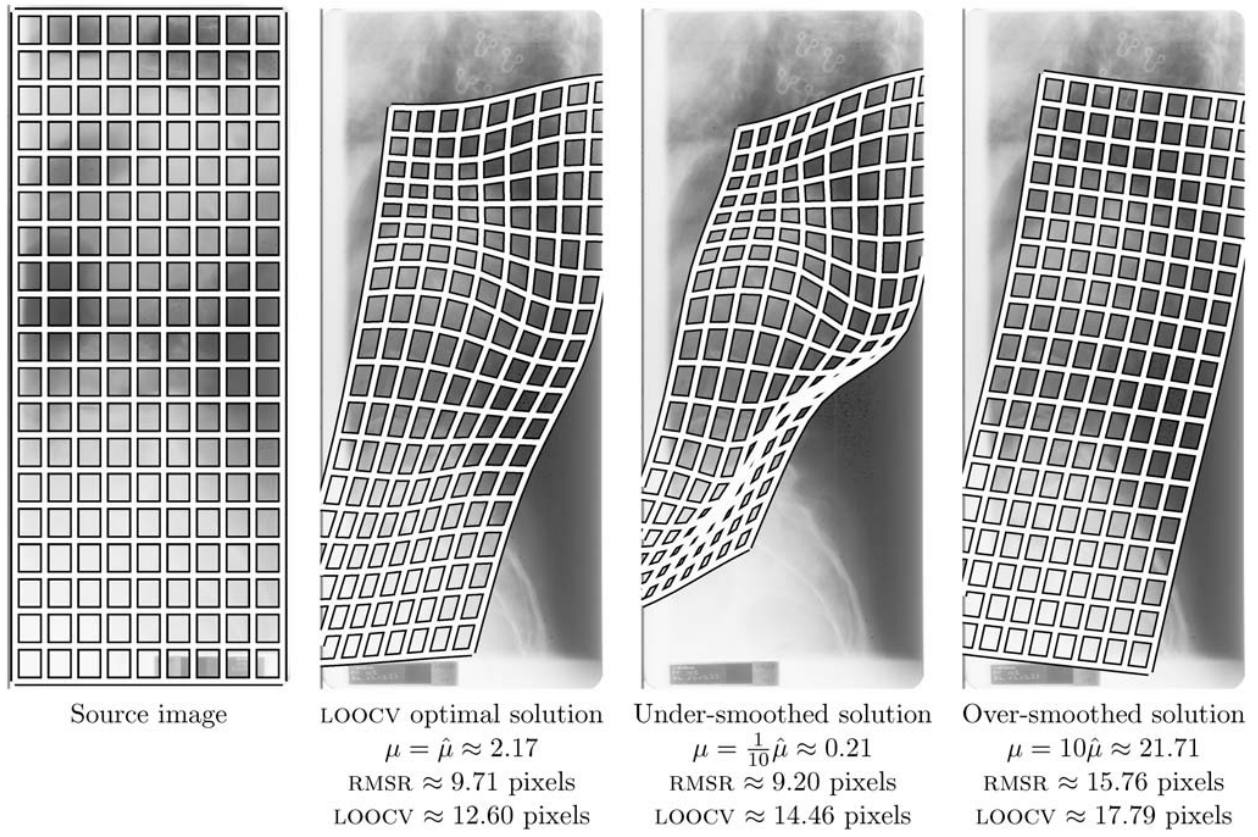


Fig. 9 (*from left to right*) The source image with the visualization grid, the LOOCV optimal smoothing parameter, an under-smoothed and an over-smoothed solutions

which is located at a slightly lower value than the LOOCV optimal smoothing parameter. We tried different combinations of images: the LOOCV score and the boundary error exhibited the same behavior in all cases, *i.e.* minimizing the LOOCV score slightly overestimates the location of the minimum for the boundary error. Recall that the warp, and thus the LOOCV score, are computed only from the 36 landmarks. This means that these landmarks and the vertebra outlines are strongly correlated, which was expected since those landmarks actually form the basis for classical semi-quantitative vertebra fracture grading strategies, see [11].

Figure 11 shows the vertebra boundaries, and allows one to visually compare the marked and the predicted boundaries in the target image. It is seen that the LOOCV optimal and the under-smoothed solutions are both visually satisfying. They obviously fail to capture all the subtle shape changes, but account for the main deformations. This was expected since the LOOCV minimum over-estimates the boundary error minimum. The over-smoothed solution clearly misses important shape changes.

5 Conclusion

We described a framework for estimating a deformable image warp from landmarks based on a compound cost function including a data term and a smoother. The method, based on leave-one-out cross-validation, automatically determines the smoothing parameter balancing the data term and the smoother. We showed that a simple closed-form solution exists for computing the leave-one-out cross-validation score given the smoothing parameter, and minimize it with a downhill simplex algorithm, yielding reasonable computation time, typically much less than a second. We report convincing experimental results on various datasets.

Generally speaking, one possible issue with leave-one-out cross-validation is the "testing-on-training data" problem. This does not occur with the kind of data we use in this paper since the landmarks are usually sparse, but should be considered if more data are available, *e.g.* a pixel-wise displacement field, by using for instance an exclusion zone around each training point. There also exist pathological cases, for which the leave-one-out cross-validation score has

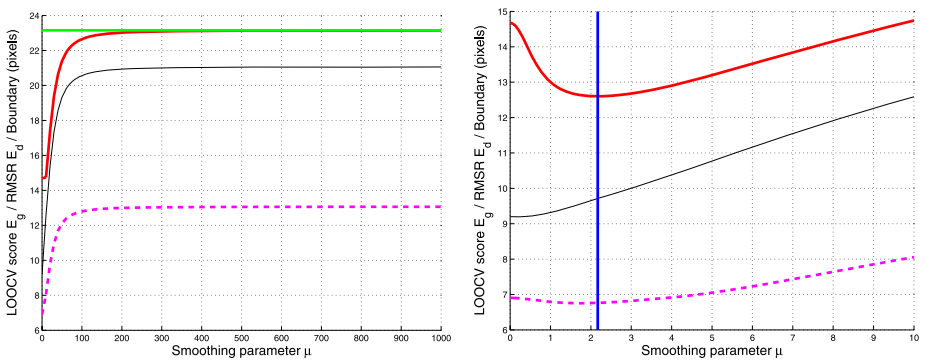


Fig. 10 (*left*) The LOOCV score (*thick*, *red curve*) and RMSR (*thin*, *black curve*) as functions of the smoothing parameter μ, as well as the boundary transfer error (*thick dashed*, *purple curve*)—the *green horizontal line* is the PRESS for an affinity. (*right*) Zoom onto the left graph—the *blue vertical line* shows the selected optimal smoothing parameter $\hat{\mu}$

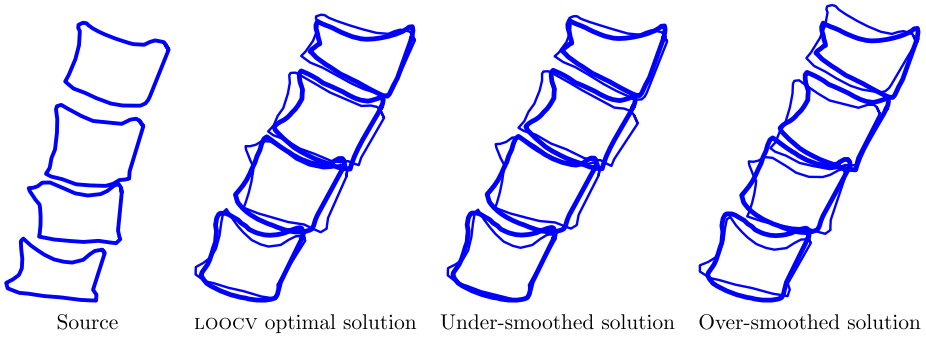


Fig. 11 (*from left to right*) The vertebra boundaries in the source image, and in the target image. The manually marked boundary is shown (*thick curve*), as well as the one transferred by the warp from the source image (*thin curve*) for different smoothing parameters

several local minima. How to find the optimal minimum in practice in a guaranteed manner is an open research topic.

Acknowledgements I want to thank Vincent Gay-Bellile for the self-occluding paper dataset, Florent Brunet for pointing me some references, Rabia Granlund and the authors of [7], in particular Mads Nielsen, for the spine dataset, and also Søren Olsen who has made my regular visits to Copenhagen possible.

Appendix 1: The Thin-Plate Spline

The TPS is an $\mathbb{R}^2 \to \mathbb{R}$ function driven by assigning target values α_k to control centres $\mathbf{c}_k$ with $k = 1, \ldots, l$ and enforcing several conditions: the TPS is the Radial Basis Function that minimizes the integral bending energy. The idea of using the thin-plate equation as an interpolation map is due to Duchon [8]. Standard $\mathbb{R}^2 \mapsto \mathbb{R}^2$ TPS-Warps are obtained by stacking two TPSs sharing their centres, as proposed by Bookstein [3]. This is described below, along with our feature-driven parameterization.

6.1 Standard Parameterization

The TPS is usually parameterized by an $l + 3$ coefficient vector $\boldsymbol{\eta}^\mathsf{T} = (\mathbf{w}^\mathsf{T}\, \mathbf{a}^\mathsf{T})$ and an internal smoothing parameter $\lambda \in \mathbb{R}^+$. There are l coefficients in $\mathbf{w}$ and three coefficients in $\mathbf{a}$. These coefficients can be computed from the $(l \times 1)$ target vector $\boldsymbol{\alpha}$. The TPS is given by:

$$\omega(\mathbf{p}, \boldsymbol{\eta}_{\boldsymbol{\alpha},\lambda}) \stackrel{\text{def}}{=} \left(\sum_{k=1}^{l} w_k \, \rho(d^2(\mathbf{p}, \mathbf{c}_k)) \right) + \mathbf{a}^\mathsf{T} \tilde{\mathbf{p}}, \tag{11}$$

where $\rho(d) = d \log(d)$ is the TPS kernel function for the squared distance and $\tilde{\mathbf{p}}^\mathsf{T} = (\mathbf{p}^\mathsf{T}\, 1)$. The coefficients in $\mathbf{w}$ must satisfy $\tilde{\mathsf{C}}^\mathsf{T}\mathbf{w} = \mathbf{0}$, where the k-th row of $\tilde{\mathsf{C}}$ is $\tilde{\mathbf{c}}_k$. These three 'side-conditions' ensure that the TPS has square integrable second derivatives. It is convenient to define the $(l + 3)$-vector $\boldsymbol{\ell}_\mathbf{p}$ as:

$$\boldsymbol{\ell}_\mathbf{p}^\mathsf{T} \stackrel{\text{def}}{=} (\rho(d^2(\mathbf{p}, \mathbf{c}_1)) \cdots \rho(d^2(\mathbf{p}, \mathbf{c}_l))\, \tilde{\mathbf{p}}^\mathsf{T}), \tag{12}$$

allowing the TPS (11) to be rewritten as a dot product:

$$\omega(\mathbf{p}, \boldsymbol{\eta}_{\boldsymbol{\alpha},\lambda}) = \boldsymbol{\ell}_\mathbf{p}^\mathsf{T} \boldsymbol{\eta}_{\boldsymbol{\alpha},\lambda}. \tag{13}$$

Equation (12) thus represents the first step in the nonlinear lifting function making the TPS-warp fit in the general warp definition (1) used in this paper.

6.2 Standard Estimation

Applying the TPS (11) to the centre $\mathbf{c}_r$ with target value α_r gives:

$$\left(\sum_{k=1}^{l} w_k \, \rho(d^2(\mathbf{c}_r, \mathbf{c}_k)) \right) + \mathbf{a}^\mathsf{T} \tilde{\mathbf{c}}_r = \alpha_r.$$

Combining the equations obtained for all the l centres with the side-conditions $\tilde{\mathsf{C}}^\mathsf{T}\mathbf{w} = \mathbf{0}$ in a single matrix equation gives:

$$\underbrace{\begin{pmatrix} \mathsf{K}_\lambda & \tilde{\mathsf{C}} \\ \tilde{\mathsf{C}}^\mathsf{T} & 0 \end{pmatrix}}_{\mathcal{D}} \underbrace{\begin{pmatrix} \mathbf{w} \\ \mathbf{a} \end{pmatrix}}_{\boldsymbol{\eta}_{\boldsymbol{\alpha},\lambda}} = \begin{pmatrix} \boldsymbol{\alpha} \\ \mathbf{0} \end{pmatrix}$$

$$\text{with } K_{r,k} \stackrel{\text{def}}{=} \begin{cases} \lambda & r = k, \\ \rho(d^2(\mathbf{c}_r, \mathbf{c}_k)) & \text{otherwise.} \end{cases}$$

Adding $\lambda\mathrm{I}$ to the leading block of the design matrix $\mathcal{D}$ to give K_λ acts as an internal smoother. An *ad hoc* method for finding λ is described in [23]. Solving for $\boldsymbol{\eta}_{\boldsymbol{\alpha},\lambda}$ by inverting $\mathcal{D}$ is the classical linear method for estimating the TPS coefficients [3]. The coefficient vector $\boldsymbol{\eta}_{\boldsymbol{\alpha},\lambda}$ is thus a nonlinear function of the internal smoothing parameter λ and a linear function of the target vector $\boldsymbol{\alpha}$.

6.3 A Feature-Driven Parameterization

We express $\boldsymbol{\eta}_{\boldsymbol{\alpha},\lambda}$ as a linear 'back-projection' of the target value vector $\boldsymbol{\alpha}$. This is modeled by the matrix $\mathcal{E}_\lambda$, nonlinearly depending on λ, given by the l leading columns of $\mathcal{D}^{-1}$:

$$\boldsymbol{\eta}_{\boldsymbol{\alpha},\lambda} = \mathcal{E}_\lambda \boldsymbol{\alpha}$$

$$\text{with } \mathcal{E}_\lambda \stackrel{\text{def}}{=} \begin{pmatrix} \mathsf{K}_\lambda^{-1}(\mathrm{I} - \tilde{\mathsf{C}}(\tilde{\mathsf{C}}^\mathsf{T}\mathsf{K}_\lambda^{-1}\tilde{\mathsf{C}})^{-1}\tilde{\mathsf{C}}^\mathsf{T}\mathsf{K}_\lambda^{-1}) \\ (\tilde{\mathsf{C}}^\mathsf{T}\mathsf{K}_\lambda^{-1}\tilde{\mathsf{C}})^{-1}\tilde{\mathsf{C}}^\mathsf{T}\mathsf{K}_\lambda^{-1} \end{pmatrix}. \tag{14}$$

This parameterization has the advantages to separate λ and $\boldsymbol{\alpha}$ and to introduce units.[4] The side-conditions are naturally enforced by this parameterization.

Incorporating the parameterization (14) into the TPS (13) we obtain what we call the *feature-driven* parameterization $\tau(\mathbf{p};\boldsymbol{\alpha},\lambda) = \omega(\mathbf{p};\boldsymbol{\eta}_{\boldsymbol{\alpha},\lambda})$ for the TPS:

$$\tau(\mathbf{p};\boldsymbol{\alpha},\lambda) \stackrel{\text{def}}{=} \boldsymbol{\ell}_\mathbf{p}^\mathsf{T}\mathcal{E}_\lambda\boldsymbol{\alpha}. \tag{15}$$

The square integral bending energy $\kappa = \int_{\mathbb{R}^2} \|\frac{\partial^2\tau}{\partial\mathbf{p}^2}(\mathbf{p};\boldsymbol{\alpha},\lambda)\|_\mathcal{F}^2 \, d\mathbf{p} = 8\pi\mathbf{w}^\mathsf{T}\mathsf{K}_\lambda\mathbf{w}$ is given by $\kappa = 8\pi\boldsymbol{\alpha}^\mathsf{T}\bar{\mathcal{E}}_\lambda\boldsymbol{\alpha}$, where $\bar{\mathcal{E}}_\lambda$ is the $(l \times l)$ *bending energy matrix* given by amputating $\mathcal{E}_\lambda$ of its last three rows:

$$\bar{\mathcal{E}}_\lambda \stackrel{\text{def}}{=} \mathsf{K}_\lambda^{-1}(\mathrm{I} - \tilde{\mathsf{C}}(\tilde{\mathsf{C}}^\mathsf{T}\mathsf{K}_\lambda^{-1}\tilde{\mathsf{C}})^{-1}\tilde{\mathsf{C}}^\mathsf{T}\mathsf{K}_\lambda^{-1}). \tag{16}$$

The bending energy matrix is symmetric and in the absence of internal regularization, *i.e.* for $\lambda = 0$, has rank $l - 3$. The eigenvectors corresponding to the $l - 3$ nonzero eigenvalues are the *principal warps*, the corresponding eigenvalues indicating their bending energy, as defined by Bookstein [3].

The TPS-warp is obtained by stacking two $\mathbb{R}^2 \mapsto \mathbb{R}$ TPSs. From (11), we get:

$$\begin{pmatrix} \tau(\mathbf{p};\boldsymbol{\alpha}_x,\lambda) \\ \tau(\mathbf{p};\boldsymbol{\alpha}_y,\lambda) \end{pmatrix} = (\boldsymbol{\ell}_\mathbf{p}^\mathsf{T}\mathcal{E}_\lambda\mathsf{L})^\mathsf{T},$$

where $\boldsymbol{\alpha}_x$ and $\boldsymbol{\alpha}_y$ are the first and second columns of L. The TPS warp is thus expressed in the form (1), *i.e.* $\mathcal{W}(\mathbf{p};\mathsf{L}) = \mathsf{L}^\mathsf{T}\nu(\mathbf{p})$, with the following nonlinear lifting function:

$$\nu(\mathbf{p}) = \mathcal{E}_\lambda^\mathsf{T}\boldsymbol{\ell}_\mathbf{p}.$$

The internal smoothing parameter λ is chosen small to ensure that matrix $\mathcal{E}_\lambda$ is well-conditioned.

Finally, the second derivative based smoother in (2) has the form:

$$\mathcal{B}^2(\mathsf{L}) = 8\pi\|\sqrt{\bar{\mathcal{E}}}\mathsf{L}\|_\mathcal{F}^2,$$

and we thus just choose Z such that $\mathsf{Z}^\mathsf{T}\mathsf{Z} = 8\pi\bar{\mathcal{E}}$ in the matrix form (3) to achieve the exact integral. Note that in practice, one does not need to compute Z since only $\mathsf{Z}^\mathsf{T}\mathsf{Z}$ is needed, *e.g.* for building the influence matrix T in (5).

Appendix 2: The LOOCV Lemma

This lemma states that replacing a target value with its prediction by the model estimated with this equation omitted does not change the result. In other words, adding equations to an LLS problem with as right-hand side the prediction by the model solving the initial problem, does not change the result.

Define $\mathsf{D}_j = \mathrm{I} - \mathrm{diag}(\mathbf{e}_j)$. Our goal is to show that (9) gives $\hat{\mathsf{L}}_{(j)}(\mu)$ as from (7). Following (5) we rewrite (7) as:

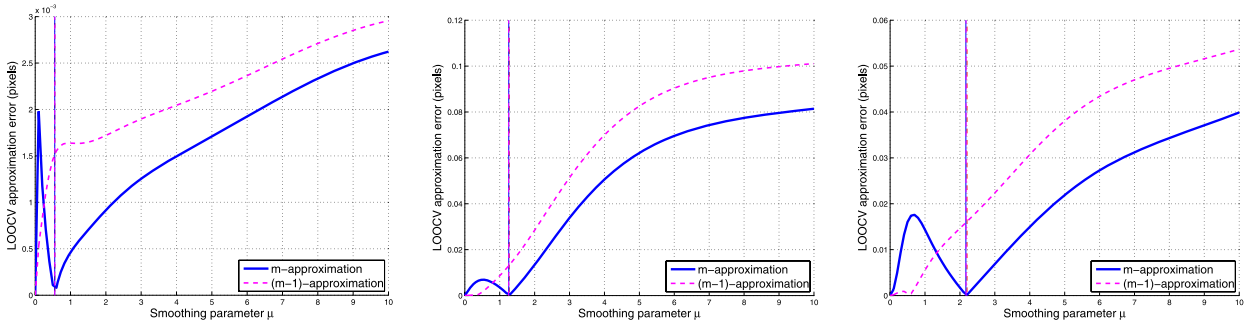
$$\begin{aligned} \hat{\mathsf{L}}_{(j)}(\mu) &= \arg\min_\mathsf{L} \left\| \begin{pmatrix} \mathsf{D}_j\mathsf{N} \\ \sqrt{m-1}\mu\mathsf{Z} \end{pmatrix}\mathsf{L} - \begin{pmatrix} \mathsf{D}_j\Xi \\ 0 \end{pmatrix} \right\|_\mathcal{F}^2 \\ &= (\mathsf{N}^\mathsf{T}\mathsf{D}_j\mathsf{N} + (m-1)\mu^2\mathsf{Z}^\mathsf{T}\mathsf{Z})^{-1}\mathsf{N}^\mathsf{T}\mathsf{D}_j\Xi, \end{aligned} \tag{17}$$

since $\mathsf{D}_j^\mathsf{T} = \mathsf{D}_j$ and $\mathsf{D}_j\mathsf{D}_j = \mathsf{D}_j$. We rewrite $\tilde{\Xi}^j$ from (8) as $\tilde{\Xi}^j = \mathsf{D}_j\Xi + (\mathrm{I} - \mathsf{D}_j)\mathsf{N}\hat{\mathsf{L}}_{(j)}$. We expand equation (9) by replacing T from (5) and $\tilde{\Xi}^j$ from just above, giving:

$$\begin{aligned} &\mathsf{T}((m-1)\mu^2)\tilde{\Xi}^j \\ &\quad = (\mathsf{N}^\mathsf{T}\mathsf{N} + (m-1)\mu^2\mathsf{Z}^\mathsf{T}\mathsf{Z})^{-1}\mathsf{N}^\mathsf{T} \\ &\qquad \cdot (\mathsf{D}_j\Xi + \mathsf{N}\hat{\mathsf{L}}_{(j)} - \mathsf{D}_j\mathsf{N}\hat{\mathsf{L}}_{(j)}). \end{aligned} \tag{18}$$

The second term rewrites to:

[4] While $\boldsymbol{\eta}_{\boldsymbol{\alpha},\lambda}$ has no obvious unit, $\boldsymbol{\alpha}$ in general has (*e.g.* pixels, meters).



The dishcloth dataset LOOCV ≈ 1.40 pixels | The paper sheet dataset LOOCV ≈ 7.27 pixels | The spine dataset LOOCV ≈ 12.60 pixels

Fig. 12 Zoom around the minimum onto the difference between the true LOOCV score from the greedy formula (6) and the m- and $(m-1)$-approximations. The *vertical lines* show the minima (the *dashed red line* is the true minima)

$$(\mathsf{N}^\mathsf{T}\mathsf{N}+(m-1)\mu^2\mathsf{Z}^\mathsf{T}\mathsf{Z})^{-1}\mathsf{N}^\mathsf{T}\mathsf{N}\hat{\mathsf{L}}_{(j)} = \hat{\mathsf{L}}_{(j)} - (m-1)\mu^2(\mathsf{N}^\mathsf{T}\mathsf{N}+(m-1)\mu^2\mathsf{Z}^\mathsf{T}\mathsf{Z})^{-1}\mathsf{Z}^\mathsf{T}\mathsf{Z}\hat{\mathsf{L}}_{(j)}. \tag{19}$$

Substituting in (18) gives:

$$\begin{aligned}&\mathsf{T}((m-1)\mu^2)\tilde{\Xi}^j \\ &= \hat{\mathsf{L}}_{(j)} + (\mathsf{N}^\mathsf{T}\mathsf{N}+(m-1)\mu^2\mathsf{Z}^\mathsf{T}\mathsf{Z})^{-1} \\ &\quad\cdot(\mathsf{N}^\mathsf{T}\mathsf{D}_j\Xi - (m-1)\mu^2\mathsf{Z}^\mathsf{T}\mathsf{Z}\hat{\mathsf{L}}_{(j)} - \mathsf{N}^\mathsf{T}\mathsf{D}_j\mathsf{N}\hat{\mathsf{L}}_{(j)}).\end{aligned} \tag{20}$$

This concludes the proof since the right-most factor vanishes, as shown below. Substitute $\hat{\mathsf{L}}_{(j)}$ from (17), this gives:

$$\begin{aligned}&\mathsf{N}^\mathsf{T}\mathsf{D}_j\Xi - (m-1)\mu^2\mathsf{Z}^\mathsf{T}\mathsf{Z}\hat{\mathsf{L}}_{(j)} - \mathsf{N}^\mathsf{T}\mathsf{D}_j\mathsf{N}\hat{\mathsf{L}}_{(j)} \\ &= \mathsf{N}^\mathsf{T}\mathsf{D}_j\Xi - ((m-1)\mu^2\mathsf{Z}^\mathsf{T}\mathsf{Z}+\mathsf{N}^\mathsf{T}\mathsf{D}_j\mathsf{N}) \\ &\quad\cdot(\mathsf{N}^\mathsf{T}\mathsf{D}_j\mathsf{N}+(m-1)\mu^2\mathsf{Z}^\mathsf{T}\mathsf{Z})^{-1}\mathsf{N}^\mathsf{T}\mathsf{D}_j\Xi \\ &= \mathsf{N}^\mathsf{T}\mathsf{D}_j\Xi - \mathsf{N}^\mathsf{T}\mathsf{D}_j\Xi \\ &= 0.\end{aligned}$$

Appendix 3: The Non-Iterative Approximation to LOOCV

In order to compare the m-approximation and the $(m-1)$-approximation we plot the difference between the true LOOCV score from (6) and each of the two approximations. This is shown for the three datasets in Fig. 12. As can be seen, both approximations are very close to the true LOOCV score. The m-approximation is in general better than the $(m-1)$-approximation, except at some points for $\mu < \hat{\mu}$. The minimum value of the m-approximation coincides with the true value at the true minimum, albeit that the location of the approximated minimum is slightly shifted from the true location. The $(m-1)$-approximation has a larger shift. Using the m-approximation is thus the best option, although the difference is very small. The order of magnitude on the location of the minimum is between 10^{-2} and 10^{-4}. The error on the minimum LOOCV score for the $(m-1)$-approximation is at 10^{-1} pixels.

References

1. Allen, D.M.: The relationship between variable selection and data augmentation and a method for prediction. Technometrics **16**, 125–127 (1974)
2. Bishop, C.M.: Neural Networks for Pattern Recognition. Oxford University Press, Oxford (1995)
3. Bookstein, F.L.: Principal warps: thin-plate splines and the decomposition of deformations. IEEE Trans. Pattern Anal. Mach. Intell. **11**(6), 567–585 (1989)
4. Bro-Nielsen, M., Gramkow, C.: Fast fluid registration of medical images. In: Visualization in Biomedical Imaging (1996)
5. Burrage, K., Williams, A., Erhel, J., Pohl, B.: The implementation of a generalized cross validation algorithm using deflation techniques for linear systems. Technical report, Seminar fur Angewandte Mathematik, July 1994
6. Christensen, G.E., He, J.: Consistent nonlinear elastic image registration. In: Workshop on Mathematical Methods in Biomedical Image Analysis (2001)
7. de Bruijne, M., Lund, M.T., Tankó, L.B., Pettersen, P.C., Nielsen, M.: Quantitative vertebral morphometry using neighbor-conditional shape models. Med. Image Anal. **11**(5), 503–512 (2007)
8. Duchon, J.: Interpolation des fonctions de deux variables suivant le principe de la flexion des plaques minces. RAIRO Anal. Numér. **10**, 5–12 (1976)
9. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Comput. Vis. Graph. Image Process. **24**(6), 381–395 (1981)
10. Fornefett, M., Rohr, K., Stiehl, H.S.: Radial basis functions with compact support for elastic registration of medical images. Image Vis. Comput. **19**(1), 87–96 (2001)
11. Genant, H., Wu, C., van Kuijk, C., Nevitt, M.: Vertebral fracture assessment using a semiquantitative technique. J. Bone Miner Res. **8**(9), 1137–1148 (1993)

12. Golub, G.H., von Matt, U.: Generalized cross-validation for large-scale problems. J. Comput. Graph. Stat. **6**(1), 1–34 (1997)
13. Hawkins, D.M., Yin, X.: A faster algorithm for ridge regression of reduced rank data. Comput. Stat. Data Anal. **40**(2), 253–262 (2002)
14. Kanatani, K.: Geometric information criterion for model selection. Int. J. Comput. Vis. **26**(3), 171–189 (1998)
15. Maintz, J.B.A., Viergever, M.A.: A survey of medical image registration. Med. Image Anal. **2**(1), 1–36 (1998)
16. Marsland, S., Twining, C.J., Taylor, C.J.: A minimum description length objective function for groupwise non-rigid image registration. In: Image and Vision Computing (2007)
17. Maybank, S., Sturm, P.: MDL, collineations and the fundamental matrix. In: British Machine Vision Conference (1999)
18. Modersitzki, J.: Numerical Methods for Image Registration. Oxford Science, Oxford (2004)
19. Nielsen, M., Johansen, P.: A PDE solution of Brownian warping. In: European Conference on Computer Vision (2004)
20. Nielsen, M., Johansen, P., Jackson, A.D., Lautrup, B.: Brownian warps: a least committed prior for non-rigid registration. In: Medical Image Computing and Computer-Assisted Intervention (2002)
21. Pilet, J., Lepetit, V., Fua, P.: Fast non-rigid surface detection, registration and realistic augmentation. Int. J. Comput. Vis. **76**(2), 109–122 (2008)
22. Poggio, T., Rifkin, R., Mukherjee, S., Niyogi, P.: General conditions for predictivity in learning theory. Nature **458**, 419–422 (2004)
23. Rifkin, R.M., Lippert, R.A.: Notes on regularized least squares. Technical Report MIT-CSAIL-TR-2007-025, MIT, May 2007
24. Rueckert, D., Sonoda, L.I., Hayes, C., Hill, D.L.G., Leach, M.O., Hawkes, D.J.: Nonrigid registration using free-form deformations: application to breast MR images. IEEE Trans. Med. Imaging **18**(8), 712–721 (1999)
25. Szeliski, R., Coughlan, J.: Spline-based image registration. Int. J. Comput. Vis. **22**(3), 199–218 (1997)
26. Torr, P.H.S.: Bayesian model estimation and selection for epipolar geometry and generic manifold fitting. Int. J. Comput. Vis. **50**(1), 27–45 (2002)
27. Wahba, G.: Splines Models for Observational Data. SIAM, Philadelphia (1990)
28. Wahba, G., Wold, S.: A completely automatic French curve: fitting spline functions by cross-validation. Commun. Stat. **4**, 1–17 (1975)

Adrien Bartoli is a permanent CNRS research scientist at the LASMEA laboratory in Clermont- Ferrand, France, since October 2004 and a visiting professor at DIKU in Copenhagen, Denmark for 2006-2009. Before that, he was a postdoctoral researcher at the University of Oxford, UK, in the Visual Geometry Group, under the supervision of Prof. Andrew Zisserman. He did his PhD in the Perception group, in Grenoble at INRIA, France, under the supervision of Prof. Peter Sturm and Prof. Radu Horaud. He received the 2004 INPG PhD Thesis prize and the 2007 best paper award at CORESA. Since September 2006, he is co-leading the ComSee research team. His main research interests are in Structure-from-Motion in rigid and non-rigid environments and machine learning within the field of computer vision.

6.2.3 Paper (BMVC'04) – *Direct Estimation of Non-Rigid Registrations*

Direct Estimation of Non-Rigid Registrations

Adrien Bartoli and **Andrew Zisserman**
Department of Engineering Science, University of Oxford
{Bartoli,az}@robots.ox.ac.uk

Abstract

Registering images of a deforming surface is a well-studied problem. Solutions include computing optic flow or estimating a parameterized motion model. In the case of optic flow it is necessary to include some regularization.

We propose an approach based on representing the induced transformation between images using Radial Basis Functions (RBF). The approach can be viewed as a direct, i.e. intensity-based, method, or equivalently, as a way of using RBFs as non-linear regularizers on the optic flow field.

The approach is demonstrated on several image sequences of deforming surfaces. It is shown that the computed registrations are sufficiently accurate to allow convincing augmentations of the images.

1 Introduction

The objective of this paper is registration of images of a non-rigidly deforming surface, such as a flag being gently blown by the wind. Our goal is to compute dense image transformations, mapping pixels from one image to corresponding pixels in the other images. This task is important in domains such as augmented reality and medical imaging. We are particularly interested in images of surfaces whose behaviour is difficult to explain using specific physics-based or learnt models, such as the motion of cloth in a skirt as a person walks.

One solution to this problem is to compute a regularized optic flow field, by minimizing an energy functional based on a data term, from e.g. the brightness constancy assumption, and a regularizer, encouraging smoothness of the flow field. A survey can be found in e.g. [12]. For example, Irani uses subspace constraints to regularize the optic flow field [8].

Another solution is to compute a parameterized image transformation. Two main approaches are possible: feature-based and direct methods. Feature-based methods first compute a set of matched features extracted from the images, such as corners or contours, and then use them to estimate the image transformations [15]. On the other hand, direct methods usually minimize an error function similar to the one used for computing the optic flow field, based on the brightness constancy assumption [9]. Both feature-based and direct methods have been shown to give good results when computing rigid transformations, such as affinities or homographies. However, feature-based methods may fail to capture the non-rigidities in the image areas where few features are present. In other words, in contrast to the rigid transformation case where e.g. an affinity is encapsulated in any three point correspondences, an arbitrary unknown number of correspondences might be needed to represent all the image deformations. For example, a low texture area might be subject to deformations, while no corner points might be found in this area. This is

one reason in favour of methods using the intensity information, i.e. optic flow and direct methods, for computing non-rigid transformations.

The method in [6] draws on the strength of both optic flow and direct methods. The idea is to learn linear motion models that are used as bases for the optic flow field.

Our objective is to avoid the need for learning the motion model. We propose to represent it using *Radial Basis Mappings* (RBM). Such transformations have been shown to be very effective in representing various image distortions induced by different kinds of non-rigidities. Radial Basis Functions (RBF) are non-linear functions defined by centres and coefficients, see e.g. [13]. An example of such a function is the Thin-Plate Spline [2, 4].

The traditional approach to estimating RBMs is feature-based, for the main reason that when landmarks are used as centres, then the coefficients of the transformation can be computed by a linear algorithm. In the Thin-Plate Spline case, the linear algorithm minimizes the 'bending energy' [4]. Chui *et al.* [5] propose an integrated feature-based approach to match points while computing a RBM.

On the other hand, very few attemps have been made towards computing RBMs by directly considering the image intensity, i.e. using a direct method, or equivalently using a RBM to regularize the optic flow field. Some progress has been made in this direction in the medical imaging community, but mainly assuming that the centres of the transformation are given by user-defined landmarks, see e.g. [10].

We propose a scheme for intensity-based estimation of RBMs. The novelty of our approach is that the number of centres is estimated on-the-fly, and directly depends on the degree of non-rigidity between the images. Our algorithm is based on estimating an affine transformation and adding centres until a registration criterion is met, while minimizing the registration residual. At each iteration, both the position of centres and the coefficients of the transformation are estimated. Our algorithm has therefore two main characteristics: it minimizes an intensity-based registration error and fits a parameterized non-rigid motion model, whose intrinsic complexity is tuned depending on the amount of non-rigid deformations. Note that an alternative method of choosing centres is given in [11].

We give some background in §2, and describe our method for the direct estimation of RBMs in §3. We propose an extension of the method to deal with image sequences in §4 and report experimental results in §5. Finally, we give our conclusions and discuss further work in §6.

Notation. Vectors and matrices are respectively typeset using bold and sans-serif fonts, e.g. $\mathbf{x}$ and A. We do not use homogeneous coordinates, i.e. image point coordinates are 2-vectors: $\mathbf{x}^\mathsf{T} = (x\ y)$, where $^\mathsf{T}$ is transposition. We denote as $\mathscr{I}$ the images, and $\mathscr{I}(\mathbf{x})$ the colour or gray-level at pixel $\mathbf{x}$. Index i is used for the frames ($i = 1,\ldots,n$), k for the centres ($k = 1,\ldots,l$) and j for point correspondences. The evaluation of an expression at some value is denoted as in $\mathsf{S}|_\mathbf{x}$. Matrix vectorization is written as in $\mathbf{a} = \mathrm{vect}(\mathsf{A})$. The identity matrix is denoted I and the zero matrix and vector as $\mathsf{0}$ and $\mathbf{0}$.

2 Background

In the following two sections, we describe direct methods and RBMs.

2.1 Direct Methods

We describe the principle of the direct, i.e. intensity-based, alignment of two images $\mathscr{I}$ and $\mathscr{I}'$, related by a point-to-point transformation. The main idea, common to most algorithms, is to minimize the sum of squared intensity differences between the aligned images, over the parameters of the transformation [9]. Let us consider the case of an affine transformation, and derive one of the possible algorithms based on Gauss-Newton. A more detailed formulation of the image alignment problem is described in [1]. In particular, a robust, coarse-to-fine framework is used to speed up convergence and prevent the algorithm getting trapped into local minima, see [3].

An affine transformation has 6 parameters and is modelled by a 2×3 matrix $\mathsf{A} = (\ \bar{\mathsf{A}}\ \ \mathbf{t})$, where $\bar{\mathsf{A}}$ is the leading 2×2 submatrix and $\mathbf{t}$ the last column. A point $\mathbf{x}$ is mapped from the first image to a point $\mathbf{x}'$ in the second image by $\mathbf{x}' = \bar{\mathsf{A}}\mathbf{x} + \mathbf{t}$. The transformation is parameterized by the 6-vector $\mathbf{a} = \text{vect}(\mathsf{A})$. The direct estimation of A consists in solving $\min_{\mathbf{a}} E(\mathbf{a})^2$, where E is an error function defined by the mean intensity difference induced by A between $\mathscr{I}$ and $\mathscr{I}'$ over a region of interest $\mathscr{X}$ with N pixels $E(\mathbf{a})^2 = \frac{1}{N}\sum_{\mathbf{x}\in\mathscr{X}} e(\mathbf{x},\mathbf{a})^2$. Each error term is given by $e(\mathbf{x},\mathbf{a})^2 = \big(\mathscr{I}(\mathbf{x}) - \mathscr{I}'(\bar{\mathsf{A}}\mathbf{x}+\mathbf{t})\big)^2$. We employ the Gauss-Newton algorithm [14] to solve this minimization problem, using the identity transformation as an initial solution.

2.2 Radial Basis Mappings

In their basic form, RBFs define a mapping from $\mathbb{R}^d$ to $\mathbb{R}$, where d is the dimension. A general description can be found in [13]. A 2D RBF f is defined by a $\mathbb{R}^2 \to \mathbb{R}$ *basis function* $\phi(\eta)$, *coefficients* represented by an $l+3$-vector $\mathbf{h}^\mathsf{T} = (w_1\ \cdots\ w_l\ \lambda\ \mu\ \nu)$ and a set of l *centres* $\mathbf{q}_k$ as:

$$f(\mathbf{x}) = \lambda x + \mu y + \nu + \sum_{k=1}^{l} w_k \phi(\|\mathbf{x} - \mathbf{q}_k\|). \tag{1}$$

It consists of a linear part, with parameters $(\lambda\ \mu\ \nu)$, and a non-linear part, a sum of l weighted terms with coefficients w_k of the basis function applied to the distance between $\mathbf{x}$ and the centre $\mathbf{q}_k$. Amongst others, the basis function can be chosen as a Gaussian $\phi(\eta) = \exp(-\eta^2/(2\sigma^2))/(2\pi\sigma^2)$ or as a Thin-Plate Spline $\phi(\eta) = \eta^2 \log(\eta)$.

Radial Basis Mappings as $\mathbb{R}^2 \to \mathbb{R}^2$ Radial Basis Functions. The usual way to construct a $\mathbb{R}^2 \to \mathbb{R}^2$ mapping m, i.e. a RBM, is to stack two $\mathbb{R}^2 \to \mathbb{R}$ RBFs f^x and f^y sharing their centres [4]:

$$m(\mathbf{x}) = \begin{pmatrix} f^x(\mathbf{x}) \\ f^y(\mathbf{x}) \end{pmatrix} = \bar{\mathsf{A}}\mathbf{x} + \mathbf{t} + \sum_{k=1}^{l} \begin{pmatrix} w_k^x \\ w_k^y \end{pmatrix} \phi(\|\mathbf{x} - \mathbf{q}_k\|), \tag{2}$$

where $\bar{\mathsf{A}}$ and $\mathbf{t}$ form an affine transformation given by:

$$\mathsf{A} = \begin{pmatrix} \lambda^x & \mu^x & \nu^x \\ \lambda^y & \mu^y & \nu^y \end{pmatrix}.$$

The coefficients are encapsulated in an $(l+3) \times 2$ matrix $\mathsf{h} = (\mathbf{h}^x\ \mathbf{h}^y)$, partitioned in a non-rigid and a rigid part as $\mathsf{h}^\mathsf{T} = (\mathsf{W}^\mathsf{T}\ \mathsf{A})$.

Computation from point correspondences. RBMs are often used to create dense smoot transformations interpolating point correspondences $\mathbf{n}_j \leftrightarrow \mathbf{n}'_j$ between two images. Points $\mathbf{n}_j$ are used as the centres of the transformation. By writing the interpolating conditions $\mathbf{n}'_j = m(\mathbf{n}_j)$ using equation (2), one obtains:

$$\mathbf{n}'_j = \bar{\mathsf{A}}\mathbf{n}_j + \mathbf{t} + \sum_{k=1}^{l} \begin{pmatrix} w_k^x \\ w_k^y \end{pmatrix} \phi(\|\mathbf{n}_j - \mathbf{n}_k\|),$$

which can be rewritten as a linear system $\mathsf{Ch} = \mathsf{D}$, see [2, 4].

The side conditions. The last three equations of the linear system $\mathsf{Ch} = \mathsf{D}$ are the 'side conditions'. They ensure that the computed transformation has square integrable second derivatives, i.e. that it smoothly behaves outside the region of interest $\mathscr{X}$. The side conditions are expressed between the coefficients $\mathbf{w}^x$ and $\mathbf{w}^y$ and the centres $\mathbf{n}_j$ as $\sum_{k=1}^{l} w_k = \sum_{k=1}^{l} w_k n_{k,1} = \sum_{k=1}^{l} w_k n_{k,2} = 0$, or, defining the r-th row of matrix P as $(\mathbf{n}_r^\mathsf{T}\ 1)$, in matrix form:

$$\mathsf{P}^\mathsf{T}\mathsf{W} = \mathbf{0}_{(3\times 1)}. \tag{3}$$

3 Direct Estimation of Radial Basis Mappings

We describe our approach for estimating a RBM by using a direct method.

3.1 Outline of the Approach

Constructing a direct method for estimating a RBM raises specific concerns since the number of centres l, as well as the centres $\mathbf{q}_k$ themselves and the coefficients h of the transformation have to be estimated. More formally, we formulate the problem as:

$$\min_{l,\alpha} E(\alpha)^2 \quad \text{such that } \mathsf{P}^\mathsf{T}\mathsf{W} = \mathbf{0}_{(3\times 1)},$$

where α encapsulates the set of parameters of the transformation as:

$$\alpha^\mathsf{T} = (q_1^x\ q_1^y\ \ldots\ q_l^x\ q_l^y\ w_1^x\ w_1^y\ \ldots\ w_l^x\ w_l^y\ a_1\ \ldots\ a_6).$$

A possible approach is to use a pre-defined set of centres, e.g. on a regular grid or corner points in the first image (as in a feature-based approach). This approach is not satisfactory. If too few centres are used, the mapping may fail to capture all the deformations, and if too many centres are used, then the computational cost might be extremely high.

Our approach is built on a *dynamic centre insertion* procedure. The idea is to iteratively insert new centres, i.e. add non-rigidity to the transformation, until it becomes satisfactory. The centres are inserted based on examining the error image, to detect where the mapping fails to provide a proper registration. At each iteration, a centre is inserted, and the error is minimized. The number of centres grows until the algorithm converges.

The initial number of centres is set to 4, since if less than 4 centres are present, the corresponding coefficients are constrained to be zero by the side-conditions (3). The algorithm is summarized in table 1 and illustrated in figure 1.

1. *Initialization:* use algorithm of §2.1 to obtain the parameters A of an initial affine transformation. Insert 4 centres as indicated in §3.3, set $l \leftarrow 4$ and setup α accordingly.
2. *Transformation refinement:* (§3.2) compute the parameters α' which minimize the error in intensity, starting from α.
3. *Convergence test:* (§3.4) if $E(\alpha) - E(\alpha') < \varepsilon$ then remove the last inserted centre(s) and stop.
4. *Parameters updating:* $\alpha \leftarrow \alpha'$.
5. *Centre insertion:* (§3.3) $l \leftarrow l + 1$. The new centre is $\mathbf{q}_l$, with corresponding coefficients $w_l^x \leftarrow 0$ and $w_l^y \leftarrow 0$.
6. *Main loop:* return to step 2.

Table 1: Direct alignment of two images $\mathscr{I}$ and $\mathscr{I}'$ based on estimating a RBM α with l centres $\mathbf{q}_k$ and coefficients h, using the Gauss-Newton algorithm and dynamic centre insertion. The iterations terminate when the decrease in the error becomes insignificant.

Figure 1: Registration of two images of a deforming Tshirt, shown top and bottom. (from left to right) Original images, the centres of the transformation and a grid mesh illustrating the mapping.

3.2 Refining the Transformation

We describe the alignment algorithm given the number l of centres. The algorithm draws on the previously described algorithm for the direct estimation of an affine transformation, see §2.1. The problem is to solve:

$$\min_{\alpha} E(\alpha)^2 \quad \text{such that } \mathsf{P}^{\mathsf{T}}\mathsf{W} = \mathbf{0}_{(3\times 1)}.$$

We use the Gauss-Newton algorithm. The differences with the affine transformation refinement algorithm are two-fold: the Jacobian matrix of the mapping is more complicated, and a special parameterization is used to enforce the side-conditions[1].

3.3 Dynamic Centre Insertion

A centre accounts for non-rigidity in the transformation around its position. Our strategy is to insert centres until the transformation gives a satisfactory registration of the two images, by looking at the error image $\mathscr{E}$, i.e. the difference between the first image, and the warped second image.

We proceed as follows. First, we compute a blurred version $\hat{\mathscr{I}}$ of the first image $\mathscr{I}$ using a Gaussian kernel. Second, using the current transformation, we warp the second image $\mathscr{I}'$ as $\hat{\mathscr{I}}'$, and blur using the same Gaussian kernel. Blurring the images before computing their difference is important to get rid of effects such as the partial pixel effect. The error image $\mathscr{E}$ (the absolute difference between $\hat{\mathscr{I}}$ and $\hat{\mathscr{I}}'$) indicates the regions where the registration is not satisfactory. One may simply insert the new centre where $\mathscr{E}$ attains a maximum. We perform an integration step by convolving with a Gaussian kernel, before looking for the maximum, to emphasize the regions where the registration is not satisfactory.

3.4 A Stopping Criterion

Inserting a centre increases the number of degrees of freedom of the transformation and reduce the registration error. One strategy to stop the iterations would be to penalize the registration error by the number of degrees of freedom, and stop the algorithm when the minimum of this function is reached, similar to a nested model selection algorithm, e.g. [16]. Another strategy is based on the fact that when the 'best' number of centres is reached, inserting a new centre will only produce a slight decrease in the error, proportional to the noise level and quantization / warping error. Thresholding the difference between two consecutive errors is consequently used as a stopping criterion.

4 Registering Multiple Images

The goal in this section is to exploit the two-image registration algorithm proposed above, to register a sequence of images. More precisely, we aim at computing the transformation between a reference image of the sequence and all other images. Without loss of generality, we choose the first image as the reference one. Denoting $f_{i_1 i_2}$ the transformation between image i_1 and image i_2, our goal is to compute f_{1i}, $i = 2, \ldots, n$.

We perform sequential processing: we first compute f_{12} starting from an identity transformation. Then, we compute f_{13} starting from f_{12} and so on.

One problem with this approach is that throughout the sequence, shadows might appear or disappear, meaning that the appearance of the surface might change. To overcome

[1] We use a QR decomposition-based subspace projection, as described in e.g. [7, §12.1.4].

this problem, we investigated two approaches. The first approach consists in updating the appearance of the reference frame while registering the frames. After having computed f_{12}, we use it to align $\mathscr{I}_2$ with the reference frame. This aligned image is called $\mathscr{I}_{21}$, and is used as an updated reference frame when computing f_{13} as the transformation between $\mathscr{I}_{21}$ and $\mathscr{I}_3$. Of course, this approach gets rid of appearance variations, but might drift since registration errors are accumulated through the process.

The second approach we propose is to apply a shadow mask. This mask is computed as the residual error image between the reference image $\mathscr{I}_1$ and $\mathscr{I}_{21}$. It is then applied to the reference image before registering it with $\mathscr{I}_3$. As in the previous approach, this gets rid of appearance variations, but might drift through the sequence. We have found however, that this approach is less likely to drift. This is due to the fact that only the shadow mask is updated, and not the reference frame itself.

5 Experimental Results

This section reports experimental results on simulated and real data.

5.1 Simulated Data

The goal of these experiments is to validate the algorithm and determinate conditions under which it converges. Starting from a first image, we generate a second image, that will be used in the algorithm. The second image is generated as illustrated on figure 2. First, a rigid transformation is applied to the first image: three points are selected and

Figure 2: The simulation setup: the original Tshirt image, after affine transformation, after full non-rigid transformation and after having applied a global illumination change and added random noise.

randomly perturbed to define an affine transformation. The direction of the perturbation is chosen at random, while its norm δ_R is user-defined. Second, a non-rigid transformation is applied: points on a regular grid are offset by a random perturbation as above, with norm δ_{NR}. Third, a global illumination change is applied, as well as a random Gaussian noise with variance σ, on the intensity of all pixels. We perform the experiments using the Tshirt image of figure 2 and an image from the Newspaper sequence, figure 5.

The default values of these three parameters are $\delta_R = 3$ pixels, $\delta_{NR} = 2$ pixels and $\sigma = 1$ (over 256 gray levels). We vary independently these parameters while measuring the residual error E at convergence. A residual error close to the noise level σ means that the algorithm successfully converged. The results are averages over 50 trials.

Figure 3 shows the results we obtained. Based on these, we can say that the convergence is independent of the rigid part δ_R, on the 0 to 10 pixels range. However, convergence is strongly affected by the non-rigid part δ_{NR}. It gracefully degrades until a break-point is reached, at roughly $\delta_{NR} = 6$ pixels for the Tshirt image and $\delta_{NR} = 4$ pixels

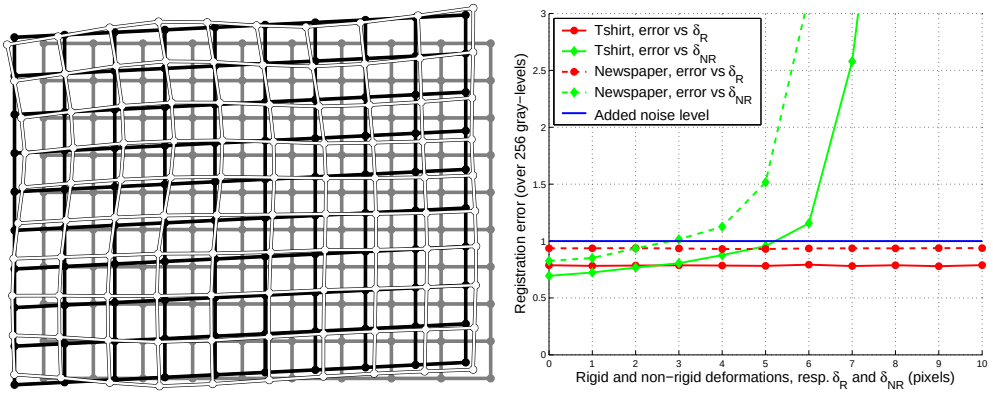


Figure 3: (left) A grid mesh illustrating the kind of simulated deformations. (right) Simulation results.

for the Newspaper image. Beyond these break-points, the algorithm does not converge to the right solution.

5.2 Real Data

Comparing grid-based and dynamic centre insertion approaches. We compare the traditional approach based on placing the centres at the nodes of a fixed, regular grid, and our dynamic centre insertion procedure. The results for the Tshirt images shown in figure 1 are displayed on figure 4. Our approach converged after 5 centres were introduced, with an error of 13.80 over 256 gray-levels.

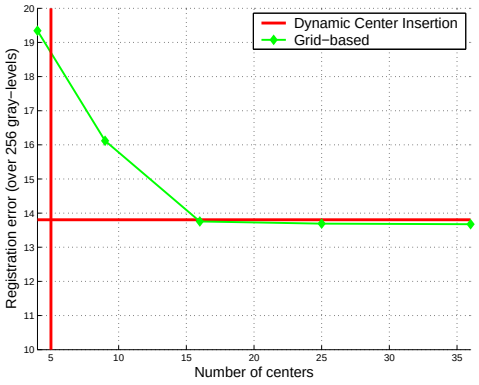


Figure 4: Comparison of our dynamic centre insertion and a fixed grid approach.

We observe that the fixed grid approach needs many more centres than ours to minimize the error function to a similar order of magnitude. More precisely, 16 centres are needed for the fixed grid approach to reach the same alignment.

The Newspaper sequence. We apply our algorithm to the 225-frame Newspaper sequence, shown in figure 5. This sequence was acquired by waving the newspaper, which non-rigidly deformed, in front of a hand-held digital camera. The movie shows that the surface is undergoing highly non-rigid deformations. We select a reference frame, shown on figure 5, to which all other frames are registered, using pair-wise non-rigid motion estimation, as described in §4.

Visually it is evident that the registration is good. The average intensity registration error over the sequence is 5.24 over 256 gray-levels, which is acceptable. The final column of figure 5 shows an augmented sequence: the original cartoon has been replaced by a new one. The video presentation associated with this paper clearly shows that visual deformations have been eliminated.

Figure 5: Registration results on the Newspaper sequence. (left) Original images. (middle) A mesh grid illustrating the computed mapping. (right) The cartoon on the reference image (indicated by a black frame) is replaced and mapped onto the other frames.

6 Conclusions and Further Work

We have proposed an intensity-based algorithm for the computation of Radial Basis Mappings, that can equivalently be viewed as a way to compute a regulatized optic flow field. The basic algorithm is intended to register pairs of views, and an extension for the registration of multiple views is proposed. Amongst the possible avenues for future research, experimenting with different, robust cost functions would be important, as well as computing super-resolution.

References

[1] S. Baker and I. Matthews. Lucas-Kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221–255, February 2004.

[2] S. Belongie, J. Malik, and J. Puzicha. Matching shapes. In *Proceedings of the 8th International Conference on Computer Vision, Vancouver, Canada*, pages 454–461, 2001.

[3] J.R. Bergen, P. Anandan, K.J. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. In *Proceedings of the 2nd European Conference on Computer Vision, Santa Margherita Ligure, Italy*, pages 237–252, 1992.

[4] F. L. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 11(6):567–585, June 1989.

[5] H. Chui and A. Rangarajan. A new point matching algorithm for non-rigid registration. *Computer Vision and Image Understanding*, 89(2):114–141, February 2003.

[6] D. Fleet, M. J. Black, Y. Yacoob, and A. D. Jepson. Design and use of linear models for image motion analysis. *International Journal of Computer Vision*, 36(3):171–193, 2000.

[7] G.H. Golub and C.F. van Loan. *Matrix Computation*. The Johns Hopkins University Press, Baltimore, 1989.

[8] M. Irani. Multi-frame optical flow estimation using subspace constraints. In *Proceedings of the 7th International Conference on Computer Vision, Kerkyra, Greece*, volume I, pages 626–633, September 1999.

[9] M. Irani and P. Anandan. About direct methods. In B. Triggs, A. Zisserman, and R. Szeliski, editors, *Vision Algorithms: Theory and Practice*, number 1883 in LNCS, pages 267–277, Corfu, Greece, July 1999. Springer-Verlag.

[10] H. J. Johnson and G. E. Christensen. Consistent landmark and intensity-based image registration. IEEE *Transactions on Medical Imaging*, 21(5):450–461, May 2002.

[11] S. Marsland and C. J. Twining. Constructing data-driven optimal representations for iterative pairwise non-rigid registration. In *Second International Workshop on Biometric Image Registration*, pages 50–60, 2003.

[12] A. Mitiche and P. Bouthemy. Computation and analysis of image motion: a synopsis of current problems and methods. *International Journal of Computer Vision*, 19(1):29–55, July 1996.

[13] M. J. L. Orr. Introduction to radial basis function networks. Technical report, University of Edinburgh, 1996.

[14] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. *Numerical Recipes in C*. Cambridge University Press, 1988.

[15] P. H. S. Torr and A. Zisserman. Feature based methods for structure and motion estimation. In W. Triggs, A. Zisserman, and R. Szeliski, editors, *Vision Algorithms: Theory and Practice*, number 1883 in LNCS, pages 278–295, Corfu, Greece, July 1999. Springer-Verlag.

[16] P.H.S. Torr. Geometric motion segmentation and model selection. *Philosophical Transactions of the Royal Society of London A*, 356:1321–1340, 1998.

6.2.4 Paper (BMVC'07) – *Feature-Driven Direct Non-Rigid Image Registration*

Feature-Driven Direct Non-Rigid Image Registration

V. Gay-Bellile[1,2] A. Bartoli[1] P. Sayd[2]
[1] LASMEA, CNRS / UBP, Clermont-Ferrand, France
[2] LCEI, CEA LIST, Saclay, France
Vincent.Gay-Bellile@univ-bpclermont.fr

Abstract

The registration problem for images of a deforming surface has been well studied. Parametric warps, for instance Free-Form Deformations and Radial Basis Functions such as Thin-Plate Spline, are often estimated using additive Gauss-Newton-like algorithms. The recently proposed compositional framework has been shown to be more efficient, but can not be directly applied to such non-groupwise warps.

We bring two contributions. First, we propose a Feature-Driven approach making possible the use of compositional algorithms for most parametric warps such as those mentioned above. This is demonstrated by an Inverse Compositional algorithm for Thin-Plate Spline warps. Second, we propose a piecewise linear learning approach to the local registration problem. Experimental results show that the basin of convergence is enlarged, computational cost reduced and alignment accuracy improved compared to previous methods.

1 Introduction

Registering images of a deforming surface is important for tasks such as video augmentation by texture editing, non-rigid Structure-from-Motion and deformation capture. This is a difficult problem since the appearance of imaged surfaces varies due to several phenomena such as camera pose, surface deformation, lighting and motion blur. Recovering a 3D surface, its deformations and the camera pose from a monocular video sequence is intrinsically ill-posed. While prior information can be used to disambiguate the problem, see *e.g.* [8, 13], it is common to avoid a full 3D model by using image-based deformation models, *e.g.* [2, 4, 6, 10]. TPS (Thin-Plate Splines) warps are one possible deformation model proposed in a landmark paper by Bookstein [4], that has been shown to effectively model a wide variety of image deformations in different contexts, including medical images. Recent work shows that TPS warps can be estimated with direct methods, *i.e.* by minimizing the intensity discrepancy between registered images [2, 10]. The Gauss-Newton algorithm with additive update of the parameters is usually used for conducting the minimization. Its main drawback is that the Hessian matrix must be recomputed and inverted at each iteration. More efficient solutions have been proposed by Baker *et al.* [1] based on a compositional update of the parameters, and lead to a constant Hessian matrix.

Most non-rigid warps do not from groups, preventing the use of compositional algorithms since they require to compose and possibly invert the warps. Despite several attempts to relax the groupwise assumption by various approximations [8, 12, 14], there is no simple solution in the literature.

This paper brings two main contributions. The first one is the *Feature-Driven* registration concept, allowing to devise compositional algorithms by relaxing the groupwise requirement for most non-rigid warps such as Radial Basis Functions (with *e.g.* the Thin-Plate Spline [4], a multiquadric [11] or the Wendland kernel function [7]). The main idea is to control the warp by a set of *driving features* (*e.g.* for Radial Basis Function warps, the target centers are used as driving features), and to act on these features directly for operations such as warp reversion and threading, approximating inversion and composition which are not guaranteed to exist or can not be easily computed. For instance, we extend the Inverse Compositional algorithm to TPS warps. The second contribution is an improvement of Cootes' linear learning approach [5] to local registration. Previous work assumes a linear relationship between the intensity discrepancy and the local parameter update [5, 9]. This assumption induces several practical problems, such as low alignment accuracy. We use a piecewise linear relationship for which a statistical map selection criterion is proposed. We experimentally show that it performs much better than most previous methods in terms of alignment accuracy, computational cost and enlarges the convergence basin. The combination of the Feature-Driven framework with learning-based local registration outperforms other algorithms for most experimental setups.

Notation. The images to be registered are written $\mathscr{I}_i$ with $i = 1,\ldots,n$. The template, *e.g.* the region of interest in the first image, is denoted $\mathscr{I}_0$. The parameterized warp is written $\mathscr{W}$. It depends on a parameter vector $\mathbf{u}_i$ for image $\mathscr{I}_i$ and maps a point $\mathbf{q}$ in the template to the corresponding point $\mathbf{q}_i$ in the i-th image: $\mathbf{q}_i = \mathscr{W}(\mathbf{q};\mathbf{u}_i)$. We write $\mathscr{R}$ the set of pixels of interest and $\text{vect}_{\mathscr{R}}(\mathscr{M})$ the operator that vectorizes the elements of $\mathscr{M}$ indicated in $\mathscr{R}$.

2 Previous Work

The registration of images of deformable surfaces has received a growing attention over the past decade. Direct registration consists in minimizing the pixel value discrepancy. Registration of an image sequence is posed as a set of nonlinear optimization problems, each of which estimating $\mathbf{u}_{i+1}$ using the registration $\mathbf{u}_i$ of the previous frame as an initial solution. The discrepancy function $\mathscr{C}$ is usually chosen as the two-norm of the difference $\mathscr{D}$ between the template and the current one, warped towards the template, *i.e.* $\mathscr{D}(\mathbf{q}) = \mathscr{I}_0(\mathbf{q}) - \mathscr{I}_{i+1}(\mathscr{W}(\mathbf{q};\mathbf{u}_{i+1}))$, giving: $\mathscr{C}(\mathbf{u}_{i+1}) = \sum_{\mathbf{q}\in\mathscr{R}} \|\mathscr{D}(\mathbf{q})\|^2$. Other choices are possible, such as Mutual Information.

Using an additive update of the parameter vector, *i.e.* $\mathbf{u}_{i+1} \leftarrow \mathbf{u}_{i+1} + \delta$, Gauss-Newton can be used in a straightforward manner for minimizing $\mathscr{C}$ or in conjunction with complexity tuning schemes as in [2, 10] for TPS warps. A second order approximation of $\mathscr{C}$, theoretically better than the Gauss-Newton one, is proposed in [3]. The major drawback of these methods is that the image gradient vector for each pixel in $\mathscr{R}$ must be recomputed at each iteration. This is the most expensive step of the process.

A major improvement was proposed by Baker and Matthews [1] through the Inverse Compositional algorithm. The key idea is to replace the additive update by the composition of the current warp $\mathscr{W}_{i+1}$ with the inverse $\tilde{\mathscr{W}}$ of the incremental warp: $\mathscr{W}_{i+1} \leftarrow \mathscr{W}_{i+1} \circ \tilde{\mathscr{W}}$. This leads to a constant Jacobian matrix and a constant Hessian matrix whose

inverse is thus pre-computed. This requires that the warp forms a group. In order to extend the approach to more involved models, several attempts have been made to relax the groupwise requirement. They are reviewed in §3.3. Previous work on learning-based registration are reviewed in §4.3.

3 Feature-Driven Registration

3.1 Feature-Driven Parametrization

The backbone of our approach is to represent the warp by a set of features in the current image, that we call *driving features*. These features have a fixed position in the template, depending on the type of warp that is being used. For RBF warps such as TPS warps, they can be placed anywhere, while for Free-Form Deformations, they must lie on a grid.

Henceforth, we assume that $\mathbf{u}_i$ contains the coordinates of the driving features in $\mathscr{I}_i$. In our implementation, we use TPS warps, whose Feature-Driven parameterization based on points is described in §A. The target centers of the TPS are used as driving features. The Feature-Driven concept and the registration algorithm we propose are generic in the sense that they are independent of the type of warp that is being used.

In this context, the warp is essentially seen as an interpolant between the driving features. There is obviously an infinite number of such warps. The best one depends on the nature of the observed surface. Loosely speaking, matching the driving features between two images is equivalent to defining a warp since the warp can be used to transfer the driving features from one image to the other, while conversely, the warp can be computed from the driving features.

The Feature-Driven framework has two main advantages. First, it often is better balanced to tune feature positions, expressed in pixels, than coefficient vectors that may be difficult to interpret, as for TPS warps. Second, it allows one to use the efficient compositional framework in a straightforward manner. Indeed, warp composition and inversion can not be directly done for non-groupwise warps. Representing image deformations by TPS warps or Free-Form Deformations is empirical. We propose empirical means for abstracting warp composition and inversion through their driving features, called threading and reversion respectively.

3.1.1 Threading Warps

Given two sets of driving features, $\mathbf{v}$ and $\mathbf{v}'$, we are looking for a third set $\mathbf{v}''$ defined such that threading the warps induced by $\mathbf{v}$ and $\mathbf{v}'$ results in the warp induced by $\mathbf{v}''$, as shown on figure 1(a). We propose a simple and computationally cheap way to do it, as opposed to previous work. This is possible thanks to the Feature-Driven parametrization. The idea is to apply the $\mathbf{v}'$ induced warp to the features in $\mathbf{v}$: the resulting set of features is $\mathbf{v}''$. This is written: $\mathbf{v}'' = \mathscr{W}(\mathbf{v};\mathbf{v}')$, where $\mathscr{W}$ is meant to be applied to each feature in $\mathbf{v}$. In the case of TPS warps, for which we use points as driving features, threading two warps is straightforward. It is also straightforward for all other kinds of RBF warps and for FFD warps.

3.1.2 Reverting Warps

Given a set $\mathbf{v}$ of driving features, we are looking for a set $\mathbf{v}'$, defined such that the warp induced by $\mathbf{v}'$ is the reversion of the one induced by $\mathbf{v}$, as illustrated on figure 1(b). As for

the threading, the Feature-Driven framework makes a very simple solution possible. The idea is that applying the $\mathbf{v}'$ induced warp to $\mathbf{v}$ gives $\mathbf{u}_0$, *i.e.* the fixed driving features in the template. This is written: $\mathscr{W}(\mathbf{v};\mathbf{v}') = \mathbf{u}_0$. This is straightforward for TPS warps. This amounts to solving an exactly determined linear system, the size of which is the number of driving features. For some classes of warps, $\mathscr{W}(\mathbf{v}';\mathbf{v}) = \mathbf{u}_0$ may be more practical to solve for $\mathbf{v}'$. Note that for all other kinds of RBF warps and FFD warps $\mathscr{W}(\mathbf{v};\mathbf{v}') = \mathbf{u}_0$ is to be preferred as well.

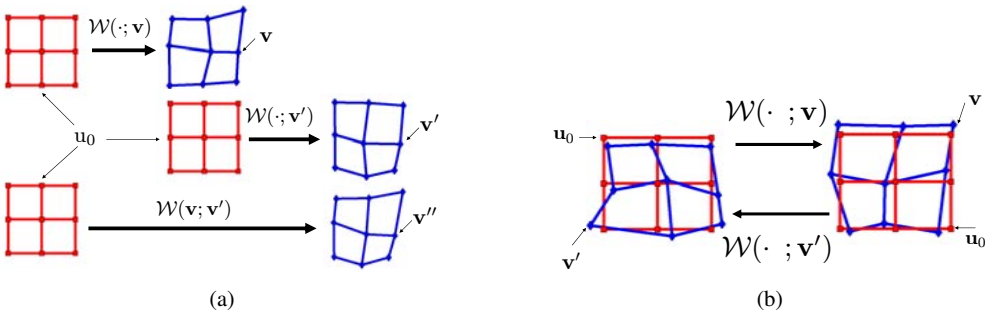


Figure 1: (a) The Feature-Driven warp threading process: $\mathbf{v}''$ is defined by $\mathbf{v}'' = \mathscr{W}(\mathbf{v};\mathbf{v}')$. (b) The Feature-Driven warp reversion process: $\mathbf{v}'$ is defined such that $\mathscr{W}(\mathbf{v};\mathbf{v}') = \mathbf{u}_0$.

3.2 Compositional Feature-Driven Alignment

Benefiting from the Feature-Driven parameterization properties, we extend the compositional algorithms to non-groupwise warps. The following three steps, illustrated on figure 2 are repeated until convergence:

- **Step 1: Warping.** The driving features $\mathbf{u}_{i+1}$ are used to warp the input image $\mathscr{I}_{i+1}$, thereby globally aligning it to the template: $\mathscr{I}_W(\mathbf{q}) = \mathscr{I}_{i+1}(\mathscr{W}(\mathbf{q};\mathbf{u}_{i+1}))$.
- **Step 2: Local alignment.** The driving features $\mathbf{u}$ are estimated in the warped image. This is described in §4. Note that for the Inverse Compositional algorithm, warp reversion is done at this step.
- **Step 3: Updating.** The current driving features $\mathbf{u}_{i+1}$ and those in the warped image $\mathbf{u}$ are combined by threading the warps to update the driving features $\mathbf{u}_{i+1}$ in the current image: $\mathbf{u}_{i+1} \leftarrow \mathscr{W}(\mathbf{u};\mathbf{u}_{i+1})$.

Note that previous work [8, 12, 14] requires a preliminary step before applying the update rule, as reviewed in the next section. In comparison, the Feature-Driven framework makes it naturally included into the third step.

Illumination changes are handled by normalizing the pixel value of the template and those of the warped image at each iteration.

3.3 Relation to Previous Work

Alternative approaches for non-groupwise warp composition as proposed in [8, 12, 14] consist in finding the best approximating warp for the pixels of interest in $\mathscr{R}$: $\mathbf{u}_{i+1} = \arg\min_{\mathbf{u}_{i+1}} \sum_{\mathbf{q}\in\mathscr{R}} \|\mathscr{W}_i(\tilde{\mathscr{W}}(\mathbf{q})) - \mathscr{W}_{i+1}(\mathbf{q})\|^2$. In [8, 12, 14] the warp is induced by a triangular mesh whose deformations are guided by a parameter vector. This minimization

problem is usually solved in two steps. First the vertices in the current image are computed using the assumption of local rigidity. They usually are not in accordance with a model instance in *e.g.* the case of 3D Morphable Models [8, 14]. Second, the parameter update is recovered by minimizing a prediction error, *i.e.* the distance between the updated vertices and those induced by the parameters. This last step may be time consuming since nonlinear optimization is required. Warp inversion is approximated with first order Taylor expansion in [12], while [14] draws on triangular meshes to avoid linearization. By comparison, our methods revert and thread warps in closed-form: they do not require optimization.

4 Local Registration

The efficiency of compositional algorithms depends on the local alignment step. For instance, the Inverse Compositional algorithm is efficient with the Gauss-Newton approximation for local alignment since this combination makes invariant the Hessian matrix. We show that learning-based local alignment fits in the Forward Compositional framework in a similar manner. Below, we propose an efficient learning-based alignment procedure. This is inspired by previous work modeling the relationship between the local increment δ and the intensity discrepancy $\mathbf{D}$ with an interaction matrix. Using a single interaction matrix has several drawbacks, as reviewed in §4.3. We propose to learn a series $\mathscr{F}_1 \dots \mathscr{F}_\kappa$ of interaction matrices, each of them covering a different range of displacement magnitudes. This forms a piecewise linear approximation to the true relationship. Each matrix thus defines a map. A statistical map selection procedure is learned in order to select the most appropriate matrix $\mathscr{F}_s$ given $\mathbf{D}$. The update vector is then given by: $\delta = \mathscr{F}_s\mathbf{D}$. Details are given below.

4.1 Learning an Interaction Matrix

An interaction matrix $\mathscr{F}$ is learned by generating artificially perturbed versions of the template $\mathscr{A}^j$.

Generating training data. The driving features in the template are disturbed from their rest position with randomly chosen direction and magnitude: $\mathbf{u}^j \leftarrow \mathbf{u}_0 + \delta\mathbf{u}^j$. The latter is clamped between a lower and an upper bound, determining the area of validity of the interaction matrix. Our Feature-Driven warp reversion process is used to warp the template.

Learning. The residual vector is computed from the pixels of interest in $\mathscr{R}$: $\mathbf{D}^j = \text{vect}_{\mathscr{R}}\left(\mathscr{I}_0 - \mathscr{A}^j\right)$. The training data are gathered in matrices $\mathscr{U} = \left(\delta\mathbf{u}^1|...|\delta\mathbf{u}^m\right)$ and $\mathscr{L} = \left(\mathbf{D}^1|...|\mathbf{D}^m\right)$. The interaction matrix $\mathscr{F}$ is computed by minimizing a Least Squares error in the image space, expressed in pixel value unit: $\mathscr{F} = \left(\mathscr{L}\mathscr{U}^\mathsf{T}(\mathscr{U}\mathscr{U}^\mathsf{T})^{-1}\right)^\dagger$. This is one of the two possibilities for learning the interaction matrix. The other possibility is dual. It minimizes an error in the parameter space, *i.e.* expressed in pixels. Experimental results show that the former gives better results, being in particular much more robust to noise.

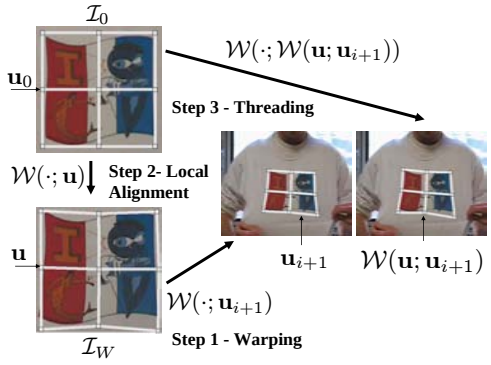


Figure 2: Compositional Feature-Driven Alignment .

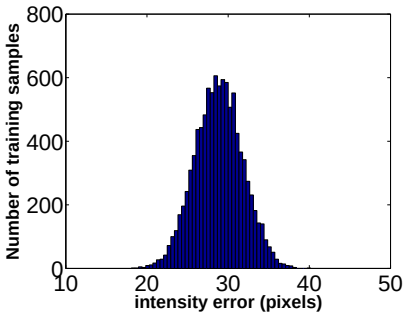


Figure 3: Distribution example of the intensity error magnitude for a perturbation interval: $[7\ldots13]$ pixels.

4.2 Statistical Map Selection

One issue with the piecewise linear model is to select the best interaction matrix at each iteration. Each of those indeed has a specific domain of validity in the displacement magnitude which unfortunately can not be determined prior to image alignment. We propose to learn a relationship between the intensity error magnitude and the best matrix to use. We express this relationship in terms of probabilities. The intensity error magnitude for a residual error vector $\mathbf{D}$ is defined as its RMS (Root Mean of Squares) $e(\mathbf{D})$. Note that $e^2(\mathbf{D}) \propto \mathscr{C}(\mathbf{u}_{i+1})$. Figure 3 shows that $P(\mathscr{F}_k|e(\mathbf{D}))$ closely follows a Gaussian distribution. The mean and variance of the learned intensity error magnitude are computed for each interaction matrix. Finding the most appropriate interaction matrix given the current intensity error is simply achieved by solving: $s = \arg\max_t P(\mathscr{F}_t|e(\mathbf{D}))$.

4.3 Relation to Previous Work

Learning approaches in the literature often assume that the relationship between the error image and the update parameters is linear [5, 9]. The drawback of those methods is that if the interaction matrix covers a large domain of deformation magnitudes, alignment accuracy is spoiled. On the other hand if the matrix is learned for small deformations only, the converge basin is dramatically reduced. Our piecewise linear relationship solves these issues.

Interaction matrices are valid only locally around the template parameters. Compositional algorithms are thus required, as in [9] for homographic warps. However in [5] the assumption is made that the domain where the linear relationship is valid covers the whole set of registrations. They thus apply the single interaction matrix around the current parameters, avoiding the warping and the composition steps. This does not appear to be a valid choice in practice. Our Feature-Driven framework naturally extends this approach to non-groupwise warps.

5 Experimental Results

We compare four algorithms in terms of convergence frequency, accuracy and convergence rate. Two classical algorithms:

- FA-GN: the Forward Additive Gauss-Newton approach used by [2, 10].
- FA-ESM: the Efficient Second Order registration algorithm [3], adapted to TPS warps.
Two algorithms we propose:
- IC-GN: the Feature-Driven Inverse Compositional registration of §3 with Gauss-Newton as local registration engine.
- FC-Le: the Feature-Driven Forward Compositional registration of §3, with local registration achieved through learning as we propose in §4.

5.1 Simulated Data

In order to assess the algorithms in different controlled conditions, we synthesize images from a template. The driving features are placed on a 3×3 grid, randomly perturbed with magnitude r. We add Gaussian noise, with variance $\sigma\%$ of the maximum greylevel value, to the warped image. We vary each of these parameters independently, using the following default values: $r = 2$ pixels and $\sigma = 1\%$. The noise variance upper bound is 10%, which corresponds to very noisy images. Estimated warps are scored by the mean Euclidean distance between the driving features which generated the warped image, and the estimated ones. Convergence to the right solution is declared if this score is lower than one pixel. The results are means over 500 trials.

Convergence frequency. This is the percentage of convergence to the right solution. Results are shown on figures 4(a) and 4(b). FC-Le has the largest convergence basin closely followed by FA-ESM. On the other hand, FC-Le has the worst performance against noise. However, it always converges for noise amplitude below 8% and converges at 95% for 10% noise which is beyond typical practical values. IC-GN has the smallest convergence basin.

Accuracy. This is measured as the mean residual error over the trials for which an algorithm converged. Results are shown on figures 4(c) and 4(d). The four algorithms are equivalent against displacement magnitude. Concerning noise amplitude, IC-GN and FC-Le are equivalent while FA-ESM is slightly worse and FA-GN clearly worse. For example, at 6% noise, the alignment errors of IC-GN and FC-Le are around 0.2 pixels, FA-ESM at about 0.25 pixels and FA-GN alignment error at 0.35 pixels.

Convergence rate. This is defined by the number of iterations required to converge. Results are shown on figures 4(e) and 4(f). The convergence rate of FC-LE and FA-ESM are almost constant against both displacement and noise amplitudes. However FC-Le does better, with a convergence rate kept below 10. FA-GN and IC-GN are efficient for small displacements, *i.e.* below 5 pixels. The convergence rate increases dramatically beyond this value for both of them. FA-GN is also inefficient for noise amplitude over 4%. This is explained by the fact that the FA-GN Jacobian matrix depends mainly on the gradient of the current image, onto which the noise is added.

5.2 Real Data

The four above described algorithms have been compared on several videos. For two of them (*paper*, *tshirt*), we show results on table 1 and registration visualization samples on figure 5. We measure the average and maximum intensity RMS along the video, computed on the pixels of interest and expressed in pixel value unit, the total number of iterations and the computational time, expressed in seconds. All algorithms have been implemented in

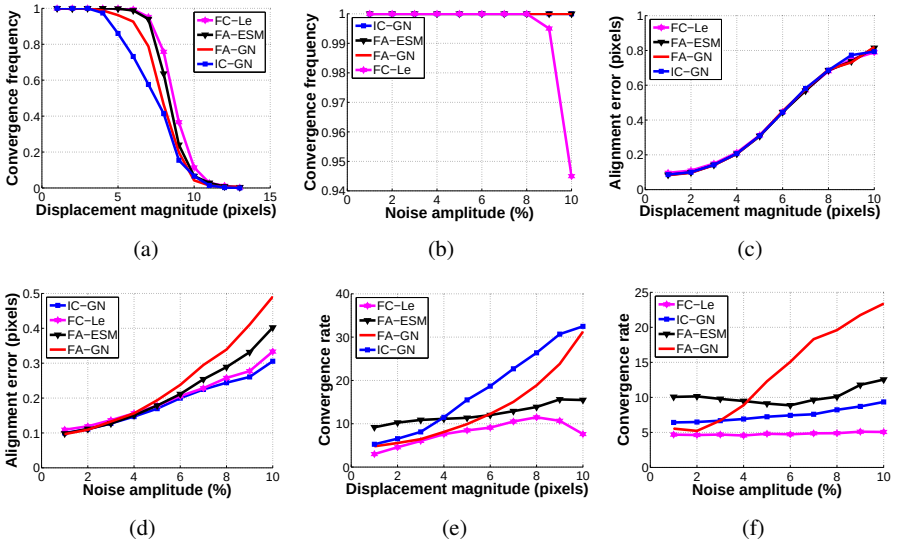


Figure 4: Comparison of the four algorithms in terms of convergence frequency (a) against displacement magnitude and (b) noise magnitude. Comparison of the four algorithms in terms of accuracy (c) against displacement magnitude and (d) noise magnitude. Comparison of the four algorithms in terms of convergence rate (e) against displacement magnitude and (f) noise magnitude.

Matlab. In practice, a great number of driving features with some amount of regularization are used, making the TPS both flexible and well-constrained. In order to illustrate the registration, we define a mesh on the template and transfer it to all other frames. Note that these meshes are different from the estimated driving features. The alignment differences between the four algorithms are visually undistinguishable, when they converge.

Summary. FA-GN is an accurate algorithm. It is however inefficient, especially for important displacements. FA-ESM has almost similar performances compared to FA-GN while being slightly more efficient. The Feature-Driven parametrization yields, via the proposed IC-GN and FC-Le algorithms, fast non rigid registration with TPS warps. While IC-GN looses effectiveness for high displacements, FC-Le has the best behavior. In fact, it is similar to FA-GN for accuracy while being 5 times faster on average and is equivalent or better than IC-GN and FA-ESM in terms of alignment accuracy, computational cost and has a larger convergence basin.

	Mean/max RMS		Iteration #		Total/mean time	
	tshirt	*paper*	*tshirt*	*paper*	*tshirt*	*paper*
FA-GN	8.7/13.6	**8.98/17.57**	9057	2422	2083/5.2	702/2.0
FA-ESM	9.2/14.7	10.22/20.49	3658	2473	877/2.2	708/2.0
IC-GN	9.7/15.8	Diverges	6231	Diverges	436/1.1	Diverges
FC-Le	**6.66/12.87**	9.44/19.4	**3309**	**1330**	**380/0.95**	**176/0.5**

Table 1: Results for the *tshirt* and *paper* videos. Bold indicates best performances.

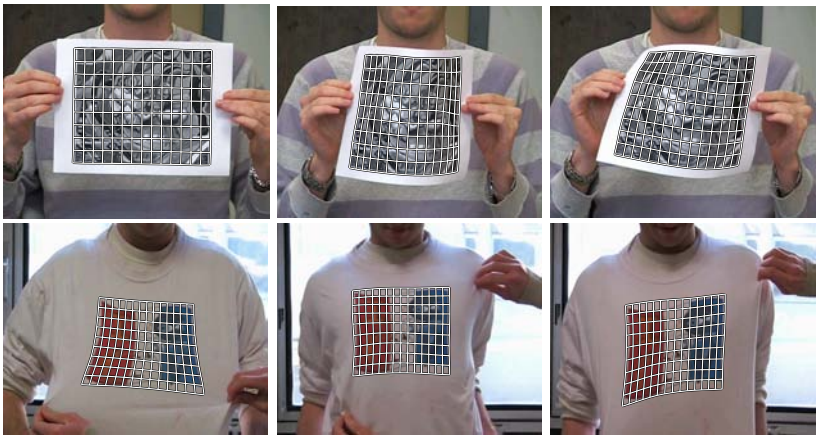


Figure 5: Registration results for FC-Le algorithm. Top: the *paper* video. Bottom: the *tshirt* video.

6 Conclusions

We addressed two important issues for the problem of non-rigid registration. First, we proposed the Feature-Driven framework, relaxing the groupwise requirement for using efficient compositional algorithms. Second, we proposed a statistically motivated piecewise linear local registration engine. Combining these two techniques results in an algorithm outperforming the other ones in terms of alignment accuracy, computational cost and having a larger convergence basin. Real-time surface registration is foreseen with this algorithm. We intend to extend the Feature-Driven approach to more complicated models, *e.g.* 3D Morphable Models [14]. This implies occlusion reasoning, that we intend to do through multiple overlapping patch registration and combination.

References

[1] S. Baker and I. Matthews. Lucas-Kanade 20 years on: A unifying framework. *IJCV*, 2004.

[2] A. Bartoli and A. Zisserman. Direct estimation of non-rigid registrations. In *BMVC*, 2004.

[3] S. Benhimane and E. Malis. Real-time image-based tracking of planes using efficient second-order minimization. In *IROS*, 2004.

[4] F. L. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. IEEE *PAMI*, 1989.

[5] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. In *ECCV*, 1998.

[6] T. F. Cootes, S. Marsland, C. J. Twining, K. Smith, and C. J. Taylor. Groupwise diffeomorphic non-rigid registration for automatic model building. In *ECCV*, 2004.

[7] Mike Fornefett, Karl Rohr, and H. Siegfried Stiehl. Elastic registration of medical images using radial basis functions with compact support. In *CVPR*, 1999.

[8] V. Gay-Bellile, M. Perriollat, A. Bartoli, and P. Sayd. Image registration by combining thin-plate splines with a 3D morphable model. In *ICIP*, 2006.

[9] F. Jurie and M. Dhome. Hyperplane approximation for template matching. IEEE *PAMI*, 2002.

[10] J. Lim and M.-H. Yang. A direct method for non-rigid motion with thin-plate spline. In *CVPR*, 2005.

[11] J. A. Little, D. L. G. Hill, and D. J. Hawkes. Deformations incorporating rigid structures. *CVIU*, 1997.

[12] I. Matthews and S. Baker. Active appearance models revisited. *IJCV*, 2004.

[13] J. Pilet, V. Lepetit, and P. Fua. Real-time non-rigid surface detection. In *CVPR*, 2005.

[14] S.Romdhani and T.Vetter. Efficient, robust and accurate fitting of a 3D morphable model. In *ICCV*, 2003

A Feature-Driven Thin-Plate Spline Warps

This parameterization is convenient for centers that remain still in the template. The TPS is an $\mathbb{R}^2 \rightarrow \mathbb{R}$ function driven by assigning target values α_k to l 2D centres $\mathbf{c}_k$ and enforcing several conditions: the TPS is the Radial Basis Function (RBF) that minimizes the integral bending energy. It is usually parameterized by an $l+3$ coefficient vector $\mathbf{h}_{\alpha,\lambda}^{\mathsf{T}} = (\ \mathbf{w}^{\mathsf{T}}\ \ \mathbf{a}^{\mathsf{T}})$ computed from the target vector α and a regularization parameter $\lambda \in \mathbb{R}^+$. The coefficients in $\mathbf{w}$ must satisfie $\mathsf{P}^{\mathsf{T}}\mathbf{w} = \mathbf{0}$, where the k-th row of P is $\left(\mathbf{c}_k^{\mathsf{T}}\ 1\right)$. These three 'side-conditions' ensure that the TPS has square integrable second derivatives. The TPS is defined by:

$$\omega(\mathbf{q}, \mathbf{h}_{\alpha,\lambda}) = \ell_{\mathbf{q}}^{\mathsf{T}} \mathbf{h}_{\alpha,\lambda}, \tag{1}$$

with $\ell_{\mathbf{q}}^{\mathsf{T}} = (\rho(d^2(\mathbf{q},\mathbf{c}_1))\ \cdots\ \rho(d^2(\mathbf{q},\mathbf{c}_l))\ \mathbf{q}^{\mathsf{T}}\ 1)$. Combining the equations obtained for all the l centres $\mathbf{c}_r$ with target values α_r in a single matrix equation gives:

$$\mathsf{K}_\lambda \mathbf{w} + \mathsf{P}\mathbf{a} = \alpha,\ K_{r,k} = \begin{cases} \lambda & r = k \\ \rho(d^2(\mathbf{c}_r, \mathbf{c}_k)) & r \neq k. \end{cases} \tag{2}$$

Adding $\lambda \mathrm{I}$ acts as a regularizer. Solving for $\mathbf{h}_{\alpha,\lambda}$ using the above equation and the side-conditions is the classical linear method for estimating the TPS coefficients due to Bookstein [4]. The coefficient vector $\mathbf{h}_{\alpha,\lambda}$ is a nonlinear function of the regularization parameter λ and a linear function of the target vector α.

We write $\mathbf{h}_{\alpha,\lambda} = \mathscr{E}_\lambda \alpha$, *i.e.* as a linear 'back-projection' of the target vector α. Matrix $\mathscr{E}_\lambda$ nonlinearly depends on λ. It is given from (2) as a function of K_λ and P by:

$$\mathscr{E}_\lambda = \begin{pmatrix} \mathsf{K}_\lambda^{-1}\left(\mathrm{I} - \mathsf{P}\left(\mathsf{P}^{\mathsf{T}}\mathsf{K}_\lambda^{-1}\mathsf{P}\right)^{-1}\mathsf{P}^{\mathsf{T}}\mathsf{K}_\lambda^{-1}\right) \\ \left(\mathsf{P}^{\mathsf{T}}\mathsf{K}_\lambda^{-1}\mathsf{P}\right)^{-1}\mathsf{P}^{\mathsf{T}}\mathsf{K}_\lambda^{-1} \end{pmatrix}.$$

This parameterization has the advantages to separate λ and α and introduces units[1]. The side-conditions are naturally enforced by this parameterization.

Incorporating this parameterization into the TPS (1) we obtain what we call the *feature-driven* parameterization for the TPS: $\tau(\mathbf{q};\alpha,\lambda) = \ell_{\mathbf{q}}^{\mathsf{T}} \mathscr{E}_\lambda \alpha$. Standard $\mathbb{R}^2 \rightarrow \mathbb{R}^2$ TPS-Warps are obtained by stacking two $\mathbb{R}^2 \rightarrow \mathbb{R}$ TPS sharing their centres and regularization parameter:

$$\mathscr{W}(\mathbf{q};\mathbf{u},\lambda) = (\tau(\mathbf{q};\alpha_x,\lambda)\ \tau(\mathbf{q};\alpha_y,\lambda))^{\mathsf{T}} = \left(\alpha_x \quad \alpha_y\right)^{\mathsf{T}} \mathscr{E}_\lambda^{\mathsf{T}} \ell_{\mathbf{q}}, \tag{3}$$

with $\mathbf{u}^{\mathsf{T}} = \left(\alpha_x^{\mathsf{T}}\ \alpha_y^{\mathsf{T}}\right)$. Notation $\mathscr{W}(\mathbf{q};\mathbf{u})$ is used for $\lambda = 0$.

[1] While $\mathbf{h}_{\alpha,\lambda}$ has no obvious unit, α in general has (*e.g.* pixels, meters).

6.2.5 Paper (ICCV'07) – *Direct Estimation of Non-Rigid Registrations with Image-Based Self-Occlusion Reasoning*

Direct Estimation of Non-Rigid Registrations with Image-Based Self-Occlusion Reasoning

V. Gay-Bellile[1,2] A. Bartoli[1] P. Sayd[2]
[1] LASMEA, Clermont-Ferrand, France [2] LCEI, CEA LIST, Saclay, France
Vincent.Gay-Bellile@univ-bpclermont.fr

Abstract

The registration problem for images of a deforming surface has been well studied. External occlusions are usually well-handled. In 2D image-based registration, self-occlusions are more challenging. Consequently, the surface is usually assumed to be only slightly self-occluding.

This paper is about image-based non-rigid registration with self-occlusion reasoning. A specific framework explicitly modeling self-occlusions is proposed. It is combined with an intensity-based, i.e. direct, data term for registration. Self-occlusions are detected as shrinking areas in the 2D warp.

Experimental results on several challenging datasets show that our approach successfully registers images with self-occlusions while effectively detecting the occluded regions.

1. Introduction

Registering monocular images of a deforming surface is important for tasks such as video augmentation by texture editing, non-rigid Structure-from-Motion and deformation capture. This is a difficult problem since the appearance of imaged surfaces varies due to several phenomena such as camera pose, surface deformation, lighting, motion blur and occlusions. The latter causes difficult issues in 2D image registration. They can be classified into external occlusions and self-occlusions. External occlusions are caused by an object entering the space between the camera and the surface of interest. Self-occlusions are due to the surface being bent in such a way that a part of it occludes another one. Deformation estimation in the presence of self-occlusions could be formulated in 3D, see *e.g.* [7]. However, recovering a 3D surface, its deformations and the camera pose from a monocular video sequence is intrinsically ill-posed. While prior information can be used to disambiguate the problem, see *e.g.* [3], it is common to avoid a full 3D model by using image-based deformation models, *e.g.* [1, 5, 8].

Previous work on 2D image registration, *e.g.* [1, 8, 11, 9, 13], usually deals with external and self-occlusions within an outlier rejection framework, *e.g.* the X84 rule, based on the data term. These methods handle external occlusions and very limited amounts of self-occlusion, as figure 1 illustrates.

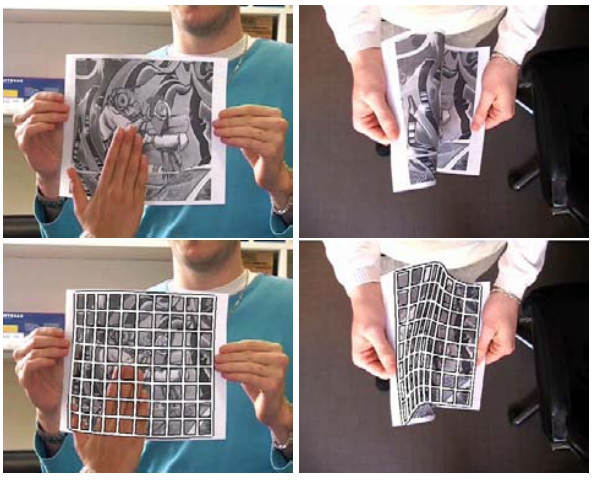


Figure 1. **Classical methods fail on extreme self-occlusions.** Example of image registration using a classical outlier rejection algorithm (see §5 for details). Left: success on an external occlusion. Right: failure on an extreme self-occlusion.

Self-occlusions thus have to be dealt with in a different manner. We propose a specific framework for non-rigid registration in spite of self-occlusions. The basic idea is to consider self-occluded pixels not as outliers, but as points at which the unknown warp is injective. This is implemented with two components: The warp is constrained to shrink rather than to 'fold' and self-occlusions are detected as the warp shrinkage areas. This allows accurate deformation estimation in the presence of self-occlusions. Experimental results demonstrating our framework are reported.

Roadmap. Previous work is reviewed in §2. Self-occlusion detection is explained in §3. Direct registration with self-occlusion reasoning is proposed in §4. Experimental results on real data are reported in §5. Finally, we give our conclusions and discuss future work in §6.

Notation. The images to be registered are written $\mathcal{I}_i$ with $i = 1, \dots, n$. The template, *e.g.* the occlusion-free region of interest in the first image, is denoted $\mathcal{I}_0$. The warp is written $\mathcal{W}$. It depends on a parameter vector $\mathbf{u}_i$ for image $\mathcal{I}_i$ and maps a points $\mathbf{q}$ in the template to the corresponding point $\mathbf{q}_i$ in image i as: $\mathbf{q}_i = \mathcal{W}(\mathbf{q}; \mathbf{u}_i)$.

We drop the image index i for clarity reasons in most of the paper. We write $\mathcal{R}$ the set of pixels of interest. We respectively denote $\mathbf{E}_c(\mathbf{d}; \mathbf{q}; \mathbf{u}), \mathbf{E}_l(\mathbf{d}; \mathbf{q}; \mathbf{u})$ and $\mathbf{E}_r(\mathbf{d}; \mathbf{q}; \mathbf{u})$ the central, left and right derivatives of the warp along direction $\mathbf{d} \in \mathbb{S}^1$, *e.g.* $\mathbf{E}_c(\mathbf{d}; \mathbf{q}; \mathbf{u}) = \frac{\mathcal{W}(\mathbf{q}+\epsilon\mathbf{d};\mathbf{u}) - \mathcal{W}(\mathbf{q}-\epsilon\mathbf{d};\mathbf{u})}{2\epsilon}$ ($\mathbb{S}^1$ is the unit circle). We denote v^x and v^y the x-component and the y-component of a two-dimensional vector $\mathbf{v}$.

2. Previous Work

Registering images of deformable surfaces has received a growing attention over the past few years. However, the self-occlusion issue is usually not explicitly tackled.

In [11], a feature-based non-rigid surface detection algorithm is proposed. The robustness of this approach to self-occlusions is not described clearly but experimental results show that only small self-occlusions are handled. Dealing with extreme self-occlusions such as those of figures 5 is very challenging. The number of features might not be large enough, in particular in the neighborhood of the self-occlusion boundary, to recover the correct warp. Direct non-rigid image registration generally yields more accurate deformation estimates. Classical methods, *e.g.* [1, 8] do not take self-occlusions into account. Some works explored this issue by using specific patterns. For example, Lin *et al.* [9] proposed a very robust algorithm for tracking a Near Regular Texture (NRT). A visibility map is used for dealing with external and self-occlusions. It is based on geometric and appearance properties of NRTs.

A method for retexturing non-rigid objects from a single viewpoint is proposed in [13]. This approach is based on texture and shading information. Deformable surface tracking is required to obtain texture coordinate before retexturing. The most impressive results are obtained with specific patterns. They allow to obtain very accurate texture coordinates even if some areas are slightly occluded.

A classical technique such as the z-buffer allows to predict the occluding contour when a 3D model is used. Ilic *et al.* [7] uses an implicit representation of 3D shapes. Occluding contours are computed as the solution of an ordinary differential equation. This allows one to recover the shape of a deforming object in monocular videos. This method can be applied to a restricted set of self-occlusions only: those for which the boundaries can not be *a priori* segmented defeat this method, see *e.g.* the one shown on figure 5.

To summarize, dealing with self-occlusions in purely 2D image-based registration is a very challenging problem that has not yet received a commonly agreed solution in the community.

3. Self-occlusion Detection Framework

3.1. Overview

The basic idea is to consider self-occluded pixels as points at which the unknown warp shrinks. Note that a natural warp behavior such as folding does not allow detection in 2D. We thus use the following three basic components. First, the warp is constrained not to fold onto itself but to shrink along the self-occlusion boundaries. Second, benefiting from this property, the warp is used to detect the self-occluded pixels. Third, these pixels are ignored in the data term in the error function. This is in contrast with classical methods which use the image data, *i.e.* the pixel intensities, for rejecting self-occluded parts as outliers.

In other words, the warp is constrained to be one-to-one in visible and externally occluded parts and many-to-one in self-occluded regions. The proposed approach, based on the *shrinking property* to detect the self-occluded pixels, outputs a binary self-occlusion map $\mathcal{H}(\mathbf{q}; \mathbf{u})$ with $\mathbf{q} \in \mathcal{R}$ and $\mathcal{H}(\mathbf{q}; \mathbf{u}) = 0$ if pixel $\mathbf{q}$ is occluded by the surface and $\mathcal{H}(\mathbf{q}; \mathbf{u}) = 1$, otherwise.

3.2. Preventing Foldings, Enforcing Shrinkage

A regularized warp naturally folds onto himself in case of extreme self-occlusions. Warp shrinkage is enforced by penalizing loops and folds, via the penalty term $\mathcal{E}_{\text{sh}}$. This behavior is required by the self-occlusion detection modules described in §3.3. Folds make the warp many-to-many in visible parts. These configurations are characterized by a variation in the sign of the partial derivatives of the warp along some direction. We note that diffeomorphic warps are proposed in [5]. They enforce one-to-one correspondences by preventing warp folds. Diffeomorphic warps are unadapted in our context since we model self-occluded areas by a many-to-one warp.
Our penalty is built via the following function:

$$\gamma(r) = \begin{cases} 0 & \text{if } r \geq 0 \\ r^2 & \text{otherwise.} \end{cases}$$

It is applied to the element-wise product between the left and the right derivatives of the warp evaluated at the points in $\mathcal{R}$ and integrated along direction $\mathbf{d} \in \mathbb{S}^1$. It allows to penalize those points for which the right and left derivatives

have opposite signs. It enforces the warp not to fold but rather to shrink along the self-occluded area. The shrinking constraint is given by:

$$\mathcal{E}_{\text{sh}}(\mathbf{u}) = \sum_{\mathbf{q}\in\mathcal{R}} \int_{\mathbf{d}\in\mathbb{S}^1} \sum_{c\in\{x,y\}} \gamma(\mathbf{E}_l^c(\mathbf{d},\mathbf{q},\mathbf{u})\mathbf{E}_r^c(\mathbf{d},\mathbf{q},\mathbf{u}))d\mathbf{d}.$$

In practice, the integral is discretized on a set of 8 directions.

3.3. Detecting Self-Occlusions

One consequence of the shrinking property is that for any self-occluded pixel $\mathbf{q}$, there exists at least one direction $\mathbf{d} \in \mathbb{S}^1$ such that the partial derivatives of the warp at $\mathbf{q}$ in direction $\mathbf{d}$ vanish. We thus define the self-occlusion map $\mathcal{H}(\mathbf{q};\mathbf{u})$ as:

$$\mathcal{H}(\mathbf{q};\mathbf{u}) = \begin{cases} 0 & \exists \mathbf{d} \in \mathbb{S}^1 \mid \|\mathbf{E}_c(\mathbf{d};\mathbf{q};\mathbf{u})\| < r \\ 1 & \text{otherwise.} \end{cases} \quad (1)$$

$\|\mathbf{E}_c(\mathbf{d};\mathbf{q};\mathbf{u})\| = 0$ implies that points $\mathcal{W}(\mathbf{q}+\epsilon\mathbf{d};\mathbf{u})$ and $\mathcal{W}(\mathbf{q}-\epsilon\mathbf{d};\mathbf{u})$ are identical. We fix r slightly over 0, *e.g.* 0.1, in order to tolerate noisy warps. In practice, the exhaustive search of directions $\mathbf{d}$ is replaced by a minimization problem:

$$\sigma_0 = \min_{\mathbf{d}\in\mathbb{S}^1} \|\mathbf{E}_c(\mathbf{d};\mathbf{q};\mathbf{u})\|^2.$$

A pixel $\mathbf{q}$ is labeled as self-occluded by comparing σ_0 with the threshold r. This minimization problem has a closed-form solution. Let $\mathcal{J}$ be the Jacobian matrix of $\mathcal{W}$ evaluated at $(\mathbf{q};\mathbf{u})$, *i.e.* $\mathcal{J} = \begin{pmatrix} \frac{\partial \mathcal{W}_x}{\partial x} & \frac{\partial \mathcal{W}_x}{\partial y} \\ \frac{\partial \mathcal{W}_y}{\partial x} & \frac{\partial \mathcal{W}_y}{\partial y} \end{pmatrix}$. We have $\mathbf{E}_c(\mathbf{d};\mathbf{q};\mathbf{u}) = \mathcal{J}\mathbf{d}$, and thus:

$$\sigma_0 = \min_{\mathbf{d}\in\mathbb{S}^1} \|\mathbf{E}_c(\mathbf{d};\mathbf{q};\mathbf{u})\|^2 = \min_{\mathbf{d}\in\mathbb{S}^1} \mathbf{d}^\mathsf{T}\mathcal{J}^\mathsf{T}\mathcal{J}\mathbf{d}.$$

Spectral decomposition of the symmetric matrix $\mathcal{O} = \mathcal{J}^\mathsf{T}\mathcal{J}$ gives:

$$\sigma_0 = \frac{1}{2}\left(\mathcal{O}_{1,1} + \mathcal{O}_{2,2} - \sqrt{(\mathcal{O}_{1,1}-\mathcal{O}_{2,2})^2 + 4\mathcal{O}_{1,2}^2}\right).$$

4. Image Registration with Self-Occlusions

4.1. The Warp

A regular grid is defined on the template. The warp is defined by $\mathcal{W}(\mathbf{q};\mathbf{u}) = \sum_{i=1}^4 \mathcal{B}_i(\mathbf{q})(\mathbf{s}_i + \mathbf{u}_i)$, where $\mathbf{s}_i$ are the mesh vertices in the vicinity of $\mathbf{q}$. $\mathcal{B}_i$ are interpolation coefficients. We use a bilinear interpolation. Other choices are possible, *e.g.* B-splines. The parameter vector $\mathbf{u}$ includes the displacement of the vertices $\mathbf{s}_i$: $\mathbf{u}^\mathsf{T} = (\mathbf{x}^\mathsf{T}\ \mathbf{y}^\mathsf{T})$. This type of parametric warps is known as Free-Form Deformations.

4.2. The Cost Function

Direct non-rigid registration algorithms usually use as discrepancy function $\mathcal{E}_{\text{data}}$ the two-norm of the difference $\mathcal{D}$ between the template and the current image, warped toward the template, *i.e.* $\mathcal{D}(\mathbf{q}) = \mathcal{I}_0(\mathbf{q}) - \mathcal{I}(\mathcal{W}(\mathbf{q};\mathbf{u}))$, giving:

$$\mathcal{E}_{\text{data}}(\mathbf{u}) = \sum_{\mathbf{q}\in\mathcal{R}} \mathcal{D}^2(\mathbf{q};\mathbf{u}).$$

More sophisticated data terms [10, 12] robust to noise or lighting variations can also be used.

For purely twodimensional registration, as is the case in this paper, smoothness constraints are used. These soft constraints can be implicitly incorporated in a parameterized warp as *e.g.* in Thin-Plate Spline warps. We use a simple regularization term $\mathcal{E}_{\text{reg}}$ that is added to the error function:

$$\sum_{\mathbf{q}\in\mathcal{R}} \mathcal{D}^2(\mathbf{q};\mathbf{u}) + \lambda_{\text{reg}}\mathcal{E}_{\text{reg}}(\mathbf{u}). \quad (2)$$

Denote $\mathcal{U}$ the displacement field reshaped on the mesh grid: $\mathbf{u} = \text{vect}(\mathcal{U})$. The regularization penalty on the displacement field $\mathcal{U}$ is a discrete approximation to the bending energy:

$$\mathcal{E}_{\text{reg}}(\mathbf{u}) = \int_{\mathcal{R}}\int_{\mathcal{R}} \left(\frac{\partial \mathcal{U}}{\partial^2 x}\right)^2 + 2\left(\frac{\partial \mathcal{U}}{\partial x \partial y}\right)^2 + \left(\frac{\partial \mathcal{U}}{\partial^2 y}\right)^2 dxdy.$$

Other choices are possible. The bending energy empirically appears to be well suited for the case of smooth surfaces.

Dealing with occlusions is usually done through a robust estimator in the data term:

$$\sum_{\mathbf{q}\in\mathcal{R}} \rho\left(\mathcal{D}^2(\mathbf{q})\right) + \lambda_{\text{reg}}\mathcal{E}_{\text{reg}}(\mathbf{u}). \quad (3)$$

As said above, it is not sufficient for self-occlusions. We rather weight each term by the self-occlusion map $\mathcal{H}$ described in §3. Simultaneously estimating the self-occlusion map and the displacement field is complicated and subject to many local minima. A two-step minimization scheme is used. First, the parameter vector is updated by using the current estimate of the self-occlusion map. Second, the latter is updated with equation (1). The associated error function is given by:

$$\sum_{\mathbf{q}\in\mathcal{R}} \mathcal{H}(\mathbf{q};\tilde{\mathbf{u}})\mathcal{D}^2(\mathbf{q};\mathbf{u}) + \lambda_{\text{reg}}\mathcal{E}_{\text{reg}}(\mathbf{u}), \quad (4)$$

where $\tilde{\mathbf{u}}$ is the current parameter estimate.

Finally, the shrinking penalty is added to the cost function. The global energy to be minimized is thus:

$$\mathcal{E}(\mathbf{u}) = \sum_{\mathbf{q}\in\mathcal{R}} \mathcal{H}(\mathbf{q};\tilde{\mathbf{u}})\mathcal{D}^2(\mathbf{q};\mathbf{u}) + \lambda_{\text{reg}}\mathcal{E}_{\text{reg}}(\mathbf{u}) + \qquad (5)$$
$$\lambda_{\text{sh}} \sum_{\mathbf{q}\in\mathcal{R}} \sum_{\mathbf{d}\in\mathcal{F}} \sum_{c\in\{x,y\}} \gamma(\mathbf{E}_l^c(\mathbf{d},\mathbf{q},\mathbf{u})\mathbf{E}_r^c(\mathbf{d},\mathbf{q},\mathbf{u})).$$

The global error function (5) is minimized using the Gauss-Newton algorithm. Note that the sparse structure of the Jacobian matrix, see figure 2, is taken into account in the minimization. A hierarchical approach [2] is used. It is mainly required at the disocclusion stage. The coarse-to-fine refinement step consists to propagate the displacement field and the self-occlusion map $\mathcal{H}$ using pyramid expansion operations [4]. The latter has to be binarized. All the non zero labels are fixed at one. This over-estimates the self-occlusion map, preventing misalignment at the boundaries.

We note that warp shrinking is the natural behavior of the warp when pixels vanish due to foreshortening under perspective projection. In these cases, the detection module is still valid. The shrinking constraint does not however activates since the warp does not fold. The proposed method naturally deals with these configurations.

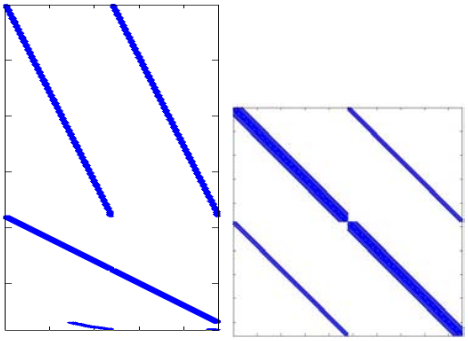


Figure 2. The Jacobian and Hessian (Gauss-Newton approximation) matrices have a very sparse structure. These example from the paper sequence of figure 4.

5. Experimental results

We tested our approach on several videos with different kinds of surfaces (paper, fabric). A 2-level pyramid is used. Various experiments show that the hierarchical framework is necessary to recover the warp at disocclusion but that additional levels do not significantly improve the registration. The image registration algorithm described in §4 takes few seconds per frame with our unoptimized Matlab code. A regular grid has been defined for visualization purposes. The latter is less dense than the one used to guide the warp.

The one-to-one constraint requirement. As said above, loops and folds are non-admissible warp configurations. In the absence of the shrinking penalty, they naturally appear when the surface is extremely self-occluded, as shown on figure 3. The self-occlusion detection is wrong since the warp derivative is not null for the whole self-occluded area. This defeats the registration process. Adding the shrinking constraint to the error function forces the warp not to fold but rather to shrink. Self-occlusion detection is then successful.

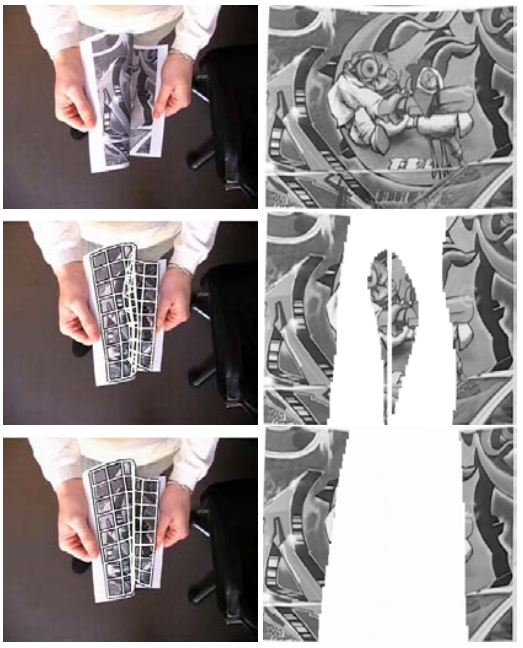
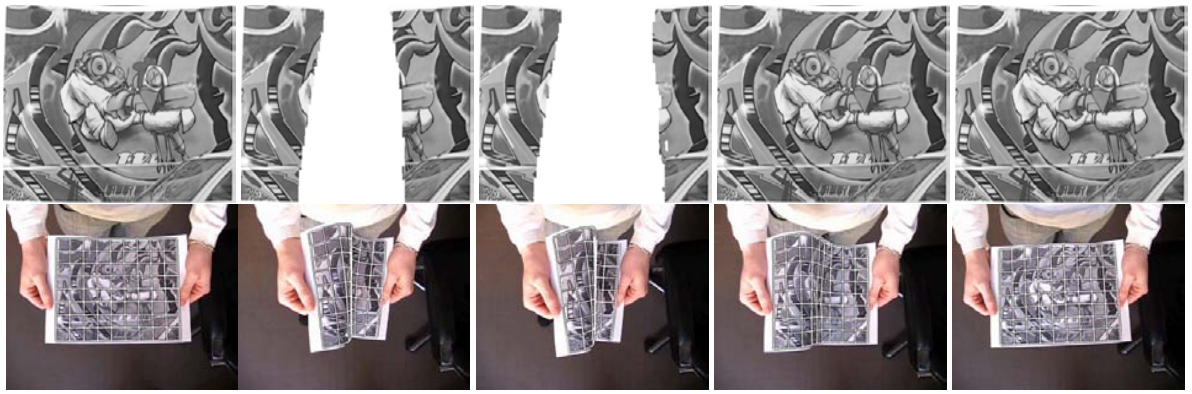


Figure 3. **Example of image registration with and without the shrinking constraint.** Self-occluded areas are shown in white. Top: the input $\mathcal{I}_i$ and the reference $\mathcal{I}_0$ images. Middle: erroneous registration results and self-occlusion detection without the shrinking constraint. Bottom: successful registration results and self-occlusion detection with the shrinking constraint.

The paper sequence. Figure 4 shows registration results on the first paper sequence. Registration of visible parts is accurate while the warp shrinks well in self-occluded regions. Recovering the true warp at disocclusion is done without misalignment.

Comparison with state-of-the-art. We compare our approach with a classical method for robust direct registration. It consists to minimize equation (3) with a hierarchical Gauss-Newton algorithm. Results are shown on figure 5. The Huber function [6] is employed as the robust kernel. Results in figure 1 are obtained with this procedure. Our approach successfully deals with self-occlusions while robust methods fail since they do not constrain the warp to shrink in self-occluded regions and yield inaccurate registration of



(#001) (#079) (#104) (#179) (#218)

Figure 4. **Registration results on the first paper sequence.** Top: the template $\mathcal{I}_0$ with detected self-occlusions shown in white. Bottom: a grid illustrating the warp on the current frame.

visible parts when the surface is extremely self-occluded. It is unlikely that they manage to keep track of the surface at the disocclusion satge. We note that a re-initialization procedure such as non-rigid surface detection [11] might be used when the registration is lost. The proposed approach allows to recover the warp at disocclusion without misalignment. A re-initialization procedure is not required unless the surface is totally occluded, contrarily to classical robust approaches.

Self-occluded boundaries. Self-occluded surface boundaries are particular cases. Classical approaches and the one we propose are both valid *i.e.* non-visible pixels are correctly rejected and visible ones are correctly registered. However, with our self-occlusion approach, the occlusion boundary is tracked as shown on figure 6. Minimizing equation (4) enforces the warp to shrink along the self-occlusion boundary. The self-occlusion boundary is lost with the classical approaches.

Other kinds of surfaces. Experiments on a fabric, figure 7, show that the specific framework we propose deal with self-occlusions for many kinds of surfaces.

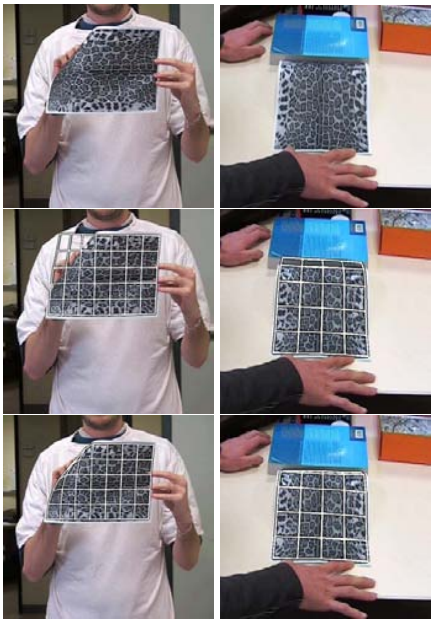


Figure 6. **Self-occlusion on the surface boundary.** Top: input images $\mathcal{I}_i$. Middle: registration results with a classical outlier rejection approach. Bottom: registration results with the proposed method.

6. Conclusion

We addressed the important and seldom explored issue of self-occlusions in non-rigid registration. A specific framework is proposed. The main idea is to constrain the warp to shrink in self-occluded regions while detecting them based on this property. Experimental results on real videos show that our approach successfully deals with extreme self-occlusions while classical robust methods fail. Future work will concentrate on improving the accuracy along the boundary of extreme self-occlusions.

References

[1] A. Bartoli and A. Zisserman. Direct estimation of non-rigid registrations. In *BMVC*, 2004.

[2] J. R. Bergen, P. Anandan, K. J. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. In *ECCV*, 1992.

[3] C. Bregler, A. Hertzmann, and H. Biermann. Recovering non-rigid 3D shape from image streams. In *CVPR*, 2000.

[4] P. J. Burt and E. H. Adelson. The laplacian pyramid as a

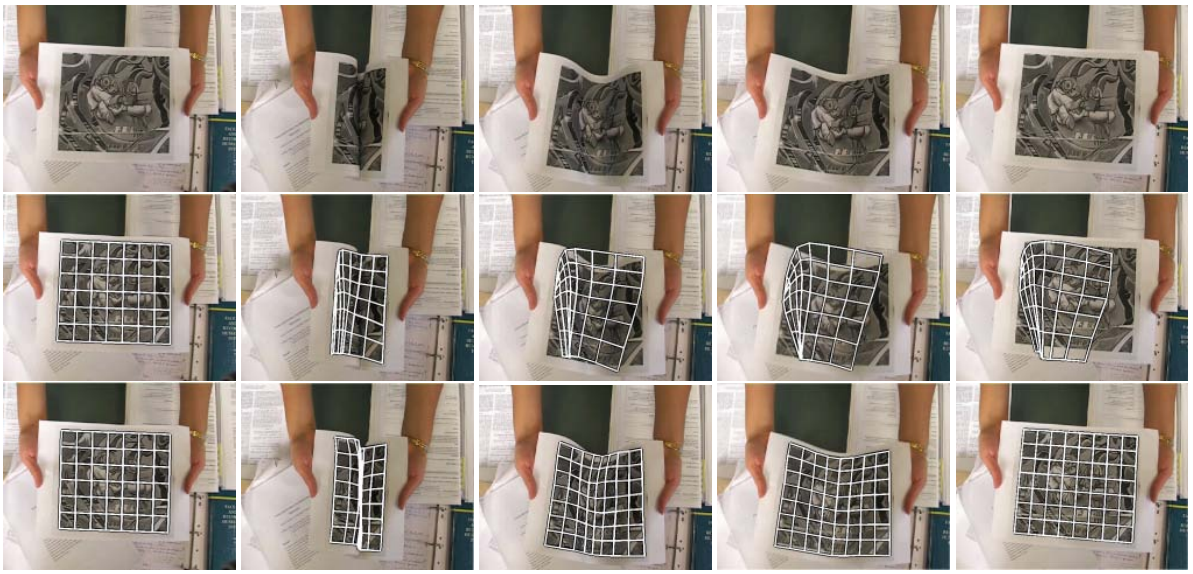
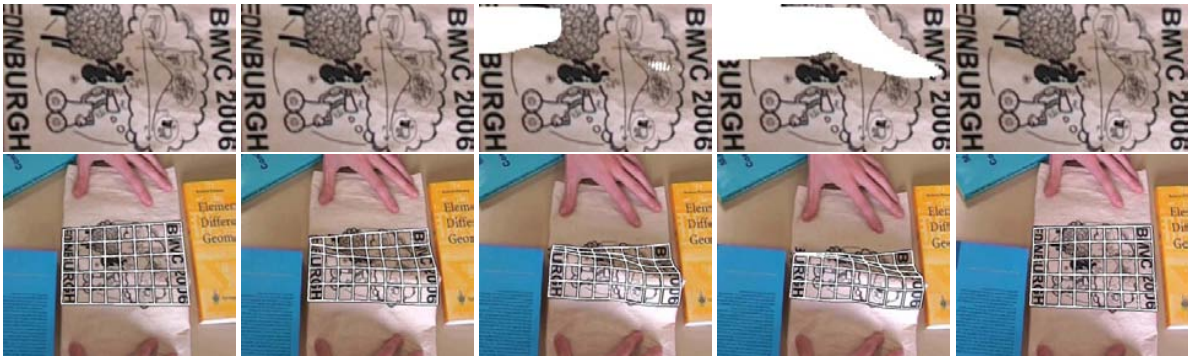


Figure 5. **Registration results on the second paper sequence.** Top: the input images $\mathcal{I}_i$. Middle: erroneous registration results with a classical outlier rejection approach. Bottom: successful registration results with the proposed method.



(#001) (#028) (#055) (#073) (#133)

Figure 7. **Registration results on the BMVC bag sequence.** Top: the template $\mathcal{I}_0$ with detected self-occlusions shown in white. Bottom: a grid illustrating the warp.

compact image code. *IEEE Transactions on Communications*, COM-31,4:532–540, 1983.

[5] T. F. Cootes, S. Marsland, C. J. Twining, K. Smith, and C. J. Taylor. Groupwise diffeomorphic non-rigid registration for automatic model building. In *ECCV*, 2004.

[6] G. D. Hager and P. N. Belhumeur. Efficient region tracking with parametric models of geometry and illumination. IEEE *TPAMI Intelligence*, 20(10):1025–1039, 1998.

[7] S. Ilic, M. Salzmann, and P. Fua. Implicit meshes for effective silhouette handling. *IJCV*, 72(2):159–178, 2007.

[8] J. Lim and M.-H. Yang. A direct method for non-rigid motion with Thin-Plate Spline. In *CVPR*, 2005.

[9] W. C. Lin and Y. Liu. Tracking dynamic near-regular textures under occlusions and rapid movements. In *ECCV*, 2006.

[10] N. Papenberg, A. Bruhn, T. Brox, S. Didas, and J. Weickert. Highly accurate optic flow computation with theoretically justified warping. *IJCV*, 67(2):141–158, 2006.

[11] J. Pilet, V. Lepetit, and P. Fua. Fast non-rigid surface detection, registration and realistic augmentation. *IJCV*, Published online January 2007.

[12] J. P. W. Pluim, J. B. A Maintz, and M. A. Viergever. Mutual-information-based registration of medical images: a survey. *IEEE Transactions on Medical Imaging*, 22:986–1004, 2003.

[13] R. White and D. Forsyth. Retexturing single views using texture and shading. In *ECCV*, 2006.

CHAPTER

7

STRUCTURE-FROM-MOTION FOR DEFORMABLE SCENES

7.1 A Single Camera

I17 **Augmenting Images of Non-Rigid Scenes Using Point and Curve Correspondences** §7.1.1
A. Bartoli, E. von Tunzelmann and A. Zisserman
CVPR'04 - IEEE *Int'l Conf. on Computer Vision and Pattern Recognition*, Washington, DC, USA, p. 699-706, vol. I, June 2004

I22 **A Batch Algorithm For Implicit Non-Rigid Shape and Motion Recovery** §7.1.2
A. Bartoli and S. Olsen
WDV'05 - *Workshop on Dynamical Vision* at ICCV'05, Beijing, China, October 2005
Other version: [N08]

J10 **Implicit Non-Rigid Structure-from-Motion with Priors** §7.1.3
S. Olsen and A. Bartoli
Journal of Mathematical Imaging and Vision, special issue: tribute to Peter Johansen, accepted December 2007
Previous version: [I42]

I50 **Coarse-to-Fine Low-Rank Structure-from-Motion** §7.1.4
A. Bartoli, V. Gay-Bellile, U. Castellani, J. Peyras, S. Olsen and P. Sayd
CVPR'08 - IEEE *Int'l Conf. on Computer Vision and Pattern Recognition*, Anchorage, Alaska, USA, June 2008

I29 **Image Registration by Combining Thin-Plate Splines with a 3D Morphable Model** §7.1.5
V. Gay-Bellile, M. Perriollat, A. Bartoli and P. Sayd
ICIP'06 - *Int'l Conf. on Image Processing*, Atlanta, GA, USA, October 2006

7.1.1 Paper (CVPR'04) – *Augmenting Images of Non-Rigid Scenes Using Point and Curve Correspondences*

Augmenting Images of Non-Rigid Scenes Using Point and Curve Correspondences

Adrien Bartoli, **Eugénie von Tunzelmann** and **Andrew Zisserman**
Department of Engineering Science, University of Oxford
{Bartoli,az}@robots.ox.ac.uk

Abstract

Our goal is to augment images of non-rigid scenes coming from single-camera footage. We do not assume any a priori *information about the scene being viewed, such as for example a parameterized 3D model or the motion of the camera. One possible solution is to use non-rigid factorization of points, from which a dense interpolating function modeled by a thin-plane spline can be computed. However, in many cases, point correspondences fail to capture precisely all the deformations occurring in the scene. Examples include the eyebrows or the lips when augmenting sequences of a face. Such deformations can be captured by tracking curves, but then point correspondences are not obtained directly due to the aperture problem.*

We propose an integrated method for non-rigid factorization and thin-plate spline interpolant estimation using point and curve correspondences over multiple views. The main novelties lie in the introduction of curves into the non-rigid factorization framework and in a direct global solution for the registration map, obtained by minimizing the registration error over all points and curves while taking all the images into account. The parameters of the registration map are set using cross-validation. The fidelity of the map is demonstrated by augmenting video footage undergoing various types of deformation.

1. Introduction

Augmenting images coming from pre-shot monocular footage is a major issue in the domain of special effects. One approach is to compute a dense mapping function between a reference frame and all other frames of the sequence. The augmentation is then performed by the user only on the reference image and is transferred automatically to the rest of the sequence.

When the observed scene is rigid, the problem is equivalent to computing a 3D surface. Techniques such as dense matching [10] and multiple-view stereo [7], or interpolation from a set of feature correspondences, can be employed to estimate this surface.

However, the assumption of rigidity is violated in many cases of interest, such as faces changing expression or clothing deforming. The problem is then particularly challenging because a different shape is observed in each image. A major step forwards for such cases was made by Torresani *et al.* [13, 14] and Brand [3]. Building on the work of [1, 6], they developed and demonstrated factorization of non-rigid scenes, where the non-rigidity was represented as a linear combination of basis shapes.

Figure 1. (left) Close-up of two frames of a 40 frame sequence from the film 'Run Lola Run'. (middle) the frames overlaid with real point and curve correspondences. (right) the frames augmented with a logo on the forehead. The top row shows the reference frame.

Unfortunately, for real sequences of smooth surfaces there is often insufficient texture to establish point correspondences which capture accurately all the deformations.

However, curves encapsulating such deformations *are* often available. In this paper we give a solution to this problem. We describe a method for computing a dense interpolating function between images of a non-rigid motion sequence. The mapping function is computed from both point *and* curve correspondences, using the non-rigid factorization framework. An example of using the computed mapping is shown in figure 1 (the sequence is shown in figure 5). In this case there are only a few areas where points can be reliably tracked, e.g. the corner of the eyes, but there are several curves (the hairline, the eyebrows) which may be used to determine the mapping.

In outline our method proceeds by computing an initial mapping based on point correspondences. From this mapping, we introduce virtual point correspondences, chosen such that the registration error of curves is minimized. Finally, the mapping parameters and the virtual points positions are globally tuned by minimizing a non-linear error function, and the process is iterated until the registration of curves is satisfactory. Thin-plate spline image interpolants are used for the mapping. The non-rigid factorization, based on a low-rank shape assumption, mainly serves to enforce global shape consistency by defining a low-rank subspace for reconstructing the virtual points.

This paper is organized as follows. §2 gives preliminaries and background. §3 describes our approach and §4 reports experimental results. Finally, §5 gives conclusions and extensions.

2. Preliminaries and Background

2.1. Notation

Without loss of generality, we use the first image as the reference image, i.e. the image that we will relate to all other ones to perform augmentation. We do not use homogeneous coordinates, i.e. image point coordinates are 2-vectors. Vectors and matrices are respectively typeset using bold and sans-serif fonts, e.g. $\mathbf{x}$ and X. We denote *real points* as $\mathbf{x}_{ij}$, for the j-th ($j = 1, \ldots, m$) point in the i-th view ($i = 1, \ldots, n$). Index $k = 1, \ldots, l$ is used for curves, i.e. $\mathcal{C}_{ik}$ is the i-th image of the k-th curve. The *virtual points* are denoted $\mathbf{y}_{ikp}$ where $p = 1, \ldots, d_k$. Points on a curve are obtained by $\mathcal{C}_{ik}(t)$, where $t \in [0, 1]$ is a parameter.

2.2. Non-Rigid Factorization

We sketch the assumptions and ideas of non-rigid factorization and the way we use it. We denote X the $(2n \times m)$ matrix defined as:

$$\mathsf{X} = \begin{pmatrix} \mathbf{x}_{11} & \mathbf{x}_{12} & \cdots & \mathbf{x}_{1m} \\ \mathbf{x}_{21} & \mathbf{x}_{22} & \cdots & \mathbf{x}_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{x}_{n1} & \mathbf{x}_{n2} & \cdots & \mathbf{x}_{nm} \end{pmatrix}.$$

Non-rigid factorization assumes the affine camera model and that centroids have been subtracted in each image. In the rigid case, matrix X has rank $r = 3$ if the scene is 3D [12], $r = 2$ if the scene is 2D. In the non-rigid case, it has been shown that X can still be of low rank, e.g. [14]. This result is obtained by assuming that the observed 3D shapes are linear combinations of basis shapes. For example, if a 3D scene is observed, and is a linear combination over 2 basis shapes, then $r = 6$ since matrix X factorizes as:

$$\mathsf{X} = \underbrace{\begin{pmatrix} \lambda_{11}\mathsf{P}_1 & \lambda_{12}\mathsf{P}_1 \\ \lambda_{21}\mathsf{P}_2 & \lambda_{22}\mathsf{P}_2 \\ \vdots & \vdots \\ \lambda_{n1}\mathsf{P}_n & \lambda_{n2}\mathsf{P}_n \end{pmatrix}}_{\mathsf{M}} \underbrace{\begin{pmatrix} \mathbf{B}_{11} & \mathbf{B}_{12} & \cdots & \mathbf{B}_{1m} \\ \mathbf{B}_{21} & \mathbf{B}_{22} & \cdots & \mathbf{B}_{2m} \end{pmatrix}}_{\mathsf{S}},$$

where P_i are the leading (2×3) submatrices of affine projection matrices, the λ_{iu} are the weights of the linear combinations and $\mathbf{B}_{uj}$ is the u-th 3D basis point for the j-th point. We call M the *motion matrix* and S the *shape matrix*. This factorization can be achieved by Singular Value Decomposition $\mathsf{X} = \mathsf{U}\Sigma\mathsf{V}^\mathsf{T}$. The motion and shape matrices are given by the r first columns of $\mathsf{U}\sqrt{\Sigma}$ and the r first rows of $\sqrt{\Sigma}\mathsf{V}^\mathsf{T}$ respectively, which, in the presence of noise, corresponds to nullifying all but the r first singular values of X. Recomposing the motion and shape matrices gives the reprojected points $\hat{\mathsf{X}} = \mathsf{MS}$. Each column $\mathbf{s}$ of the shape matrix S corresponds to a physical point. More precisely, it is a *basis point* in $\mathbb{R}^r$: it encapsulates the low-rank information which, coupled to the motion matrix M, enables all images of the point to be predicted. While the basis shapes are obtained in a straightforward manner from the shape matrix, the weights and the projection matrices are difficult to extract properly from the motion matrix. Moreover, it is sometimes difficult to choose between 2D and 3D deforming scenes. Finally, the non-rigidity induces additional ambiguities in the reconstruction, see [9], and bundle adjustment needs therefore strong regularization terms.

Our algorithm relies on non-rigid factorization. However, it does not need the motion matrix to be explicitly decomposed into projection matrices and weight factors, nor the shape matrix to be cast as sets of 2D or 3D points. This avoids making an explicit choice between a 2D or 3D scene.

In practice, there is the issue of choosing the rank r. Previous solutions to this problem include [9] (using model selection), and [6] (by examining the singular values). In the experimental section §4, we use cross-validation measures to choose r.

2.3. Thin-Plate Spline Image Interpolants

A $\mathbb{R}^2 \rightarrow \mathbb{R}^2$ radial basis function has the form $f(\mathbf{x}) = (f^x(\mathbf{x}) \;\; f^y(\mathbf{x}))^\mathsf{T}$, where, if we define $* \in \{x, y\}$:

$$f^*(\mathbf{x}) = \alpha^* + \gamma^* x + \delta^* y + \sum_{j=1}^{m} w_j^* E(\|\mathbf{x}_j - \mathbf{x}\|). \quad (1)$$

A 2D thin-plate spline is obtained by choosing the basis functions as $E(\rho) = \rho^2 \log \rho$. This choice minimizes the *bending energy* [2]:

$$\Phi(f) = \iint \left(\left(\frac{\partial^2 f}{\partial x^2} \right)^2 + 2 \left(\frac{\partial^2 f}{\partial x \partial y} \right)^2 + \left(\frac{\partial^2 f}{\partial y^2} \right)^2 \right) dx dy.$$

Interpolating or approximating mappings can be constructed from m point correspondences $\mathbf{x}_j \leftrightarrow \mathbf{x}'_j$. The parameters of a mapping are encapsulated into a $(m+3) \times 2$ matrix $\mathsf{h} = (\mathbf{h}^x \;\; \mathbf{h}^y)$ where the $m+3$ vectors $\mathbf{h}^x$ and $\mathbf{h}^y$ are defined by ${\mathbf{h}^*}^\mathsf{T} = (w_1^* \;\; \cdots \;\; w_m^* \;\; \alpha^* \;\; \gamma^* \;\; \delta^*)$.

In practice, parameters h are computed by minimizing the following cost function [4]:

$$\psi(f, \mathbf{x}_j, \mathbf{x}'_j, \sigma) = \sigma \Phi(f) + \sum_{j=1}^{m} d^2(\mathbf{x}'_j, f(\mathbf{x}_j)).$$

The regularization parameter $\sigma \in \mathbb{R}^+$ controls the trade-off between the smoothness of the mapping and the interpolation. In the limiting case $\sigma = 0$, f interpolates the point correspondences. When $\sigma > 0$, they are approximated, and when $\sigma \rightarrow +\infty$, the mapping tends to an affine transformation.

By taking the partial derivatives of ψ with respect to h, one obtains the following linear system:

$$\mathsf{C}(\mathbf{x}_j)\mathsf{h} = \mathsf{D}(\mathbf{x}'_j). \quad (2)$$

where the $(3+m) \times (3+m)$ matrix $\mathsf{C}(\mathbf{x}_j)$ depends only on the coordinates in the source image, while the $(3+m) \times 2$ matrix $\mathsf{D}(\mathbf{x}'_j)$ contains the coordinates of the points in the target image:

$$\mathsf{C}(\mathbf{x}_j) = \begin{pmatrix} \mathsf{K} + \sigma \mathrm{I} & \mathsf{P} \\ \mathsf{P}^\mathsf{T} & 0 \end{pmatrix}, \; \mathsf{D}(\mathbf{x}'_j)^\mathsf{T} = (\mathbf{x}'_1 \;\; \cdots \;\; \mathbf{x}'_m \;\; \mathbf{0} \;\; \mathbf{0} \;\; \mathbf{0}),$$

where the (r, c)-th entry of matrix K is $E(\|\mathbf{x}_r - \mathbf{x}_c\|)$ and the r-th row of matrix P is $(1 \;\; \mathbf{x}_r^\mathsf{T})$. The last three equations are the 'side conditions', which ensure that the computed mapping has square integrable second derivatives, i.e. that it behaves smoothly outside the region of interest.

The outcome is a vector valued function $f(\mathbf{x})$, which exactly maps the corresponding points $\mathbf{x}_j \leftrightarrow \mathbf{x}'_j$ onto each other (for $\sigma = 0$) and is a smooth interpolant for other points.

3. Computing the Mappings

We are given an image sequence of a deforming surface, where there may be relative motion between the surface and camera. Our goal is to recover a set of thin-plate spline mappings between the reference frame and each of the other frames of the sequence. We wish to exploit curve, as well as point, correspondences in computing these maps. Whilst mappings can be constructed from point correspondences as reviewed in §2.3, the extension to curve correspondences is not trivial.

1. *Tracking*. Compute point $\mathbf{x}_{ij}$ and curve $\mathcal{C}_{ik}$ tracks. Gather the points in matrix X.
2. *Rank and regularization parameter selection*, §3.1. Compute the rank r to be used for non-rigid factorization and the regularization parameter σ of the thin-plate spline mappings by cross-validation.
3. *Non-rigid factorization*, §2.2. Factorize X into motion M and shape S. The reprojections are $\hat{\mathsf{X}} = \mathsf{MS}$.
4. *Thin-plate spline mappings estimation*, §2.3. Use reprojected points in matrix $\hat{\mathsf{X}}$ to estimate the f_i.
5. *Curves registration assessment*, §3.2. For each curve k, if the registration error $\mathcal{D}_k^2 > \mu^2$, introduce a virtual point on this curve, as described in §3.3. If all curves are well-registered, converge.
6. *Basis points reconstruction*, §3.4. For all virtual points, reconstruct a basis point by appending a column $\mathbf{s}_{kp}$ to the shape matrix S. Append the corresponding column to matrix X.
7. *Global refinement (optional)*, §3.5. Minimize the global error function $\mathcal{E}$, equation (6), over the mappings and all virtual basis points $\mathbf{s}_{kp}$.
8. *Iteration*. Loop on 3.

Table 1. Point- and curve-based multiple-view dense registration algorithm. The curve registration threshold μ is typically chosen as 1/10 pixels.

The idea is to first use measured point correspondences throughout the sequence to compute an initial estimate of the cameras and basis points via non-rigid factorization. A thin-plate spline mapping can then be built from the estimated point correspondences between the reference image and any other frame. Note that the rank r used in the non-rigid factorization and the regularization parameter σ used for the mappings are computed at this stage by minimizing a cross-validation error, as described in §3.1. The computed thin-plate spline provides an initial estimate of the desired

map. However, this initial estimate will induce a registration error on the curves, and the map is then refined by minimizing this registration error. Our method differs from that of [4] principally in this use of points to provide an initial estimate that is consistent over multiple frames.

More precisely, the algorithm proceeds as follows. The initial parameters h_i of mappings f_i are computed using the real point correspondences $\mathbf{x}_{ij}$, where f_i is the mapping between the reference image and the i-th image. For each curve correspondence $\mathcal{C}_{ik}$, $i = 1, \ldots, n$, a multi-view registration error is computed, as the mean distance between the reference curve $\mathcal{C}_{1k}$ mapped with f_i to images $i = 2, \ldots, n$ and the corresponding curves $\mathcal{C}_{ik}$. The computation of the curve registration error is described in §3.2. If the registration error of a curve is too high, say the distance is greater than 1/10 pixels, this means that the image area surrounding the curve is badly mapped, and that the curve may provide useful constraints to improve the mapping. To incorporate this information, we introduce a virtual point $\mathbf{y}_{ikq}$ on the curve, as described in §3.3. This process is iterated until the registration of all curves becomes satisfactory. The role of the virtual point correspondences, chosen on corresponding curves, is to make the mappings computed throughout the iterations register the curves better and better. In other words, the virtual point correspondences are used to incorporate into the mapping the information provided by the curve correspondences.

The iterative process is important since it ensures that all curves are well-registered by the mapping, and that the set of virtual points is minimal. The alternative solution of introducing a fixed number of virtual points on each curve is not satisfactory, since this set of points could be redundant for some curves, while other curves could be badly registered by the mapping.

Introducing point correspondences on curves yields one major problem: nothing guarantees that they are actually consistent, in the sense of being the image of a unique 3D point. We deal with this problem using the non-rigid factorization reviewed in §2.2. A similar idea is followed in [13] for the reconstruction of unreliable point correspondences. Using non-rigid factorization, we factorize the real point correspondences into a motion and a shape matrix. Given the motion matrix, instead of computing directly the positions of all virtual corresponding points $\mathbf{y}_{ikq}$ along the curves in all images, we instead compute the corresponding 3D basis points $\mathbf{B}$, which makes the virtual points consistent with the low-rank shape constraint. More details are given in §3.4.

Our global refinement step turns into minimizing the distance between the virtual points predicted by the mappings and those predicted by the reconstruction, and the global registration error on curves. The cost function is described in §3.5 and the optimization procedure in appendix A. The algorithm is summarized in table 1.

3.1. Computing the Rank and Regularization

We propose to choose the rank r used in the non-rigid factorization and the regularization parameter σ of the thin-plate splines by minimizing a cross-validation error $\tau(r, \sigma)$. Cross-validation reflects how well the model can interpolate the data. It consists of applying the algorithm to the data but leaving out one of the correspondences. The error in the predicted positions of this correspondence is then measured using the mappings computed from all the remaining correspondences. We perform non-rigid factorization, and use the reprojected points to compute thin-plate splines. The point left out is then transferred from the first view to all other views using the thin-plate splines, and the difference between its actual and predicted positions is computed. The cross-validation error $\tau(r, \sigma)$ is obtained by averaging these differences over all views and all points. By varying the rank r used in non-rigid factorization and the regularization parameter σ of the thin-plate splines, different cross-validation errors are obtained. We choose r and σ such that $\tau(r, \sigma)$ is minimized.

3.2. Assessing Curve Registration

This section deals with computing the multi-view registration error $\mathcal{D}_k$ of a curve correspondence $\mathcal{C}_{ik}$, $i = 1, \ldots, n$, given a multi-view mapping f_i, $i = 2, \ldots, n$. As explained previously, we use a transfer error, given by the mean of the distances between $f_i(\mathcal{C}_{1k})$, the reference curve $\mathcal{C}_{1k}$ mapped to image $i > 1$, and the corresponding curve $\mathcal{C}_{ik}$:

$$\mathcal{D}_k^2 = \frac{1}{n-1} \sum_{i=2}^{n} \varepsilon^2(f_i(\mathcal{C}_{1k}), \mathcal{C}_{ik}), \tag{3}$$

where ε is a distance measure between two curves. A natural distance measure is the average of the distances between e regularly sampled points on $\mathcal{C}$ with parameters t_q, $q = 1, \ldots, e$, and $\mathcal{C}'$:

$$\varepsilon^2(\mathcal{C}, \mathcal{C}') = \frac{1}{e} \sum_{q=1}^{e} \left(\min_{t' \in [0,1]} d^2(\mathcal{C}(t_q), \mathcal{C}'(t')) \right). \tag{4}$$

By substituting equation (4) into equation (3), we obtain:

$$\mathcal{D}_k^2 = \nu \sum_{i=2}^{n} \sum_{q=1}^{e} \left(\min_{t' \in [0,1]} d^2(f_i(\mathcal{C}_{1k}(t_{kq})), \mathcal{C}_{ik}(t')) \right), \tag{5}$$

where $\nu = \frac{1}{e(n-1)}$ is a normalizing factor. This expression can be evaluated by computing $e(n-1)$ independent one-dimensional minimisations over t'. We use the Newton algorithm with analytic differentiation. Reliable initial solutions are provided by $t' = t_{kq}$, and the constraint $t' \in [0, 1]$ is enforced by a simple clamping. Note that for certain

curve parameterizations, direct specific solutions may exist to compute $\mathcal{D}_k$.

3.3. Choosing the Virtual Points

At each iteration of the algorithm, the current estimated mapping is tested against each curve. Assume that the registration error $\mathcal{D}_k$ of the k-th curve is not satisfactory. We drop the curve index k for clarity throughout this section.

A virtual point $\mathbf{y}_{1q} = \mathcal{C}_1(s_{1q})$ is introduced on the curve in the reference image (see below). The corresponding virtual points in the other views are chosen on the corresponding curves, such the mapping computed at the next iteration will tend to better match the curves. A trivial solution to chose the corresponding points $\mathbf{y}_{iq}$ is to evaluate all images of the curve with the same parameters, i.e. $\mathbf{y}_{iq} = \mathcal{C}_i(s_{1q})$. However, this solution highly depends on the parameterisation of the curves. We prefer to choose for the $\mathbf{y}_{iq}$ the closest points to the points predicted by the current mapping:

$$\mathbf{y}_{iq} = \mathcal{C}(s_{iq}) \text{ with } s_{iq} = \arg\min_s d^2(f_i(y_{1q}), \mathcal{C}_i(s)).$$

The virtual point introduced in the reference image is chosen such that it has a high probability to reduce the curve registration error, i.e. such the total error $\sum_{i=2}^{n} d^2(f_i(y_{1q}), \mathcal{C}_i(s_{iq}))$ induced by chosing the corresponding points is maximized.

3.4. Reconstructing the Basis Points

We drop the curve index k for clarity throughout this section. We tackle the problem of finding the virtual corresponding points of $\mathbf{y}_{ip}$, and reconstructing the underlying basis point $\mathbf{s}_p$. Point $\mathbf{y}_{1p}$ lies on $\mathcal{C}_1$ in the reference image and has been introduced to incorporate information from this curve correspondence in the mapping. Computing the corresponding points $\mathbf{y}_{2p}, \ldots, \mathbf{y}_{np}$ is difficult due to the aperture problem: they may be located anywhere along the curves $\mathcal{C}_2, \ldots, \mathcal{C}_n$ in frames $2, \ldots, n$ respectively. This problem is $(n-1)$-dimensional, where n is the number of views. To reduce this high-dimensional problem, we consider the previously computed non-rigid factorization $\hat{\mathsf{X}} = \mathsf{MS}$. Each column $\mathbf{s}$ of the shape matrix S corresponds to a particular point. For any virtual point $\mathbf{y}_{1p}$, we append a column $\mathbf{s}_p$ to matrix S. The corresponding image points are given by $\begin{pmatrix}\hat{\mathbf{y}}_{1p}^\mathsf{T} & \cdots & \hat{\mathbf{y}}_{np}^\mathsf{T}\end{pmatrix}^\mathsf{T} = \mathsf{M}\mathbf{s}_p$. We have reduced our $(n-1)$-dimensional problem to a r-dimensional one, where r is the rank used for non-rigid factorization, e.g. from 39 to 3 on the 40 frame Lola sequence. The accurate estimation of $\mathbf{s}_p$ is part of the global non-linear refinement described in the next section. We compute $\mathbf{s}_p$ such that the distances between the predicted points $\hat{\mathbf{y}}_{ip}$ and the points $\mathbf{y}_{ip}$ is minimized for $i = 2, \ldots, n$, and such that $\hat{\mathbf{y}}_{1p} = \mathbf{y}_{1p}$:

$$\min_{\mathbf{s}_p \mid \hat{\mathbf{y}}_{1p} = \mathbf{y}_{1p}} \sum_{i=2}^{n} d^2(\hat{\mathbf{y}}_{ip}, \mathbf{y}_{ip}).$$

This is a linear least squares problem under linear constraints, that we solve using the framework described in e.g. [5, A3.4.4].

3.5. Non-Linear Global Refinement

We deal with the last step of our iterative algorithm, the global refinement of the mappings parameters and virtual points positions. We minimize an error function consisting of two terms:

$$\min_{\mathsf{h}_i, \mathbf{s}_{kp}} \mathcal{E} \text{ with } \mathcal{E}^2 = \mathcal{E}_{rec}^2 + \zeta^2 \mathcal{E}_{reg}^2, \tag{6}$$

where h_i are the parameters of the i-th mapping f_i and ζ^2 is a weight that we currently choose as 1. The reconstruction term $\mathcal{E}_{rec}$ and the registration term $\mathcal{E}_{reg}$ are described below. The appendix shows how the optimization can be carried out using sparse matrix inversion.

The reconstruction term concerns the reconstruction of the basis points for the virtual points. It measures the difference between the virtual image points predicted by the reconstruction and those predicted by the mapping: $\mathcal{E}_{rec}^2 = \sum_{k=1}^{l} \sum_{i=2}^{n} \sum_{p=1}^{d_k} d^2(\hat{\mathbf{y}}_{ikp}, f_i(\mathbf{y}_{1kp}))$, where d_k is the number of virtual points on curve k.

The registration term accounts for the registration of the curves across the images. It is based on the multi-view curve registration error $\mathcal{D}_k$ of §3.2: $\mathcal{E}_{reg}^2 \propto \sum_{k=1}^{l} \mathcal{D}_k^2$.

4. Experimental Results

4.1. Implementation Details

We mark real points and curves in the first image, and use the Shi-Tomasi point tracker [11] to get the real point correspondences. Curves are modeled by natural splines. We track each curve by computing a local non-rigid transformation g by minimizing an intensity-based registration error on the image patch $\mathcal{X}$ surrounding the curve:

$$\min_{g_i} \sum_{\mathbf{x} \in \mathcal{X}} \left(\mathcal{I}_1(\mathbf{x}) - \mathcal{I}_i(g_i(\mathbf{x}))\right)^2,$$

where $\mathcal{I}_i(\mathbf{x})$ is the intensity or colour of the pixel with coordinates $\mathbf{x}$ in image $\mathcal{I}_i$, and g_i is a thin-plate spline with a fixed number of centres, chosen as 4 in our implementation. Transformations g_i are used to transfer the control points of the curve, which are finely tuned by maximizing the normal image gradient along the curve.

More sophisticated curve trackers, see e.g. [8], based on geodesic snakes, could be used as well.

4.2. Computing the Rank and Regularization

We use the procedure of §3.1 based on cross-validation to determine the rank r to be used in non-rigid factorization and the regularization parameter of the thin-plate splines.

Figure 2. (top row) Sample frames (5 out of 105) of the Bears sequence. (second row) the point and curve correspondences. (third row) the computed flow field. (last row) the augmented sequence.

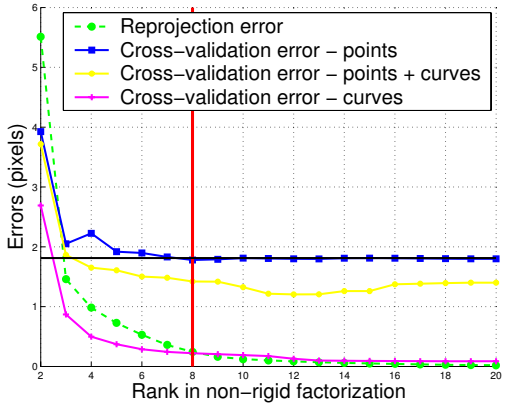


Figure 3. Cross-validation on the Bears sequence, measured against the regularization parameter σ of the thin-plate spline mappings and the rank r used in non-rigid factorization. The vertical line indicates the selected rank $r = 8$. The horizontal line indicates the cross-validation computed on raw point tracks. The cross-validation for the curves clearly shows that they improve the computed mappings.

Cross-validation also provides a quantitative analysis. We use the 105 frames of the Bears sequence shown in figure 2. We chose this sequence since many stable point and curve correspondences can be obtained. More precisely, we tracked 94 points and 6 curves.

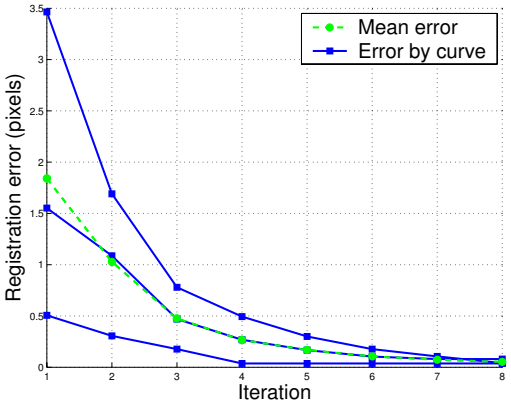


Figure 4. Multi-view curve registration error through the iterations of the Lola sequence, from top to bottom: hairline, left and right eyebrows.

Figure 3 shows the cross-validation plotted against different values of r. For each value of r, the σ minimizing the cross-validation is employed.

The same experiment run on the Lola sequence, not shown here, gave $r = 3$.

4.3. Augmenting Images

We give a qualitative evaluation of the registration algorithm by augmenting sequences. We first consider the Lola sequence, consisting of 40 frames, from which samples are given in figure 5. The 33 real points and 3 curves used are shown on figure 1. The algorithm requires a total of 7 iterations to converge, to a 0.052 pixels average registration error on the curves, and 16 virtual points to be inserted – 7 on the hairline, 6 on the left eyebrow and 3 on the right eyebrow. Figure 4 shows the evolution of the registration error for each curve and the mean over all curves, as the iterations proceed. Figure 6 shows the curves mapped from the reference image to another image of the sequence through the iterations. We observe that the quality of the mapping in the vicinity of the curves clearly improves through the iterations, until finally the mapped curve becomes indistinguishable from the tracked curve. Figure 7 shows images of the Lola sequence augmented with a logo on the forehead. There is a significant gain of quality between the result obtained using only the real points and the result obtained by applying our algorithm to real points and curves.

Figure 7. (left) The augmented reference image. (middle) The augmentation transferred based on real points. (right) Using real points and curves. In the former case, the logo is deformed and shifted towards the hairline, while in the latter case, it is centred on the forehead, as in the reference image.

5. Conclusions and Extensions

One drawback of using the reference image registration approach is that occlusions can not be handled easily. This limits the number of sequences that can be dealt with. A more explicit 3D surface representation would address this problem.

We plan to introduce a final step designed to tune the mapping's parameters based on a direct method, i.e. intensity-based, that should capture more completely all the fine deformations of the surface.

A. Sparse Optimization Algorithm

We show how to carry out the optimization of criterion $\mathcal{E}$, equation (6), using a sparse Levenberg-Marquart algorithm. By merging the two sums over k and on i, and incorporating the inner minimization over t' in the main outer minimization by introducing the parameters t'_{ikq}, the problem becomes:

$$\min_{\mathsf{h}_i, \mathbf{s}_{kp}, t'_{ikq} \in [0,1]} \quad \sum_{k=1}^{l} \sum_{i=2}^{n} \left(\sum_{q=1}^{e} d^2(f_i(\mathcal{C}_{1k}(t_{kq})), \mathcal{C}_{ik}(t')) \right) + \left(\sum_{p=1}^{d_k} d^2(\hat{\mathbf{y}}_{ikp}, f_i(\mathbf{y}_{1kp})) \right).$$

Let J denotes the Jacobian matrix of $\mathcal{E}$. The Levenberg-Marquardt algorithm consists in iteratively solving normal equations $(\mathsf{H} + \theta \mathrm{I})\boldsymbol{\delta} = -\mathbf{g}$, where $\mathbf{g} = \mathsf{J}^\mathsf{T}\mathbf{r}$ is the gradient and $\mathsf{H} = \mathsf{J}^\mathsf{T}\mathsf{J}$ the Gauss-Newton approximation of the Hessian matrix. Parameter $\theta \in \mathbb{R}$ is tuned heuristically. We

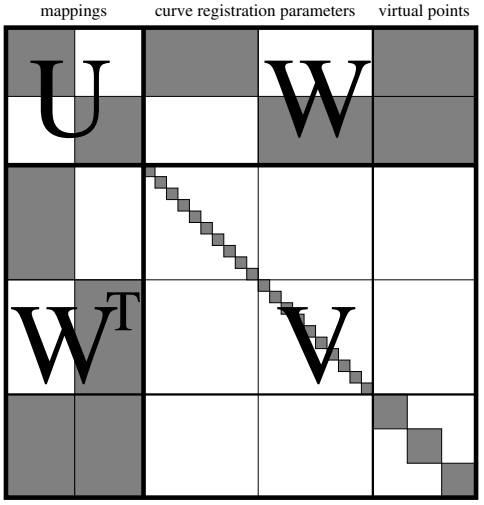


Figure 8. Shape of the Hessian matrix H (Gauss-Newton approximation) for a toy example with $n = 3$ images, $m = 3$ real points, $l = 2$ curves, $e = 5$ sampled points for curve registration error estimation, $d_1 = 2$ virtual points on curve 1, $d_2 = 1$ virtual point on curve 2 and rank $r = 3$ for non-rigid factorization.

refer to e.g. [5, A4.2] for more details. In order to solve the normal equations efficiently, we investigate the shape of matrix H, shown in figure 8. As can be seen, this matrix has

Figure 5. Sample frames (5 out of 40) from the film 'Run Lola Run'.

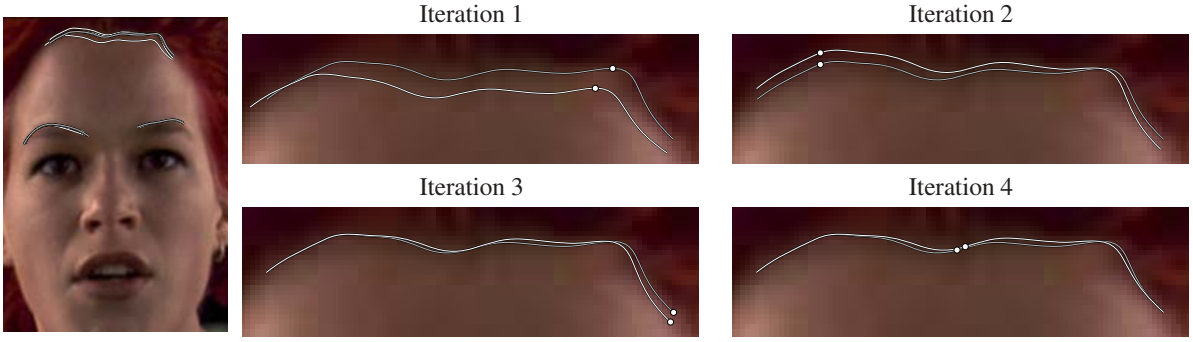


Figure 6. Evolution of the mapping of the curves through the iterations. The grey curve is the tracked curve, while the white curves are predicted by the mappings. (left) All curves. (right) close-up on the hairline shows that the mapping is improved through the iterations in the vicinity of the curve. On each image, we show the virtual point correspondence that is incorporated at the next iteration.

a strong sparse block structure that we exploit to solve the normal equations using partitioning techniques from bundle adjustment, as described in e.g. [5, A4.3], based on the U, V and W blocks shown on figure 8. Assuming that the initial solution is in the best region of convergence, we enforce the constraints $t'_{ikq} \in [0, 1]$ using a simple clamping. The derivatives are computed in a very simple manner since $\mathsf{C}(\mathbf{x}_j)$, equation (2) is a constant matrix and hence the thin-plate spline mapping, equation (1), is linear in the unknowns.

References

[1] B. Bascle and A. Blake. Separability of pose and expression in facial tracing and animation. *International Conference on Computer Vision, Bombay, India*, pages 323–328, 1998.

[2] F. L. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. *Pattern Analysis and Machine Intelligence*, 11(6):567–585, June 1989.

[3] M. Brand. Morphable 3D models from video. *Proceedings of the Conference on Computer Vision and Pattern Recognition, Kauai, Hawaii, USA*, pages II: 456–463, 2001.

[4] H. Chui and A. Rangarajan. A new point matching algorithm for non-rigid registration. *Computer Vision and Image Understanding*, 89(2):114–141, February 2003.

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

[6] M. Irani. Multi-frame optical flow estimation using subspace constraints. *International Conference on Computer Vision*, volume I, pages 626–633, September 1999.

[7] R. Koch, M. Pollefeys, and L. J. Van Gool. Realistic surface reconstruction of 3D scenes from incalibrated image sequences. *Journal of Visualization and Computer Animation*, 11(3):115–127, 2000.

[8] N. Paragios and R. Deriche. Geodesic active regions for tracking. *Proceedings of the 7th International Conference on Computer Vision, Kerkyra, Greece*, pages 688–694, 1999.

[9] H. Aanæs and F. Kahl. Estimation of deformable structure and motion. *Proceedings of the Vision and Modelling of Dynamic Scenes Workshop, Copenhagen, Denmark*, June 2002.

[10] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1):7–42, May 2002.

[11] J. Shi and C. Tomasi. Good features to track. *Proceedings of the Conference on Computer Vision and Pattern Recognition, Seattle, Washington, USA*, pages 593–600, 1994.

[12] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *International Journal of Computer Vision*, 9(2):137–154, 1992.

[13] L. Torresani and C. Bregler. Space-time tracking. *Proceedings of the European Conference on Computer Vision, Copenhagen, Denmark*, pages 801–812, June 2002.

[14] L. Torresani, D. Yang, G. Alexander, and C. Bregler. Tracking and modelling non-rigid objects with rank constraints. *Proceedings of the Conference on Computer Vision and Pattern Recognition, Kauai, Hawaii, USA*, 2001.

7.1.2 Paper (WDV'05) – *A Batch Algorithm For Implicit Non-Rigid Shape and Motion Recovery*

A Batch Algorithm For Implicit Non-Rigid Shape and Motion Recovery

Adrien Bartoli[1] and Søren I. Olsen[2]

[1] CNRS – LASMEA, France
[2] DIKU, Denmark
Adrien.Bartoli@gmail.com and Ingvor@diku.dk

Abstract. The recovery of 3D shape and camera motion for non-rigid scenes from single-camera video footage is a very important problem in computer vision. The low-rank shape model consists in regarding the deformations as linear combinations of basis shapes. Most algorithms for reconstructing the parameters of this model along with camera motion are based on three main steps. Given point tracks and the rank, or equivalently the number of basis shapes, they factorize a measurement matrix containing all point tracks, from which the camera motion and basis shapes are extracted and refined in a bundle adjustment manner. There are several issues that have not been addressed yet, among which, choosing the rank automatically and dealing with erroneous point tracks and missing data.
We introduce theoretical and practical contributions that address these issues. We propose an implicit imaging model for non-rigid scenes from which we derive non-rigid matching tensors and closure constraints. We give a non-rigid Structure-From-Motion algorithm based on computing matching tensors over subsequences, from which the implicit cameras are extrated. Each non-rigid matching tensor is computed, along with the rank of the subsequence, using a robust estimator incorporating a model selection criterion that detects erroneous image points.
Preliminary experimental results on real and simulated data show that our algorithm deals with challenging video sequences.

1 Introduction

Structure-From-Motion – the recovery of 3D shape and camera motion from images – is one of the most studied problems in computer vision. The decades of work has led to significant successes, especially when the observed environment is static. However, the assumption of rigidity is violated in many cases of interest, for example expressive faces, moving cars, etc. For that reason, dealing with non-rigid scenes coming from single-camera footage has received an increasing attention over the last few years. The problem is highly challenging since both the camera motion and the non-rigid 3D shape have to be recovered. A major step forwards for such cases was made by Bregler *et al.* [5, 8], Brand [4] and Aanæs *et al.* [1]. Building on the work of [2, 6], they developed and demonstrated factorization of images of non-rigid scenes, where the non-rigidity was

represented as a linear combination of *basis shapes*. Xiao *et al.* [12] studied the degenerate deformations that may defeat the reconstruction algorithms.

This paper tackles the two following open problems. *(i)* the factorization of a measurement matrix containing all point tracks in the presence of missing and erroneous image points. This must be done to recover the parameters of the implicit imaging model. Most previous work do not deal with missing data [1, 4, 5, 8, 11]. *(ii)* the automatic choice of the rank r of the measurement matrix, characterising the degree of non-rigidity in the sequence. Most previous work rely on a user-defined rank [4, 5, 8, 11].

More precisely, we build on the low-rank shape model to derive an *implicit imaging model* projecting points affinely from $\mathbb{R}^r$ – the implicit shape points – onto the images using *implicit camera matrices*. The rank r reflects the degree of non-rigidity of the model and is thus a very important parameter. This implicit model is simpler than the *explicit model* used in *e.g.* [5, 8], in the sense that it ignores the replicated block structure of the camera matrices. The implicit model gives weaker constraints on point tracks than the explicit model. It is the model used for non-rigid factorization in *e.g.* [5, 8, 11]. Based on this model, we derive *non-rigid matching tensors* that constrain point tracks and encapsulate information about the implicit camera matrices. We define non-rigid closure constraints relating the matching tensors to the implicit camera matrices. These theoretical concepts are based on the fact that implicit reconstruction is performed in $\mathbb{R}^r$. They lead to a batch algorithm for computing the motion and structure matrices in the presence of erroneous and missing data. The idea is to robustly compute a set of matching tensors over several subsequences using MAPSAC and the GRIC criterion to choose the associated rank [7]. From these matching tensors, we solve for the implicit camera matrices using the closure constraints. The next step consists in computing the basis shapes by non-rigid triangulation. We refine both the implicit cameras and implicit shape in a bundle adjustment manner. Finally, each image point is classified as an inlier or an outlier. Almost all steps in this algorithm are done robustly, meaning that blunders are detected and thus do not corrupt the computation.

Roadmap. In §2, we derive the non-rigid shape and imaging models. We examine previous work in §3. We derive the non-rigid matching tensors and closure constraints in §§4 and 5 respectively. Our Structure-From-Motion algorithm is derived in §6 while the robust estimation of matching tensors and associated ranks is given in §7. Experimental results are reported in §8 and our conclusions in §9.

Notation. Vectors are denoted using bold fonts, *e.g.* $\mathbf{x}$ and matrices using sans-serif or calligraphic characters, *e.g.* M or $\mathcal{X}$. Index $i = 1, \ldots, n$ is used for the images, $j = 1, \ldots, m$ for the points and $k = 1, \ldots, l$ for the basis shapes, *e.g.* $\mathbf{x}_{ij}$ is the position of the j-th point track in the i-th image and $\mathbf{B}_{kj}$ is the k-th basis shape for the j-th point. Visibility indicators modeling occlusions are denoted v_{ij}. The Hadamard (element-wise) product is written $\odot$. The zero and one vectors are respectively $\mathbf{0}$ and $\mathbf{1}$, $\mathsf{0}$ is the zero matrix and $^\mathsf{T}$ is vector and matrix

transpose. Bars indicate centred data, as in *e.g.* $\bar{\mathcal{X}}$. Notation $[i, i']$ refers to a subsequence between image i and image i', *e.g.* $\mathcal{X}_{[i,i']}$ is the measurement matrix for this subsequence. $\{\}$ is a set over some variable. We use the Singular Value Decomposition, denoted SVD, *e.g.* $\mathcal{X} = \mathsf{U}\Sigma\mathsf{V}^\mathsf{T}$ where U and V are orthonormal matrices, and Σ is diagonal, containing the singular values of $\mathcal{X}$ in decreasing order.

2 Non-Rigid Imaging Model

2.1 Explicit Model

The low-rank shape assumption consists in writing the coordinates of a time-varying set of points $\mathbf{Q}_{ij}$ as linear combinations over l *basis shapes* $\mathbf{B}_{kj}$ with the *configuration weights* α_{ik}: $\mathbf{Q}_{ij} = \sum_{k=1}^{l} \alpha_{ik}\mathbf{B}_{kj}$. Points $\mathbf{Q}_{ij}$ are projected onto the images by affine cameras: $\mathbf{x}_{ij} = \mathsf{P}_i\mathbf{Q}_{ij} + \mathbf{t}_i$, from which the explicit imaging model is obtained:

$$\mathbf{x}_{ij} = \mathsf{P}_i \left(\sum_{k=1}^{l} \alpha_{ik}\mathbf{B}_{kj} \right) + \mathbf{t}_i. \tag{1}$$

This trilinear equation is the most explicit form of the low-rank shape imaging model. Only rank-3 basis shapes are considered for simplicity, but rank-2 and rank-1 basis shapes can be modeled as well [12].

2.2 Implicit Model

Rewriting (1), one obtains:

$$\begin{aligned} \mathbf{x}_{ij} &= \begin{pmatrix} \alpha_{i1}\mathsf{P}_i & \cdots & \alpha_{il}\mathsf{P}_i \end{pmatrix} \begin{pmatrix} \mathbf{B}_{1j}^\mathsf{T} & \cdots & \mathbf{B}_{lj}^\mathsf{T} \end{pmatrix}^\mathsf{T} + \mathbf{t}_i \\ &= \mathsf{M}_i\mathbf{S}_j + \mathbf{t}_i \qquad \text{with} \qquad \mathsf{M}_i = \begin{pmatrix} \alpha_{i1}\mathsf{P}_i & \cdots & \alpha_{il}\mathsf{P}_i \end{pmatrix} \end{aligned} \tag{2}$$

We call M_i a $(2 \times 3l)$ *explicit camera matrix* and $\mathbf{S}_j^\mathsf{T} = \begin{pmatrix} \mathbf{B}_{1j}^\mathsf{T} & \cdots & \mathbf{B}_{lj}^\mathsf{T} \end{pmatrix}$ a $(3l \times 1)$ *shape vector*. Introduce $r = 3l$, the rank of the model, a $(r \times r)$ full-rank matrix $\mathcal{A}$ and relaxing the replicated structure yields the bilinear *implicit model*. From (2), $\mathbf{x}_{ij} = \mathsf{M}_i\mathbf{S}_j + \mathbf{t}_i = \left(\mathsf{M}_i\mathcal{A}^{-1}\right)\left(\mathcal{A}\mathbf{S}_j\right) + \mathbf{t}_i$, giving:

$$\mathbf{x}_{ij} = \mathsf{J}_i\mathbf{K}_j + \mathbf{t}_i. \tag{3}$$

We call $\mathsf{J}_i = \mathsf{M}_i\mathcal{A}^{-1}$ and $\mathbf{K}_j = \mathcal{A}\mathbf{S}_j$ the *implicit camera matrix* and the *implicit shape matrix* respectively. Matrix $\mathcal{A}$ represents a *corrective transformation*. As shown in the next section, this is the model used for non-rigid factorization. The model generalizes, in some sense, the $\mathbb{P}^k \to \mathbb{P}^2$ projection matrices introduced by Wolf *et al.* [10].

3 Previous Work

Most of the previous work [1, 4, 5, 8, 11] is based on factorizing a measurement matrix using SVD and hence do not cope with missing data. We note that Torresani *et al.* [8] propose an approach where the likelihood of the explicit model is maximized over the entire image sequence using a generalized EM (Expectation Maximization) algorithm which finds the nearest local optimum. The important rank selection problem is neglected in most papers, besides [1]. Below, we describe the three main steps involved in most algorithms. The inputs are the complete measurement matrix $\mathcal{X}$ and the rank r. The outputs are the camera pose, the configuration weights and the basis shapes.

Step 1: Factorizing. A $(2n \times m)$ measurement matrix $\mathcal{X}$ is built by gathering all point coordinates. The translation part of the imaging model, *i.e.* the $\mathbf{t}_i$, is estimated as the mean of the point coordinates in each image. A $(2n \times 1)$ joint translation vector $\mathbf{t}^\mathsf{T} = (\mathbf{t}_1^\mathsf{T} \ \cdots \ \mathbf{t}_n^\mathsf{T})$ is built and used to centre the measurement matrix: $\bar{\mathcal{X}} \leftarrow \mathcal{X} - \mathbf{t} \cdot \mathbf{1}^\mathsf{T}$, from which we get:

$$\underbrace{\begin{pmatrix} \mathbf{x}_{11} & \cdots & \mathbf{x}_{1m} \\ \vdots & \ddots & \vdots \\ \mathbf{x}_{n1} & \cdots & \mathbf{x}_{nm} \end{pmatrix}}_{\bar{\mathcal{X}}_{(2n \times m)}} = \underbrace{\begin{pmatrix} \mathsf{J}_1 \\ \vdots \\ \mathsf{J}_n \end{pmatrix}}_{\mathcal{J}_{(2n \times r)}} \underbrace{\begin{pmatrix} \mathbf{K}_1 & \cdots & \mathbf{K}_m \end{pmatrix}}_{\mathcal{K}_{(r \times m)}},$$

where $\mathcal{J}$ and $\mathcal{K}$ are the joint implicit camera and shape matrices. The centred measurement matrix is factorized using SVD as $\bar{\mathcal{X}} = \mathsf{U}\Sigma\mathsf{V}^\mathsf{T}$. The joint implicit camera and shape matrices $\mathcal{J}$ and $\mathcal{K}$, are recovered as the r leading columns of *e.g.* U and $\Sigma\mathsf{V}^\mathsf{T}$ respectively.

Step 2: Upgrading. The implicit model is upgraded to the explicit one by computing a corrective transformation. Xiao *et al.* [11] show that constraints on both the explicit camera and shape matrices must be considered to achieve a unique solution, namely the 'rotation' and the 'basis' constraints. They give a closed-form solution based on these constraints. Previous work [4, 5, 8] use only the rotation constraints, leading to ambiguous solutions. For instance, Brand [4] shows that a block-diagonal corrective transformation is a good practical approximation. Once the replicated structure has been approximately enforced, the rotation matrices are extracted using orthonormal decomposition. The configuration weights are then recovered using the orthonormality of the rotation matrices. Bregler *et al.* [5] assume that the information about each basis shape is distributed in the appropriate column triple in the shape matrix by the initial SVD, in other words that the entries off the block-diagonal of the corrective transformation matrix are negligible. Experiments show that this assumption restricts the cases that can be dealt with since only limited non-rigidity can be handled. A second factorization round on the reordered weighted motion matrix elements enforces the replicated block structure, yielding the weight factors and

the P_i, which are upgraded to Euclidean by computing a linear transformation as in the rigid factorization case. Aanæs *et al.* [1] assume that the structure resulting from rigid factorization gives the mean non-rigid structure and camera motion. Given the camera motion, recovering the structure is done by examining the principal components of the estimated variance.

Step 3: Nonlinear refinement. The solution obtained so far is finely tuned in a bundle adjustment manner by minimizing *e.g.* the reprojection error. The algorithms proposed in [4, 8] differ by the prior they are using to regularize the solution. These priors state that the reconstructed shapes should not vary too much between consecutive images.

4 Non-Rigid Matching Tensors

Matching tensors are known for the rigid case. Examples are the fundamental matrix and the trifocal tensor. They relate the image position of corresponding points over multiple images. The implicit imaging model allows us to derive matching tensors for non-rigid scenes.

A non-rigid matching tensor is a matrix $\mathcal{N}$ whose columns span the d dimensional nullspace of the $(2n \times m)$ centred measurement matrix $\bar{\mathcal{X}}$:

$$\mathcal{N}^\mathsf{T}\bar{\mathcal{X}} = \mathsf{0}. \tag{4}$$

The size of matrix $\mathcal{N}$ is $(2n \times d)$ where the tensor dimension is $d = 2n - r$. Loosely speaking, $\mathcal{N}$ constrain each point track $\bar{\mathbf{x}}_j$ – the j-th column of $\bar{\mathcal{X}}$ – by $\mathcal{N}^\mathsf{T}\bar{\mathbf{x}}_j = \mathbf{0}$. These constraints easily extend to the non centred measurement matrix $\mathcal{X}$ by substituting $\bar{\mathcal{X}} = \mathcal{X} - \mathbf{t} \cdot \mathbf{1}^\mathsf{T}$ into equation (4): $\begin{pmatrix}\mathcal{N}^\mathsf{T} & -\mathcal{N}^\mathsf{T}\mathbf{t}\end{pmatrix}\begin{pmatrix}\mathcal{X} \\ \mathbf{1}^\mathsf{T}\end{pmatrix} = \mathsf{0}$.

Minimal number of points and views. The three following parameters are characteristic of an image sequence: the number of images n, the number of point tracks m and the rank r. They can be related to each other, in particular for, given r, deriving what the minimal number of point tracks and views are for computing the matching tensor. The computation is possible if the $(2n \times m)$ centred measurement matrix $\bar{\mathcal{X}}$ is at least of size $(r \times r)$. Counting the point track needed to compute the translations for centring the measurement matrix, we directly get the minimal number of point tracks as $m \geq r + 1$. From $2n \geq r$, we obtain the minimal number of views as $n \geq \lfloor \frac{r}{2} \rfloor + 1$. These numbers can also be derived by counting the number of degrees of freedom in the tensor and the number of independent constraints given by equation (4).

5 Non-Rigid Closure Constraints

The closure constraints introducted by Triggs in [9] relate matching tensors to projection matrices. These constraints are used to derive a batch Structure-From-Motion algorithm dealing with high amounts of missing data.

In this section, we derive new types of closure constraints for the non-rigid case, based on the above-derived matching tensors, namely the $\mathcal{N}$-closure. Our derivation is valid for any rank r.

Let $\mathbf{K} \in \mathbb{R}^r$ be an implicit shape point. We project $\mathbf{K}$ in the images using the joint implicit camera matrix $\mathcal{J}$: $\bar{\mathbf{x}} = \mathcal{J}\mathbf{K}$, $\forall \mathbf{K} \in \mathbb{R}^r$. From the definition (4) of the matching tensors, $\mathcal{N}^\mathsf{T}\bar{\mathbf{x}} = \mathbf{0}$. Substituting the joint projection equation yields $\mathcal{N}^\mathsf{T}\mathcal{J}\mathbf{K} = \mathbf{0}$, $\forall \mathbf{K} \in \mathbb{R}^r$, which gives the *$\mathcal{N}$-closure constraint*:

$$\mathcal{N}^\mathsf{T}\mathcal{J} = 0. \tag{5}$$

This constraint means that the joint implicit camera matrix lies in the right nullspace of $\mathcal{N}^\mathsf{T}$.

6 Non-Rigid Structure-From-Motion

Our batch algorithm for implicit non-rigid Structure-From-Motion is based on the above-derived non-rigid matching tensors and closure constraints. It is summarized in table 1. We consider only sets of consecutive images for simplicity. It

---

OBJECTIVE

Given m point tracks over n images as a an incomplete $(2n \times m)$ measurement matrix $\mathcal{X}$ and a $(n \times m)$ visibility matrix $\mathcal{V}$, compute the implicit non-rigid cameras J_i, the non-rigid shape points $\mathbf{K}_j$ and the rank r.

ALGORITHM

1. Partition the sequence, see §6.1 while robustly computing the matching tensors $\{\mathcal{N}_{[i_b,i'_b]}\}$ and associated ranks, see §7.2.
2. Solve for the implicit cameras $(\mathsf{J}_i, \mathbf{t}_i)$ using the closure constraints, see §6.2.
3. Triangulate the point tracks to get the implicit shape points $\mathbf{K}_j$, see §6.3.
4. Nonlinearly refine the implicit cameras and shape points by minimizing the reprojection error, see §6.4.
5. Classify each image point track as an inlier or an outlier.

Table 1. Summary of our non-rigid implicit Structure-From-Motion algorithm.

begins by selecting a set of s subsequences $\{[i_b, i'_b]\}_{b=1}^{b=s}$ and by computing a set of matching tensors $\{\mathcal{N}_{[i_b,i'_b]}\}$, one for each subsequence, and the associated rank estimates $\{r_{[i_b,i'_b]}\}$. Our joint tensor and rank estimation algorithm is presented in §7. The full sequence rank r is the maximum over all subsequence ranks: $r = \max_b(r_{[i_b,i_b]})$.

6.1 Partitioning the Sequence

The measurement matrix is partitioned into overlapping blocks with points visible in all of the selected images. Before going into further details, we must figure out what the minimal tensor dimension is, and how many views each tensor should operate on. Let $[i_b, i'_b]$ and $[i_{b+1}, i'_{b+1}]$ be two consecutive subsequences and let $\delta_{b,b+1} = i_{b+1} - i_b$ be the offset between them. We need to determine what the maximum value of $\delta_{b,b+1}$ is. The b-th matching tensor, with dimension $d_b = 2n_b - r_b$, gives d_b constraints. The number of unknowns constrained by the first matching tensor only is $\delta_{1,2}$, from which we get $\delta_{1,2} \leq n_1 - \lfloor \frac{r_1+1}{2} \rfloor$. Making the same reasoning for the b-th tensor, *i.e.* ignoring the constraints coming from previous overlapping sets, gives a bound on $\delta_{b,b+1}$:

$$\delta_{b,b+1} \leq n_b - \lfloor \frac{r_b + 1}{2} \rfloor. \tag{6}$$

Taking into account the other constraints lead to a tighter bound on $\delta_{b,b+1}$, but requires a cumbersome formalism to count the number of constraints and unknowns. Requiring $\delta_{b,b+1} > 0$ gives the minimal size of each image set as:

$$n_b \geq \lfloor \frac{r_b + 1}{2} \rfloor + 1. \tag{7}$$

For instance, for a 2D rigid scene, *i.e.* $r = 2$, the minimal n_b is 2 from equation (7) and the maximal $\delta_{b,b+1}$ is 1 from equation (6), *i.e.* using the affine transformations over pairs of consecutive views is fine. For a 3D rigid scene, *i.e.* $r = 3$, the minimal n_b is 3 and the maximal $\delta_{b,b+1}$ is 1, meaning that using trifocal tensors over triplets of consecutive of views is fine[3].

In practice, we do not know the ranks r_b at this step. We tune an initial guess while jointly partitioning the sequence and computing the matching tensors, as described in §7.2.

6.2 Solving For the Implicit Cameras

The leading part. We solve for the non-rigid cameras using the closure constraints. Equation (5) gives the following constraints on the joint camera matrix $\mathcal{J}$: $\left(\mathsf{0}_{(d_b \times 2(i_b-1))}\ \mathcal{N}_{[i_b,i'_b]}^{\mathsf{T}}\ \mathsf{0}_{(d_b \times 2(n-i'_b))}\right)\mathcal{J} = \mathsf{0}$. Stacking the constraints for all $\{[i_b, i'_b]\}_{b=1}^{b=s}$ yields an homogeneous system $\mathsf{A}\mathcal{J} = \mathsf{0}$. It must be solved, *e.g.* in the least-squares sense, while ensuring that matrix $\mathcal{J}$ has full column rank: $\min_{\mathcal{J}} \|\mathsf{A}\mathcal{J}\|^2$ s.t. $\det(\mathcal{J}) \neq 0$. We replace the full column rank constraint by a column orthonormality constraint, *i.e.* $\mathcal{J}^{\mathsf{T}}\mathcal{J} = \mathrm{I}_{(r\times r)}$. Note that the latter implies the former. This is done without loss of generality since for any full column rank joint camera matrix $\mathcal{J}$, there exist several coordinate transformations, say $\mathsf{G}_{(r\times r)}$, such that $\mathcal{J}\mathsf{G}$ is column orthonormal. One such a transformation is given

[3] Triggs [9] states this result and shows the equivalence of using pairs of fundamental matrices over triplets of consecutive views.

by the QR decomposition of $\mathcal{J} = \mathcal{J}'\mathsf{G}^{-1}$. The transformed problem is solved by using the SVD $\mathsf{A} = \mathsf{U}\Sigma\mathsf{V}^\mathsf{T}$. Matrix $\mathcal{J}$ is given by the r last columns of V. Note that matrix A typically has a band-diagonal shape that one might exploit to efficiently compute its singular vectors, see *e.g.* [3].

The translations. The implicit imaging model (3) is $\mathbf{x}_{ij} = \mathsf{J}_i\mathbf{K}_j + \mathbf{t}_i$. By minimizing a least-squares error over all image points, the translations $\mathbf{t}_i$ in the joint translation vector $\mathbf{t}$, along with the basis shape vectors $\mathbf{K}_j$ can be reconstructed. We prefer to postpone the basis shape vector reconstruction to the next step, for robustness purposes. Instead, we consider the translation estimate $\mathbf{y}_{[i,i']}$ for each subsequence $[i, i']$, giving the centroid with respect to the points visible in the subsequence. We reconstruct these centroids along with vector $\mathbf{t}$. Note that in the absence of missing data, these centroids coincide. We minimize the reprojection error $\sum_{b=1}^{s} \|\mathbf{y}_{[i_b,i'_b]} - \mathcal{J}_{[i_b,i'_b]}\mathbf{Y}_{[i_b,i'_b]} - \mathbf{t}_{[i_b,i'_b]}\|^2$, where $\mathcal{J}_{[i,i']}$ and $\mathbf{t}_{[i,i']}$ are respectively a partial joint projection matrix and a partial joint translation vector restricted to the subsequence $[i, i']$, and $\mathbf{Y}_{[i,i']}$ is the reconstructed centroid. By expanding the cost function, the reprojection error is rewritten $\|\mathsf{A}\mathbf{w} - \mathbf{b}\|^2$, where the unknown vector $\mathbf{w}$ contains the $\mathbf{Y}_{[i_b,i'_b]}$ and $\mathbf{t}$. The solution is given by using the pseudo-inverse of matrix A, as $\mathbf{w} = \mathsf{A}^\dagger\mathbf{b}$. One must use a pseudo-inverse, since there is a r-dimensional ambiguity, making A rank deficient with a left nullspace of dimension r. This is a translational ambiguity between the basis shapes and the joint translation $\mathbf{t}$, that one can see by considering that $\forall\boldsymbol{\gamma} \in \mathbb{R}^r$, $\mathbf{x}_j = \mathcal{J}\mathbf{K}_j + \mathbf{t} = \mathcal{J}(\mathbf{K}_j - \boldsymbol{\gamma}) + \mathcal{J}\boldsymbol{\gamma} + \mathbf{t} = \mathcal{J}\mathbf{K}'_j + \mathbf{t}'$, with $\mathbf{K}'_j = \mathbf{K}_j - \boldsymbol{\gamma}$ and $\mathbf{t}' = \mathcal{J}\boldsymbol{\gamma} + \mathbf{t}$.

6.3 Reconstructing the Implicit Shape Points

We compute the basis shape vectors by non-rigid triangulation. This is done by minimizing the reprojection error. Assume that the j-th point is visible in the subsequence $[i, i']$, then this is formulated by $\min_{\mathbf{K}_j} \|\bar{\mathbf{x}}_{[i,i']} - \mathcal{J}_{[i,i']}\mathbf{K}_j\|^2$ with $\bar{\mathbf{x}}_{[i,i']} = \mathbf{x}_{[i,i']} - \mathbf{t}_{[i,i']}$. The solution is $\mathbf{K}_j = \mathcal{J}^\dagger_{[i,i']}\bar{\mathbf{x}}_{[i,i']}$. We perform the minimization in a robust manner to eliminate erroneous image points. We use a RANSAC-like algorithm with adaptive number of trials. The number of image points sampled in the inner loop is $\lfloor \frac{r}{2} \rfloor + 1$.

6.4 Nonlinear Refinement

We complete the reconstruction algorithm by minimizing the reprojection error in order to finely tune the estimate $\min_{\mathcal{J},\mathbf{t},\mathcal{K}} \|\mathcal{V}^+ \odot (\mathcal{X} - \mathcal{J}\mathcal{K} - \mathbf{t} \cdot \mathbf{1}^\mathsf{T})\|^2$ where $\mathcal{V}^+$ is obtained by duplicating[4] each row of the $(n \times m)$ visibility matrix $\mathcal{V}$. The minimization is done in a bundle adjustment manner. More precisely, we use a damped Gauss-Newton algorithm with a robust kernel. The damping is important to avoid singularities in the Hessian matrix, due to the $r(r+1)$ dimensional coordinate frame ambiguity. Contrarily to the explicit case, see [1, 11], no extra regularizing constraint is necessary.

[4] This is simply to make it the same size as $\mathcal{X}$.

7 Estimating the Non-Rigid Matching Tensors and Ranks

Our method estimates a non-rigid matching tensor over a (sub)sequence, *i.e.* for a complete measurement matrix, in a Maximum Likelihood framework. First, we tackle the case where the data do not contain outliers, and when the rank is given. Second, we examine the case where the data may contain outliers, and when the rank have to be estimated.

7.1 Outlier-Free Data, Known Rank

We describe a Maximum Likelihood Estimator, that handles minimal and redundent data. The translation $\mathbf{t}$ is obtained by averaging the point positions, and the measurement matrix is then centred as $\bar{\mathcal{X}} = \mathcal{X} - \mathbf{t} \cdot \mathbf{1}^\mathsf{T}$. The problem of finding the optimal $\mathcal{N}$ is formulated by $\min_{\hat{\mathcal{X}}} \|\bar{\mathcal{X}} - \hat{\mathcal{X}}\|^2$ s.t. $\mathcal{N}^\mathsf{T}\hat{\mathcal{X}} = \mathsf{0}$, where $\hat{\mathcal{X}}$ contains predicted point positions. This is a matrix approximation problem under rank deficiency constraint. It is solved by computing the SVD $\bar{\mathcal{X}} = \mathsf{U}\Sigma\mathsf{V}^\mathsf{T}$, from which $\hat{\mathcal{X}}$ is obtained by nullifying all but the r leading singular values in Σ and recomposing the SVD. Matrix $\mathcal{N}$ is given by the $2n - r$ last columns of U.

7.2 Contaminated Data, Unknown Rank

In most previous work, the rank of the sequence is assumed to be given. One exception is Aanæs *et al.* [1] who use the BIC model selection criterion to select the rank, but do not deal with blunders. When one uses subsequences, the subsequence rank may be lower than the sequence rank, and must be estimated along with the matching tensor. In addition, one has to deal with erroneous image points. We propose to use the robust estimator MAPSAC in conjunction with the GRIC model selection criterion proposed in [7]. GRIC is a modified BIC for robust least-squares problems. Our algorithm maximizes the GRIC score, as follows. In the inner loop of the robust estimator, we sample point tracks and not only compute a single matching tensor, but multiple ones by varying the rank. Obviously, an upper bound r_{max} on the rank is necessary to fix the number of point tracks that one samples at each trial. One must take into account that the computational cost rises with r_{max}. One possible solution is to divide the sequence of trials into groups using gradually narrower intervals of possible rank values. The GRIC score is given by $\text{GRIC} = \sum_{j=1}^{m} \rho\left(\frac{e_j^2}{\sigma^2}\right) + \lambda d + rm\log(m)$, where e_j is the prediction error for the j-th point track, $\lambda = 4d\log(z) - \log(2\pi\sigma^2)$ and z is chosen as the image side length. Function ρ is $\rho(x) = x$ for $x < t$ and $\rho(x) = t$ otherwise, where the threshold $t = 2\log(\theta) + d\lambda/(2n)$ with θ the ratio of the percentage of inliers to the percentage of outliers. The noise level is robustly estimated using the weakest model, *i.e.* for a tensor dimension $d = 1$, as $\sigma^2 = \text{med}(e_j^2)/0.6745^2$. We refer the reader to [7] for more details.

8 Experimental Results

Most other methods do not handle missing data, and hence can not be compared to our. The method from Torresani *et al.* [8] handles missing data but uses the explicit model.

	3	6	9	12	15	18
0%	3.82	6.06	8.48	11.28	13.82	16.22
10%	3.86	6.02	8.60	11.02	13.66	16.24
20%	3.72	5.98	8.48	11.20	13.84	16.44
30%	3.64	5.94	8.52	11.00	13.52	16.58
40%	3.60	5.98	8.44	11.00	13.58	16.28
50%	3.40	5.88	8.30	10.86	13.68	16.16

	3	6	9	12	15	18
0%	0.38	0.42	0.57	0.66	0.65	1.12
10%	0.35	0.37	0.49	0.65	0.55	1.14
20%	0.45	0.37	0.50	0.60	0.58	0.50
30%	0.48	0.37	0.57	0.53	0.61	0.67
40%	0.49	0.32	0.57	0.53	0.64	1.08
50%	0.49	0.62	0.70	0.63	0.71	1.17

Table 2. (left) Average estimated rank r and (right) its standard deviation σ_r versus the true rank $\underline{r}$ and percentage of outliers.

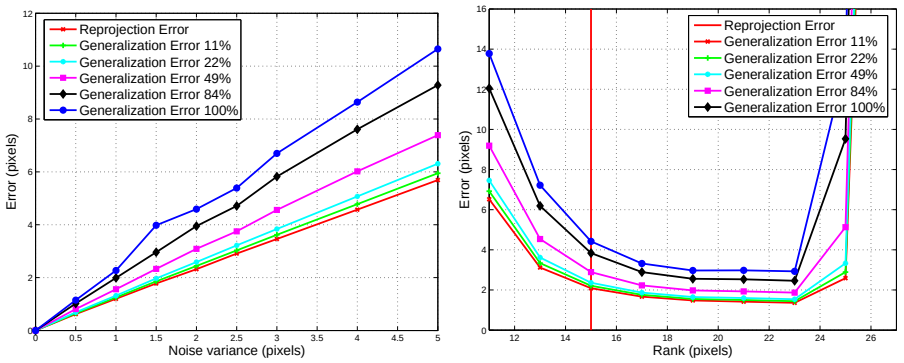
8.1 Simulated Data

We simulated $n = 180$ cameras observing a set of $m = 1000$ points generated from $l = 5$ basis shapes, hence with rank $\underline{r} = 3l = 15$. The configuration weights are chosen in order to give a decaying energy to successive deformation modes. The simulation setup produces a complete measurement matrix $\tilde{\mathcal{X}}$, from which we extract a sparse, band-diagonal measurement matrix $\mathcal{X}$, similar to what a real intensity-based point tracker would produce. A Gaussian centred noise with variance $\sigma^2 = 1$ is added to the image points.

In the experiments, we measured the *reprojection error* and the *generalization error*, which are dubbed in a machine learning context *training* and *test* error respectively. The reprojection error is $\mathcal{E} = \sqrt{\frac{1}{e}\|\mathcal{V}^+ \odot (\mathcal{X} - \mathcal{J}\mathcal{K} - \mathbf{t}\cdot\mathbf{1}^\mathsf{T})\|^2}$, where e is the total number of visible image points. In other words, the reprojection error reflects the difference between the measures and the predictions. The generalization error is given by $\mathcal{G}_\gamma = \sqrt{\frac{1}{e_\gamma}\|\tilde{\mathcal{V}}_\gamma^+ \odot (\tilde{\mathcal{X}} - \mathcal{J}\mathcal{K} - \mathbf{t}\cdot\mathbf{1}^\mathsf{T})\|^2}$, where γ indicates the percentage of hidden image points in $\tilde{\mathcal{X}}$ involved in the estimation and e_γ is the total number of image points used in the calculation. The $(n \times m)$ matrix $\tilde{\mathcal{V}}_\gamma$ indicates which image points are used in the calculation: it is constructed by including points further away from the visible points area while γ grows, *i.e.* $\tilde{\mathcal{V}}_0 = \mathcal{V}$ and $\tilde{\mathcal{V}}_{100} = 1_{(n\times m)}$. For example, $\mathcal{G}_0 = \mathcal{E}$ and $\mathcal{G}_{100} = \sqrt{\frac{1}{nm}\|\tilde{\mathcal{X}} - \mathcal{J}\mathcal{K} - \mathbf{t}\cdot\mathbf{1}^\mathsf{T}\|^2}$, *i.e.* all the visible and hidden image points are used to compute the error. Obviously, we expect the generalization error to be greater than the reprojection error, and to grow with γ.

The first experiment we performed consists in varying the level of added noise σ for different percentages γ of hidden points to compute the generalization error.

The results are shown on figure 1 (b). We observed that the reprojection error is slightly higher than the level of noise. The ability to generalize is accurate for a 1 pixel noise level, and smoothly degrades for larger noise levels, but is still reasonable: in the tested rang $\sigma = 0, \ldots, 5$ pixels, the $\gamma = 100\%$ generalization error is slightly higher than twice the noise level.



(a) (b)

Fig. 1. Reprojection and generalization error versus (a) the variance of added noise σ for different percentages γ of hidden points to compute the generalization error and (b) the rank r for different percentages γ of hidden points to compute the generalization error. The true rank $\underline{r} = 15$ is indicated with a vertical bar..

The second experiment we performed consists in varying the rank used in the computation, namely we tested $r = 11, \ldots, 27$, for different percentages γ of hidden points to compute the generalization error. The results are shown on figure 1 (b). We observed that it is preferable to overestimate rather than to underestimate the rank, up to some upper limit. A similar experiment with roughly equal magnitude configuration weights to generate the data shows that r can be slightly underestimated and largely overestimated. The conclusion is that in practice, overestimating the rank is safe.

The third experiment is devised to assess the quality of the rank estimation based on GRIC in the presence of outliers. We tested for true ranks in the range $r = 3, \ldots, 18$ which covers what one expects to meet in practice. The results we obtained are shown in table 2, which shows averages over 50 trials. We observed that these results are acceptable, even if the GRIC criterion we used is slightly biased since low ranks, *i.e.* less than 6, are slightly overestimated, while larger ranks, *i.e.* greater than 9 are slightly underestimated. It is however possible to correct for this bias in accordance with our conclusions on the previous experiment.

Fig. 2. (top) 5 out of the 154 frames and (bottom) the visibility matrix $\mathcal{V}$ for the 'Groundhog Day' sequence.

8.2 Real Data

We tested our algorithm on several image sequences. For one of them, extracted from the movie 'Groundhog Day', we show results. The sequence shows a man driving a car with a groundhog seated on his knees. The head of the man is rotating and deforming since he is speaking, and the animal is looking around, deforming its fur, opening and closing its mouth. Finally, the interior of the car is almost static, while the exterior is rigid, but moving with respect to the car.

The sequence contains 154 images, see figure 2 (top). We ran a KLT-like point tracker. We obtained a total of 1502 point tracks after having removed the small point tracks, namely which last less than 20 views. The visibility matrix, shown on figure 2 (bottom) is filled to 29.58%.

Fig. 3. (left) One frame with points and motion vectors reprojected from the reconstructed model and (right) Closeup on the actor, the groundhog and the background overlaid with points and motion vectors reprojected from the reconstructed model (white dots), original points (light grey squares) and outliers (dark grey diamonds).

For some parts of the sequence, where the motion of the different moving and deforming parts in the images is slow, computing the matching tensors is quite

easy. Indeed, blunders can clearly be detected and classified as outliers. However, other parts in the sequence contain significant motion between single frames and motion blur occurs, making the point tracks slightly diverging from their 'true' position, and making the detection of outliers difficult. Large illumination changes sometimes make the tracker fails for entire areas of the image.

The reprojection errors we obtained at the non-rigid matching tensors estimation stage were distributed between 0.5 and 0.9 pixels, and 0.65 pixels on average. We used a user-defined rank $r = 15$. The initialization step yielded 58021 inliers over 68413 image points, *i.e.* the inlier rate was 84.8%, with a reprojection error of 1.19 pixels. The robust bundle adjustment yielded 61151 inliers, *i.e.* the inlier rate was 89.4%, with a reprojection error of 0.99 pixels. We believe it is a successful result on this challenging image sequence.

9 Conclusions

We proposed an implicit imaging model for non-rigid scenes, from which we derived non-rigid matching tensors and closure constraints. Based on these theoretical concepts, we proposed a robust batch implicit Structure-From-Motion algorithm for monocular image sequences of non-rigid scenes, dealing with missing data and blunders. Future work will be devoted to comparing various model selection criteria, and segmenting the scene based on the configuration weights, to recover objects that move or deform independently.

References

1. H. Aanæs and F. Kahl. Estimation of deformable structure and motion. In *Proceedings of the Vision and Modelling of Dynamic Scenes Workshop*, 2002.
2. B. Bascle and A. Blake. Separability of pose and expression in facial tracking and animation. In *International Conference on Computer Vision*, 1998.
3. A. Björck. *Numerical Methods For Least-Squares Problems*. Society For Industrial and Applied Mathematics, 1996.
4. M. Brand. Morphable 3D models from video. In *CVPR*, 2001.
5. C. Bregler, A. Hertzmann, and H. Biermann. Recovering non-rigid 3D shape from image streams. In *Computer Vision and Pattern Recognition*, 2000.
6. M. Irani. Multi-frame optical flow estimation using subspace constraints. In *Proceedings of the International Conference on Computer Vision*, 1999.
7. P. H. S. Torr. Bayesian model estimation and selection for epipolar geometry and generic manifold fitting. *Int'l Journal of Computer Vision*, 50(1):27–45, 2002.
8. L. Torresani and A. Hertzmann. Automatic non-rigid 3D modeling from video. In *Proceedings of the European Conference on Computer Vision*, 2004.
9. B. Triggs. Linear projective reconstruction from matching tensors. *Image and Vision Computing*, 15(8):617–625, August 1997.
10. L. Wolf and A. Shashua. On projection matrices $P^k \rightarrow P^2$, $k = 3, \ldots, 6$, and their applications in computer vision. *Int'l J. of Computer Vision*, 48(1), June 2002.
11. J. Xiao, J.-X. Chai, and T. Kanade. A closed-form solution to non-rigid shape and motion recovery. In *European Conference on Computer Vision*, 2004.
12. J. Xiao and T. Kanade. Non-rigid shape and motion recovery: Degenerate deformations. In *Computer Vision and Pattern Recognition*, 2004.

7.1.3 Paper (JMIV'08) – *Implicit Non-Rigid Structure-from-Motion with Priors*

J Math Imaging Vis
DOI 10.1007/s10851-007-0060-3

Implicit Non-Rigid Structure-from-Motion with Priors

S.I. Olsen · A. Bartoli

© Springer Science+Business Media, LLC 2008

Abstract This paper describes an approach to implicit Non-Rigid Structure-from-Motion based on the low-rank shape model. The main contributions are the use of an implicit model, of matching tensors, a rank estimation procedure, and the theory and implementation of two smoothness priors. Contrarily to most previous methods, the proposed method is fully automatic: it handles a substantial amount of missing data as well as outlier contaminated data, and it automatically estimates the degree of deformation. A major problem in many previous methods is that they generalize badly. Although the estimated model fits the visible training data well, it often predicts the missing data badly. To improve generalization a temporal smoothness prior and a surface shape prior are developed. The temporal smoothness prior constrains the camera trajectory and the configuration weights to behave smoothly. The surface shape prior constrains consistently close image point tracks to have similar implicit structure. We propose an algorithm for achieving a Maximum A Posteriori (MAP) solution and show experimentally that the MAP-solution generalizes far better than the prior-free Maximum Likelihood (ML) solution.

Keywords Computer vision · Structure-from-motion · Non-rigid · Low-rank shape

This paper combines and extends two conference papers; the first one appeared in the Workshop on Dynamical Vision held at ICCV'05 [2], and the second one appeared at the 2007 British Machine Vision Conference [11]. This paper integrates the two publications into a comprehensive presentation of our approach to Non-Rigid Structure-from-Motion.

S.I. Olsen (✉)
Department of Computer Science, University of Copenhagen, Universitetsparken 1, 2100 Copenhagen Ø, Denmark
e-mail: ingvor@diku.dk

A. Bartoli
LASMEA (CNRS / UBP), Clermont-Ferrand, France
e-mail: adrien.bartoli@gmail.com

1 Introduction

Non-rigid Structure-from-Motion concerns the simultaneous recovery of the deforming world structure and camera motion from image features. This extends the classical rigid Structure-from-Motion [13] to situations with deforming scenes such as expressive faces, deforming bodies *etc*. A major step forward was made by Bregler *et al.* [5], Brand [3] and Aanæs *et al.* [1]. They represent non-rigidity as a linear combination of a small number of 3D *basis shapes*. The *low-rank shape* model is generic: it does not prescribe any particular type of 3D shape or deformation. Using only limited assumptions it allows a simultaneous recovery of 3D deformable shape and camera motion from monocular videos. Xiao *et al.* [23] studied the deformations that may defeat the reconstruction algorithms. Apart from [1, 2], most methods assume that the amount of non-rigidity—the number l of basis shapes—is known. If l is underestimated the deformation cannot be well modeled, and if overestimated the model will contain too many parameters. In the latter case the model will fit the noise in the data, and will not generalize well.

The data used in this and most previous methods consist of point coordinates obtained by tracking image interest points through a sequence. Because of occlusions and imperfect tracking the registered data often is partial: some or all points are visible only in a subset of the frames. Many early methods could not handle situations with missing data

[1, 3, 5, 16, 21, 22]. Recently a number of methods [6, 10] have been proposed to handle the missing data problem for the rigid Structure-from-Motion problem.

Estimating a model from partial data allows one to predict the projection of all world points on all images. The model generalizes well if the predicted points, on frames where the point is not registered, are accurate. If the degree of deformation is overestimated the model is unlikely to generalize well. Priors have been shown to improve generalization. In [7] a prior for rigidity was presented. In [18] a probabilistic PCA model using hierarchical priors is applied to avoid overfitting.

The present paper draws on and extends our previous work [2, 11]. It gives an in-depth description of the implicit Maximum-Likelihood (ML) approach to non-rigid Structure-from-Motion [2], and its extension with temporal and shape smoothness priors [11], by which a Maximum A Posteriori (MAP) solution is formulated. The proposed MAP-estimator is based on four main steps: an initial solution is computed by using an ML-estimator minimizing the reprojection error. Second, the implicit coordinate frame is changed to maximize a temporal smoothness prior. Third, the implicit structure is re-estimated by minimizing a combination of the reprojection error and a surface shape prior. Finally, the motion and structure estimates are jointly refined by nonlinear optimization. The paper reports results on simulated and real data. It shows that the generalization ability of the MAP solution is greatly improved compared to the standard ML-estimation. Experiments show that tracks split by imperfect tracking can be glued correctly together.

Contributions Among the various methods using the low-rank shape model for non-rigid Structure-from-Motion, our framework is the first one to bring all of the following features:

- *Missing image points.* Most of the other methods are based on SVD to factorize a measurement matrix, *e.g.* [1, 3–5, 7, 17], and thus do not deal with missing image points. Our framework handles cases for which only a few percents of the image points are observed (for instance, we successfully handle a sequence with only about 13% of the image points being observed), meaning that extreme occlusions and highly dynamical scenes can be handled. This is achieved thanks to the low-rank matching tensors and closure constraints we propose.
- *Robustness.* Most of the other methods assume that the noise on the image point positions follows a centered Gaussian *i.i.d.* distribution, *e.g.* [1, 4, 5, 7, 18]. While this might be a sensible assumption if the points are manually clicked or at least checked by the user, this certainly is not true if the output of an automatic KLT-like point tracker is directly used as input. The points may drift from the ideal track, and may also be totally mismatched. Our algorithm outputs, for each image point, a binary variable indicating if it is an inlier or an outlier with respect to the low-rank shape model.
- *Rank selection.* Most of the other methods assume that the rank, *i.e.* the degree of deformation, is known, *e.g.* [4, 5, 7, 18]. This is definitely not a realistic assumption, since the rank is highly dependent on the scene content. Inspired by the GRIC model selection criterion, a robustified BIC, our algorithm computes the rank from the available data automatically.
- *Generic prior knowledge.* Most other methods assume the low-rank shape model as the only generic prior, *i.e.* the scene shape deforms according to a finite set of 'few' deformation modes, *e.g.* [1, 5, 7]. This clearly is not enough to obtain a model that will generalize well to the entire sequence when only a small fraction of the data is available. Natural generic priors such as smooth camera motion, shape deformation and continuous surface shapes, are easily included in our framework. We show that the high generalization ability of the recovered model allows us to glue point tracks split during tracking, due to *e.g.* an occlusion or a tracking failure.

Our framework is entirely automatic, as it takes as input the point tracks produced by some point tracker, computes the rank, classify each image point as valid or erroneous, and outputs the sought after implicit reconstruction. A further step is to upgrade the implicit reconstruction to an explicit, *i.e.* metric one, which has been described in details in several recent papers [4, 22].

Organization of the Paper Section 2 reviews the implicit low-rank imaging model, its matching tensors and closure constraints. In Sect. 3 we derive a method for estimating the degree of deformation. In Sect. 4 model estimation on partial data is described. Sections 5 and 6 describe the proposed priors and their implementation. Section 7 reports the experimental results. Finally, Sect. 8 concludes the paper.

Notation Vectors are denoted using bold fonts, *e.g.* $\mathbf{x}$ and matrices using sans-serif or calligraphic characters, *e.g.* M or $\mathcal{A}$. Index $i = 1, \ldots, N$ is used for the images, $j = 1, \ldots, M$ for the points. The Hadamard (element-wise) product is written $\odot$. Bars indicate 'centered' data, as in $\bar{\mathsf{X}}$. We use the Singular Value Decomposition, denoted SVD, *e.g.* $\mathsf{X} = \mathsf{U}\Sigma\mathsf{V}^\mathsf{T}$ where U and V are orthonormal matrices, and Σ is diagonal, containing the singular values of X in decreasing order. Operator vect(X) performs column-wise matrix vectorization.

2 The Implicit Low-Rank Non-Rigid Model

The standard rigid model describes the affine projection $\mathbf{x}_{ij}$ of a set of M 3D world points $\mathbf{W}_j$, represented by a $3 \times M$ shape matrix W onto N images represented by a $2N \times 3$ motion matrix P of stacked 2×3 affine camera projection matrices P_i:

$$\mathbf{x}_{ij} = \mathsf{P}_i \mathbf{W}_j + \mathbf{t}_i + \eta_{ij}, \tag{1}$$

where $\mathbf{t}_i$ is the position of the i-th camera and η_{ij} is a noise term. The $2N \times M$ matrix X of time varying coordinates $\mathbf{x}_{ij}$ is called the *measurement matrix* and has rank $r = 3$ [13]. In the non-rigid case, $r > 3$. The low-rank assumption is $r \ll \min\{2N, M\}$. In *explicit* non-rigid models it is assumed that the shape points $\mathbf{W}_{ij}$ can be written as linear combinations over l *basis shapes* $\mathbf{B}_{kj}$ with the *configuration weights* α_{ik}: $\mathbf{W}_{ij} = \sum_{k=1}^{l} \alpha_{ik} \mathbf{B}_{kj}$. The explicit imaging model is:

$$\mathbf{x}_{ij} = \mathsf{P}_i \left(\sum_{k=1}^{l} \alpha_{ik} \mathbf{B}_{kj} \right) + \mathbf{t}_i + \eta_{ij}. \tag{2}$$

Defining $\mathbf{K}_j^{\mathsf{T}} = (\mathbf{B}_{1j}^{\mathsf{T}} \ \cdots \ \mathbf{B}_{lj}^{\mathsf{T}})$, and $\mathsf{M}_i = (\alpha_{i1}\mathsf{P}_i \ \cdots \ \alpha_{il}\mathsf{P}_i)$ the model writes $\mathbf{x}_{ij} = \mathsf{M}_i \mathbf{K}_{ij} + \mathbf{t}_i + \eta_{ij}$ and thus $\mathsf{X} = \mathsf{MK} + \mathbf{t}\mathbf{1}^{\mathsf{T}} + \eta$ where M and K are $2N \times 3l$ and $3l \times M$ matrices and $\mathbf{1}$ is a vector of ones.

Replacing the model rank assumption $3l$ with the more general assumption $r = 3l$ being a positive integer, introducing a $(r \times r)$ full-rank matrix $\mathcal{A}$ and relaxing the replicated structure of the explicit motion matrix M_i gives the *implicit model*:

$$\begin{aligned} \mathbf{x}_{ij} &= \mathsf{M}_i \mathbf{K}_j + \mathbf{t}_i + \eta_{ij} \\ &= (\mathsf{M}_i \mathcal{A})\,(\mathcal{A}^{-1} \mathbf{K}_j) + \mathbf{t}_i + \eta_{ij} = \mathsf{J}_i \mathbf{S}_j + \mathbf{t}_i + \eta_{ij} \end{aligned} \tag{3}$$

and thus:

$$\mathsf{X} = \mathsf{JS} + \mathbf{t}\mathbf{1}^{\mathsf{T}} + \eta,$$

where the $2N \times r$ matrix $\mathsf{J} = \mathsf{M}\mathcal{A}$ is called the *implicit motion matrix*, and where the $r \times M$ matrix $\mathsf{S} = \mathcal{A}^{-1}\mathsf{K}$ is named the *implicit shape matrix*. The matrix $\mathcal{A}$ in (3) often is called *the mixing matrix* and represents a *corrective transformation* by which an implicit model can be upgraded to an explicit one. $\mathcal{A}$ defines the implicit coordinate frame in which the motion and the shapes are represented.

The model (3) is called implicit because no assumption is made on the replicated block structure of the motion matrices that often is used in explicit approaches *e.g.* [4, 5, 17]. Thus the implicit model is simpler than the explicit one but gives weaker constraints on point tracks. Note that the implicit (basis) shape vectors $\mathbf{S}_j$ are more difficult to interpret in terms of world coordinates. Similarly, the implicit motion matrices J_i (comprising camera pose and configuration weights) do no longer directly relate to the camera orientation. Here we assume that r is known. In Sect. 3 we describe how r is estimated.

Because the factorization of X is ambiguous, due to the freedom of choosing $\mathcal{A}$, an upgrading from implicit to explicit representation is important. Xiao *et al.* [22] show that constraints on both the explicit motion and shape matrices must be considered to achieve a unique solution, namely the 'rotation' and the 'basis' constraints. They give a closed-form solution based on these constraints. In [4] Brand presents an alternative less noise sensitive method without the 'basis' constraints. We consider the upgrading as a postprocessing step that is not further dealt with in this paper.

Our goal is thus, given X with missing and erroneous elements, to recover J, S, and $\mathbf{t}$ while detecting the erroneous elements, and predicting the missing ones. If X is complete (no missing data), one approximate factorization can be found using SVD as $\bar{\mathsf{X}} = \mathsf{U}\Sigma\mathsf{V}^{\mathsf{T}}$, where $\bar{\mathsf{X}}$ is the centered measurement matrix, *i.e.* with the translational part being canceled. The implicit motion and shape matrices J and S, are recovered as the r leading columns of *e.g.* U and the rows of $\Sigma\mathsf{V}^{\mathsf{T}}$ respectively. The assumption is that the information in the $d = 2N - r$ dimensional discarded subspace corresponds to the noise η. This method however has limited interest in practice since real data almost always contain errors and missing points. We propose a method that deals with this kind of measurements. It is based on extending the rigid matching tensors [20] to the low-rank shape model—we call them *low-rank matching tensors*. Matching tensors relate corresponding points over multiple images. Examples are the fundamental matrix and the trifocal tensor. In the non-rigid affine case the matching tensor is a $2N \times d$ matrix $\mathcal{N}$ whose columns span the d dimensional left nullspace of the centered measurement matrix $\bar{\mathsf{X}}$:

$$\mathcal{N}^{\mathsf{T}} \bar{\mathsf{X}} = 0. \tag{4}$$

As before $\mathcal{N}$ can be estimated using SVD. The closure constraints relate matching tensors to projection matrices. From (1) and (4) and for all implicit shape points $\mathbf{S}_j \in \mathbb{R}^r$ we have $\mathcal{N}^{\mathsf{T}}\mathsf{J}\mathbf{S}_j = \mathbf{0}$, which gives our $\mathcal{N}$*-closure constraint*:

$$\mathcal{N}^{\mathsf{T}} \mathsf{J} = 0. \tag{5}$$

The implicit motion matrix J consequently lies in the right nullspace of $\mathcal{N}^{\mathsf{T}}$ and may be estimated using an SVD. From J, $\mathbf{S}_j$ can be retrieved point-wise by triangulation. From $\mathbf{x}_j = \mathsf{J}\mathbf{S}_j$ we get $\mathbf{S}_j = \mathsf{J}^{\dagger}\mathbf{x}_j$, where $\mathsf{J}^{\dagger}$ is the pseudoinverse of J. In case of outlier contaminated data the computation of $\mathcal{N}$ as well as the triangulation must be done robustly so that blunders do not corrupt the computation. We use a RANSAC-based approach called MSAC [15].

3 Estimating the Rank

Estimating the rank r of the measurement matrix is of utmost importance. If r is chosen too small the model will not be able to express the deformations; if chosen too large the model will fit the noise. For many real sequences the transition between the singular value subspaces containing deformation information and those containing noise is blurred. This makes a guessing of r difficult. For explicit models (using $l = \lceil \frac{r}{3} \rceil$) an upgrade to a metric model may be difficult if l is selected too large [4]. In [18] it is argued that if appropriate priors are used an overestimation of r is not severe. We have made similar observations using the priors described in Sect. 5. However still a good guess of r is needed.

Most previous work assumes that the rank of X is given. A simple rank estimation by thresholding the singular value spectrum is used in [24]. If outliers corrupt the data or if the energy of the weakest non-rigid components is comparable in magnitude to the noise then such methods are unlikely to work. In [12] a deformation index is based on the correlation matrix of the in-frame position information. In [1] the Bayes Information Criteria (BIC) is used for rank selection. We propose to use the GRIC model selection criterion proposed in [14]. GRIC is a robustified version of BIC. Let k be the number of parameters of the model and $\mathcal{L}$ the log-likelihood of the error distribution, both functions of r. Then we aim at selecting the r minimizing $-2\mathcal{L} + k\log(M)$. In GRIC the error distribution is obtained from a mixture between a Gaussian inlier part and a uniform outlier part:

$$\mathrm{P} = \gamma \mathrm{P}_{\mathrm{in}} + (1-\gamma)\mathrm{P}_{\mathrm{out}} \tag{6}$$

$$= \frac{\gamma}{c}\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^{2N-r} \exp\left(-\frac{e^2}{2\sigma^2}\right) + \frac{1-\gamma}{v}, \tag{7}$$

where the error of the fit for the inliers can be modeled by an isotropic zero mean Gaussian. Here e is the $2N-r$ dimensional error perpendicular to the fitting manifold and $\frac{1}{c}$ is a prior of the point track, assuming a uniform distribution on the volume (with size c) on which an observation may occur. v similarly is the volume of space in which an outlier can occur. When, for a point track, $\frac{e^2}{\sigma^2}$ exceeds a value T, the track can be classified as being more probable to belong to the outlier distribution than to the inlier distribution. It is easy to show that:

$$T = 2\log\left(\frac{\gamma}{1-\gamma}\right) + (2N-r)\lambda, \tag{8}$$

where

$$\lambda = 2\log(U) - \log(2\pi\sigma^2), \tag{9}$$

$$U = \left(\frac{v}{c}\right)^{\frac{1}{2N-r}}. \tag{10}$$

Replacing the mixture model with a maximization approach, and using:

$$\rho\left(\frac{e2}{\sigma^2}\right) = \begin{cases} \frac{e2}{\sigma^2} & \text{if } \frac{e2}{\sigma^2} \le T, \\ T & \text{if } \frac{e2}{\sigma^2} > T \end{cases} \tag{11}$$

the log-likelihood term $-2\mathcal{L} = -2\log(P)$ can be shown to equate:

$$\sum_i^M \rho\left(\frac{e2}{\sigma^2}\right) - 2M\log\left(\frac{\gamma}{c}\right) - M(2N-r)\lambda, \tag{12}$$

where we have assumed independence of the M observations. Because the matching tensor has $d = 2N-r$ equations in $2N$ coordinates, and because the equations are homogeneous and orthogonal we have:

$$k = 2Nd - d - \frac{d(d-1)}{2} = (2N^2 - N) - \frac{1}{2}r(r-1). \tag{13}$$

Ignoring the constant terms $2M\log(\frac{\gamma}{c})$, $(2N^2-N)\log(M)$, and $2MN\lambda$ the GRIC measure becomes:

$$\mathrm{GRIC} = \sum_{j=1}^{M} \rho\left(\frac{e_j^2}{\sigma^2}\right) + Mr\lambda - \frac{1}{2}r(r-1)\log(M). \tag{14}$$

It is clear that the r minimizing this equation depends on the value of U defined in (10). To avoid estimating U we notice that an often used alternative approach to the estimation of T is by the value of the inverse cumulative χ^2 distribution with $2N-r$ degrees of freedom [8]. For relevant values of $2N-r$ this is approximately linear with a slope of λ. Thus we estimate λ directly.

Because the data may contain outliers we use the robust estimator MSAC [15] in conjunction with the GRIC. Thus to find the rank value minimizing GRIC we must sample this repeatedly for all relevant values of r. To limit the computational cost the sequence of trials is divided into groups using gradually narrower intervals of possible rank values.

As described in the next section a limitation to partial data forces the rank-value analysis to be made in (overlapping) frame intervals. From such block-rank values a global one may then be estimated, *e.g.* by the maximum. In general the block-rank values will be smaller than the global rank value, because no block may contain the complexity of deformation present in the full sequence. If the frame span for each block is very small the underestimate may be severe. In the next section a heuristic for selecting the block size is discussed. Here the deformation within each block is assumed sufficiently representative for the deformation within the compound sequence. In this case the maximum may be a good rank estimate, because it is the largest underestimate that there is evidence for. One alternative would be to apply a robust maximum operator.

J Math Imaging Vis

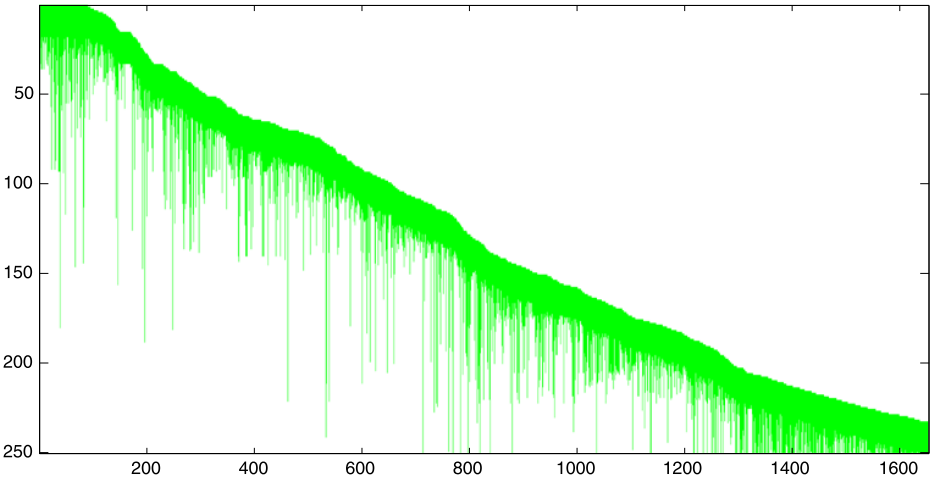


Fig. 1 1654 tracks in 250 frames. The degree of visibility is 13.26%

4 Handling Partial Data

Because of occlusions and imperfect tracking the measurement matrix X often will be defined only in a diagonal band. Figure 1 shows an example of the visibility matrix $\mathcal{V}$ for a sequence with 250 frames. $\mathcal{V}$ is a $N \times M$ binary matrix associated to X specifying if $\mathbf{x}_{ij}$ is defined or is missing. For practical use, handling of partial data is of great importance. A direct application of SVD, as described in the previous sections and used in a large number of previous methods [1, 3, 5, 16, 22], is not possible. One method to proceed is to extract blocks of complete data from the diagonal band of X [9, 10]. A survey of this and other methods for partial data handling can be found in [6]. As discussed in a following section we need an initial factorization of X which we then can improve by an iterative non-linear refinement step. For this purpose an application of the matching tensor approach on a block partitioning is ideal.

Given r, a d dimensional matching tensor $\mathcal{N}_b$ can be computed robustly for each block b. For each matching tensor, (5) gives a closure constraint on the joint motion matrix J:

$$\left(\mathsf{0}_{(d\times 2(i_b-1))} \quad \mathcal{N}_b^{\mathsf{T}} \quad \mathsf{0}_{(d\times 2(N-i'_b))}\right)\mathsf{J} = \mathsf{0}, \tag{15}$$

where i_b and i'_b are indexes of the first and last frame in block b. Stacking the constraints for all blocks yields an homogeneous linear least squares problem $\|\mathsf{AJ}\|^2$ which must be solved such that J has full column rank. Without loss of generality the full column rank constraint can be replaced by constraining J to be column orthonormal. A solution is given by the r last columns of V in the SVD $\mathsf{A} = \mathsf{U}\Sigma\mathsf{V}^{\mathsf{T}}$.

For each block the translation vector $\mathbf{t}^b$ is computed prior to $\mathcal{N}_b$. The joint translation vector $\mathbf{t}$ can be found by minimizing the reprojection error $\sum_b \|\mathbf{t}^b - \mathsf{J}_b\mathbf{T}_b - \mathbf{t}_b\|^2$, where $\mathbf{T}$ is the reconstructed centroid, and where the subscript b in J_b, $\mathbf{T}_b$, and $\mathbf{t}_b$ denotes the restriction of the joint matrices and vectors to the frames within block b. The reprojection error is rewritten $\|\mathsf{B}\mathbf{w} - \mathbf{b}\|^2$, where the unknown vector $\mathbf{w}$ contains $\mathbf{T}$ and $\mathbf{t}$. The solution is given by using the pseudo-inverse since there is an r dimensional ambiguity, making B rank deficient with a left nullspace of dimension r. This correspond to the translational ambiguity between the basis shapes and the joint translation $\mathbf{t}$: $\forall \boldsymbol{\gamma} \in \mathbb{R}^r$, $\mathbf{x}_j = \mathsf{J}\mathbf{S}_j + \mathbf{t} = \mathsf{J}(\mathbf{S}_j - \boldsymbol{\gamma}) + \mathsf{J}\boldsymbol{\gamma} + \mathbf{t} = \mathsf{J}\mathbf{S}'_j + \mathbf{t}'$.

Given the estimates of J and $\mathbf{t}$, the shape vectors $\mathbf{S}_j$ now can be computed by a robust minimization of the reprojection error. An advantage is that this makes possible a detection of outliers. Alternatively, as described in Sect. 6.2, computation of $\mathbf{S}_j$ may be postponed until the prior information is included.

As discussed in the previous section it is an advantage for the rank estimation if the data partitioning is made to maximize the block frame span. However, increasing this span will decrease the number of tracks visible in all of the frames within the block. As a minimum M_b must be larger than $2N_b$, where N_b and M_b are the block frame span and the number of tracks within the block. Because a RANSAC- based estimation is used $M_b \gg 2N_b$ is preferred. One heuristic is to choose N_b by the maximal value such that $M_b > 4N_b$. In practice this may not be possible if $N \gg M$, if only very few data is visible, or if some tracks are very short. Often is an advantage to eliminate tracks shorter than 10–20 frames. To guide the block partitioning one heuristic is to start with the previously mentioned choice of $M_b = 4N_b$, and then decrease or increase N_b, still requiring $M_b > 2N_b$, towards a situation where the block shape becomes similar to the shape of the measurement matrix itself.

5 The Priors

Prior knowledge on both motion and shape can be very useful in Non-Rigid Structure-from-Motion. In [7, 19] the analysis is bootstrapped from an assumption of a rigid scene. More basis shapes are incrementally added if necessary. In [24] a prior matrix is built from observed trajectories. Using a spectral clustering method a RANSAC-based motion segmentation is then derived. In [18] a probabilistic principal component analysis is used as an hierarchical Bayesian prior. The method makes possible a simultaneous estimation of 3D shape and motion, and of the deformation model. A Gaussian prior is put on the configuration weights α_{ik} in (2). Thus, the shapes are sought as similar as possible to each other. To a large degree the purpose of this approach and our surface shape prior (see below) are similar.

It is generally recognized [4, 23] that upgrading an initial factorization to a metric one, *i.e.* estimating the mixing matrix, is a hard problem. The methods in [4, 23] both rely on a non-trivial optimization step. The global optimum is rarely found unless the optimization is initialized at a point close to it. Thus an initial choice of coordinate frame according to a prior may show crucial for successful upgrading.

For nonlinear models with many parameters often many completely different parameter settings may result in fits which are approximately equally good when measured on the training data. However, when measured on data held back for test usage some solutions may predict these much better that others. Such solutions are said to generalize better. Often it is an advantage to select a solution that generalizes well but have a slightly worse reprojection error than the other way around. Below we motivate and formulate a temporal smoothness prior and a surface shape prior, both intended to improve generalization.

5.1 Temporal Smoothness Prior

For most image sequences, the camera motion is smooth. For points on a smoothly deforming surface the configuration weights smoothly vary as well which means that the surface does not 'jump' between poses but rather smoothly interpolates them. Since both the configuration weights and the camera parameters are encapsulated in the J_i matrices, these should vary smoothly from frame to frame giving the smoothness measure:

$$\mathcal{E}_{\mathrm{J}}(\mathsf{J}) = \sum_{i=1}^{N-1} \|\mathsf{J}_i - \mathsf{J}_{i+1}\|^2 = \|\mathsf{L}\|^2, \tag{16}$$

where L is the $2(N-1) \times r$ matrix of stacked projection difference matrices. The previously described factorization is ambiguous up to an $r \times r$ full rank mixing matrix $\mathcal{A}$. From (16) we see that $\mathcal{E}_{\mathrm{J}}(\mathsf{J}) \neq \mathcal{E}_{\mathrm{J}}(\mathsf{J}\mathcal{A})$. This suggests to select the $\mathcal{A}$ minimizing (16). Note that since $\mathcal{E}_{\mathrm{J}}(\mathsf{J}\mathcal{R})$ is invariant to any orthonormal matrix $\mathcal{R}$ we will not totally fix the mixing matrix but leave freedom for any orthogonal transform.

5.2 Surface Shape Prior

Points which are close in space also project closely on the images. In case of points on a deforming continuous surface the opposite is true as well. Solutions obtained by the previously described method does not encourage such behavior. As a consequence the projected trajectories for such close tracks may deviate significantly outside the estimation area. Often the ability to generalize acceptably disappears just 2–5 frames away from the images in which the points are visible. To improve generalization a surface shape prior is imposed.

First notice that without fixing the coordinate frame in which the shapes in S are represented the usual norm distance between two shapes is meaningless. However having fixed the mixing matrix (up to an orthogonal matrix) it becomes meaningful.

The shape similarity $\alpha(j_1, j_2)$ of two point tracks $j_1 \neq j_2$ is measured by a decreasing function of a distance measure $d(j_1, j_2)$ between the point tracks. The surface shape prior then is:

$$\mathcal{E}_{\mathrm{S}}(\mathsf{S}) = \sum_{(j_1,j_2)\in\Omega} \alpha(j_1, j_2) \cdot \|\mathbf{S}_{j_1} - \mathbf{S}_{j_2}\|^2, \tag{17}$$

where Ω is the set of track tuples simultaneously visible for a minimum number of, say 10, frames. As for the shape similarity a Gaussian $\alpha(j_1, j_2) = \exp(-\frac{d(j_1,j_2)^2}{2\sigma^2})$ is appropriate. In the experiments we computed σ as 0.03 times the image width. One measure of track distance is the maximum: $d(j_1, j_2) = \max_i\{\|\mathbf{x}_{ij_1} - \mathbf{x}_{ij_2}\|_2\}$. Alternative measures include robust estimates of the average or maximum point track distance. As for the temporal smoothness prior, $\mathcal{E}_{\mathrm{S}}$ is invariant to any orthonormal matrix $\mathcal{R}$.

6 Non-Rigid Structure-from-Motion with Priors

The model simultaneously minimizing the reprojection error, the smoothness prior and the surface shape prior, *i.e.* the cost:

$$\mathcal{E}_{\mathrm{RE}}(\mathsf{J}, \mathsf{S}) + \gamma\mathcal{E}_{\mathrm{J}}(\mathsf{J}) + \beta\mathcal{E}_{\mathrm{S}}(\mathsf{S}) \tag{18}$$

must be minimized by nonlinear optimization. To ensure a good starting point, and because the coordinate frame in which the shapes are represented influences the solution through the priors, we choose (initially) this frame by minimizing the temporal smoothness prior. This fixes the mixing matrix up to an orthogonal matrix, to which the surface

J Math Imaging Vis

Table 1 Summary of our non-rigid low-rank implicit structure-from-motion algorithm

OBJECTIVE

Given M point tracks over N images as a possibly incomplete $(2N \times M)$ measurement matrix X, compute the implicit non-rigid motion J_i, the implicit non-rigid shape points $\mathbf{S}_j$, and an estimate of the rank r. Classify each image point as an inlier or an outlier.

ALGORITHM

1. Partition the sequence into overlapping blocks with complete data (Sect. 4). For each block, robustly estimate the block rank and the associated matching tensor (Sects. 2 and 3).
2. Estimate the global rank r: apply the closure constraints to solve for the joint implicit motion matrix J and the joint translation vector $\mathbf{t}$ (Sect. 4).
3. Detect the outliers by robustly fitting the model to each point track using RANSAC.
4. Use the temporal smoothness prior to select the coordinate transformation $\mathcal{A}$ and apply this to J (Sects. 5.1 and 6.1).
5. Estimate the shape vectors $\mathbf{S}_j$ minimizing a weighted sum of the reprojection error and the shape smoothness prior measure (Sects. 5.2 and 6.2).
6. Nonlinearly refine the implicit motion and shape points by minimizing a combination of the reprojection error, the temporal smoothness measure and the shape smoothness measure.
7. Estimate the missing data and glue tracks if they comply with the estimation.

shape prior is invariant. Next, by using the surface shape prior an initial guess for S is obtained. Finally J and S are jointly refined by nonlinear least-squares optimization. The constants γ and β in (18) are chosen *ad hoc* such that the two priors initially contribute relative to the reprojection error with certain amounts, say 0.2 and 0.02. Below, the initial application of the two priors is described. The algorithm is summarized in Table 1.

6.1 The Coordinate Frame

The temporal smoothness prior measure (16) obviously depends on the mixing matrix. Consequently we (partially) determine this as the $r \times r$ full rank matrix $\mathcal{A}$ minimizing $\mathcal{E}_{\mathsf{J}}(\mathsf{J}\mathcal{A}) = \|\mathsf{L}\mathcal{A}\|^2$. The motivation is that determining the mixing matrix ensures that the camera motion is 'close' to the optimal one. To avoid the shrinking effect of reducing the prior value by simply scaling down J we require $\det(\mathcal{A}) = 1$. Let $\mathsf{L} = \mathsf{U}\Sigma\mathsf{V}^\mathsf{T}$ be an SVD of L. A closed-form solution for $\mathcal{A}$ is provided in the Appendix.

$$\mathcal{A} = \left(\sqrt[r]{\prod_{k=1}^{r} \sigma_k} \right) \mathsf{V}\Sigma^{-1}. \tag{19}$$

Given $\mathcal{A}$ we change the coordinate frame by $\mathsf{J} \leftarrow \mathsf{J}\mathcal{A}$ and $\mathsf{S} \leftarrow \mathcal{A}^{-1}\mathsf{S}$ without changing the reprojection error. However the value of the prior $\mathcal{E}_{\mathsf{J}}(\mathsf{J})$ is significantly reduced.

6.2 Surface Shape Prior Implementation

Having fixed the non-rotational part of the mixing matrix it becomes meaningful to compute an estimate of the structure S. Given the modified joint motion matrix J, S is sought to minimize a weighted sum of the reprojection error and the surface shape prior:

$$\begin{aligned}\mathcal{E}_{\mathrm{RE}}(\mathsf{J},\mathsf{S}) + \beta\mathcal{E}_{\mathsf{S}}(\mathsf{S}) &= \|\mathcal{V} \odot (\mathsf{X} - \mathsf{J}\mathsf{S} - \mathbf{t}\mathbf{1}^\mathsf{T})\|^2 \\ &\quad + \beta \sum_{(j_1,j_2)\in\Omega} \alpha(j_1,j_2) \cdot \|\mathbf{S}_{j_1} - \mathbf{S}_{j_2}\|^2,\end{aligned} \tag{20}$$

where $\mathcal{V}$ is the combined inlier and visibility matrix and Ω is the set of 'close' point tracks. The S minimizing this expression leads to a larger reprojection error compared to the initial solution. The reprojection error increases with β. We choose a value of β such that the reprojection error either remains below say 2 pixels or is increased by a factor smaller than say 0.5. Since the result is not sensitive to an accurate value of β an approximate value is found using an iterative approach with only few iterations. Equation (20) can be rewritten:

$$\mathcal{E}_{\mathrm{RE}}(\mathsf{J},\mathsf{S}) + \beta\mathcal{E}_{\mathsf{S}}(\mathsf{S}) = \|\mathbf{v} \cdot (\bar{\mathbf{x}} - \mathcal{M}\mathbf{s})\|^2 + \beta\|\mathcal{L}\mathbf{s}\|^2, \tag{21}$$

where $\bar{\mathbf{x}} = \mathrm{vect}(\bar{\mathsf{X}})$ and $\mathbf{s} = \mathrm{vect}(\mathsf{S})$. $\mathcal{M} = \mathrm{diag}_M(\mathsf{J})$ is a $(2NM) \times (rM)$ block diagonal matrix with M repetitions of J. If $p = |\Omega|$ is the number of 'close' pairs of tracks then $\mathcal{L}$ has p row blocks $\mathcal{L}_{(j_1,j_2)}$ of the form:

$$\mathcal{L}_{(j_1,j_2)} = \alpha(j_1,j_2) \cdot (\mathsf{0}\ldots\mathsf{0}, \mathrm{I}, \mathsf{0}\ldots\mathsf{0}, -\mathrm{I}, \mathsf{0}\ldots\mathsf{0}), \tag{22}$$

where I and $\mathsf{0}$ are the $r \times r$ identity and zero matrices, and where the position of the two identity matrices correspond to the indexes j_1 and j_2. Thus $\mathcal{L}$ has size $(rp) \times (rM)$. With this rewriting we can directly see that the least squares solution is:

$$\mathbf{s} = [\mathcal{M}^\top\mathcal{M} + \beta\mathcal{L}^\top\mathcal{L}]^{-1}\mathcal{M}^\top\mathbf{x}. \tag{23}$$

Due to the sparseness of the matrices an implementation using a sparse matrix representation is advantageous.

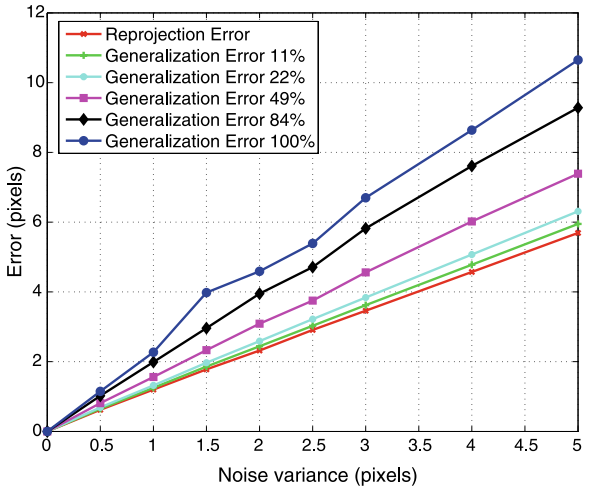


Fig. 2 Reprojection error and generalization error versus the variance of added noise σ for different percentages of hidden points to compute the generalization error

7 Experimental Results

In the experiments reported below we will first test the ability of the basic estimation method (without priors) to behave well on "easy" synthetic sequences with partial, noisy, and outlier corrupted data. Next we report experiments showing the advantages of the temporal smoothness prior and the shape smoothness prior on real images. Finally, an example of track gluing is reported.

7.1 Experiments on Easy Synthetic Data

We simulated $N = 180$ cameras observing a set of $M = 1000$ points generated from $l = 5$ basis shapes, hence with rank $r = 3l = 15$. The configuration weights were chosen in order to give a decaying energy to successive deformation modes, and such that the singular spectrum decayed smoothly to zero at the true rank value. The simulation setup produces a complete measurement matrix from which we extract a sparse, band-diagonal measurement matrix X with about 50% of the data, similar to what a real intensity-based point tracker produces. Gaussian distributed noise with zero mean and a variance of σ^2 was added to the image points. In the first experiment the true rank was assumed known. No prior information was used. Figure 2 shows plots of the reprojection error and the generalization error as functions of σ. The generalization measures are made in 5 bands including the training data and 11%, 22%, 49%, 84%, and all of the data held back for test. Figure 2 shows that the error is approximately proportional and slightly larger to the noise level. As expected the generalization error increases with the generalization distance. The reason that the generalization error is only slightly larger compared to the reprojection error is that the data was designed to be "easy".

Table 2 Average and standard deviation of estimated rank estimate as function of the true rank. In the test 300 "easy" sequences with a varying amount of up to 50% outliers was used

True rank	3	6	9	12	15	18
Average	3.67	5.98	8.47	11.06	13.68	16.32
Std.	0.44	0.41	0.57	0.60	0.62	0.76

In the second experiment the rank value estimation was tested using the same data. Table 2 show the average and the standard deviation of the estimated rank value as function of the true rank value. In the test 50 sequences for each of 6 different amounts of outlier contamination (0%, 10%, 20%, 30%, 40% and 50%) was used. No matter the degree of outlier contamination the results were, as expected, very similar. In Table 2 these numbers are averaged. As seen GRIC slightly over/under-estimates the rank when this is small/large.

The results show that the basic implicit non-rigid structure-from-motion modeling works well on "easy" synthetic data. When the difficulty of the data increases, *e.g.* when the singular value spectrum becomes flatter, the modeling error increases, and the rank estimate gets more uncertain with a tendency to be overestimated. In case of real data, overestimation is less meaningful because the model is empirical, *i.e.* no real data are fully explained by the model. In any case such "overestimation" does not seem serious if priors are used.

7.2 Experiments with Priors Using Real Videos

In the following experiments we measure the improvement in generalization by applying the two priors. The generalization is measured by the average point prediction accuracy as a function of the generalization distance, *i.e.* the column-wise distance in frames to the closest data point used for training. To make the experiment realistic only real sequences are used. Figure 3 shows single frames from the two sequences called *Bears* and *Groundhog day*. The sequence *Bears* shows a limited amount of deformation of a continuous surface. In total 94 points were visible in 94 images. The *Groundhog day* sequence is more difficult showing several independent deformations. Originally the measurement matrix was partial, so a complete sub-matrix of 75 frames and 117 point tracks was extracted. From the two complete matrices diagonal bands with 50% entries were selected for training. A third sequence was constructed from the (complete) sequence *Bears* by splitting each track in three sub-tracks. To make the test more realistic the data related to the last 1–7 frames of each track was randomly

Fig. 3 Images from the *Bears* sequence (*top*) and the *Groundhog day* sequence (*bottom*) with marked points

deleted. This resulted in a new measurement matrix with 282 tracks. The visibility matrix is shown to the left in Figure 6.

On the sequence *Bears* with partial data the rank was estimated to 5. After initial estimation $\mathcal{E}_{\text{RE}} = 0.82$ pixels. Applying the priors increased this to 1.20 pixels. The temporal smoothness measure was reduced by a factor of 108.7. Fig. 4 shows on the top a plot of the average generalization error as function of the generalization distance. Without prior use the generalization becomes bad even for short generalization distances. With prior use the error is significantly reduced. For this, relatively easy, sequence the estimated model seems reliable up to a distance of about 15–20 frames. To illustrate the effect of the prior usage Fig. 5 show a close-up of 4 tracks from the *Bears* sequence. The positions computed by using the two priors (squares) are much closer to the true positions (stars) than the ones obtained by not using the priors (diamonds).

On the sequence constructed from *Bears* by track splitting the rank was, as before, estimated to 5. The reprojection error was increased from 0.53 pixel to 1.74 pixels by application of the priors. The temporal smoothness measure was reduced by a factor of 124.2. Figure 4 shows on the bottom that the average generalization error with prior usage is almost constant about 2–3 pixels independently of the generalization distance. This is much less than without prior

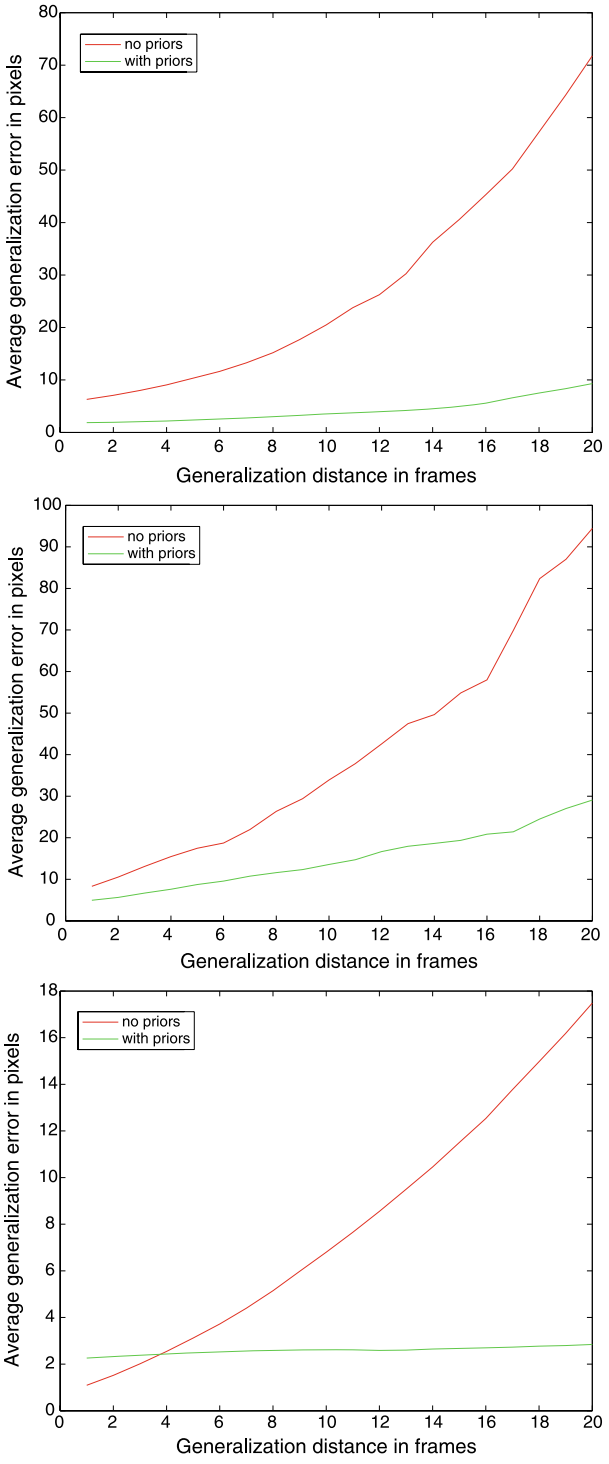


Fig. 4 Average generalization error as function of the generalization distance for the sequence *Bears* (*top*) and *Groundhog day* (*middle*), and the sequence obtained from *Bears* by track splitting (*bottom*)

usage. The reason the generalization errors here is 3–4 times smaller compared to the one showed on the top of Figure 4

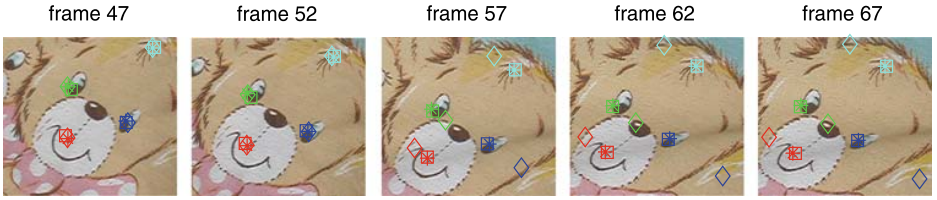


Fig. 5 Close-up sequence of 4 point tracks which visible parts (use for training) all ending close to frame number 47. 'True' positions, given by the tracker, are shown by *stars*. Predicted positions estimated without using the priors are shown by *diamonds*. Predicted positions estimated with use of the priors are shown by *squares*

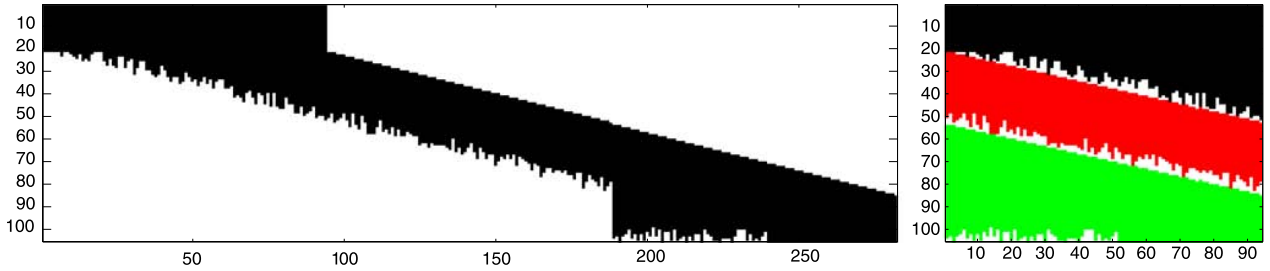


Fig. 6 *Left*: Visibility matrix for measurement matrix constructed by splitting the tracks of *Bears* in 3. *Right*: Glued tracks

is that here most data is maintained. The data is just split into more tracks.

On the sequence *Groundhog day* the rank was estimated to 14, indicating the difficulty of the sequence. The initial reprojection error was increased from 0.96 pixel to 1.62 pixels by application of the priors. The temporal smoothness measure was reduced by a factor of 5660.3. Figure 4 shows in the middle that the improvement in generalization still is significant, but less impressive compared to the other two sequences. A main reason is that here the assumption on scene smoothness made for the surface shape prior does not exactly match the physical scene behavior.

7.3 Experiments with Track Gluing

Often tracks are split due to imperfect tracking and it would be advantageous to glue together the parts. The experiments reported above indicate that without the priors the generalization error would be too large to allow a detection of split tracks. With use of the priors this however might be possible. Below we report a simple experiment using the previously described data obtained from *Bears* by splitting each track in three. Figure 6 show to the left the visibility matrix of the data.

After having estimated the model a gluing algorithm was run. This worked by iteratively gluing a point track with the best fitting track located up to 8 frames before or after the point track. A threshold on the fit was used to stop the gluing process. The resulting matrix of glued tracks showed to the right in Fig. 6 was identical to the original unsplit measurement matrix, *i.e.* the gluing was perfect. Probably this result is due to the simplicity of the deformation. In cases where tracks are split in more shorter sub-tracks a complete gluing cannot be made in a single pass because the reliable generalization distance still is limited. In such cases the estimation and gluing processes may be iterated.

8 Conclusions

We described an implicit non-rigid Structure-from-Motion approach with priors for temporal smoothness and surface shape coherency. We showed that the priors significantly improves the prediction of points projections in frames where data is missing, *i.e.* the generalization ability. Experiments have shown the improvement of the priors sufficient for gluing together point tracks split by imperfect tracking. To our knowledge our approach to Non-rigid Structure-from-Motion is the first that simultaneously can handle a substantial amount of missing data and outliers, can estimate the rank of the measurement matrix, and includes generic prior knowledge on temporal and surface smoothness. We expect the temporal smoothness prior to drive the estimated model closer to an explicit configuration. Further work will show how much this helps in upgrading to metric.

Appendix: Proof: Maximizing the Temporal Smoothness Prior

Below we sketch a proof that by choosing $\mathcal{A}$ as in (19) the temporal smoothness measure (16) is minimized. Thus

we find the implicit coordinates maximizing the temporal smoothness. Let $\mathsf{L} = \mathsf{U}\Sigma\mathsf{V}^\mathsf{T}$ be an SVD of L. Let $\mathcal{A} = \mathsf{QDW}$ be an SVD of $\mathcal{A}$. We parameterize $\mathcal{A}$ as $\mathcal{A} = \mathsf{QD}$ since $\mathcal{E}_\mathrm{J}(\mathsf{J}\mathcal{A}) = \mathcal{E}_\mathrm{J}(\mathsf{JQD})$. Let $\mathsf{Y} = \mathsf{V}^\mathsf{T}\mathsf{Q} \in \mathcal{O}(r)$. We can rewrite $\mathcal{E}_\mathrm{J}(\mathsf{J}\mathcal{A})$ as:

$$\begin{aligned}\|\mathsf{L}\mathcal{A}\|^2 &= \|\mathsf{U}\Sigma\mathsf{V}^\mathsf{T}\mathsf{QD}\|^2 \\ &= \|\Sigma\mathsf{YD}\|^2 = d_1^2\|\Sigma\mathbf{y}_1\|^2 + \cdots + d_r^2\|\Sigma\mathbf{y}_r\|^2, \end{aligned} \tag{24}$$

where $d_r \geq d_{r-1} \geq \cdots \geq d_1 \geq 0$ and with $\mathbf{y}_i$ the columns of Y. We want to find the $\mathbf{y}_i$ and the d_k minimizing the expression under the constraints that $\prod d_k = 1$, and that Y is orthonormal. Due to the ordering of the singular values we can split the minimization problem into r subproblems corresponding to the terms in the sum. From this we get $\mathsf{Y} = \mathrm{I}$, *i.e.* $\mathsf{Q} = \mathsf{V}$. The minimization problem then is reduced to:

$$\min_{\{d_k\}, \prod d_k = 1, d_r \geq \cdots \geq d_1 \geq 0} \sum_{k=1}^{r} (\sigma_k d_k)^2. \tag{25}$$

Introducing Lagrange multipliers λ and μ_j a compound objective function is formulated:

$$\min_{\{d_k\}} \sum_{k=1}^{r} (\sigma_k d_k)^2 + \lambda \left(\prod_{z=1}^{r} d_z - 1 \right) + \sum_{j=1}^{r} \mu_j (d_j - d_{j-1}). \tag{26}$$

It can easily be shown that this function has a minimum given by:

$$2\sigma_k^2 d_k = \lambda \left(\prod_{z=1, z\neq k}^{r} d_z \right) = \frac{\lambda}{d_k}. \tag{27}$$

Letting $\alpha = \sqrt{\lambda/2}$ and checking the unit determinant constraint it is seen that:

$$\alpha = \sqrt[r]{\prod_{k=1}^{r} \sigma_k}. \tag{28}$$

Putting things together we reach expression (19).

To show that the minimum is global the Karush-Kuhn-Tucker conditions can be applied. A sufficient condition for the minimum to be global is that the three terms in (26) are twice differentiable and that the Hessian matrix evaluated in $\mathbb{R}^{r+}$ is positive semi-definite. The Hessian for the first term is diagonal with elements $2\sigma_k^2$. The last term is linear so the Hessian is a positive semi-definite null matrix. The Hessian for the second term $\prod_{z=1}^{r} d_z$ is given by:

$$\mathsf{H} = \begin{pmatrix} 0 & \prod_{i\neq 1,2}^{r} d_i & \cdots & \prod_{i\neq 1,r}^{r} d_i \\ \prod_{i\neq 1,2}^{r} d_i & 0 & \cdots & \prod_{i\neq 2,r}^{r} d_i \\ \vdots & \vdots & \vdots & \vdots \\ \prod_{i\neq 1,r}^{r} d_i & \prod_{i\neq 2,r}^{r} d_i & \cdots & 0 \end{pmatrix}. \tag{29}$$

For $\mathbf{x} \in \mathbb{R}^{r+}$ it is clear that $\mathbf{x}^\top \mathsf{H}\mathbf{x} \geq 0$, so H is positive semi-definite in $\mathbb{R}^{r+}$.

References

1. Aanæs, H., Kahl, F.: Estimation of deformable structure and motion. In: The Vision and Modeling of Dynamic Scenes Workshop (2002)
2. Bartoli, A., Olsen, S.: A batch algorithm for implicit non-rigid shape and motion recovery. In: Workshop on Dynamical Vision at ICCV'05 (2005)
3. Brand, M.: Morphable 3D models from video. In: Conf. on Computer Vision and Pattern Recognition, pp. 456–463 (2001)
4. Brand, M.: A direct method for 3D factorization of nonrigid motion observed in 2D. In: Conf. on Computer Vision and Pattern Recognition, pp. 122–128 (2005)
5. Bregler, C., Hertzmann, A., Biermann, H.: Recovering non-rigid 3D shape from image streams. In: Conf. on Computer Vision and Pattern Recognition, pp. 690–696 (2000)
6. Buchanan, A.M., Fitzgibbon, A.W.: Damped Newton algorithms for matrix factorization with missing data. In: Conf. on Computer Vision and Pattern Recognition, pp. 316–322 (2005)
7. Del Bue, A., Lladó, X., de Agapito, L.: Non-rigid metric shape and motion recovery from uncalibrated images using priors. In: Conf. on Computer Vision and Pattern Recognition, pp. 1191–1198 (2006)
8. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision, 2nd edn. Cambridge University Press, Cambridge (2003)
9. Jacobs, D.W.: Linear fitting with missing data for structure-from-motion. Comput. Vis. Image Underst. **82**(1), 57–81 (2001)
10. Martinec, D., Pajdla, T.: 3D reconstruction by fitting low-rank matrices with missing data. In: Conf. on Computer Vision and Pattern Recognition, pp. 198–205 (2005)
11. Olsen, S., Bartoli, A.: Using priors for improving generalization in non-rigid structure-from-motion. In: Proceedings of the British Machine Vision Conference, pp. 1050–1059 (2007)
12. Roy-Chowdhury, A.K.: A measure of deformability of shapes, with application to human motion analysis. In: Conf. on Computer Vision and Pattern Recognition, pp. 398–404 (2005)
13. Tomasi, C., Kanade, T.: Shape and motion from image streams under orthography: A factorization method. Int. J. Comput. Vis. **9**(2), 137–154 (1992)
14. Torr, P.H.S.: Bayesian model estimation and selection for epipolar geometry and generic manifold fitting. Int. J. Comput. Vis. **50**(1), 27–45 (2002)
15. Torr, P.H.S., Zisserman, A.: MLESAC: A new robust estimator with application to estimating image geometry. Comput. Vis. Image Underst. **78**, 138–156 (2000)
16. Torresani, L., Bregler, C.: Space-time tracking. In: European Conference on Computer Vision, pp. 801–812 (2002)
17. Torresani, L., Hertzmann, A.: Automatic non-rigid 3D modeling from video. In: European Conference on Computer Vision, pp. 299–312 (2004)
18. Torresani, L., Hertzmann, A., Bregler, C.: Non-rigid structure-from-motion: Estimating shape and motion with hierarchical priors. In: IEEE PAMI (2007)
19. Torresani, L., Yang, D.B., Alexander, E.J., Bregler, C.: Tracking and modeling non-rigid objects with rank constraints. In: Conf. on Computer Vision and Pattern Recognition, pp. 493–500 (2001)
20. Triggs, B.: Linear projective reconstruction from matching tensors. Image Vis. Comput. **15**(8), 617–625 (1997)

21. Vidal, R., Abretske, D.: Nonrigid shape and motion from multiple perspective views. In: European Conference on Computer Vision, pp. 205–218 (2006)
22. Xiao, J., Chai, J.-X., Kanade, T.: A closed-form solution to non-rigid shape and motion recovery. In: European Conference on Computer Vision, pp. 573–587 (2004)
23. Xiao, J., Kanade, T.: Non-rigid shape and motion recovery: Degenerate deformations. In: International Conference on Computer Vision and Pattern Recognition, pp. 668–675 (2004)
24. Yan, J., Pollefeys, M.: Articulated motion segmentation using RANSAC with priors. In: Workshop on Dynamical Vision (2005)

S.I. Olsen received a M.Sc. in Computer Science in 1984 and a Ph.D. in Computer Science in 1988 both from University of Copenhagen. Since 1991 he has been associate professor at the Department of Computer Science at the University of Copenhagen. Currently he is with the E-Science center at the University of Copenhagen. His main research interest is computer vision, image processing, and pattern recognition.

A. Bartoli is a permanent CNRS research scientist at the LASMEA laboratory in Clermont-Ferrand, France, since October 2004 and a visiting professor at DIKU in Copenhagen, Denmark for 2006–2009. Before that, he was a post-doctoral researcher at the University of Oxford, UK, in the Visual Geometry Group, under the supervision of Prof. Andrew Zisserman. He did his Ph.D. in the Perception group, in Grenoble at INRIA, France, under the supervision of Prof. Peter Sturm and Prof. Radu Horaud. He received the 2004 INPG Ph.D. Thesis prize and the 2007 best paper award at CORESA. Since September 2006, he is co-leading the ComSee research team. His main research interests are in Structure-from-Motion in rigid and non-rigid environments and machine learning within the field of computer vision.

7.1.4 Paper (CVPR'08) – *Coarse-to-Fine Low-Rank Structure-from-Motion*

Coarse-to-Fine Low-Rank Structure-from-Motion

A. Bartoli[1,2] V. Gay-Bellile[1,3] U. Castellani[4] J. Peyras[1] S. Olsen[2] P. Sayd[3]
[1] LASMEA, Clermont-Ferrand [2] DIKU, Copenhagen [3] CEA, LIST, Saclay [4] VIPS, Verona

Abstract

We address the problem of deformable shape and motion recovery from point correspondences in multiple perspective images. We use the low-rank shape model, i.e. the 3D shape is represented as a linear combination of unknown shape bases.

We propose a new way of looking at the low-rank shape model. Instead of considering it as a whole, we assume a coarse-to-fine ordering of the deformation modes, which can be seen as a model prior. This has several advantages. First, the high level of ambiguity of the original low-rank shape model is drastically reduced since the shape bases can not anymore be arbitrarily re-combined. Second, this allows us to propose a coarse-to-fine reconstruction algorithm which starts by computing the mean shape and iteratively adds deformation modes. It directly gives the sought after metric model, thereby avoiding the difficult upgrading step required by most of the other methods. Third, this makes it possible to automatically select the number of deformation modes as the reconstruction algorithm proceeds. We propose to incorporate two other priors, accounting for temporal and spatial smoothness, which are shown to improve the quality of the recovered model parameters.

The proposed model and reconstruction algorithm are successfully demonstrated on several videos and are shown to outperform the previously proposed algorithms.

1. Introduction

Recovering 3D shape and camera parameters from images is a major research topic in computer vision. The classical Structure-from-Motion paradigm assumes that the observed shape is rigid. It often uses image point tracks obtained by some means. The rigid shape assumption means that the viewing rays corresponding to the same physical point seen in different cameras intersect in space.

For the case of a deforming 3D shape, the assumption that the viewing rays intersect does not hold true. Model-free non-rigid Structure-from-Motion is tackled in *e.g.* [5, 4, 11, 13]. An approach that recently proved successful is the one using the low-rank shape model, which represents the 3D deforming shape by a linear combination of shape bases we call deformation modes or simply *modes*. The modes are point-dependent while the linear combination coefficients, called *configuration weights*, are view-dependent. The representative power of this model lies in its ability to capture, as Principal Component Analysis does, the structure underlying the actual deformations of the 3D shape. The main assumption on the 3D shape is that it consists of a single moving and deforming object, so that the deformation at each point has some sort of consistency with the other points, as is formally defined in [15]. The low-rank terminology stems from the fact that the number of modes is assumed much lower than the number of images and points.

The major difference of the proposed method with the previous ones lies in the coarse-to-fine definition of the low-rank shape model we use. Most of the previous methods treat modes equally, resulting in ambiguities as any mode can be replaced by a linear combination of the other ones. In contrast, we use the rule that a deformation mode encapsulates as much of the data variance left unexplained by the preceding modes as possible. This has important practical impacts, as the level of ambiguity is drastically reduced and makes a coarse-to-fine reconstruction algorithm possible. The idea is that the modes capture decreasingly important details in the deformation. Our model is based on composing this coarse-to-fine low-rank shape model with euclidean transformations accounting for the global displacement of the object of interest. The number of modes is automatically chosen based on Cross-Validation.

To summarize, this paper brings a novel low-rank Structure-from-Motion method which handles missing data, automatically selects the number of deformation modes and makes use of several different priors. We report experimental results on challenging datasets showing that the method gives sensible 3D shapes, allowing us to convincingly augment the video by adding a virtual 3D object on a deforming surface.

2. Previous Work and Contributions

Previous low-rank Structure-from-Motion methods differ by the optimization method and the priors they use,

and if they order the modes or not. Early methods such as [5] use no prior. They are based on computing an 'implicit model' for which the configuration weights and camera parameters are mixed up together through a mixing matrix. The implicit model is upgraded to the 'explicit model' (the model described so far). An efficient implicit model reconstruction method is described in [9]. Recent papers focus on how to compute the implicit to explicit upgrade [4, 13]. While most papers use an affine camera model, some recent papers consider the case of a perspective camera, *e.g.* [6, 12].

Aanaes and Kahl [1] take a different approach: they view the low-rank shape model as a mean shape, that they compute using rigid Structure-from-Motion, and modes that are found through Principal Component Analysis of the directional variance. The overall model parameters are refined together through bundle adjustment. In contrast, we compute the mean shape and iteratively add modes by minimizing the reprojection error. This has the advantage to result in a coarse-to-fine model, expressed in a metric framework, thus *avoiding the difficult implicit-to-explicit upgrading step*. The coarse-to-fine scheme ensures that the leading modes encapsulate coarsest deformations. We show that the deformation mode estimation problem can be splitted into several much smaller problems. The resulting algorithm is efficient and copes with missing data resulting from occlusions.

Finally, there are few papers on the crucial problem of selecting the number of modes. Existing approaches are based either on inspecting the eigenvalues of the data matrix [14] or on model selection criteria such as BIC [1] or GRIC [3]. We provide a solution based on Cross-Validation which, contrarily to previous approaches, does not assume a gaussian *iid* distribution with known variance on the residuals. We show that it gives very sensible results with respect to ground truth.

3. Background

3.1. Notation and Camera Model

Everything is in homogeneous coordinates. A 3D point $\mathbf{Q}$ projects to a 2D point $\hat{\mathbf{q}} \overset{\text{def}}{\sim} \mathsf{P}\mathbf{Q}$ through camera P, where P is a (3×4) perspective projection matrix. The *reprojection error* for this image point is the euclidean distance $d(\mathbf{q}, \hat{\mathbf{q}})$ between the model-predicted point $\hat{\mathbf{q}}$ and the corresponding data point $\mathbf{q}$. The corresponding *algebraic reprojection error* is given by using the following algebraic distance:

$$\mu^2(\mathbf{q}, \hat{\mathbf{q}}) \overset{\text{def}}{=} \|\mathsf{S}(\mathbf{q} \times \hat{\mathbf{q}})\|^2 \qquad \text{with} \qquad \mathsf{S} \overset{\text{def}}{=} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \quad (1)$$

where $\|\cdot\|$ is the two-norm for vectors and Frobenius norm for matrices. The point-to-line orthogonal distance between $\mathbf{q}$ and $\mathbf{l}$ is written $d_\perp(\mathbf{q}, \mathbf{l})$. The following is an algebraic approximation:

$$\mu_\perp^2(\mathbf{q}, \mathbf{l}) \overset{\text{def}}{=} \left(\mathbf{q}^\mathsf{T}\mathbf{l}\right)^2. \quad (2)$$

We use 'normalized' image coordinates which are known to improve the performance of algebraic approximations [7]. The data points lying on the deforming object in the image are written $\mathbf{q}_{i,j}$ where $i = 1, \ldots, n$ is the image index and $j = 1, \ldots, m$ the point index. The binary entries $v_{i,j}$ of the $(n \times m)$ visibility matrix V indicate missing data.

We write $SE(3)$ the group of euclidean transformations in 3-space; $\mathsf{E} \in SE(3)$ is a (4×4) matrix. We define $\mathcal{R}(\mathsf{E}) \overset{\text{def}}{=} \mathsf{R}$ and $\mathcal{T}(\mathsf{E}) \overset{\text{def}}{=} \mathbf{t}$ as the (3×3) rotation matrix and (3×1) translation vector in E respectively.

3.2. The Low-Rank Non-Rigid Shape Model

The deforming 3D points $\mathbf{S}_{i,j}$ are modeled by combining l modes and a mean shape $\mathbf{M}_j^\mathsf{T} = (\bar{\mathbf{M}}_j^\mathsf{T} \;\; 1)$. Mode k is defined point-wise by $b_{k,j}\mathbf{C}_{k,j}$ with $\|\mathbf{C}_{k,j}\| = 1$ with $\mathbf{C}_{k,j}^\mathsf{T} = (\bar{\mathbf{C}}_{k,j}^\mathsf{T} \;\; 0)$ a direction vector and $b_{k,j}$ a deformation magnitude. Camera-wise configuration weights are written $a_{i,k}$. The l-mode shape is:

$$\mathbf{S}_{i,j}^l \overset{\text{def}}{\sim} \mathsf{D}_i \left(\mathbf{M}_j + \sum_{k=1}^{l} a_{i,k} b_{k,j} \mathbf{C}_{k,j} \right), \quad (3)$$

where the $\mathsf{D}_i \in SE(3)$ are aligning transformations, so that the mean shape and its deformations are expressed in an object-centred coordinate frame. Each mode allows a 3D point to move in some direction by a point-dependent and a view-dependent magnitude. The aligning transformations D_i are important since we want the deformation modes to express intrinsic object deformations as opposed to object displacements. The prediction of an image point, *i.e.* the reprojection of a 3D point under this model, writes:

$$\mathbf{s}_{i,j}^l \overset{\text{def}}{\sim} \mathsf{P}_i \mathbf{S}_{i,j}^l \sim \mathsf{P}_i \mathsf{D}_i \mathbf{M}_j + \bar{\mathsf{P}}_i \mathcal{R}(\mathsf{D}_i) \sum_{k=1}^{l} a_{i,k} b_{k,j} \bar{\mathbf{C}}_{k,j}, \quad (4)$$

with $\mathsf{P}_i = \mathsf{K}_i(\mathsf{I} \;\; \mathbf{0})\mathsf{E}_i$. We define the n-vector $\mathbf{a}_l \overset{\text{def}}{=} (a_{1,l} \;\; \cdots \;\; a_{n,l})$, the m-vector $\mathbf{b}_l \overset{\text{def}}{=} (b_{l,1} \;\; \cdots \;\; b_{l,m})$, the $3m$-vector $\bar{\mathbf{C}}_l^\mathsf{T} \overset{\text{def}}{=} (\bar{\mathbf{C}}_{l,1}^\mathsf{T} \;\; \cdots \;\; \bar{\mathbf{C}}_{l,m}^\mathsf{T})$ and $\bar{\mathbf{B}}_l$ similarly.

This model has ambiguities caused by internal 'gauge freedoms'. There is obviously an undetermined euclidean transformation between the mean shape and modes, and the aligning transformations. For globally estimated modes, as in standard approaches, there is an l^2 representational ambiguity since any mode can also be replaced by any linear combination of all modes. In our method, each mode is estimated conditioned on the coarser ones, yielding only a single degree of ambiguity for each mode. Indeed, equation (4) shows that mode l contributes through the exterior product $\mathbf{a}_l\mathbf{b}_l^\mathsf{T}$ which factors can be rescaled since $\forall \nu \in \mathbb{R}^\star$, $\mathbf{a}_l\mathbf{b}_l^\mathsf{T} = (\nu\mathbf{a}_l)\left(\frac{1}{\nu}\mathbf{b}_l^\mathsf{T}\right)$.

3.3. More Priors

The low-rank shape model is very sensitive to the number of modes. Since this is a empirical model, there might not be an ideal such number. Bad results are reported in [11] when the basic low-rank shape model is used to find the 3D shape. Priors are needed in order to better constrain the model. We review some generic priors, where generic means not specific to some object or object-class.

A simple prior is the one of assuming a part of the scene to be rigid [6]. [1] uses as prior the fact that the shape should be close in neighbouring frames. [11] uses a gaussian distribution prior on the configuration weights. This allows to marginalize the configuration weights out of the estimation, which can then be performed very efficiently. They also propose to model temporal camera smoothness through a Linear Dynamics model. The transition matrix is estimated along with the other unknown parameters.

[9] uses a temporal smoothness prior penalizing variations in the implicit camera matrices, embedding both the camera parameters and configuration weights:

$$\sum_{k=1}^{l} \|\Delta \mathbf{a}_k\|^2 = \left\|\Delta \begin{pmatrix} \mathbf{a}_1 & \cdots & \mathbf{a}_l \end{pmatrix}\right\|^2 \tag{5}$$

where Δ is some finite difference operator. They also propose a surface-shape prior. It is based on the fact that points close in the images are close in space, provided they lie on a continuous surface.

We use those two priors. We measure the closeness of points on the mean shape: $\varphi_{j,g} \stackrel{\text{def}}{=} \rho(d^2(\mathbf{M}_j, \mathbf{M}_g))$, with ρ some localized kernel (we use a truncated gaussian) and write the surface-shape penalty as:

$$\sum_{k=1}^{l}\sum_{j=1}^{m}\sum_{g=1}^{m} \varphi_{j,g}^2 \|\bar{\mathbf{B}}_{k,j} - \bar{\mathbf{B}}_{k,g}\|^2 = \sum_{k=1}^{l} \left\|\Omega \bar{\mathbf{B}}_k\right\|^2, \tag{6}$$

where Ω is a highly sparse matrix depending on the $\varphi_{j,g}$ with three times as many rows as non-zero $\varphi_{j,g}$ and $3m$ columns.

We consider another class of priors that has not been used so far in the literature, on the ordering of the deformation modes. We require mode $l+1$ to express as much of the variance remaining unexplained by the l-mode shape as possible. This naturally leads early modes to explain coarse deformations. This kind of priors is difficult to express in the classical framework where all modes are estimated at once. It however fits very well into the framework of iteratively adding modes of variations, as shown below.

4. Coarse-to-Fine Low-Rank Shape

4.1. Overview

The algorithm we propose is based on recovering the mean shape points $\mathbf{M}_j$, giving a coarse approximation to the true shape, in accordance with the mean shape definition in [15]. Modes are added until some criterion is met.

Most of the other methods estimate all the modes and configuration weights at once. In contrast, our solution tries to embed as much of the variance of the data as possible in the current mode to be estimated, which naturally complies with the mode ordering prior described in the previous section. More precisely, the $l+1$ mode is selected so that the shape minimizes the cost. We thus end up with a series of nested minimization problems. This way of solving the problem has several computational advantages, as is shown later in the paper.

Our algorithm is based on the following relationships steming from the shape model (3):

$$\mathbf{S}_{i,j}^0 = \mathsf{D}_i \mathbf{M}_j \tag{7}$$

$$\mathbf{S}_{i,j}^{l+1} = \mathbf{S}_{i,j}^l + a_{i,l+1} b_{l+1,j} \mathsf{D}_i \mathbf{C}_{l+1,j}. \tag{8}$$

We proceed as follow. First, we compute the mean shape points $\mathbf{M}_j$ and the aligning displacements D_i through the 0-mode shape (7). Second, we iteratively triangulate the modes[1], *i.e.* the shapes bases $b_{k,j}\mathbf{C}_{k,j}$ and configuration weights $a_{i,k}$ from (8). A cost function using the reprojection error as data term and the above-mentioned priors is minimized at each step. We stop adding modes when some model complexity selection criterion is met, see §4.4.

4.2. Mean Shape and Aligning Displacements

In order to find the displacements D_i that globally align the deforming object to the world coordinate frame and the mean shape points $\mathbf{M}_j$, we minimize the reprojection error[2] for the 0-mode shape:

$$\min_{\mathbf{M}_1,\ldots,\mathbf{M}_m,\mathsf{D}_1,\ldots,\mathsf{D}_n} \sum_{i=1}^{n}\sum_{j=1}^{m} v_{i,j}\, d^2(\mathbf{q}_{ij}, \mathsf{P}_i\mathsf{D}_i\mathbf{M}_j),$$

which is a calibrated camera instance of the Structure-from-Motion problem, that we solve using standard techniques including bundle adjustment, see *e.g.* [7]. The cameras P_i can either be estimated based on some rigid part in the scene such as the background or be set to some canonical position. We stress that it does not change the result of our algorithm, *i.e.* the estimated deforming surface will be the same whatever the P_i thanks to the D_i.

4.3. Mode Triangulation

The mode triangulation problem is stated as:

$$\min_{\mathbf{a}_{l+1},\bar{\mathbf{B}}_{l+1}} \sum_{i,j} v_{i,j}\, d^2(\mathbf{q}_{i,j}, \mathbf{s}_{i,j}^{l+1}) + \lambda\|\Delta\mathbf{a}_{l+1}\|^2 + \kappa\|\Omega\bar{\mathbf{B}}_{l+1}\|^2. \tag{9}$$

[1] Since the global motion of the object is known at this step, we call 'mode triangulation' the estimation of a mode.

[2] Using a temporal or spatial prior at this stage is not very important since rigid Structure-from-Motion is usually well-posed.

This is a nonlinear least squares optimization problem since *(i)* there is a product between the configuration weights and *(ii)* the modes, and the euclidean distance is used to compare the image points. As in the rigid triangulation case, the euclidean distance can be dealt with an algebraic approximation. The problem however remains nonlinear and difficult to handle in this form since the different views and points are all linked.

First, we drop the priors and compute an initial solution. Second, we refine the complete cost function (9) through nonlinear minimization.

We show that the optimal, *i.e.* reprojection error minimizing, directions in $\bar{\mathbf{C}}_{l+1}$ of the modes can be computed independently from each other and from the other unknowns. We thus split the computation into two main steps. First, we compute the optimal directions in $\bar{\mathbf{C}}_{l+1}$. Second, we compute the optimal configuration weights in a_{l+1} and magnitudes of the modes in b_{l+1}. Each step finds a suboptimal initial solution using linear least squares approximations and refines it by minimizing the reprojection error in a nonlinear manner.

4.3.1 Initializing the Mode Directions in $\bar{\mathbf{C}}_{l+1}$

Splitting the problem. We show how problem (9) can be reformulated on a point-wise basis by estimating independently the direction $\mathbf{C}_{l+1,j}$ of each mode. This is based on casting the reprojection error as a sum of squared point-to-line distances. Substituting equation (3) into (4):

$$\mathbf{s}_{i,j}^{l+1} \sim \underbrace{\mathsf{P}_i \mathsf{D}_i \mathbf{S}_{i,j}^{l}}_{\sim \mathbf{s}_{i,j}^{l}} + a_{i,l+1} b_{l+1,j} \bar{\mathsf{P}}_i \mathcal{R}(\mathsf{D}_i) \bar{\mathbf{C}}_{l+1,j}. \quad (10)$$

This represents an image point parameterized by its position $a_{i,l+1}b_{l+1,j}$ on an image line parameterized by its base point $\mathbf{s}_{i,j}^{l}$ and direction $\bar{\mathsf{P}}_i \mathcal{R}(\mathsf{D}_i)\bar{\mathbf{C}}_{l+1,j}$. By replacing the reprojected points $\mathbf{s}_{i,j}^{l+1}$ from (10) into each reprojection error term in (9), we get:

$$\min_{\mathbf{a}_{l+1}, \mathbf{B}_{l+1}} \sum_{i,j} v_{i,j}\, d^2(\mathbf{q}_{i,j}, \mathbf{s}_{i,j}^{l} + a_{i,l+1} b_{l+1,j} \bar{\mathsf{P}}_i \mathcal{R}(\mathsf{D}_i) \bar{\mathbf{C}}_{l+1,j}).$$

Each term is the squared euclidean distance between an image point $\mathbf{q}_{i,j}$ and the above described point on line. In order to get rid of the offset which depends on the unknown configuration weight $a_{i,l+1}$ and mode magnitude $b_{l+1,j}$, we replace the point-to-point distance d by the point-to-line distance $d_\perp$. This is done by introducing the line coordinates $\mathbf{s}_{i,j}^{l} \times (\bar{\mathsf{P}}_i \mathcal{R}(\mathsf{D}_i)\bar{\mathbf{C}}_{l+1,j})$, giving:

$$\min_{\bar{\mathbf{C}}_{l+1}} \sum_{i=1}^{n} \sum_{j=1}^{m} v_{i,j}\, d_\perp^2(\mathbf{q}_{i,j}, \mathbf{s}_{i,j}^{l} \times (\bar{\mathsf{P}}_i \mathcal{R}(\mathsf{D}_i) \bar{\mathbf{C}}_{l+1,j})).$$

In this reformulated minimization problem, each mode direction $\bar{\mathbf{C}}_{l+1,j}$ in $\bar{\mathbf{C}}_{l+1}$ is independent. It can thus be split as m independent smaller problems:

$$\min_{\bar{\mathbf{C}}_{l+1,j}} \sum_{i=1}^{n} v_{i,j}\, d_\perp^2(\mathbf{q}_{i,j}, \mathbf{s}_{i,j}^{l} \times (\bar{\mathsf{P}}_i \mathcal{R}(\mathsf{D}_i) \bar{\mathbf{C}}_{l+1,j})). \quad (11)$$

Linear estimation. The first step to compute each mode direction $\bar{\mathbf{C}}_{l+1,j}$ is to make a linear least squares approximation to the above stated optimization problem. We approximate the euclidean point-to-line distance by the algebraic one in (2):

$$d_\perp^2(\mathbf{q}_{i,j}, \mathbf{s}_{i,j}^{l} \times (\bar{\mathsf{P}}_i \mathcal{R}(\mathsf{D}_i) \bar{\mathbf{C}}_{l+1,j})) \approx \left(\mathbf{q}_{i,j}^{\mathsf{T}} [\mathbf{s}_{i,j}^{l}]_\times \bar{\mathsf{P}}_i \mathcal{R}(\mathsf{D}_i) \bar{\mathbf{C}}_{l+1,j}\right.$$

The sum over i is minimized to get the initial estimate of $\bar{\mathbf{C}}_{l+1,j}$ with $\|\bar{\mathbf{C}}_{l+1,j}\| = 1$ as required, by the right singular vector corresponding to the smallest singular value of:

$$\mathsf{A} = \begin{pmatrix} v_{1,j}\ \mathbf{q}_{1,j}^{\mathsf{T}} [\mathbf{s}_{1,j}^{l}]_\times \bar{\mathsf{P}}_1 \mathcal{R}(\mathsf{D}_1) \\ \vdots \\ v_{n,j}\ \mathbf{q}_{n,j}^{\mathsf{T}} [\mathbf{s}_{n,j}^{l}]_\times \bar{\mathsf{P}}_n \mathcal{R}(\mathsf{D}_n) \end{pmatrix},$$

where the rows vanishing due to a missing image point (*i.e.* for which $v_{i,j} = 0$) are obviously dropped. The minimum number of image points is $n \geq 2$.

Nonlinear refinement. The second step is to nonlinearly refine the initial estimate of each $\bar{\mathbf{C}}_{l+1,j}$. We minimize the reprojection error using Levenberg-Marquardt. This is very computationally efficient since each of the directions has only 3 parameters and is processed independently. Among the 3 parameters, only 2 are independent, which makes rank-deficient the Jacobian matrix J in the normal equations. This can be dealt with by adding a penalty $(\|\bar{\mathbf{C}}_{l+1,j}\|^2 - 1)^2$ to the error function.

4.3.2 Initializing the Configuration Weights in $\mathbf{a}_{l+1}$ and the Mode Magnitudes in $\mathbf{b}_{l+1}$

Principle. The optimal estimate depends on all the unknown parameters since the image points $\mathbf{s}_{i,j}^{l+1}$ for all views and points depend on $\mathbf{a}_{l+1}\mathbf{b}_{l+1}^{\mathsf{T}}$. We exploit the 1D model ambiguity: we normalize by each of the unknown parameters in $\mathbf{a}_{l+1}$ on turn, making linear the product with the other factor. The results are then combined together.

The constraints. Assume $a_{\zeta,l+1} \neq 0$ for some $\zeta \in 1, \ldots, n$, and define $\mathbf{a}_{l+1}^{\zeta} \stackrel{\text{def}}{=} \frac{\mathbf{a}_{l+1}}{a_{\zeta,l+1}}$ and $\mathbf{b}_{l+1}^{\zeta} \stackrel{\text{def}}{=} a_{\zeta,l+1}\mathbf{b}_{l+1}$. Keeping only the terms related to view ζ in the cost function (9) gives:

$$\min_{\mathbf{b}_{l+1}^{\zeta}} \sum_{j=1}^{m} v_{\zeta,j}\, d^2(\mathbf{q}_{\zeta,j}, \mathbf{s}_{\zeta,j}^{l} + b_{l+1,j}^{\zeta} \bar{\mathsf{P}}_\zeta \mathcal{R}(\mathsf{D}_\zeta) \bar{\mathbf{C}}_{l+1,j}).$$

This minimization problem can be split on a point-wise basis, and is equivalent to solving m 1D problems:

$$\min_{b^{\zeta}_{l+1,j}} v_{\zeta,j}\, d^2(\mathbf{q}_{\zeta,j}, \mathbf{s}^l_{\zeta,j} + b^{\zeta}_{l+1,j}\bar{\mathsf{P}}_{\zeta}\mathcal{R}(\mathsf{D}_{\zeta})\bar{\mathbf{C}}_{l+1,j}).$$

This is a single-view point-on-line triangulation problem, solved by orthogonally projecting $\mathbf{q}_{\zeta,j}$ onto the image line $\mathbf{l}^{l+1}_{\zeta,j} \sim \mathbf{s}^l_{\zeta,j} \times (\bar{\mathsf{P}}_{\zeta}\mathcal{R}(\mathsf{D}_{\zeta})\bar{\mathbf{C}}_{l+1,j})$ to give $b^{\zeta}_{l+1,j}$. The problem can not be solved, however, if $v_{\zeta,j} = 0$, *i.e.* if the point j is not seen in view ζ, and also if the line $\mathbf{l}^{l+1}_{\zeta,j}$ is not well-defined, *i.e.* if $d(\mathbf{s}^l_{\zeta,j}, \bar{\mathsf{P}}_{\zeta}\mathcal{R}(\mathsf{D}_{\zeta})\bar{\mathbf{C}}_{l+1,j}) < \epsilon$, where ϵ is some threshold that we typically choose as few pixels. This problem happens if $\bar{\mathbf{C}}_{l+1,j}$ deforms the point along the viewing ray with respect to camera i.

At this stage, we end up with several, scaled versions $\mathbf{b}^{\zeta}_{l+1}$, $\zeta = 1, \ldots, n$ of $\mathbf{b}_{l+1}$, with missing data, related by $\mathbf{b}^{\zeta}_{l+1} = a_{\zeta,l+1}\mathbf{b}_{l+1}$.

Finding the factors. The $\mathbf{b}^{\zeta}_{l+1}$ vectors must be registered together in order to get the overall sought-after vector $\mathbf{b}_{l+1}$ without holes. This is done by computing the other factor $\mathbf{a}_{l+1}$. The $\mathbf{b}^{\zeta}_{l+1}$ are defined in such a way that $\mathbf{b}^{\zeta}_{l+1}a_{\eta,l+1} - \mathbf{b}^{\eta}_{l+1}a_{\zeta,l+1} = 0$. We solve for $\mathbf{a}_{l+1}$ through:

$$\min_{\mathbf{a}_{l+1}} \sum_{\zeta=1}^{n}\sum_{\eta=1}^{n} \|\mathbf{b}^{\zeta}_{l+1}a_{\eta,l+1} - \mathbf{b}^{\eta}_{l+1}a_{\zeta,l+1}\|^2,$$

which is a linear least squares problem, under the constraint $\|\mathbf{a}_{l+1}\| = 1$. Thanks to $\mathbf{a}_{l+1}$, the $\mathbf{b}^{\zeta}_{l+1}$ are rescaled and averaged to get $\mathbf{b}_{l+1}$.

Another possible way to solve the problem is to consider equation $\mathbf{b}^{\zeta}_{l+1} = a_{\zeta,l+1}\mathbf{b}_{l+1}$. This actually shows that we can formulate the problem as rank-1 matrix factorization with missing data, $\begin{pmatrix}\mathbf{b}^1_{l+1} & \cdots & \mathbf{b}^n_{l+1}\end{pmatrix} \rightarrow \mathbf{b}_{l+1}\mathbf{a}^{\mathsf{T}}_{l+1}$.

4.3.3 Nonlinear Refinement

We have to solve the minimization problem (9). Optimizing over the $\bar{\mathbf{B}}_{l+1,j} = b_{l+1,j}\bar{\mathbf{C}}_{l+1,j}$ directly allows to get rid of the constraints $\|\bar{\mathbf{C}}_{l+1,j}\| = 1$. The issue is that $3m + n$ unknowns must be tuned jointly. Carefully examining the pattern of the Jacobian matrix is thus very important for efficient nonlinear least squares minimization. Indeed, it defines the pattern of the Gauss-Newton approximation to the Hessian matrix, the design matrix in the normal equations to be solved at each iteration of the minimization. The Jacobian has three parts, illustrated for a toy example on figure 1. The first part, related to the data term looks like the one obtained in classical bundle adjustment with well-organized blocks. The second part is related to the temporal prior. Choosing for instance a first order derivative prior gives an $((n-1) \times n)$ Jacobian matrix Δ with ones on the main diagonal and minus ones on the first upper diagonal. The third part depends on the amount of interaction between the points, contained in the $\varphi_{j,g}$ parameters. It typically is very sparse since the localized kernel ρ allows a point to interact with its nearest neighbours only.

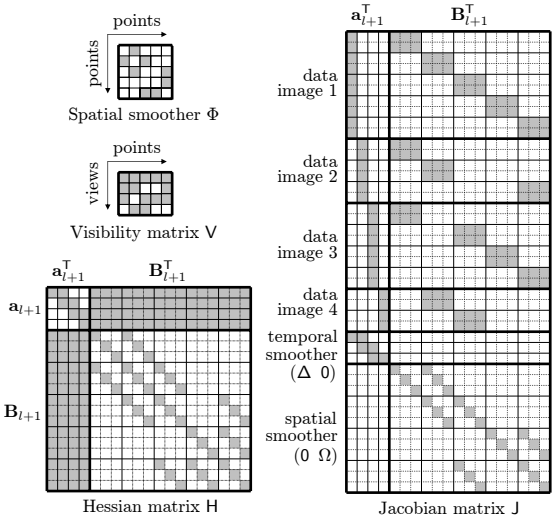


Figure 1. Structure of the Jacobian and Hessian matrices on a toy example with $n = 4$ views and $m = 5$ points.

4.4. A Stopping Criterion

The algorithm we describe in the previous sections is based on iteratively adding modes to the low-rank shape model. A criterion for stopping adding new modes is thus necessary. Each time a mode is added, the number of degrees of freedom of the model grows, making the cost decrease, as is shown in our experimental results. This makes one naturally thinks of using a model selection approach as a stopping criterion. However, the problem at hand does not fulfill the usual model selection assumptions. The first reason is that the number of modes is virtually unlimited: as many modes as desired can be added to the shape model, whereas classical model selection usually operates onto a limited number of models. The second reason is that model selection criteria such as AIC, BIC or GRIC are based on a particular distribution of the residuals, namely a possibly robustified gaussian distribution, see [8, 10]. For the low-rank shape model, the residuals should be interpreted differently. Their dependency on the noise on image point position is very weak. They are mostly due to the deviance of the empirical low-rank shape model from the physics of the actual images. It is difficult to assume any prior distribution for this deviance.

We propose to use Cross-Validation as a criterion for selecting the number of modes. The idea is to partition the

data in a training and a test set, and average the test error over several such partitions. This approach, which has rarely been used for geometric model selection in computer vision, does not require a specific known distribution of the residuals, and directly reflects the ability of the model to extrapolate to new data. More precisely, we use u-fold Cross-Validation, which splits the data into u subsets or 'folds'. Typical values for u range from 3 to 10. We use $u = 4$ in our experiments, and split the data image-point-wise: each fold is a subset of image points, and must allow the algorithm to reconstruct all views and points (for instance, we do not remove all image points in a single view). The test error is obtained by comparing the test dataset with its prediction.

The typical behaviour of the Cross-Validation score is to decrease until the optimal number of modes is reached, and then to increase. It first decreases since the model with not enough modes is too restrictive to explain well the data and thus can not make good predictions. It then increases since with more modes than enough, the model fits unwanted effects in the data, *i.e.* it is too flexible to predict new data. This typical behaviour is however not what we observe when the priors are used. In this case, the Cross-Validation score decreases rapidly until the optimal number of modes is reached, and then remains steady. This is explained by the fact that the priors inhibitate the degrees of freedom of the extra modes, as also reported in [11]. Our stopping criterion has two parts: we stop adding modes when either the Cross-Validation score increases or when its decrease is below some threshold, that we choose as $\varepsilon = 10^{-4}$ in our experiments.

Computing the Cross-Validation score requires to fit the new mode to each of the u training sets. For that purpose, and for computational efficiency, we keep $u + 1$ models: the u models which use the folds as training set, and the one which uses all the data.

OBJECTIVE

Given a set of corresponding image points $\mathbf{q}_{i,j}$ on a deforming object and cameras $\mathsf{P}_i \sim \mathsf{K}_i(\mathrm{I}\ \mathbf{0})\mathsf{E}_i$ obtained by some means, compute globally aligning displacements $\mathsf{D}_i \in SE(3)$ for each frame i and a set of frame-varying, low-rank 3D shapes $\mathbf{S}_{i,j}^l$ in a coarse-to-fine manner, *i.e.* the cost for $\mathbf{S}_{i,j}^{l+1}$ is lower than for $\mathbf{S}_{i,j}^l$. The number of modes l is estimated using Cross-Validation (CV): each computation is carried out over u randomly selected folds to compute the CV score $\mathcal{G}^l$.

ALGORITHM

Mean Shape and Aligning Displacement Computation

1. (§4.2) Run calibrated camera Structure-from-Motion with the image points $\mathbf{q}_{i,j}$ as inputs and intrinsic parameters K_i giving new cameras $\mathsf{K}_i(\mathrm{I}\ \mathbf{0})\mathsf{A}_i$ and mean shape points $\mathbf{M}_j$
2. Set the aligning displacements $\mathsf{D}_i \leftarrow \mathsf{E}_i^{-1}\mathsf{A}_i$
3. (§4.4) Compute the CV score $\mathcal{G}^0$, and set $l \leftarrow 0$
4. Initialize the shape estimate with the mean shape for every frame: $\mathbf{S}_{i,j} \leftarrow \mathbf{M}_j$

Iterative Mode Triangulation

1. (§4.3.1) Initialize the mode directions $\bar{\mathbf{C}}_{l+1,j}$
2. (§4.3.2) Compute the configuration weights $a_{i,l+1}$ and mode magnitudes $b_{i,l+1}$
3. (§4.3.3) Nonlinear refinement: minimize the reprojection error over the modes and configuration weights
4. (§4.4) Compute the CV score $\mathcal{G}^{l+1}$
5. (§4.4) Stop if $\mathcal{G}^{l+1} \geq \mathcal{G}^l$ or $\mathcal{G}^l - \mathcal{G}^{l+1} \leq \varepsilon$
6. Update the 3D shape: $\mathbf{S}_{i,j}^{l+1} \leftarrow \mathbf{S}_{i,j}^l + a_{i,l+1}b_{l+1,j}\mathbf{C}_{l+1,j}$
7. Set $l \leftarrow l + 1$ and loop to step 1

Table 1. Overview of our coarse-to-fine (C2F) low-rank Structure-from-Motion algorithm. The priors are taken into account at step 3 of mode triangulation.

5. Experimental Results

We provide experimental results on simulated and real data. For each dataset, we compare our algorithm with the one by Torresani *et al.* which is shown in [11] to give the best results compared to other methods in the literature. We name it TORRESANI. Our algorithm is summarized in table 1. We use two variants: C2F - NO PRIOR which does not use the two smoothness priors, and C2F - PRIORS which uses them.

We did not encounter any local minimum in the Cross-Validation score in our experiments.

5.1. Simulated Data

We have two data generation models. The first one is the Candide face model [2]. The second one is the shark sequence available from the authors of [11]. We found that the CMU mocap datasets were either close to rigid or not 'homogeneous' enough for the low-rank shape model. For each dataset, we measure the reprojection error, the Cross-Validation score and the 3D error as functions of the number of modes, the amount of missing data and the number of images. The graphs we show are for the Candide face model – similar results as obtained for the shark sequence. The default setup is $n = 10$ images and $m = 113$ points.

The first set of experiments is illustrated on figure 2 (left and middle). It is meant to assess if Cross-Validation effectively gives a sensible way of selecting the number of modes. We observe that our C2F - NO PRIOR is very sensitive to an overestimated number of modes: with more than 2 modes, the 3D error grows rapidly, while both C2F - PRIORS and TORRESANI remains stable. The Cross-Validation

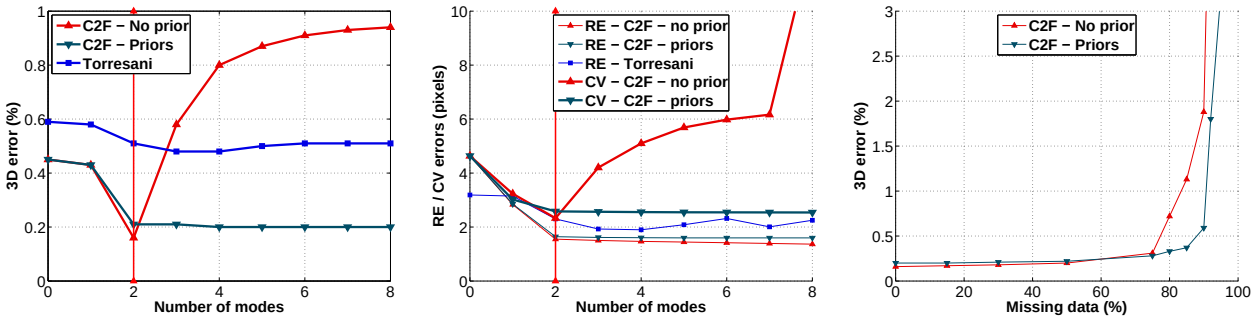


Figure 2. (left) The 3D error as a function of the number of modes. (middle) The reprojection error (RE) and Cross-Validation score (CV) as functions of the number of modes. (right) The 3D error as a function of the percentage of missing data. The vertical bars show minima of the CV score and 3D error curves.

score behaves similarly to the 3D error. In particular, it allows us selecting the optimal number of modes with respect to the 3D error for our C2F - NO PRIOR while for our C2F - PRIORS, the number of modes is slightly overestimated, which does not degrade the quality of the 3D shape, as already observed for TORRESANI in [11]. As expected, the reprojection error decreases as the number of modes increases.

The second set of experiments, shown on figure 2 (right) shows how the algorithms behave against the amount of missing data. Our C2F - PRIORS recovers the 3D shape with up to more than 92% missing data. Thanks to the good behavior of the Cross-Validation score, which allows our C2F - NO PRIOR selecting a sensible number of modes, even with no prior, it handles up to 90% missing data. As for TORRESANI it diverges in most cases.

The third set of experiments computes the success rate of the selected number of modes for C2F - NO PRIOR with the Cross-Validation score. The success rate is 94%, 89% and 88% for respectively no missing data, 25% and 50% missing data. This is very satisfying since in most failures, the number of modes is mis-estimated by only 1.

The fourth set of experiments compares the behaviour of the algorithms with respect to the number of points and views. The graphs are not shown here due to lack of space. As expected, the smaller the number of points or views, the smaller the reprojection error, and the larger the 3D error and Cross-Validation score.

5.2. Real Data

The paper dataset. This video has 203 images of size 720×576. We used a direct, *i.e.* intensity based, approach to recover the parameters of a Free-Form Deformation (FFD) that provided us with 140 point correspondences. Figure 3 shows the results we obtained. Our C2F - NO PRIOR and C2F - PRIORS selected 0 mode and 3 modes and reached 7.10 and 0.84 pixels of reprojection error respectively. C2F - NO PRIOR thus performs very badly for this sequence, giving a very distorted 3D shape. This shows that using the priors can not be avoided, since C2F - PRIORS gives good results, with 1.18 pixels for the Cross-Validation score, showing good predictivity.

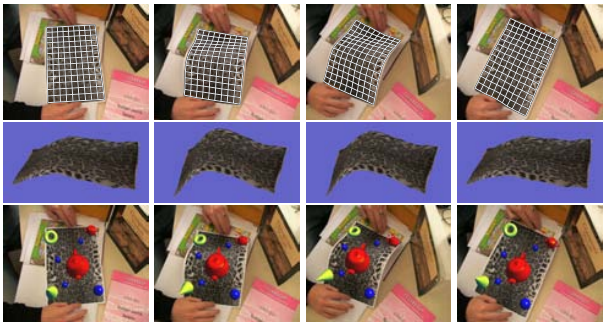


Figure 3. The paper dataset. (first row) Some of the images with the FFD mesh we track. (second row) New view synthesis with the reconstructed surface. (third row) The augmented images.

We then simulated an occlusion by removing 24 points on 120 images, *i.e.* slightly more than 10% of the data. C2F - PRIORS selected 3 modes, and reached 1.44 pixels of reprojection error and 1.82 pixels for the Cross-Validation score, which, although slightly higher than in the full data case, is reasonable.

The face dataset. We extracted a 100 image, 624×352, video of Gabrielle Solis from the series "Desperate Housewives", and ran a 2D Active Appearance Model (AAM) to track her face. We then reconstructed the camera and the 68 vertices of the AAM with our algorithm. Figure 4 shows the result. Both C2F - PRIORS and C2F - NO PRIOR found that 4 modes are required. They respectively obtained 0.91 and 0.82 pixels for the reprojection error, and 1.15 and 1.22 pixels for the Cross-Validation score. These values show that the reconstructed model has a good predictivity. We stress

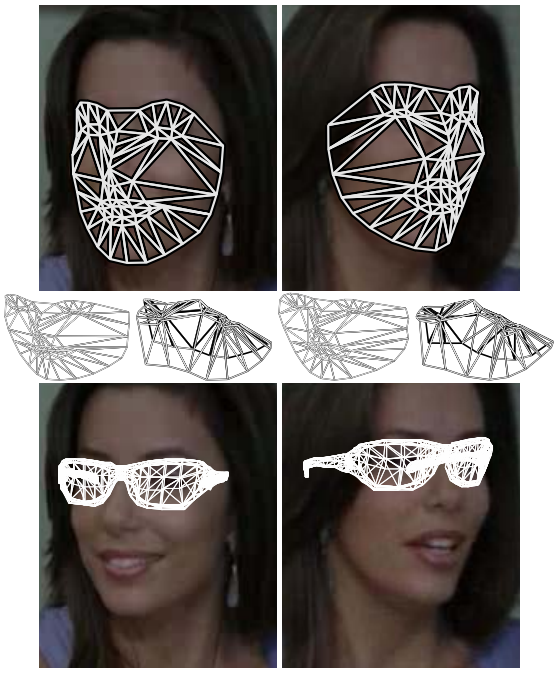


Figure 4. The face dataset. (top) Two out of the 100 images overlaid with the face AAM used for tracking. (middle) The reconstructed AAM vertices. (bottom) The augmented images.

that the a priori knowledge that a face is in the images is used only at the tracking step: our method reconstructs the deforming structure in a generic manner.

6. Conclusion

We proposed a method that allows reconstructing a new coarse-to-fine low-rank shape model of a deforming object from a single video. Our method handles missing data, uses the full perspective camera model and automatically selects the optimal number of deformation modes by Cross-Validating the model. Experimental results on simulated data show that the automatically selected number of modes corresponds to the minimal 3D error. We use two smoothness priors which are shown to improve the quality of the reconstruction. Our method outperforms previous ones in terms of accuracy. The main statement we make is that Cross-Validation is a sensible way of assessing the number of modes in the model in that it looks similar to the 3D error.

An open research topic is the one of automatically selecting the weighting parameters for the priors. Most of the authors reports heuristic means or uses trial and error, as we did in our experiments. A possible solution is to minimize the Cross-Validation score over the weighting parameters. It is not clear if it can be done in a reasonable amount of time, though.

Acknowledgments. We would like to thank Mathieu Perriollat for his help on new view synthesis.

References

[1] H. Aanæs and F. Kahl. Estimation of deformable structure and motion. In *Proceedings of the Vision and Modelling of Dynamic Scenes Workshop*, 2002. 2, 3

[2] J. Ahlberg. CANDIDE-3 – an updated parameterized face. Technical report, Dept. of Electrical Engineering, Linkping University, Sweden, 2001. 6

[3] A. Bartoli and S. Olsen. A batch algorithm for implicit non-rigid shape and motion recovery. In *Proceedings of the Workshop on Dynamical Vision at ICCV*, 2005. 2

[4] M. Brand. A direct method for 3D factorization of non-rigid motion observed in 2D. In *International Conference on Computer Vision and Pattern Recognition*, 2005. 1, 2

[5] C. Bregler, A. Hertzmann, and H. Biermann. Recovering non-rigid 3D shape from image streams. In *International Conference on Computer Vision and Pattern Recognition*, 2000. 1, 2

[6] A. D. Bue, X. Lladó, and L. Agapito. Non-rigid metric shape and motion recovery from uncalibrated images using priors. In *International Conference on Computer Vision and Pattern Recognition*, 2006. 2, 3

[7] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003. Second Edition. 2, 3

[8] K. Kanatani. Geometric information criterion for model selection. *International Journal of Computer Vision*, 26(3):171–189, 1998. 5

[9] S. Olsen and A. Bartoli. Using priors for improving generalization in non-rigid structure-from-motion. In *British Machine Vision Conference*, 2007. 2, 3

[10] P. H. S. Torr. Bayesian model estimation and selection for epipolar geometry and generic manifold fitting. *International Journal of Computer Vision*, 50(1):27–45, 2002. 5

[11] L. Torresani, A. Hertzmann, and C. Bregler. Non-rigid structure-from-motion: Estimating shape and motion with hierarchical priors. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 2007. To appear. 1, 3, 6, 7

[12] R. Vidal and D. Abretske. Nonrigid shape and motion from multiple perspective views. In *European Conference on Computer Vision*, 2006. 2

[13] J. Xiao and T. Kanade. A linear closed-form solution to non-rigid shape and motion recovery. *International Journal of Computer Vision*, 67(2):233–246, March 2006. 1, 2

[14] J. Yan and M. Pollefeys. A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate. In *European Conference on Computer Vision*, 2006. 2

[15] A. J. Yezzi and S. Soatto. Deformotion: Deforming motion, shape average and the joint registration and approximation of structures in images. *International Journal of Computer Vision*, 53(2):153–167, March 2003. 1, 3

7.1.5 Paper (ICIP'06) – *Image Registration by Combining Thin-Plate Splines With a 3D Morphable Model*

IMAGE REGISTRATION BY COMBINING THIN-PLATE SPLINES WITH A 3D MORPHABLE MODEL

Vincent Gay-Bellile [1,2] – *Mathieu Perriollat* [1] – *Adrien Bartoli* [1] – *Patrick Sayd* [2]

[1] LASMEA – CNRS / Université Blaise Pascal – Clermont-Ferrand, France
[2] CEA Saclay, France.
Vincent.GAY-BELLILE@univ-bpclermont.fr

ABSTRACT

Registering images of a deforming surface is a well-studied problem. It is common practice to describe the image deformation fields with Thin-Plate Splines. This has the advantage to involve small numbers of parameters, but has the drawback that the 3D surface is not explicitly reconstructed. We propose an image deformation model combining Thin-Plate Splines with 3D entities – a 3D control mesh and a camera – overcoming the above mentioned drawback. An original solution to the non-rigid image registration problem using this model is proposed and demonstrated on simulated and real data.

1. INTRODUCTION

Registering images of deformable surfaces is important for tasks such as video augmentation, dense Stucture-From-Motion, or deformation capture. This is a difficult problem since the appearance of imaged surfaces varies due to several phenomena such as camera pose, surface deformation, lighting and motion blur. Recovering a generic 3D surface, its deformations and the imaging sensor parameters from monocular video sequences is intrinsically ill-posed. For this reason, most work avoid a full 3D model by directly using image-based deformation models [1, 2, 3]. The obvious drawback of these approaches is that they do not reconstruct the 3D surface.

We propose a novel approach which jointly register the images and computes the 3D surface. It has two main originalities. First, we propose a mixed 3D and image-based generative model combining Thin-Plate Splines (TPS) with a 3D mesh and a camera. This model is dubbed 3D+TPS. It induces a piecewise smooth image deformation field while allowing one to reconstruct a 3D surface corresponding to each image of the sequence. In order to deal with the ill-posedness of the 3D surface and camera pose recovery, admissible surface deformations are learnt as a 3D Morphable Model [2, 4]. Second, we extend a tracking method that was successfully applied to twodimensional cases [1]. It consists in learning an interaction matrix, modeling as a Jacobian matrix does, the relationship between the image intensity variations and those of the model parameters.

Our 3D+TPS model is described in §2 and image registration in §3. Experimental results are reported in §4 and our conclusions are given in §5.

Notation. Vectors are typeset using bold fonts, *e.g.* $\mathbf{q}$, matrices using sans-serif fonts, *e.g.* E, and scalars in italics, *e.g.* α. Matrix and vector transposition in denoted as in A^T.

Previous Work. The registration of images of deformable objects using a single camera has received a growing attention over the past decade. Many approaches have been proposed, based on features or direct image intensity comparison.

Feature-based approaches locate image features on the model, then solve for the registration. For example, a highly efficient surface detection approach is proposed in [5]. The authors use a 2D regularized surface mesh in conjunction with a highly robust estimator to match feature points.

Direct approaches minimize an error expressed on image intensities. Active Appearance Models [1] are 2D learnt generative models that can be fitted to images to track deforming objects. They have been recently extended to 3D [6]. In [3], Radial Basis Mappings represent the transformation.

2. A GENERATIVE IMAGE MODEL

We present the image-based and 3D approaches to modeling image deformations, and then show how to combine them in a single model, drawing on the strengths of both approaches.

2.1. The Image-Based Part: Thin-Plate Splines

A TPS represents a smooth image deformation field. It maps a point $\mathbf{x}$ from the reference image $\mathcal{I}_0$ to the corresponding point $\mathbf{x}' = \tau(\mathbf{x}; \mathbf{q}_t) = \tau_t(\mathbf{x})$ in the target image $\mathcal{I}_t$, see *e.g.* [3] with:

$$\tau(\mathbf{x}; \mathbf{q}_t) = \mathsf{A}\mathbf{x} + \mathbf{y} + \sum_{j=1}^{m} \mathbf{w}_j \phi(\|\mathbf{x} - \mathbf{q}_{tj}\|),$$

where $(\mathsf{A}, \mathbf{y})$ represents a 2D affine transformation and the $\mathbf{w}_j$ and the $\mathbf{q}_{tj}$ are respectively the coefficients and the centers of

the transformation. The kernel function is $\phi(\mu) = \mu^2 \log(\mu)$. TPS are traditionally estimated from point correspondences, see *e.g.* [7].

2.2. The 3D Part: Control Mesh and Camera

A piecewise planar 3D control mesh approximating the surface is used along with a camera model to explain the deformations in the images. So as to deal with the ill-posedness inherent to deforming surface recovery, a 3D Morphable Model is used for the control mesh. The typical deformations are learnt prior to image registration, using PCA (Principal Component Analysis) on a collection of admissible meshes. More details are given in §4.1. The 3D mesh $\mathbf{Q}_t$ is thus expressed in terms of a mean mesh $\bar{\mathbf{E}}$ and l eigenmeshes $\mathbf{E}_k$, and parameterized by view-dependent *configuration weights* $\boldsymbol{\alpha}_t$, so that a 3D vertex is given by:

$$\mathbf{Q}_{tj} = \bar{\mathbf{E}}_j + \sum_{k=1}^{l} \alpha_{tk} \mathbf{E}_{kj}.$$

The 3D mesh can be rotated and translated to account for camera pose: $\mathbf{Q}'_{tj} = \mathsf{R}(\mathbf{a}_t)\mathbf{Q}_{tj} + \mathbf{y}_t$, where $\mathbf{a}_t$ is a 3-vector containing the 3 rotation angles. A projective camera with fixed intrinsic parameters is used. Defining the vector of parameters $\mathbf{S}_t^\mathsf{T} = (\mathbf{a}_t^\mathsf{T}, \mathbf{y}_t^\mathsf{T}, \boldsymbol{\alpha}_t^\mathsf{T})$, the projection is written $\mathbf{q}_t = \Pi(\mathbf{S}_t)$, where $\mathbf{q}_t$ contains the vertices of the imaged mesh.

2.3. The 3D+TPS Model

The main idea for building the 3D+TPS model is that a TPS τ_t can be controlled by using as centers the projected vertices of the 3D control mesh in the reference and target image t. The registration error induced by the TPS will in turn constrain the 3D entities parameters. This tight coupling allows us to infer 3D information from images only.

The transfer function τ_t induced by the 3D+TPS model is used in two ways. First, in the interaction matrices learning stage, for generating images from locally perturbed model parameters. Second, in the registration stage, for warping the target image $\mathcal{I}_t$ onto the reference image. In the first case, τ_t^{-1} is required for warping the reference image onto the perturbed image, while in the second case, τ_t is required. We propose an efficient approximation of τ^{-1} for image warping in §3.2.

To sum up, our 3D+TPS model has the advantage of explicitly involving a simple 3D surface mesh and camera pose and produces a smooth deformation field.

3. REGISTERING IMAGES

3.1. The Error Function

A great variety of feature and intensity based error function are proposed in the litterature. We adopt the direct approach and minimize the sum of squares of intensity differences over pixels $\mathcal{K}$ which are Canny edge points in the reference image:

$$\mathcal{C}(\mathbf{S}_t) = \sum_{\mathbf{x} \in \mathcal{K}} \left(\mathcal{I}_0[\mathbf{x}] - \mathcal{I}_t\left[\tau(\mathbf{x}; \Pi(\mathbf{S}_t))\right] \right)^2. \quad (1)$$

We work with edge points only because they carry the most important texture information. Minimizing $\mathcal{C}$ is a nonlinear least squares problem. One of the most convincing approaches in the litterature is [8]. The error criterion (1) is linearized around $\mathbf{S}_0$ yielding:

$$\delta\mathbf{S}_0 \quad = \mathcal{F} \cdot \mathrm{vect}(\delta\mathcal{I}), \quad (2)$$

where vect is the matrix vectorization operator. A closed-formed expression is derived for matrix $\mathcal{F}$ using the inverse Jacobian image. Cootes *et al.* [1] propose to estimate $\mathcal{F}$ from training images obtained by perturbing the generative model parameters around $\mathbf{S}_0$. The learnt matrix $\mathcal{F}$ is called the *interaction matrix*.

3.2. Learning the Interaction Matrix

Perturbations are drawn randomly and selected if the maximum displacement of image vertices is below some threshold γ that we typically choose as a few pixels. For each selected perturbation, a synthetic image is generated in order to compute the change in appearance, see figure 1. Given the generative model parameters $\mathbf{S}^i$ for the perturbed image, we form the transfer function τ^{-1} and warp the texture image:

$$\tilde{\mathcal{I}}^i[\mathbf{x}] = \mathcal{I}_0[\tau^{-1}(\mathbf{x}; \Pi(\mathbf{S}^i))].$$

This function is implemented using a TPS interpolating the vertices. To compute the TPS coefficients, we generate a regular grid where vertices play the role of centers for the TPS. Thanks to the vertex correspondences between the reference and the perturbed images, we lineary compute the TPS parameters and so the transfer function τ^{-1}.

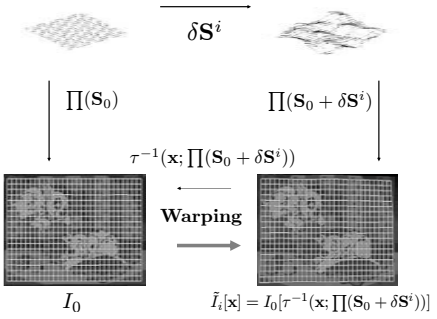


Fig. 1. Training images, needed to learn the interaction matrices, are generated by perturbing the model parameters.

3.3. Registration of an Image Sequence

We proceed as follows. We initialize $\mathbf{S}_t$ to any initial guess, for example $\mathbf{S}_t \leftarrow \mathbf{S}_{t-1}$. We warp the current image $\mathcal{I}_t$ to $\tilde{\mathcal{I}}_t$ using the mapping induced by $\mathbf{S}_t$, and compute the difference image $\delta\mathcal{I}_t = \mathcal{I}_0 - \tilde{\mathcal{I}}_t$. The local update $\delta\mathbf{S}_0$ is then computed from (2) as $\delta\mathbf{S}_0 = \mathcal{F} \cdot \text{vect}(\delta\mathcal{I}_t)$. It must be composed with the current $\mathbf{S}_t$ in order to update it, as illustrated on figure 2. This is a forward compositional strategy [9]. How to compose the local mapping correction $\delta\mathbf{S}_0$ with $\mathbf{S}_t$ is not straightforward. We solve this problem by mapping the control mesh vertices from the reference to the target view giving $\mathbf{q}_t = \tau(\Pi(\mathbf{S}_0 + \delta\mathbf{S}_0); \mathbf{S}_t)$, and minimizing the discrepancy between these vertices and the vertices predicted by the model, *i.e.* the reprojection error, over $\mathbf{S}_t$:

$$\min_{\mathbf{S}_t} \|\mathbf{q}_t - \Pi(\mathbf{S}_t)\|^2. \tag{3}$$

The minimization is solved using the nonlinear least squares algorithm Levenberg-Marquardt.

As underlined above, the linear relationship (2) represents a local approximation of the cost function around $\mathbf{S}_0$. Obviously, the validity of the approximation is conditioned upon the magnitude γ (expressed in pixels), of the perturbation used for generating the training images. In order to increase the speed of convergence and widen the size of the bassin of convergence, we learn not only one, but rather a serie $\mathcal{F}_1, \ldots, \mathcal{F}_\kappa$ of interaction matrices, with a gradually lower perturbation magnitude. This forms a coarse to fine set of linear approximations to the error function, that we apply in turn.

This approach is different from the one in [6], which penalizes a 2D Active Appearance Model, by jointly computing a 3D Morphable Model. In the approach we propose, the 3D model and the TPS are represented with the same set of parameters. One of the differences with [1] is that they assumed that the domain where the linear relationship (2) is valid covers the whole set of registrations, thus avoiding the need of the difficult composition step. This however not appears to be a valid choice in practice.

4. EXPERIMENTAL RESULTS

4.1. The Control Mesh

Depending on the kind of surface that one may want to register, different surface generation schemes are used. For instance, the 3D Morphable Model proposed in [4] can be used for faces. We are interested in registering images of surfaces such as a rug or a sheet of paper, and follow [2] to generate a set of training meshes by deforming a regular, flat mesh by randomly changing the angles between the different facets.

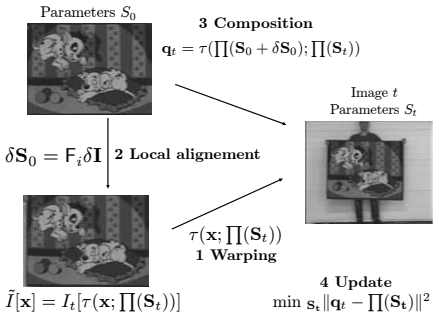


Fig. 2. Our image registration algorithm follows the forward compositionnal strategy, see text for details.

4.2. Simulated Data

In order to assess the behaviour of our algorithm in different conditions, we synthesized images under controlled conditions. Given a reference image, we applied a random perturbation to our model such that the mean rigid displacement of the pixels, caused by the relative displacement between the camera and the control mesh, is δ_R, and the mean non-rigid displacement of the pixels, caused by the deformation of the control mesh, is δ_{NR}. We added gaussian noise, with variance σ % of the maximum greylevel value, to the warped image. We varied each of these parameters independently, using the following default values: $\delta_R = 5$ pixels, $\delta_{NR} = 3$ pixels, $\sigma = 1\%$ while measuring the residual error defined as the mean of Euclidean distance between the vertices of the mesh which generated the warped image, and those of the estimated mesh. Figure 3 shows the results we obtained. We observed on figure 3 (a) that when the magnitude of the perturbation is greater than 20 pixels, the registration efficiency quickly decreases. Those perturbation magnitudes have actually not been learnt, causing the linear approximation being less accurate. We expect the average displacement between consecutive images to be far less than 20 pixels in real cases. Figure 3 (b) shows that the alignment error in pixels is approximatively linear in the variance of the noise on image intensities. In practice, one can expect the noise magnitude to be in the order of 2% of the maximum grey value, making our algorithm well-adapted to many real image sequences.

4.3. Real Data

We tested our algorithm on several image sequences. One of them, consisting of 40 images see figure 5, is used to demonstrate our approach. To initialize the tracker, we made the assumption that the 3D mesh associated to the first image was flat. Consequently, the initialisation problem is equivalent to estimating the relative pose of a plane. Even if some images

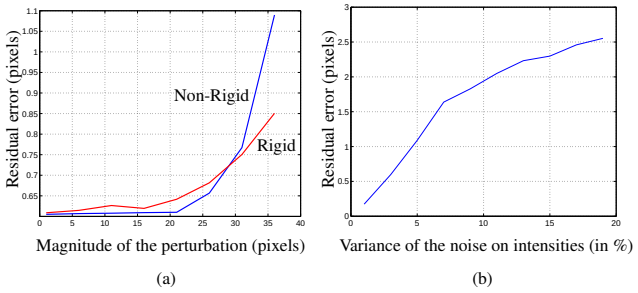


Fig. 3. Registration of simulated data. The results are median over 100 trials. (a) shows the residual error when the magnitude of the perturbation δ_N or δ_{NR} is varied. (b) shows the residual error for varying noise on the image intensities.

of the sequence are blurred, our algorithm achieved successful alignment. In some cases, however, the model drifted away from its ideal position due to lack of constraints in the texture, making some contour points sliding along their edge. Figure 4 shows the composition error and the registration error for each frame. We observed that the composition error, the one minimized in equation (3), is kept around a pixel, meaning that the composition step is successful. The registration error, proportional to equation (1), and expressed in image intensity unit, is kept around typical values, indicating that our model reliably fits the images. The algorithm has been implemented in Matlab, the registation is done at about 7s per image on a pentium IV 3GHz.

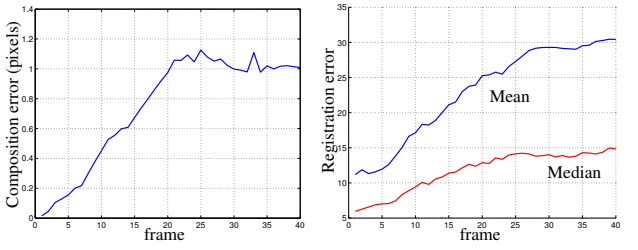


Fig. 4. The composition (a) and the registration (b) errors for the sequence shown in figure (5)

5. CONCLUSIONS

We proposed a novel generic approach to image registration based on a mixed 3D and image-based model. Combining TPS with a 3D mesh and a camera yields smooth image deformation fields while allowing one to recover a 3D surface for each image of a sequence. We plan to extend the method to deal with occlusions, by exploiting the reconstructed 3D surface to predict self-occlusions.

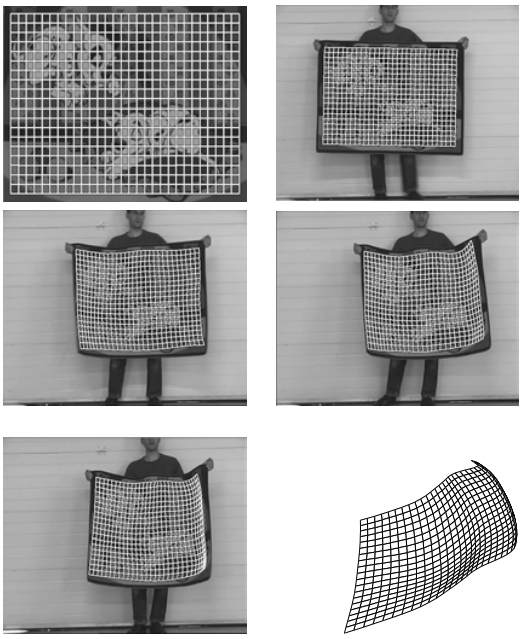


Fig. 5. (Top left) the reference image and his associated mesh. (Next) registration of an image sequence, the projected 3D mesh is shown in white. (Bottom right) the recovered 3D mesh for the last image.

6. REFERENCES

[1] T. F. Cootes, G. J. Edwards, and C. J. Taylor, "Active appearance models," PAMI, 23(6):681-685, 2001.

[2] M. Salzmann, S. Ilic, and P. Fua, "Physically valid shape parameterization for monocular 3-D deformable surface tracking," in BMVC, 2005.

[3] A. Bartoli and A. Zisserman, "Direct estimation of non-rigid registrations," in BMVC, 2004.

[4] V. Blanz and T. Vetter, "Face recognition based on fitting a 3D morphable model," PAMI, 25(9), September 2003.

[5] J. Pilet, V. Lepetit, and P. Fua, "Real-time non-rigid surface detection," in CVPR, 2005.

[6] J. Xiao, S. Baker, I. Matthews, and T. Kanade, "Real-time combined 2D+3D active appearance models," in CVPR, 2004.

[7] F. L. Bookstein, "Principal warps: Thin-plate splines and the decomposition of deformations," PAMI, 11(6):567-585, June 1989.

[8] G. D. Hager and P. N. Belhumeur, "Efficient region tracking with parametric models of geometry and illumination," PAMI, 20(10):1025-1039, 1998.

[9] S. Baker and I. Matthews, "Lucas-Kanade 20 years on: A unifying framework," IJCV, 56(3):221-255, February 2004.

7.2 Multiple Synchronized Cameras and Range Sensors

I36 **A Quasi-Minimal Model for Paper-Like Surfaces** §7.2.1
M. Perriollat and A. Bartoli
BenCOS'07 - ISPRS *Int'l Workshop "Towards Benmarking Automated Calibration, Orientation, and Surface Reconstruction from Images"* at CVPR'07, Minneapolis, USA, June 2007
Previous versions: [I27,N07]
Version in French: [N11]

I25 **Towards 3D Motion Estimation from Deformable Surfaces** §7.2.2
A. Bartoli
ICRA'06 - IEEE *Int'l Conf. on Robotics and Automation*, Orlando, Florida, USA, May 2006
Previous version: [I21]

I48 **Automatic Quasi-Isometric Surface Recovery and Registration from 4D Range Data** §7.2.3
T. Collins, A. Bartoli and R. Fisher
BMVA *Symposium on 3D Video - Analysis, Display and Applications*, London, UK, Feb. 2008

I38 **Joint Reconstruction and Registration of a Deformable Planar Surface Observed by a 3D Sensor** §7.2.4
U. Castellani, V. Gay-Bellile and A. Bartoli
3DIM'07 - *Int'l Conf. on 3D Digital Imaging and Modeling*, Montréal, Québec, Canada, August 2007

7.2.1 Paper (BenCOS'07) – *A Quasi-Minimal Model for Paper-Like Surfaces*

A Quasi-Minimal Model for Paper-Like Surfaces

Mathieu Perriollat
LASMEA – CNRS / UBP
Clermont-Ferrand, France
Mathieu.Perriollat@gmail.com

Adrien Bartoli
LASMEA – CNRS / UBP
Clermont-Ferrand, France
Adrien.Bartoli@gmail.com

Abstract

Smoothly bent paper-like surfaces are developable. They are however difficult to minimally parameterize since the number of meaningful parameters is intrinsically dependent on the actual deformation. Previous generative models are either incomplete, i.e. limited to subsets of developable surfaces, or depend on huge parameter sets.

We propose a generative model governed by a quasi-minimal set of intuitive parameters, namely rules and angles. More precisely, a flat mesh is bent along guiding rules, while a number of extra rules controls the level of smoothness. The generated surface is guaranteed to be developable. A fully automatic multi-camera threedimensional reconstruction algorithm, including model-based bundle-adjustment, demonstrates our model on real images.

1. Introduction

The behaviour of the real world depends on numerous physical phenomena. This makes general-purpose computer vision a tricky task and motivates the need for prior models of the observed structures, *e.g.* [1, 4, 8, 10]. For instance, a 3D morphable face model makes it possible to recover camera pose from a single face image [1].

This paper focuses on paper-like surfaces. More precisely, we consider paper as an unstretchable surface with everywhere vanishing Gaussian curvature. This holds if smooth deformations only occur. This is mathematically modeled by developable surfaces, a subset of ruled surfaces. Broadly speaking, there are two modeling approaches. The first one is to describe a continuous surface by partial differential equations, parametric or implicit functions. The second one describes a mesh representing the surface with as few parameters as possible. The number of which must thus adapt to the actual surface. We follow the second approach.

One of the properties of paper-like surfaces is inextensibility. This is a nonlinear constraint which is not obvious to apply to meshes, as figure 1 illustrates. For instance, Salzmann *et al.* [10] use constant length edges to generate training meshes from which a generating basis is learnt using Principal Component Analysis. The nonlinear constraints are re-injected as a penalty in the eventual fitting cost function. The main drawback of this approach is that the model does not guarantee that the generated surface is developable.

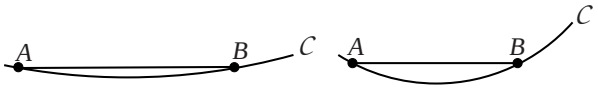


Figure 1. Inextensibility and approximation: a one dimensional example. Curve $\mathcal{C}$ represents an inextensible object, A and B are two points lying on it. The linear approximation of arc (AB) is the straight segment AB. When $\mathcal{C}$ bows, although the arc length (AB) remains constant, the length of segment AB changes. A constant length edge model is thus not a valid parameterization for inextensible surfaces.

We propose a model generating a 3D mesh satisfying the above mentioned properties, namely inextensibility and vanishing Gaussian curvature at any point on the mesh. The model is based on bending a flat surface around rules together with an interpolation process leading to a smooth surface mesh. The number of parameters lies very close to the minimal one because only the global shape is parameterized. A continuous smooth surface is then interpolated. This model is suitable for image fitting applications. We describe an algorithm to recover the deformations and rigid pose of a paper-like object from multiple views. It does not guarantee to find this minimal set, but it estimates a set of few physical parameters explaining the images.

Previous work. Developable surfaces are usually chosen as a basic modeling tool. Most of the work uses a continuous representation of the surface [3, 4, 7, 9]. They are thus not well adapted for fast image fitting, except [4] which initializes the model parameters with a discrete system of rules. [11] constructs developable surfaces by partitioning a surface and curving each piece along a generalized cone defined by its apex and a cross-section spline. This param-

eterization is limited to piecewise generalized cones.

[6] simulates bending and creasing of virtual paper by applying external forces on the surface. This model has a lot of parameters since external forces are defined for each vertex of the mesh. A method for undistorting paper is proposed in [8]. The generated surface is not developable due to a relaxation process that does not preserve inextensibility.

Roadmap. We present our model in §2 and its reconstruction from multiple images in §3. Experimental results on image sequences are reported in §4. Finally, §5 gives our conclusions and discusses future work.

2. A Quasi-Minimal Model

2.1. Principle

Developable surfaces. Since developable surfaces form a subset of ruled surfaces, they can be defined as constrained ruled surfaces. Their continuous mathematical formulation is given in *e.g.* [11] by:

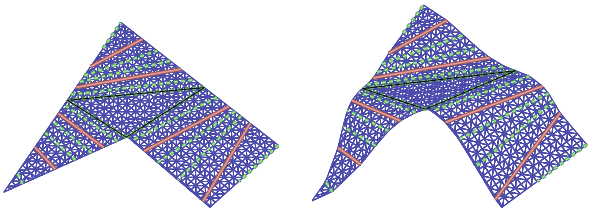
$$\begin{cases} X(t,v) = \alpha(t) + v\beta(t)\ , \quad t \in I \quad v \in \mathbb{R} \quad \beta(t) \neq 0 \\ \det(\alpha'(t), \beta(t), \beta'(t)) = 0. \end{cases} \tag{1}$$

The first equation defines a ruled surface using a differentiable space curve $\alpha(t)$, namely the directrix and a vector field $\beta(t)$. The ruled surface is actually generated by the line pencil $(\alpha(t), \beta(t))$. The second equation enforces vanishing Gaussian curvature, making the ruled surface a developable one.

Most of the previous work on developable surfaces exhibits functions that satisfy this system. For example [3] uses a tensor product B-Spline representation, [7] uses 'cone spline surfaces' and [9] combines dual representation with NURBS surfaces.

Rule-based generation. We propose an intuitive method to build developable surfaces inspired by the observation of real paper sheets. The main idea is to use a discrete set of rules instead of a continuous formulation. This leads to a piecewise planar surface. The constraint on curvature in (1) turns into a formulation in terms of bending angles. The rules are chosen such that they do not intersect each other, which corresponds to the modeling of smooth deformations.

Generating a surface mesh using our model has three main steps, provided the planar boundary shape. First we extract from the parameter set the position of the guiding rules on the flat shape and their bending angle. Second, we add extra rules by interpolating the positions and the angles of the guiding rules. The number of extra rules controls the smoothness of the generated surface. Third, the flat mesh is bent along the rules. Figure 2 illustrates this generating process. It is guaranteed to be admissible in the sense that the surface underlying the generated mesh is developable. Figure 3 shows the generated surface when the number of rules increases.



Flat mesh Bent mesh

Figure 2. Surface mesh generation. (left) Flat mesh with guiding rules (thick and pink) and extra rules (thin and green). (right) Mesh folded along the guiding and extra rules.

Balancing model complexity and surface smoothness. It is obvious that the density of rules is related to the smoothness of the surface: the higher the number of rules, the smoother the surface. It is also linked to the model complexity: the higher the number of rules, the more complex the model. These two observations lead us to consider a huge number of rules to generate a smooth and accurate surface. To avoid an overly large number of parameters, we propose to control a subset of the rules and to interpolate the other ones. They are respectively called guiding and extra rules. This has the advantage to generate a smooth surface with a small set of parameters. The aspect of the final surface depends on both the guiding rules and the interpolation process. Figure 3 illustrates the effect of the proportion between the guiding and extra rules. The surface generated by 6 guiding rules and 12 extra rules is an interesting trade-off: there are enough parameters to capture all deformations since the smoothness given by the extra rules significantly decreases the error, and adding guiding rules does not really improve the accuracy.

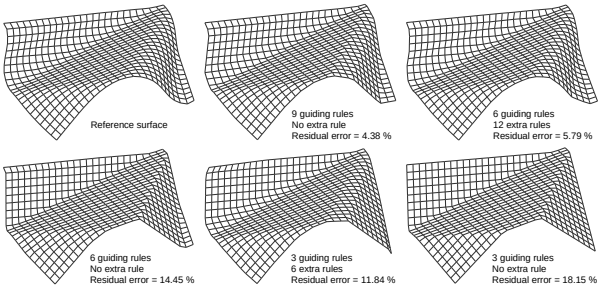


Figure 3. Surface generation behaviour. The reference mesh is estimated by our model with a varying number of parameters. The residual error represents the mean distance to the reference mesh, it is given as a percentage of the meshgrid step.

Internal consistency constraints. A rule is valid if it does not intersect other rules on the surface and, in the case of a non convex boundary, if the segment joining the two intersections is entirely on the mesh, see figure 4 for an example.

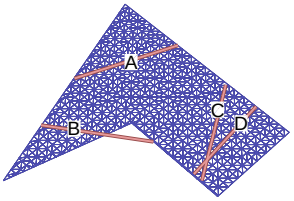


Figure 4. Rule validity examples. Rule A is valid. Rule B is not valid since it gets outside the mesh. Rules C and D are not valid because they intersect each other on the mesh.

2.2. Parameterization

Our model has a parameter set into which we distinguish two parts. The first part describes the shape of the flat mesh. The second part controls the deformations. The shape is defined by a planar curve, often a planar polygon, and gives the boundary of the object.

The deformations are parameterized by the guiding rules and their bending angles. Since each rule intersects the boundary curve at exactly two points, a minimal parameterization of the rules is the arc length of these two points along the shape curve. To build a realistic surface, the rules must not intersect each other on the surface. This is enforced by constraining the arc lengths of the rules. More details are given in §2.3.

The deformations are eventually defined by coupling each rule with a bending angle, choosing the number of extra rules and the interpolation functions.

Table 1 summarizes the model parameters. The model has $2+S+3n$ parameters, with S the number of parameters describing the mesh boundary (for instance, width and height in the case of a rectangular shape) and n the number of guiding rules.

Parameters	Description	Size
n	number of guiding rules	1
n_e	number of extra rules	1
$\mathcal{S}$	mesh boundary parameters	S
s_A	arc length of the guiding rules	$2n$
θ	bending angles of the guiding rules	n

Table 1. Summary of the model parameters. (top) Discrete parameters (kept fixed during nonlinear refinement). (bottom) Continuous parameters.

2.3. Surface Generation

We bring together rules that belong to the same 'bending region' on the paper. We define a region as a set of consecutive rules. Two rules are consecutive if both of their endpoints are. Figure 5 (top left) shows the labeled guiding rules on the flat mesh.

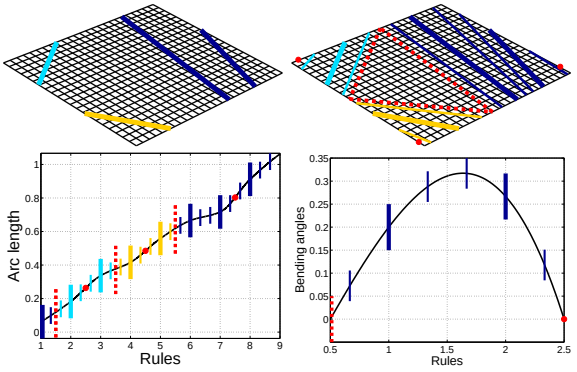


Figure 5. Interpolation process. (top left) Flat mesh with the labeled guiding rules. Three regions are defined. (bottom left) Rule interpolation process. The black curve is an increasing interpolation. (top right) Flat shape with rules. (bottom right) Bending angles interpolation process. The black curve is an interpolation function. The thick lines are the guiding rules. The thin lines are the extra rules. The dashed red lines are the region limits. The red dots are the region extremities.

We report the arc lengths of guiding rules onto a graph, represented on figure 5 (bottom left) and compute an increasing interpolation function passing through these points. The monotonicity constraint is important to guarantee that rules do not intersect. We use a piecewise cubic Hermite interpolating polynomial as interpolation function. This function is resampled to get extra rules, limits and extremities of each regions. Region limits are chosen in the middle of two consecutive rules having different labels. The result of resampling is visible on figure 5 (top right).

The interpolation of bending angles is region-dependent. For each region, we represent the bending angles of the guiding rules on a graph, see figure 5 (bottom right). We compute an interpolation function (a spline) passing through the bending angles with the following side conditions to ensure continuity between regions: the bending angles are null at the limit and the extremity of the region. We get the bending angles of extra rules by resampling this curve.

Since all rules have been computed, we split the shape into cells, each cell being a region between two consecutive rules. With this representation, folding the flat mesh is done by rotating and translating each cell. The rigid transformations are formed by composing those induced by each rule starting from a reference cell. Figure 6 shows the result of this last step. Table 2 gives an overview of the surface generation process.

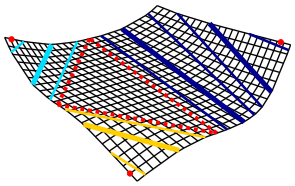


Figure 6. Bent shape with rules. The thick lines are the guiding rules. The thin lines are the extra rules. The dashed red lines are the region limits. The red dots are the region extremities.

SURFACE GENERATION PROCESS
1. Define the shape boundary on the flat mesh
2. Gather the rules into regions
3. Interpolate the rule positions and their angles
4. Resample the interpolating functions to get the extra rules
5. Fold the flat mesh

Table 2. Overview of the surface generation process.

3. A Multiple View Fitting Algorithm

Our goal is to fit the model to multiple images. We assume that a 3D point set and camera pose have been reconstructed from image point features by some means. We use the reprojection error as an optimization criterion. As is usual for dealing with such a nonlinear criterion, we compute a suboptimal initialization that we iteratively refine.

3.1. Initialization

We begin by reconstructing a surface interpolating the given 3D points. A rule detection process is then used to infer our model parameters.

Step 1: Interpolating surface fitting. Details about how the 3D points are reconstructed are given in §4. The interpolating surface is represented by a 2D to 1D Thin-Plate Spline function [2], mapping some planar parameterization of the surface to point height. We use the mean plane. Defining a regular grid on this plane thus allows us to infer a dense set of points on the 3D surface. Figure 7 (top right) and figure 8 (top left) show an example.

Step 2: Model initialization by rule detection. The model is initialized from the 3D surface. The side length is chosen as the size of the 3D mesh.

Guiding rules must be defined on the surface. This set of n rules must represent the surface as accurately as possible. In [3] an algorithm is proposed to find a rule on a given surface. It tries rules with varying direction and passing through several points on the surface . We use it to detect rules along the sites visible on figure 7 (bottom left).

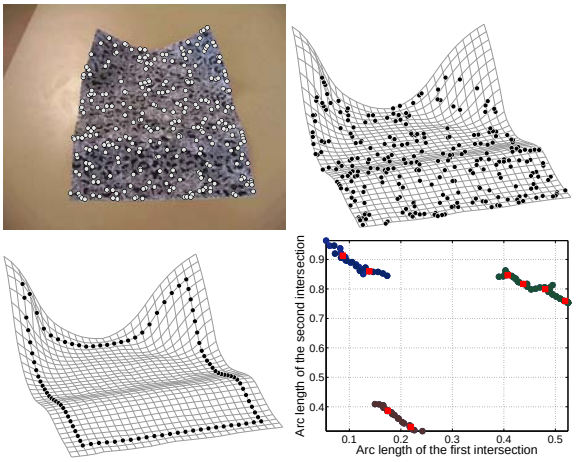


Figure 7. Rules detection process. (top left) Feature points. (top right) Reconstructed 3D points and the interpolating surface. (bottom left) Points where rules are detected on the interpolated surface. (bottom right) Initial detected rules (circles) and automatically detected guiding rules (red squares).

The rules are described by the arc length of their intersection points with the mesh boundary. The two arc lengths defining a rule can be interpreted as a point in $\mathbb{R}^2$, as shown in figure 7 (bottom right). The groups of rules in this figure represent the bending regions of the surface. The guiding rules are chosen in the groups. We fix the number of guiding rules by hand, but a model selection approach could be used to determine it automatically from the set of detected rules.

This gives the n guiding rules. The bending angle vector θ is obtained from the 3D surface by assuming planarity between consecutive rules. The initial suboptimal model we obtain is shown on figure 8 (top right).

3.2. Refinement

The reprojection error describes how well the model fits the actual data, namely the image feature points. We thus introduce latent variables representing the position of each point onto the modeled mesh with two parameters. Let L be the number of images and N_i the number of points in image i, the reprojection error is:

$$e = \sum_{i=1}^{L} \sum_{j=1}^{N_i} (m_{j,i} - \Pi(C_j, M(S, x_i, y_i)))^2. \qquad (2)$$

In this equation, $m_{i,j}$ is the j-th feature point in image i, $\Pi(C,M)$ projects the 3D point M in the camera C and $M(S,x_i,y_i)$ is a twodimensional parameterization of the points lying on the surface, with S the surface parameters. The points on the surface are initialized by computing each (x_i,y_i) such that their individual reprojection error is minimized, using the initial surface model.

To minimize the reprojection error, the following parameters are tuned: the surface parameters (the number of guiding and extra rules is fixed), see table 1, the pose of the surface (rotation and translation) and the 3D point parameters.

The Levenberg-Marquardt algorithm [5] is used to minimize the reprojection error. Upon convergence, the solution is the Maximum Likelihood Estimate under the assumption of an additive *i.i.d.* Gaussian noise on the image feature points.

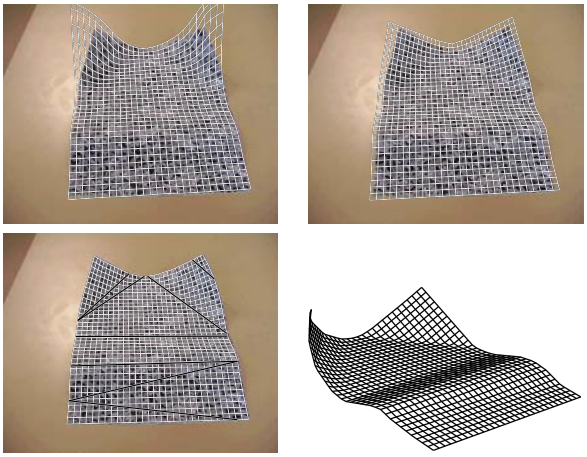


Figure 8. Paper fitting with eight guiding rules. (top left) Interpolated surface. (top right) Initial model. (bottom left) Refined model overlaid with the guiding rules. (bottom right) Estimated model.

4. Experimental Results

We demonstrate the representational power of our fitting algorithm on several sets of images. For five of them, we show results. Some threedimensional representation of the sequence are represented on figures 9 and 14. The 3D point cloud is generated by triangulating point correspondences between several views. These correspondences are obtained while recovering camera calibration and pose using Structure-from-Motion [5]. Points off the object of interest are removed by hand. Figure 7 (top) shows an example of such a reconstruction.

The paper dataset. The following results have been obtained from five views. We used a model with eight guiding rules and sixteen extra rules. Figures 8 and 10 show the reprojection of the 3D surfaces into the first image of the sequence and the reprojection error distribution for the paper sequence for the three main steps of our algorithm: reconstruction with Structure-from-Motion, initialization and refinement. Although the former one has the lowest reprojection error, the associated surface is not satisfying, since it is

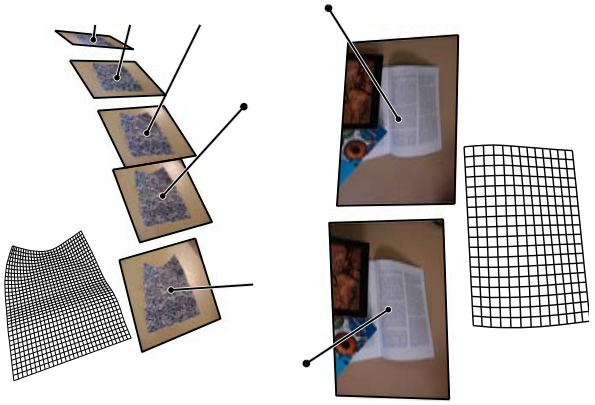


Figure 9. (left) Paper sequence. (right) Book sequence.

not regular enough and does not fit the actual boundary. The initialization makes the model more regular, but is not accurate enough to fit the boundary of the paper, so that important reprojection errors are introduced. Eventually, the refined model is visually acceptable and its reprojection error is very close to the unconstrained set of points obtained by Structure-from-Motion. It means that our model accurately fits the image points, while being governed by a much lower number of parameters than the initial set of independent 3D points. The reprojection error significantly decreases thanks to the refinement step, which validates its relevance. Comparing these errors in the object space leads to the same conclusions: the average distance between the triangulated points and the predicted points before (respectively after) the refinement step is 0.16 cm (respectively 0.06 cm), the paper size being estimated to 25 cm by 21 cm. To make the model converge to the actual paper, we manually selected the four corners in one of the five views.

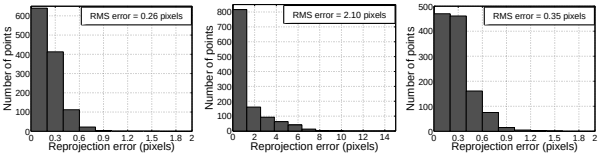


Figure 10. Reprojection errors distribution for the images shown in figure 8. (left) 3D point cloud. (middle) Initial model. (right) Refined model.

Since we have a 3D model of the paper sheet and its reprojection into the images, it is possible to overlay some pictures or to change the texture map. We use the augmentation process described in table 3 to change the whole texture map of the paper and to synthetically generate a view of the paper with the new texture. The results are shown on figure 11.

The book dataset. The second dataset is an image pair of a book. We estimate the page surface with two guiding

AUGMENTING IMAGES
1. Run the proposed algorithm to fit the model to images
2. Choose illumination model and light sources
3. For each image, automatically do
(a) Transfer the new texture map
(b) Apply lighting changes

Table 3. Overview of the augmentation process.

Figure 11. (left) Changing the whole texture map of the paper. (right) Synthetically generated view of the paper with new texture.

rules and eight extra rules. Figure 12 shows the reprojection of the estimated surface and the 3D mesh. The reprojection of the computed model is fine: the reprojection error of the 3D points is 0.26 pixels and the one for the refined model is 0.69 pixels, taking the triangulated points as groung truth, the final error in object space is 0.06 cm for a page size of 18 cm by 13 cm. It means that we accurately recover the page shape with a surface governed by only nine parameters.

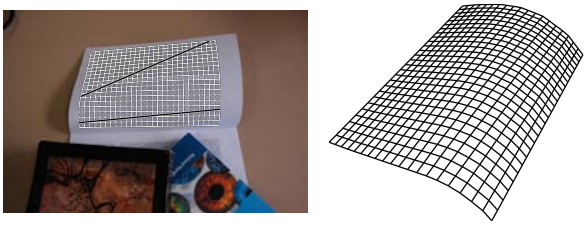


Figure 12. Reconstruction of a book's page. (left) Reprojection onto the images. (right) Estimated model.

One application of the algorithm in the case of a written page is shown on figure 13: our surface estimate is used to unwarp the page's texture and to get a rectified image of the text.

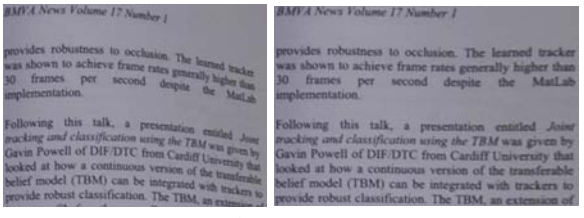


Figure 13. Unwarping. (left) The original page. (right) The rectified page.

The map dataset. The third example is a sequence of a wavely folded map shown on figure 14. Although all parts of the paper are seen in several images, the whole paper is never entirely seen in a single image. The fitting algorithm naturally deals with this kind of occlusion because the initialization is based on the reconstruction of 3D points, and the 3D points cloud is dense enough since all parts of the paper are visible in several views. The missing points do not perturb convergence because the bundle adjustment minimizes the distance between the actual image points and the reprojection of the 3D points. Since for each images, the set of visible feature points is known, only the corresponding 3D points are projected to compute the residual error. The reprojection of the model onto one of the original images is shown on figure 15. Since the 3D model of the surface and the position of the cameras are known it is possible to compute an occlusion map for each images. This is useful to unwarp the texture map from each image and to combine them to get the whole texture map. Some partial texture maps and the whole one are shown in figure 15. The reprojection error of the model is 0.45 pixels, very close to the error of the initial triangulation (0.31 pixels), in object space the refined model error is 0.07 cm for a sheet size of 28 cm by 19 cm.

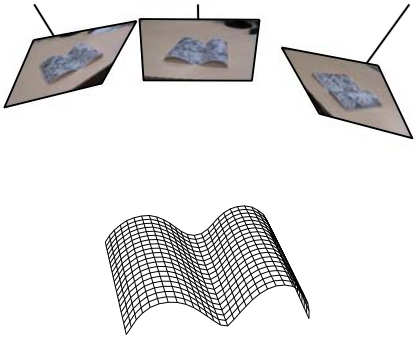


Figure 14. Reconstructed paper and cameras for the map dataset.

The poster dataset. The former examples deal with small paper sheets where the developable constraints are always satisfied. A poster is a more challenging object because singularities may appear on the surface due to its larger size. The input data are two images of the poster obtained from a calibrated stereo system, see figure 16. The surface of the poster is smooth enough, enabling our model to capture the deformations: the RMS error of the triangulated 3D points is 0.35 pixels and the one for our model is 0.65 pixels.

The rug dataset. For this last example, the model is used to estimate a surface whose physical behavior does not satisfy the developable constraints except under special assumption, for example a suspended piece of fabric or in this

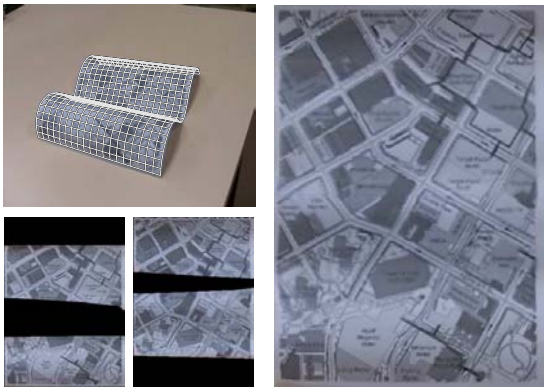


Figure 15. Results for the map dataset. (top left) Reprojection onto one of the original images. (bottom left) Partial texture maps. (right) Unwarped texture map.

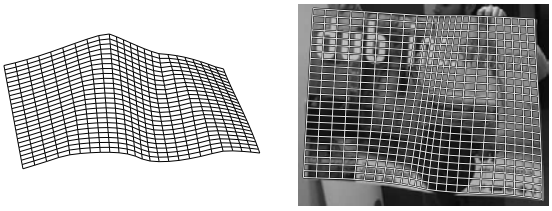


Figure 16. Poster mesh reconstruction. (left) Estimated Model. (right) Reprojection onto the first image.

case an hanged rug. Even though the results are slightly less accurate, the global shape is well-fitted. The difference between the errors of the triangulated points and the model is representative of the lack of accuracy : 0.34 pixels for the original points against 1.36 pixels for the model. This is mainly visible along the boundary of the rug on figure 17.

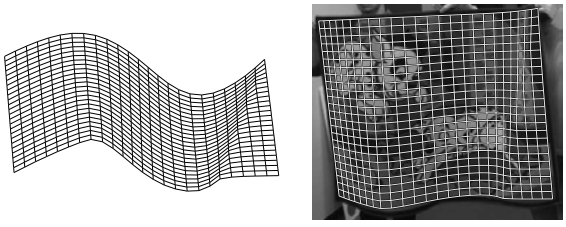


Figure 17. Rug mesh reconstruction. (left) Reprojection onto the first image. (right) Estimated Model.

5. Conclusion and Future Work

This paper describes a quasi-minimal model for paper-like objects and its estimation from multiple images. Although there are few parameters, the generated surface is a good approximation to smoothly deformed paper-like objects. This is demonstrated on real image datasets thanks to a fitting algorithm which initializes the model and refines it in a bundle adjustment manner. Both a surface and its boundary curve are inferred from images.

There are many possibilities for further research. The proposed model could be embedded in a monocular tracking framework or used to generate sample meshes for a surface learning model. The fitting algorithm should be compared to other surface models and estimation methods, in terms of computation and accuracy performances.

References

[1] V. Blanz and T. Vetter. Face recognition based on fitting a 3D morphable model. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 25(9), September 2003. 1

[2] F. L. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 11(6):567–585, June 1989. 4

[3] H.-Y. Chen and H. Pottmann. Approximation by ruled surfaces. *Journal of Computational and Applied Mathematics*, 102:143–156, 1999. 1, 2, 4

[4] N. A. Gumerov, A. Zandifar, R. Duraiswami, and L. S. Davis. Structure of applicable surfaces from single views. In *Proceedings of the European Conference on Computer Vision*, 2004. 1

[5] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003. Second Edition. 5

[6] Y. Kergosien, H. Gotoda, and T. Kunii. Bending and creasing virtual paper. IEEE *Computer Graphics & Applications*, 14(1):40–48, 1994. 2

[7] S. Leopoldseder and H. Pottmann. Approximation of developable surfaces with cone spline surfaces. *Computer-Aided Design*, 30:571–582, 1998. 1, 2

[8] M. Pilu. Undoing page curl distortion using applicable surfaces. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, December 2001. 1, 2

[9] H. Pottmann and J. Wallner. Approximation algorithms for developable surfaces. *Computer Aided Geometric Design*, 16:539–556, 1999. 1, 2

[10] M. Salzmann, S. Ilic, and P. Fua. Physically valid shape parameterization for monocular 3-D deformable surface tracking. In *Proceedings of the British Machine Vision Conference*, 2005. 1

[11] M. Sun and E. Fiume. A technique for constructing developable surfaces. In *Proceedings of Graphics Interface*, pages 176–185, May 1996. 1, 2

7.2.2 Paper (ICRA'06) – *Towards 3D Motion Estimation from Deformable Surfaces*

Towards 3D Motion Estimation From Deformable Surfaces

Adrien Bartoli
CNRS / LASMEA – Adrien.Bartoli@gmail.com
24, avenue des Landais – 63177 Aubière cedex, France

***Abstract*— Estimating the pose of an imaging sensor is a central research problem. Many solutions have been proposed for the case of a rigid environment. In contrast, we tackle the case of a non-rigid environment observed by a 3D sensor, which has been neglected in the literature. We represent the environment as sets of time-varying 3D points explained by a low-rank shape model, that we derive in its implicit and explicit forms. The parameters of this model are learnt from data gathered by the 3D sensor. We propose a learning algorithm based on minimal 3D non-rigid tensors that we introduce. This is followed by a Maximum Likelihood nonlinear refinement performed in a bundle adjustment manner. Given the learnt environment model, we compute the pose of the 3D sensor, as well as the deformations of the environment, that is, the non-rigid counterpart of pose, from new sets of 3D points. We validate our environment learning and pose estimation modules on simulated and real data.**

I. INTRODUCTION

Aligning 3D views – sets of 3D points – gathered by a 3D sensor, such as a calibrated stereo rig, is important for constructing comprehensive 3D models of the environment or updating the position of a mobile imaging sensor. When the environment is rigid, the 3D views are related by rigid Euclidean transformations. Many approaches have been proposed to compute these transformations, *e.g.* [1]. Aligning 3D views is one of the building blocks of hierarchical approaches to Structure-From-Motion. However, the assumption of rigidity is violated in many cases of interest, for instance a garment deforming as a person moves. The alignment problem is then particularly challenging because a different shape is observed in each 3D view.

A large body of work has been done in the medical imaging community but with the aim of estimating dense deformation fields from dense, often voxel-based, reconstructions. Dealing with non-rigid scenes coming from single-camera footage has received an increasing attention over the last few years. The problem is highly challenging since both the cameras and the non-rigid shape have to be recovered. A major step forwards for such cases was made by Bregler *et al.* [2] and Brand [3]. Building on the work of [4], they developed and demonstrated factorization of images of non-rigid scenes, where the non-rigidity was represented as a linear combination of basis shapes. It is shown in [5] how the constraints coming from two synchronized cameras can be incorporated into non-rigid factorization.

We tackle the problem of computing the pose of a 3D sensor with respect to a non-rigid scene, that we represent using the low-rank shape model used in non-rigid factorization methods. Most previous work, *e.g.* [3], [2], [5], [6] use the weak perspective camera model. In contrast, we do not specify a camera model, since we directly consider 3D views. We assume that spatial and temporal point correspondences are established. Pose estimation in a non-rigid environment raises two main problems. First, one has to define the meaning of non-rigid pose. One benefit of using the low-rank shape model is that the 'true' camera pose is recovered. Second, contrarily to classical model-based pose estimation in a rigid environment, a prior model of the non-rigid environment is not available in many cases. We propose to learn this model from a collection of unregistered 3D views gathered by the 3D sensor. Once this learning stage has been passed, our non-rigid pose estimator can be launched.

We bring the following contributions. First, §III, we state the implicit and explicit low-rank shape models, and state the notion of pose in this context. Second, §IV, we propose algorithms to learn the non-rigid environment. The implicit model parameters are learnt using a factorization technique, while for the explicit model, we use what we call *minimal 3D non-rigid tensors*. Third, §V, we show how the pose of the 3D sensor can be computed with respect to the learnt model while the environment is moving and deforming. Experimental results on simulated and real data are reported in §VI. We give our conclusions in §VII.

II. NOTATION

Matrices are written in sans-serif fonts, *e.g.* R, and vectors using bold fonts, *e.g.* $\mathbf{x}$. The n 3D views are sets of m points denoted $\mathbf{Q}_{tj}$, where t is the time index and j the point index. We do not use homogeneous coordinates, *e.g.* $\mathbf{Q}_{tj}$ is a 3-vector. The identity matrix of size $(s \times s)$ is written $\mathrm{I}_{(s)}$, the zero matrix $\mathsf{0}$ and the zero vector $\mathbf{0}$. We use I for the (3×3) identity matrix. The Kronecker product is written $\otimes$, matrix Frobenius norm as $\|\cdot\|$ and the Moore-Penrose pseudo-inverse as †.

III. NON-RIGID SHAPE AND POSE

A. Non-Rigid Shape

We describe the low-rank non-rigid shape model. The pose of the 3D sensor is modeled by 3D Euclidean transformations $\{(\mathsf{R}_t, \mathbf{y}_t)\}$ with R_t an orthonormal matrix and $\mathbf{y}_t \in \mathbb{R}^3$ such that $\hat{\mathbf{Q}}_{tj} = \mathsf{R}_t \tilde{\mathbf{Q}}_{tj} + \mathbf{y}_t$. The $\{\tilde{\mathbf{Q}}_{tj}\}$ form a motionless version of the 3D views, *i.e.* that do not undergo any 'global motion',

but are deforming through time. The low-rank shape model represents the $\{\tilde{\mathbf{Q}}_{tj}\}$ as linear combinations of l *basis shapes* $\{\mathbf{B}_{kj}\}$: $\tilde{\mathbf{Q}}_{tj} = \sum_{k=1}^{l} \xi_{tk}\mathbf{B}_{kj}$. The time-varying $\{\xi_{tk}\}$ are the *configuration weights*. Introducing the $\{(\mathsf{R}_t, \mathbf{y}_t)\}$, we obtain the *explicit model*:

$$\hat{\mathbf{Q}}_{tj} = \mathsf{R}_t \left(\sum_{k=1}^{l} \xi_{tk}\mathbf{B}_{kj} \right) + \mathbf{y}_t \quad (1)$$

$$= \mathsf{M}_t\mathbf{B}_j + \mathbf{y}_t \quad \text{with} \quad (2)$$

$$\mathsf{M}_t = \mathsf{R}_t \begin{pmatrix} \xi_{t1}\mathrm{I} & \cdots & \xi_{tl}\mathrm{I} \end{pmatrix}. \quad (3)$$

We call M_t a $(3 \times r)$ *explicit non-rigid motion matrix* and $\mathbf{B}_j = \begin{pmatrix} \mathbf{B}_{1j}^{\mathsf{T}} & \cdots & \mathbf{B}_{lj}^{\mathsf{T}} \end{pmatrix}$ a $(r \times 1)$ *non-rigid basis shape vector*. Parameter $r = 3l$ is the *rank* of the model. For reasons that are made clearer below, we derive a bilinear *implicit model*. Let $\mathcal{A}$ be a $(3l \times 3l)$ rank-$3l$ matrix. It is seen that $\hat{\mathbf{Q}}_{tj} = \mathsf{M}_t\mathbf{B}_j + \mathbf{y}_t = (\mathsf{M}_t\mathcal{A}^{-1})(\mathcal{A}\mathbf{B}_j) + \mathbf{y}_t$, yielding:

$$\hat{\mathbf{Q}}_{tj} = \mathsf{N}_t\mathbf{S}_j + \mathbf{y}_t, \quad (4)$$

with $\mathsf{N}_t = \mathsf{M}_t\mathcal{A}^{-1}$ and $\mathbf{S}_j = \mathcal{A}\mathbf{B}_j$. We call N_t and $\mathbf{S}_j$ the *implicit* non-rigid motion matrix and shape vector, and $\mathcal{A}$ a *corrective transformation matrix*.

B. Non-Rigid Pose

Pose in a non-rigid environment has a rigid and a non-rigid counterpart. The rigid part $\{(\mathsf{R}_t, \mathbf{y}_t)\}$ represents the 'global' motion of the environment relative to the sensor. It gives the 'true' *relative* sensor displacement. In contrast, the non-rigid part only concerns the environment, and not the imaging sensor. In the above-described model, it is represented by the configuration weights $\{\xi_{tk}\}$, giving the intrinsic, *i.e.* motionless, deformations of the environment at each time instant. The motionless and deformationless environment is modeled by the basis shapes $\{\mathbf{B}_{kj}\}$.

The implicit model is useless for pose estimation: it can be seen as an 'uncalibrated' model of the environment. However, its ML (Maximum Likelihood) Estimate can be computed very reliably, as will be seen in the next section.

IV. LEARNING THE ENVIRONMENT

Given a collection of 3D views, we learn the environment by estimating the parameters of the low-rank shape model. Note that only the basis shapes $\{\mathbf{B}_{kj}\}$ are subsequently used for pose estimation, see §V. However, to get an ML Estimate, all parameters of the model must be computed.

We state the ML residual error and show how to compute the translations. We first tackle the case of the implicit model and then the explicit one. We assume all points to be visible in all 3D views.

A. Maximum Likelihood residual error

Assuming that the error on the 3D points is Gaussian, centred and i.i.d., the ML residual error is:

$$\mathcal{D}^2 = \frac{1}{nm} \sum_{t=1}^{n} \sum_{j=1}^{m} d^2(\hat{\mathbf{Q}}_{tj}, \mathbf{Q}_{tj}), \quad (5)$$

where $d^2(\mathbf{X}, \mathbf{Y}) = \|\mathbf{X} - \mathbf{Y}\|^2$ is the Euclidean distance measure and $\{\hat{\mathbf{Q}}_{tj}\}$ are corrected points, exactly explained by the non-rigid shape model.

B. Computing the Translations

We show that the translations $\mathbf{y}_t$ can be eliminated prior to estimating the other parameters. By substituting equation (1) or equation (4) in the residual error (5) and nullifying its partial derivatives with respect to $\mathbf{y}_t$, we obtain $\mathbf{y}_t = \frac{1}{m}\left(\sum_{j=1}^{m} \mathbf{Q}_{tj} - \hat{\mathbf{Q}}_{tj}\right)$. The origin of the r-dimensional space containing the non-rigid shape vectors is arbitrary and is chosen such that $\sum_{j=1}^{m} \mathbf{S}_j = \mathbf{0}$ in the implicit case and $\sum_{j=1}^{m} \mathbf{B}_j = \mathbf{0}$ in the explicit case, giving for the translation $\mathbf{y}_t$ the centroid $\mathbf{y}_t = \frac{1}{m}\sum_{j=1}^{m} \mathbf{Q}_{tj} = \bar{\mathbf{Q}}_t$ of the t-th 3D view. This means that one cancels the translations out by centring each set of 3D points on its centroid: $\mathbf{Q}_{tj} \leftarrow \mathbf{Q}_{tj} - \bar{\mathbf{Q}}_t$. Henceforth, we assume that this has been done.

C. Shape Learning With the Implicit Model

We consider the implicit non-rigid shape model of equation (4). We factorize the 3D views $\{\mathbf{Q}_{tj}\}$ into implicit non-rigid motion matrices $\{\mathsf{N}_t\}$ and shape vectors $\{\mathbf{S}_j\}$. The problem is to minimize the ML residual error (5) over the $\{\hat{\mathbf{Q}}_{tj}\}$ such that $\hat{\mathbf{Q}}_{tj} = \mathsf{N}_t\mathbf{S}_j$. Rewrite (5) as:

$$\mathcal{D}^2 \propto \|\hat{\mathcal{Q}} - \mathcal{Q}\|^2,$$

where $\mathcal{Q}$ is the $(3n \times m)$ *measurement matrix*:

$$\mathcal{Q} = \begin{pmatrix} \mathbf{Q}_{11} & \cdots & \mathbf{Q}_{1m} \\ \vdots & \ddots & \vdots \\ \mathbf{Q}_{n1} & \cdots & \mathbf{Q}_{nm} \end{pmatrix},$$

and $\hat{\mathcal{Q}}$ is defined by the implicit $(3n \times 3l)$ 'non-rigid joint motion matrix' $\mathcal{N}$ and the $(3l \times m)$ 'non-rigid joint structure matrix' $\mathcal{S}$ as $\hat{\mathcal{Q}} = \mathcal{N}\mathcal{S}$ with $\mathcal{N}^{\mathsf{T}} = \begin{pmatrix} \mathsf{N}_1^{\mathsf{T}} & \cdots & \mathsf{N}_n^{\mathsf{T}} \end{pmatrix}$ and $\mathcal{S} = \begin{pmatrix} \mathbf{S}_1 & \cdots & \mathbf{S}_m \end{pmatrix}$. Since $\mathcal{N}$ has $3l$ columns and $\mathcal{S}$ has $3l$ rows, $\hat{\mathcal{Q}}$ has maximum rank $3l$. The problem is to find the closest rank-$3l$ matrix $\hat{\mathcal{Q}}$ to $\mathcal{Q}$. Let $\mathcal{Q} = \mathsf{U}\Sigma\mathsf{V}^{\mathsf{T}}$ be a Singular Value Decomposition (SVD) of matrix $\mathcal{Q}$, see *e.g.* [7], where U and V are orthonormal matrices and Σ is diagonal and contains the singular values of $\mathcal{Q}$. Let $\Sigma = \Sigma_u\Sigma_v$ be any decomposition of Σ, *e.g.* $\Sigma_u = \Sigma_v = \sqrt{\Sigma}$. The non-rigid joint motion and structure matrices are obtained by, loosely speaking, 'truncating' the decomposition by nullifying all but the $3l$ largest singular values, which leads, assuming the singular values in decreasing order in Σ, to $\mathcal{N} = \psi_{3l}(\mathsf{U}\Sigma_u)$ and $\mathcal{S} = \psi_{3l}^{\mathsf{T}}(\mathsf{V}\Sigma_v^{\mathsf{T}})$, where $\psi_c(\mathsf{W})$ is formed with the c leading columns of matrix W.

D. Shape Learning With the Explicit Model

The aim is to compute the ML Estimate of the configuration weights, rotation matrices and non-rigid structure in equation (1) by minimizing the residual error (5). This is a nonlinear problem for which two approaches have been followed in the non-rigid factorization litterature. On the one hand Bregler *et al.* [2], Brand [3], Aanaes *et al.* [8], Del Bue *et al.* [5] and

Xiao *et al.* [6] compute a matrix $\mathcal{A}$ that upgrades the implicit motion matrix $\mathcal{N}$ so that the metric constraints of the explicit model are enforced. Xiao *et al.* show that in order to get the correct solution, two types of metric constraints must be taken into account: the *rotation constraints* and the *basis constraints*, from which they derive a closed-form solution for matrix $\mathcal{A}$.

On the other hand, Torresani *et al.* [9] directly learn the parameters of the explicit model. They propose a comprehensive system based on a generalized EM (Expectation Maximization) algorithm. An important, still unsolved problem is to find a suitable initialization, since EM performs local optimization only.

Our solution lies in the second category: a suboptimal initialization is computed and subsequently refined in a bundle adjustment manner. These two steps are presented below, followed by an analysis of the ambiguities of the solution.

1) Initializing:

a) The rotations: Brand proposes a solution based on upgrading the implicit motion matrices [3], which requires at least $n \geq \frac{l(9l+3)}{4}$ 3D views to compute a corrective transformation and is thus not feasible for many practical cases. For example, at least 39 views giving independent constraints are necessary to use this method with the sequence presented in §VI-B. In [5], the authors compute a block-diagonal corrective transformation matrix. Another solution used in [8] is to assume that the environment has a sufficiently strong rigid component, and to estimate the rotation using a standard procedure such as [1]. This approach is not feasible for highly deforming environments.

In contrast, we propose an approach taking the non-rigid nature of the environment into account. Our algorithm is presented below in the occlusion-free case for simplicity, but can be easily extended to the missing data case. Consider the explicit non-rigid joint motion equation $\mathcal{Q} = \mathcal{M}\mathcal{B}$ with:

$$\mathcal{M} = \begin{pmatrix} \xi_{11}\mathsf{R}_1 & \cdots & \xi_{1l}\mathsf{R}_1 \\ \vdots & \ddots & \vdots \\ \xi_{n1}\mathsf{R}_n & \cdots & \xi_{nl}\mathsf{R}_n \end{pmatrix} \quad \text{and} \quad \mathcal{B} = \begin{pmatrix} \mathbf{B}_{11} & \cdots & \mathbf{B}_{1m} \\ \vdots & \ddots & \vdots \\ \mathbf{B}_{l1} & \cdots & \mathbf{B}_{lm} \end{pmatrix}.$$

Define two subsets $\mathbb{A}$ and $\mathbb{B}$ of n_a and n_b 3D views respectively, $\mathcal{Q}_a = \mathcal{M}_a\mathcal{B}$ and $\mathcal{Q}_b = \mathcal{M}_b\mathcal{B}$. Our goal is to eliminate the structure $\mathcal{B}$ from the equations. We assume without loss of generality $\mathrm{rank}(\mathcal{Q}_b) \geq 3l$. This implies $n_b \geq l$. We express $\mathcal{B}$ in terms of $\mathcal{Q}_b$ and $\mathcal{M}_b$ using the equation subset $\mathbb{B}$ as $\mathcal{B} = \mathcal{M}_b^\dagger\mathcal{Q}_b$. Plugging this into the equation subset $\mathbb{A}$ yields $\mathcal{Q}_a = \mathcal{M}_a\mathcal{B} = \mathcal{M}_a\mathcal{M}_b^\dagger\mathcal{Q}_b$ that we rewrite:

$$\underbrace{\begin{pmatrix} \mathrm{I}_{(3n_a)} & -(\mathcal{M}_a\mathcal{M}_b^\dagger) \end{pmatrix}}_{\mathcal{Z}} \mathcal{Q}_{ab} \quad = \quad \mathsf{0}_{(3n_a \times m)} \tag{6}$$

where $n_{ab} = n_a + n_b$ and $\mathcal{Q}_{ab}^\mathsf{T} = (\mathcal{Q}_a^\mathsf{T} \;\; \mathcal{Q}_b^\mathsf{T})$. We call matrix $\mathcal{Z}_{(3n_a \times 3n_{ab})}$ a *3D non-rigid tensor*. Let us examine more closely the expression of $\mathcal{M}_a\mathcal{M}_b^\dagger$. The joint motion matrix can be rewritten as $\mathcal{M} = \mathcal{R}(\Xi \otimes \mathrm{I})$ where $\mathcal{R} = \mathrm{diag}(\mathsf{R}_1, \ldots, \mathsf{R}_n)$ is an orthonormal matrix and Ξ is an $(n \times l)$ matrix containing the $\{\xi_{ik}\}$. Similarly, $\mathcal{M}_a = \mathcal{R}_a(\Xi_a \otimes \mathrm{I})$ and $\mathcal{M}_b = \mathcal{R}_b(\Xi_b \otimes \mathrm{I})$, yielding:

$$\begin{aligned} \mathcal{M}_a\mathcal{M}_b^\dagger &= \mathcal{R}_a(\Xi_a \otimes \mathrm{I})(\mathcal{R}_b(\Xi_b \otimes \mathrm{I}))^\dagger \\ &= \mathcal{R}_a(\Xi_a \otimes \mathrm{I})((\Xi_b \otimes \mathrm{I}))^\dagger\mathcal{R}_b^\mathsf{T}, \end{aligned}$$

since $\mathcal{R}_b$ is an orthonormal matrix. We make use of the following properties: $(\mathsf{S} \otimes \mathrm{I})^\dagger = \mathsf{S}^\dagger \otimes \mathrm{I}$ and $(\mathsf{S} \otimes \mathrm{I})(\mathsf{S}' \otimes \mathrm{I}) = (\mathsf{S}\mathsf{S}') \otimes \mathrm{I}$ to get:

$$\mathcal{M}_a\mathcal{M}_b^\dagger \quad = \quad \mathcal{R}_a\left((\Xi_a\Xi_b^\dagger) \otimes \mathrm{I}\right)\mathcal{R}_b^\mathsf{T}.$$

Substituting in equation (6) and multiplying on the left by the orthonormal $\mathcal{R}_a^\mathsf{T}$ yields:

$$\left(\mathcal{R}_a^\mathsf{T} \quad -\left((\Xi_a\Xi_b^\dagger) \otimes \mathrm{I}\right)\mathcal{R}_b^\mathsf{T}\right)\mathcal{Q}_{ab} \quad = \quad \mathsf{0}_{(3n_a \times m)}. \tag{7}$$

From this equation, knowing $\mathcal{R}_a$ and using the orthonormality constraints on $\mathcal{R}_b$ to eliminate the weights $\Xi_a\Xi_b^\dagger$ should allow to compute $\mathcal{R}_b$. We use the fact that the coordinate frame can be aligned on a reference view t, *i.e.* such that $\mathsf{R}_t = \mathrm{I}$ and choose one view in the initial set of 3D views $\mathbb{A}$ to be the reference one.

The first idea that comes to mind to solve this problem is to consider the left nullspace of $\mathcal{Q}_{ab}$. Define a $(3n_{ab} \times (3n_{ab} - 3l))$ matrix U whose columns span the left nullspace of $\mathcal{Q}_{ab}$: $\mathsf{U}^\mathsf{T}\mathcal{Q}_{ab} = \mathsf{0}$. Using equation (7), we obtain:

$$\left(\mathcal{R}_a^\mathsf{T} \quad -\left((\Xi_a\Xi_b^\dagger) \otimes \mathrm{I}\right)\mathcal{R}_b^\mathsf{T}\right) \quad = \quad \mathsf{H}\mathsf{U}^\mathsf{T},$$

where H accounts for the fact that any linear combination of the columns of U are in the left nullspace of $\mathcal{Q}_{ab}$. While this approach works fine in the absence of noise contaminating the data, it is however very unstable and useless when even very slight noise is present in the data. Indeed, if one employs *e.g.* SVD to compute matrix U, then the singular vectors corresponding to the lowest singular values will be selected, and will not in general allow to recover the sought-after rotations, since the SVD mixes the singular vectors to obtain the lowest residual error as possible.

The second idea that comes to mind is to estimate each rotation in $\mathbb{B}$ and the corresponding weight at a time. Consider a 3D view $g \in \mathbb{A}$. Equation (7) induces the following residual error:

$$\sum_{j=1}^{m} \|\mathsf{R}_g^\mathsf{T}\mathbf{Q}_{gj} - \sum_{t \in \mathbb{B}} \zeta_t \mathsf{R}_t^\mathsf{T}\mathbf{Q}_{tj}\|^2, \tag{8}$$

where $\{\zeta_t\}$ are unknown weights. Initialize all rotations in $\mathcal{R}_b$ to the identity: $\mathsf{R}_t^0 = \mathrm{I}$, $t \in \mathbb{B}$. Let $p \leftarrow 0$ be the iteration counter. The idea is to iteratively compute the t-th rotation for $t \in \mathbb{B}$ while holding the other $n_b - 1$ rotations in $\mathbb{B}$ until convergence, by minimizing the residual error (8) that we rewrite:

$$\sum_{j=1}^{m} \|\mathbf{E}_j^p - \zeta_t^{p+1}\left(\mathsf{R}_t^{p+1}\right)^\mathsf{T}\mathbf{Q}_{tj}\|^2 \tag{9}$$

with:

$$\mathbf{E}_j^p \quad = \quad \mathsf{R}_g^\mathsf{T}\mathbf{Q}_{gj} - \left(\sum_{t \in \mathbb{B}} (\mathsf{S}_t^p)^\mathsf{T}\mathbf{Q}_{tj}\right), \tag{10}$$

where S_t^p is the latest estimate, *i.e.* :

$$\mathsf{S}_t^p = \begin{cases} \zeta_t^{p+1}\mathsf{R}_t^{p+1} & \text{if it is computed} \\ \zeta_t^p \mathsf{R}_t^p & \text{otherwise.} \end{cases}$$

We use a standard procedure for computing the 3D rotation and scale from 3D point correspondences – here $\{\mathbf{E}_j^p \leftrightarrow \mathbf{Q}_{ij}\}$ – due to [1] to solve this problem. Our algorithm is summarized in table I. Note that at most l rotations in $\mathcal{R}_b$ can be computed at each iteration which implies that the number of rotations in $\mathcal{R}_b$ must be l. This is why only the smallest, *i.e.* minimal 3D non-rigid tensors can be used by our algorithm. Also, the unknown $\mathcal{M}_b$ must be full-rank. We use the corresponding implicit $\mathcal{N}_b$ to check that this is the case, since there exists a full-rank corrective transformation matrix $\mathcal{A}$ such that $\mathcal{N}_a\mathcal{A} = \mathcal{M}_a$. In the case of missing data, the sum in equations (8) and (9) is simply replaced by a sum over the points seen in subsets $\mathbb{A}$ and $\mathbb{B}$.

OBJECTIVE

Given n 3D views $\{\mathbf{Q}_{tj}\}$ of m corresponding points and the rank $3l$ of the non-rigid model, compute the relative pose $\{(\mathsf{R}_t, \mathbf{y}_t)\}$ of the 3D sensor, the non-rigid pose of the environment, *i.e.* the configuration weights $\{\xi_{tk}\}$, while learning the low-rank non-rigid shape model $\{\mathbf{B}_{kj}\}$.

ALGORITHM

1) **Set initial equation sets.** $\mathbb{A}$ is any 3D view t, $\mathbb{B}$ is any l 3D views at least one of them not in $\mathbb{A}$ and such that $\mathcal{N}_b$ is full-rank, $\mathsf{R}_t \leftarrow \mathrm{I}$ and $\mathcal{R}_a \leftarrow \mathrm{I}$.
2) **Compute the rotations:**
   a) Set initial rotations $\mathsf{R}_t^0 \leftarrow \mathrm{I}$, $t \in \mathbb{B}$ and the iteration counter $p \leftarrow 0$.
   b) For $t \in \mathbb{B}$: form the $\{\mathbf{E}_j^p\}$, equation (10). Compute R_t^{p+1} by minimizing (9), see Horn *et al.* [1].
   c) $p \leftarrow p + 1$.
   d) If the decrease in the residual error is smaller than ε, go to step 3 else go to step b.
3) **Test convergence.** If all rotations are computed, stop.
4) **Update equation sets.** $\mathbb{A} \leftarrow \mathbb{A} \cup \mathbb{B}$ and $\mathbb{B}$ is any l 3D views, at least one of them not in $\mathbb{A}$ and such that $\mathcal{N}_b$ is full-rank.
5) **Iterate.** go to step 2.

TABLE I
THE PROPOSED INITIALIZATION ALGORITHM FOR THE EXPLICIT MODEL PARAMETERS.

b) The configuration weights and non-rigid structure: Consider the ML residual error (5) that we rewrite below for convenience:

$$\mathcal{D}^2 = \frac{1}{nm}\sum_{t=1}^{n}\sum_{j=1}^{m} \|\mathbf{Q}_{tj} - \mathsf{R}_t\left(\sum_{k=1}^{l}\xi_{tk}\mathbf{B}_{kj}\right)\|^2.$$

Let $\tilde{\mathbf{Q}}_{tj} = \mathsf{R}_t^\mathsf{T}\mathbf{Q}_{tj}$ be a motionless version of the 3D points, the residual error transforms in:

$$\mathcal{D}^2 = \frac{1}{nm}\sum_{t=1}^{n}\sum_{j=1}^{m} \|\tilde{\mathbf{Q}}_{tj} - \left(\sum_{k=1}^{l}\xi_{tk}\mathbf{B}_{kj}\right)\|^2.$$

Introduce matrices $\mathcal{L}_{(n\times 3m)}$ and $\mathcal{T}_{(l\times 3m)}$ which are obtained by reorganizing $\tilde{\mathcal{Q}}$ and $\mathcal{B}$, respectively:

$$\tilde{\mathcal{L}} = \begin{pmatrix} \tilde{\mathbf{Q}}_{11}^\mathsf{T} & \cdots & \tilde{\mathbf{Q}}_{1m}^\mathsf{T} \\ \vdots & \ddots & \vdots \\ \tilde{\mathbf{Q}}_{n1}^\mathsf{T} & \cdots & \tilde{\mathbf{Q}}_{nm}^\mathsf{T} \end{pmatrix} \quad \text{and} \quad \mathcal{T} = \begin{pmatrix} \mathbf{B}_{11}^\mathsf{T} & \cdots & \mathbf{B}_{1m}^\mathsf{T} \\ \vdots & \ddots & \vdots \\ \mathbf{B}_{l1}^\mathsf{T} & \cdots & \mathbf{B}_{lm}^\mathsf{T} \end{pmatrix}.$$

The residual error is rewritten $\mathcal{D}^2 = \frac{1}{nm}\|\tilde{\mathcal{L}} - \Xi\mathcal{T}\|^2$. This means that matrix $\tilde{\mathcal{L}}$ has rank l at most. Similarly to §IV-C, let $\tilde{\mathcal{L}} = \mathsf{U}\Sigma\mathsf{V}^\mathsf{T}$ be an SVD of matrix $\tilde{\mathcal{L}}$, we get $\Xi = \psi_l(\mathsf{U}\Sigma_u)$ and $\mathcal{T} = \psi_l^\mathsf{T}(\mathsf{V}\Sigma_v^\mathsf{T})$.

2) Bundle Adjustment: Starting from the above-derived initial solution, we minimize the ML residual error (5) using nonlinear least-squares in a bundle adjustment manner, see *e.g.* [10]. We use the Levenberg-Marquardt algorithm, implemented to exploit the sparse block structure of the Jacobian and (Gauss-Newton approximation of) the Hessian matrices. Bundle adjustment in the non-rigid case is developed in [8], [5], where the authors show that compared to the rigid case, additional 'gauge freedoms' in the recovered structure and motion must be handled. However, the Levenberg-Marquardt optimization engine deals with those by damping the approximated Hessian matrix which makes it full rank. We found that the regularization term employed in [5] does not have a significant effect on the results we obtained. This is mainly due to the fact that we directly use 3D data, while [5] use image points.

3) Ambiguities of the Solution: The ambiguity of the solution demonstrated by Xiao *et al.* [6] in the 2D case when only the rotation constraints are used does not hold for our algorithm. The reason is that it enforces the replicated block structure of the joint motion matrix $\mathcal{M}$, which provides stronger constraints than the rotation constraints only. The ambiguity matrix $\mathcal{E}$ on the learnt model is $\mathcal{E} = \mathrm{diag}_l(\mathsf{S})(\Lambda_{(l\times l)} \otimes \mathrm{I})$, where $\mathrm{diag}_l(\mathsf{S})$ is a l block diagonal matrix for some 3D orthonormal matrix S, representing the indeterminateness of the orientation for the global coordinate frame. Matrix $\Lambda_{(l\times l)} \otimes \mathrm{I}$ models linear combinations of the basis shapes. This shows that it is not possible to recover the 'true' basis shapes and configuration weights, but that 'true' camera pose can still be computed.

V. COMPUTING POSE

Given the non-rigid model of the environment – the basis shapes $\{\mathbf{B}_{kj}\}$ – and a 3D view $\{\mathbf{Q}_j\}$, we want to estimate the pose of the 3D sensor, namely the Euclidean transformation $(\mathsf{R}, \mathbf{y})$, jointly with the non-rigid counterpart of the pose, *i.e.* the configuration weights $\{\xi_k\}$. Note that we drop index t since only one 3D view is considered in this section. It is not necessary to observe all points used in the learning phase to compute pose. The ML residual error is:

$$\mathcal{C}^2 = \frac{1}{m}\sum_{j=1}^{m} d^2(\mathsf{M}\mathbf{B}_j + \mathbf{y}, \mathbf{Q}_j). \tag{11}$$

It must be minimized over $(\mathsf{R}, \mathbf{y})$ and $\{\xi_k\}$. Matrix M is defined by equation (3).

We propose to nonlinearly minimize the ML residual error (11) using the Levenberg-Marquardt algorithm. It is not possible to use a direct estimator as [1] due to the configuration weights. Note that, as shown below, the translation $\mathbf{y}$ can be eliminated from the equation. The minimization is thus performed over R and $\{\xi_k\}$. Such an algorithm as Levenberg-Marquardt requires one to provide an initial solution. Our algorithm for finding it is described below.

a) Eliminating the translation: The derivatives of the ML residual error (11) with respect to $\mathbf{y}$ must vanish: $\frac{\partial \mathcal{C}^2}{\partial \mathbf{y}} = \mathbf{0}$, which leads to $\mathbf{y} = \frac{1}{m}\sum_{j=1}^{m}(\mathbf{Q}_j - \mathsf{M}\mathbf{B}_j)$. This result means that $\mathbf{y}$ is given by the difference between the centroid of the points $\{\mathbf{Q}_j\}$ and the centroid predicted by the points from the shape model $\{\mathsf{M}\mathbf{B}_j\}$, which vanishes if the set of points used for computing the pose is exactly the same as the one used in the learning phase. In any case, by centring the points on their centroid, the translation vanishes. Henceforth, we assume this has been done and rewrite the ML residual error (11) as:

$$\mathcal{C}^2 \quad = \quad \frac{1}{m}\sum_{j=1}^{m} d^2(\mathsf{M}\mathbf{B}_j, \mathbf{Q}_j). \qquad (12)$$

b) Initializing the rotation and configuration weights: We linearly compute a motion matrix $\tilde{\mathsf{M}}$ without enforcing the correct replicated structure by $\min_{\tilde{\mathsf{M}}} \sum_{j=1}^{m} d^2(\tilde{\mathsf{M}}\mathbf{B}_j, \mathbf{Q}_j)$, which yield:

$$\tilde{\mathsf{M}} \quad = \quad \begin{pmatrix}\mathbf{Q}_1 & \cdots & \mathbf{Q}_m\end{pmatrix}\begin{pmatrix}\mathbf{B}_1 & \cdots & \mathbf{B}_m\end{pmatrix}^{\dagger}.$$

We extract the $\{\xi_k\}$ and R from $\tilde{\mathsf{M}}$ by solving $\min_{\mathsf{R},\{\xi_k\}} \sum_{k=1}^{l} \|\tilde{\mathsf{M}}_k - \xi_k \mathsf{R}\|^2$, where the $\tilde{\mathsf{M}}_k$ are (3×3) blocks from $\tilde{\mathsf{M}}$. By vectorizing and reorganizing the residual error, we obtain:

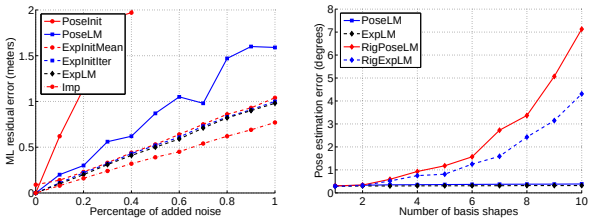
$$\| \underbrace{\begin{pmatrix}\mathrm{vect}^{\mathsf{T}}(\tilde{\mathsf{M}}_1)\\ \vdots \\ \mathrm{vect}^{\mathsf{T}}(\tilde{\mathsf{M}}_l)\end{pmatrix}}_{\Lambda} - \underbrace{\begin{pmatrix}\xi_1\\ \vdots \\ \xi_l\end{pmatrix}}_{\boldsymbol{\xi}} \underbrace{\mathrm{vect}^{\mathsf{T}}(\tilde{\mathsf{R}})}_{\tilde{\mathbf{r}}} \|^2,$$

which is a rank-1 approximation problem that we solve by 'truncating' the SVD $\Lambda = \mathsf{U}\Sigma\mathsf{V}^{\mathsf{T}}$, as in §IV-C: $\boldsymbol{\xi} = \psi_1(\mathsf{U}\Sigma)$ and $\tilde{\mathbf{r}} = \psi_1^{\mathsf{T}}(\mathsf{V})$. Note that $\|\tilde{\mathbf{r}}\| = \|\tilde{\mathsf{R}}\| = 1$. Matrix $\tilde{\mathsf{R}}$ must be subsequently corrected to give R by enforcing the orthonormality constraints. This is done by finding the closest orthonormal matrix to $\tilde{\mathsf{R}}$ using SVD, see [1]: $\tilde{\mathsf{R}} = \mathsf{U}\Sigma\mathsf{V}^{\mathsf{T}}$ gives $\mathsf{R} = \frac{1}{3}\mathrm{tr}(\Sigma)\det(\mathsf{U})\det(\mathsf{V})\mathsf{U}\mathsf{V}^{\mathsf{T}}$, while compensating the possible sign change by $\boldsymbol{\xi} \leftarrow \det(\mathsf{U})\det(\mathsf{V})\boldsymbol{\xi}$.

VI. EXPERIMENTAL EVALUATION

A. Simulated Data

We report experimental results on simulated data. The default simulation setup consists of $n = 15$ time-varying 3D views, each containing $m = 35$ points. They are generated by randomly drawn linear combinations of $l = 3$ basis shapes, all of them lying in a sphere with unit radius. An additive, zero-mean Gaussian noise with variance $\sigma = 0.01$ (*i.e.* 1% of the scene scale) is added to the 3D points. We vary each of these parameters in turn. We average the error measures over 100 trials. The true number of basis shapes is used by the algorithms.



(a) (b)

Fig. 1. (a) ML residual error against the level of added noise and (b) pose error against the number of basis shapes.

c) Environment learning: We observe on figure 1 (a) that the ML residual error is very close to σ. The implicit learning IMP consistently gives a significantly lower residual error than the explicit learning algorithms EXP⋆. This means that, despite the fact that the data were generated using the explicit low-rank shape model, the extra degrees of freedom of the implicit model represent quite well the added Gaussian noise.

We observe that the difference between the three explicit learning methods is small compared to the difference with IMP. EXPLM (from §IV-D.2, 'LM' stands for Levenberg-Marquardt) always performs better than EXPINITITER (from table I), which always performs better than EXPINITMEAN (based on [1] to get the rotations). This means that the residual error (8), which is minimized by EXPINITITER while estimating the minimal 3D non-rigid tensors, is well-adapted to our problem.

Figure 1 (b) compares the error raised by the rotation part of the pose, in degrees, between our non-rigid algorithms and rigid SFM and pose algorithms, respectively dubbed RIGEXPLM and RIGPOSELM. We observe that the proposed EXPLM gives errors independent of the number of basis shapes, while, as could have been expected, RIGEXPLM rapidly degrades as the number of basis shapes grows.

d) Pose computation: Figure 1 (a) shows that all pose algorithms POSE⋆ consistently give a higher residual error than the explicit learning algorithms. This is explained by the fact that pose estimation suffers from the errors in the learnt model *and* in the 3D view. POSEINIT gives quite high errors, roughly 5σ, while POSELM converges to roughly 1.5σ which is reasonable. The same remarks as for the learning algorithms can be made in the case of pose, for figure 1 (b).

Another experiment was intended to assess to which extent, reliable pose estimate can be obtained when the environment is deforming in a very different way compared to the learning stage. Let ν be the mean value of the configuration weights. We alter them by adding randomly drawn perturbations with increasing magnitude μ, and generate a 3D view with these parameters, from which pose is estimated. Obviously, the results depend on the simulation setup, the number of points, views, basis shapes and the level of noise. However, we

observe that for $\mu \leq 1.3\nu$, the residual error indicates that the pose estimate is reasonable for most configurations. For $\mu > 1.3\nu$, the pose estimate rapidly degrades.

B. Real Data

We tested our algorithms on sets of 3D points reconstructed from a calibrated stereo rig. The sequence consists of $n = 650$ pairs of views. The $m = 30$ point tracks were obtained semi-automatically and reconstruction was performed using ML triangulation, *i.e.* by minimizing the reprojection error. The reprojection error we obtained is 4.7276 pixels, which is rather large and explained by the low quality of the manually entered point tracks.

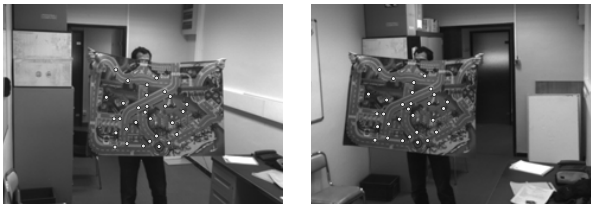
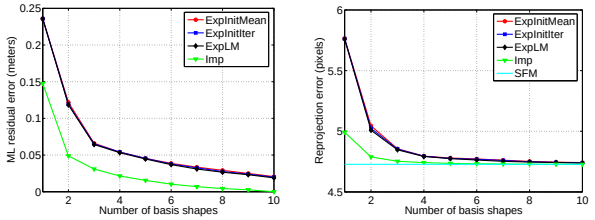


Fig. 2. One out of the 650 stereo pairs used in the experiments, overlaid with the 30 point tracks.

We used a subset of the full sequence, made of 1 3D view over 25 from 1 to 551, that is 23 3D views, for learning the environment. The remaining 3D views are registered by computing pose. For views $1 < i < 551$, this can be viewed as 'interpolation' since the surrounding 3D views are used for learning the environment, while for views $551 < i < 650$ this can be viewed as an 'extrapolation' of the model since new pose and deformations are seen in these views.

An important aspect is the choice of the number l of basis shapes. If l is too low, the model is not able to represent all the possible deformations, while if l is too high, the noise is modeled, resulting in unreliable pose estimates in both cases. We propose to manually choose l by examining the graphs shown on figure 3. It shows the ML residual errors and the reprojection errors, *i.e.* the Sum of Squared Differences between measured and predicted image points, resulting of the learning algorithms for different numbers of basis shapes. We



(a) (b)

Fig. 3. (a) ML residual error and (b) 2D reprojection error versus the number of basis shapes.

observe that the 3D ML residual error and the 2D reprojection error decrease while l increases, the former towards 0 and the latter towards the reconstruction error, shown by an horizontal line 'SFM' on the graph, which was expected. Based on this graph, we choose $l = 4$, for which the EXPLM ML residual error is 5.32 centimeters and the 2D reprojection error is 4.7922 pixels. For comparison, a rigid environment model gives a 23.58 centimeters ML residual error and a 5.7605 pixels 2D reprojection error. It is important to note that for $l = 5$ and $l = 6$ basis shapes, very similar pose estimates are subsequently obtained.

The learnt path appears visually satisfying, However, the mean difference in the rotations is 2.81 degrees, which is significant, but difficult to illustrate visually. The mean ML residual errors are 8.66 and 12.83 centimeters for the 'interpolated' and the 'extrapolated' poses respectively.

The computation time for the learning phase is of the order of a minute while pose estimation is roughly a tenth of a second.

VII. CONCLUSIONS

One weakness of the approach is to rely on 3D point correspondences. We are currently working on using more robust types of inputs, such as contours or image patches, that can be reliably tracked through sequences of stereo pairs using *e.g.* particule filtering techniques. This is intended to be part of an iterative deforming environment learning system. Essential issues that will be dealt with are assessing what kind of deformations can be represented by the low-rank shape model and choosing the number of basis shapes, which will be examined in the framework of model selection.

REFERENCES

[1] B. K. P. Horn, H. M. Hilden, and S. Negahdaripour, "Closed-form solution of absolute orientation using orthonormal matrices," *Journal of the Optical Society of America A*, vol. 5, no. 7, pp. 1127–1135, July 1988.

[2] C. Bregler, A. Hertzmann, and H. Biermann, "Recovering non-rigid 3D shape from image streams," in *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, 2000.

[3] M. Brand, "Morphable 3D models from video," in *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, 2001.

[4] M. Irani, "Multi-frame optical flow estimation using subspace constraints," in *Proceedings of the International Conference on Computer Vision*, 1999.

[5] A. D. Bue and L. Agapito, "Non-rigid 3D shape recovery using stereo factorization," in *Proceedings of the Asian Conference on Computer Vision*, 2004.

[6] J. Xiao, J.-X. Chai, and T. Kanade, "A closed-form solution to non-rigid shape and motion recovery," in *Proceedings of the European Conference on Computer Vision*, 2004.

[7] G. H. Golub and C. F. van Loan, *Matrix Computations*. Baltimore: The Johns Hopkins University Press, 1989.

[8] H. Aanæs and F. Kahl, "Estimation of deformable structure and motion," in *Proceedings of the Vision and Modelling of Dynamic Scenes Workshop*, 2002.

[9] L. Torresani and A. Hertzmann, "Automatic non-rigid 3D modeling from video," in *Proceedings of the European Conference on Computer Vision*, 2004.

[10] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon, "Bundle ajustment — a modern synthesis," in *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*, 2000.

7.2.3 Paper (BMVA Symposium'08) – *Automatic Quasi-Isometric Surface Recovery and Registration from 4D Range Data*

Automatic Quasi-Isometric Surface Recovery and Registration from 4D Range Data

Toby Collins
University of Edinburgh, UK

Dr. Adrien Bartoli
LASMEA (CNRS / UBP), Clermont-Fd, France

Prof. Robert Fisher
University of Edinburgh, UK

1 Introduction

To unlock the huge potential in video-rate range image analysis, arguably one the most significant hurdles to overcome is that of accurately recovering motion over a given sequence. For a single object, this largely amounts to uncovering the isomorphism undergone by the object's surface from a set of point sample observations in each frame. Unfortunately even with 3D data this is a highly challenging task, since in the absence of salient 3D structure (e.g. the region around a person's cheek or a section of cloth), motion cannot be resolved over the manifold (much like the aperture affect in 2D.) To alleviate this, we must impose assumptions about the object's deformation and/or appearance sub-space a priori (a concept well established in 2D image analysis). In 2D these priors must be extremely broad to account for the nonlinear, unknown and largely irrecoverable interaction between the object's deformed state and it's appearance in 2D. In general, the best we can do is to use extremely broad assumptions about smooth 2D motion, which has been demonstrated time again to be insufficient to adequately constrain the registration/tracking tasks.

By contrast, in our work we have focused on investigating general classes of deformation priors in 3D, where the story is very different. In 3D we can directly reason in terms of true 3D dynamics, and our assumptions on the underlying isomorphism can lead to very well defined registration problems. So well defined in fact that we show here how to go beyond mere registration (i.e. matching each frame to a particular template) to the harder problem of automatically recovering a complete, registered 3D model given only limited deformed observations. Solutions to this would find considerable application in the computer vision and graphics communities, where it is desirable to obtain object models that undergo non-rigid motion (e.g. a flying flag or a deforming body), undergoing occlusion (both self and external), never fully observable and present in cluttered scenes.

In this work our attention is focused on one important class of deforming objects; those whose surfaces undergo isometric or near-isometric deformations. These are a particular kind of isomorphism which relates a surface's embedding in 3D space (i.e. a Riemannian manifold) by a transformation preserving distances on the manifold (geodesics). Qausi-isometric transformations describe well the deformations undergone by a large range of real-world objects, such as many fabrics, paper, plastics, articulated objects and, to an extent, facial expressions. Robust methods for detecting and registering objects of this class therefore have uses in several fields including computer graphics, such as texture extraction, texture mapping, deformation transfer and video augmentation; computer vision, such as unsupervised deformable object learning, tracking, and deformation analysis.

2 Method Overview and Contribution

From a sequence of range observations our overarching goal is to automatically reconstruct the surfaces of deforming objects appearing in the sequence. This task is nontrivial for a number of reasons. Firstly, occlusion boundaries in range images, particularly those generated by stereo, cannot be reliably used for segmentation. Occlusion cues rely on depth discontinuities, which are often smoothed by the stereo algorithm, or not present if the occluding object is in contact with the surface. Without reliable segmentation, any priors on the isomorphism (including isometry or smoothness assumptions) are not valid (and thus detrimental) over regions which contain occlusion zones. The second problem that of global nonrigid alignment. Even if given a correct segmentation, matching pairs of range segments involves locally solving the non-rigid registration task. Without correspondences known a priori, this generally results in motion models with dense Jacobian and Hessian matrices. Scaling this up to a global alignment readily becomes intractable using numerical optimisation, yet this is highly desirable since local methods tend to result in global misalignments (as demonstrated in, for example 2D panorama stitching.)

However, we can obtain a foothold to the problem by considering certain deformation-invariant properties over the deforming surface. Here we present a global method for nonrigidly aligning surface segments located in range data that mutually agree with respect to the assumed isomorphism model. For near-isometry, this is the preservation of geodesic lengths on the manifold. We essentially perform surface model completion by embedded mosaicing, where a composite is formed in the surfaces intrinsic coordinate space. This has the desirable property that deformed segments are now relatable by far simpler transformations. For quasi-developable surfaces (i.e. isometric surfaces with very low Gaussian curvature such as cloth), isometry reduces to near-Euclidean transformations.

The key stages to our approach for automatic deformable surface reconstruction are shown in figure 1, and a brief overview of each stage is described in the following sections.

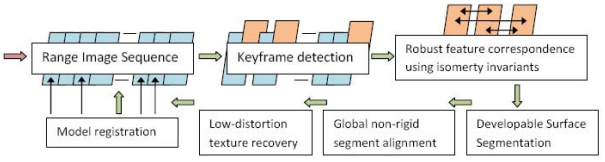


Figure 1: *Deformable model recovery and registration framework*

Since video rate range sequences contain a vast amount of redundant surface information, we would prefer to sample those frames which convey the most new surface information with which to build the model. Initially this set is selected using simple strategies (e.g. every n^{th} frame or random selection) from the range sequence, with further frame being added and included in the model based on (i) agreeing with the current model and (ii) revealing unseen sections of the surface.

2.1 Robust Feature Correspondence

Point correspondences are often key components in registration tasks largely for guiding the registration towards global optima. Typically, feature matchers in 2D (e.g. SIFT) and 3D (e.g. spin images) will misalign, so some form of correspondence annealing is often employed. We instead establish correspondence by appealing to the isometric assumption; that is mutual point distances on

Figure 2: *Deformable surface reconstruction from multiple range observations. TL: Regions matched and extracted from several range images (some of which are shown in TR. BL the resulting reconstruction comparing (left) the true surface texture, (middle) rigidly aligned patches and (right) the nonrigidly aligned and stitched model. BL: An example distortion map for a particular segment.*

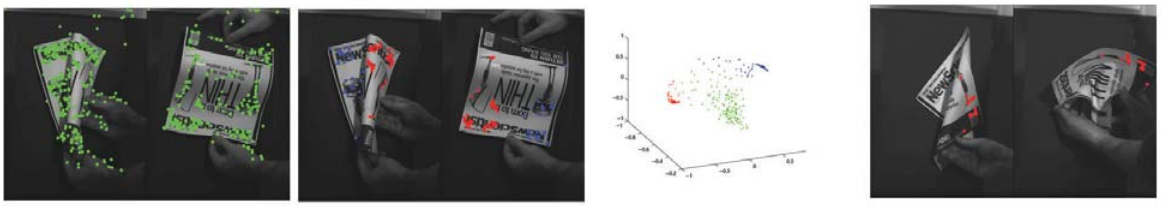


Figure 3: *Robust point correspondence between range pairs using spectral clustering. From left to right: Correspondences matched by SIFT; Robust correspondence and clustering using our method; Spectral embedding of matches (green indicates outliers); Another example.*

the manifold (geodesics) should be preserved. We establish a measure of isometric conformity between all pairs of candidate correspondences , and the optimal solution is the subset set of matches with maximal mutual agreement. In our method we find an efficient inexact solution to this using spectral methods to find strongly connected clusters in a compatibility graph whose nodes represent matches and edges weighted by their mutual match scores. Since occlusion zones (undetected or otherwise) break geodesic agreements, multiple clusters may exist in the graph which are each bounded by an occlusion zone. Thus, we use k-way spectral clustering to jointly recover the correct matches and segmentation (Figure 3.) Currently, we extract features based only on intensity, since 3D features on smooth developable surfaces are almost entirely ambiguous.

2.2 Quasi-Developable Surface Segmentation

Given a set of segmented high-quality feature correspondences, we proceed to extract the surrounding region agreeing with our assumption of developabliy. Although zero Gaussian curvature characterises perfectly developable surfaces, its computation is often too unstable to derrive a segmentation. Furthermore, for real surfaces low Gaussian curvature is usually present. Instead, from our matched features we grow a region over the range image such that the distortion (angular and stretch) induced by flattening it is bounded by some tolerance. Once grown, we further refine the region using other segmentation cues (i.e. strong intensity/depth gradients and colour histograms), combined proabablistically and solved using graph cuts.

2.3 Global Nonrigid Segment Alignment

Given multiple surface segments, from multiple range images, we bring them into nonrigid alignment by minimising two error criteria related to (i) mutual feature distances between matched segments and (ii) mutual feature distances over each segment. The first enforces the alignment of matched features whilst the second preserves the system's rigidity. Since (ii) is very nonlinear (quadric in position), we settle for a slightly weaker interpretation which enforces conformality rather than rigidity over a segment (i.e. angle preservation.) This is quadratic in position and results in a full-rank least squares system that can be solved in closed form using sparse linear least squares. The second stage propagates the transformations from the features through each segment. Since our underlying assumption is of near-rigidity on the 2D plane, we use an as-rigid-as-possible transformation similar to that proposed recently by [Schaefer et al. 2006] used in interactive shape manipulation.

2.4 Low-Distortion Surface Texture Recovery

Our final stage of recovering the surface model is to generate a rendering of the surface's texture. Given the match correspondences, shading artifacts can be removed using standard techniques. To generate the render, blending techniques used in image mosaicing

are not suitable, due to residual ghosting from small misalignments. Instead we are driven by the goal of a low-distortion rendering. In our method we greedily select and stitch those regions which were were transformed onto the plane with the least distortion. Given an initial patch (i.e. the one least distorted), we treat the addition of a second patch as a binary labeling problem, combining a distortion penalty with a seam cost, and solve this incrementally using graph cuts.

2.5 Registration

In order to fit the model to the rest of the data, we adopt an energy minimisation process which penalises states that diagree in data evidence (i.e. 3D distance to the point cloud and feature correspondence), and a quadratic isometric within-plane bending model [Bergou et al. 2006]. For all other frames, we perform the same strategy but initialise the model to its deformed state in the previous frame. Global drift is avoided by using hard feature constraints between the model and scan as described in section 2.1. The newly registered instances can then be further incorporated into our model, along with new neighbouring regions agree with respect to the model's isomorphism prior.

3 Results

We have tested our approach on several sequences capturing deforming paper and fabrics. Figures 2 and 3 show examples of some good results attained on a seqeunce of a deforming magazine cover. In the near future we aim to further validate our methods for other sequences. We also aim to extend our work by better modelling the relationship between mesh distortion and nonrigid deformation constraints on the 2D plane, and will be investigating our framework for recovering other classes of deforming surfaces.

References

BERGOU, M., WARDETZKY, M., HARMON, D., ZORIN, D., AND GRINSPUN, E. 2006. A quadratic bending model for inextensible surfaces. In *SGP '06: Proceedings of the fourth Eurographics symposium on Geometry processing*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 227–230.

SCHAEFER, S., MCPHAIL, T., AND WARREN, J. 2006. Image deformation using moving least squares. In *International Conference on Computer Graphics and Interactive Techniques*.

7.2.4 Paper (3DIM'07) – *Joint Reconstruction and Registration of a Deformable Planar Surface Observed by a 3D Sensor*

Joint Reconstruction and Registration of a Deformable Planar Surface Observed by a 3D Sensor

Umberto Castellani[1], Vincent Gay-Bellile[2,3] and Adrien Bartoli[2]
[1] VIPS, University of Verona, Italy
[2] LASMEA, Clermont-Ferrand, France
[3] CEA-LIST, Saclay, France

Abstract

We address the problem of reconstruction and registration of a deforming 3D surface observed by some 3D sensor giving a cloud of 3D points at each time instant. This problem is difficult since the basic data term does not provide enough constraints.

We bring two main contributions. First, we examine a set of data and penalty terms that make the problem well-posed. The most important terms we introduce are the non-extensibility penalty and the attraction to boundary shape. Second, we show how the error function combining all these terms can be efficiently minimized with the Levenberg-Marquardt algorithm and sparse matrices.

We report convincing results for challenging datasets coming from different kinds of 3D sensors. The algorithm is robust to missing and erroneous data points, and to spurious boundary detection.

1 Introduction

Recent advances in real-time 3D scanners [12, 18] have led to fast model acquisition and to 3D Human Computer Interaction systems [5]. The basic task is the registration of several 3D point clouds. Most of the work is devoted to rigid scenes e.g., [19]. The case of deformable objects has been addressed since about a decade [7, 2, 13, 23]. Research on deformable models is active in fields such as computer graphics for 3D morphing and animation [1], medical image processing for data alignment and segmentation [9, 22], and computer vision for e.g. contour detection [11], face synthesis and expression recognition [3]. Commonly used tools are the Thin-Plate Splines (TPS) [7] and Principal Component Analysis [3].

In this paper, the aim is to capture the possibly complex deformations of a smoothly deforming object with planar topology (such as the page of a book being turned by someone). We target applications such as 3D data compression and augmented reality, requiring an accurate registration of the point clouds over time, as well as a reconstruction of the underlying surface. For instance, videos can be synthesized using the captured deformations. We assume as input data point clouds and a coarse boundary of the surface of interest.

We propose a novel approach. The whole process is highly robust, filling in possible holes in the point clouds and detecting erroneous points, while establishing reliable point correspondences even for flat regions.

We use a generic and flexible deformable model represented by a grid mesh such as the ones in [14, 16] for 2D image registration. The problem in 3D is more challenging since it lacks reliable features for correspondence computation. No pre-established correspondences between grid points and data points are given, and no texture information is available. The problem is thus strongly ill-posed[1]. Nevertheless, the 3D domain is more robust to some lighting variations and the ambiguity due to projection needs not be taken into account. In this sense our approach has similarities with methods for non-rigid 3D point registration [7].

Our joint reconstruction and registration framework is implemented through two main lines of contributions. First, we show that the problem is well-modelled by using a mesh that is deformed to fit each point cloud. This model allows us to write an error function which global minimum is the sought after solution. This error function has several data and penalty terms. The data terms incorporate the boundary information in a robust manner. It explicitly embeds a $\min$ operator, thus avoiding the traditional two steps in ICP-like algorithms through distance transform. The penalty terms include spatial, i.e. surface-related, and temporal smoothness as well as inextensibility of the surface, if applicable. The data terms are robustified in order to deal with missing and erroneous points.

[1] Note that in the 2D domain, a template image is available, allowing reliable feature matching [14]. This is not the case for the 3D problem we tackle.

Second, following [8], we use the Levenberg-Marquardt algorithm to minimize the error function. A careful analysis reveals that the Jacobian matrix involved in the normal equations to be solved at each iteration is highly sparse, for all the data and penalty terms we use. This makes tractable and fast the estimation of dense deformation fields.

Roadmap. Section 2 describes the state-of-the-art. The problem modelling is given in Section 3, and the strategy for finding the optimal solution is described in Section 4. Exhaustive experimental results are reported in Section 5. Finally, conclusions are drawn in Section 6.

2 Previous Work

The registration of 3D point clouds is a challenging topic mainly tackled in the framework of ICP [8, 19] for rigid scenarios. However, research has recently addressed the case of deformable objects, onto which we focus our state-of-the-art. Roughly speaking the literature on non-rigid registration can be divided into two main categories. The first one directly uses the point clouds. The second one abstracts the point clouds with some probabilistic models.

Point-based approaches. In [7] the authors propose to jointly compute the correspondences and the non-rigid transformation parameters between two point clouds. The algorithm is inspired by the Expectation-Maximization (EM) paradigm. It combines the soft-assign and deterministic annealing within a robust framework. TPS are used for representing the spatial mapping. Non-rigid alignment is proposed in [4] to account for errors in the point clouds, obtained by scanning a rigid object. The authors use TPS to represent the non-rigid warp between a pair of views, that they estimate through hierarchical ICP [19]. Medical applications are proposed in [9, 22]. In [22], MR brain scan registration is performed by a modified Newton method over a hierarchical spline-based optical flow representation. In [9], a localized Radial Basis Function (RBF) is proposed, making a point to depend only on its neighboring centers. Other approaches are introduced for cloth motion capture [17, 24] by using both intensity and geometry information. In [17] features points are matched by adopting a novel seed-and-grow approach to adapt the feature extraction to deformable geometry. In [24], a direct estimation of the deformable motion parameters is proposed for range-image sequences. The range flow is estimated by introducing depth constraint, to motion.

Probabilistic approaches. Probabilistic approaches [2, 13, 23] are based on modelling each of the point sets by a kernel density function [21]. The (dis)similarity among such densities is computed by introducing appropriate distance functions. Registration is carried out without explicitly establishing correspondences. In [2], the authors propose a correlation-based approach [21] to point set registration by representing the point sets as Gaussian Mixture Models (GMMs). A closed-form solution for the L_2 norm distance between two Gaussian mixtures makes fast computation possible. In [23], registration is carried out simultaneously for several 3D range datasets. The method proposes an information-theoretic approach based on the Jensen-Shannon divergence measure. In [13], non-rigid registration is treated as a Maximum Likelihood (ML) estimation problem by introducing the Coherent Point Drift (CPD) paradigm. Smoothness constraints are introduced based on the assumption that points close to one another tend to move coherently over the velocity field. The proposed energy function is minimized with the EM algorithm.

The proposed approach. In contrast to previous work, we do not attempt to directly register pairs of point clouds. We rather process each point cloud independently. For each, we jointly reconstruct the surface and register it to some generic deformable model. This naturally gives the registration of multiple point clouds.

3 Modelling the Problem

3.1 Surface Representation

A deformable surface with planar topology is observed. The 3D sensor provides a sequence of 3D point clouds D_i:

$$D_i = \begin{pmatrix} d^x_{i,1} & d^y_{i,1} & d^z_{i,1} \\ \vdots & \vdots & \vdots \\ d^x_{i,l_i} & d^y_{i,l_i} & d^z_{i,l_i} \end{pmatrix}.$$

The reconstructed surface at time i is represented by *geometry images*. The model M is organized as three $R \times C$ matrices, representing the deformation of a regular flat grid. Each matrix is reshaped in a single vector of size $\mu = RC$, giving M_i as:

$$M_i = \begin{pmatrix} m^x_{i,1} & m^y_{i,1} & m^z_{i,1} \\ \vdots & \vdots & \vdots \\ m^x_{i,\mu} & m^y_{i,\mu} & m^z_{i,\mu} \end{pmatrix}.$$

In practice, the number of data points is much larger than the number of model points, i.e. $l_i \gg \mu$. Upon convergence, our algorithm determines for each model point if there is a corresponding point in the current point cloud. Points may be missing because of occlusions or bad sensor output. This approach has the advantage that it naturally gives the reconstructed surface by interpolating the mesh points. Point cloud registration is obtained by composing the deformation fields.

3.2 Error Function

Our error function combines two data and three penalty terms:

$$e(M) = e_g(M) + \lambda_b e_b(M) + \lambda_s e_s(M) + \lambda_x e_x(M) + \lambda_t e_t(M), \quad (1)$$

where λ_b, λ_s λ_x and λ_t are weighting parameters.

The data terms are used to attract the estimated surface to the actual point cloud. The first term e_g is for global attraction, while the second one e_b deals with the boundary. These terms must account for possible erroneous points through robust statistics. The penalty terms are a *smoothness* constraint e_s, a *non-extensibility* constraint e_x and a *temporal* smoothness constraint e_t.

Data term: global attraction. This term globally attracts the model to the data points in a closest point manner. In order to avoid the traditional two steps arising in ICP-like algorithms, we explicitly embed the $\min$ operator in this data term, as suggested in [8]. Denoting E_M and E_D the sets of boundary points in the model and in the data, we get:

$$\sum_{m \in M \setminus E_M} \min_{d \in D \setminus E_D} \| d - m \|^2, \quad (2)$$

where d and m are $3-$vectors representing a data and a model point respectively. An *outliers rejection* strategy is introduced by defining a robust function $w(\cdot)$. Following the $X84$ rule [6], the function $w(\cdot)$ discards (i.e., it puts their residual to zero) those correspondences which residual error differs by more than 5.2 MAD (Median Absolute Deviation) from the median. The value 5.2 corresponds to about 3.5 standard deviations, which includes more than 99.9% of a gaussian distribution. Therefore, (2) is modified as:

$$e_g(M) = \sum_{m \in M \setminus E_M} w\left(\min_{d \in D \setminus E_D} \| d - m \|^2 \right). \quad (3)$$

Data term: boundary attraction. This term attracts boundary model points to boundary data points. It is defined in a similar manner to the global attraction term (3):

$$e_b(M) = \sum_{m \in E_M} w\left(\min_{d \in E_D} \| d - m \|^2 \right). \quad (4)$$

Penalty term: spatial smoothness. This term discourages surface discontinuities by penalizing the second derivatives. According to the geometry image definition, the model M is a displacement field parameterized[2] by (u, v) with $u = 1 \dots R$ and $v = 1 \dots C$, i.e., $M(u, v) = [M^x(u, v), M^y(u, v), M^z(u, v)]$. The spatial smoothness term is the bending or TPS energy function:

$$e_s(M) = \int_{\mathcal{R}} \int_{\mathcal{R}} \left(\frac{\partial M}{\partial^2 u} \right)^2 + 2 \left(\frac{\partial M}{\partial u \partial v} \right)^2 + \left(\frac{\partial M}{\partial^2 v} \right)^2 du dv.$$

Using a finite difference approximation for the first and second derivatives [16], the bending energy can be expressed in discrete form as a quadratic function of M:

$$e_s(M) = vect(M)^\top \mathcal{K} vect(M), \quad (5)$$

where $\mathcal{K}$ is a $3\mu \times 3\mu$ matrix, and $vect(M)$ is the vectorization operator which rearranges matrix M to a vector.

Penalty term: non-extensibility. This term discourages surface stretching. It favors the mesh vertices to preserve their distance with their local neighborhood [20]:

$$e_X(M) = \sum_{m \in M} \sum_{k \in \mathcal{N}(m)} \left(\| m - k \|^2 - L_{m,k}^2 \right)^2, \quad (6)$$

where $L_{m,k}$ are constants which are computed at the first frame after robust initialization and $\mathcal{N}(m)$ is the neighborhood of the mesh vertex m, with $\#\mathcal{N}(m) = 8$.

Penalty term: temporal smoothness. This defines a dependency between the current and the previous point clouds, M and $\tilde{M}$:

$$e_t(M) = \| M - \tilde{M} \|^2 . \quad (7)$$

This is intended to sequential processing and is thus not used on the first frame of the sequence.

4 Estimating the Solution

In order to minimize the error function (1), we use a nonlinear optimization algorithm, namely the Levenberg-Marquardt (LM) algorithm, since the error function in (1) is a sum of nonlinear squared terms. We extend the LM-ICP approach proposed in [8] to deformable objects. LM requires one to provide the partial derivatives of the error terms through a Jacobian matrix.

Since the Hessian-matrix $H = J^\top J$ must be inverted at each LM iteration, the problem is not tractable if the number of model points is too high (if the deformation field is too dense). In more details, with our formulation, the Jacobian matrix is:

$$J^\top = \begin{pmatrix} J_d^\top & J_b^\top & J_s^\top & J_x^\top & J_t^\top \end{pmatrix}.$$

where $J_d^{\mu \times 3\mu}$, $J_b^{E_B \times 3\mu}$, $J_s^{3\mu \times 3\mu}$, $J_x^{\xi \times 3\mu}$, $J_t^{\mu \times 3\mu}$, are related to the global attraction, boundary attraction, spatial smoothness, non-extensibility and temporal smoothness terms respectively, and $\xi = \#\mathcal{N}(M)$. For instance, a grid with

[2] Remember that the model points lie on a grid.

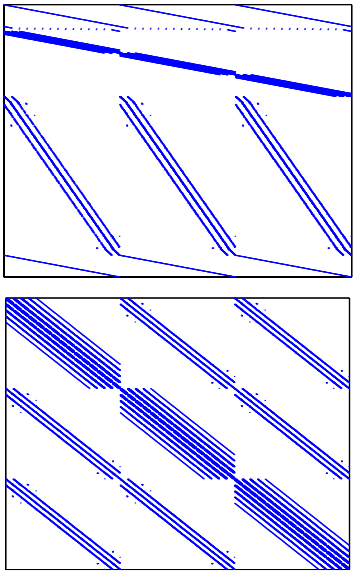


Figure 1. Jacobian (top) and Hessian (bottom) matrices: dark means non-zero.

15×20 points has a Jacobian matrix with $3760 \cdot 900$ elements ($\mu = 300$, $E_B = 66$, $\xi = 2194$). One advantage of the proposed approach is that the Jacobian matrix J is very sparse. We thus use the sparsity to solve the problem [15]. Figure 1 shows a plotting of the Jacobian matrix $J^{3760\times900}$ (top), and the corresponding Hessian matrix H (bottom). Dark points are associated to non zero entries. The sparseness of the Jacobian and the Hessian matrices is clearly evidenced.

5 Experiments

Two kinds of experiments have been set up. In the first one, a structured-light 3D scanner[3] is used for single image acquisition. A sheet of paper is observed as the deformable surface. In the second one, the sensor is a passive-stereo system[4] which allows us to acquire a sequence of 3D point clouds in real-time. The deformation of a portion of a cover is modelled.

Initial conditions determine an estimation for both the model position, and the grid size. In practice, a correct starting grid allows LM to converge, as well as to determine the parameters $L_{m,k}$ in (6). The boundary in the data points are necessary for using the boundary constraint [5].

[3]Courtesy of Purdue University (http://web.ics.purdue.edu).

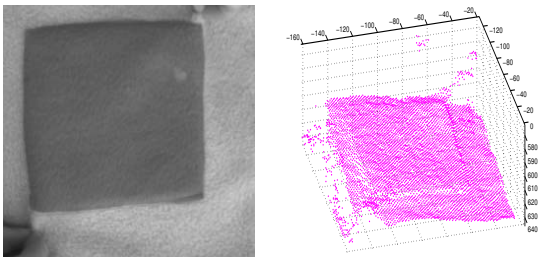
[4]Courtesy of eVS (http://www.evsys.net).

[5]We do not give details on boundary detection since usually those are known in advance, or can be extracted from the input data, e.g. by detecting depth discontinuities in the range image.

Experiment 1: paper sheet from a structured-light scanner. Several acquisitions have been carried out while bending the paper. The sensor provides accurate and high-resolution 3D point clouds. The initial orientation of the grid is estimated by fitting a plane to the data. By projecting the points to the plane, both the grid size and boundaries are easily computed. There is not a temporal dependency between the views. The temporal constraint is thus inhibited.

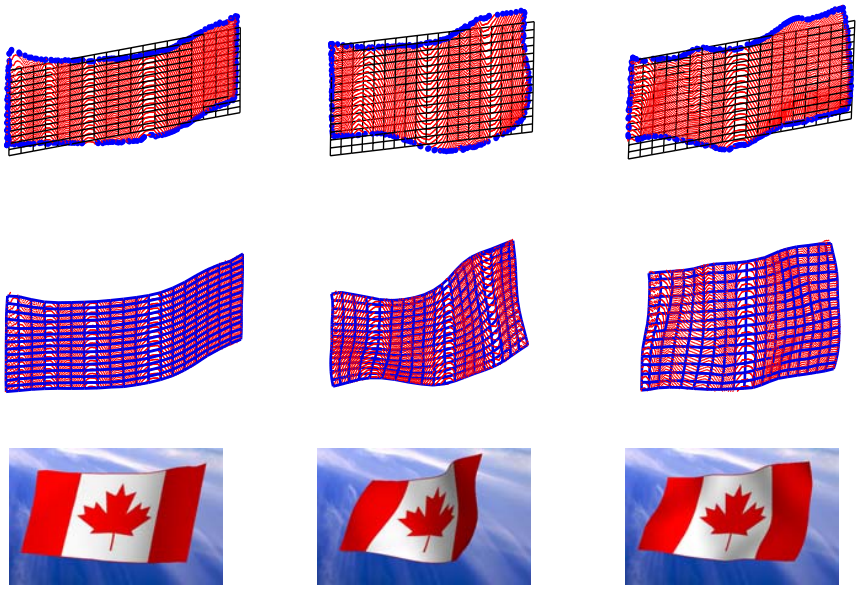
Figure 2 shows three examples. Images on the top row visualize the model and data before the fitting. Boundary points are highlighted. In the first example (Figure 2.a), the deformation is mainly on the horizontal boundary. In the second one (Figure 2.b), the paper is bent from the top-right to the bottom-left corners. In the third one (Figure 2.c), the deformation is basically spread to the whole paper. Images on the central row show the result of our robust fitting. Registration is accurate for both the interior points and the boundary. Moreover, the grids are smooth as expected. Finally, three synthetic reconstructions are shown on the bottom row. Any texture can be projected to the model, for realistic simulation of paper deformation at arbitrary points of view.

Experiment 2: cover from a stereo system. A long sequence of point clouds is acquired for the second experiment. The sensor acquires the images at 25 fps, and provides both intensity (i.e., 2D) and 3D information. The quality of the 2D images is low, and the 3D data is noisy. Moreover, the sensor can operate only on a very limited field of view (i.e., $30cm^3$). We use a cover as target object. Figure 3.a shows a picture of the cover. We aim at observing the cover deformation only on the portion delimited by the dark square. Figure 3.b shows the 3D point cloud. There are many spurious points especially on the boundaries, and the scene is not easily recognizable. We use the intensity im-



(a) (b)

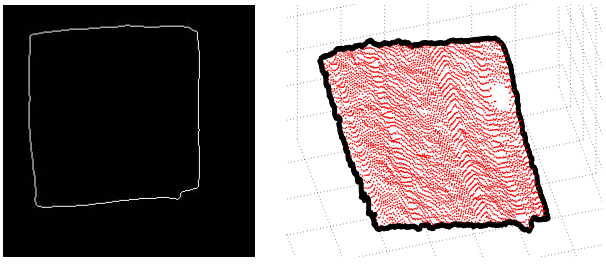
Figure 3. The cover sequence: intensity image of the cover (a) and the 3D point cloud (b).



(a) (b) (c)

Figure 2. The paper sequence. Three examples of point clouds with the grid mesh superimposed at the starting position (top) and after model fitting (center). Boundary points are evidenced. The reconstructed model is shown with a new texture (bottom).

age[6] for selecting automatically our region of interest (i.e., the dark square), from which we recovered both the 3D data and the boundary. Figure 4.a shows the image-boundary extracted by standard image processing techniques, while Figure 4.b depicts the 3D data (i.e., the selected point cloud and 3D boundary). The sequence is made of 100 point clouds.



(a) (b)

Figure 4. Data extraction: 2D boundary (a) and selected 3D data (b). 3D boundary are highlighted with dark color.

Model initialization is carried out for the first cloud only. Each iteration uses the output of the previous one as an initial condition. Figure 5 shows a selection of the output sequence. For each frame, is visualized: 1) the intensity image, with the extracted 2D boundary and the 2D projection of the estimated model, and 2) the point cloud - after the region of interest selection -, evidencing both the 3D boundary and the grid. The cover is handled from the bottom-left and upper-right corners, respectively. On the early frames, the cover is gradually bent toward the square center, then it is strongly taut, moving the corners far from each other. Finally, in the late frames, random deformations are generated especially around the corners. Some frames are particularly challenging. In frame (c) a strong shrinking is evidenced on the top-right corner. In frame (f) a wide hole appears on top-right side. In frames (h) and (i) data boundaries are clearly wrong on the bottom-left side. Results are satisfying since the fitting is correct for the whole sequence. The mesh grids are well superimposed on data points maintaining the shape smooth. Nevertheless, the projections of the grids to the 2D images confirm the accuracy of the registration. Finally, after joint reconstruction and registration, a dense set of accurate deformable models is available. We used them to synthesize a video as if it was projected to a deforming screen. We take a video and project every video-frame to a

[6]The intensity is the left image of the stereo-pair, which is associated to the disparity map. Indeed, there is a mapping between the 2D and 3D information. Note that we do not use intensity information for fitting.

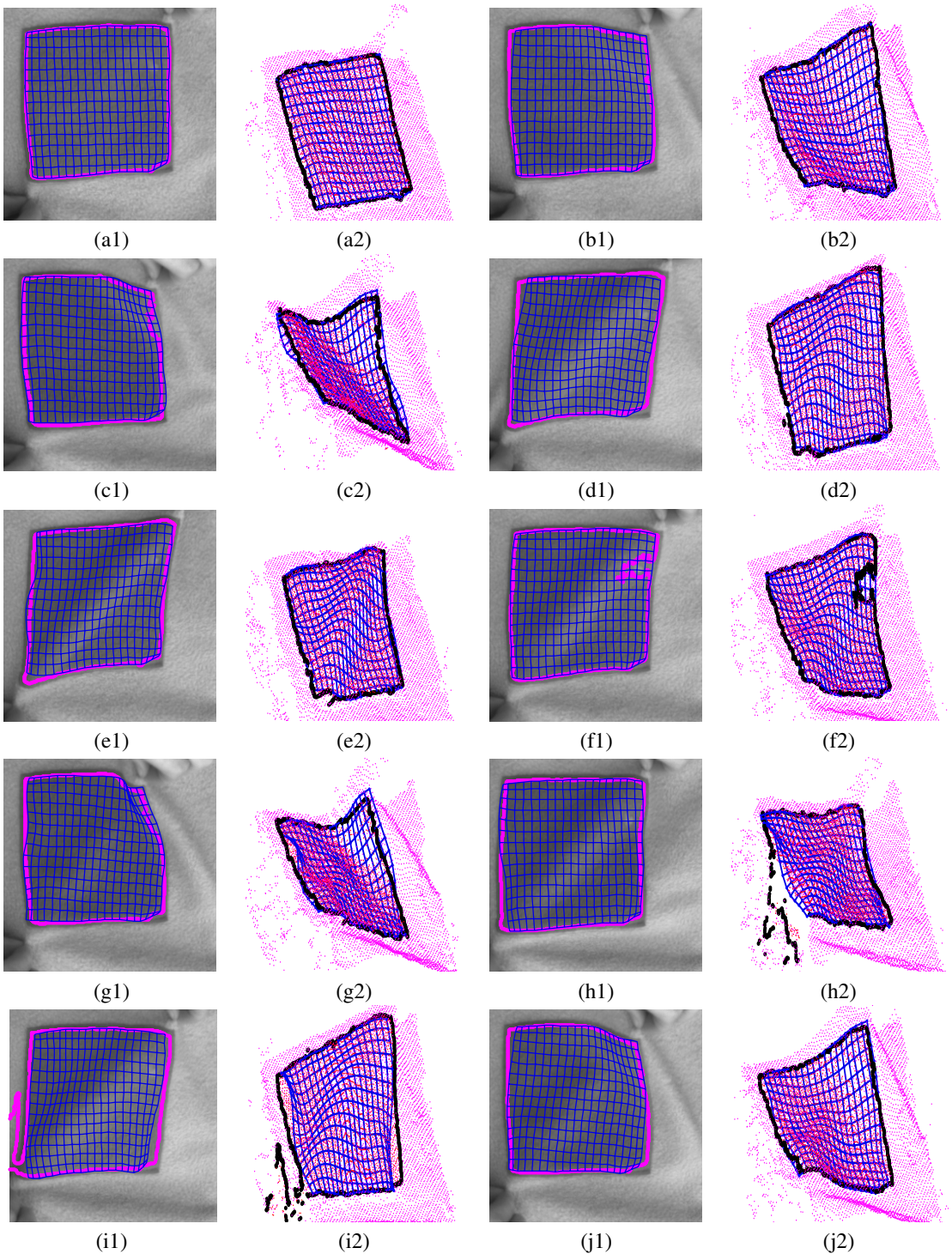


Figure 5. Cover sequence: 10 **selected frames. For each frame the 2D intensity (·,1) and the 3D data (·,2) is visualized. The grid models are shown in the 3D space as well as their projection in the 2D image.**

model of our sequence[7]. Some frames are shown in Figure 6.

For both experiments, a model of size 15×20 is used. We have verified that a higher value of λ_b is necessary (i.e., $\lambda_b = 1.5$) for a correct convergence of the algorithm to the optimal solution. The other terms are set almost equally to 1. The distance transform parameters are important: the size of the voxels trades off speed and result accuracy. The method has been implemented in Matlab, and takes around 30 seconds per frame[8]. Note that the computational cost of one LM iteration depends only on the chosen grid size, being independent on the amount of input data.

6 Conclusions

We propose a new approach for capturing the deformation of 3D surfaces from 3D scans. Reconstruction and registration are jointly performed into a fitting-based framework. We deform as model a generic geometric image, which is aligned with the observed data. An error function is designed to combine the influence of a priori information, such as spatiotemporal smoothness, and observations. Both the non-extensibility and boundary attraction terms are crucial for disambiguating this intrinsically ill-posed problem. The optimization phase os solved with the Levenberg-Marquardt algorithm, while taking advantage of the sparsity of the Jacobian and Hessian matrices.

Results are promising since the performances are satisfying for the analyzed cases. The method has been tested onto two kinds of datasets by evidencing the versatility in dealing with different sensors. In the first experiment, the source data was accurate and the estimated models was according to what we expected. In the second experiment, a whole sequence of 3D point clouds has been processed. This has allowed us to observe real-time deformations. Although data was very noisy, especially around the boundary, the method performed robustly. We have discussed also the behavior of our algorithm in the presence of holes and broken boundary. Finally, some graphical results have been shown for simulated deformations of a paper-sheet by changing its original appearance, as well as for synthesizing a video.

Acknowledgments

We would like to thank Johnny Park from Purdue University for providing us 3D scans of the paper sheet, and eVS (http://www.evsys.net) for the cover sequence.

References

[1] M. Alexa. Recent advances in mesh morhping. *Computer Graphics Forum*, 21(2):173–196, 2002.

[2] B.Jian and B. Vemuri. A robust algorithm for point set registration using mixture of gaussians. In *CVPR*, 2005.

[3] V. Blanz and T. Vetter. A morphable model for the synthesis of 3d faces. In *SIGGRAPH*, 1999.

[4] B. Brown and S. Rusinkiewicz. Non-rigid range-scan alignment using thin-plate splines. In *3DPVT*, Sept. 2004.

[5] A. Cassinelli, S. Perrin, and M. Ishikawa. Smart laser-scanner for 3D human-machine interface. In *Extended abstracts on Human factors in computing systems*, 2005.

[6] U. Castellani, A. Fusiello, and V. Murino. Registration of multiple acoustic range views for underwater scene reconstruction. *Computer Vision and Image Understanding*, 87(3):78–89, July 2002.

[7] H. Chui and A. Rangarajan. A new point matching algorithm for non-rigid registration. *Computer Vision and Image Understanding*, 89(2-3):114–141, 2003.

[8] A. Fitzgibbon. Robust registration of 2D and 3D point sets. *Image Vision Computing*, 21(13-14):1145–1153, 2003.

[9] M. Fornefett, K. Rohr, and H. Stiehl. Radial basis functions with compact support for elastic registration of medical images. *Image and Vision Computing*, 19:87–96, 2001.

[10] X. Gu, S. Gortler, and H. Hoppe. Geometry images. In *SIGGRAPH*, 2002.

[11] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1987.

[12] T. Koninckx, A. Griesser, , and L. V. Gool. Real-time range scanning of deformable surfaces by adaptively coded structured light. In *3DIM*, 2003.

[13] A. Myronenko, X. Song, and M. Carreira-Perpinan. Non-rigid point set registration: Coherent point drift. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *NIPS*. MIT Press, Cambridge, MA, 2007.

[14] J. Pilet, V. Lepetit, and P. Fua. Real-time non-rigid surface detection. In *CVPR, San Diego, CA*, June 2005.

[15] S. Pissanetzky. *Sparse Matrix Technology*. Academic Press, 1984.

[16] M. Prasad, A. Zisserman, and A. Fitzgibbon. Single view reconstruction of curved surfaces. In *CVPR, New York*, 2006.

[17] D. Pritchard and W. Heidrich. Cloth motion capture. In *Eurographics*, 2003.

[18] S. Rusinkiewicz, O. Hall-Holt, and M. Levoy. Real-time 3D model acquisition. In *SIGGRAPH*.

[19] S. Rusinkiewicz and M. Levoy. Efficient variants of the ICP algorithm. In *3DIM*, Quebec City (Canada), 2001.

[20] M. Salzmann, S. Ilic, and P. Fua. Physically valid shape parameterization for monocular 3–D deformable surface tracking. In *BMVC*, 2005.

[21] Y. Tsin and T. Kanade. A correlation-based approach to robust point set registration. In *ECCV*, 2004.

[22] B. Vemuri, S. Huang, S. Sahni, C. Leonard, C. Mohr, R. Gilmore, and J. Fitzsimmons. An efficient motion estimator with application to medical image registration. *Medical Image Analysis*, 2(1):79–98, 1998.

[23] F. Wang, B. Vemuri, and A. Rangarajan. Groupwise point pattern registration using a novel CDF-based Jensen-Shannon Divergence. In *CVPR*, 2006.

[24] M. Yamamoto, P. Boulanger, J. Beraldin, M. Rioux, and J. Domey. Direct estimation of deformable motion parameters from range image sequence. In *CVPR*, Osaka, Japan, 1990.

[7] We loop over the extracted 100 models.

[8] Experiments are carried out on a Pentium M 1.86GHz.

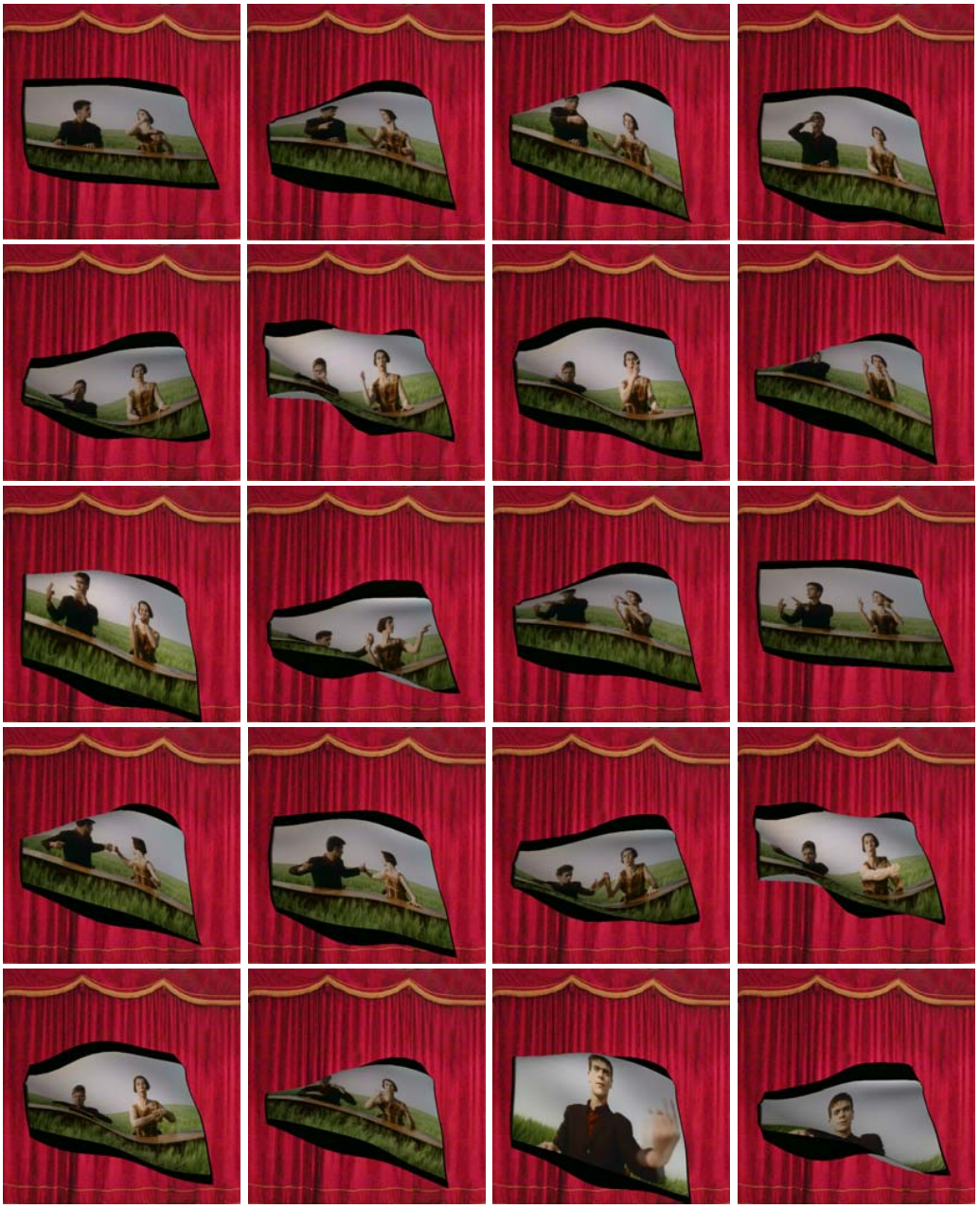


Figure 6. Synthesized movie: some selected frames. Each frame of the movie is projected to the reconstructed model by simulating a deforming video screen.

CHAPTER

8

STRUCTURE-FROM-MOTION FOR RIGID SCENES

8.1 Structure-from-Motion with Points

I33 **Algorithms for Batch Matrix Factorization with Application to Structure-from-Motion** §8.1.1
J.-P. Tardif, A. Bartoli, M. Trudeau, N. Guilbert and S. Roy
CVPR'07 - IEEE *Int'l Conf. on Computer Vision and Pattern Recognition*, Minneapolis, USA, June 2007

I32 **On Constant Focal Length Self-Calibration From Multiple Views** §8.1.2
B. Bocquillon, A. Bartoli, P. Gurdjos and A. Crouzil
CVPR'07 - IEEE *Int'l Conf. on Computer Vision and Pattern Recognition*, Minneapolis, USA, June 2007
Version in French: [N10]

I23 **Handling Missing Data in the Computation of 3D Affine Transformations** §8.1.3
H. Martinsson, A. Bartoli, F. Gaspard and J.-M. Lavest
EMMCVPR'05 - IAPR *Int'l Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, St. Augustine, Florida, USA, p. 90-106, November 2005
Version in French: [N06]
Previous version: [I20]

8.1.1 Paper (CVPR'07) – *Algorithms for Batch Matrix Factorization with Application to Structure-from-Motion*

Algorithms for Batch Matrix Factorization with Application to Structure-from-Motion

Jean-Philippe Tardif* Adrien Bartoli† Martin Trudeau*
Nicolas Guilbert‡ Sébastien Roy*

Abstract

Matrix factorization is a key component for solving several computer vision problems. It is particularly challenging in the presence of missing or erroneous data, which often arise in Structure-from-Motion. We propose batch algorithms for matrix factorization. They are based on closure *and* basis *constraints, that are used either on the cameras or the structure, leading to four possible algorithms. The constraints are robustly computed from complete measurement sub-matrices with e.g. random data sampling. The cameras and 3D structure are then recovered through Linear Least Squares. Prior information about the scene such as identical camera positions or orientations, smooth camera trajectory, known 3D points and coplanarity of some 3D points can be directly incorporated. We demonstrate our algorithms on challenging image sequences with tracking error and more than 95% missing data.*

1. Introduction

Matrix factorization is an essential tool for solving several computer vision problems including Structure-from-Motion [20, 23], plane-based pose estimation [21], non rigid 3D reconstruction [6] and motion segmentation [25]. When no data are missing or corrupted by outliers, an efficient algorithm based on Singular Value Decomposition (SVD) can be used. However, missing and erroneous data are certainly unavoidable in many real-life situations, and they make factorization much more difficult. Furthermore, the SVD-based algorithm makes it difficult to enforce constraints specific to the formulation or provided by prior information about the problem.

We propose algorithms for batch matrix factorization with special attention given to the Structure-from-Motion (SfM) problem. *Closure* or *basis constraints* on one of the two factors are computed from complete measurement sub-matrices. For example, *Camera Closure Constraints* (CCC) can be computed from their left kernel [24]. We investigate three variations: *Camera Basis Constraints* (CBC), *Structure Closure Constraints* (SCC) and *Structure Basis Constraints* (SBC), that are respectively left basis, right kernel and right basis of the measurement matrix. Our experiments and comparison with other state-of-the-art batch factorization algorithms show that basis constraints usually give better results than closures, for both affine and perspective camera models. We also show that structure constraints can be used first to compute the 3D structure instead of the cameras, which allows to enforce directly constraints such as known 3D points or known planar surface. In the case of affine SBC, an approximation to the reprojection error is minimized.

Organization. The next section reviews previous work on factorization and Structure-from-Motion and points out the main differences with ours. In §3, we give our notation and formally introduce the factorization problem with his specialization to the affine Structure-from-Motion problem. Camera Closure Constraints (CCC) are first reviewed in §3.1, then follows our contribution in §4. The details are provided for the affine camera model and we discuss how the theory applies to the perspective model in §5. Robustness is discussed in §6, experiments are described and analyzed in §7, followed by conclusion in §8.

2. Previous Work

Structure-from-Motion can be formulated most generally as a bilinear (modulo homogeneous scale) inverse problem. The seminal work on affine factorization [23] however showed that it could be relaxed to a bilinear problem for the affine camera model. For perspective cameras, it can be formulated similarly when the homogeneous feature point coordinates are rescaled by their projective depth, which must be estimated *a priori* [14, 20].

The algorithms for matrix factorization despite missing data can be divided into three main categories: iterative, batch and hierarchical. In the first one, factorization is performed by minimizing directly the factorization error either

*Université de Montréal, Canada {tardifj,trudeaum,roys}@iro.umontreal.ca
†CNRS - LASMEA, France, adrien.bartoli@univ-bpclermont.fr
‡Lund University, Sweden, nicolas@maths.lth.se

by non-linear methods [7] or alternation [5, 13, 17]. However, the convergence to the global minimum is not guaranteed because they can get stuck in a local minimum. Although good performances have been reported when initializing the algorithms with a random solution [7, 12, 17], an initialization as close as possible to the global minimum is recommended. Hierarchical approaches proceed by factorizing overlapping sub-blocks of the measurement matrix [8, 16]. The solutions are then merged in a hierarchical manner, and care must be taken in choosing the merging scheme. This allows to deal with very large factorization problems.

Batch algorithms provide a solution for initializing iterative algorithms with a low computational cost. They usually minimize an approximation to the reprojection error to simplify the optimization and avoid local minima [9, 14, 24]. In the absence of noise, they find the global minimum. The approximation is done by computing the two factors through two linear steps. Constraints on one of the two factors are computed to span the whole solution space. Once the first factor is estimated, the second one can be easily computed. Non-linear and batch algorithms are usually considered complementary solutions. In [19], essential matrices are used between pairwise views to estimate the motion of all the cameras without the structure. This is similar to our solution in philosophy although a very different approach is taken.

It has been shown that the reconstruction problem is considerably simplified when observing a reference plane in all the images of the sequence [11, 22]. The structure constraints we propose handle this situation naturally. The solution must still proceed in two steps, but has the benefit that the objects **do not** need to be visible in all of the images. Finally, robustness is enforced one constraint at a time, rather than globally [1, 2, 12], thereby allowing the use of RANSAC-type algorithms.

3. Notation and Preliminaries

Notation. Matrices are in *sans-serif*, *e.g.* M, and 'joint matrices' in calligraphic characters, *e.g.* $\mathcal{M}$. Vectors are always in bold, *e.g.* $\mathbf{v}$. The matrix operator $\odot$ is the Hadamard element-wise product. Finally, $\mathbb{P}$ is the projective space.

Problem statement. We are given a measurement matrix M such that $\mathsf{M} = \mathsf{M}^* + N(0, \sigma^2$ The factorization of a matrix M with missing data is formulated as the problem of finding a weighted approximation of M with the closest rank r matrix (AB) such that:

$$\min_{\mathsf{A},\mathsf{B}} \|\mathsf{W}_{(n\times m)} \odot \left(\mathsf{M}_{(n\times m)} - \mathsf{A}_{(n\times r)}\mathsf{B}_{(r\times m)}\right) \|,$$

where M is called the *measurement matrix* composed of points $\mathbf{m}_p^j$, and W is a weighting matrix with zeros for missing elements in M. In some problems, constraints on elements of A and B must be enforced, *e.g.* affine SfM. In SfM, B is called the Joint Structure Matrix (JSM) and represents the 3D points $\mathbf{q}^j \in \mathbb{P}^3$, $\mathsf{A} = (\mathcal{P} \quad \mathbf{t})$ is the Joint Projection Matrix (JPM) and consists of the stacked camera projection matrices.

Affine SfM can be formulated as a rank-3 or a rank-4 factorization of a measurement matrix $\mathcal{M}$, depending on whether the input matrix has missing data or not (with the exception of [5], where predictions are made for the missing data). In the rank-3 case, the projection can be expressed with:

$$\mathbf{m}^j_{p(2\times 1)} = \mathsf{P}_{p(2\times 3)}\mathbf{q}^j_{(3\times 1)} + \mathbf{t}_{p(2\times 1)} \tag{1}$$

where an optimal choice for the *joint translation vector* $\mathbf{t}$ can be computed as the column means of $\mathcal{M}$. It can be eliminated from (1), giving the centered measurement matrix $(\mathcal{M} - \mathbf{t}\mathbf{1}^\mathsf{T})$. The factorization of this matrix computed using SVD is an optimal solution in terms of the reprojection error [23]. We have rank-4 when data are missing, so the joint translation vector cannot be computed *a priori*. Bilinear matrix factorization [17] provides a solution as long as the last row of the JSM is constrained to unity:

$$\mathcal{M}_{(2n\times m)} = \begin{pmatrix}\mathcal{P}_{(2n\times 3)} & \mathbf{t}\end{pmatrix}\begin{pmatrix}\mathcal{Q}_{(3\times m)} \\ \mathbf{1}^\mathsf{T}\end{pmatrix}. \tag{2}$$

3.1. Camera Closure Constraints (CCC)

We review the *Closure Constraints* [9, 10, 24] for affine cameras and give a generic formulation for estimating matching tensors between many views.

Deriving the constraints. Let $\hat{\mathcal{M}}$ be a sub-block of $\mathcal{M}$ without missing data. Selecting a subset of views is done by multiplying to the left by some row-amputated block-diagonal matrix Π with (2×2) identity blocks. Selecting a subset of features is done similarly by multiplying to the right by Γ, an identity matrix amputated of some of its columns, yielding:

$$\hat{\mathcal{M}}_{(2\hat{n}\times\hat{m})} \stackrel{\text{def}}{=} \Pi\mathcal{M}\Gamma.$$

In this case, the measurements can be expressed as:

$$\hat{\mathcal{M}} = \Pi\mathcal{M}\Gamma = \Pi\mathcal{P}\mathcal{Q}\Gamma + \Pi\mathbf{t}\mathbf{1}^\mathsf{T}\Gamma = \hat{\mathcal{P}}\hat{\mathcal{Q}} + \hat{\mathbf{t}}\mathbf{1}^\mathsf{T}. \tag{3}$$

We define $\boldsymbol{\mu}_m$ as:

$$\boldsymbol{\mu}_m \stackrel{\text{def}}{=} \frac{1}{m}\mathbf{1}_{(m\times 1)},$$

which computes the column means of an $(n \times m)$ matrix by multiplying to the right. We define the centered measurement matrix as:

$$\bar{\mathcal{M}} \stackrel{\text{def}}{=} \hat{\mathcal{M}} - \hat{\mathcal{M}}\boldsymbol{\mu}_{\hat{m}}\mathbf{1}^\mathsf{T}_{(1\times\hat{m})}, \tag{4}$$

which does not equal $(\hat{\mathcal{P}}\hat{\mathcal{Q}})$ in general, as the row means of the sub-blocks are not necessarily those of the complete matrix. The SVD $\bar{\mathcal{M}} = \mathsf{U}\Sigma\mathsf{V}^\mathsf{T}$ can be used to compute optimal rank-3 factors given by the leading 3 columns of U and 3 rows of $\Sigma\mathsf{V}^\mathsf{T}$. The first factor gives the a solution for the partial Joint Projection Matrix while the remaining columns of U form a basis for the best approximation to the left kernel of $\bar{\mathcal{M}}$, which we call *centered matching tensor*, denoted $\bar{\mathcal{N}}$. We have:

$$\bar{\mathcal{N}}^\mathsf{T}\bar{\mathcal{M}} = \mathbf{0},$$

and from (4):

$$\bar{\mathcal{N}}^\mathsf{T}\begin{pmatrix}\hat{\mathcal{M}} & -\hat{\mathcal{M}}\boldsymbol{\mu}_{\hat{m}}\mathbf{1}^\mathsf{T}\end{pmatrix} = \mathbf{0},$$

that rewrites as:

$$\underbrace{\begin{pmatrix}\bar{\mathcal{N}}^\mathsf{T} & -\bar{\mathcal{N}}^\mathsf{T}\hat{\mathcal{M}}\boldsymbol{\mu}_{\hat{m}}\end{pmatrix}}_{\mathcal{N}^\mathsf{T}}\begin{pmatrix}\hat{\mathcal{M}}\\ \mathbf{1}^\mathsf{T}\end{pmatrix} = \mathbf{0}, \tag{5}$$

where appears the non-centered matching tensor $\mathcal{N}$ we are seeking. Note that directly computing a tensor from $\left(\hat{\mathcal{M}}^\mathsf{T}\ \mathbf{1}\right)^\mathsf{T}$ would not be optimal in terms of reprojection error. Tensor $\mathcal{N}$ corresponds to the classical affine matching tensor. The affine fundamental matrix has 4 degrees of freedom, and the non-centered left kernel obtained from a measurement matrix with four rows has 5 components up to scale. A matching tensor computed from a matrix with six rows is of size (3×6) and has orthonormal rows, which leaves 12 degrees of freedom like the affine trifocal tensor.

Estimating the Joint Projection Matrix. The unity constraint on the last row of the Joint Structure Matrix can be expressed with extra rows in $\mathcal{M}$ and $\mathcal{P}$:

$$\begin{pmatrix}\mathcal{M}\\ \mathbf{1}^\mathsf{T}\end{pmatrix} = \begin{pmatrix}\mathcal{P} & \mathbf{t}\\ \mathbf{0}_{(1\times 3)} & 1\end{pmatrix}\begin{pmatrix}\mathcal{Q}\\ \mathbf{1}^\mathsf{T}\end{pmatrix}. \tag{6}$$

Let:

$$\mathsf{D} \stackrel{\text{def}}{=} \Pi^\mathsf{T}\bar{\mathcal{N}};$$

multiplying (6) by $\begin{pmatrix}\Pi & \mathbf{0}\\ \mathbf{0} & 1\end{pmatrix}$ to the left and Γ to the right and substituting in (5), we obtain:

$$\begin{pmatrix}\mathsf{D}^\mathsf{T} & -\bar{\mathcal{N}}^\mathsf{T}\hat{\mathcal{M}}\boldsymbol{\mu}_{\hat{m}}\end{pmatrix}\begin{pmatrix}\mathcal{P} & \mathbf{t}\\ \mathbf{0} & 1\end{pmatrix} = \mathbf{0},$$

$$\boxed{\mathsf{D}^\mathsf{T}\begin{pmatrix}\mathcal{P} & \mathbf{t}\end{pmatrix} = \begin{pmatrix}\mathbf{0}_{(1\times 3)} & \bar{\mathcal{N}}^\mathsf{T}\hat{\mathcal{M}}\boldsymbol{\mu}_{\hat{m}}\end{pmatrix}.}$$

Stacking every such constraint computed from different sub-blocks of $\mathcal{M}$ in a single matrix equation, we get:

$$\begin{pmatrix}\mathsf{D}_1^\mathsf{T}\\ \vdots\\ \mathsf{D}_l^\mathsf{T}\end{pmatrix}\begin{pmatrix}\mathcal{P} & \mathbf{t}\end{pmatrix} = \begin{pmatrix}\mathbf{0} & \bar{\mathcal{N}}_1^\mathsf{T}\hat{\mathcal{M}}_1\boldsymbol{\mu}_{\hat{m}}\\ \vdots & \vdots\\ \mathbf{0} & \bar{\mathcal{N}}_l^\mathsf{T}\hat{\mathcal{M}}_l\boldsymbol{\mu}_{\hat{m}}\end{pmatrix}.$$

The design matrix, denoted $\mathcal{D} \stackrel{\text{def}}{=} (\mathsf{D}_1, \ldots, \mathsf{D}_l)^\mathsf{T}$, is highly sparse. This is exploited when computing the Linear Least Square (LLS) solution. As explained in [9], choosing the projection matrix of some of the cameras fixes the gauge of the system and ensures a full-rank design matrix. The error minimized by this method is difficult to interpret because it is expressed in terms of matching tensors. Once the JPM is estimated, the structure can be computed by affine triangulation.

4. Batch Matrix Factorization for Affine SfM and Generalization of CCC

4.1. Summary of the Algorithms

The algorithms we propose follow the typical steps of batch algorithms: 1) Measurement sub-matrices without missing data are found. 2) Each of these matrices is used to compute a constraint, either on the partial Joint Projection Matrix or partial Joint Structure Matrix. 3) The constraints are combined to estimate one of the factors. 4) The second factor is estimated by camera resectioning or triangulation. 5) Finally, non-linear or alternation methods refine the solution.

4.2. Camera Basis Constraints (CBC)

We show how basis constraints can be used instead of matching tensors. The partial JPM $\mathcal{P}$ can be computed alone, *i.e.* without the joint translation vector. This is done by aligning bases of the projection matrices of partial reconstructions performed on measurement sub-matrices. Once $\mathcal{P}$ is recovered, the joint translation vector and the structure can be computed together to minimize the reprojection error.

Consider a centered sub-matrix $\bar{\mathcal{M}}$ of $\mathcal{M}$ and compute its SVD $\bar{\mathcal{M}} = \mathsf{U}\Sigma\mathsf{V}^\mathsf{T}$. The 3 leading columns of U, denoted $\bar{\mathsf{U}}$, form a basis of $\hat{\mathcal{P}}$, that is, there is a 3×3 invertible matrix Z (the aligning transformation) such that:

$$\hat{\mathcal{P}} = \bar{\mathsf{U}}\mathsf{Z}, \tag{7}$$

leading to the Camera Basis Constraint:

$$\boxed{\Pi\mathcal{P} = \bar{\mathsf{U}}\mathsf{Z}.} \tag{8}$$

Because of the remark following (4), this is not trivial. To demonstrate this, we note that multiplying an $(n \times m)$ matrix by:

$$\boldsymbol{\vartheta}_m \stackrel{\text{def}}{=} \mathrm{I} - \frac{1}{m}1_{(m\times m)} = \mathrm{I} - \boldsymbol{\mu}_m\mathbf{1}_{(1\times m)},$$

to the right subtracts the row mean from each of its entries. Consequently:

$$\bar{\mathcal{M}} = \Pi\mathcal{M}\Gamma\boldsymbol{\vartheta}_{\hat{m}} = \Pi\mathcal{P}\mathcal{Q}\Gamma\boldsymbol{\vartheta}_{\hat{m}} + \Pi\mathbf{t}\mathbf{1}^\mathsf{T}\Gamma\boldsymbol{\vartheta}_{\hat{m}} = \Pi\mathcal{P}\mathcal{Q}\Gamma\boldsymbol{\vartheta}_{\hat{m}},$$

since $\Pi \mathbf{t} \mathbf{1}^\mathsf{T} \Gamma \boldsymbol{\vartheta}_{\hat{m}} = 0$. Hence, the columns of $\bar{\mathcal{M}}$ are linear combinations of those of $\Pi\mathcal{P}$, and since $\bar{\mathcal{M}}$ has rank 3, the same is true of $\bar{\mathsf{U}}$.

4.2.1 Solving for the Joint Projection Matrix

Computing many CBC's for different sub-blocks of the measurement matrix amounts to performing partial affine reconstructions. Solving for the Joint Projection Matrix is done by: 1) aligning basis constraints to recover the reduced JPM and 2) recovering the joint translation vector and the structure.

Let l be the number of CBC's. Aligning bases together involves minimizing:

$$\sum_{k=1}^{l} \|\bar{\mathsf{U}}_k \mathsf{Z}_k - \Pi_k \mathcal{P}\|^2 = \sum_{k=1}^{l} \left\| \begin{pmatrix} \Pi_k & -\bar{\mathsf{U}}_k \end{pmatrix} \begin{pmatrix} \mathcal{P} \\ \mathsf{Z}_k \end{pmatrix} \right\|^2,$$

which is a simple LLS system, that can be rewritten as:

$$\left\| \underbrace{\begin{pmatrix} \Pi_1 & -\bar{\mathsf{U}}_1 & & \mathbf{0} \\ \vdots & & \ddots & \\ \Pi_l & \mathbf{0} & & -\bar{\mathsf{U}}_l \end{pmatrix}}_{\mathcal{D}} \underbrace{\begin{pmatrix} \mathcal{P} \\ \mathsf{Z}_1 \\ \vdots \\ \mathsf{Z}_l \end{pmatrix}}_{\mathsf{X}} \right\|^2. \tag{9}$$

Although the size of the design matrix $\mathcal{D}$ is quadratic in the number of bases, the equation system can be minimized efficiently. Indeed it is extremely sparse, and we do not need to compute the aligning matrices Z_k. The minimized error is the alignment of the optimal cameras of partial reconstructions. Although this error is algebraic, as when expressing the error in terms of matching tensor constraints, our experiments suggest it is more stable, *c.f.* §7.

Once the partial JPM $\mathcal{P}$ is estimated, we propose two approaches for estimating the translations and the structure. The best solution comes from doing both at the same time, by using an LLS formulation. It minimizes the reprojection error. This is because the orientation and intrinsics of the camera are already estimated up to a (3×3) invertible transformation G, which has no effect on the minimized error:

$$\|\mathcal{M} - \mathcal{P}\mathcal{Q} - \mathbf{1}^\mathsf{T}\mathbf{t}\| = \|\mathcal{M} - \mathcal{P}\mathsf{G}\mathsf{G}^{-1}\mathcal{Q} - \mathbf{1}^\mathsf{T}\mathbf{t}\|. \tag{10}$$

However, for a long sequence, this equation system uses a lot of memory and is rather long to minimize. In this case, it is more efficient to estimate only the joint translation vector and then perform individual triangulation for each feature track. To this end, we combine the computed basis $\bar{\mathsf{U}}$ with the translation $\bar{\mathbf{t}} = \hat{\mathcal{M}}\boldsymbol{\mu}_{\hat{m}}$ of the cameras of the partial reconstructions. Thus, the camera alignment can also be performed with:

$$\Pi \begin{pmatrix} \mathcal{P} & \mathbf{t} \end{pmatrix} = \begin{pmatrix} \hat{\mathcal{P}} & \hat{\mathbf{t}} \end{pmatrix} = \begin{pmatrix} \bar{\mathsf{U}} & \bar{\mathbf{t}} \end{pmatrix} \begin{pmatrix} \mathsf{Z} & \mathbf{v} \\ \mathbf{0}_{(1\times3)} & 1 \end{pmatrix},$$

where matrices $\hat{\mathsf{U}}$ and Z are those from (8). The joint translation vector can be estimated by minimizing:

$$\left\| \mathcal{D} \begin{pmatrix} \mathbf{t} \\ \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_l \end{pmatrix} - \begin{pmatrix} \bar{\mathbf{t}}_1 \\ \vdots \\ \bar{\mathbf{t}}_l \end{pmatrix} \right\|^2, \tag{11}$$

where $\mathcal{D}$ is the design matrix of (9).

In [14], bases for cameras were computed from $\hat{\mathcal{M}}$, not $\bar{\mathcal{M}}$ like we do. The partial reconstruction corresponding to these cameras is not optimal. Furthermore, a method for estimating $\hat{\mathbf{t}}$ from the bases independently of the structure is not given. This is essential when dealing with very large sequences or data corrupted by outliers (*c.f.* §6).

4.3. Structure Closure Constraints (SCC)

Structure Closure Constraint is the analogue of the CCC applied to the Joint Structure Matrix. From (2), a matching tensor is in the column space of $\mathcal{Q}$ and $\mathbf{1}^\mathsf{T}$. It can be computed using SVD by minimizing:

$$\boxed{\|\hat{\mathcal{M}}\mathcal{N}\|^2, \text{ subject to } \mathbf{1}^\mathsf{T}\mathcal{N} = 0.}$$

Note that unlike the CCC, the SCC is not computed from a centered input matrix. This is because the last row of the JSM must be $\mathbf{1}^\mathsf{T}$. A more formal explanation is given in §4.4. By accumulating many constraints, we can form a design matrix $\mathcal{D}$ with:

$$\mathsf{D}_i \stackrel{\text{def}}{=} \Gamma\mathcal{N}_i.$$

By construction $\mathcal{D}$ is rank deficient because each of its rows vanishes on $\mathbf{1}^\mathsf{T}$. Hence, the right singular vector corresponding to the smallest singular value, equal to zero, is $\mathbf{1}/\|\mathbf{1}\|$. The estimate for $\mathcal{Q}^\mathsf{T}$ is given by the next three right singular vectors of $\mathcal{D}$.

4.4. Structure Basis Constraints (SBC)

As with Camera Closures, using Structure Closures implies minimizing a purely algebraic function difficult to interpret. Structure Bases can be used to estimate partial reconstructions which are then aligned together in a single computation. From an SVD of $\bar{\mathcal{M}} = \mathsf{U}\Sigma\mathsf{V}^\mathsf{T}$, the three leading columns of V estimate the structure, up to an affine transformation, in the partial reconstruction corresponding to $\hat{\mathcal{M}}$. However we prefer using $\bar{\mathsf{V}}$ as the three leading columns of $\mathsf{V}\Sigma^\mathsf{T}$, as explained below. It can be aligned with the structure through:

$$\hat{\mathcal{Q}}^\mathsf{T} = \mathsf{Z}_{(3\times4)} \begin{pmatrix} \bar{\mathsf{V}} & \mathbf{1} \end{pmatrix}^\mathsf{T},$$

leading to the Structure Basis Constraint:

$$\boxed{\Gamma^\mathsf{T}\mathcal{Q}^\mathsf{T} = \mathsf{Z}\bar{\mathsf{V}}^\mathsf{T}.}$$

Note that unlike a CBC, an SBC cannot be aligned with a (3×3) matrix. This is because the row space of $\bar{\mathsf{V}}$ is that of:

$$\Pi\mathcal{P}\mathcal{Q}\Gamma\boldsymbol{\vartheta}_{\hat{m}} = \Pi\mathcal{P}\begin{pmatrix}\hat{\mathsf{Q}}^1\boldsymbol{\vartheta}_{\hat{m}}\\ \hat{\mathsf{Q}}^2\boldsymbol{\vartheta}_{\hat{m}}\\ \hat{\mathsf{Q}}^3\boldsymbol{\vartheta}_{\hat{m}}\\ \mathbf{1}^\mathsf{T}\boldsymbol{\vartheta}_{\hat{m}}\end{pmatrix} = \Pi\mathcal{P}\begin{pmatrix}\hat{\mathsf{Q}}^1 - \hat{\mathsf{Q}}^1\boldsymbol{\mu}_{\hat{m}}\mathbf{1}_{(1\times m)}\\ \hat{\mathsf{Q}}^2 - \hat{\mathsf{Q}}^2\boldsymbol{\mu}_{\hat{m}}\mathbf{1}_{(1\times m)}\\ \hat{\mathsf{Q}}^3 - \hat{\mathsf{Q}}^3\boldsymbol{\mu}_{\hat{m}}\mathbf{1}_{(1\times m)}\\ \mathbf{0}^\mathsf{T}\end{pmatrix},$$

where the $\hat{\mathsf{Q}}^i$'s are the rows of $\hat{\mathsf{Q}}$, that is, the row space of $\bar{\mathsf{V}}$ is the one generated by the $\hat{\mathsf{Q}}^i$'s and $\mathbf{1}^\mathsf{T}$, but not necessarily only by the $\hat{\mathsf{Q}}^i$'s. Partial reconstructions can be aligned together by solving an equation system similar to (9).

Choosing the bases. Consider two partial reconstructions $\bar{\mathsf{V}}$ and $\bar{\mathsf{V}}'$. Aligning them together amounts to finding:

$$\arg\min_{\mathsf{Z}} \|\bar{\mathsf{V}}' - \mathsf{Z}\bar{\mathsf{V}}\|^2. \tag{12}$$

We show that when $\bar{\mathsf{V}}$ is chosen so that the corresponding projection matrix $\hat{\mathcal{P}}$ is orthonormal, the 3D error approximates the reprojection error [3]. Projecting the residual given by (12) into the images corresponding to the block, we obtain:

$$\begin{aligned}\|\hat{\mathcal{P}}\bar{\mathsf{V}}' + \mathbf{1}^\mathsf{T}\hat{\mathbf{t}} - (\hat{\mathcal{P}}\mathsf{Z}\bar{\mathsf{V}} + \mathbf{1}^\mathsf{T}\hat{\mathbf{t}})\|^2 &= \|\hat{\mathcal{P}}\bar{\mathsf{V}}' - \hat{\mathcal{P}}\mathsf{Z}\bar{\mathsf{V}}\|^2\\ &= \|\hat{\mathcal{P}}(\bar{\mathsf{V}}' - \mathsf{Z}\bar{\mathsf{V}})\|^2.\end{aligned}$$

Thanks to the orthonormal property $\hat{\mathcal{P}}^\mathsf{T} = \hat{\mathcal{P}}^\dagger$, our error function simplifies to:

$$\mathrm{tr}\left((\bar{\mathsf{V}}' - \mathsf{Z}\bar{\mathsf{V}})^\mathsf{T}\underbrace{\hat{\mathcal{P}}^\mathsf{T}\hat{\mathcal{P}}}_{\mathrm{I}_{(3\times 3)}}(\bar{\mathsf{V}}' - \mathsf{Z}\bar{\mathsf{V}})\right) = \|\bar{\mathsf{V}}' - \mathsf{Z}\bar{\mathsf{V}}\|^2.$$

The minimization is only exact if both $\bar{\mathsf{V}}$ and $\bar{\mathsf{V}}'$ have been estimated without error in their respective partial reconstruction. Hence, under noise, it only approximates the reprojection error.

4.5. Enforcing Constraints

In many situations, prior knowledge about the configuration of the scene structure is available. Examples are two cameras with identical position and/or orientation, smooth camera path, known 3D points and planar structure. In our algorithms, a certain number of constraints can be enforced. This is done when estimating the first factor. As a consequence, one can only force constraints on either cameras or structure.

Most of our equation systems are homogeneous, like (9). In order to simplify the constrained optimization, a well known trick is to select a gauge, *i.e.* to give a value to some rows of X, and solve the resulting regression problem. A valid gauge fixes the degrees of freedom of the affine ambiguity, which makes the original design matrices rank deficient. Once this is done, each column $\mathbf{X}^i$ of the solution matrix X can be individually estimated by regression under linear constraints, which is an instance of convex quadratic programming [4]. Aligning two cameras together can be done by choosing:

$$\begin{aligned}\begin{pmatrix}\mathsf{P}_a & \mathbf{t}_a\end{pmatrix} = \begin{pmatrix}\mathsf{P}_b & \mathbf{t}_b\end{pmatrix} &= \begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\end{pmatrix}\\ \begin{pmatrix}\mathsf{P}_c & \mathbf{t}_c\end{pmatrix} &= \begin{pmatrix}0 & 1 & 1 & 0\\ * & * & * & *\end{pmatrix}\end{aligned}$$

Others can also be aligned through equality constraints $\mathsf{P}_e - \mathsf{P}_f = \mathsf{0}$, or variable elimination.

Enforcing known 3D points can be done similarly with structure constraints. The gauge is fixed with at least four non-coplanar points. Three planar surfaces can also be enforced by forcing groups of points to have their (X, Y, Z) coordinates to either $(0, *, *)$, $(*, 0, *)$ or $(*, *, 0)$, where $*$ means the coordinate is not fixed[1]. For more than three planes, their absolute equation has to be known. Observe that the gauge is at least partially fixed by the points located on the planes. Consequently, caution must be taken to ensure that these planes are really orthogonal up to an affine transformation. Prior information from points and planar structures can also be incorporated as long as the constraints are compatible, *i.e.* the affine ambiguity is fixed by the same matrix.

5. The Perspective Camera Model

In projective SfM, the point coordinates are multiplied by their projective depth λ_p^j and the projection is performed by (3×4) matrices defined up to scale:

$$\lambda_p^j\begin{pmatrix}\mathbf{m}_p^j\\ 1\end{pmatrix} = \begin{pmatrix}\mathsf{P}_{p(3\times 3)} & \mathbf{t}_{p(3\times 1)}\end{pmatrix}\mathbf{q}_{(4\times 1)}^j$$

where $\mathbf{m}_p^j \in \mathbb{R}^2$ is an interest point tracked throughout the sequence and $\mathbf{q}^j \in \mathbb{P}^3$. We assume that the projective depths are estimated *a priori*. In our experiments, we used the algorithm presented in [14]. Rank-4 factorization is then performed without any restriction on $\mathbf{q}^j$, unlike for the affine case. Closures and rank-4 bases are purely algebraic and are computed from $\hat{\mathcal{M}}_{(3\hat{n}\times\hat{m})} = \Pi\mathcal{M}\Gamma$ instead of $\bar{\mathcal{M}}$. This is akin to what was presented in [14]. The main difference is the way system (9) is minimized. They used Matlab's EIGS method to estimate four orthonormal solution vectors. We used a solution based on fixing one of the bases $\bar{\mathsf{U}}_i$ to identity and solve the resulting sparse regression problem. Finally, we performed a QR factorization on the estimated JPM to orthonormalize its columns. It was suggested in [14] that gluing via points cannot be used with the perspective camera model. We did not encounter this limitation.

[1] Details will be provided in a technical report.

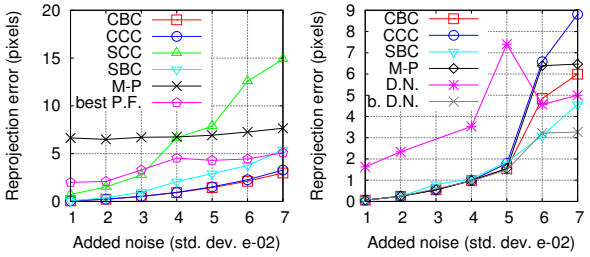


Figure 1. Comparison of the algorithms for the simulated sequence. M-P stands for Martinec-Pajdla [14], P.F. for Powerfactorization [17], D.N. for Damped Newton [7] and b. for the best solution out of the 15 trials. **Left)** Affine model. **Right)** Perspective model (SCC not shown).

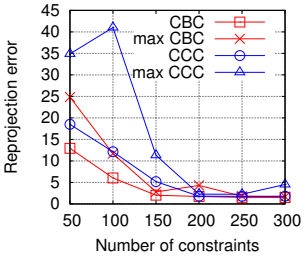


Figure 2. Comparison for the number of constraints between CCC and CBC, for our simulated sequence.

6. Robustness

To deal with outliers, we use random sampling to compute each Closure/Basis Constraint. Once the camera path or 3D points have been computed, we rely again on random sampling to perform robust triangulation or resection. We do not reject points directly from the computed matching tensors, which can leave certain outliers behind. We avoid performing the optimization using a robust (non-convex) cost function, that would typically have a lot of local minima, or using alternation with re-weighting.

For CBC, we rely on (11) to compute the position of the cameras, because the estimation of the structure and the translation vectors in a single step (*i.e.* using (10)) would be computationally expensive in a random sampling strategy.

7. Experiments and Analysis

We compared our three algorithms to Guilbert *et al.* [9], Martinec-Pajdla [14], Hartley-Schaffalitzky [17] and Buchanan-Fitzgibbon [7] methods on simulated and real sequences. Powerfactorization and Damped Newton were used respectively for the affine and perspective camera model. They were limited to 1000 iterations, which was sufficient to attain convergence from a random solution in most cases. The tracks were sorted in order of appearance in the sequence and we assumed that the resulting measurement matrix was approximately band-diagonal as in figure 4. Heuristics were used to find complete sub-blocks, making sure that constraints are approximately equally distributed among the cameras or the 3D points, depending on the constraint type.

Simulation. The simulated sequence consisted of 50 cameras and around 900 3D points with a lot of occlusion, resulting in around 96% missing data. In figure 1, the algorithms are compared for both camera models at different levels of Gaussian noise. Our results strongly suggest that Camera Basis performs best, especially under affine projection. Under perspective projection, Closure and Structure Basis gave similar results with a slight advantage to SBC in the presence of very high noise. The advantage vanished when projective depths were exact (not shown here). Hence, SBC seems to be the most robust to erroneous projective depths. The CBC method performed slightly better than Martinec-Padjla's, a likely result of our balancing of the constraints. Structure Closures performed rather poorly. This is not surprising, at least for the affine case, since it is the only one of our algorithms whose constraint is not optimal. Powerfactorization and Damped Newton provided good performance in the best case, but on average, they did not converge to satisfying solutions.

Most of the computation time was spent finding complete sub-blocks from the measurement matrix. The time for solving the two factors was almost negligible. Hence, a good algorithm performs well even with a small number of constraints. We compared CCC and CBC on this criteria with the simulated sequence as well as with the *Teddy Bear* sequence (*c.f.* figure 2 and 3(a)). The number of closures had to be nearly twice as large as that of bases to obtain a comparable average reprojection error. The maximal reprojection error also suggests more stability for bases.

Real sequences. We compared batch algorithms and Powerfactorization on five real sequences, four affine and one perspective, (*c.f.* table 1). To take them out of the comparison, we removed the outliers in each of the sequence using the robust CBC-based algorithm (see test below). For the *Dinosaur* sequence, the algorithm of Guilbert *et al.* and Martinec-Pajdla obtained a better mean reprojection error than what they reported in their paper, respectively 5.4 and 2.57 pixels. This is probably because we used more constraints (around 350). Computation time on an AMD Athlon 64 3500+, in Matlab, for the camera estimation (not including sub-block search) were 0.01 and 0.29 seconds for CCC and CBC (for which five different gauges were tested) and 0.34 seconds for Martinec-Pajdla. On the larger *Teddy bear* sequence, computation time were respectively 0.49, 0.73, 0.91 seconds (not including the translations since it was much longer minimizing (10) than (11)). Structure Constraint based algorithms were slower because their equation systems are larger. Iterative

Sequence (# Img., # 3D pts, # 2D pts, miss. data)	Mean (max) reprojection error in pixels					
	CCC [9]	CBC	SCC	SBC [3]	P.F. [17]	M-P [14]
Dinosaur (36,2683,11832,96.9%)	0.56 (5.49)	**0.49 (4.62)**	0.65 (7.27)	0.66 (7.12)	1.75 (73.1)	0.56 (6.99)
Book (95, 254, 10253, 89%)	0.54 (6.86)	**0.50 (5.59)**	0.55 (5.25)	0.56 (5.66)	0.54 (5.96)	2.56 (41.1)
Building (194, 779, 17233, 97%)	**0.86 (14.4)**	0.95 (22.8)	1.24 (17.5)	1.21 (21.5)	(3.45) 256.8	1.28 (39.2)
Teddy Bear (196, 2480, 93589, 95%)	**0.65 (8.14)**	**0.65 (8.14)**	4.67 (174.5)	1.13 (35.3)	1.91 (38.8)	4.453 (96.97)
Desk (66, 2483, 26771, 95.9%)	0.99 (43.36)	**0.87 (19.5)**	3.93 (132.66)	1.44 (45.18)	—-	**0.83 (24.4)**

Table 1. Comparison between batch methods and Powerfactorization for four real sequences. All were reconstructed using the affine camera model except for the *Desk* sequence. The best solutions are in bold.

algorithms achieved convergence after a few hundred iterations, resulting in minutes, if not hours, of computation. When initialized using a batch method, only a few iterations were necessary.

The *Desk* sequence (*c.f*. figure 4, 5 and 6(c)) was reconstructed using the perspective algorithms and the one based on affine CBC. In the former case, outliers were removed before factorization using fundamental matrices. All batch algorithms but the one based on SCC provided satisfying results. The reconstruction shown in figure 6(c) is the result of refining the solution with Mahamud *et al.* method followed by self-calibration. Projective and Euclidean bundle adjustment improved the reconstruction only slightly. We also achieved reconstruction by initializing a Euclidean bundle adjustment with the robust algorithm based on affine CBC. After convergence, the recovered focal length was similar to the one recovered using projective reconstruction.

Outliers issue. The performance of CCC and CBC for handling outliers was also tested similarly to [18]. Up to 7 % outliers where added to the *Dinosaur* sequence. This gives up to $7n$ % unusable n-view constraints (we used up to 4). The constraints were computed from robustly selected sub-blocks. Thus, as the number of outliers increased, the number of points used to compute the constraints decreased. We tested: 1) the percentage of recovered valid outliers, 2) the percentage of removed points that were in fact inliers (false positive) and 3) the average reprojection error of the reconstruction using the original data set. Our results are shown in figure 3(b) and 3(c).

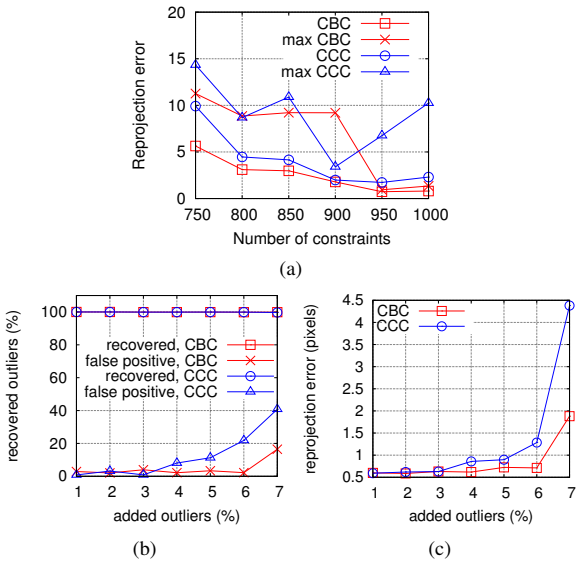


Figure 3. Comparison between CCC and CBC with real data. **a)** number of constraints for the *Teddy Bear* sequence. For added outliers in the *Dinosaur* sequence: **b)** percentage of outliers recovered and percentage of false positive, **c)** average reprojection error of the original data.

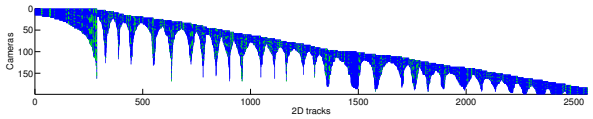


Figure 4. Tracks (blue/dark) and detected outliers (green/light) from the *Desk* sequence.

8. Conclusion

We presented algorithms for efficient batch matrix factorization. Constraints on measurement sub-matrices are combined to estimate one of the two factors. We extended Trigg's Camera Closure Constraints to Structure Closure Constraints and proposed Camera and Structure Basis. Experimental results showed that Basis Constraints fared better than state-of-the-art methods on most of our tests, with simulated and real sequences and both for affine and perspective camera models. Future work will focus on the measurement sub-matrix search and selection mechanism, and on experiments for enforcing *a priori* on the structure.

Acknowledgments. To A. Buchanan and D. Martinec for providing their source code, to A. Fitzgibbon and A. Zisserman for the Dinosaur sequence and to K. McHenry and G. Petit for the Teddy bear sequence.

References

[1] H. Aanaes, R. Fisker, K. Astrom, J. M. Carstensen, Robust Factorization. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 9, pp. 1215-1225, Sept., 2002.

Figure 5. Five out of 66 images of the *Desk* sequence. Reconstruction shown in 6(c).

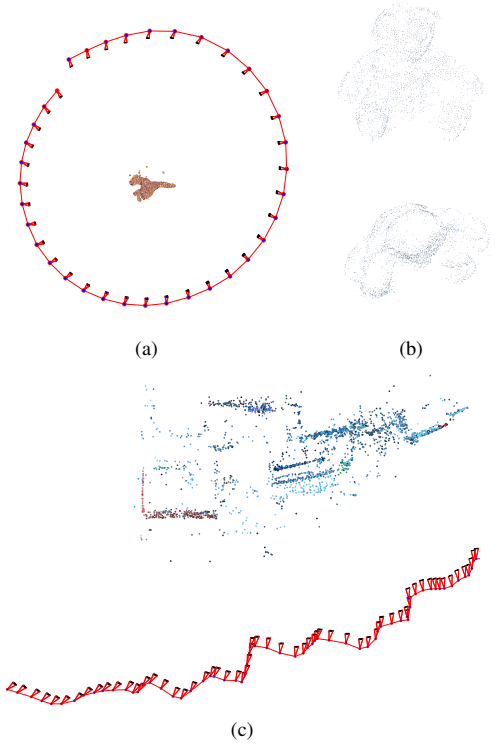


Figure 6. Reconstruction results. **a)** Top view of the *Dinosaur* sequence without using a closing constraint. **b)** Side and top view of the *Teddy Bear* model. **c)** Top view of the *Desk* sequence. All but the *Desk* sequence were obtained solely with an algorithm based on CBC.

[2] P. Anandan and M. Irani, Factorization with Uncertainty. International Journal of Computer Vision (IJCV), 49(2-3): 101-116, September 2002.

[3] A. Bartoli, H. Martinsson, F. Gaspard, J.-M. Lavest, On Aligning Sets of Points Reconstructed from Uncalibrated Affine Cameras. SCIA 2005.

[4] S. Boyd and L. Vandenberghe, Convex Optimization. Cambridge University Press, 2004.

[5] S. Brandt. Closed-form solutions for affine reconstruction under missing data. ECCV 2002.

[6] C. Bregler, A. Hertzmann, and H. Biermann, Recovering non-rigid 3D shape from image streams. CVPR 2000.

[7] A. M. Buchanan, A. W. Fitzgibbon, Damped Newton Algorithms for Matrix Factorization with Missing Data. CVPR 2005.

[8] A.W. Fitzgibbon, A. Zisserman, Automatic Camera Recovery for Closed or Open Image Sequences. ECCV 1998

[9] N. Guilbert, A. Bartoli and A. Heyden, Affine Approximation for Direct Batch Recovery of Euclidean Motion From Sparse Data. International Journal of Computer Vision, Vol. 69, No. 3, pp. 317-333, September 2006.

[10] F. Kahl, A. Heyden, Affine Structure and Motion from Points, Lines and Conics. International Journal of Computer Vision, 33(3):163-180, 1999.

[11] R. Kaucic, R. Hartley, and N. Dano, Plane-based Projective Reconstruction. ICCV 2001.

[12] Q. Ke, T. Kanade, Robust L-1 Norm Factorization in the Presence of Outliers and Missing Data by Alternative Convex Programming. CVPR 2005.

[13] S. Mahamud, M. Hebert, Y. Omori, J. Ponce, Provably-Convergent Iterative Methods for Projective Structure from Motion. CVPR 2001.

[14] D. Martinec and T. Pajdla, 3D Reconstruction by Fitting Low-rank Matrices with Missing Data. CVPR 2005.

[15] D. Martinec and T. Pajdla, 3D Reconstruction by Gluing Pair-wise Euclidean Reconstructions, or "How to Achieve a Good Reconstruction from Bad Images". 3DPVT 2006.

[16] D. Nistèr, Reconstruction from uncalibrated sequences with a hierarchy of trifocal tensors. ECCV 2003.

[17] R. Hartley and F. Schaffalitzky, PowerFactorization: an approach to affine reconstruction with missing and uncertain data. In Australia-Japan Advanced Workshop on Computer Vision, 2003.

[18] K. Sim, R. Hartley. Removing Outliers Using The L-∞ Norm. CVPR 2006.

[19] K. Sim, R. Hartley, Recovering Camera Motion Using L-∞ Minimization. CVPR 2006.

[20] P. Sturm and B. Triggs, A factorization based algorithm for multi-image projective structure and motion. ECCV 1996.

[21] P. Sturm, Algorithms for Plane-Based Pose Estimation. CVPR 2000.

[22] C. Rother and S. Carlsson, Linear Multi View reconstruction and Camera Recovery using a Reference Plane. IJCV 2002, 49(2/3):117-141.

[23] C. Tomasi, T. Kanade, Shape and motion from image streams under orthography: a factorization method. International Journal of Computer Vision, 9(2):137-154, November 1992.

[24] B. Triggs, Linear projective reconstruction from matching tensors. Image and Vision Computing, 15(8):617625, 1997.

[25] R. Vidal and R. Hartley, Motion Segmentation with Missing Data using PowerFactorization and GPCA. CVPR 2004.

8.1.2 Paper (CVPR'07) – *On Constant Focal Length Self-Calibration From Multiple Views*

On Constant Focal Length Self-Calibration From Multiple Views

Benoît Bocquillon* Adrien Bartoli† Pierre Gurdjos* Alain Crouzil*

Abstract

We investigate the problem of finding the metric structure of a general 3D scene viewed by a moving camera with square pixels and constant unknown focal length. While the problem has a concise and well-understood formulation in the stratified framework thanks to the absolute dual quadric, two open issues remain.

The first issue concerns the generic Critical Motion Sequences, i.e. camera motions for which self-calibration is ambiguous. Most of the previous work focuses on the varying focal length case. We provide a thorough study of the constant focal length case.

The second issue is to solve the nonlinear set of equations in four unknowns arising from the dual quadric formulation. Most of the previous work either does local nonlinear optimization, thereby requiring an initial solution, or linearizes the problem, which introduces artificial degeneracies, most of which likely to arise in practice. We use interval analysis to solve this problem. The resulting algorithm is guaranteed to find the solution and is not subject to artificial degeneracies. Directly using interval analysis usually results in computationally expensive algorithms. We propose a carefully chosen set of inclusion functions, making it possible to find the solution within few seconds.

Comparisons of the proposed algorithm with existing ones are reported for simulated and real data.

1. Introduction

Structure-from-Motion, the recovery of a metric reconstruction, i.e. cameras and points, from images is a fundamental computer vision problem. Its extensive study over the last few decades led to clear geometrical formulations and numerous solution methods. One of the key results is that a projective reconstruction can be computed from uncalibrated images, providing that neither the cameras nor the points lie on a critical surface. The projective reconstruction is equivalent to the sought after metric one up to an unknown upgrading homography of the projective space. Computing this homography from assumptions on the cameras is a *self-calibration* problem, and is equivalent to recovering the unknown intrinsic parameters of the cameras.

Three main approaches can be distinguished from the literature. *(i)* The Kruppa equations [2, 10], requiring pairwise epipolar geometry. *(ii)* The stratified approach, relying on upgrading a projective reconstruction to affine, and linearly solving for the affine-to-metric transformation. The former is solved using the modulus constraint [12] or exhaustive search [5]. *(iii)* Directly computing the projective-to-metric upgrading homography [7]. Triggs [21] proposed a convenient model with the so-called absolute dual quadric, encapsulating the plane at infinity and camera intrinsics in a compact manner. [11] drew on this model for proposing a linear algorithm dealing with the varying focal length case.

We tackle the self-calibration problem for a camera with constant intrinsic parameters in Triggs' absolute dual quadric framework. More precisely, we assume that the camera has square pixels and known principal point. Only the constant focal length thus remains to be computed. Our contributions deal with two important aspects of this problem.

First, section 3, we give a thorough study of the Critical Motion Sequences (CMS) which are generic for this problem. These are camera motions for which self-calibration is ambiguous, i.e. that defeat any self-calibration algorithm. Sturm [15] gave a complete classification of the CMS for constant, all unknown intrinsics. Note that CMS for which the calibration can be partly performed only lead to specific methods, see [6, chap. 19]. An example is the purely translational case for which the affine to metric upgrade is not uniquely defined. The case of a varying focal length with all other intrinsics known has been extensively studied [9, 13, 18], while, except for two cameras [19], the case of a constant focal length remained open.

Second, section 4, we propose a deterministic method for solving the nonlinear self-calibration problem which does not introduce artificial CMS and does not require an initial solution. Previous methods [11, 21] linearize the problem to find an initial solution and refine it through iterative nonlinear optimization. The basic step towards linearization is to neglect the rank deficiency of the dual absolute quadric. Appart from being suboptimal, linearizing the problem introduces artificial CMS, most of which likely to appear in

*UPS - IRIT, France, {bocquillon,gurdjos,crouzil}@irit.fr
†CNRS - LASMEA, France, Adrien.Bartoli@gmail.com

practice. For instance, a fixating camera is not a generic CMS but defeats linear algorithms. Iterative nonlinear optimization is prone to falling in local minima, in particular if the initial solution lies "too far" from the optima, sought after one. Our algorithm is based on Interval Analysis Global Optimization. It finds the global minimum of the nonlinear error function over the four unknown parameters – the focal length and the plane at infinity. Computational time is reasonable, the order of which ranges from few seconds to a minute. Experimental results show that it scales linearly with the number of images while being almost unaffected by the level of noise on the data. We note that self-calibration based on Interval Analysis Global Optimization and the Kruppa equations is proposed in [3]. This approach is however different from ours, requiring hours to find the solution, and subject to important singularities related to the Kruppa equations.

Section 5 reports experimental evaluation and comparison with other algorithms. Section 6 concludes and discusses the paper. Next section reviews some background.

2. Background

In some metric coordinate frame, the camera projection matrices are written as $\mathtt{P}^i_M = \mathtt{K}^i\mathtt{R}^{i\top}\left(\mathtt{I}|-\mathbf{t}^i\right)$ with $i = 1, ..., n$, where $\mathtt{K}^i$ is the upper triangular calibration matrix which encodes the intrinsic parameters, $\mathtt{R}^i$ represents the orientation of the camera and $\mathbf{t}^i$ the camera centre in the world coordinate frame. The reconstruction problem is equivalent to finding a 4×4 rectifying homography $\mathtt{H}$ to upgrade the projective cameras $\mathtt{P}^i$ to $\mathtt{P}^i_M$, such that $\mathtt{P}^i_M = \mathtt{P}^i\mathtt{H}$ for $i = 1, ..., n$. The absolute dual quadric $\mathtt{Q}^*_\infty$ can be represented by a 4×4, semi-definite, rank-3 matrix which projects to the dual absolute conic $\boldsymbol{\omega}^* = \mathtt{K}\mathtt{K}^\top$:

$$\boldsymbol{\omega}^{*i} = \mathtt{P}^i\mathtt{Q}^*_\infty\mathtt{P}^{i\top}. \tag{1}$$

This quadric encodes both the absolute conic $\boldsymbol{\Omega}_\infty$ and the plane at infinity $\boldsymbol{\Pi}_\infty$. In a metric coordinate frame, $\mathtt{Q}^*_\infty$ has the form $\hat{\mathtt{I}} = \begin{pmatrix} \mathtt{I} & \mathbf{0} \\ \mathbf{0}^\top & 0 \end{pmatrix}$ and in a projective frame we have $\mathtt{Q}^*_\infty = \mathtt{H}\hat{\mathtt{I}}\mathtt{H}^\top$. If intrinsic parameters are constant, $\boldsymbol{\omega}^{*i} \sim \boldsymbol{\omega}^{*j}$, where $\sim$ denotes equality up to a scale factor. For all known intrinsic parameters except the focal length α and writing $\mathtt{P}^1$ as $\mathtt{P}^1 = (\mathtt{I}|\mathbf{0})$, we have $\boldsymbol{\omega}^{*1} = \mathrm{diag}(\alpha^2, \alpha^2, 1)$. Each image pair $(1, j)$ gives us a set of five equations:

$$\begin{array}{ccc} \boldsymbol{\omega}^{*j}_{12} = 0 & \boldsymbol{\omega}^{*j}_{13} = 0 & \boldsymbol{\omega}^{*j}_{23} = 0 \\ \boldsymbol{\omega}^{*j}_{11} = \boldsymbol{\omega}^{*j}_{22} & \alpha^2\boldsymbol{\omega}^{*j}_{33} = \boldsymbol{\omega}^{*j}_{22} & \text{for } j = 2, ..., n. \end{array} \tag{2}$$

Solving for the 4 unknowns $\gamma = \alpha^2$ and $\boldsymbol{\Pi}_\infty$ is possible for $n \geq 2$ images. There are mainly three approaches: *(i) Linear method:* It consists in keeping the first four equations in (2), which are linear. *(ii) Quasi-linear method:* In [21], Triggs linearizes the problem by introducing additional unknowns and by solving for the entries of $\mathtt{Q}^*_\infty$ and $\boldsymbol{\omega}^{*j}$. *(iii) Nonlinear method:* This is the direct approach. Local iterative minimization methods are usually used. They require a good initialization. The first two methods have two major problems: First, the rank-3 constraint on $\mathtt{Q}^*_\infty$ is not automatically ensured (generally, this is imposed in a further step). This means that the obtained plane at infinity cannot be the supporting plane of the obtained absolute conic. Second, they do not ensure either the positive semi-definiteness condition on $\boldsymbol{\omega}^*$, preventing the Cholesky decomposition of this latter. Notice that in the two view case, the problem is equivalent to solve a quadratic equation.

In this work, we propose an algorithm based on the nonlinear formulation and on a global optimization method.

3. Generic Singularities

3.1. Previous Works

We distinguish the problems of recovering the intrinsics from those of recovering the affine and Euclidean structures, all three problems being usually packaged under the single term self-calibration. These problems carry theoretical singularities i.e., degenerate configurations at which problems have no solution or ambiguous solutions, which are generally due to (so-called critical) special camera motions. Theoretical singularities are generic in the sense that they cannot be overcome by *any* algorithm. In contrast, singularities introduced by algorithms are called artificial.

As suggested in [17], self-calibration algorithms may be classified into at least three groups, according to increasing levels of singularities. The first includes algorithms which only have the generic singularities. The second, which is that of Pollefeys [11] and Triggs' [21] linear algorithms, add singularities by not enforcing the sought-after dual absolute quadric to be rank-3. The last group – including algorithms based on Kruppa equations – add more singularities by not enforcing such quadrics to be the same for any pair of views. Remind that our aim is to design a self-calibration algorithm that falls into the first group i.e., whose implementation does not introduce additional/artificial singularities.

The first thorough study of critical motions for $n \geq 2$ views is due to Sturm in [16], for constant intrinsics. A generalisation progressively incorporating the assumptions of known skew, aspect ratio and principal point has been given by Kahl [8] even if the case of all known intrinsics except a varying focal length has also been studied by Sturm in [18]. Both authors have reported[1] the following (generic) critical motions for $n > 2$:

1. Optical axes are parallel i.e., cameras are translating.

[1] Even if the most complete analysis can be found in [18, §5].

2. Camera centres are aligned. All camera optical axes coincide except at, at most, two positions at which they can be arbitrarily oriented.
3. Camera centres move on two conics, one ellipse $\mathtt{E}$ and one hyperbola $\mathtt{H}$, lying on orthogonal planes that meet in the focal axis of $\mathtt{E}$, such that $\mathtt{E}$ and $\mathtt{H}$ have, as principal vertices, the foci of $\mathtt{H}$ and $\mathtt{E}$, respectively. Optical axes lie on the supporting planes and are tangent to the supported conic at each position.

Before going further, we set up some general background i.e., for any camera model.

In the sequel, the term *virtual conic* will refer to some proper purely imaginary conic i.e., whose real order-3 matrix is definite in its supporting plane. If $\mathbf{\Phi} \in \mathbb{R}^{3\times3}$ represents a conic, then $\mathbf{\Phi}^* \in \mathbb{R}^{3\times3}$ represents its dual (envelope) and $\mathtt{Q}^*_\Phi \in \mathbb{R}^{4\times4}$ represents the associated rank-3 (disk) quadric in 3-space [14].

$\mathcal{D} \subset \mathcal{K}$ will denote some subset of the group $\mathcal{K}$ of upper triangular order-3 matrices, determined by the constraints on intrinsics at our disposal. In accordance with [8, 18], a motion sequence M is *critical for the Euclidean reconstruction* (or simply *critical*) if there exists a virtual conic $\mathbf{\Phi}$ on some plane $\mathbf{\Pi}$ such that its dual image $\boldsymbol{\phi}^{*i} \sim \mathtt{P}^i\mathtt{Q}^*_\Phi\mathtt{P}^{i\top}$ satisfies:

$$\boldsymbol{\phi}^{*i} = \mathtt{D}^i\mathtt{D}^{i\top}, \quad \mathtt{D}^i \in \mathcal{D} \tag{3}$$

$$\text{and } \mathtt{Q}^*_\Phi \nsim \mathtt{Q}^*_\infty. \tag{4}$$

M is *critical for the affine reconstruction* if M is critical and $\mathbf{\Pi} \nsim \mathbf{\Pi}_\infty$. M is *critical for recovering the intrinsics* if M is critical and $\boldsymbol{\phi}^{*i} \nsim \boldsymbol{\omega}^{*i}$. Note that M is *critical for the Euclidean reconstruction* if it is critical for the affine reconstruction or for recovering the intrinsics.

3.2. Unknown *Constant* Focal Length

We now describe the generic singularities for $n \geq 2$ views of a camera, with all known intrinsics except a constant focal length α. We will refer to this case as the *unknown constant focal length* case or simply "our case".

For an unknown (possibly varying) focal length α, it has been shown [8, 18] that projection matrices can be considered as "calibrated" i.e., $\mathtt{P}^i = \mathtt{R}^{i\top}\left(\mathtt{I} \mid -\mathbf{t}^i\right)$, requiring that, in (3), $\mathcal{D}$ be the subgroup of diagonal matrices subject to condition (H1) below. Now, in our case, as α is supposedly constant, we will additionally require the relation (H2) to be satisfied:

- (H1) $D_{11} = D_{22}, \quad \mathtt{D} \in \mathcal{D}$;
- (H2) $\phi^{*i}_{11}/\phi^{*i}_{33} = \phi^{*j}_{22}/\phi^{*j}_{33}, \quad 1 \leq i, j \leq n.$

Clearly, critical motions for an unknown constant focal length belong to the set of critical motions for an unknown varying focal length as (H2) is simply an equivalence relation on $\mathcal{D}$.

Only virtual conics on finite planes have to be considered. For virtual conics on $\mathbf{\Pi}_\infty$, criticality is independent of camera positions and only depends on camera orientations [8, 18]. It has been shown that, under (H1), a motion is critical for the intrinsics but not for the affine structure if and only if all optical axes are parallel. Of course, it also holds under (H2).

Proposition 1 (Critical Camera Positions) *Let $\mathbf{\Phi}$ be a virtual conic on some* finite *plane $\mathbf{\Pi}$. In the unknown constant focal length case, a necessary condition for a motion sequence to be critical w.r.t. $\mathbf{\Phi}$ is that the camera centres be:*

- *(P1) at two different positions;*
- *(P2) at three/four distinct positions corresponding to the vertices of a right triangle/of a rectangle.*

Proof. As in [8, §8.3], we choose a Euclidean frame in which $\mathbf{\Phi}$ has, as supporting plane, $\mathbf{\Pi}$ with equation $z = 0$ and associated quadric $\mathtt{Q}^*_\Phi = \mathrm{diag}(d_1, d_2, 0, d_3)$, assuming $d_1 \geq d_2$ and $d_1, d_2, d_3 > 0$. Thus, $\mathtt{Q}^*_\Phi$ projects to:

$$\boldsymbol{\phi}^{i*} \sim \mathtt{P}^i\mathtt{Q}^*_\Phi\mathtt{P}^{i\top} = \mathtt{R}^{i\top}\mathbf{\Phi}^{*i}_\infty\mathtt{R}^i, \tag{5}$$
$$\text{where } \mathbf{\Phi}^{*i}_\infty = \mathrm{diag}(d_1, d_2, 0) + d_3\mathbf{t}^i\mathbf{t}^{i\top};$$

$\mathbf{\Phi}^i_\infty$ is the conic generated by intersecting $\mathbf{\Pi}_\infty$ with the ith projection cone of $\mathbf{\Phi}$, whose vertex is the camera centre $\mathbf{t}^i$.

Assume the position $\mathbf{t}^i$ to be critical w.r.t. $\mathbf{\Phi}$ i.e., equation (5) holds with (H1) and (H2) being satisfied, cf. §3.2. Hence, thanks to the spectral decomposition of $\mathbf{\Phi}^{*i}_\infty$, one infer from (H1) that two of its eigenvalues $\lambda^i_1, \lambda^i_2, \lambda^i_3$ are equal and, from (H2), the constraint $\lambda^i_1\lambda^i_2/(\lambda^i_3)^2 = \kappa^2$ for some fixed $\kappa > 0$. As said earlier, by (only) using the constraint resulting from (H1), the authors of [8, 18] obtain, as locus $\mathscr{L}_{(\mathrm{H1})}$ of critical positions, the union of two real central conics, one ellipse in the yz-plane and one hyperbola in the xz-plane. In the yz-plane, the obtained ellipse is represented by matrix $\mathtt{L}_1$, cf. (6); it is centered at the origin O with Oy and Oz as symmetry axes.

Now, only assume that (H2) holds. Let us determine what is the locus $\mathscr{L}_{(\mathrm{H2})}$ in the yz-plane, induced by applying the constraint resulting from (H2). We easily establish[2], that $\mathscr{L}_{(\mathrm{H2})}$ is a degenerate conic, with matrix $\mathtt{L}_2$, cf. (7), consisting of two distinct real parallel lines, being symmetric about the axis Oz.

$$\mathtt{L}_1 = \mathrm{diag}\left(d_1d_3, (d_1 - d_2)d_3, (d_2 - d_1)d_1\right), \tag{6}$$

$$\mathtt{L}_2 = \mathrm{diag}\left(0, d_2d_3, -d_1^2\kappa^2\right). \tag{7}$$

Since two conics meet at four points, we infer that the common points of $\mathtt{L}_1$ and $\mathtt{L}_2$ are, in our case, the four vertices of a rectangle, whose homogeneous yz-coordinates are:

$$[\pm\sqrt{(d_1 - d_2)(d_2 - d_1\kappa^2)}, \pm d_1\kappa, \sqrt{d_2d_3}]^\top. \tag{8}$$

[2] Due to lack of space, the details are omitted.

	Critical motions in the unknown constant focal length case	Self-calibration ambiguity
1	*All cameras* having parallel axes.	(A)
2	*Four cameras*, with centres defining a rectangle ; optical axes lie in the supporting plane of the rectangle and for each pair of cameras with adjacent centres, axes are symmetric about the symmetry axis of the rectangle separating the two centres.	(P)
3	*Three cameras*, out of the four cameras of case 2 (so defining a right triangle).	(P)
4	*Two cameras*, out of the four cameras of case 2, providing they have adjacent centres (optical axes intersect in a real finite point, with the centres being equidistant from this point).	(P)
5	*Two cameras* having coinciding axes.	(P)

Table 1. List of all critical motions for known skew, aspect ratio and principal point and unknown constant focal length α. Ambiguities are classified as (A) if $\mathbf{\Pi}_\infty$ can be recovered and (P) if neither α nor $\mathbf{\Pi}_\infty$ can be recovered.

As $d_1 \geq d_2$, all these are real providing $d_1 \geq d_2 \geq d_1\kappa^2$.

Similar results are obtained in the xz-plane, except that the four vertices must satisfy $d_1\kappa^2 \geq d_1 \geq d_2$ to be real. As the two above inequalities cannot simultaneously hold for non-zero coordinates, there are at most four (real) positions that can be critical.

A first special case (C1) must be considered if $d_1 = d_2$ i.e., if $\mathbf{\Phi}$ is a circle. In this case, $\mathtt{L}_1$ "degenerates"to the (repeated) Oz axis while $\mathtt{L}_2$ is unchanged: their intersection yields two distinct points, being symmetric about Oy. A second case (C2) is when $d_2/d_1 = \kappa^2$, for which (8) also reduces to the same two points. Similar results can be obtained in the xz-plane for exactly the *same* conditions. ■

3.2.1 Critical Camera Orientations

From each of the critical positions (P1) and (P2) listed in proposition 1, we now determine the critical camera orientations i.e., sufficient conditions for the motions to be critical.

Proposition 2 *In the case of all known intrinsics except a focal length, the critical orientations are the same whether the focal length be* varying *or* constant.

Proof. Only the "unsigned" directions of optical axes have to be considered. Indeed, if a camera orientation is critical, then it is also critical for any rotation around the optical axis or reversed direction. The study of critical motions in [8, 18] gives rise to two cases, according to whether $\mathbf{\Phi}$ is a circle, corresponding to the previously mentioned case (C1) in proof of proposition 1, or an ellipse. If $\mathbf{\Phi}$ is a circle, a direction is critical if and only if it is orthogonal to the supporting plane of $\mathbf{\Phi}$ for all $n - m$ cameras, with $m \in \{0, 1, 2\}$, and arbitrary for the m others. The configuration $m > 0$ happens when $\mathbf{\Phi}_\infty^{*i} \sim \mathtt{I}$, from which we infer that (H2) is also satisfied, cf. equation (5). If $\mathbf{\Phi}$ is an ellipse, the direction is critical if and only if it is parallel to tangents of any conic of $\mathscr{L}_{(\mathrm{H1})}$ lying on its supporting plane. Thus, the fact that directions are uniquely determined or in some configuration such that (H2) holds ends the proof. ■

By proposition 2, we inherit the results of [8, 18] and can adjust them to the unknown constant focal length case.

- Given critical positions (P1), critical orientations are:
  - either (O1a) arbitrary or
  - (O1b) such that optical axes coincide.
- Given (P2), they are (O2) such that the optical axes of any pair of adjacent cameras i.e., whose centres correspond to adjacent vertices of the rectangle, are symmetric about the symmetry axis of the rectangle separating their two centres while not being parallel; all four axes cannot intersect in a single real point.

Critical orientation (O1a) only applies to case (C2) given in the proof of proposition 1, which entails criticality only for affine structure.

3.2.2 Describing All Critical Camera Motions

All the critical motions in the unknown constant focal length case are listed in Table 1. Except for pure translations, the only motions that are likely to occur in practice are when centres correspond to only $p = 2$ different positions. At $2 < p \leq 4$ critical positions (coinciding with vertices of some right triangle or rectangle), some ambiguities can be removed by using cheirality invariants [6, chap. 21]. This is reassuring regarding our motivation to design a self-calibration algorithm with the fewest singularities.

4. A Guaranteed Solution Method

We describe how Interval Analysis (IA) and particularly IA-based Global Optimization finds the global optimum of nonlinear objective functions arising in computer vision problems. We do not give an extensive review of the background, which can be found in [4]. IA is an arithmetic of intervals such as $\mathbf{x} = [\underline{x}, \overline{x}]$, defined by two real bounds. One interest of IA is its ability of bounding the range of a function. An inclusion function $\mathbf{f}$ of a function f over a multidimensional interval $\mathbf{X} = ([\underline{x}_1, \overline{x}_1], ..., [\underline{x}_n, \overline{x}_n])$, called a box, is an interval $\mathbf{f} = [\underline{f}, \overline{f}]$ containing the range of f

over $\mathbf{X}$: $f(\mathbf{X}) \subseteq \mathbf{f}(\mathbf{X})$. By good "inclusion functions", we mean those with tight bounds around the range of f.

IA-based Global Optimization is a global deterministic guaranteed optimization method. From a general (nonlinear, non convex) objective function f to be minimized and from an initial box $\mathbf{X}_0$, it returns a box which encloses the global minimum of f, if it exists. It consists of: *(i)* A Branch and Bound algorithm to subdivide $\mathbf{X}_0$ into subboxes $\mathbf{X}_i$; *(ii)* An interval inclusion function $\mathbf{f}$ for any $\mathbf{X}_i$; *(iii)* Several tests to reject $\mathbf{X}_i$ if it does not contain the global minimum. Taking into account constraints in the parameter space is possible. Exhaustive examination of the whole solution space is a NP-hard problem. Therefore, good inclusion functions, allowing to reject boxes as early as possible in the optimization process, are required to achieve acceptable computational time. The latter varies from a few seconds to several days depending on $\mathbf{f}$.

Finding good inclusion functions is difficult. The difficulties arise from properties of IA such as: *(i) Subdistributivity:* If $\mathbf{x}, \mathbf{y}, \mathbf{z}$ are three intervals, then $\mathbf{x} \times (\mathbf{y} + \mathbf{z}) \subseteq \mathbf{x} \times \mathbf{y} + \mathbf{x} \times \mathbf{z}$; *(ii) Variable dependencies:* $\mathbf{x} - \mathbf{x} \neq 0$. If each variable appears only once in an inclusion function, then the thighest bounds can be easily formulated. Unfortunately, this case is unlikely to happen in real problems. *(iii) Wrapping:* A box is always mapped into a box, introducing pessimism on the bounds.

One possible inclusion function for a rational function f is the so-called natural extension. This is obtained by replacing the real variables by intervals in f. It is known that this solution gives pessimistic bounds. A better solution consists in rewriting this natural extension by factorizing variables and by avoiding repetitions to improve the bounds. Better inclusion functions can be computed using decompositions such as Taylor expansion but the related implementation is less straightforward.

Computational time may be reduced with several tricks. One of them consists in reducing the size of the initial box by propagating constraints, scaling the unknowns or using analytical derivatives of the objective function. Two major packages are freely available: GlobSol[3] and ALIAS[4]. GlobSol only needs a well rewritten natural extension and automatically differentiates it whereas ALIAS requires analytical bounds for the function and possibly for its derivatives.

The use of IA-based Global Optimization in computer vision is somewhat marginal. The self-calibration problem (five unknowns), based on the Kruppa equations, is treated in [3]. Computational times over one hour for a few images are reported. In [1], a guaranteed solution for the plane-based self-calibration problem, using a simplified camera model (three unknowns), is obtained in less than one minute.

The objective function we use is the sum of squares of the residuals corresponding to equations (2). Four unknowns are involved, namely the focal length and the plane at infinity. The initial box is set to a large domain (see section 5). Note that we could reduce this initial box by propagating cheirality constraints [6, chap. 21]. As an inclusion function, we used the natural extension of the equations and applied automatic factorizations with the Maple software. Experience has shown that factorization has to be done first by the focal length, then by the other unknowns. As an example, the factorized expression of the residual $\alpha^2 \boldsymbol{\omega}_{33}^{*j} - \boldsymbol{\omega}_{22}^{*j}$ is:

$$\begin{gathered}\alpha^2(\alpha^2(-(\mathrm{P}_{32}^j)^2 + p_1(-(\mathrm{P}_{34}^j)^2 p_1 + 2\mathrm{P}_{31}^j \mathrm{P}_{34}^j) + \\ p_2(-(\mathrm{P}_{34}^j)^2 p_2 + 2\mathrm{P}_{32}^j \mathrm{P}_{34}^j) - (\mathrm{P}_{31}^j)^2) + ((\mathrm{P}_{22}^j)^2 + \\ p_1((\mathrm{P}_{24}^j)^2 p_1 - 2\mathrm{P}_{21}^j \mathrm{P}_{24}^j) + p_2((\mathrm{P}_{24}^j)^2 p_2 - 2\mathrm{P}_{22}^j \mathrm{P}_{24}^j) + \\ (\mathrm{P}_{21}^j)^2 - (\mathrm{P}_{33}^j)^2 + p_3(-(\mathrm{P}_{34}^j)^2 p_3 + 2\mathrm{P}_{33}^j \mathrm{P}_{34}^j))) + \\ (\mathrm{P}_{23}^j)^2 + p_3((\mathrm{P}_{24}^j)^2 p_3 - 2\mathrm{P}_{23}^j \mathrm{P}_{24}^j),\end{gathered} \tag{9}$$

where $\boldsymbol{\Pi}_\infty = (p_1, p_2, p_3, 1)$. To assess the benefit of the rewriting, figure 1 shows the evaluation of two inclusion functions of this residual: the matricial form $\alpha^2 \boldsymbol{\omega}_{33}^{*j} - \boldsymbol{\omega}_{22}^{*j}$ and the expression (9). In this figure, the inclusion functions are evaluated on boxes which enclose exactly the true plane at infinity on boxes which enclose exactly the true plane at infinity and whose intervals for the focal length are 100 pixels wide. The bounds obtained from the factorized inclusion function are significally tighter than the bounds without rewriting. This effect is amplified when we consider the square of the residuals and when the parameters of the plane at infinity are also considered as variables in the box. Our implementation uses the GlobSol package with automatic differentiation.

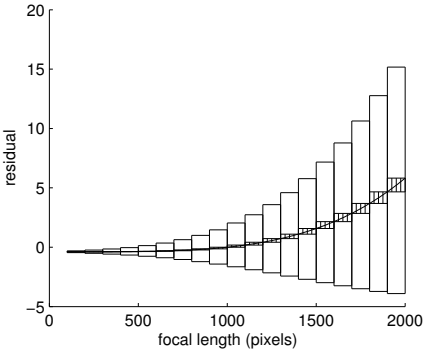


Figure 1. Examples of bounds of two inclusion functions of a residual used in the objective function, without factorization (empty boxes) and with factorization (hatched boxes).

[3] http://interval.louisiana.edu/GlobSol

[4] http://www-sop.inria.fr/coprin/logiciels/ALIAS

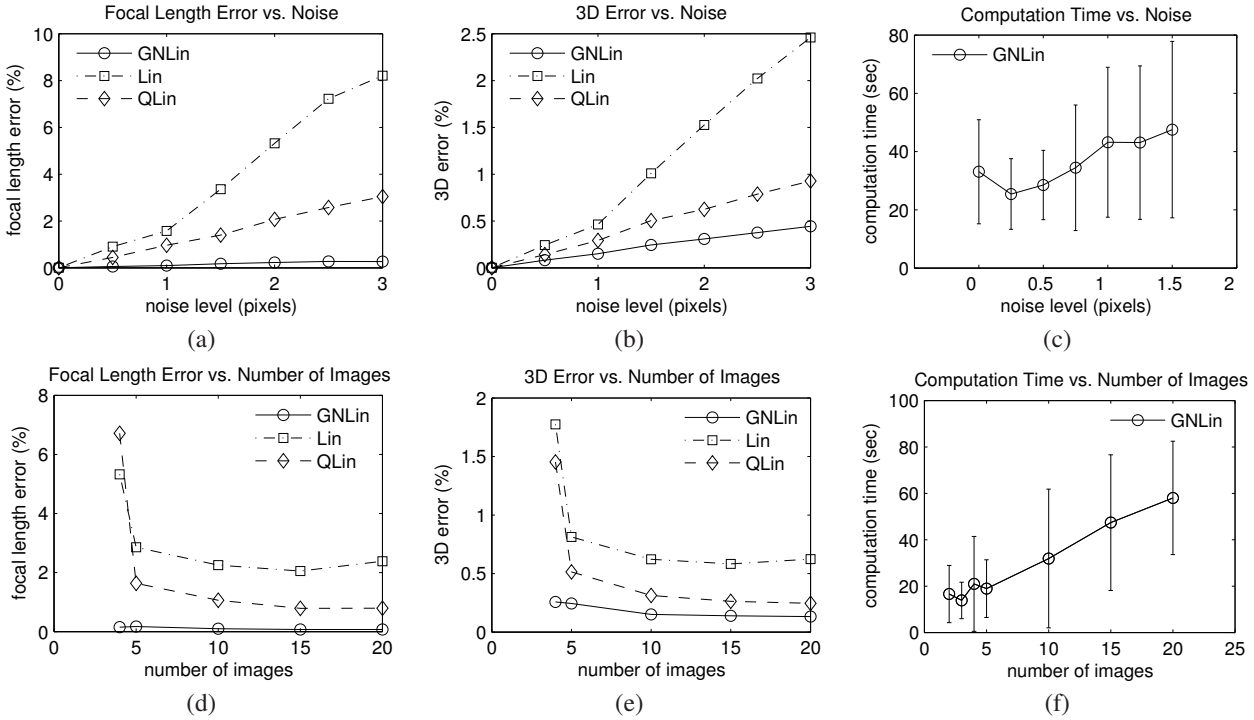


Figure 2. Mean relative error on the focal length (a), mean 3D error (b) and computation time (mean and standard deviation) (c) for varying noise level and 10 images. (d), (e), (f): Same criteria for varying number of images and a 1 pixel noise level. Squares stand for Lin, diamonds for QLin and circles for GNLin.

5. Results

Our approach has been validated on synthetic and real data. A software implementation is available online[5].

5.1. Synthetic Data

The experimental setup is composed of 100 points randomly distributed in a 3D sphere of unit radius. The 3D points are seen in n images with side length 256 pixels. Cameras have a fixed focal length of 1000 pixels, square pixels and principal point fixed at the image centre. They are randomly placed at a mean distance of 2.5 ± 0.25 units from the scene origin. Each camera fixates a random point located in a sphere of 0.1 unit radius centred at the origin. Gaussian noise is added to the 2D projected points. Projective bundle adjustment is used to compute noise contaminated cameras. Cameras are standardized so that the focal length is scaled around unity. We compared the linear method (Lin), the quasi-linear method (QLin), described in section 2, and our approach, the guaranteed nonlinear method (GNLin). The initial search box is set to $[100, 10000]$ for the focal length and $[-10^9, 10^9]^3$ for the plane at infinity. We measure the mean relative error on the focal length and the mean 3D error. The 3D error is the mean distance between the true 3D points and the recontructed 3D points, obtained from Euclidean rectification and alignment. It is expressed as a percentage of the scene size. All results are averages over 50 trials.

In the first experiment, n is fixed to 10 images and the noise level is varied from 0 to 3 pixels. As expected, the 3D error and the error on the focal length (figures 2.a and 2.b) are lower for GNLin than for Lin and QLin. The reason is that the rank and the positive semi-definiteness condition are implicitly enforced in GNLin. For all methods, the error increases linearly with the noise level. For clarity reasons, error bars are not displayed on the figures. The standard deviation on the focal length error, for a 3 pixel noise, are 5% for Lin, 2.5% for QLin and 0.3% for GNLin.

In the second experiment, the noise level is fixed to 1 pixel and n is varied from 4 to 20 images. Again, GNLin has the lowest error (figures 2.d and 2.e). We can also see that a few images are sufficient to obtain a good result. Although the computation time (figures 2.c and 2.f) required to find the global minimum is significantly larger than for the other methods (less than a second), it is reasonable. It increases linearly against the number of images while being weakly dependent on noise.

In order to see what happens near artificial degeneracies arising in Lin and GLin, we tested the following camera

[5] http://www.irit.fr/~Benoit.Bocquillon

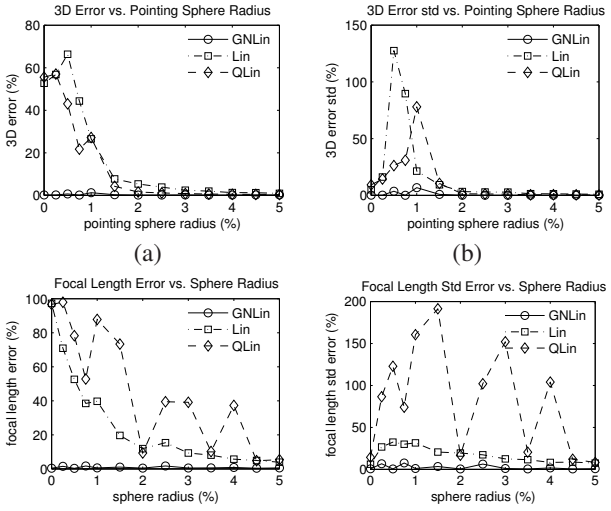


Figure 3. (a) and (b): Mean relative 3D error and standard deviation for a motion near the "fixating cameras" singularity. (c) and (d): Mean relative error on the focal length and standard deviation for a motion near the "aligned optical axes" singularity. All the distances are expressed as a percentage of the scene size. Squares stand for Lin, diamonds for QLin and circles for GNLin.

motions.

Fixating cameras. Cameras are placed as previously described except that all their optical centres lie on a plane. Each camera fixates a random point in a sphere of radius r centred at the origin. The singularity occurs at $r = 0$. Figures 3.a and 3.b show that the 3D error is important for $r < 2\%$ of the scene size. Our algorithm is not affected by the singularity, finding the correct answer in all cases.

Aligned optical axes. All the optical centres are aligned at different positions. All cameras look at the origin, except for one camera whose orientation is arbitrary. To move away from this singularity, cameras look in a sphere of radius r_1 and the optical centres are located in a sphere of radius r_2 centred on the critical position. Due to lack of space, we show results for $r_1 = r_2 = r$ only. Mean relative error on the focal length and standard deviation are reported in figures 3.c and 3.d. The error for Lin and QLin are quite important below 5% of the scene size. QLin is very unstable. GNLin is not affected by the singularity.

5.2. Real data

We took 4 images, shown on figure 4, by moving around a building, so that the motion happens to be near the "fixating camera", a critical motion for Lin and QLin. We semi-automatically detected and matched 63 interest points lying in the two dominant planes of the scene. To get the projective cameras, we used Sturm-Triggs projective factorization [20], followed by projective bundle adjustment. Lin and QLin failed in the sense that they gave meaningless solutions. GNLin converged within 17 seconds on a 1.7GHz Core Duo computer and found a 3604 pixels focal length. From this focal length and the retrieved plane at infinity, a Euclidean rectification was made on the 3D points obtained from the projective reconstruction. Figure 4.e shows a top view of the 3D point cloud. The ratio between the two wall lengths was evaluated to 1.36, obtained from real measurements, whereas our reconstruction has a ratio of 1.35. In the same way, the angle between the two reconstructed planes is equal to 91.7 degrees. We tried to add other images to the sequence to get off the singularity: Lin and QLin have remained very unstable, often failing or giving bad reconstructions in terms of distance ratio and angle (see figure 4.e). Therefore, these methods cannot be used in practice for this kind of motion. On the contrary, GNLin has always given an acceptable solution.

6. Conclusion and Future Work

This paper has addressed two issues associated with metric reconstruction in the case of all known intrinsic parameters except a constant focal length. First, we described the critical motion sequences for that case. Second, we proposed a self-calibration algorithm which does not introduce artificial singularities. It is based on Interval Analysis Global Optimization and is able to find the guaranteed solution within few seconds. We also noticed that the linear or quasi-linear methods are very unstable, even away from singular cases. These results show that such methods are sometimes difficult to use in practice.

Acknowledgment. The authors would like to thank Peter Sturm for very fruitful discussions.

References

[1] B. Bocquillon, P. Gurdjos, and A. Crouzil. Towards a guaranteed solution to plane-based self-calibration. In *ACCV*, volume 1, pages 11–20, Hyderabad, India, Jan. 2006.

[2] O. Faugeras, Q.-T. Luong, and S. J. Maybank. Camera Self-Calibration: Theory and Experiments. In *ECCV*, pages 321–334, Santa Margherita Ligure, Italy, May 1992.

[3] A. Fusiello, A. Benedetti, M. Farenzena, and A. Busti. Globally Convergent Autocalibration Using Interval Analysis. *PAMI*, 26(12):1633–1638, Dec. 2004.

[4] E. R. Hansen and G. W. Walster. *Global Optimization Using Interval Analysis*. Second ed. Marcel Dekker, 2003.

[5] R. Hartley, E. Hayman, L. de Agapito, and I. Reid. Camera calibration and the search for infinity. In *ICCV*, volume 1, pages 510–517, Kerkyra, Greece, Sept. 1999.

[6] R. Hartley and A. Zisserman. *Multiple View Geometry*. Second ed. Cambridge University Press, 2003.

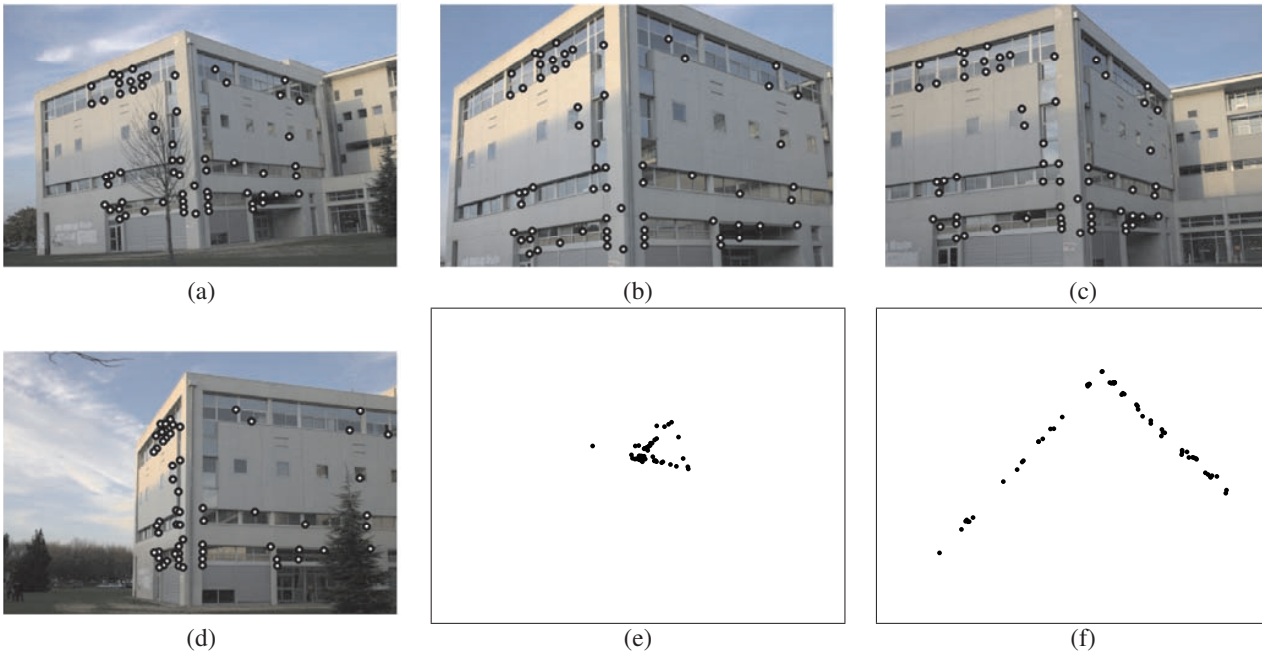


Figure 4. (a) to (d): Four images of a building with interest points. (e): An example of bad reconstruction, obtained with Lin. (f): A top view of the 63 3D points reconstructed with GNLin.

[7] A. Heyden and K. Åström. Euclidean Reconstruction from Constant Intrinsic Parameters. In *ICPR*, volume 1, pages 339–343, Vienna, Austria, Aug. 1996.

[8] F. Kahl. *Geometry and Critical Configurations of Multiple Views*. PhD thesis, Lund Institute of Technology, Sweden, 2001.

[9] F. Kahl and B. Triggs. Critical Motions in Euclidean Structure from Motion. In *CVPR*, volume 2, pages 2366–2372, Fort Collins, Colorado, USA, June 1999.

[10] Q.-T. Luong and O. Faugeras. Self-calibration of a moving camera from point correspondences and fundamental matrices. *IJCV*, 22(3):261–289, Mar. 1997.

[11] M. Pollefeys, R. Koch, and L. Van Gool. Self-Calibration and Metric Reconstruction in spite of Varying and Unknown Internal Camera Parameters. In *ICCV*, pages 90–95, Bombay, India, Jan. 1998.

[12] M. Pollefeys and L. Van Gool. Stratified Self-Calibration with the Modulus Constraint. *PAMI*, 21(8):707–724, Jan. 1999.

[13] M. Pollefeys and L. Van Gool. Some issues on self-calibration and critical motion sequences. In *ACCV*, pages 893–898, Taipei, Taiwan, Jan. 2000.

[14] J. Semple and G. Kneebone. *Algebraic Projective Geometry*. Oxford Classic Series, Clarendon Press, 1952, reprinted, 1998.

[15] P. Sturm. Critical Motion Sequences for Monocular Self-Calibration and Uncalibrated Euclidean Reconstruction. In *CVPR*, pages 1100–1105, San Juan, Puerto Rico, June 1997.

[16] P. Sturm. *Vision 3D non calibrée : contributions à la reconstruction projective et étude des mouvements critiques pour l'auto-calibrage*. PhD thesis, Institut National Polytechnique de Grenoble, France, Dec. 1997.

[17] P. Sturm. A case against Kruppa's equations for camera self-calibration. *PAMI*, 22(10):1199–1204, Sept. 2000.

[18] P. Sturm. Critical Motion Sequences for the Self-Calibration of Cameras and Stereo Systems with Variable Focal Length. *IVC*, 20(5-6):415–426, Mar. 2002.

[19] P. Sturm, Z. Cheng, P. C. Y. Chen, and A. N. Poo. Focal length calibration from two views: Method and analysis of singular cases. *CVIU*, 99(1):58–95, July 2005.

[20] B. Triggs. Factorization Methods for Projective Structure and Motion. In *CVPR*, pages 845–851, San Francisco, California, USA, June 1996.

[21] B. Triggs. Autocalibration and the absolute quadric. In *CVPR*, pages 609–614, San Juan, Puerto Rico, June 1997.

8.1.3 Paper (EMMCVPR'05) – *Handling Missing Data in the Computation of 3D Affine Transformations*

Handling Missing Data in the Computation of 3D Affine Transformations

H. Martinsson[1], A. Bartoli[2], F. Gaspard[1], and J-M. Lavest[2]

[1] CEA LIST – LIST/DTSI/SARC/LCEI Bât 528, 91 191 Gif sur Yvette, France
tel: +33(0)1 69 08 82 98, fax: +33(0)1 69 08 83 95
hanna.martinsson@cea.fr
[2] LASMEA (CNRS / UBP) – 24 avenue des Landais, 63 177 Aubière, France
tel: +33(0)4 73 40 76 61, fax: +33(0)4 73 40 72 62
adrien.bartoli@univ-bpclermont.fr

Abstract. The reconstruction of rigid scenes from multiple images is a central topic in computer vision. Approaches merging partial 3D models in a hierarchical manner have proven the most effective to deal with large image sequences. One of the key building blocks of these hierarchical approaches is the alignment of two partial 3D models, which requires to express them in the same 3D coordinate frame by computing a 3D transformation. This problem has been well-studied for the cases of 3D models obtained with calibrated or uncalibrated pinhole cameras.
We tackle the problem of aligning 3D models – sets of 3D points – obtained using uncalibrated affine cameras. This requires to estimate 3D affine transformations between the models. We propose a factorization-based algorithm estimating simultaneously the aligning transformations and corrected points, exactly matching the estimated transformations, such that the reprojection error over all cameras is minimized. In the case of incomplete image data our algorithm uses an Expectation Maximization (EM) based scheme that alternates prediction of the missing data and estimation of the affine transformation.
We experimentally compare our algorithm to other methods using simulated and real data.

1 Introduction

Threedimensional reconstruction from multiple images of a rigid scene, often dubbed Structure-From-Motion (SFM), is one of the most studied problems in computer vision. The difficulties come from the fact that, using only feature correspondences, both the 3D structure of the scene and the cameras have to be computed. Most approaches rely on an initialisation phase optionally followed by self-calibration and bundle adjustment. Existing initialisation algorithms can be divided into three families, namely *batch*, *sequential* and *hierarchical* processes. Hierarchical processes [1] have proven the most successful for large image sequences. Indeed, batch processes such as the factorization algorithms [2] which reconstruct all features and cameras in a single computation step, do not easily

handle occlusions, while sequential processes such as [3] which reconstruct each view on turn, may typically suffer from accumulation of the errors. Hierarchical processes merge partial 3D models obtained from sub-sequences, which allows to distribute the error over the sequence, and efficiently handle open and closed sequences. A key step of hierarchical processes is the fusion or the *alignment* of partial 3D models, which is done by *computing 3D motion from 3D feature correspondences*. This problem has been extensively studied in the projective [4,1] and the metric and Euclidean [5] cases.

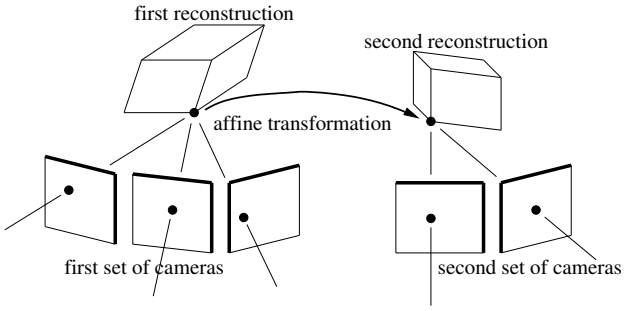


Fig. 1. The problem tackled in this paper is the Maximum Likelihood Estimation of 3D affine transformations between two affine reconstructions obtained from uncalibrated affine cameras.

We focus on the affine camera model [6], which is a reasonable approximation to the perspective camera model when the depth of the observed scene is small compared to the viewing distance. In this case, the partial 3D models obtained from sub-sequences, *i.e.* multiple subsets of cameras, are related by 3D affine transformations. We deal with the computation of such transformations from point correspondences, as illustrated on Fig. 1. We propose a Maximum Likelihood Estimator based on factorizing modified image point coordinates. We compute a 3D affine transformation and a set of 3D point correspondences which perfectly match, such that *the reprojection error in all sets of cameras is minimized*. It is intended to fit in hierarchical affine SFM processes of which the basic reconstruction block is, *e.g.* the affine factorization [2]. Our method does not make any assumption about the cameras, besides the fact that a reconstruction of each camera set using an affine camera model has been performed. The method relies on the important new concept of *orthonormal bases*. In the occlusion-free case, our algorithm needs one Singular Value Decomposition (SVD). However, in the case of incomplete measurement data, *i.e.* when some of the 3D points used for the alignment are not visible in all views, the factorization algorithm must be extended. We propose an Expectation-Maximization (EM) based scheme. The Expectation step predicts the missing data while the Maximization step maximizes the log likelihood.

We proposed the Maximum Likelihood Estimator in the case of complete data in [7]. The contribution of this paper with respect to the former one resides in the handling of missing data. We have also completed the experiments.

This paper is organized as follows. We give our notation and preliminaries in Sect. 2. In Sect. 3, we review the factorization approach to uncalibrated affine Structure-From-Motion. Our alignment method is described in Sect. 4, while other methods are summarized in Sect. 5. Experimental results are reported in Sect. 6. Our conclusions are given in Sect. 7.

2 Notation and Preliminaries

Vectors are typeset using bold fonts, *e.g.* $\mathbf{x}$, and matrices using sans-serif, calligraphic and greek fonts, *e.g.* A, $\mathcal{Q}$ and Λ. We do not use homogeneous coordinates, *i.e.* image point coordinates are 2-vectors: $\mathbf{x}^\mathsf{T} = (x \;\; y)$, where $^\mathsf{T}$ is transposition. The different sets of cameras are indicated with primes, *e.g.* P_1 and P'_1 are the first cameras of the camera sets. Index $i = 1 \dots n$ is used for the cameras of a camera set and index $j = 1 \dots m$ is used for the 3D points. The mean vector of a set of vectors, say $\{\mathbf{Q}_j\}$, is denoted $\bar{\mathbf{Q}}$. The Moore-Penrose pseudoinverse of matrix A is denoted $\mathsf{A}^\dagger$.

Let $\mathbf{Q}_j$ be a 3-vector and $\mathbf{x}_{ij}$ a 2-vector representing respectively a 3D and an image point. The uncalibrated affine camera is modeled by a (2×3) matrix P_i and a (2×1) translation vector $\mathbf{t}_i$, giving the projection equation

$$\mathbf{x}_{ij} = \mathsf{P}_i \mathbf{Q}_j + \mathbf{t}_i \;\; . \tag{1}$$

Calligraphic fonts are used for the measurement matrices, *e.g.*

$$\mathcal{X}_{(2n \times m)} = \begin{pmatrix} \mathcal{Y}_1 \cdots \mathcal{Y}_m \end{pmatrix} \qquad \text{and} \qquad \mathcal{Y}_j = \begin{pmatrix} \mathbf{x}_{1j}{}^\mathsf{T} \cdots \mathbf{x}_{nj}{}^\mathsf{T} \end{pmatrix}^\mathsf{T} \;\; ,$$

where $\mathcal{Y}_j$ contains all the measured image coordinates for the j-th point. The $(2n \times 3)$ 'joint projection' and $(3 \times m)$ 'joint structure' matrices are defined by

$$\mathcal{P} = \begin{pmatrix} \mathsf{P}_1{}^\mathsf{T} \cdots \mathsf{P}_n{}^\mathsf{T} \end{pmatrix}^\mathsf{T} \qquad \text{and} \qquad \mathcal{Q} = \begin{pmatrix} \mathbf{Q}_1 \cdots \mathbf{Q}_m \end{pmatrix} \;\; .$$

We assume that the noise on the image point positions has a Gaussian centered distribution and is i.i.d. Under these hypotheses, minimizing the reprojection error yields Maximum Likelihood Estimates.

3 Structure-From-Motion Using Factorization

Given a set of point matches $\{\mathbf{x}_{ij}\}$, the factorization algorithm is employed to recover all cameras $\{\hat{\mathsf{P}}_i, \hat{\mathbf{t}}_i\}$ and 3D points $\{\hat{\mathbf{Q}}_j\}$ at once [2]. Under the aforementioned hypotheses on the noise distribution, this algorithm computes Maximum Likelihood Estimates [8] by minimizing the reprojection error

$$\min_{\hat{\mathcal{P}}, \hat{\mathcal{Q}}, \{\hat{\mathbf{t}}_i\}} \mathcal{R}^2(\hat{\mathcal{P}}, \hat{\mathcal{Q}}, \{\hat{\mathbf{t}}_i\}) \quad \text{with} \quad \mathcal{R}^2(\mathcal{P}, \mathcal{Q}, \{\mathbf{t}_i\}) = \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} d^2(\mathbf{x}_{ij}, \mathsf{P}_i \mathbf{Q}_j + \mathbf{t}_i) \;\; , \tag{2}$$

where $d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|$ is the Euclidean distance between $\mathbf{x}$ and $\mathbf{y}$.

Step 1: Computing the translation. Given the uncalibrated affine projection (1), the first step of the algorithm is to compute the translation $\hat{\mathbf{t}}_i$ of each camera in order to cancel it out from the projection equation. This is achieved by nullifying the partial derivatives of the reprojection error (2) with respect to $\hat{\mathbf{t}}_i$: $\frac{\partial \mathcal{R}^2}{\partial \hat{\mathbf{t}}_i} = 0$. A short calculation shows that if we fix the arbitrary centroid of the 3D points to the origin, then $\hat{\mathbf{t}}_i = \bar{\mathbf{x}}_i$. Each set of image points is therefore centered on its centroid, *i.e.* $\mathbf{x}_{ij} \leftarrow \mathbf{x}_{ij} - \bar{\mathbf{x}}_i$, to obtain *centered coordinates*: $\mathbf{x}_{ij} = \mathsf{P}_i \mathbf{Q}_j$.

Step 2: Factorizing. The problem is reformulated as

$$\min_{\hat{\mathcal{P}}, \hat{\mathcal{Q}}} \mathcal{R}^2(\hat{\mathcal{P}}, \hat{\mathcal{Q}}) \qquad \text{with} \qquad \mathcal{R}^2(\mathcal{P}, \mathcal{Q}) = \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} d^2(\mathbf{x}_{ij}, \mathsf{P}_i \mathbf{Q}_j) \ .$$

The reprojection error can be rewritten by gathering the terms using the measurement, the 'joint projection' and the 'joint structure' matrices as

$$\mathcal{R}^2(\mathcal{P}, \mathcal{Q}) \propto \|\mathcal{X} - \mathcal{P}\mathcal{Q}\|^2 \ ,$$

and the problem is solved by computing the Singular Value Decomposition [9] of matrix $\mathcal{X}$, $\mathcal{X}_{2n \times m} = \mathsf{U}_{2n \times m} \mathbf{\Sigma}_{m \times m} \mathsf{V}^{\mathsf{T}}_{m \times m}$. Let $\mathbf{\Sigma} = \mathbf{\Sigma}_u \mathbf{\Sigma}_v$ be any decomposition of matrix $\mathbf{\Sigma}$. The motion and structure are obtained by 'truncating' the decomposition or nullifying all but the 3 first singular values, which leads to

$$\mathcal{P} = \psi(\mathsf{U}\mathbf{\Sigma}_u) \qquad \text{and} \qquad \mathcal{Q} = \psi^{\mathsf{T}}(\mathsf{V}\mathbf{\Sigma}_v^{\mathsf{T}}) \ ,$$

where $\psi(\mathsf{W})$ returns the matrix formed with the 3 leading columns of matrix W. Note that the solution $\mathcal{P} = \psi(\mathsf{U})$ and $\mathcal{Q} = \psi^{\mathsf{T}}(\mathsf{V}\mathbf{\Sigma})$ has the property $\mathcal{P}^{\mathsf{T}}\mathcal{P} = \mathrm{I}$, which is useful for our alignment method, see Sect. 4.

The 3D model is obtained up to a global affine transformation. Indeed, for any (3×3) invertible matrix B,

$$\tilde{\mathcal{P}} = \hat{\mathcal{P}}\mathsf{B} \qquad \text{and} \qquad \tilde{\mathcal{Q}} = \mathsf{B}^{-1}\hat{\mathcal{Q}} \tag{3}$$

give the same reprojection error that $\mathcal{P}$ and $\mathcal{Q}$ since $\mathcal{R}^2(\tilde{\mathcal{P}}, \tilde{\mathcal{Q}}) = \|\mathcal{X} - \tilde{\mathcal{P}}\tilde{\mathcal{Q}}\| = \|\mathcal{X} - \hat{\mathcal{P}}\mathsf{B}\mathsf{B}^{-1}\hat{\mathcal{Q}}\|^2 = \|\mathcal{X} - \mathcal{P}\mathcal{Q}\|^2 = \mathcal{R}^2(\mathcal{P}, \mathcal{Q})$.

As presented above, the factorization algorithm do not handle occlusions. Though some algorithms have been proposed, see *e.g.* [10], they are not appropriate for Structure-From-Motion from large image sequences.

4 Alignment of 3D Affine Reconstructions

We formally state the alignment problem in the two camera set case and present our algorithm, dubbed 'FACTMLE-EM'.

4.1 Problem Statement

Consider two sets of cameras $\{(\mathsf{P}_i, \mathbf{t}_i)\}_{i=1}^{n}$ and $\{(\mathsf{P}'_i, \mathbf{t}'_i)\}_{i=1}^{n'}$ and associated structures $\{\mathbf{Q}_j \leftrightarrow \mathbf{Q}'_j\}_{j=1}^{m}$ obtained by reconstructing a rigid scene using *e.g.* the above-described factorization algorithm. Without loss of generality, we take $n = n'$ and the reprojection error over two sets is given by

$$\mathcal{C}^2(\mathcal{Q}, \mathcal{Q}') = \frac{1}{2nm}\left(\mathcal{R}^2(\mathcal{P}, \mathcal{Q}, \{\mathbf{t}_i\}) + \mathcal{R}'^2(\mathcal{P}', \mathcal{Q}', \{\mathbf{t}'_i\})\right) \ . \tag{4}$$

Letting $(\hat{\mathsf{A}}, \hat{\mathbf{t}})$ represent the aligning (3×3) affine transformation, the Maximum Likelihood Estimator is formulated by

$$\min_{\hat{\mathcal{Q}}, \hat{\mathcal{Q}}'} \mathcal{C}^2(\hat{\mathcal{Q}}, \hat{\mathcal{Q}}') \qquad \text{s.t.} \qquad \hat{\mathbf{Q}}'_j = \hat{\mathsf{A}}\hat{\mathbf{Q}}_j + \hat{\mathbf{t}} \ . \tag{5}$$

4.2 A Factorization-Based Algorithm

Our method to solve problem (5) uses a three-step factorization strategy. We first describe it in the occlusion-free case and then propose an iterative extension for the missing data case.

Step 1: Orthonormalizing. We propose the important concept of *orthonormal bases*. We define a reconstruction to be in an orthonormal basis if the joint projection matrix is column-orthonormal. Given a joint projection matrix $\mathcal{P}$, one can find a 3D affine tranformation represented by the (3×3) matrix N, which applies as B in (3), such that $\mathcal{P}\mathsf{N}$ is column-orthonormal, *i.e.* such that $\mathsf{N}^\mathsf{T}\mathcal{P}^\mathsf{T}\mathcal{P}\mathsf{N} = \mathrm{I}_{(3\times3)}$. We call the transformation N an *orthonormalizing transformation*. The set of orthonormalizing tranformations is 3-dimensional since for any 3D rotation matrix U, $\mathsf{N}\mathsf{U}$ still is an orthonormalizing transformation for $\mathcal{P}$. We use the QR decomposition $\mathcal{P} = \mathsf{Q}\mathsf{R}$, see *e.g.* [9], giving an upper triangular orthonormalizing transformation $\mathsf{N} = \mathsf{R}^{-1}$. Other choices are possible for computing an N, *e.g.* if $\mathcal{P} = \mathsf{U}\mathbf{\Sigma}\mathsf{V}^\mathsf{T}$ is an SVD of $\mathcal{P}$, then $\mathsf{N} = \mathsf{V}\mathbf{\Sigma}^{-1}$ has the required property. Henceforth, we assume that all 3D models are expressed in orthonormal bases

$$\begin{cases} \mathcal{P} \leftarrow \mathcal{P}\mathsf{N} \\ \mathcal{P}' \leftarrow \mathcal{P}'\mathsf{N}' \end{cases} \qquad \text{and} \qquad \begin{cases} \mathcal{Q} \leftarrow \mathsf{N}^{-1}\mathcal{Q} \\ \mathcal{Q}' \leftarrow \mathsf{N}'^{-1}\mathcal{Q}' \end{cases} \ .$$

An interesting property of orthonormal bases is that $\mathcal{P}^\dagger = \mathcal{P}^\mathsf{T}$. Hence, triangulating points in these bases is simply done by $\mathcal{Q} = \mathcal{P}^\mathsf{T}\mathcal{X}$.

Note that the matrix $\mathcal{P}$ computed by factorization, see Sect. 3, may already satisfy $\mathcal{P}^\mathsf{T}\mathcal{P} = \mathrm{I}$. However, if at least one of the cameras is not used for the alignment, *e.g.* if none of the 3D point correspondences project in this camera, or if the cameras come as the result of the alignment of partial 3D models, then $\mathcal{P}$ will *not* satisfy $\mathcal{P}^\mathsf{T}\mathcal{P} = \mathrm{I}$, thus requiring the orthonormalization step.

Step 2: Eliminating the translation. The translation part of the sought-after transformation can not be computed directly, but can be eliminated from the equations. First, center the image points to eliminate the translation part of the cameras: $\mathbf{x}_{ij} \leftarrow \mathbf{x}_{ij} - \mathbf{t}_i$ and $\mathbf{x}'_{ij} \leftarrow \mathbf{x}'_{ij} - \mathbf{t}'_i$. Second, consider that the partial derivatives of the reprojection error (4) with respect to $\hat{\mathbf{t}}$ must vanish: $\frac{\partial \mathcal{C}^2}{\partial \hat{\mathbf{t}}} = 0$. By using the constraint $\hat{\mathbf{Q}}'_j = \hat{\mathsf{A}}\hat{\mathbf{Q}}_j + \hat{\mathbf{t}}$ from (4) and expanding using (4), we get

$$\begin{aligned}
\sum_{i=1}^{n'} \sum_{j=1}^{m} \left(\mathsf{P}_i'^{\mathsf{T}} \mathsf{P}'_i \hat{\mathbf{t}} - \mathsf{P}_i'^{\mathsf{T}} \mathbf{x}'_{ij} + \mathsf{P}_i'^{\mathsf{T}} \mathsf{P}'_i \hat{\mathsf{A}} \hat{\mathbf{Q}}_j \right) &= 0 \\
\sum_{j=1}^{m} \left(\mathcal{P}'^{\mathsf{T}} \mathcal{P}' \hat{\mathbf{t}} - \mathcal{P}'^{\mathsf{T}} \mathcal{Y}'_j + \mathcal{P}'^{\mathsf{T}} \mathcal{P}' \hat{\mathsf{A}} \hat{\mathbf{Q}}_j \right) &= 0 \\
m \mathcal{P}'^{\mathsf{T}} \mathcal{P}' \hat{\mathbf{t}} - m \mathcal{P}'^{\mathsf{T}} \bar{\mathcal{Y}}' + m \mathcal{P}'^{\mathsf{T}} \mathcal{P}' \hat{\mathsf{A}} \bar{\hat{\mathcal{Q}}} &= 0 \ ,
\end{aligned}$$

which leaves us with $\hat{\mathbf{t}} = (\mathcal{P}'^{\mathsf{T}} \mathcal{P}')^{-1} (\mathcal{P}'^{\mathsf{T}} \bar{\mathcal{Y}}' - \mathcal{P}'^{\mathsf{T}} \mathcal{P}' \hat{\mathsf{A}} \bar{\hat{\mathcal{Q}}})$ that, thanks to the orthonormal basis property $\mathcal{P}'^{\dagger} = \mathcal{P}'^{\mathsf{T}}$, further simplifies to

$$\hat{\mathbf{t}} = \mathcal{P}'^{\mathsf{T}} \bar{\mathcal{Y}}' - \hat{\mathsf{A}} \bar{\hat{\mathcal{Q}}} \ . \tag{6}$$

Note that if the same entire sets of reconstructed points are used for the alignment, then we directly obtain $\hat{\mathbf{t}} = \mathbf{0}$ since $\bar{\mathcal{Y}}' = \mathbf{0}$ and $\bar{\hat{\mathcal{Q}}} = \mathbf{0}$. This is rarely the case in practice, especially if the alignment is used to merge partial 3D models.

Third, consider that the m partial derivatives of the reprojection error (4) with respect to each $\hat{\mathbf{Q}}_j$ must vanish as well: $\frac{\partial \mathcal{C}^2}{\partial \hat{\mathbf{Q}}_j} = 0$, and expand as above

$$\begin{aligned}
\sum_{i=1}^{n} \left(\mathsf{P}_i^{\mathsf{T}} \mathsf{P}_i \hat{\mathbf{Q}}_j - \mathsf{P}_i^{\mathsf{T}} \mathbf{x}_{ij} \right) + \sum_{i=1}^{n'} \left(\hat{\mathsf{A}}^{\mathsf{T}} \mathsf{P}_i'^{\mathsf{T}} \mathsf{P}'_i \hat{\mathsf{A}} \hat{\mathbf{Q}}_j - \hat{\mathsf{A}}^{\mathsf{T}} \mathsf{P}_i'^{\mathsf{T}} \mathbf{x}'_{ij} + \hat{\mathsf{A}}^{\mathsf{T}} \mathsf{P}_i'^{\mathsf{T}} \mathsf{P}'_i \hat{\mathbf{t}} \right) &= 0 \\
\mathcal{P}^{\mathsf{T}} \mathcal{P} \hat{\mathbf{Q}}_j - \mathcal{P}^{\mathsf{T}} \mathcal{Y}_j + \hat{\mathsf{A}}^{\mathsf{T}} \mathcal{P}'^{\mathsf{T}} \mathcal{P}' \hat{\mathsf{A}} \hat{\mathbf{Q}}_j - \hat{\mathsf{A}}^{\mathsf{T}} \mathcal{P}'^{\mathsf{T}} \mathcal{Y}'_j + \hat{\mathsf{A}}^{\mathsf{T}} \mathcal{P}'^{\mathsf{T}} \mathcal{P}' \hat{\mathbf{t}} &= 0 \ .
\end{aligned}$$

The sum over j of all these derivatives also vanishes, giving

$$\mathcal{P}^{\mathsf{T}} \mathcal{P} \bar{\hat{\mathcal{Q}}} - \mathcal{P}^{\mathsf{T}} \bar{\mathcal{Y}} + \hat{\mathsf{A}}^{\mathsf{T}} \mathcal{P}'^{\mathsf{T}} \mathcal{P}' \hat{\mathsf{A}} \bar{\hat{\mathcal{Q}}} - \hat{\mathsf{A}}^{\mathsf{T}} \mathcal{P}'^{\mathsf{T}} \bar{\mathcal{Y}}' + \hat{\mathsf{A}}^{\mathsf{T}} \mathcal{P}'^{\mathsf{T}} \mathcal{P}' \hat{\mathbf{t}} = 0 \ .$$

By replacing $\hat{\mathbf{t}}$ by its expression (6), and after some minor algebraic manipulations, we obtain

$$\mathcal{P}^{\mathsf{T}} \mathcal{P} \bar{\hat{\mathcal{Q}}} - \mathcal{P}^{\mathsf{T}} \bar{\mathcal{Y}} = 0 \quad \Longrightarrow \quad \bar{\hat{\mathcal{Q}}} = \mathcal{P}^{\dagger} \bar{\mathcal{Y}} \tag{7}$$

and by substituting in (6) and using the orthonormal basis property, we get

$$\hat{\mathbf{t}} = \mathcal{P}'^{\mathsf{T}} \bar{\mathcal{Y}}' - \hat{\mathsf{A}} \mathcal{P}^{\mathsf{T}} \bar{\mathcal{Y}} \ . \tag{8}$$

It is common in factorization methods to center the data with respect to their centroid to cancel the translation part of the transformation. Equation (8) means

that the data must be centered with respect to the *reconstructed centroid* of the image points, not with respect to the actual 3D centroid.

Obviously, if the 3D models have been obtained by the factorization method of Sect. 3, then the centroid of the 3D points corresponds to the reconstructed centroid, *i.e.* $\bar{\mathbf{Q}} = \mathcal{P}^\mathsf{T}\bar{\mathcal{Y}}$ and $\bar{\mathbf{Q}}' = \mathcal{P}'^\mathsf{T}\bar{\mathcal{Y}}'$, provided that the same sets of views are used for reconstruction and alignment.

To summarize, we cancel the translation part out of the sought-after transformation by translating the reconstructions and the image points as shown below

$$\begin{cases} \mathbf{Q}_j \leftarrow \mathbf{Q}_j - \mathcal{P}^\mathsf{T}\bar{\mathcal{Y}} \\ \mathbf{Q}'_j \leftarrow \mathbf{Q}'_j - \mathcal{P}'^\mathsf{T}\bar{\mathcal{Y}}' \end{cases} \quad \text{and} \quad \begin{cases} \mathbf{x}_{ij} \leftarrow \mathbf{x}_{ij} - \mathsf{P}_i\mathcal{P}^\mathsf{T}\bar{\mathcal{Y}} \\ \mathbf{x}'_{ij} \leftarrow \mathbf{x}'_{ij} - \mathsf{P}'_i\mathcal{P}'^\mathsf{T}\bar{\mathcal{Y}}' \end{cases} .$$

The reprojection error (4) is rewritten

$$\mathcal{C}^2(\mathcal{Q}, \mathcal{Q}') = \frac{1}{2nm}\left(\|\mathcal{X} - \mathcal{P}\mathcal{Q}\|^2 + \|\mathcal{X}' - \mathcal{P}'\mathcal{Q}'\|^2\right) \tag{9}$$

and problem (5) is reformulated as

$$\min_{\hat{\mathcal{Q}}, \hat{\mathcal{Q}}'} \mathcal{C}^2(\hat{\mathcal{Q}}, \hat{\mathcal{Q}}') \qquad \text{s.t.} \qquad \hat{\mathbf{Q}}'_j = \hat{\mathsf{A}}\hat{\mathbf{Q}}_j \ . \tag{10}$$

Step 3: Factorizing. Thanks to the orthonormal basis property $\mathcal{P}^\mathsf{T}\mathcal{P} = \mathrm{I}$, and since for any column-orthonormal matrix $\mathcal{A}$, $\|\mathcal{A}\mathbf{x}\| = \|\mathbf{x}\|$, we can rewrite the reprojection error for a single set of cameras as

$$\mathcal{R}^2(\mathcal{P}, \mathcal{Q}) \ \propto \ \|\mathcal{X} - \mathcal{P}\mathcal{Q}\|^2 \ = \ \|\mathcal{P}^\mathsf{T}\mathcal{X} - \mathcal{Q}\|^2 \ .$$

This allows to rewrite the reprojection error (9) as

$$\mathcal{C}^2(\hat{\mathcal{Q}}, \hat{\mathcal{Q}}') \ \propto \ \|\mathcal{P}^\mathsf{T}\mathcal{X} - \hat{\mathcal{Q}}\|^2 + \|\mathcal{P}'^\mathsf{T}\mathcal{X}' - \hat{\mathcal{Q}}'\|^2 \ = \ \| \underbrace{\begin{pmatrix} \mathcal{P}^\mathsf{T}\mathcal{X} \\ \mathcal{P}'^\mathsf{T}\mathcal{X}' \end{pmatrix}}_{\Lambda} - \underbrace{\begin{pmatrix} \hat{\mathcal{Q}} \\ \hat{\mathcal{Q}}' \end{pmatrix}}_{\Delta} \|^2 \ .$$

By introducing the constraint $\hat{\mathcal{Q}}' = \hat{\mathsf{A}}\hat{\mathcal{Q}}$ from (10) and, as in Sect. 3, an unknown global affine transformation B we can write

$$\Delta \ = \ \begin{pmatrix} \mathrm{I} \\ \hat{\mathsf{A}} \end{pmatrix} \mathsf{B}\mathsf{B}^{-1}\hat{\mathcal{Q}} \ = \ \underbrace{\begin{pmatrix} \mathsf{B} \\ \hat{\mathsf{A}}\mathsf{B} \end{pmatrix}}_{\tilde{\mathcal{M}}} \underbrace{\mathsf{B}^{-1}\mathcal{Q}}_{\tilde{\mathcal{Q}}} \ .$$

The problem is reformulated as

$$\min_{\tilde{\mathcal{M}}, \tilde{\mathcal{Q}}} \|\Lambda - \tilde{\mathcal{M}}\tilde{\mathcal{Q}}\|^2 \ .$$

A solution is given by the SVD of matrix Λ

$$\Lambda_{(6\times m)} = \mathsf{U}_{(6\times 6)}\Sigma_{(6\times 6)}\mathsf{V}^\mathsf{T}_{(6\times m)} \ .$$

As in Sect. 3, let $\Sigma = \Sigma_u \Sigma_v$ be any decomposition of matrix Σ. We obtain $\tilde{\mathcal{M}} = \psi(\mathsf{U}\Sigma_u)$ and $\tilde{\mathcal{Q}} = \psi^{\mathsf{T}}(\mathsf{V}\Sigma_v^{\mathsf{T}})$. Using the partitioning $\tilde{\mathcal{M}} = \begin{pmatrix} \tilde{\mathsf{M}} \\ \tilde{\mathsf{M}}' \end{pmatrix}$, we get

$$\begin{cases} \mathsf{B} = \tilde{\mathsf{M}} \\ \hat{\mathsf{A}} = \tilde{\mathsf{M}}'\mathsf{B}^{-1} \\ \hat{\mathcal{Q}} = \mathsf{B}\tilde{\mathcal{Q}} \end{cases} .$$

Obviously, one needs to undo the effect of the orthonormalizing transformations, as follows

$$\begin{cases} \hat{\mathsf{A}} \leftarrow \mathsf{N}'\hat{\mathsf{A}}\mathsf{N}^{-1} \\ \hat{\mathcal{Q}} \leftarrow \mathsf{N}\hat{\mathcal{Q}} \end{cases} .$$

This algorithm runs with $m \geq 4$ point correspondences.

Note that it is possible to solve the problem without using the orthonormalizing transformations. This solution requires however to compute the SVD of a $(2(n+n') \times m)$ matrix, made by stacking the measurement matrices $\mathcal{X}$ and $\mathcal{X}'$, and is therefore much more computationally expensive than the algorithm above, and may be intractable for large sets of cameras and points.

4.3 Dealing with missing data.

The missing data case arises when some of the 3D points used for the alignment are not visible in all views. We propose an Expectation Maximization based extension of the algorithm to handle this case.

The EM algorithm is an iterative method which estimates the model parameters, given an incomplete set of measurement data. The main idea is to alternate between predicting the missing data and estimating the model. Since the log likelihood cannot be maximized using factorization, due to the missing data, it is replaced by its conditional expectation given the observed data, using the current estimate of the parameters. In the case where the log likelihood is a linear function of the missing data, this simply consists in replacing the missing data by their conditional expectations given the observed data at current parameter values. This approximated log likelihood is then maximized so as to yield a new estimate of the parameters. Monotone convergence to a local minimum of the Maximum Likelihood residual error (4) is shown *e.g.* in [11].

Since the reconstruction of both camera sets using factorization needs a complete data set, we are limited to the points visible in all views for the initial reconstruction. This allows to reconstruct all cameras, but only part of the 3D points. We then triangulate the missing points in order to complete the 3D point cloud. This preliminary expectation step yields a completed set of 3D data, that can be used in the alignement algorithm.

However, the reprojection error, *i.e.* the negative log likelihood, still cannot be minimized because of the incomplete measurement matrix $\mathcal{X}$. The expectation step predicts the missing image points by reprojecting them from the completed 3D points, namely for the missing point $\mathbf{x}_{ij}$, we set $\mathbf{x}_{ij} \leftarrow \mathsf{P}_i\hat{\mathbf{Q}}_j + \mathbf{t}_i$.

Table 1. The proposed Maximum Likelihood alignment algorithm.

OBJECTIVE

Given $m \geq 4$ 3D point correspondences $\{\mathbf{Q}_j \leftrightarrow \mathbf{Q}'_j\}$ obtained by affine reconstruction and triangulation of the missing data from two sets of images, with respectively n cameras $\{(\mathsf{P}_i, \mathbf{t}_i)\}$ and n' cameras $\{(\mathsf{P}'_i, \mathbf{t}'_i)\}$, as well as measured image points $\{\mathbf{x}_{ij}\}$ and $\{\mathbf{x}'_{ij}\}$ forming an incomplete data set, compute the affine transformation $(\hat{\mathsf{A}}, \hat{\mathbf{t}})$ and corrected point positions $\{\hat{\mathbf{Q}}_j \leftrightarrow \hat{\mathbf{Q}}'_j\}$ such that the reprojection error e is minimized.

ALGORITHM

1. **Compute the orthonormalizing transformations**:

$$\begin{pmatrix} \cdots & \mathsf{P}_i{}^{\mathsf{T}} & \cdots \end{pmatrix}^{\mathsf{T}} \overset{\text{QR}}{=} \mathcal{P}\mathsf{N}^{-1} \quad \text{and} \quad \begin{pmatrix} \cdots & \mathsf{P}'_i{}^{\mathsf{T}} & \cdots \end{pmatrix}^{\mathsf{T}} \overset{\text{QR}}{=} \mathcal{P}'\mathsf{N}'^{-1} \ .$$

2. **Form the 'joint projection' and the measurement matrices**:

$$\mathcal{X} = \begin{pmatrix} \vdots \\ \cdots \ (\mathbf{x}_{ij} - \mathbf{t}_i) \ \cdots \\ \vdots \end{pmatrix} \quad \text{and} \quad \mathcal{X}' = \begin{pmatrix} \vdots \\ \cdots \ (\mathbf{x}'_{ij} - \mathbf{t}'_i) \ \cdots \\ \vdots \end{pmatrix} \ .$$

3. **Expectation-Maximization**:
   (a) **Expectation**. Predict the missing point $\mathbf{x}_{ij}$ by setting $\mathbf{x}_{ij} \leftarrow \mathsf{P}_i\hat{\mathbf{Q}}_j$. Compute the reconstructed centroids:

$$\mathbf{C} = \frac{\mathcal{P}^{\mathsf{T}}}{m} \sum_{j=1}^{m} \begin{pmatrix} \vdots \\ \mathbf{x}_{ij} \\ \vdots \end{pmatrix} \quad \text{and} \quad \mathbf{C}' = \frac{\mathcal{P}'^{\mathsf{T}}}{m} \sum_{j=1}^{m} \begin{pmatrix} \vdots \\ \mathbf{x}'_{ij} \\ \vdots \end{pmatrix} \ .$$

   Cancel the translations:

$$\mathcal{X} = \begin{pmatrix} \vdots \\ \cdots \ (\mathbf{x}_{ij} - \mathsf{P}_i\mathbf{C}) \ \cdots \\ \vdots \end{pmatrix} \quad \text{and} \quad \mathcal{X}' = \begin{pmatrix} \vdots \\ \cdots \ (\mathbf{x}'_{ij} - \mathsf{P}'_i\mathbf{C}') \ \cdots \\ \vdots \end{pmatrix} \ .$$

   (b) **Maximization**. Factorize:

$$\begin{pmatrix} \mathcal{P}^{\mathsf{T}}\mathcal{X} \\ \mathcal{P}'^{\mathsf{T}}\mathcal{X}' \end{pmatrix} \overset{\text{SVD}}{=} \mathsf{U}\Sigma\mathsf{V}^{\mathsf{T}} \quad \text{and set} \quad \begin{pmatrix} \tilde{\mathsf{M}} \\ \tilde{\mathsf{M}}' \end{pmatrix} = \psi(\mathsf{U}\sqrt{\Sigma}) \quad \text{and} \quad \tilde{\mathcal{Q}} = \psi^{\mathsf{T}}(\mathsf{V}\sqrt{\Sigma}) \ .$$

   (c) **Recover the corrected points**. Set $\hat{\mathcal{Q}} = \mathsf{N}\tilde{\mathsf{M}}\tilde{\mathcal{Q}}$ and $\hat{\mathcal{Q}}' = \mathsf{N}'\tilde{\mathsf{M}}'\tilde{\mathcal{Q}}$.
   (d) **Transfer the points to the original coordinate frames**. Extract the corrected points $\hat{\mathbf{Q}}_j$ from $\hat{\mathcal{Q}}$. Translate them as $\hat{\mathbf{Q}}_j \leftarrow \hat{\mathbf{Q}}_j + \mathbf{C}$.
   (e) **Compute the reprojection error**:
   Set $e^2 = \frac{1}{2nm}\left(\sum_{j=1}^{m}\left(\sum_{i=1}^{n} d^2(\mathbf{x}_{ij} - \mathsf{P}_i\hat{\mathbf{Q}}_j) + \sum_{i=1}^{n'} d^2(\mathbf{x}'_{ij} - \mathsf{P}'_i\hat{\mathbf{Q}}'_j)\right)\right)$.
   (f) **Loop**. If convergence is not reached (see Sect. 4.3), loop on step (a).
4. **Recover the transformation**: Set $\hat{\mathsf{A}} = \mathsf{N}'\tilde{\mathsf{M}}'\tilde{\mathsf{M}}^{-1}\mathsf{N}^{-1}$ and $\mathbf{t} = \mathbf{C}' - \hat{\mathsf{A}}\mathbf{C}$.

The maximization step consists in applying the algorithm described in the complete data case. This yields an estimate of the sought-after transformation $(\hat{\mathsf{A}}, \hat{\mathbf{t}})$ as well as corrected point positions $\{\hat{\mathbf{Q}}_j \leftrightarrow \hat{\mathbf{Q}}'_j\}$.

These two steps are alternated, thus forming an iterative procedure where the corrected points are used in the expectation at the next iteration. In order to decide whether convergence is reached, the change in reprojection error between two iterations is measured. When the reprojection error stabilizes, the final result is returned.

Table 1 gives a summary of the algorithm with its EM extension.

5 Other Algorithms

We briefly describe two other alignment algorithms. They do not yield Maximum Likelihood Estimates under the previously-mentioned hypotheses on the noise distribution. They rely on 3D measurements and therefore naturally handle missing image data.

5.1 Minimizing the Non-Symmetric Transfer Error

This algorithm, dubbed 'TRERROR', is specific to the two camera set case. It is based on minimizing a non-symmetric 3D transfer error $\mathcal{E}(\hat{\mathsf{A}})$ as follows

$$\min_{\hat{\mathsf{A}},\hat{\mathbf{t}}} \mathcal{E}^2(\hat{\mathsf{A}}, \hat{\mathbf{t}}) \qquad \text{with} \qquad \mathcal{E}^2(\hat{\mathsf{A}}) = \frac{1}{m}\sum_{j=1}^{m} \|\mathbf{Q}'_j - \hat{\mathsf{A}}\mathbf{Q}_j - \hat{\mathbf{t}}\|^2 \ .$$

Differentiating $\mathcal{E}^2$ with respect to $\hat{\mathbf{t}}$ and nullifying the result allows to eliminate the translation by centering each 3D point set on its centroid. By rewriting the error function and applying standard linear least-squares, one obtains

$$\hat{\mathsf{A}} \ = \ \mathcal{Q}'\mathcal{Q}^{\dagger} \qquad \text{and} \qquad \hat{\mathbf{t}} \ = \ \hat{\mathbf{Q}}' - \hat{\mathsf{A}}\hat{\mathbf{Q}} \ .$$

5.2 Direct 3D Factorization

This algorithm, dubbed 'FACT3D', is based on directly factorizing the 3D reconstructed points. It is not restricted to the two camera set case, but for simplicity, we only describe this case. Generalization to multiple camera sets is trivial. The algorithm computes the aligning transformation $(\hat{\mathsf{A}}, \hat{\mathbf{t}})$ and perfectly corresponding points $\{\hat{\mathbf{Q}}_j \leftrightarrow \hat{\mathbf{Q}}'_j\}$. The reconstructed cameras are not taken into account by this algorithm, which entirely relies on 3D measures on the reconstructed points. Under certain conditions, this algorithm is equivalent to the proposed FACTMLE-EM.

The problem is stated as

$$\min_{\hat{\mathcal{Q}},\hat{\mathcal{Q}}'} \mathcal{D}^2(\hat{\mathcal{Q}}, \hat{\mathcal{Q}}') \qquad \text{s.t.} \qquad \hat{\mathbf{Q}}'_j = \hat{\mathsf{A}}\hat{\mathbf{Q}}_j + \hat{\mathbf{t}} \ ,$$

where the 3D error function employed is defined by

$$\mathcal{D}^2(\hat{\mathcal{Q}}, \hat{\mathcal{Q}}') = \frac{1}{2m}\left(\|\mathcal{Q} - \hat{\mathcal{Q}}\|^2 + \|\mathcal{Q}' - \hat{\mathcal{Q}}'\|^2\right) \ .$$

Minimizing this error function means that if the noise *on the 3D point coordinates* were Gaussian, centered and i.i.d., which is *not* the case with our actual hypotheses, then this algorithm would yield the Maximum Likelihood Estimate.

Step 1: Eliminating the translation. By using the technique from Sect. 4.2, we obtain $\hat{\mathbf{t}} = \bar{\mathbf{Q}}' - \hat{\mathsf{A}}\bar{\mathbf{Q}}$. As in most factorization methods, cancelling the translation part out according to the error function $\mathcal{D}$ is done by centering each set of 3D points on its actual centroid: $\hat{\mathbf{Q}}_j \leftarrow \hat{\mathbf{Q}}_j - \bar{\mathbf{Q}}$ and $\hat{\mathbf{Q}}'_j \leftarrow \hat{\mathbf{Q}}'_j - \bar{\mathbf{Q}}'$. Henceforth, we assume to work in centered coordinates. The problem is rewritten as

$$\min_{\hat{\mathcal{Q}}, \hat{\mathcal{Q}}'} \mathcal{D}^2(\hat{\mathcal{Q}}, \hat{\mathcal{Q}}') \qquad \text{s.t.} \qquad \hat{\mathbf{Q}}'_j = \hat{\mathsf{A}}\hat{\mathbf{Q}}_j \ .$$

Step 2: Factorizing. Following the approach in Sect. 4.2, we rewrite $\mathcal{D}$ as

$$\mathcal{D}^2(\hat{\mathcal{Q}}, \hat{\mathcal{Q}}') \propto \| \begin{pmatrix} \mathcal{Q} \\ \mathcal{Q}' \end{pmatrix} - \begin{pmatrix} \hat{\mathcal{Q}} \\ \hat{\mathcal{Q}}' \end{pmatrix} \|^2 = \| \underbrace{\begin{pmatrix} \mathcal{Q} \\ \mathcal{Q}' \end{pmatrix}}_{\Lambda} - \underbrace{\begin{pmatrix} \mathsf{B} \\ \mathsf{AB} \end{pmatrix}}_{\tilde{\mathcal{M}}} \underbrace{\mathsf{B}^{-1}\hat{\mathcal{Q}}}_{\tilde{\mathcal{Q}}} \|^2 \ .$$

Using SVD of matrix $\Lambda = \mathsf{U}\Sigma\mathsf{V}^\mathsf{T}$, we obtain $\tilde{\mathcal{M}} = \psi(\mathsf{U}\Sigma_u)$ and $\tilde{\mathcal{Q}} = \psi^\mathsf{T}(\mathsf{V}\Sigma_v^\mathsf{T})$. By using the partitioning $\tilde{\mathcal{M}} = \left(\tilde{\mathsf{M}}\tilde{\mathsf{M}}'\right)^\mathsf{T}$, we get

$$\hat{\mathsf{A}} = \tilde{\mathsf{M}}'\tilde{\mathsf{M}}^{-1} \qquad \text{and} \qquad \hat{\mathcal{Q}} = \tilde{\mathsf{M}}\tilde{\mathcal{Q}} \ .$$

6 Experimental Evaluation

We evaluate our algorithm using simulated and real data. The implementation of all three compared algorithms, *i.e.* FACTMLE-EM, TRERROR and FACT3D, as well as the generation of simulated data, have been done in C++.

6.1 Simulated Data

We generate m 3D points and two sets of n weak perspective cameras each. The pose of a camera is defined by its three dimensional location, viewing direction and roll angle (rotation angle around the optical axis). The corresponding affine projection matrix is given by a (2×3), truncated, rotation matrix $\bar{\mathsf{R}}_i$ together with a two-dimensional translation vector $\mathbf{t}_i$, both of which premultiplied by an internal calibration matrix. More precisely, we use weak perspective cameras $\mathsf{P}_i = \mathsf{A}_i\bar{\mathsf{R}}_i$ and $\mathbf{t}_i = \mathsf{A}_i\bar{\mathbf{T}}_i$, where A_i is the internal calibration matrix

$$\mathsf{A}_i = k_i \begin{pmatrix} \tau_i & 0 \\ 0 & 1 \end{pmatrix} \ .$$

The scale factor k_i models the average depth of the object and the focal length of the camera, and τ models the aspect ratio that we choose very close to 1. The 3D points are chosen from a uniform distribution inside a thin rectangular parallelepiped with dimensions $1 \times 1 \times (1-d)$, and the scale factors k_i are chosen so that the points are uniformly spread in 400×400 pixel images.

We generate three point sets containing the point visibles (i) in the first camera set, (ii) in the second one and (iii) in both camera sets. The third subset contains m_c points, whereas the two first subsets both contains $m - m_c$ points. Hence, m points are used to perform SFM on each camera set, while m_c points are used for the alignment. The points are projected onto the images where they are visible and gaussian noise with zero mean and standard deviation σ is added.

In order to assess the behaviour of the algorithms in the presence of non-perfectly affine cameras, we introduce the factor $0 \leq a \leq 1$. Let Z_{ij} be the depth of the j-th 3D point with respect to camera i, we scale the projected points $\mathbf{x}_{ij}$ by $\mathbf{x}_{ij} \leftarrow \frac{1}{\nu}\mathbf{x}_{ij}$ with $\nu = a + (1-a)Z_{ij}$, meaning that for $a = 1$, the points does not change and the projection is perfectly affine, and when a tends towards 0, the points undergo stronger and stronger perspective effects. The points are further scaled so that their standard deviation remains invariant, in order to keep them wellspread in the images.

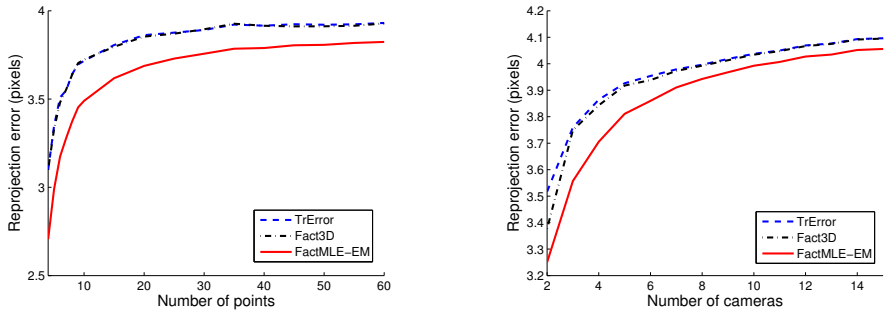
So as to simulate the problem of incomplete data, *e.g.* due to occlusions, we generate a list of missing image points. We introduce the probability p_{point} that any given 3D point is occluded in some images and the probability p_{image} that it is occluded in one particular image. For simplicity, we take $p_{point} = p_{image} = p$, which gives a rate of missing data of p^2.

A 3D model is reconstructed from each of the two camera sets using the factorization algorithm described in Sect. 3. Once the camera matrices and 3D points are estimated, only the m_c points common to the two camera sets are considered for the alignment. We define the overlap ratio of the two camera sets to be $\theta = m_c/m$, *i.e.* for $\theta = 1$ all points are seen in all views, while for $\theta = 0$, the two sets of cameras do not share corresponding points.

Each of the three alignment algorithms yields estimates for the 3D affine transformation and corrected point clouds, except TRERROR which only gives the transformation. The comparison of the algorithms being based on the re-projection error, the point clouds used to compute it need to be re-estimated so that this error is minimized, given an estimated transformation. This must be done for TRERROR and FACT3D, but is useless for FACTMLE-EM.

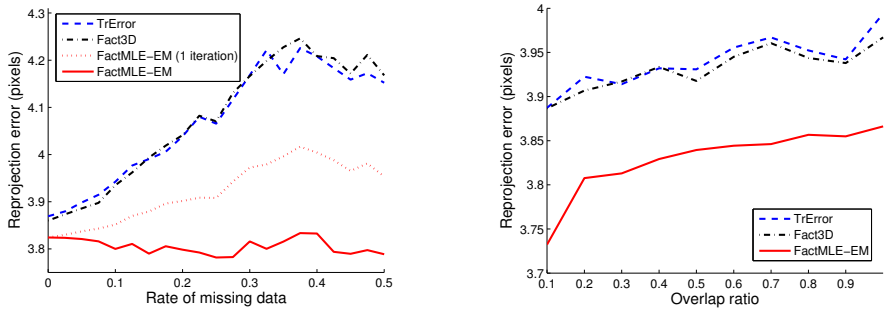
We use the following default setting for the simulations: $n = 5$ views, $m = 250$ points, $\theta = 0.2$ (*i.e.* a 20% overlap and $m_c = 50$ points common to the two 3D models), $\sigma = 3.0$ pixels, $d = 0.95$ (flat 3D scene), $a = 1$ (perfectly affine projections) and $p = 0.3$ (rate of missing data $p^2 = 0.09$). We vary each parameter at a time. Figures 2, 3 and 4 show the reprojection error averaged over 500 simulations for the three algorithms for different parameter values.

In Fig. 2, we vary the number of common points m_c (coupled with the total number of points m, so as to keep the overlap constant) and the number of cameras n, the former from 4 to 60, corresponding respectively to $m = 20$ and



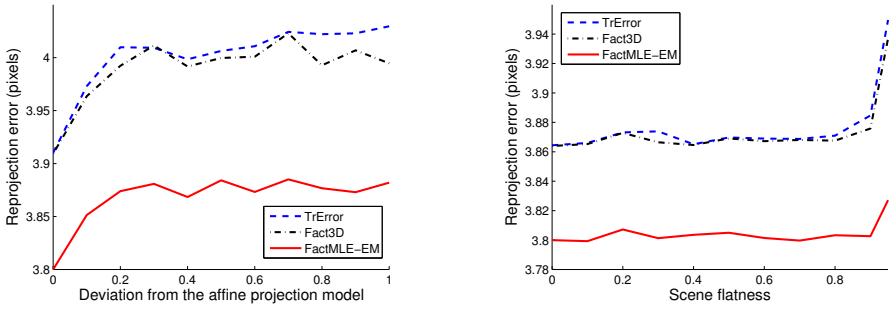
(a) (b)

Fig. 2. Reprojection error against (a) the number of points m_c and (b) the number of cameras n.



(a) (b)

Fig. 3. Reprojection error against (a) the rate of missing data and (b) the extent of overlap θ between the two sets of cameras. For $\theta = 1$, all points are seen in all views.



(a) (b)

Fig. 4. Reprojection error against (a) the deviation a from the affine projection model and (b) the scene flatness d. For $a = 1$, the projection is perfectly affine. For $d = 1$, all 3D points lie on a plane.

$m = 300$, and the latter from 2 to 15. We see that for $m_c > 20$, the number of points has a much smaller influence on the errors. Whereas FACT3D and TRERROR show similar behaviour, FACTMLE-EM is distinguished by its lower reprojection error. The difference between our method and the other two seems to be more important in the cases where we have few points or few cameras.

In Fig. 3, the rate of missing data and the overlap ratio (coupled with the number of common points m_c, so as to keep the total number of points m constant) are varied, the former from 0 to 0.5 and the latter from 0.1 to 1.0. In order to emphasize the contribution of the EM scheme, in Fig. 3(a) we also display the reprojection error of FACTMLE-EM after the first iteration. When the rate of missing data grows, the three methods show different tendencies. Whereas FACTMLE-EM handles missing data well, the other methods prove to be unstable. However, considering only one iteration of FACTMLE-EM, the reprojection error increases just as for the other methods. The difference in performance is thus provided by the EM iterations.

In Fig. 4 the deviation from the affine model a varies from 0 to 1, from a perfectly affine projection, and the flatness of the simulated data d varies from 0 to 1, *i.e.* from a cube to a plane. Despite the fact that the alignement is affine, even completely projective cameras seem to be well modeled by the three methods. In fact, the error induced by the affine approximation is small compared to the added noise. The flatness of the scene does not change the result, except for very flat scenes making the algorithms unstable, FACT3D and TRERROR somewhat more than FACTMLE-EM. This result was expected since planar scenes are singular for the computation of a 3D affine transformation.

Simulations with varying σ reveal a quasi linear relationship between the the noise level and the reprojection error. The slope is somewhat less steep in the case of FACTMLE-EM than for the other two methods, indicating that our method is less sensitive to noise.

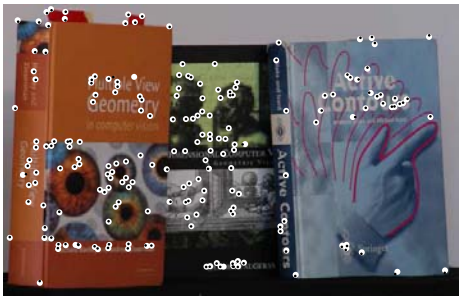
Although the three algorihtms have similar behaviour throughout the sequence of tests, except when varying the rate of missing data, FACTMLE-EM consistently outperforms the other ones.

6.2 Real Data

We applied the algorithms to real image sequences as follows. A number of images of a scene were taken from different angles and grouped into two sets. A certain number of point correspondences were defined within each one of the image sets, as well as for all the images, thus forming the measurement matrices $\mathcal{X}$ and $\mathcal{X}'$.

The camera used is an uncalibrated digital Nikon D100 with a lens of focal length $80 - 200$ mm, giving an image size of 2240×1488 pixels.

The 'books' sequence. We used a series of images of a rather flat scene, together with a large set of point correspondences, given by a tracking algorithm, shown in Fig. 5(a). So as to keep the experimental conditions close to the hypothesis of affine cameras, the photos are taken far away from the object using a large zoom.



(a) One image from the 'books' sequence overlaid with the $m_c = 196$ point correspondences in white and reprojected points in black.

(b) A detail from the image in (a) showing the original points in black and reprojected points in white, from FACTMLE-EM (points), FACT3D (stars) and TRERROR (crosses).

Fig. 5. Results from the 'books' sequence.

This group of images consists of two sets of respectively $n = 2$ and $n' = 3$ images, together with the $m_c = 196$ common point correspondences, and respectively $m = 628$ and $m' = 634$ correspondences for the two sets, giving an approximate overlap of 80%. The reprojection errors we obtained are:

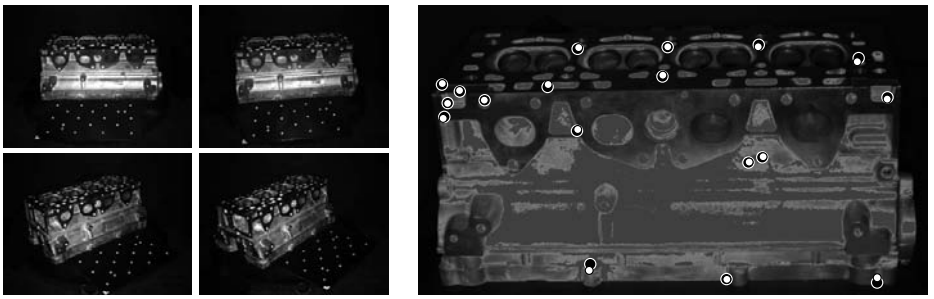
FACTMLE-EM	1.90 pixels
FACT3D	1.97 pixels
TRERROR	2.17 pixels

A detail of an image with the reprojected points due to all three methods is shown in Fig. 5(b). As predicted by the tests on simulated data, FACTMLE-EM performs better than FACT3D and TRERROR.

The 'cylinder head' sequence. This sequence was acquired under different conditions than the previous one. The photos were taken with the same camera, using a lens with a focal length of 12 mm, at a distance of 60 cm of the object, which is 40 cm long. The points, shown in Fig. 6(b), were manually entered. Using these settings, the affine camera model does not apply and the reconstruction performed prior to the alignment is therefore less reliable. Nevertheless, the result of the alignment is rather good. This group of images consists of two sets of respectively $n = n' = 2$ images, together with the $m_c = 18$ common point correspondences, and respectively $m = 22$ and $m' = 23$ correspondences for the two sets, giving an approximate overlap of 31%. The reprojection errors are:

FACTMLE-EM	3.03 pixels
FACT3D	3.04 pixels
TRERROR	3.05 pixels

The two sets of images are displayed in Fig. 6(a) and the given point matches together with the FACTMLE-EM reprojections are displayed in Fig. 6(b).



(a) The two sets of images of the 'cylinder head' sequence.

(b) The original points in black together with their FACTMLE-EM reprojections in white.

Fig. 6. Results from the 'cylinder head' sequence.

The 'building' sequence. The point correspondences are once again given by a tracking algorithm, but this time the data set is incomplete. We need at least two views of a 3D point in order to use it for the reconstruction, so we keep only those points that are present in two or more images. We then define a point correspondence to be common to the two sets and thus used for the alignment of the two reconstructions, as soon as it is present in (at least two images in each one of) the two sets. This group of images consists of two sets of respectively $n = n' = 5$ images, together with the $m_c = 40$ common point correspondences, and respectively $m = 94$ and $m' = 133$ correspondences for the two sets, giving an overlap of 43% and 30% respectively. The rates of missing data are for the first camera set 31% (13% for the common points) and for the second camera set 22% (11% for the common points). We note that the missing points are essentially not due to occlusions but to failure in the tracking algorithm or to the points being out of range in the images. The reprojection errors we obtained are:

FACTMLE-EM	0.78 pixels
FACT3D	0.84 pixels
TRERROR	0.85 pixels

As predicted by the simulations with varying rate of missing data, the difference between the methods is more important when processing incomplete data. Whereas FACT3D and TRERROR yield similar errors, FACTMLE-EM distinguishes itself with a significantly lower error. The results are displayed in Fig. 7.

7 Conclusions

We presented a method to compute the Maximum Likelihood Estimate of 3D affine transformations, under standard hypotheses on the noise distribution, aligning sets of 3D points obtained from uncalibrated affine cameras. The method

(a) (b)

Fig. 7. The original common points in white together with their FACTMLE-EM reprojections in black. The two images are the first ones in the respective camera sets.

computes all aligning transformations in a single computation step in the occlusion-free case, by minimizing the reprojection error over all points and all images. An iterative extension is presented for the missing data case. Experimental results on simulated and real data show that the proposed method consistently performs better than other methods based on 3D measurements.

Future work could be devoted to the incorporation of other types of features.

References

1. Fitzgibbon, A., Zisserman, A.: Automatic camera recovery for closed or open image sequences. In: ECCV. (1998) 311–326
2. Tomasi, C., Kanade, T.: Shape and motion from image streams under orthography: A factorization method. IJCV **9** (1992) 137–154
3. Beardsley, P., Zisserman, A., Murray, D.: Sequential updating of projective and affine structure from motion. IJCV **23** (1997) 235–259
4. Csurka, G., Demirdjian, D., Horaud, R.: Finding the collineation between two projective reconstructions. Comp. Vision and Image Underst. **75** (1999) 260–268
5. Walker, M., Shao, L., Volz, R.: Estimating 3D location parameters using dual number quaternions. Computer Vision, Graphics and Image Processing: Image Understanding **54** (1991) 358–367
6. Mundy, J., Zisserman, A., eds.: Geometric Invariance in Computer Vision. The MIT Press, Cambridge, MA, USA (1992)
7. Bartoli, A., Martinsson, H., Gaspard, F., Lavest, J.M.: On aligning sets of points reconstructed from uncalibrated affine cameras. In: SCIA. (2005) 531–540
8. Reid, I., Murray, D.: Active tracking of foveated feature clusters using affine structure. IJCV **18** (1996) 41–60
9. Golub, G., van Loan, C.: Matrix Computation. The Johns Hopkins University Press, Baltimore (1989)
10. Jacobs, D.: Linear fitting with missing data: Applications to structure-from-motion and to characterizing intensity images. In: CVPR. (1997) 206–212
11. McLachlan, G., Krishnan, T.: The EM algorithm and extensions. John Wiley & Sons, Inc. (1997)

8.2 Structure-from-Motion with Lines

I16 **A Framework for Pencil-of-Points Structure-From-Motion** §8.2.1
A. Bartoli, M. Coquerelle and P. Sturm
ECCV'04 - *European Conf. on Computer Vision*, Prague, Czech Republic, p. 28-40, vol. II, May 2004

J09 **Triangulation for Points on Lines** §8.2.2
A. Bartoli and J.-T. Lapresté
Image and Vision Computing, Vol. 26, No. 2, p. 315-324, February 2008
Previous version: [I26]

I35 **Kinematics From Lines in a Single Rolling Shutter Image** §8.2.3
O. Ait-Aider, A. Bartoli and N. Andreff
CVPR'07 - IEEE *Int'l Conf. on Computer Vision and Pattern Recognition*, Minneapolis, USA, June 2007

8.2.1 Paper (ECCV'04) – *A Framework For Pencil-of-Points Structure-From-Motion*

A Framework For Pencil-of-Points Structure-From-Motion

Adrien Bartoli[1,2], Mathieu Coquerelle[2], and Peter Sturm[2]

[1] Department of Engineering Science, University of Oxford, UK
[2] équipe MOVI, INRIA Rhône-Alpes, France
Bartoli@robots.ox.ac.uk, Coquerel@inria.fr, Sturm@inria.fr

Abstract. Our goal is to match contour lines between images and to recover structure and motion from those. The main difficulty is that pairs of lines from two images do not induce direct geometric constraint on camera motion. Previous work uses geometric attributes — orientation, length, etc. — for single or groups of lines. Our approach is based on using Pencil-of-Points (points on line) or POPs for short. There are many advantages to using POPs for structure-from-motion. The most important one is that, contrarily to pairs of lines, pairs of POPs may constrain camera motion. We give a complete theoretical and practical framework for automatic structure-from-motion using POPs — detection, matching, robust motion estimation, triangulation and bundle adjustment. For wide baseline matching, it has been shown that cross-correlation scores computed on neighbouring patches to the lines gives reliable results, given 2D homographic transformations to compensate for the pose of the patches. When cameras are known, this transformation has a 1-dimensional ambiguity. We show that when cameras are unknown, using POPs lead to a 3-dimensional ambiguity, from which it is still possible to reliably compute cross-correlation. We propose linear and non-linear algorithms for estimating the fundamental matrix and for the multiple-view triangulation of POPs. Experimental results are provided for simulated and real data.

1 Introduction

Recovering structure and motion from images is one of the key goals in computer vision. A common approach is to detect and match image features while recovering camera motion. The goal of this paper is the automatic matching of lines and recovery of structure and motion. This problem is difficult for the reason that a pair of corresponding lines does not give direct geometric constraint on the camera motion. Hence, one has to work on a three-view basis or assume that camera motion is known a priori, e.g. [10].

In this paper, we attack directly the two view case by introducing a type of image primitive that we call *Pencil-of-Points* or POP for short. A POP is made of a *supporting line* and a set of *supporting points* lying on the supporting line. Physically, a POP corresponds to a set of interest points on a contour line. POPs can be built on the top of most contour lines. Contrarily to pairs of corresponding lines, pairs of corresponding POPs may give geometric constraints on camera motion, provided that what we call the *local geometry*, relating corresponding points along

the supporting lines, has been computed. We exploit these geometric constraints for matching POPs and recovering structure and motion. Once camera motion has been recovered using POPs, it can be employed for a reliable guided-matching and reconstruction of other types of features.

The closest work to ours is [10]. The main difference is that the authors consider that the cameras are known and propose a wide-baseline guided-matching algorithm for lines. They show that reliable results are obtained based on cross-correlation scores, computed by warping the neighbouring textures of the lines using the 2D homography $\mathsf{H}(\mu) \sim [\mathbf{l}']_\times \mathsf{F} + \mu \mathbf{e}' \mathbf{l}^\mathsf{T}$, where $\mathbf{l} \leftrightarrow \mathbf{l}'$ are corresponding lines, F is the fundamental matrix and $\mathbf{e}'$ the second epipole. The projective parameter μ is computed by minimizing the cross-correlation score.

Before going into further details about our approach, we underline some of the advantages of using POPs for automatic structure and motion recovery. First, a POP has fewer degrees of freedom than the supporting line and the individual supporting points which implies that (*i*) its localization is often more accurate that those of the individual features, (*ii*) finding POPs in a set of interest points and contour lines increase their individual repeatability rate and (*iii*) structure and motion parameters estimated from POPs are more accurate than that recovered from points and/or lines. Second, matching or tracking POPs through images is more reliable than individual contour lines or interest points, since a pair of corresponding POPs defines a local geometry, used to score matching hypotheses based on geometric or photometric criteria. Third, the robust estimation of camera motion based on random sampling from putative correspondences, i.e. in a RANSAC-like manner [3], is more efficient using POPs than other standard features, since only three pairs of POPs define a fundamental matrix, versus seven pairs of points.

Contributions and paper organization. Using POPs for structure-from-motion is a new concept. We propose a comprehensive framework for multiple-view matching and recovery of structure and motion. Our framework is based on the following traditional steps, which also give the organization of this paper.

First, §2, we investigate the detection of POPs in images and their matching. We define and study the *local geometry* of a pair of POPs. We propose methods for its estimation, which allow to obtain putative POP correspondences, from which the epipolar geometry can be robustly estimated.

Second, §3, we propose techniques for estimating the epipolar geometry from POP correspondences. Minimal and redundent cases are studied.

Third, §4, we tackle the problem of triangulating POPs from multiple images. We derive and approximate the optimal (in the Maximum Likelihood sens) solution by an algorithm based on the triangulation of the supporting line, then the supporting points.

Finally, bundle adjustment is described in §5. We provide experimental results on simulated data and give our conclusions and further work in §§6 and 7 respectively. Experimental results on real data are provided throughout the paper. The following two paragraphs give our notation, some preliminaries and definitions.

Notation and preliminaries. We make no formal distinction between coordinate vectors and physical entities. Equality up to a non-null scale factor is denoted

by $\sim$. Vectors are typeset using bold font ($\mathbf{q}$, $\mathbf{Q}$), matrices using sans-serif fonts (F, H) and scalars in italic (α). Transposition and transposed inverse are denoted by $^{\mathsf{T}}$ and $^{-\mathsf{T}}$. The (3×3) skew-symmetric cross-product matrix is written as in $[\mathbf{q}]_\times \mathbf{x} = \mathbf{q} \times \mathbf{x}$. Indices are used to indicate the size of a matrix or vector ($\mathsf{F}_{(3\times3)}$, $\mathbf{q}_{(3\times1)}$), to index a set of entities ($\mathbf{q}_i$) or to select coefficients of matrices or vectors (q_1, $q_{i,1}$). Index i is used for the n images, j for the m features and k for the p supporting points of a POP[3]. The supporting lines are written $\mathbf{l}_{ij}$ (the supporting line of the j-th POP in image i) and supporting points as $\mathbf{q}_{ijk}$ (the k-th supporting point of the j-th POP in image i). Indices are sometimes dropped for clarity. The identity matrix is written I and the null-vector as $\mathbf{0}$. We use the Euclidean distance between points, denoted d_e and an algebraic distance defined by:

$$d_a^2(\mathbf{q}, \mathbf{u}) = \|\mathsf{S}[\mathbf{q}]_\times \mathbf{u}\|^2 \quad \text{with} \quad \mathsf{S} = \left(\begin{smallmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{smallmatrix}\right). \tag{1}$$

Definitions. A pencil of points is a set of p supporting points lying on a supporting line. If $p \geq 3$, the POP is said to be *complete*, otherwise, it is said to be *incomplete*. A *complete correspondence* is a correspondence of complete POPs. As shown in the next section, only complete correspondences may define a local geometry.

We distinguish two kinds of correspondences of POPs: *line-level* and *point-level* correspondences. A line-level correspondence means that only the supporting lines are known to match. A point-level correspondence is stronger and means that a point-to-point mapping along the supporting lines has been established.

2 Detecting and Matching Pencil-of-Points

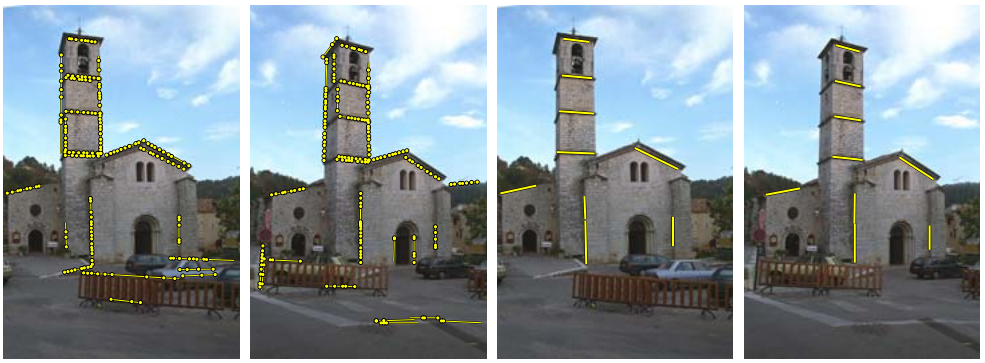
2.1 Detecting

Detecting POPs in images is the first step of the structure-from-motion process. One of the most important properties of a detector is its ability to achieve repeatability rates[4] as high as possible, which reflects the fact that it can detect the same features in different images. In order to ensure high repeatability rates, we formulate our POP detector based on interest points and contour lines, for which there exist detectors achieving high repeatability rates, see [9] for interest points and [2] for contour lines.

In order to detect salient POPs, we merge nearby contour lines. Algorithms based on the Hough transform or RANSAC [3] can be used to detect POPs within a set of points and/or lines. We propose the following simple solution. First, an empty POP is instanciated for each line (which gives the supporting line). Second, each point is attached to the POPs whose supporting line is at a distance lower than a threshold, that we typically choose as a few pixels. Finally, incomplete POPs, i.e. those for which the number of supporting points is less than three, are eliminated. Note that we use a loose threshold for interest point and contour line detection, to get as many as possible POPs. The less significant interest points and contour lines are generally pruned as they are respectively not attached to any POP or form incomplete POPs. An example of POP detection is shown on figures 1

[3] To simplify the notation, we assume without loss of generality that all POPs have the same number of supporting points.

[4] The repeatability rate between two images is the number of corresponding features over the mean number of detected points [9].



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

Fig. 1. (a) & (b) show the detected POPs. The repeatability rate is 51% while for points and lines it is lower, respectively 41% and 37%. (c) & (d) show the 9 putative matches obtained with our algorithm. On this example, all of them are correct, which shows the robustness of our local geometry based cross-correlation measure.

(a) & (b). It is observed that the repeatability rate of POPs is higher than each of the repeatability rates of points and lines.

2.2 Matching

Traditional structure-from-motion algorithms using interest points usually rely on an initial matching, followed by the robust estimation of camera geometry and a guided-matching step, see e.g. [6]. The initial matching step is often based on similarity measures between points such as correlation or grey-value invariants. Guided-matching uses the estimated camera geometry to constrain the search-area. In the case of POPs, the initial matching step is based on the local geometry defined by a pair of POPs. This step is described below followed by the robust estimation of the epipolar geometry.

Matching Based on Local Geometry As mentioned above, the idea is to use the local geometry defined by a pair of POPs. We show that this local geometry is modeled by a 1D homography and allows to establish dense correspondences between the two supporting lines. Given a hypothesized line-level POP correspondence, we upgrade it to point-level by computing its local geometry. Given a point-level correspondence, a similarity score can be computed using cross-correlation, in a manner similar to [10]. For each POP in one image, the score is computed for all POPs in the other image and a 'winner takes all' scheme is employed to extract a set of putative POP matches. Putative matches obtained by our algorithm are shown on figures 1 (c) & (d).

Defining and computing the local geometry. We study the local geometry induced by a point-level correspondence, and propose an estimation method.

Proposition 1. *Corresponding supporting points are linked by a 1D homography, related to the* epipolar transformation*, relating corresponding epipolar lines.*

Proof: Corresponding supporting points lie on corresponding epipolar lines: there is a trivial one-to-one correspondence between supporting points and epipolar lines (provided the supporting lines do not contain the epipoles). The proof follows from the fact that the epipolar pencils are related by a 1D homography [12]. ■

First, we shall define a local $\mathbb{P}^1$ parameterization of the supporting points, using two Euclidean transformation matrices A and A' acting such that the supporting lines are rotated to be vertical and aligned with the y-axes of the images. The transformed supporting points are $\mathbf{x}_k \sim \mathsf{A}\mathbf{q}_k \sim (0 \;\; y_k \;\; 1)^\mathsf{T}$ and $\mathbf{x}'_k \sim \mathsf{A}'\mathbf{q}'_k \sim (0 \;\; y'_k \;\; 1)^\mathsf{T}$. Second, we introduce a 1D homography g as:

$$\begin{pmatrix} y'_k \\ 1 \end{pmatrix} \sim \mathsf{g} \begin{pmatrix} y_k \\ 1 \end{pmatrix} \quad \text{with} \quad \mathsf{g} \sim \begin{pmatrix} g_1 & g_2 \\ g_3 & 1 \end{pmatrix}, \tag{2}$$

which is equivalent to $\mathbf{x}' \sim \mathsf{G}(\boldsymbol{\mu})\mathbf{x}$ with $\mathsf{G}(\boldsymbol{\mu}) \sim \begin{pmatrix} \mu_1 & 0 & 0 \\ \mu_2 & g_1 & g_2 \\ \mu_3 & g_3 & 1 \end{pmatrix}$, where the 3-vector $\boldsymbol{\mu}^\mathsf{T} \sim (\mu_1 \;\; \mu_2 \;\; \mu_3)$ represents projective parameters which are significant only when $\mathsf{G}(\boldsymbol{\mu})$ is applied to points off the supporting line. The 2D homography mapping corresponding points along the supporting lines is $\mathsf{H}(\boldsymbol{\mu}) \sim \mathsf{A}'^{-1}\mathsf{G}(\boldsymbol{\mu})\mathsf{A}$.

The 1D homography g can be estimated from $p \geq 3$ pairs of supporting points using equation (2). This is the reason why complete POPs are defined as those which have at least 3 supporting points. Given g, $\mathsf{H}(\boldsymbol{\mu})$ can be formed.

Computing $\mathsf{H}(\boldsymbol{\mu})$. The above-described algorithm can not be applied directly since at this stage, we only have line-level POP correspondence hypotheses. We have to upgrade them to point-level to estimate $\mathsf{H}(\boldsymbol{\mu})$ with the previously-given algorithm and score them by computing cross-correlation. We propose the following algorithm:

- for all valid pairs of triplets of supporting points[5]:
  - compute the local geometry represented by $\mathsf{H}(\boldsymbol{\mu})$.
  - compute the cross-correlation score based on $\mathsf{H}(\boldsymbol{\mu})$, see below.
- return the $\mathsf{H}(\boldsymbol{\mu})$ corresponding to the highest cross-correlation score.

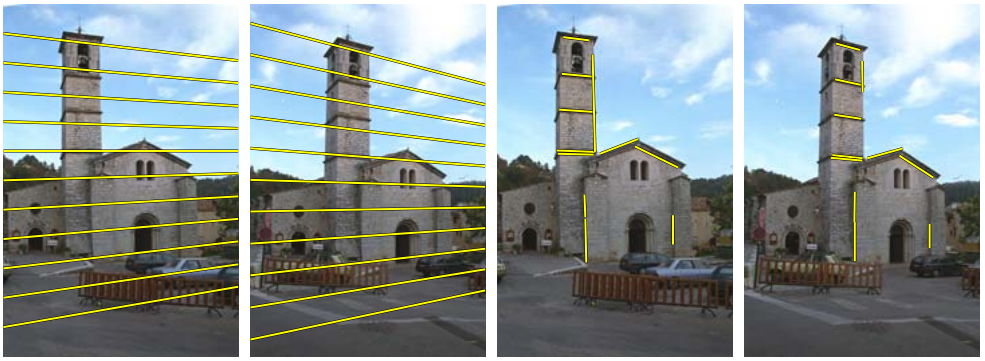
Computing cross-correlation. For a pair of POPs, the matching score is obtained by evaluating the cross-correlation using $\mathsf{H}(\boldsymbol{\mu})$ to associate corresponding points. The cross-correlation is evaluated within rectangular strips centered onto the supporting lines. The length of the strips are given by the overlap of the supporting lines in each image. The width of the strips must be sufficiently large for cross-correlation to be discriminative. During our experiments, we found that a width of 3 to 7 pixels was appropriate. For pixels off the supporting lines, the $\boldsymbol{\mu}$ parameters are significant. The following solutions are possible: compute these parameters by minimizing the cross-correlation score, as in [10], or use the median luminance and chrominance of the regions adjacent to the supporting lines [1]. The first solution is computationally too expensive to be used in our inner loop, since 3 parameters have to be estimated, while the second solution is not discriminative enough. We propose to map pixels along lines perpendicular to the supporting lines. Hence, the method uses neighbouring texture while being independent of $\boldsymbol{\mu}$. In order to take

[5] Valid triplets satisfy an ordering constraint, namely middle points have to match.

into account a possible non-planarity surrounding the supporting lines, we weight the contribution of each pixel to cross-correlation proportionally to the inverse of its distance to the supporting line.

Robustly Computing the Epipolar Geometry At this stage, we are given a set of putative POP correspondences. We employ a robust estimator, allowing to estimate the epipolar geometry and to discriminate between inliers and outliers. We use a scheme based on RANSAC [3], which maximizes the number of inliers. In order to use RANSAC, one must provide a *minimal estimator*, i.e. an estimator which computes the epipolar geometry from the minimum number of correspondences, and a function to discriminate between inliers and outliers, given an hypothesized epipolar geometry. The number of trials required to ensure a good probability of success, say 0.99, depends on the minimal number of correspondences needed to compute the epipolar geometry. Our minimal estimator described in §3 needs 3 pairs of POPs. Applying a RANSAC procedure is therefore much more efficient with POPs than with points: with 50% of outliers, 35 trials are sufficient with POPs, while 588 trials are required for points (values taken from [6]).

Our inlier/outlier discriminating function is based on computing the cross-correlation score using [10]. Inliers are selected by thresholding this score. We use a threshold of few percents (2% — 5%) of the maximal grey value. Figures 2 (a-d) show an example of epipolar geometry computation, and the set of corresponding POPs obtained after guided-matching based on the method of [10].



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

Fig. 2. (a) & (b) show a representative set of corresponding epipolar lines while (c) & (d) show the 11 matched lines obtained after guided-matching using the algorithm of [10].

3 Computing the Epipolar Geometry

Proposition 2. *The minimal number of pairs of* POP*s in general position*[6] *needed to define a unique fundamental matrix is 3.*

Proof: Due to lack of space, this proof is left for an extended version of the paper.

3.1 The 'Eight Corrected Point' Algorithm

This linear estimator is based on the constraints induced by the supporting points. Pairs of supporting points $\mathbf{q}_{jk} \leftrightarrow \mathbf{q}'_{jk}$ are obtained based on the previously estimated local geometries $\mathsf{H}(\boldsymbol{\mu})$. The first idea that comes to mind is to use the supporting points as input to the eight point algorithm [7]. This algorithm minimizes an algebraic distance between predicted epipolar lines and observed points. The eight corrected point algorithm consists in correcting the position of the supporting points, i.e. to make them colinear, prior to applying the eight point algorithm. Using this procedure reduces the noise on the points positions, as we shall verify experimentally.

3.2 The 'Three POP' Algorithm

This linear algorithm compares observed points and predicted points. This algorithm is more statistically meaningful than the eight point algorithm, in the case of POPs, in that observed and predicted features are directly compared.

We wish to predict the supporting point positions. We intersect the predicted epipolar lines, i.e. $\mathsf{F}\mathbf{q}_{jk}$ in the second image, with the supporting lines $\mathbf{l}'_j$: the predicted point is given by $[\mathbf{l}'_j]_\times \mathsf{F}\mathbf{q}_{jk}$. Our cost function is given by summing the squared algebraic distances between observed and predicted points: $\sum_j d_a^2(\mathbf{q}'_{jk}, [\mathbf{l}'_j]_\times \mathsf{F}\mathbf{q}_{jk})$. In order to obtain a symmetric criterion, we consider predicted and observed points in the first image also, which yields:

$$\mathcal{C}_a = \sum_j \sum_k \left(d_a^2(\mathbf{q}_{jk}, [\mathbf{l}_j]_\times \mathsf{F}^\mathsf{T} \mathbf{q}'_{jk}) + d_a^2(\mathbf{q}'_{jk}, [\mathbf{l}'_j]_\times \mathsf{F}\mathbf{q}_{jk}) \right). \tag{3}$$

After introducing explicitly d_a from equation (1) and minor algebraic manipulations, we obtain the matrix form $\mathcal{C}_a = \sum_j \sum_k (\|\mathsf{B}_{jk}\mathbf{f}\|^2 + \|\mathsf{B}'_{jk}\mathbf{f}\|^2)$ where $\mathbf{f} = \text{vect}(\mathsf{F})$ is the row-wise vectorization of F and:

$$\mathsf{B}_{jk} = \mathsf{S}[\mathbf{q}_{jk}]_\times [\mathbf{l}_j]_\times \begin{pmatrix} q'_{jk,1}\mathrm{I} & q'_{jk,2}\mathrm{I} & q'_{jk,3}\mathrm{I} \end{pmatrix}, \mathsf{B}'_{jk} = \mathsf{S}[\mathbf{q}'_{jk}]_\times [\mathbf{l}'_j]_\times \text{diag}(\mathbf{q}_{jk}^\mathsf{T} \;\; \mathbf{q}_{jk}^\mathsf{T} \;\; \mathbf{q}_{jk}^\mathsf{T}).$$

The cost function becomes $\mathcal{C}_a = \|\mathsf{B}\mathbf{f}\|^2$ with $\mathsf{B}^\mathsf{T} \sim \begin{pmatrix} \mathsf{B}_{11}^\mathsf{T} & \mathsf{B}'_{11}{}^\mathsf{T} & \dots & \mathsf{B}_{mp}^\mathsf{T} & \mathsf{B}'_{mp}{}^\mathsf{T} \end{pmatrix}$. The singular vector associated to the smallest singular value of B gives the $\mathbf{f}$ that minimizes $\mathcal{C}_a$. Similarly to the eight point algorithm, the obtained fundamental matrix does not satisfy the rank-deficiency constraint in general, and has to be corrected by nullifying its smallest singular value, see e.g. [6].

[6] General position means that the supporting lines are not coplanar and do not lie on an epipolar plane, i.e. the image lines do not contain the epipoles.

3.3 Non-Linear 'Reduced' Estimation

The previously-described three POP estimator is statistically sound in the sense that observed and predicted points are compared in the linear cost function (3). However, the comparison is done using the algebraic distance d_a. This is the price to pay to get a linear estimator. In this section, we consider a cost function with a similar form, but using the Euclidean distance d_e to compare observed and predicted points:

$$\mathcal{C}_e = \sum_j \sum_k \left(d_e^2(\mathbf{q}_{jk}, [\mathbf{l}_j]_\times \mathsf{F}^\mathsf{T} \mathbf{q}'_{jk}) + d_e^2(\mathbf{q}'_{jk}, [\mathbf{l}'_j]_\times \mathsf{F} \mathbf{q}_{jk}) \right). \tag{4}$$

We use the Levenberg-Marquart algorithm, see e.g. [6], with a suitable parameterization of the fundamental matrix [12] to minimize this cost function, based on the initial solution provided by the three POP algorithm.

4 Multiple-View Triangulation

We deal with the triangulation of POP seen in multiple views. Note that since the triangulation of a line is independent from the others, we drop the index j in this section.

4.1 Optimal Triangulation

The optimal 3D POP is the one which better explains the data, i.e. which minimizes the sum of squared Euclidean distances between predicted and observed supporting points. Assuming that 3D POPs are represented by two points $\mathbf{M}$ and $\mathbf{N}$ for the supporting line and p scalars α_k for the supporting points $\mathbf{Q}_k \sim \alpha_k \mathbf{M} + (1 - \alpha_k)\mathbf{N}$, the following non-linear problem is obtained:

$$\min_{\mathbf{M},\mathbf{N},\ldots,\alpha_k,\ldots} \mathcal{C}_{pop} \text{ with } \mathcal{C}_{pop} = \sum_{i=1}^{n} \sum_{k=1}^{p} d_e^2(\mathsf{P}_i(\alpha_k \mathbf{M} + (1 - \alpha_k)\mathbf{N}), \mathbf{q}_{ik}). \tag{5}$$

We use the Levenberg-Marquart algorithm, e.g. [6]. We examine the difficult problem of finding a reliable initial solution in the next section.

4.2 Initialization

Finding an initial solution which is close to the optimal one is of primary importance. The initialization method must minimize a cost function as close as possible to (5). We propose a two-step initialization algorithm consisting in triangulating the supporting line, then each supporting point. Our motivations for these steps are explained while reviewing line triangulation below.

Line Triangulation Line triangulation from multiple views is a standard structure-from-motion problem and has been widely studied, see e.g. [5]. The optimal line $< \mathbf{M}, \mathbf{N} >$ is given by minimizing the sum of squared Euclidean distances between the predicted lines $(\mathsf{P}_i\mathbf{M}) \times (\mathsf{P}_i\mathbf{N})$ and the observed points $\mathbf{q}_{ik}$ as $\min_{\mathbf{M},\mathbf{N}} \sum_{i=1}^{n} \sum_{k=1}^{p} d_e^2((\mathsf{P}_i\mathbf{M}) \times (\mathsf{P}_i\mathbf{N}), \mathbf{q}_{ik})$. To make the relationship with the

cost function (5) appear, we introduce a set of points $\mathbf{Q}_{ik}$ on the 3D line. Using the fact that the Euclidean distance between a point and a line is equal to the Euclidean distance between the point and the projection of this point on the line, we rewrite the line triangulation problem as:

$$\min_{\mathbf{M},\mathbf{N},\ldots,\alpha_{ik},\ldots} \mathcal{C}_{line} \text{ with } \mathcal{C}_{line} = \sum_{i=1}^{n}\sum_{k=1}^{p} d_e^2(\mathsf{P}_i(\alpha_{ik}\mathbf{M} + (1-\alpha_{ik})\mathbf{N}), \mathbf{q}_{ik}). \quad (6)$$

Compare this cost function (5): the difference is that for line triangulation, the points are not supposed to match between the different views. Hence, a 3D point on the line is reconstructed for each image point, while in the POP triangulation problem, a 3D point on the line is reconstructed for each image point correspondence. Now, the interesting point is to determine if, in practice, cost functions (5) and (6) yield close solutions for the reconstructed 3D line. Obviously, an experimental study is necessary, and we refer to §6. However, we intuitively expect that the results are close.

Point-on-Line Triangulation We study the problem of point-on-line optimal triangulation: given a 3D line, represented by two 3D points $\mathbf{M}$ and $\mathbf{N}$, a set of corresponding image points $\ldots, \mathbf{q}_{ik}, \ldots$, find a 3D point $\mathbf{Q}_k \sim \alpha_k\mathbf{M}+(1-\alpha_k)\mathbf{N}$ on the given 3D line, such that the squared Euclidean distances between the predicted and the observed points is minimized.

For point-on-line triangulation, we formalise the problem as $\min_{\alpha_k} \sum_{i=1}^{n} d_e^2(\mathsf{P}_i(\alpha_k\mathbf{M} + (1-\alpha_k)\mathbf{N}), \mathbf{q}_{ik})$ and by introducing $\mathbf{b}_i = \mathsf{P}_i(\mathbf{M}-\mathbf{N})$ and $\mathbf{d}_i = \mathsf{P}_i\mathbf{N}$, we obtain:

$$\min_{\alpha_k} \mathcal{C}_{pol} \text{ with } \mathcal{C}_{pol} = \sum_{i=1}^{n} d_e^2(\alpha_k\mathbf{b}_i + \mathbf{d}_i, \mathbf{q}_{ik}). \quad (7)$$

Sub-optimal linear algorithm. We give a linear algorithm, based on approximating the optimal cost function (7) by replacing the Euclidean distance d_e by the algebraic distance d_a. The algebraic cost function is $\sum_{i=1}^{n} d_a^2(\alpha_k\mathbf{b}_i + \mathbf{d}_i, \mathbf{q}_{ik}) = \sum_{i=1}^{n} \|\alpha_k\mathsf{S}[\mathbf{q}_{ik}]_\times\mathbf{b}_i + \mathsf{S}[\mathbf{q}_i]_\times\mathbf{d}_i\|^2$. A closed-form solution giving the best α_k in the least-squares sens is $\alpha_k = -\frac{\sum_{i=1}^{n}\mathbf{b}_i{}^\mathsf{T}[\mathbf{q}_{ik}]_\times\tilde{\mathsf{I}}[\mathbf{q}_{ik}]_\times\mathbf{d}_i}{\sum_{i=1}^{n}\mathbf{b}_i{}^\mathsf{T}[\mathbf{q}_{ik}]_\times\tilde{\mathsf{I}}[\mathbf{q}_{ik}]_\times\mathbf{b}_i}$ with $\tilde{\mathsf{I}} \sim \mathsf{S}^\mathsf{T}\mathsf{S} \sim \begin{pmatrix} 1 & & \\ & 1 & \\ & & 0 \end{pmatrix}$.

Optimal polynomial algorithm. This algorithm consists in finding the roots of a degree-$(3n-2)$ polynomial in the parameter α_k, whose coefficients depend on the $\mathbf{b}_i$, the $\mathbf{d}_i$ and the $\mathbf{q}_{ik}$. Due to lack of space, details are left to an extended version of the paper.

5 Bundle Adjustment

Bundle adjustment consists in minimizing the reprojection error over structure and motion parameters:

$$\min_{\mathsf{P}_1,\ldots,\mathsf{P}_n,\mathbf{M}_1,\mathbf{N}_1,\ldots,\mathbf{M}_m,\mathbf{N}_m,\ldots,\alpha_{jk},\ldots} \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{p} d_e^2(\mathsf{P}_i(\alpha_{jk}\mathbf{M}_j + (1-\alpha_{jk})\mathbf{N}_j), \mathbf{q}_{ijk}),$$

where we consider without loss of generality that all points are visible in all views. We use the Levenberg-Marquardt algorithm to minimize this cost function, starting from an initial solution obtained by matching pairs of images and computing pair-wise fundamental matrices using the algorithms of §§2 and 3, from which the multiple-view geometry is extracted as in [11]. Multiple-view matches are formed, and the POPs are triangulated using the optimal method described in §4.

6 Experimental Results

We simulate a set of 3D POPs observed by two cameras, with focal length 1000 pixels. To simulate a realistic scenario, each POP is made of 5 supporting points. The supporting points are projected onto the images, and a Gaussian centered noise is added. The images of the supporting lines are determined as the best fit to the noisy supporting points. These data are used to compare quasi-metric reconstructions of the scene, obtained using different algorithms. We mesure the reprojection error and a 3D error, obtained as the minimum residual of $\min_{\mathsf{H}_u} \sum_j d^2(\underline{\mathbf{Q}}_j, \mathsf{H}_u \mathbf{Q}_j)$, where $\underline{\mathbf{Q}}_j$ are the groung truth 3D points, $\mathbf{Q}_j$ the reconstruction points and H_u an aligning 3D homography.

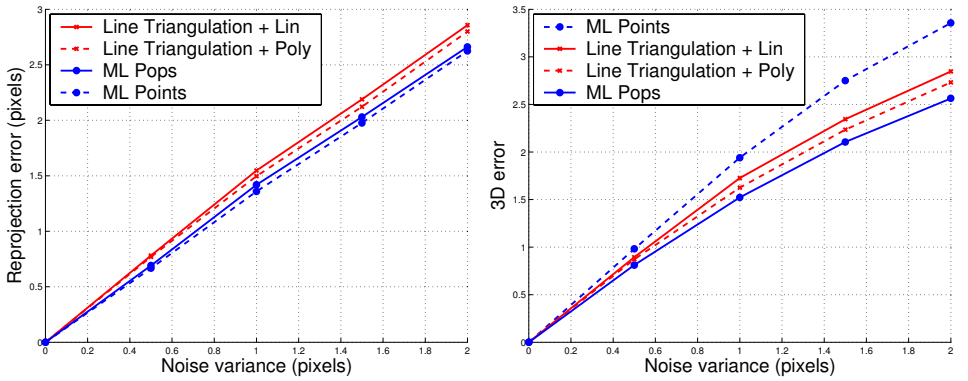


Fig. 3. Reprojection and 3D error when varying the added image noise variance to compare structure and motion recovery methods.

Comparing triangulation algorithms. The two first methods are based on triangulating the supporting line, then each supporting point using the linear solution (method 'Line Triangulation + Lin') or using the optimal polynomial solution (method 'Line Triangulation + Poly'). The third method is Levenberg-Marquardt minimization of the reprojection error, for POPs (method 'ML Pops') or points (method 'ML Points'). We observe on figure 3 (a) that triangulating the supporting line followed by the supporting points on this line (methods 'Line Triangulation + *') produce results close to the non-linear minimization of the reprojection

error of the reprojection error of the POP (method 'ML Pops'). Minimizing the reprojection error individually for each point (method 'ML Points') produce lower reprojection errors.

Concerning the 3D error, shown on figure 3 (b), we also observe that methods 'Line Triangulation + *' produce results close to method 'ML Pop'. However, we observe that method 'ML Points' gives results worse than all other methods. This is due to the fact that this method does not benefit from the structural constraints defining POPs.

Comparing bundle adjustment algorithms. The two first methods are based on computing the epipolar geometry using the eight point algorithm (method 'Eight Point Alg.') or the three POP algorithm (method 'Three Pop Alg.'), then triangulating the POPs using the optimal triangulation method. The two other methods are bundle adjustment of POPs and points respectively. We observe on figure 4 (a)

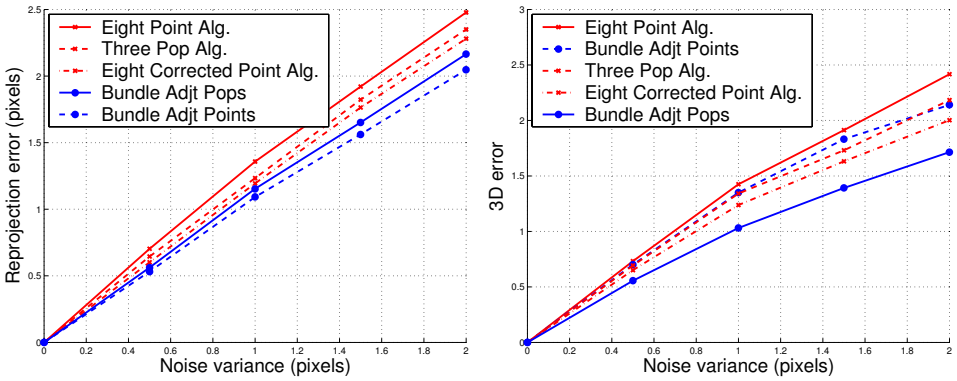


Fig. 4. Reprojection and 3D error when varying the added image noise variance to compare triangulation methods.

that the eight point algorithm yields the worse reprojection error, followed by the three POP algorithm and the eight corrected point algorithm. Bundle adjustement of POPs gives reprojection error slightly higher than with points. However, figure 4 (b) shows that bundle adjustment of POPs gives a better 3D structure than point, due to the structural constraints. It also shows that the eight corrected point algorithm yields good results.

7 Conclusions and Further Work

We addressed the problem of automatic structure and motion recovery from images containing lines. We introduced a feature that we call POP, for Pencil-of-Points. We demonstrated our matching algorithm on real images. This confirms that the repeatability rate of POPs is higher than the repeatability rates of the points and

lines from which they are detected. This also shows that using POPs, wide baseline matching and the epipolar geometry can be successfully computed in an automatic manner, using simple cross-correlation. Experimental results on simulated data show that due to the strong structural constraints, POPs yield structure and motion estimates more accurate than with points.

Advantages for using POPs are numerous. Briefly, localization, repeatability rate and structure and motion estimate are better with POPs than with points, and robust estimation is very efficient since only three pairs of POPs define an epipolar geometry. For this reason, we believe that this new feature could become standard for automatic structure-and-motion in man-made environment, i.e. based on lines.

Further work will consist in investigating the determination of parameters $\boldsymbol{\mu}$ needed to compute undistorted cross-correlation, since we believe that it could strongly improve the initial matching step, and studying methods for estimating the trifocal tensor from triplets of POPs.

Acknowledgements. The first author would like to thank Frederik Schaffalitzky from the University of Oxford for fruitful discussions. This paper benefited from suggestions from one of the anonymous reviewers. Images of the Valbonne church have been provided by INRIA Sophia-Antipolis.

References

1. F. Bignone, O. Henricsson, P. Fua, and M. Stricker. Automatic extraction of generic house roofs from high resolution aerial imagery. In *ECCV*, pp.85–96. April 1996.
2. J. Canny. A computational approach to edge detection. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.
3. M.A. Fischler and R.C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Graphics and Image Processing*, 24(6):381 – 395, June 1981.
4. R. Hartley and P. Sturm. Triangulation. *Computer Vision and Image Understanding*, 68(2):146–157, 1997.
5. R.I. Hartley. Lines and points in three views and the trifocal tensor. *International Journal of Computer Vision*, 22(2):125–140, 1997.
6. R.I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, June 2000.
7. H.C. Longuet-Higgins. A computer program for reconstructing a scene from two projections. *Nature*, 293:133–135, September 1981.
8. G. Médioni and R. Nevatia. Segment-based stereo matching. *Computer Vision, Graphics and Image Processing*, 31:2–18, 1985.
9. K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. In *ECCV*, volume I, pages 128–142, May 2002.
10. C. Schmid and A. Zisserman. Automatic line matching across views. In *CVPR*, pages 666–671, 1997.
11. B. Triggs. Linear projective reconstruction from matching tensors. *Image and Vision Computing*, 15(8):617–625, August 1997.
12. Z. Zhang. Determining the epipolar geometry and its uncertainty: A review. *International Journal of Computer Vision*, 27(2):161–195, March 1998.

8.2.2 Paper (IVC'08) – *Triangulation for Points on Lines*

Available online at www.sciencedirect.com
ScienceDirect
Image and Vision Computing 26 (2008) 315–324
www.elsevier.com/locate/imavis

Triangulation for points on lines

Adrien Bartoli *, Jean-Thierry Lapresté

LASMEA – CNRS/Université Blaise Pascal, Clermont-Ferrand, France

Received 13 May 2006; received in revised form 23 February 2007; accepted 1 June 2007

Abstract

Triangulation consists in finding a 3D point reprojecting the best as possible onto corresponding image points. It is classical to minimize the reprojection error, which, in the pinhole camera model case, is nonlinear in the 3D point coordinates. We study the triangulation of points lying on a 3D line, which is a typical problem for Structure-From-Motion in man-made environments. We show that the reprojection error can be minimized by finding the real roots of a polynomial in a single variable, which degree depends on the number of images. We use a set of transformations in 3D and in the images to make the degree of this polynomial as low as possible, and derive a practical reconstruction algorithm. Experimental comparisons with an algebraic approximation algorithm and minimization of the reprojection error using Gauss–Newton are reported for simulated and real data. Our algorithm finds the optimal solution with high accuracy in all cases, showing that the polynomial equation is very stable. It only computes the roots corresponding to feasible points, and can thus deal with a very large number of views – triangulation from hundreds of views is performed in a few seconds. Reconstruction accuracy is shown to be greatly improved compared to standard triangulation methods that do not take the line constraint into account.
© 2007 Elsevier B.V. All rights reserved.

Keywords: Triangulation; Structure-From-Motion; Point; Line

1. Introduction

Triangulation is one of the main building blocks of Structure-From-Motion algorithms. Given image feature correspondences and camera matrices, it consists in finding the position of the underlying 3D feature, by minimizing some error criterion. This criterion is often chosen as the reprojection error – the Maximum Likelihood criterion for a Gaussian, centered and *i.i.d.* noise model on the image point positions – though other criteria are possible [5,9,10].

Traditionally, triangulation is carried out by some sub-optimal procedure and is then refined by local optimization, see *e.g.* [7]. A drawback of this is that convergence to the optimal solution is not guaranteed. Optimal procedures for triangulating points from two and three views were proposed in [6,13].

We address the problem of triangulating points lying on a line, that is, given image point correspondences, camera matrices and a 3D line, finding the 3D point lying on the 3D line, such that the reprojection error is minimized.

Our main contribution is to show that the problem can be solved by computing the real roots of a degree-$(3n-2)$ polynomial, where n is the number of views. Extensive experiments on simulated data show that the polynomial is very well balanced since large number of views and large level of noise are handled. The method is valid whatever the calibration level of the cameras is – projective, affine, metric or Euclidean.

One may argue that triangulating points on a line only has a theoretical interest since in practice, triangulating a line from multiple views is done by minimizing the reprojection error over its supporting points which 3D positions are hence reconstructed along with the 3D line. Indeed, most work consider the case where the supporting points do *not* match across the images, see *e.g.* [3]. When one identifies correspondences of supporting points accross the

* Corresponding author.
E-mail address: Adrien.Bartoli@gmail.com (A. Bartoli).

0262-8856/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.imavis.2007.06.003

images, it is fruitful to incorporate these constraints into the bundle adjustment, as is demonstrated by our experiments. This is typically the case in man-made environments, where one identifies, *e.g.* matching corners at the meet of planar facades or around windows. Bartoli et al. [2] dubbed Pencil-of-Points or 'POP' this type of features. In order to find an initial 3D reconstruction, a natural way is to compute the 3D line by some means (*e.g.* by ignoring the matching constraints of the supporting points, from 3D primitives such as the intersection of two planes, or from a registered wireframe CAD model) and then to triangulate the supporting point correspondences using point on line triangulation. The result can then be plugged into a bundle adjustment incorporating the constraints.

We review some related work in Section 2. Our triangulation method is derived in Section 3. A linear least squares method minimizing an algebraic distance is provided in Section 4. Gauss–Newton refinement is summarized in Section 5. Experimental results are reported in Section 6 and our conclusions in Section 7.

Notation. Vectors are written using bold fonts, *e.g.* $\mathbf{q}$, and matrices using sans-serif fonts, *e.g.* P. Almost everything is homogeneous, *i.e.* defined up to scale. Equality up to scale is denoted $\sim$. The inhomogenous part of a vector is denoted using a bar, *e.g.* $\mathbf{q}^\mathsf{T} \sim (\bar{\mathbf{q}}^\mathsf{T}\ 1)$ where T is transposition. Index $i = 1,\ldots,n$, and sometime j are used for the images. The point in the ith image is $\mathbf{q}_i$. Its elements are $\mathbf{q}_i^\mathsf{T} \sim (q_{i,1}\ q_{i,2}\ 1)$. The 3D line joining points $\mathbf{M}$ and $\mathbf{N}$ is denoted $(\mathbf{M},\mathbf{N})$. The $\mathcal{L}_2$-norm of a vector is denoted as in $\|\mathbf{x}\|^2 = \mathbf{x}^\mathsf{T}\mathbf{x}$. The Euclidean distance measure d_e is defined by

$$d_e^2(\mathbf{x},\mathbf{y}) = \left\|\frac{\mathbf{x}}{x_3} - \frac{\mathbf{y}}{y_3}\right\|^2 = \left(\frac{x_1}{x_3} - \frac{y_1}{y_3}\right)^2 + \left(\frac{x_2}{x_3} - \frac{y_2}{y_3}\right)^2. \quad (1)$$

2. Related work

Optimal procedures for triangulating points in 3D space, and points lying on a plane were previously studied, as summarized in Table 1. Hartley and Sturm [6] showed that triangulating points in 3D space from two views, in other words finding a pair of points satisfying the epipolar geometry and lying as close as possible to the measured points, can be solved by finding the real roots of a degree-6 polynomial. The optimal solution is then selected by straightforward evaluation of the reprojection error. Stewénius et al. [13] extended the method to three views. The optimal solution is one of the real roots of a system of 3 degree-6 polynomials in the three coordinates of the point. Chum et al. [4] show that triangulating points lying on a plane, in other words finding a pair of points satisfying an homography and lying as close as possible to the measured points, can be solved by finding the real roots of a degree-8 polynomial.

Error functions different from the reprojection error were considered in the literature. The directional error in two views is proposed in [10], along with a triangulation method for calibrated cameras. The $\mathcal{L}_\infty$-norm is considered in [5,9], instead of the usual $\mathcal{L}_2$-norm. A triangulation method for two views is given in [9], while it is shown in [5] that the n-view case can be cast as a convex optimization problem Table 2.

3. Minimizing the reprojection error

We derive our optimal triangulation algorithm for point on line, dubbed 'POLY'.

3.1. Problem statement and parameterization

We want to compute a 3D point $\mathbf{Q}$, lying on a 3D line $(\mathbf{M},\mathbf{N})$, represented by two 3D points $\mathbf{M}$ and $\mathbf{N}$. The (3×4) perspective camera matrices are denoted P_i with $i = 1,\ldots,n$ the image index. The problem is to find the point $\hat{\mathbf{Q}}$ such that

$$\hat{\mathbf{Q}} \sim \arg\min_{\mathbf{Q}\in(\mathbf{M},\mathbf{N})} \mathcal{C}_n^2(\mathbf{Q}),$$

where $\mathcal{C}_n$ is the n-view *reprojection error*

$$\mathcal{C}_n^2(\mathbf{Q}) = \sum_{i=1}^{n} d_e^2(\mathbf{q}_i, \mathsf{P}_i\mathbf{Q}). \quad (2)$$

We parameterize the point $\mathbf{Q}\in(\mathbf{M},\mathbf{N})$ using a single parameter $\lambda \in \mathbb{R}$ as

$$\mathbf{Q} \sim \lambda\mathbf{M} + (1-\lambda)\mathbf{N} \sim \lambda(\mathbf{M}-\mathbf{N}) + \mathbf{N}. \quad (3)$$

Introducing this parameterization into the reprojection error (2) yields

$$\mathcal{C}_n^2(\lambda) = \sum_{i=1}^{n} d_e^2(\mathbf{q}_i, \mathsf{P}_i(\lambda(\mathbf{M}-\mathbf{N}) + \mathbf{N})).$$

Defining $\mathbf{b}_i = \mathsf{P}_i(\mathbf{M}-\mathbf{N})$ and $\mathbf{d}_i = \mathsf{P}_i\mathbf{N}$, we get

$$\mathcal{C}_n^2(\lambda) = \sum_{i=1}^{n} d_e^2(\mathbf{q}_i, \lambda\mathbf{b}_i + \mathbf{d}_i). \quad (4)$$

Table 1
Different types of triangulation and methods minimizing the $\mathcal{L}_2$-norm reprojection error

Type of triangulation	Number of views	Polynomial system			Reference
		Number	Degree	Variables	
Point in 3D space	2	1	6	1	[6]
	3	3	6/6/6	3	[13]
Point on plane	2	1	8	1	[4]
Point on line	1	1	1	1	This paper
	2	1	4	1	
	3	1	7	1	
	4	1	10	1	
	n	1	$3n-2$	1	

The number of polynomials to be solved, their degrees and the number of variables is given in the column 'Polynomial system'.

Table 2
The proposed point on line triangulation algorithm 'POLY'

Objective

Given a point correspondence $\mathbf{q}_i$ over $n \geqslant 1$ views ($i = 1, \ldots, n$), a 3D line $(\mathbf{M}, \mathbf{N})$ and camera matrices P_i, compute the 3D point $\hat{\mathbf{Q}}$ lying on $(\mathbf{M}, \mathbf{N})$ such that the reprojection error e in all images is minimized.

Algorithm

- *Canonical 3D coordinate frame*. Express the 3D line and the cameras in a canonical 3D coordinate frame

$$\mathsf{H} \leftarrow \begin{pmatrix} & \mathsf{P}_1 & \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \mathsf{P}_i \leftarrow \mathsf{P}_i\mathsf{H}^{-1} \quad \mathbf{M} \leftarrow \mathsf{H}\mathbf{M} \quad \mathbf{N} \leftarrow \mathsf{H}\mathbf{N}.$$

Normalize the homogeneous coordinates: $\mathbf{M} \leftarrow \mathbf{M}/M_3$ and $\mathbf{N} \leftarrow \mathbf{N}/N_3$.

- *Line reparameterization*. Reparameterize the 3D line by shifting point $\mathbf{N}$ such that it projects to a finite point in every views

$$\mathbf{N} \leftarrow \frac{(\mathsf{P}_1\mathbf{N} - \mathbf{q}_1)^{\mathsf{T}}\mathsf{P}_1(\mathbf{M} - \mathbf{N})}{\|\mathsf{P}_1(\mathbf{M} - \mathbf{N})\|^2}(\mathbf{M} - \mathbf{N}) + \mathbf{N}.$$

Project the 3D line onto the images: $\mathbf{b}_i \leftarrow \mathsf{P}_i(\mathbf{M} - \mathbf{N})$ and $\mathbf{d}_i \leftarrow \mathsf{P}_i\mathbf{N}$.

- *Rigid image transformations*. Align the projected line with the y-axis in each view such that point $\mathbf{N}$ projects to the origin:

$$\mathbf{t}_i \leftarrow \bar{\mathbf{d}}_i/d_{i,3} \quad \mathbf{r}_i \leftarrow \bar{\mathbf{b}}_i - b_{i,3}\mathbf{t}_i \quad \mathsf{R}_i \leftarrow \begin{pmatrix} r_{i,2} & -r_{i,1} \\ r_{i,1} & r_{i,2} \end{pmatrix} \Big/ \|\bar{\mathbf{r}}_i\| \quad \mathsf{T}_i \leftarrow \begin{pmatrix} \mathsf{R}_i & -\mathsf{R}_i\mathbf{t}_i \\ \mathbf{0}^{\mathsf{T}} & 1 \end{pmatrix} \mathbf{d}_i \leftarrow \mathsf{T}_i\mathbf{d}_i \quad \mathbf{b}_i \leftarrow \mathsf{T}_i\mathbf{b}_i \quad \mathsf{P}_i \leftarrow \mathsf{T}_i\mathsf{P}_i \quad \mathbf{q}_i \leftarrow \mathsf{T}_i\mathbf{q}_i.$$

- *Solving*. See Section 3.3 for how to find the real roots λ_k of the polynomial $\tilde{\mathcal{D}}(\lambda)$ given by Eq. (5). Select the root $\hat{\lambda}$ for which the reprojection error is minimized: $\hat{\lambda} = \arg\min_k \mathcal{C}_n^2(\lambda_k)$.
- *Finishing*. Compute the mean reprojection error $e = \sqrt{\frac{1}{n}\mathcal{C}_n^2(\hat{\lambda})}$ and recover the 3D point in the original coordinate frame: $\hat{\mathbf{Q}} \sim \mathsf{H}^{-1}(\hat{\lambda}\mathbf{M} + (1 - \hat{\lambda})\mathbf{N})$.

Note that a similar parameterization can be derived by considering the inter-image homographies induced by the 3D line [12]. The main motivation to reducing the number of parameters with parameterization (3) instead of using, *e.g.* a Lagrange multiplier for the point on line constraint, is that it allows us to find the global minimum of the reprojection error through a simple polynomial formulation.

3.2. Simplification

We simplify the expression (4) of the reprojection error by changing the 3D coordinate frame and the image coordinate frames. This is intended to lower the degree of the polynomial equation that will ultimately have to be solved. Since the reprojection error is based on Euclidean distances measured in the images, only rigid image transformations are allowed to keep invariant the error function, while full projective homographies can be used in 3D. We thus setup a standard canonical 3D coordinate frame, see *e.g.* [8], such that the first camera matrix becomes $\mathsf{P}_1 \sim (\mathsf{I}\ \mathbf{0})$. Note that using a projective basis does not harm Euclidean triangulation since the normalization is undone once the point is triangulated. The canonical basis is setup by the following simple operations:

$$\mathsf{H} \leftarrow \begin{pmatrix} & \mathsf{P}_1 & \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \mathsf{P}_i \leftarrow \mathsf{P}_i\mathsf{H}^{-1} \quad \mathbf{M} \leftarrow \mathsf{H}\mathbf{M} \quad \mathbf{N} \leftarrow \mathsf{H}\mathbf{N}.$$

Within this coordinate frame, we can write $\mathbf{M}^{\mathsf{T}} = (\bullet\ \bullet\ 1\ \bullet)$ and $\mathbf{N}^{\mathsf{T}} = (\bullet\ \bullet\ 1\ \bullet)$ without loss of generality, as pointed out in [7, Section A6], from which we get

$$\mathbf{b}_1 = \mathsf{P}_1(\mathbf{M} - \mathbf{N}) = (b_{1,1}\ b_{1,2}\ 0)^{\mathsf{T}},$$
$$\mathbf{d}_1 = \mathsf{P}_1\mathbf{N} = (d_{1,1}\ d_{1,2}\ 1)^{\mathsf{T}}.$$

We then apply a rigid transformation T_i in each image defined such that $\mathsf{T}_i\mathbf{b}_i$ lies on the y-axis and such that $\mathsf{T}_i\mathbf{d}_i = \mathsf{T}_i\mathsf{P}_i\mathbf{N}$ lies at the origin. This requires that point $\mathbf{N}$ does not project at infinity in any of the images. We ensure this by constraining $\mathbf{N}$ to project as close as possible to one of the image points,[1] say $\mathbf{q}_1$. The reprojection error (4) for the first view is $\mathcal{C}_1^2(\lambda) = d_e^2(\mathbf{q}_1, \lambda\mathbf{b}_1 + \mathbf{d}_1) = \|\lambda\bar{\mathbf{b}}_1 + \bar{\mathbf{d}}_1 - \bar{\mathbf{q}}_1\|^2$. We compute λ as the solution of $\frac{\partial\mathcal{C}_1^2}{\partial\lambda} = 0$, which gives, after some minor calculations, $\lambda = (\bar{\mathbf{q}}_1 - \bar{\mathbf{d}}_1)^{\mathsf{T}}\bar{\mathbf{b}}_1/\|\bar{\mathbf{b}}_1\|^2$. Substituting in Eq. (3) yields the following operations:

$$\mathbf{N} \leftarrow \frac{(\mathsf{P}_1\mathbf{N} - \mathbf{q}_1)^{\mathsf{T}}\mathsf{P}_1(\mathbf{M} - \mathbf{N})}{\|\mathsf{P}_1(\mathbf{M} - \mathbf{N})\|^2}(\mathbf{M} - \mathbf{N}) + \mathbf{N}.$$

Obviously, the $\mathbf{d}_i = \mathsf{P}_i\mathbf{N}$ must be recomputed. These simplifications lead to

$$\begin{cases} \mathbf{b}_1 & = & (0\ b_{1,2}\ 0)^{\mathsf{T}}, \\ \mathbf{d}_1 & = & (0\ 0\ 1)^{\mathsf{T}}, \\ \mathbf{b}_{i>1} & = & (0\ b_{i,2}\ b_{i,3})^{\mathsf{T}}, \\ \mathbf{d}_{i>1} & = & (0\ 0\ d_{i,3})^{\mathsf{T}}. \end{cases}$$

The rigid transformations T_i are quickly derived below. For each image i, we look for T_i mapping $\mathbf{d}_i$ to the origin,

[1] Note that this is equivalent to solving the single view triangulation problem. Point $\mathbf{N}$ does not project at infinity in any of the views since both point $\mathbf{q}_1$ and the supporting line have observable correspondences in all images.

and $\mathbf{b}_i$ to a point on the y-axis. We decompose T_i as a rotation around the origin and a translation

$$\mathsf{T}_i = \begin{pmatrix} \mathsf{R}_i & \mathbf{0} \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \begin{pmatrix} \mathsf{I} & -\mathbf{t}_i \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix}.$$

The translation is directly given from $\mathsf{T}_i\mathbf{d}_i \sim (0\ 0\ 1)^\mathsf{T}$ as $\mathbf{t}_i = \bar{\mathbf{d}}_i/d_{i,3}$. For the rotation, we consider $\mathsf{T}_i\mathbf{b}_i \sim (0\ \bullet\ \bullet)^\mathsf{T}$, from which, setting $\mathbf{r}_i = \bar{\mathbf{b}}_i - b_{i,3}\mathbf{t}_i$, we obtain

$$\mathsf{R}_i = \begin{pmatrix} r_{i,2} & -r_{i,1} \\ r_{i,1} & r_{i,2} \end{pmatrix} / \|\bar{\mathbf{r}}_i\|.$$

This leads to the following expression for the reprojection error (4) where we separated the leading term:

$$\begin{aligned} \mathcal{C}_n^2(\lambda) &= d_e^2(\mathbf{q}_1, \lambda\mathbf{b}_1 + \mathbf{d}_1) + \sum_{i=2}^{n} d_e^2(\mathbf{q}_i, \lambda\mathbf{b}_i + \mathbf{d}_i), \\ &= q_{1,1}^2 + (\lambda b_{1,2} - q_{1,2})^2 \\ &\quad + \sum_{i=2}^{n} \left(q_{i,1}^2 + \left(\frac{\lambda b_{i,2}}{\lambda b_{i,3} - d_{i,3}} - q_{i,2} \right)^2 \right). \end{aligned}$$

The constant terms $q_{1,1}^2$ and $q_{i,1}^2$ represent the vertical counterparts of the point to line distance in the images. This means that only the errors along the lines are to be minimized.

3.3. Solving the polynomial equation

Looking for the minima of the reprojection error $\mathcal{C}_n^2$ is equivalent to finding the roots of its derivative, *i.e.* solving $\frac{\partial \mathcal{C}_n^2}{\partial \lambda} = 0$. Define $\mathcal{D}_n = \frac{1}{2}\frac{\partial \mathcal{C}^2}{\partial \lambda}$:

$$\begin{aligned} \mathcal{D}_n(\lambda) &= (\lambda b_{1,2} - q_{1,2}) b_{1,2} \\ &\quad + \sum_{i=2}^{n} \left(\frac{\lambda b_{i,2}}{\lambda b_{i,3} + d_{i,3}} - q_{i,2} \right) \left(\frac{b_{i,2} d_{i,3}}{(\lambda b_{i,3} + d_{i,3})^2} \right). \end{aligned}$$

This is a nonlinear function. Directly solving $\mathcal{D}_n(\lambda) = 0$ is therefore very difficult in general. We thus define $\tilde{\mathcal{D}}_n(\lambda) = \mathcal{D}_n(\lambda)\mathcal{K}_n(\lambda)$, where we choose $\mathcal{K}_n$ in order to cancel out the denominators including λ in $\mathcal{D}_n$. Finding the zeros of $\tilde{\mathcal{D}}_n$ is thus equivalent to finding the zeros of $\mathcal{D}_n$. Inspecting the expression of $\mathcal{D}_n$ reveals that $\mathcal{K}_n(\lambda) = \prod_{i=2}^{n} (\lambda b_{i,3} + d_{i,3})^3$ does the trick.

$$\begin{aligned} \tilde{\mathcal{D}}_n(\lambda) &= (\lambda b_{1,2} - q_{1,2}) b_{1,2} \prod_{i=2}^{n} (\lambda b_{i,3} + d_{i,3})^3 \\ &\quad + \sum_{i=2}^{n} \Bigg(b_{i,2} d_{i,3} (\lambda b_{i,2} - q_{i,2} (\lambda b_{i,3} + d_{i,3})) \\ &\quad \times \prod_{j=2, j\neq i}^{n} (\lambda b_{j,3} + d_{j,3})^3 \Bigg). \end{aligned} \tag{5}$$

As expected, $\tilde{\mathcal{D}}_n$ is a polynomial function, whose degree depends on the number of images n. We observe that cancelling the denominator out for the contribution of each $(i > 1)$-image requires to multiply $\mathcal{D}_n$ by a cubic, namely $(\lambda b_{i,3} + d_{i,3})^3$. Since the polynomial required for image $i = 1$ is linear, the degree of the polynomial to solve is $3(n-1) + 1 = 3n - 2$.

Given the real roots λ_k of $\tilde{\mathcal{D}}_n(\lambda)$, that we compute as detailed below for different number of images, we simply select the one for which the reprojection error is minimized, *i.e.* $\hat{\lambda} = \arg\min_k \mathcal{C}_n^2(\lambda_k)$, substitute it in Eq. (3) and transfer the recovered point back to the original coordinate frame

$$\hat{\mathbf{Q}} \sim \mathsf{H}^{-1}(\hat{\lambda}\mathbf{M} + (1 - \hat{\lambda})\mathbf{N}).$$

A single image. For $n = 1$ image, the point is triangulated by projecting its image onto the image projection of the line. The intersection of the associated viewing ray with the 3D line gives the 3D point. In our framework, Eq. (5) is indeed linear in λ for $n = 1$

$$\tilde{\mathcal{D}}_1(\lambda) = (\lambda b_{1,2} - q_{1,2}) b_{1,2} = b_{1,2}^2 \lambda - q_{1,2} b_{1,2}.$$

A pair of images. For $n = 2$ images, Eq. (5) gives

$$\begin{aligned} \tilde{\mathcal{D}}_2(\lambda) &= (\lambda b_{1,2} - q_{1,2}) b_{1,2} (\lambda b_{2,3} + d_{2,3})^3 + b_{2,2} d_{2,3} (\lambda b_{2,2} \\ &\quad - q_{2,2} (\lambda b_{2,3} + d_{2,3})), \end{aligned}$$

which is a quartic in λ that can be solved in closed-form using Cardano's formulas: $\tilde{\mathcal{D}}_2(\lambda) \sim \sum_{d=1}^{4} c_d \lambda^d$, with:

$$\begin{cases} c_0 &= -q_{2,2} d_{2,3}^2 b_{2,2} - b_{1,2} q_{1,2} d_{2,3}^3, \\ c_1 &= d_{2,3} (b_{2,2}^2 - 3 b_{1,2} q_{1,2} b_{2,3} d_{2,3} + b_{1,2}^2 d_{2,3}^2 - q_{2,2} b_{2,3} b_{2,2}), \\ c_2 &= 3 b_{1,2} b_{2,3} d_{2,3} (b_{1,2} d_{2,3} - q_{1,2} b_{2,3}), \\ c_3 &= b_{1,2} b_{2,3}^2 (3 b_{1,2} d_{2,3} - q_{1,2} b_{2,3}), \\ c_4 &= b_{1,2}^2 b_{2,3}^3. \end{cases}$$

Multiple images. Solving the $n \geqslant 3$ view case is done in two steps. The first step is to compute the coefficients c_j, $j = 0, \ldots, 3n-2$ of a polynomial. The second step is to compute its real roots. Computing the coefficients in closed-form from Eq. (5), as is done above for the single- and the two-view cases, lead to very large, awkward formulas, which may lead to roundoff errors. We thus perform a numerical computation, while reparameterizing the polynomial, as described below.

A standard root-finding technique is to compute the eigenvalues of the $((3n-2) \times (3n-2))$ companion matrix of the polynomial, see *e.g.* [1]. Computing all the roots ensures the optimal solution to be found. This can be done if the number of images is not too large, *i.e.* lower than 100, and if computation time is not an issue. However, for large numbers of images, or if real-time computation must be achieved, it is not possible to compute and try all roots. In that case, we propose to compute only the roots corresponding to feasible points.

Let λ_0 be an approximation of the sought-after root. For example, one can take the result of the algebraic method of Section 4, or even $\lambda_0 = 0$ since our parameterization takes the sought-after root very close to 0. Obviously, we could launch an iterative root-finding procedure such as Newton–Raphson from λ_0 but this would not guarantee that the optimal solution is found.

One solution to efficiently compute only the feasible roots is to reparameterize the polynomial such that those lie close to 0, and use an iterative algorithm for computing the eigenvalues of the companion matrix on turn. For example, Arnoldi or Lanczos' methods, compute the eigenvalues with increasing magnitude starting from the smallest one. Let λ_c be the last computed eigenvalue, and $\mathbf{Q}_1$ and $\mathbf{Q}_2$ the reconstructed points corresponding to λ_c and $-\lambda_c$. If both $\mathbf{Q}_1$ and $\mathbf{Q}_2$ reproject outside the images, the computation is stopped. Indeed, the next root that would be computed would have greater magnitude than λ_c, and would obviously lead to a point reprojecting further away than the previous one outside the images.

The reparameterization is done by computing a polynomial $\mathcal{P}_n(\lambda) = \tilde{\mathcal{D}}_n(\lambda + \lambda_0)$. A simple way to achieve this reparameterization is to estimate the coefficients c_j, $j = 1, \ldots, 3n-1$, of $\mathcal{P}_n$, as follows. We evaluate $z \geqslant 3n-1$ values $v_k = \tilde{\mathcal{D}}_n(\lambda_k + \lambda_0)$ from Eq. (5) for $\lambda_k \in [-\delta, \delta]$, and solve the associated Vandermonde system

$$\sum_{j=0}^{3n-2} c_j \lambda_k^j = v_k \quad \text{for } k = 1, \ldots, z.$$

We typically use $z = 10(3n-1)$. The parameter $\delta \in \mathbb{R}^{*+}$ reflects the size of the sampling interval around λ_0. We noticed that this parameter does not influence the results, and typically chose $\delta = 1$. Obviously, in theory, using $z = 3n-1$, *i.e.* the minimum number of samples, at distinct points, is equivalent for finding the coefficients. However we experimentally found that using extra samples evenly spread around the expected root λ_0 has the benefit of 'averaging' the roundoff error, and stabilizes the computation.

One could argue that with this method for estimating the coefficients, the simplifying transformations of Section 3.2 are not necessary. A short calculation shows that this is partly true since if the canonical 3D projective basis were not used along with the normalization of the third entries of $\mathbf{M}$ and $\mathbf{N}$ to unity, then the degree of the polynomial would be $3n$ instead of $3n-2$. While this makes little difference for large n, this is important, *e.g.* for finding a closed-form solution in the two-view case. Such low n cases are likely to be embedded in RANSAC schemes, making triangulation time critical.

4. An algebraic criterion

We give a linear algorithm, dubbed 'ALGEBRAIC', based on approximating the reprojection error (2) by replacing the Euclidean distance measure d_e by the algebraic distance measure d_a defined by

$$d_a^2(\mathbf{x}, \mathbf{y}) = \mathsf{S}[\mathbf{x}]_\times \mathbf{y} \quad \text{with} \quad \mathsf{S} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix},$$

and

$$d_a^2(\mathbf{x}, \mathbf{y}) = \mathsf{S}[\mathbf{x}]_\times \mathbf{y} \quad \text{with} \quad \mathsf{S}_{(2\times 3)} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix},$$

where $[\mathbf{x}]_\times$ is the (3×3) skew-symmetric matrix associated to cross-product, *i.e.* $[\mathbf{x}]_\times \mathbf{y} = \mathbf{x} \times \mathbf{y}$. This gives an algebraic error function

$$\mathcal{E}_n^2(\lambda) = \sum_{i=1}^{n} d_a^2(\lambda \mathbf{b}_i + \mathbf{d}_i, \mathbf{q}_i) = \sum_{i=1}^{n} \left\| \lambda \mathsf{S}[\mathbf{q}_i]_\times \mathbf{b}_i + \mathsf{S}[\mathbf{q}_i]_\times \mathbf{d}_i \right\|^2,$$

in matrix form

$$\mathcal{E}_n^2(\lambda) = \left\| \begin{pmatrix} \cdots \\ \mathsf{S}[\mathbf{q}_i]_\times \mathbf{b}_i \\ \cdots \end{pmatrix}_{(2n\times 1)} \lambda + \begin{pmatrix} \cdots \\ \mathsf{S}[\mathbf{q}_i]_\times \mathbf{d}_i \\ \cdots \end{pmatrix}_{(2n\times 1)} \right\|^2.$$

A closed-form solution is obtained, giving λ_a in the least squares sense

$$\lambda_a = -\frac{\sum_{i=1}^{n} \mathbf{b}_i^\mathsf{T} [\mathbf{q}_i]_\times \tilde{\mathbf{I}} [\mathbf{q}_i]_\times \mathbf{d}_i}{\sum_{i=1}^{n} \mathbf{b}_i^\mathsf{T} [\mathbf{q}_i]_\times \tilde{\mathbf{I}} [\mathbf{q}_i]_\times \mathbf{b}_i} \quad \text{with} \quad \tilde{\mathbf{I}} \sim \mathsf{S}^\mathsf{T}\mathsf{S} \sim \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

Algorithms based on algebraic distances are highly conditioned by the image coordinate frame, see *e.g.* [7]. We experimentally tried the classical normalization used in, *e.g.* the eight-point algorithm in [7], and did not notice any difference with the normalization proposed in Section 3.2 for the polynomial algorithm.

5. Gauss–Newton refinement

As is usual for triangulation and bundle adjustment [7], we use the Gauss–Newton algorithm for refining an estimate of $\hat{\lambda}$ by minimizing the nonlinear least squares reprojection error (2). The algorithm, that we do not derived in details, is dubbed 'GAUSS–NEWTON'. We use the best solution amongst POLY and ALGEBRAIC as the initial solution.

6. Experimental results

6.1. Simulated data

We simulated a 3D line observed by n cameras P_i. In order to simulate realistic data, we reconstructed the 3D line as follows. We projected the line onto the images, and regularly sampled points on it, that were offset orthogonally to the image line with a Gaussian centered noise with variance σ_l. The 3D line was then reconstructed from the noisy points using the Maximum Likelihood triangulation method in [3], which provided[2] $\mathbf{M}$ and $\mathbf{N}$. Note that

[2] The line triangulation method in [3] provides the Plücker coordinates of the 3D line. Points $\mathbf{M}$ and $\mathbf{N}$ are extracted as the two singular vectors associated to the two non-zero singular values of the rank-two Plücker matrix using SVD. Note that the position of $\mathbf{M}$ and $\mathbf{N}$ along the line does not change the result.

any line triangulation method, see *e.g.* [14], can be used. Finally, a point lying on the true 3D line was projected onto the images, and corrupted with a Gaussian centered noise with variance σ_p, which gave the $\mathbf{q}_i$. We varied some parameters of this setup, namely n and σ_p, and the spatial configuration of the cameras, in order to compare the algorithms under different conditions. We compared two cases for the cameras: a stable one, in which they were evenly spread around the 3D line, and an unstable one, in which they were very close to each other. The default parameters of the setup are $\sigma_l = 0.1$ pixels, $\sigma_p = 3$ pixels, $n = 10$ views and stable cameras.

We had two main goals in these experiments. First, we wanted to determine what in practice is the maximum number of views and noise that the proposed triangulation method can deal with, for stable and unstable camera configurations. Second, we wanted to determine to which extent the line constraint improves the accuracy of the reconstructed 3D point, compared to standard unconstrained triangulation. We measured two kinds of error: the reprojection error, quantifying the ability of the methods to fit the measurements, and a 3D error, quantifying the accuracy of the reconstruction.

We compared the three algorithms, described in the paper (POLY, Section 3; ALGEBRAIC, Section 4; GAUSS–NEWTON, Section 5) and 3DTRIANGULATION, which is a standard Maximum Likelihood triangulation, ignoring the line constraint, *e.g.* [7].

Fig. 1 shows the results for varying noise level on the image points ($\sigma_p = 1, \ldots, 10$ pixels), and Fig. 2 for varying number of views ($n = 2, \ldots, 200$). Note the logarithmic scaling on the abscissa. General comments can be made about these results:

- 3DTRIANGULATION always gives the lowest reprojection error.
- ALGEBRAIC always gives the highest reprojection error and 3D error.
- POLY and GAUSS–NEWTON always give the lowest 3D error.

Small differences in the reprojection error may lead to large discrepancies in the 3D error. For example, POLY and GAUSS–NEWTON are undistinguisable on Figs. 1 (left) and 2 (left), showing the reprojection error, while they can clearly be distinguished in Figs. 1 (right) and 2 (right), showing the 3D error. This is due to the fact that GAUSS–NEWTON converges when some standard precision is reached on the reprojection error. Increasing the precision may improve the results, but would make convergence slower.

For $n = 10$ views, Fig. 1 shows that the accuracy of the 3D reconstruction is clearly better for the optimal methods POLY and GAUSS–NEWTON using the line constraint, compared to 3DTRIANGULATION that does not use this constraint. The difference in 3D accuracy is getting larger as the noise level increases. For a $\sigma_p = 1$ pixel noise, which is what one can expect in practice, the difference in accuracy is 1 cm, corresponding to 1% of the simulated scene scale. This is an important difference.

However, for $\sigma_p = 3$ pixels, beyond 20 views, Fig. 2 (left) shows that the reprojection error for 3DTRIANGULATION and POLY/GAUSS–NEWTON are hardly distinguishable, while we expect from Fig. 2 (right) the difference in 3D error to be negligible beyond 200 views.

The results presented above concern the stable camera setup. For the unstable case, we obtained slightly lower reprojection errors, which is due to the fact that the 3D model is less constrained, making the observations easier to "explain". However, as was expected, the 3D errors are higher by a factor of around 2. The order of the different methods remains the same as in the stable case. We noticed that incorporating the line constraint improves the accuracy compared to 3DTRIANGULATION to a much higher extent than in the stable case.

6.2. Real data

We tested the four reconstruction algorithms on several real data sets. For three of them, we show results. We used a Canny detector to retrieve salient edgels in the images, and adjusted segments using robust least squares. Finally, we matched the segments by hand between the images, except for the 387 frame 'building' sequence where automatic tracking was used. The point on line correspondences were manually given, again besides for the 'building' sequence for which correlation based tracking was used. We reconstructed the 3D lines from the edgels by the Maximum Likelihood method in [3].

The 'office' sequence. This data set consists of two images of an indoor scene, shown overlaid with 5 input segments and 10 input points in Fig. 3. A standard nonlinear least squares algorithm was used to recover the fundamental matrix, from which we extracted a pair of uncalibrated projection matrices. Note that for two views, line triangulation is an exact process. The end-points of matching segments correspond to the same physical point. We thus use them as input to our algorithms. A calibration grid was used to get the radial distortion parameters, that was corrected.

This data set is interesting in particular for the reason that one of the segments almost lies on the epipolar lines associated to its end-points, which is one case where line triangulation is singular. Indeed, any 3D line lying on the epipolar plane reprojects to the same image lines. For this segment, we used the fact that the end-points are corresponding, and can thus be triangulated on their own using standard unconstrained triangulation, to disambiguate the 3D line.

ALGEBRAIC and POLY/GAUSS–NEWTON gave, respectively, a 3.45 pixels and a 1.42 pixels reprojection error,[3] while

[3] These are RMS (Root Mean of Squares) errors over all images and all points.

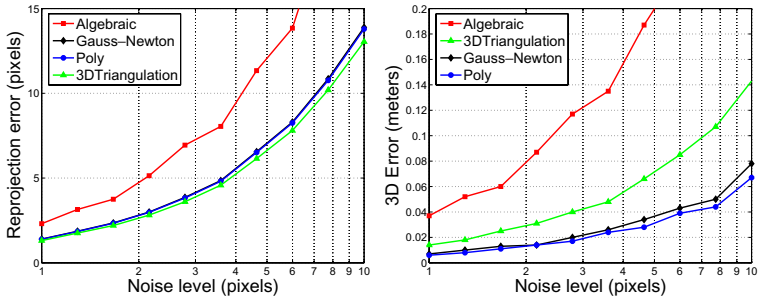


Fig. 1. Reprojection error (left) and 3D error (right) versus the level of noise.

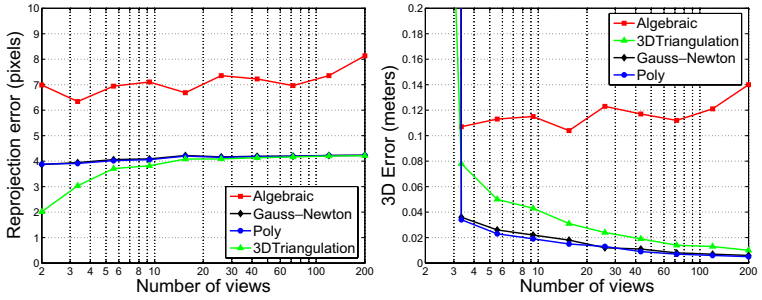


Fig. 2. Reprojection error (left) and 3D error (right) versus the number of views.

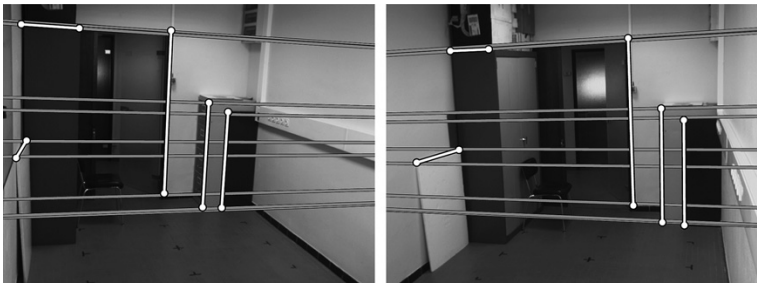


Fig. 3. The two original images of the 'office' sequence, overlaid with 5 matching segments, 10 corresponding end-points (in white), and their epipolar lines (in gray).

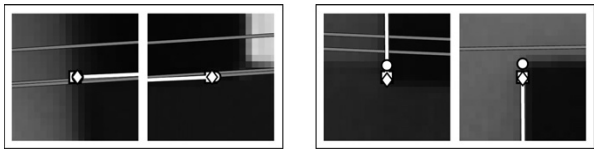


Fig. 4. Close up on some reprojected features, around the two end-points for the segment in the left hand corner of Fig. 3 (left) and in the right most part (right). The epipolar lines are shown in gray, and the segments in white, with their end-points plotted with squares. The diamonds are the points predicted from method ALGEBRAIC, and the circles from methods POLY and GAUSS–NEWTON (they are undistinguishable).

3DTRIANGULATION achieved 0.98 pixels. A close up on the reprojected points can be seen in Fig. 4.

The 'Valbonne church' sequence. We used 6 views from the popular 'Valbonne church' image set. Some of them are shown in Fig. 5, together with the 6 input segments and 13 inputs points. The cameras were obtained by Euclidean bundle adjustment over a set of points [11]. The RMS reprojection errors we obtained were

ALGEBRAIC	POLY	GAUSS-NEWTON	3DTRIANGULATION
1.37 pixels	0.77 pixels	0.77 pixels	0.56 pixels

Fig. 5. Three out of the 6 images taken from the 'Valbonne church' sequence, overlaid with 6 matching segments and 13 corresponding points.

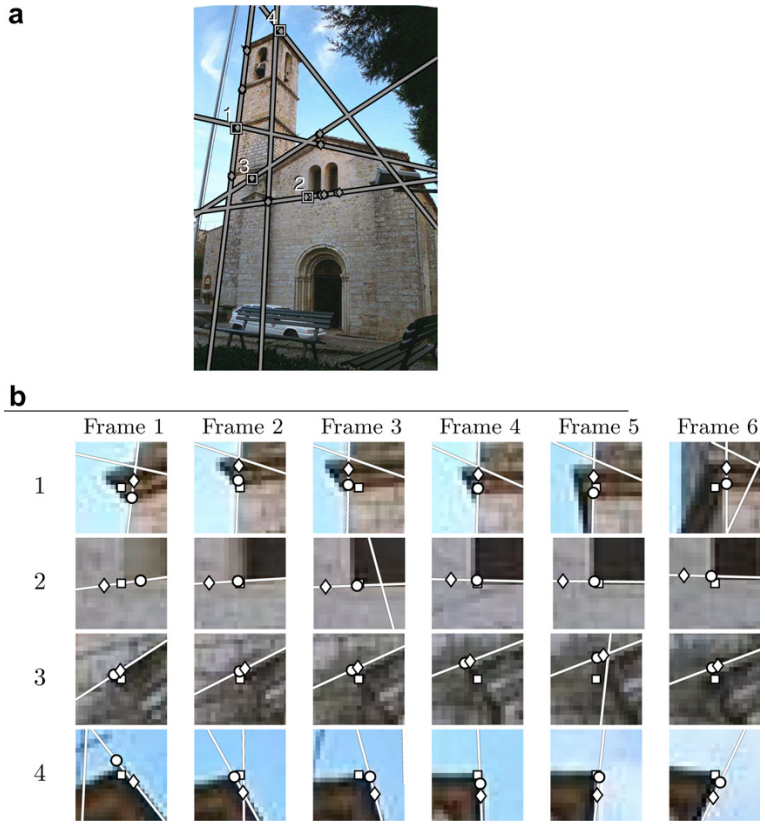


Fig. 6. Reprojected 3D lines and 3D points. (a) Shows four different numbered points, for which (b) shows a close up for all the 6 images. The squares are the original points, the diamonds are the points reconstructed by ALGEBRAIC, and the circles are the points reconstructed from POLY and GAUSS–NEWTON (they are undistinguishable).

ALGEBRAIC	POLY	GAUSS–NEWTON	3DTRIANGULATION
1.37 pixels	0.77 pixels	0.77 pixels	0.56 pixels

Fig. 6a shows lines and points reprojected from the 3D reconstruction. The reprojection errors we obtained for the points shown in Fig. 6b were

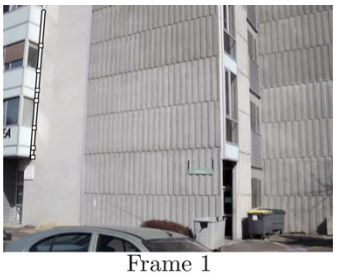


Fig. 7. Three out of the 387 images of the 'building' sequence, overlaid with the matching segments and 7 corresponding points.

Point	ALGEBRAIC (pixels)	POLY (pixels)	GAUSS–NEWTON (pixels)	3DTRIANGULATION (pixels)
1	4.03	2.14	2.14	1.83
2	6.97	1.95	1.95	1.52
3	2.84	2.21	2. 21	1.61
4	4.65	2.14	2.14	1.79

The reprojection error for 3DTRIANGULATION is slightly lower than for POLY/GAUSS–NEWTON. This indicates that the point on line constraint is feasible on these data.

The 'Building' sequence. This sequence is a continuous video stream consisting of 387 frames, showing a building imaged by a hand-held camera, see Fig. 7. We reconstructed calibrated cameras by bundle adjustment from interest points that were tracked using a correlation based tracker.

The segment we tracked is almost the only one that is visible throughout the sequence, and thus allows to test our triangulation methods for a very large number of views, namely 387. For the seven points we selected, we obtained a mean reprojection error of 4.57 pixels for ALGEBRAIC, of 3.45 pixels for POLY and GAUSS–NEWTON. Unconstrained triangulation gave a 2.90 pixels reprojection error. These errors which are higher than for the two previous data sets, are explained by the fact that there is non-negligible radial distortion in the images, as can be seen in Fig. 7.

7. Conclusions

We proposed an algorithm for the optimal triangulation, in the Maximum Likelihood sense, of a point lying on a given 3D line. Several transformations of 3D space and in the images lead to a degree-$(3n-2)$ polynomial equation. An efficient algorithm computes the real roots leading to feasible points only. Experimental evaluation on simulated and real data show that the method can be applied to large numbers of images, up to 387 in our experiments. The experiments were done for many different real data sets, indoor and outdoor, small, medium and large number of images, calibrated and uncalibrated reconstructions. Comparison of triangulated points with ground truth for the case of simulated data show that using the line constraint greatly improves the accuracy of the reconstruction.

Future work will be devoted to extending the method to the triangulation of points lying on parameterized curves.

Acknowledgements

The first author thanks Frederik Schaffalitzky and Andrew Zisserman for the projection matrices of the 'Valbonne church' sequence.

References

[1] F.S. Acton, Numerical Methods That Work, Mathematical Association of America, Washington, 1990 (Corrected edition).
[2] A. Bartoli, M. Coquerelle, P. Sturm. A framework for pencil-of-points structure-from-motion, in: Proceedings of the European Conference on Computer Vision, 2004.
[3] A. Bartoli, P. Sturm. Multiple-view structure and motion from line correspondences, in: Proceedings of the International Conference on Computer Vision, 2003.
[4] O. Chum, T. Pajdla, P. Sturm, The geometric error for homographies, Computer Vision and Image Understanding 97 (1) (2005) 86–102.
[5] R. Hartley, F. Schaffalitzky. L_∞ minimization in geometric reconstruction problems, in: Proceedings of the International Conference on Computer Vision and Pattern Recognition, 2004.
[6] R. Hartley, P. Sturm, Triangulation, Computer Vision and Image Understanding 68 (2) (1997) 146–157.
[7] R.I. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, second ed., Cambridge University Press, 2003.
[8] Q.T. Luong, T. Vieville, Canonic representations for the geometries of multiple projective views, Computer Vision and Image Understanding 64 (2) (1996) 193–229.
[9] D. Nistèr. Automatic Dense Reconstruction From Uncalibrated Video Sequences. PhD thesis, Royal Institute of Technology, KTH, March 2001.
[10] J. Oliensis, Exact two-image structure from motion, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (12) (2002) 1618–1633.
[11] F. Schaffalitzky, A. Zisserman. Multi-view matching for unordered image sets, in: Proceedings of the European Conference on Computer Vision, 2002.
[12] C. Schmid, A. Zisserman, The geometry and matching of lines and curves over multiple views, International Journal of Computer Vision 40 (3) (2000) 199–234.
[13] H. Stewénius, F. Schaffalitzky, D. Nistér. How hard is 3-view triangulation really? in: Proceedings of the International Conference on Computer Vision, 2005.
[14] A.W.K. Tang, T.P. Ng, Y.S. Hung, C.H. Leung, Projective reconstruction from line correspondences in multiple uncalibrated images, Pattern Recognition 39 (5) (2006) 889–896.

8.2.3 Paper (CVPR'07) – *Kinematics From Lines in a Single Rolling Shutter Image*

Kinematics from Lines in a Single Rolling Shutter Image

Omar Ait-Aider
LASMEA (CNRS / UBP)
Clermont-Ferrand, France
Omar.Ait-Aider@univ-bpclermont.fr

Adrien Bartoli
LASMEA (CNRS / UBP)
Clermont-Ferrand, France
Adrien.Bartoli@gmail.com

Nicolas Andreff
LASMEA (CNRS / UBP)
Clermont-Ferrand, France
Nicolas.Andreff@ifma.fr

Abstract

Recent work shows that recovering pose and velocity from a single view of a moving rigid object is possible with a rolling shutter camera, based on feature point correspondences.

We extend this method to line correspondences. Owing to the combined effect of rolling shutter and object motion, straight lines are distorted to curves as they get imaged with a rolling shutter camera. Lines thus capture more information than points, which is not the case with standard projection models for which both points and lines give two constraints.

We extend the standard line reprojection error, and propose a nonlinear method for retrieving a solution to the pose and velocity computation problem. A careful inspection of the design matrix in the normal equations reveals that it is highly sparse and patterned. We propose a blockwise solution procedure based on bundle-adjustment-like sparse inversion. This makes nonlinear optimization fast and numerically stable. The method is validated using real data.

1. Introduction

CMOS cameras offer several advantages: low cost, low power demand, easy region of interest selection, on-chip characteristics and high frame rate. This makes them a natural fit for wireless hand-held applications, visual servoing and some in-vehicle uses. In the last years, the performances of CMOS sensors in terms of signal-to-noise ratio have been considerably improved, reaching the level of CCD sensors. This has made CMOS cameras more and more used by the vision community in applications for which a high frame rate and accurate feature detection is necessary (fast robot control and identification, road traffic, balistic).

Standard and cheap CMOS cameras frequently use rolling shutter sensors. This shuttering mode enables adequate exposure time without reducing the frame rate by overlapping exposure and readout. It reduces the number of in-pixel transistors, improving the fill factor (percentage of the pixel array sensitive to light) and the signal-to-noise ratio. The drawback of rolling shutter cameras is that they distort images of moving objects because the pixels are not all exposed simultaneously but row by row with a time delay defined by the sensor technology (Fig. 1). This distortion may represent a major obstacle in tasks such as localization, 3D reconstruction or default detection (the system may see an ellipse where in fact there is a circular hole). Therefore, CMOS rolling shutter cameras could offer a good compromise between cost and frame rate performances if the problem of deformations is taken into account.

Figure 1. An example of distortion of a rotating ventilator observed with a rolling shutter camera: static object (left image) and moving object (right image).

The work done by Wilburn et al. [1] concerned the correction of image deformations due to rolling shutter by constructing a single image using several images from a dense camera array. Knowing the time delay due to rolling shutter and the chronograms of release of the cameras, one complete image is constructed by combining lines exposed at the same instant in each image from the different cameras. In [2] Meingast describes an approximated projection model of rolling shutter cameras which, in the case of fronto-parallel motion, is similar to a Crossed-Slits camera model [3]. Ait-Aider et al. [4] present a general and exact perspective projection model by removing the assumption of small motion during image acquisition. A nonlinear algorithm for simultaneous pose and velocity computation using a single view is then developed. It extends bundle

adjustment with point correspondences, to the case of moving objects observed with a rolling shutter camera. A linear algorithm is proposed in the particular case of planar objects. It provides an initial estimate of the pose and velocity parameters. To our knowledge, there is no other work in the vision community literature on taking into account effects of rolling shutter in pose recovery algorithms nor on exploiting them to compute the velocity parameters using a single view. Indeed, traditional pose recovery methods (for instance [5, 6, 7, 8, 9]) make the assumption that all image sensor pixels are exposed simultaneously.

In this paper, we propose an extension of the pose and velocity computation algorithm presented in [4] to line correspondences. When observed with a rolling shutter camera, a moving 3D line is projected to a curve on the image. A typical example is the observation of a polyhedral object. Straight edges are detected by segmenting image into contours. Why is this extension worthwhile? First, in the point-based algorithm of [4] the local distorsions in the image point neighborhood are neglected so that interest point detectors and descriptors such as Harris or SIFT remain usable. When motion artefacts are too important, this approximation may result in both false negatives in correlation-based matching and bad point localization. Using curves, even local distortions are modelled with an accuracy only bounded by image resolution. Second, for a rolling shutter projection model, lines capture more information about the motion than points. This is not the case with standard projection models for which both points and lines give two constraints. Finally, using all the pixels of a contour provides redundant information which is exploited against noise. Note that the pixels are here used directly without high level image processing. Conversely, detecting a straight line (with a classical camera) implies finding the function which best fits aligned pixels.

The main difficulty is that one can not derive an algebraic formulation of the curve corresponding to the projection of a straight line for a general motion. It is also difficult to find a metric which measures the distance between two such curves. We write the error between an observed and a reprojected curve as the sum of distances between the observed contour pixels and the reprojected 3D line. The point-to-curve distance does not however have a closed-form solution in general. The error is thus computed in practice by introducing, for each contour point, a corresponding point moving along the 3D line, so as to minimize the distance. This is done by introducing additional unknowns called nuisance variables in addition to the desired pose and velocity parameters. This results in a large but sparse Jacobian matrix. The latter property is taken into account in the resolution process. The optimization is achieved thanks to a blockwise solution procedure based on bundle-adjustment-like sparse inversion [10]. This ensures fast and numerically stable nonlinear optimization.

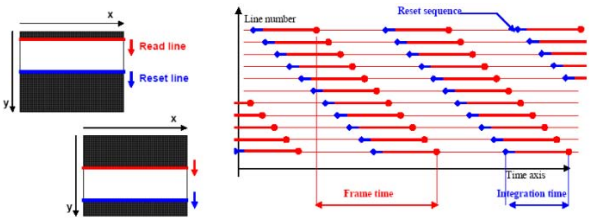


Figure 2. Reset and reading chronograms in rolling shutter sensor (Silicon Imaging documentation).

In section 2, we briefly recall the point projection model of a rolling shutter camera. In section 3, we formulate the pose and velocity computation problem under the form of a cost function derived from a set of line-to-curve correspondences. In section 4, we focus on improving efficiency and numerical stability of the nonlinear optimization of the cost function by exploiting the sparse structure of the Jacobian matrix. Finally, experimental results obtained with real images illustrate the performances of the method.

2. Rolling Shutter Camera Projection Model

In a CMOS camera operating in rolling shutter mode, the sensor pixels are exposed sequentially starting at the top and proceeding row by row to the bottom. The readout process proceeds in exactly the same fashion and the same speed with a time delay after the reset (exposure time). The benefit of rolling shutter mode is that exposure and readout are overlapping, enabling full frame exposures without reducing the frame rate. Each row in the image has the same amount of integration, however the starting and ending time of integration are shifted in time as the image is scanned (rolled) out of the sensor array, as shown in Fig. 2. If an observed object is moving during the integration time, some artefacts may appear and its image is distorted. The faster the object the larger the distortion. A simple case where the object undergoes a pure translational motion is illustrated on Fig.3.

Assume that an object of known geometry, modelled by a set of n points $\mathbf{P}_i = [X_i, Y_i, Z_i, 1]^{\mathrm{T}}$, undergoing a motion with instantaneous angular velocity Ω around an instantaneous axis of unit vector $\mathbf{a} = [a_x, a_y, a_z]^{\mathrm{T}}$, and instantaneous linear velocity $\mathbf{V} = [V_x, V_y, V_z, 1]^{\mathrm{T}}$, is snapped with a rolling shutter camera at time t_0. Denoting $\mathbf{R}$ and $\mathbf{T}$ the instantaneous object pose at t_0, it was demonstrated in [4] that the 2D projection $\mathbf{m}_i = [u_i, v_i, 1]^{\mathrm{T}}$ of $\mathbf{P}_i$ can be expressed up to an arbitrary scale factor s as follows:

$$s\mathbf{m}_i = \mathbf{K}\,[\mathbf{R}\delta\mathbf{R}_i \quad \mathbf{T} + \delta\mathbf{T}_i]\,\mathbf{P}_i \qquad (1)$$

with:

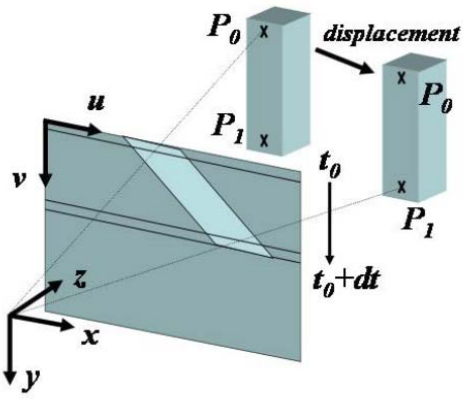


Figure 3. Perspective projection of a moving 3D object: due to the time delay, points $\mathbf{P}_0$ and $\mathbf{P}_1$ are not projected under the same pose

.

$$\delta\mathbf{R}_i = \mathbf{a}\mathbf{a}^T\left(1 - cos\left(\tau v_i \Omega\right)\right) + \mathbf{I}cos\left(\tau v_i \Omega\right) + \hat{\mathbf{a}}sin\left(\tau v_i \Omega\right) \tag{2}$$

and:

$$\delta\mathbf{T}_i = \tau v_i \mathbf{V} \tag{3}$$

where $\mathbf{I}$ is the 3×3 identity matrix, $\hat{\mathbf{a}}$ is the antisymmetric matrix associated to $\mathbf{a}$, τ is the image scanning speed (in raws per second) and $\mathbf{K}$ contains the classical intrinsic parameters of a pinhole camera. Note that $\mathbf{V}$ is the sum of two vectors $\mathbf{V}_L$ and $\mathbf{V}_R$. The first component is due to the pure translational motion and is thus expressed in the camera frame. The second component is induced by the rotation (tangential velocity) and must be expressed in the object frame. Thus we have $\mathbf{V} = \mathbf{R}\mathbf{V}_R + \mathbf{V}_L$.

Equation (1) is the expression of the projection of a 3D point from a moving solid object using a rolling shutter camera with respect to object pose, object velocity and the parameter τ. Note that it contains the unknown v_i on its two sides. This is due to the fact that coordinates of the projected point on the image depend on both the kinematics of the object and the imager sensor scanning velocity.

3. Pose and Velocity Computation with Lines

If a moving polyhedral object is observed with a rolling shutter camera, its straight edges are projected into the image as curved contours. Assume that a set of N straight edges, defined in the object frame by their direction vectors $\mathbf{L}_k$, are matched with a set of curved image contours $\mathbf{l}_k$. Considering an arbitrary point $\mathbf{M}_{k0}$ on $\mathbf{L}_k$, any other point $\mathbf{M}_{ki}$ on the latter edge can be expressed in the object frame as follows:

$$\mathbf{M}_{ki} = \mathbf{M}_{k0} + \sigma_{ki}\mathbf{L}_k \tag{4}$$

Thus, for each pixel on the curve one can write the following projection equation:

$$s\mathbf{m}_{ki} = \mathbf{K}\left[\mathbf{R}\delta\mathbf{R}_i \quad \mathbf{T} + \delta\mathbf{T}_i\right]\left(\mathbf{M}_{ki} + \sigma_{ki}\mathbf{L}_k\right) \tag{5}$$

This means that each pixel of the contour yields a pair of contraints of the form:

$$\begin{aligned} u_{ki} &= \alpha_u \frac{\mathbf{R}_{1i}(\mathbf{M}_{ki}+\sigma_{ki}\mathbf{L}_k)+T_{xi}}{\mathbf{R}_{3i}(\mathbf{M}_{ki}+\sigma_{ki}\mathbf{L}_k)+T_{zi}} + u_0 \\ v_{ki} &= \alpha_u \frac{\mathbf{R}_{2i}(\mathbf{M}_{ki}+\sigma_{ki}\mathbf{L}_k)+T_{yi}}{\mathbf{R}_{3i}(\mathbf{M}_{ki}+\sigma_{ki}\mathbf{L}_k)+T_{zi}} + v_0 \end{aligned} \tag{6}$$

It is obvious that matching a 3D straight edge with an image curve does not tell us for each contour pixel which is the corresponding 3D edge point. In other words, the values of σ_{ki} are unknown. Thus, equation (6) can be expressed as follows:

$$\begin{aligned} u_{ki} &\triangleq \xi_{uki}\left(\mathbf{R}, \mathbf{T}, \Omega, \mathbf{a}, \mathbf{V}, \mathbf{\Sigma}\right) \\ v_{ki} &\triangleq \xi_{vki}\left(\mathbf{R}, \mathbf{T}, \Omega, \mathbf{a}, \mathbf{V}, \mathbf{\Sigma}\right) \end{aligned} \tag{7}$$

where $\mathbf{\Sigma}$ is the vector of all the parameters σ_{ki}.

Considering the observation of m pixels $[\hat{u}_{ki}, \hat{v}_{ki}]$ on each one of the n image curves matched with a straight edge, and comparing them with the theoretical projections using (6) we obtain a $n \times m$ equation system representing the reprojection error:

$$\epsilon_{ki} = \begin{bmatrix} \hat{u}_{ki} - \xi_{uki}\left(\mathbf{R}, \mathbf{T}, \Omega, \mathbf{a}, \mathbf{V}, \mathbf{\Sigma}\right) \\ \hat{v}_{ki} - \xi_{vki}\left(\mathbf{R}, \mathbf{T}, \Omega, \mathbf{a}, \mathbf{V}, \mathbf{\Sigma}\right) \end{bmatrix} \tag{8}$$

From this, a cost function in the least square sense is expressed with respect to pose and velocity parameters $\mathbf{R}, \mathbf{T}, \Omega, \mathbf{a}, \mathbf{V}$ and also with respect to the so-called nuisance variables $\mathbf{\Sigma}$:

$$\begin{aligned} \epsilon = \sum_{k=1}^{n}\sum_{i=1}^{m} &\left[\hat{u}_{ki} - \xi_{uki}\left(\mathbf{R}, \mathbf{T}, \Omega, \mathbf{a}, \mathbf{V}, \mathbf{\Sigma}\right)\right]^2 \\ &+ \left[\hat{v}_{ki} - \xi_{vki}\left(\mathbf{R}, \mathbf{T}, \Omega, \mathbf{a}, \mathbf{V}, \mathbf{\Sigma}\right)\right]^2 \end{aligned} \tag{9}$$

This cost function is minimized using the Levenberg-Marquardt algorithm [11].

4. Blockwise Nonlinear Minimization

Let $\mathbf{y}$ be the vector of image projections in the left hand side of equation (7) and $\mathbf{x}$ the vector of pose, velocity and nuisance parameters $\mathbf{\Pi} = [\mathbf{R}, \mathbf{T}, \Omega, \mathbf{a}, \mathbf{V}, \mathbf{\Sigma}]$. The relationship between these two vectors is denoted $\mathbf{y} = \xi\left(\mathbf{x}\right)$. Given a set of noisy observations $\hat{\mathbf{y}}$, we want to converge toward

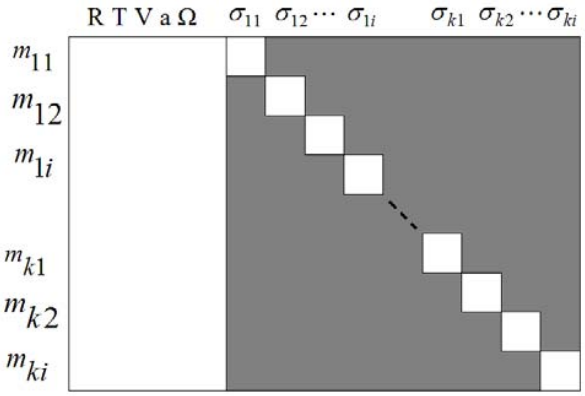


Figure 4. Structure of the Jacobian matrix $\mathbf{J}$.

the value $\hat{\mathbf{x}}$ so that the error $\hat{\mathbf{e}}$ in the relation $\hat{\mathbf{y}} = \xi(\hat{\mathbf{x}}) + \hat{\mathbf{e}}$ is minimal. The minimization starts from an initial guess $\mathbf{x_0}$ and is updated iteratively by applying variations δ to $\hat{\mathbf{x}}$. This is generally achieved by assuming local linearity of ξ under which one can write $\xi(\mathbf{x}_0 + \delta) = \xi(\mathbf{x}_0) + \mathbf{J}\delta$ with $\mathbf{J}$ the Jacobian matrix of $\xi(\mathbf{x})$. This implies to solve at each iteration the so called normal equations:

$$\mathbf{J}^{\mathrm{T}}\mathbf{J}\delta = \mathbf{J}^{\mathrm{T}}\mathbf{e} \tag{10}$$

In our case, the Jacobian matrix is very sparse because each image pixel $\mathbf{m}_{ki}$ on a matched contour depends on all the pose and velocity parameters $\mathbf{P} = [\mathbf{R}, \mathbf{T}, \Omega, \mathbf{a}, \mathbf{V}]$ but only on its own nuisance variable σ_{ki}. Thus $\frac{\partial \mathbf{m}_{ki}}{\partial \sigma_{lj}} \neq 0$ only for $k = l$ and $i = j$ but is null elsewhere. This results for $\mathbf{J}$ in the structure illustrated in Fig.4, which in turn produces the $\mathbf{J}^{\mathrm{T}}\mathbf{J}$ pattern illustrated in Fig.5 with:

$$\mathbf{U}_{p,q} = \sum_k \sum_i \left(\frac{\partial \mathbf{m}_{ki}}{\partial \mathbf{P}_p}\right)\left(\frac{\partial \mathbf{m}_{ki}}{\partial \mathbf{P}_q}\right) \tag{11}$$

$$\mathbf{V}_{p,q} = \sum_k \sum_i \left(\frac{\partial \mathbf{m}_{ki}}{\partial \mathbf{\Sigma}_p}\right)\left(\frac{\partial \mathbf{m}_{ki}}{\partial \mathbf{\Sigma}_q}\right) \tag{12}$$

$$\mathbf{W}_{p,q} = \sum_k \sum_i \left(\frac{\partial \mathbf{m}_{ki}}{\partial \mathbf{P}_p}\right)\left(\frac{\partial \mathbf{m}_{ki}}{\partial \mathbf{\Sigma}_q}\right) \tag{13}$$

A similar strucutre is exploited in bundle adjustment, for instance in [10], to reduce the computational cost for solving the normal equations by rewriting it as follows:

$$\begin{bmatrix} \mathbf{U} & \mathbf{W} \\ \mathbf{W}^{\mathrm{T}} & \mathbf{V} \end{bmatrix} \begin{bmatrix} \delta_{\mathbf{P}} \\ \delta_{\mathbf{\Sigma}} \end{bmatrix} = \begin{bmatrix} \mathbf{E}_{\mathbf{P}} \\ \mathbf{E}_{\Sigma} \end{bmatrix} \tag{14}$$

where $\delta_{\mathbf{P}}$ and $\delta_{\mathbf{\Sigma}}$ contains the small variations of respectively $\mathbf{P}$ and $\mathbf{\Sigma}$. The blocks $\mathbf{E}_{\mathbf{P}}$ and $\mathbf{E}_{\mathbf{\Sigma}}$ form the vector on the right hand side of the normal equation (10). Their components are defined as follows:

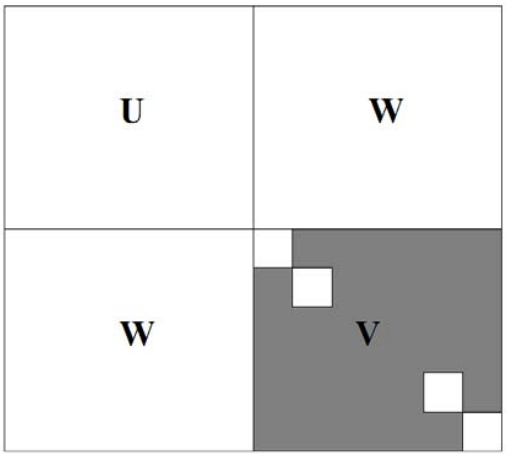


Figure 5. Structure of $\mathbf{J}^T\mathbf{J}$ in the normal equations.

$$\mathbf{E}_{\mathbf{P}q} = \sum_k \sum_i \left(\frac{\partial \mathbf{m}_{ki}}{\partial \mathbf{\Pi}_q}\right) \epsilon_{ki} \tag{15}$$

$$\mathbf{E}_{\mathbf{\Sigma}q} = \sum_k \sum_i \left(\frac{\partial \mathbf{m}_{ki}}{\partial \mathbf{\Pi}_q}\right) \epsilon_{ki} \tag{16}$$

Equation (14) can be rewritten as follows:

$$\begin{bmatrix} \mathbf{U} - \mathbf{W}\mathbf{V}^{-1}\mathbf{W}^{\mathrm{T}} & \mathbf{0} \\ \mathbf{W}^{\mathrm{T}} & \mathbf{V} \end{bmatrix} \begin{bmatrix} \delta_{\mathbf{P}} \\ \delta_{\mathbf{\Sigma}} \end{bmatrix} = \begin{bmatrix} \mathbf{E}_{\mathbf{P}} - \mathbf{W}\mathbf{V}^{-1}\mathbf{E}_{\Sigma} \\ \mathbf{E}_{\Sigma} \end{bmatrix} \tag{17}$$

which can be decomposed into two separate equation systems:

$$\left(\mathbf{U} - \mathbf{W}\mathbf{V}^{-1}\mathbf{W}^{\mathrm{T}}\right)\delta_{\mathbf{P}} = \mathbf{E}_{\mathbf{P}} - \mathbf{W}\mathbf{V}^{-1}\mathbf{E}_{\Sigma} \tag{18}$$

and

$$\delta_{\mathbf{\Sigma}} = \mathbf{V}^{-1}\left(\mathbf{E}_{\mathbf{\Sigma}} - \mathbf{W}^{\mathrm{T}}\delta_{\mathbf{P}}\right) \tag{19}$$

Equation (18) can be solved very efficiently because $\mathbf{V}$ is diagonal. Equation (19) is then solved by substituting the solution of (18).

5. Experimental Evaluation

The pose and velocity computation algorithm was tested on real image data. A reference 3D polyhedral object with both point and line features was used. A Silicon Imaging CMOS rolling shutter camera SI1280M-CL was first calibrated using the method described in [12] and then used to capture image sequences of the reference polyhedral object while undergoing rotational and translational motion at a high velocity. Fig.6 shows samples of images from these sequences.

Table 1. Differences between results of point-based algorithm and line-based algorithm (Mean value and standard deviation on the basis of 20 test images)

Parameter	R (deg)	T (m)	V (perc.)	Omega (perc.)
mean value	1.4	0.015	1.55	2.60
stand. dev.	1.0	0.006	1.05	1.80

Acquisition was done with a resolution of 640×480 square pixels and at a rate of 30 frames per second so that $\tau = 39.5 \times 10^{-6}$ s.

Point features served to generate groundtruth values for pose and velocity parameters. Indeed, since the point based algorithm was validated and evaluated in [4] using ground truth values, it was used here as a reference, simultaneously with the line-based algorithm and on the same image data. Image point coordinates were accurately obtained to sub-pixel accuracy estimation of the white spot centers and corrected according to the lens distortion parameters.

Thin image curves were detected thanks to Canny's criterion and chained to obtain contour curves. No additional processing was done on the contour pixels. The pixel coordinates were used directly in the algorithm.

For the nonlinear optimization, all nuisance and velocity parameters were initialized to zero. The position was initialized at $[0, 0, 1]^T$ (the object is in front of the camera) with a random orientation.

Both point and line correspondences with the model points and lines were established with a supervised method. The pose and velocity parameters were computed for each image using first our line-based algorithm, and compared with results obtained using the point-based algorithm and the classical pose recovery algorithm described in [12]. In the latter, an initial estimate of the solution is first computed using the algorithm of Dementhon [7] and then the pose parameters are refined thanks to a nonlinear method.

As shown in Fig.7, the trajectories and velocities computed by the line and the point-based algorithms are very close. The differences between the position, orientation, and velocity computed by the two algorithms are given in Table 1. Pose results obtained with a classical algorithm (which does not take into account the rolling shutter effects) show a shift proportional to the speed in the direction of the motion.

Fig.8 shows an example of correcting the object image by removing the velocity parameters in the projection equation. This corresponds to a global shutter image taken at t_0 (the instant of exposure of the first line of the sensor).

5.1. Conclusion and Perspectives

We presented a method for simultaneously computing the pose and instantaneous velocity (both translational and rotational) of rigid objects from a single rolling shutter image of straight lines. It benefits of an inherent defect of rolling shutter CMOS cameras consisting in exposing one after the other the rows of the image, yielding optical distortions due to high object velocity. The approach extends previous point-based methods to line correspondences. This offers, in the case of rolling shutter projection, real ad-

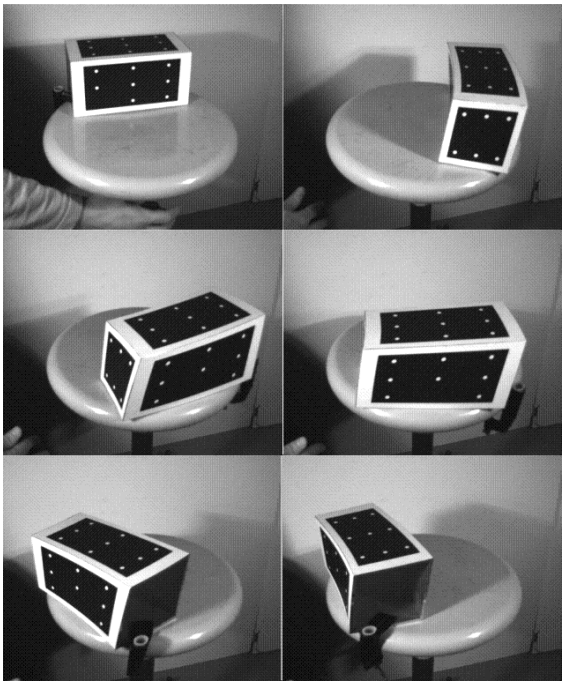


Figure 6. Samples of rolling shutter images of the moving reference object.

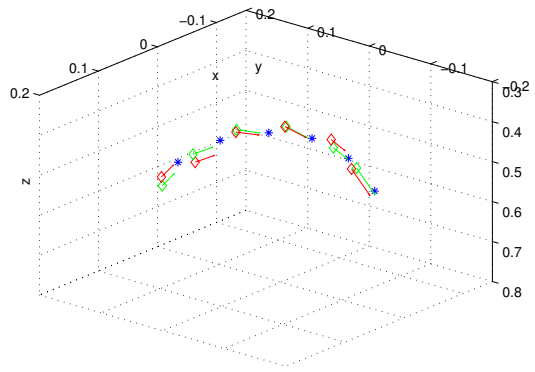


Figure 7. A comparison between two sets of trajectories and velocities computed by the line-based (red curve and arrows) and the point-based (green curve and arrows) algorithms respectively. The '*' symbols represent results obtained with a classical algorithm which does not take into account rolling shutter distorsions

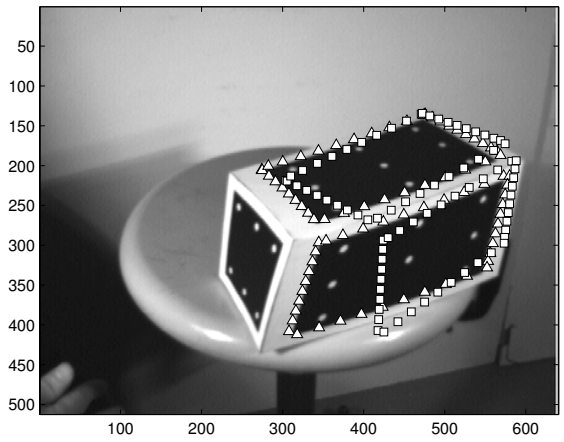


Figure 8. An example of reprojecting edge points using a rolling shutter model (triangles). Image correction: lines represented by square symbols are obtained by removing rolling shutter distorsions.

vantages (more information about motion, redundant information). An efficient optimization procedure was also proposed to improve numerical stability and computational cost of the approach.

The approach was validated on real data showing its relevance and feasibility. Hence, the proposed method is as accurate as similar classical algorithms in the case of static objects, but also preserves the accuracy of pose estimation when the object moves. In addition to pose estimation, the proposed method gives the instantaneous velocity using a single view. Thus, it avoids the use of finite differences between successive images (and the associated constant velocity assumption) to estimate a 3D object velocity.

Hence, carefully taking into account rolling shutter turns a low cost imager into a powerful pose and velocity sensor. Indeed, such a tool can be useful for many research areas. For instance, instantaneous velocity information may be used as evolution models in motion tracking to predict the state of observed moving patterns. It may also have applications in robotics, either in visual servoing or dynamic identification. In the latter domain our approach can make the difference when image processing leaves little time to other tasks (control, data fusion) by reducing drastically the amount of data necessary for motion analysis by using a single view instead of image sequences.

From a more theoretical point of view, several issues open. First, the proposed method uses a rolling shutter camera model based on instantaneous row exposure, but it could be easily extended to more general models where each pixel has a different exposure time. One could also imagine that an uncalibrated version of this method could be derived for applications where Euclidean information is not necessary (virtual/augmented reality or qualitative motion reconstruction, for instance). This certainly will make this work relevant to a broader range of scenes (where the identity of lines is not known a-priori).

References

[1] B. Wilburn, N. Joshi, V. Vaish, M. Levoy, and M. Horowitz, "High-speed videography using a dense camera array", in *IEEE Society Conference on Pattern Recognition (CVPR'04)*, 2004. 1

[2] M. Meingast, C. Geyer, and S. Sastry, "Geometric models of rolling-shutter cameras", in *Proc. of the 6th Workshop on Omnidirectional Vision, Camera Networks and Non-Classical Cameras*, Beijing, China, October 2005. 1

[3] A. Zomet, D. Feldman, S. Peleg, and D. Weinshall, "Mosaicing new views: The crossed-slits projection", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 6, pp. 741–754, 2003. 1

[4] O. Ait-Aider, N. Andreff, J. M. Lavest, and P. Martinet, "Simultaneous object pose and velocity computation using a single view from a rolling shutter camera", in *Proc. European Conference on Computer Vision, Vol. 2, pp. 56-68*, Graz, Austria, January 2006. 1, 2, 5

[5] D. G. Lowe, "Fitting parameterized three-dimensional models to image", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13(5), pp. 441–450, May 1991. 2

[6] T. Q. Phong, R. Horaud, and P. D. Tao, "Object pose from 2-d to 3-d point and line correspondences", *International Journal of Computer Vision*, pp. 225–243, 1995. 2

[7] D. Dementhon and L.S. Davis, "Model-based object pose in 25 lines of code", *International Journal of Computer Vision*, vol. 15, no. 1/2, pp. 123–141, June 1995. 2, 5

[8] M. Dhome, M. Richetin, J. T. Lapreste, and G. Rives, "Determination of the attitude of 3-d objects from a single perspective view", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 12, pp. 1265–1278, December 1989. 2

[9] R. Y. Tsai, "An efficient and accurate camera calibration technique for 3d machine vision", in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, Miami Beach, 1986, pp. 364–374. 2

[10] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment-a modern synthesis", in *Proc. of the International Workshop on Vision Algorithms: Theory and Practice, pp. 298-372*, London, UK, 1999. 2, 4

[11] D. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters", *SIAM Journal on Applied Mathematics*, vol. 11, no. 6, pp. 431–441, 1963. 3

[12] JM. Lavest, M. Viala, and M. Dhome, "Do we really need an accurate calibration pattern to achieve a reliable camera calibration", in *Proc. European Conference on Computer Vision ECCV, pp. 158-174*, Freiburg, Germany, June 1998. 4, 5

8.3 Structure-from-Motion with Curves Applied to Quality Control

I30 **Reconstruction of 3D Curves for Quality Control** §8.3.1
H. Martinsson, F. Gaspard, A. Bartoli and J.-M. Lavest
SCIA'07 - *Scandinavian Conf. on Image Analysis*, Aalborg, Denmark, June 2007
Version in French: [N09]

I37 **Energy-Based Reconstruction of 3D Curves for Quality Control** §8.3.2
H. Martinsson, F. Gaspard, A. Bartoli and J.-M. Lavest
EMMCVPR'07 - IAPR *Int'l Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, EZhou, Hubei, China, August 2007
Related paper: [I45]

8.3.1 Paper (SCIA'07) – *Reconstruction of 3D Curves for Quality Control*

Reconstruction of 3D Curves for Quality Control

H. Martinsson[1], F. Gaspard[1], A. Bartoli[2], and J.-M. Lavest[2]

[1] CEA, LIST, Boîte Courrier 94, F-91 191 Gif sur Yvette, France;
hanna.martinsson@cea.fr
[2] LASMEA (CNRS/UBP), 24 avenue des Landais, F-63 177 Aubière, France;
adrien.bartoli@gmail.fr

Abstract. In the area of quality control by vision, the reconstruction of 3D curves is a convenient tool to detect and quantify possible anomalies. Whereas other methods exist that allow us to describe surface elements, the contour approach will prove to be useful to reconstruct the object close to discontinuities, such as holes or edges.
We present an algorithm for the reconstruction of 3D parametric curves, based on a fixed complexity model, embedded in an iterative framework of control point insertion. The successive increase of degrees of freedom provides for a good precision while avoiding to over-parameterize the model. The curve is reconstructed by adapting the projections of a 3D NURBS snake to the observed curves in a multi-view setting.

1 Introduction

The use of optical sensors in metrology applications is a complicated task when dealing with complex or irregular structures. More precisely, projection of structured light allows for an accurate reconstruction of surface points but does not allow for a precise localization of the discontinuities of the object. This paper deals with the problem of reconstruction of 3D curves, given the CAD model, for the purpose of a control of conformity with respect to this model. We dispose of a set of images with given perspective projection matrices. The reconstruction will be accomplished by means of the observed contours and their matching, both across the images and to the model.

Algorithms based on active contours [9] allows for a local adjustment of the model and a precise reconstruction of primitives. More precisely, the method allows for an evolution of the reprojected model curves toward the image edges, thus to minimize the distance in the images between the predicted curves and the observed edges.

The parameterization of the curves as well as the optimization algorithms we use must yield an estimate that meets the requirements of accuracy and robustness necessary to perform a control of conformity. We have chosen to use NURBS curves [11], a powerful mathematical tool that is also widely used in industrial applications.

In order to ensure stability, any method used ought to be robust to erroneous data, namely the primitives extracted from the images, since images of metallic objects incorporate numerous false edges due to reflections.

Although initially defined for ordered point clouds, active contours have been adapted to parametric curves. Cham and Cipolla propose a method based on affine epipolar geometry [3] that reconstructs a parametric curve in a canonical frame using stereo vision. The result is two coupled snakes, but without directly expressing the 3D points. In [15], Xiao and Li deal with the problem of reconstruction of 3D curves from two images. However, the NURBS curves are approximated by B-splines, which makes the problem linear, at the expense of loosing projective invariance. The reconstruction is based on a matching process using epipolar geometry followed by triangulation. The estimation of the curves is performed independently in the two images, that is, there is no interactivity between the 2D observations and the 3D curve in the optimization. Kahl and August introduce in [8] a coupling between matching and reconstruction, based on an a priori distribution of the curves and on an image formation model. The curves are expressed as B-splines and the optimization is done using gradient descent.

Other problems related to the estimation of parametric structures have come up in the area of surfaces. In [14], Siddiqui and Sclaroff present a method to reconstruct rational B-spline surfaces. Point correspondences are supposed given. In a first step, B-spline surface patches are estimated in each view, then the surface in 3D, together with the projection matrices, are computed using factorization. Finally, the surface and the projection matrices are refined iteratively by minimizing the 2D residual error. So as to avoid problems due to over-parameterization, the number of control points is limited initially, to be increased later on in a hierarchical process by control point insertion.

In the case of 2D curve estimation, other aspects of the problem are addressed. Cham and Cipolla adjust a spline curve to fit an image contour [4]. Control points are inserted iteratively using a new method called PERM (potential for energy-reduction maximization). An MDL (minimal description length [7]) strategy is used to define a stopping criterion. In order to update the curve, the actual curve is sampled and a line-search is performed in the image to localize the target shape. The optimization is performed by gradient descent. Brigger et al. present in [2] a B-spline snake method without internal energy, due to the intrinsic regularity of B-spline curves. The optimization is done on the knot points rather than on the control points, which allows the formulation of a system of equations that can be solved by digital filtering. So as to increase numerical stability, the method is embedded in a multi-resolution framework. Meegama and Rajapakse introduce in [10] an adaptive procedure for control point insertion and deletion, based on the euclidean distance between consecutive control points and on the curvature of the NURBS curve. Local control is ensured by adjustment of the weights. The control points evolve in each iteration in a small neighborhood (3×3 pixels). Yang et al. use a distance field computed a priori with the fast marching method in order to adjust a B-spline snake [16]. Control points are added in the segment presenting a large estimation error, due to a degree of freedom insufficient for a good fit of the curve. The procedure is re-

peated until the error is lower than a fixed threshold. Redundant control points are then removed, as long as the error remains lower than the threshold.

2 Problem Formulation

Given a set of images of an object, together with its CAD model, our goal is to reconstruct in 3D the curves observed in the images. In order to obtain a 3D curve that meets our requirements regarding regularity, rather than reconstructing a point cloud, we estimate a NURBS curve. The reconstruction is performed by minimizing the quadratic approximation error. The minimization problem is formulated for a set of M images and N sample points by

$$\mathbf{C}(\mathcal{P}) = \arg\min_{\mathcal{P}} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} (\mathbf{q}_{ij} - T_i(\mathbf{C}(\mathcal{P}, t_j)))^2, \tag{1}$$

where $\mathbf{q}_{ij}$ is a contour point associated with the curve point of parameter t_j, T_i is the projective operator for image i and $\mathcal{P}$ is the set of control points.

Our choice to use NURBS curves is justified by several reasons. First, NURBS curves have interesting geometrical properties, namely concerning regularity and continuity. An important geometrical property that will be of particular interest is the invariance under projective transformations.

3 Properties of NURBS curves

Let $U = \{u_0, \cdots, u_m\}$ be an increasing vector, called the knot vector. A NURBS curve is a vector valued, piecewise rational polynomial over U, defined by

$$\mathbf{C}(t) = \sum_{i=0}^{n} \mathbf{P}_i R_{i,k}(t) \quad \text{with} \quad R_{i,k}(t) = \frac{w_i B_{i,k}(t)}{\sum_{j=0}^{n} w_j B_{j,k}(t)}, \tag{2}$$

where $\mathbf{P}_i$ are the control points, $B_{i,k}(t)$ the B-spline basis functions defined over U, w_i the associated weights and k the degree.

It is a common choice to take $k = 3$, which has proved to be a good compromise between required smoothness and the problem of oscillation, inherent to high degree polynomials. For our purposes, the parameterization of closed curves, we consider periodic knot vectors, that is, verifying $u_{j+m} = u_j$. Given all these parameters, the set of NURBS defined on U forms, together with the operations of point-wise addition and multiplication with a scalar, a vector space.

For details on NURBS curves and their properties, refer to [11].

3.1 Projective Invariance

According to the pinhole camera model, the perspective projection $T(\cdot)$ that transforms a world point into an image point is expressed in homogeneous coordinates by means of the transformation matrix $\mathbf{T}_{3\times4}$. Using weights associated

with the control points, NURBS curves have the important property of being invariant under projective transformations. Indeed, the projection of (2) remains a NURBS, defined by its projected control points and their modified weights. The curve is written

$$\mathbf{c}(t) = T(\mathbf{C})(t) = \frac{\sum_{i=0}^{n} w_i' \, T(\mathbf{P}_i) \, B_{i,k}(t)}{\sum_{i=0}^{n} w_i' \, B_{i,k}(t)} = \sum_{i=0}^{n} T(\mathbf{P}_i) R'_{i,k}(t) \tag{3}$$

The new weights are given by

$$w_i' = (T_{3,1}X_i + T_{3,2}Y_i + T_{3,3}Z_i + T_{3,4}) \, w_i = \mathbf{n} \cdot (\mathbf{C}_\mathrm{O} - \mathbf{P}_i) \, w_i, \tag{4}$$

where $\mathbf{n}$ is a unit vector along the optical axis and $\mathbf{C}_\mathrm{O}$ the optical center of the camera.

3.2 Control Point Insertion

One of the fundamental geometric algorithms available for NURBS is the control point insertion. The key is the knot insertion, which is equivalent to adding one dimension to the vector space, consequently adapting the basis. Since the original vector space is included in the new one, there is a set of control points such that the curve remains unchanged.

Let $\bar{u} \in [u_j, u_{j+1})$. We insert $\bar{u}$ in U, forming the new knot vector $\bar{U} = \{\bar{u}_0 = u_0, \cdots, \bar{u}_j = u_j, \bar{u}_{j+1} = \bar{u}, \bar{u}_{j+2} = u_{j+1}, \cdots, \bar{u}_{m+1} = u_m\}$. The new control points $\bar{\mathbf{P}}_i$ are given by the linear system

$$\sum_{i=0}^{n} \mathbf{P}_i R_{i,k}(t) = \sum_{i=0}^{n+1} \bar{\mathbf{P}}_i \bar{R}_{i,k}(t). \tag{5}$$

We present the solution without proof. The new control points are written

$$\bar{\mathbf{P}}_i = \alpha_i \, \mathbf{P}_i + (1 - \alpha_i) \, \mathbf{P}_{i-1}, \tag{6}$$

with

$$\alpha_i = \begin{cases} 1 & i \leq j - k \\ \dfrac{\bar{u} - u_i}{u_{i+k} - u_i} & \text{if } \; j - k + 1 \leq i \leq j \\ 0 & i \geq j + 1 \end{cases} . \tag{7}$$

Note that only k new control points need to be computed, due to the local influence of splines.

4 Curve Estimation

Using the NURBS formulation, the minimization problem (1) is written

$$\min_{\{\hat{\mathbf{P}}_l\}} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \left(\mathbf{q}_{ij} - \sum_{l=0}^{n} T_i(\hat{\mathbf{P}}_l) R_{l,k}^{(i)}(t_j) \right)^2 , \tag{8}$$

where $R_{l,k}^{(i)}$ are the basis functions for the projected NURBS curve in image i.

The problem has two parts. First, the search for candidate edge points $\mathbf{q}_{ij}$ in the images, then the optimization of the 3D NURBS curve by optimization on the control points. The search for candidate points is carried out independently in the images using a method inspired by the one used by Drummond and Cipolla in [6]. For the optimization problem, we use the non-linear Levenberg-Marquardt minimization method. This optimization allows the control points to move in 3D, but does not change their number. In order to obtain an optimal reconstruction of the observed curve, we iteratively perform control point insertion.

4.1 Search for Image Contours

We sample the NURBS curve projected in the image, to use as starting points in the search for matching contour points. A line-search is performed in order to find the new position of the curve, ideally corresponding to an edge. Our approach is based solely on the contours. Due to the aperture problem, the component of motion of an edge, tangent to itself, is not detectable locally and we therefore restrict the search for the new edge position to the edge normal at each sample point. As we expect the motion to be small, we define a search range (typically in the order of 20 pixels) so as to limit computational cost. In order to find the new position of a sample point, for each point belonging to the normal within the range, we evaluate the gradient and compute a weight based on the intensity and the orientation of the gradient and the distance from the sample point. The weight function v_j for a sample point p_j and the candidate point p_ξ will be of the form

$$v_j(p_\xi) = \varphi_1(|\nabla I_\xi|) \cdot \varphi_2\left(\frac{\hat{n}_j \cdot \nabla I_\xi}{|\nabla I_\xi|}\right) \cdot \varphi_3(|p_j - p_\xi|),$$

where $\hat{n}_j$ is the normal of the projected curve at sample point j, ∇I_ξ is the gradient at the candidate point and the φ_k are functions to define. The weight function will be evaluated for each candidate p_ξ and the point p'_j with the highest weight, identified by its distance from the original point $d_j = |p_j - p'_j|$, will be retained as the candidate for the new position of the point.

The bounded search range and the weighting of the point based on their distance from the curve yield a robust behavior, close to that of an M-estimator.

4.2 Optimization on the Control Points

The first step of the optimization consists in projecting the curve in the images. Since the surface model is known, we can identify the visible parts of the curve in each image and retain only the parts corresponding to knot intervals that are completely visible. During the iterations, so as to keep the same cost function, the residual error must be evaluated in the same points in each iteration. Supposing small displacements, we can consider that visible pieces will remain visible throughout the optimization.

The optimization of (8) is done on the 3D control point coordinates, leaving the remaining parameters of the NURBS curve constant. The weights associated

with the control points are modified by the projection giving 2D weights varying with the depth of each control point, according to the formula (4), but they are not subject to the optimization.

In order to avoid over-parameterization for stability reasons, the first optimization is carried out on a limited number of control points. Their number is then increased by iterative insertion, so that the estimated 3D curve fits correctly also high curvature regions. As mentioned earlier, the insertion of a control point is done without influence on the curve and a second optimization is thus necessary to estimate the curve. We finally need a criterion to decide when to stop the control point insertion procedure.

Control Point Insertion Due to the use of NURBS, we have a method to insert control points. What remains is to decide where to place them. Several strategies have been used. Cham and Cipolla consider in [4] the dual problem of knot insertion. They define an error energy reduction potential and propose to place the knot point so as to maximize this potential. The control point is placed using the method described earlier. In our algorithm, since every insertion is followed by an optimization that locally adjusts the control points, we settle for choosing the interval where to place the point. Since the exact location within the interval is not critical, the point is placed at its midpoint. Dierckx suggests in [5] to place the new point at the interval that presents the highest error. This is consistent with an interpretation of the error as the result of a lack of degrees of freedom that inhibits a good description of the curve. If, however, the error derives from other sources, this solution is not always optimal. In our case, a significant error could also indicate the presence of parasite edges or that of a parallel structure close to the target curve. We have therefore chosen a heuristic approach, that consists in considering all the intervals of the NURBS curve, in order to retain the one that allows for the largest global error decrease.

Stopping Criterion One of the motives for introducing parametric curves was to avoid treating all curve points, as only the control points are modified during the optimization. If the number of control points is close to the number of samples, the benefit is limited. Too many control points could also cause numerical instabilities, due to an over-parameterization of the curve on the one hand and the size of the non-linear minimization problem on the other hand. It is thus necessary to define a criterion that decides when to stop the control point insertion.

A strategy that aims to avoid the over-parameterization is the use of statistical methods inspired by the information theory. Based in a Maximum Likelihood environment, these methods combine a term equivalent, in the case of a normal distributed errors, to the sum of squares of the residual errors with a term penalizing the model complexity. Given two estimated models, in our case differentiated by their number of control points, the one with the lowest criterion will be retained. The first criterion of this type, called AIC (Akaike Information

Criterion), was introduced by Akaike in [1] and is written

$$AIC = 2k + n \ln \frac{RSS}{n}, \tag{9}$$

where k is the number of control points, n is the number of observations and RSS is the sum of the squared residual errors. Another criterion, based on a bayesian formalism, is the BIC (Bayesian Information Criterion) presented by Schwarz [13]. It stresses the number of data points n, so as to ensure an asymptotic consistency and is written

$$BIC = 2k \ln n + n \ln \frac{RSS}{n}. \tag{10}$$

Another family of methods uses the MDL [12] formulation, which consists in associating a cost with the quantity of information necessary to describe the curve. Different criteria follow, depending on the formulation of the estimation problem. In the iterative control point insertion procedure of Cham et Cipolla [4], the stopping criterion is defined by means of MDL. The criterion depends, on the one hand on the number of control points and on the residual errors, on the other hand on the number of samples and on the covariance.

Yet another way of choosing an appropriate model complexity is the classical method of cross-validation. The models are evaluated based on their capacity to describe the data. A subset of the data is used to define a fixed complexity model, while the rest serve to validate it. The process is repeated and a model is retained if its performance is considered good enough.

We have chosen to use the BIC for this first version of our algorithm. A more thorough study of the influence of the stopping criterion in our setting will be performed at a later stage.

4.3 Algorithm

The algorithm we implemented has two parts. The optimization of a curve using a fixed complexity model is embedded in an iterative structure that aims to increase the number of control points. The non-linear optimization of the 3D curve is performed by the Levenberg-Marquardt algorithm, using a cost function based on a search for contour points in the images.

5 Experimental Evaluation

5.1 Virtual Images

In order to validate our algorithms for image data extraction and for curve reconstruction, we have performed a number of tests on virtual images. The virtual setting also allows us to simulate deformations of the target object.

We construct a simplified model of an object, based on a single target curve. We then apply our 3D reconstruction algorithm, starting at a modified “model

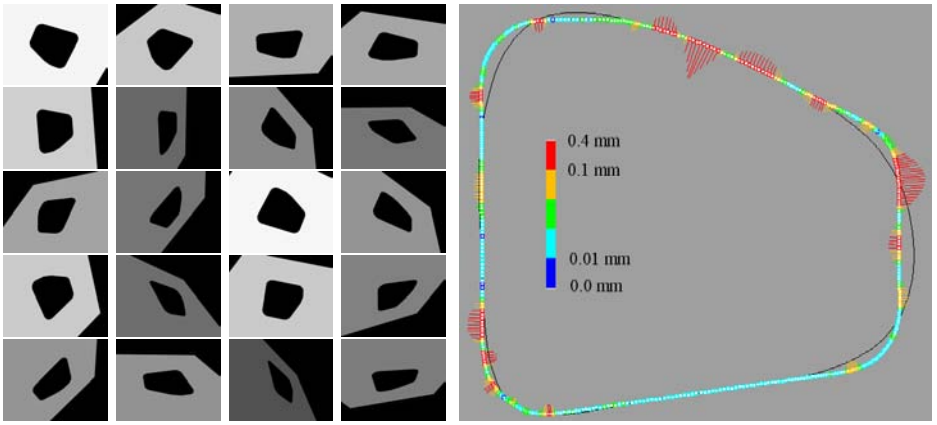


Fig. 1. *Left:* The virtual images used for the reconstruction of the central curve. *Right:* The distances from the sampled points from the reconstructed 3D curve to the model curve. The reconstruction is based on 20 virtual images. The cloud of sample points from the estimated curve is shown together with the target curve. The starting curve is shown in black. The differences are represented by lines with length proportional to the distance between the curve and the target, using a scale factor of 30.

curve", on a set of virtual views, see Fig. 1. The image size is 1284×1002 pixels. The starting curve has 10 control points, to which 11 new points are added. The sampling used for the computations is of 200 points. To fix the scale, note that at the mean distance from the object curve, one pixel corresponds roughly to 0.22 mm. The evaluation of the results is done by measuring the distance from a set of sampled points from the estimated curve to the target model curve. The distances from the target curve are shown in Fig. 1. We obtain the following results:

Mean error	0.0621 mm
Median error	0.0421 mm
Standard deviation	0.0528 mm

We note that the error corresponds to less than a pixel in the images, which indicates a sub-pixel image precision.

5.2 Real Images

We also consider a set of real images, see Fig. 2, with the same target curve, using the same starting "model curve" as in the virtual case. We now need to face the problem of noisy image data, multiple parallel structures and imprecision in the localization and the calibration of the views. The image size is 1392×1040 pixels. The starting curve has 10 control points, to which 11 new points are added. The sampling used for the computations is of 200 points. At the mean distance from the object curve, one pixel corresponds roughly to 0.28 mm. The distances from the target curve are shown in Fig. 2. We obtain the following results:

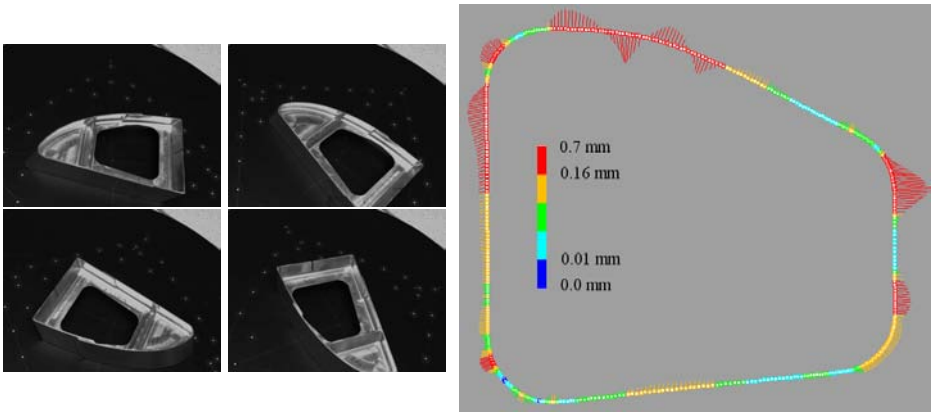


Fig. 2. *Left:* Four of the 18 real images used for the reconstruction of the curve describing the central hole. *Right:* The distances from the sampled points from the reconstructed 3D curve to the model curve. The reconstruction is based on 18 real images. The differences are represented by lines with length proportional to the distance between the curve and the target, using a scale factor of 20.

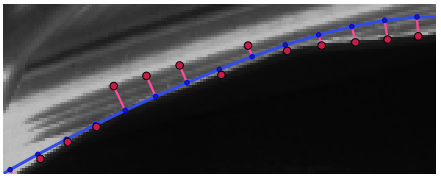


Fig. 3. Problems related to specularities and to the search for candidate points. Starting at the projection of the initial curve (in blue), some candidate points (in magenta) belong to a parasite edge.

Mean error	0.157 mm
Median error	0.124 mm
Standard deviation	0.123 mm

Even if the errors are higher than in the case of virtual images, we note that they still correspond to less than a pixel in the images. The difference is partly explained by the noise and the parallel structures perturbing the edge tracking algorithm. An example of candidate points located on a parallel image contour, due to specularities, is given in Fig. 3.

6 Conclusions

We have presented an adaptive 3D reconstruction method using parametric curves, limiting the degrees of freedom of the problem. An algorithm for 3D reconstruction of curves using a fixed complexity model is embedded in an iterative framework, allowing an enhanced approximation by control point insertion.

An experimental evaluation of the method, using virtual as well as real images, has let us validate its performance in some simple, nevertheless realistic, cases with specular objects subject to occlusions and noise.

Future work will be devoted to the integration of knowledge of the CAD model in the image based edge tracking. Considering the expected neighborhood of a sample point, the problem of parasite contours should be controlled and has limited impact on the obtained precision.

We also plan to do a deeper study around the stopping criterion used in the control point insertion process, using cross-validation.

References

1. H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automated Control*, 19(6):716–723, 1974.
2. P. Brigger, J. Hoeg, and M. Unser. B-spline snakes: A flexible tool for parametric contour detection. *IEEE Trans. on Image Processing*, 9(9):1484–1496, July 2000.
3. T.-J. Cham and R. Cipolla. Stereo coupled active contours. In *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition*, pages 1094–1099. IEEE Computer Society, 1997.
4. T.-J. Cham and R. Cipolla. Automated B-spline curve representation incorporating MDL and error-minimizing control point insertion strategies. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(1):49–53, January 1999.
5. P. Dierckx. *Curve and Surface Fitting with Splines*. Oxford University Press, Inc., New York, NY, USA, 1993.
6. T. Drummond and R. Cipolla. Real-time visual tracking of complex structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7:932–946, 2002.
7. M. H. Hansen and B. Yu. Model selection and the principle of minimum description length. *Journal of the American Statistical Association*, 96(454):746–774, 2001.
8. F. Kahl and J. August. Multiview reconstruction of space curves. In *9th International Conference on Computer Vision*, volume 2, pages 1017–1024, 2003.
9. M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 4(1):321–331, 1987.
10. R.G.N. Meegama and J. C. Rajapakse. NURBS snakes. *Image and Vision Computing*, 21:551–562, 2003.
11. L. Piegl and W. Tiller. *The NURBS book*. Monographs in visual communication. Springer Verlag, 2nd edition, 1997.
12. J. Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.
13. G. Schwarz. Estimating the dimension of a model. *Ann. of Stat.*, 6:461–464, 1978.
14. M. Siddiqui and S. Sclaroff. Surface reconstruction from multiple views using rational B-splines and knot insertion. In *First International Symposium on 3D Data Processing Visualization and Transmission*, pages 372–378, 2002.
15. Y.J. Xiao and Y.F. Li. Stereo vision based on perspective invariance of NURBS curves. In *IEEE International Conference on Mechatronics and Machine Vision in Practice*, volume 2, pages 51–56, 2001.
16. H. Yang, W. Wang, and J. Sun. Control point adjustment for B-spline curve approximation. *Computer-Aided Design*, 36:639–652, 2004.

8.3.2 Paper (EMMCVPR'07) – *Energy-Based Reconstruction of 3D Curves for Quality Control*

Energy-Based Reconstruction of 3D Curves for Quality Control

H. Martinsson[1], F. Gaspard[1], A. Bartoli[2], and J.-M. Lavest[2]

[1] CEA, LIST, Boîte Courrier 94, F-91 191 Gif sur Yvette, France;
tel: +33(0)1 69 08 82 98, fax: +33(0)1 69 08 83 95
hanna.martinsson@cea.fr
[2] LASMEA (CNRS/UBP), 24 avenue des Landais, F-63 177 Aubière, France;
tel: +33(0)4 73 40 76 61, fax: +33(0)4 73 40 72 62
adrien.bartoli@gmail.fr

Abstract. In the area of quality control by vision, the reconstruction of 3D curves is a convenient tool to detect and quantify possible anomalies. Whereas other methods exist that allow us to describe surface elements, the contour approach will prove to be useful to reconstruct the object close to discontinuities, such as holes or edges.
We present an algorithm for the reconstruction of 3D parametric curves, based on a fixed complexity model, embedded in an iterative framework of control point insertion. The successive increase of degrees of freedom provides for a good precision while avoiding to over-parameterize the model. The curve is reconstructed by adapting the projections of a 3D NURBS snake to the observed curves in a multi-view setting. The optimization of the curve is performed with respect to the control points using an gradient-based energy minimization method, whereas the insertion procedure relies on the computation of the distance from the curve to the image edges.

1 Introduction

The use of optical sensors in metrology applications is a complicated task when dealing with complex or irregular structures. More precisely, projection of structured light allows for an accurate reconstruction of surface points but does not allow for a precise localization of the discontinuities of the object. This paper deals with the problem of reconstruction of 3D curves, given the CAD model, for the purpose of a control of conformity with respect to this model. We dispose of a set of images with given perspective projection matrices. The reconstruction will be accomplished by means of the observed contours and their matching, both across the images and to the model. We proposed a previous version of our algorithm, based on edge distances, in [12]. The contributions of this paper with respect to the former one resides in the energy formulation, giving a new structure to the problem. We have also completed the experimental evaluation.

Algorithms based on active contours [11] allows for a local adjustment of the model and a precise reconstruction of primitives. More precisely, the method allows for an evolution of the reprojected model curves toward the image edges,

thus to minimize the distance in the images between the predicted curves and the observed edges.

The parameterization of the curves as well as the optimization algorithms we use must yield an estimate that meets the requirements of accuracy and robustness necessary to perform a control of conformity. We have chosen to use NURBS curves [14], a powerful mathematical tool that is also widely used in industrial applications.

In order to ensure stability, any method used ought to be robust to erroneous data, namely the primitives extracted from the images, since images of metallic objects incorporate numerous false edges due to reflections.

Although initially defined for ordered point clouds, active contours have been adapted to parametric curves. Cham and Cipolla propose a method based on affine epipolar geometry [4] that reconstructs a parametric curve in a canonical frame using stereo vision. The result is two coupled snakes, but without directly expressing the 3D points. In [19], Xiao and Li deal with the problem of reconstruction of 3D curves from two images. However, the NURBS curves are approximated by B-splines, which makes the problem linear, at the expense of loosing projective invariance. The reconstruction is based on a matching process using epipolar geometry followed by triangulation. The estimation of the curves is performed independently in the two images, that is, there is no interactivity between the 2D observations and the 3D curve in the optimization. Kahl and August introduce in [10] a coupling between matching and reconstruction, based on an a priori known distribution of the curves and on an image formation model. The curves are expressed as B-splines and the optimization is done using gradient descent.

Other problems related to the estimation of parametric structures have come up in the area of surfaces. In [18], Siddiqui and Sclaroff present a method to reconstruct rational B-spline surfaces. Point correspondences are supposed given. In a first step, B-spline surface patches are estimated in each view, then the surface in 3D, together with the projection matrices, are computed using factorization. Finally, the surface and the projection matrices are refined iteratively by minimizing the 2D residual error. So as to avoid problems due to over-parameterization, the number of control points is limited initially, to be increased later on in a hierarchical process by control point insertion.

In the field of medical imaging, energy minimization methods have been developed to reconstruct 3D curves in a stereo setting. Sbert and Solé reconstruct in [16] a 3D curve using an energy based evolution method. The associated PDE of the energy functional, derived by the Euler-Lagrange formulation, is solved using a level-set approach. In [3], Canero et al. define in a force field by reprojecting external image forces, given by the distance to the edges. A 3D curve is then reconstructed via the evolution of an active contour, guided by the force field.

In the case of 2D curve estimation, other aspects of the problem are addressed. Cham and Cipolla adjust a spline curve to fit an image contour [5]. Control points are inserted iteratively using a new method called PERM (poten-

tial for energy-reduction maximization). An MDL (minimal description length [9]) strategy is used to define a stopping criterion. In order to update the curve, the actual curve is sampled and a line-search is performed in the image to localize the target shape. The optimization is performed by gradient descent. Brigger et al. present in [2] a B-spline snake method without internal energy, due to the intrinsic regularity of B-spline curves. The optimization is done on the knot points rather than on the control points, which allows the formulation of a system of equations that can be solved by digital filtering. So as to increase numerical stability, the method is embedded in a multi-resolution framework. In [8], Figueiredo et al. address the problem from a statistical point of view, proposing a completely automatic contour estimator, in the sense that no parameter need to be adjusted by the user. Supposing a uniform distribution of the knot points, the B-spline curve that approximates a given set of contour points at best, in the least squares sense, is given by a linear system depending only on the number of control points. This number is fixed in advance using an MDL criterion. Meegama and Rajapakse introduce in [13] an adaptive procedure for control point insertion and deletion, based on the euclidean distance between consecutive control points and on the curvature of the NURBS curve. Local control is ensured by adjustment of the weights. The control points evolve in each iteration in a small neighborhood (3×3 pixels). Yang et al. use a distance field computed a priori with the fast marching method in order to adjust a B-spline snake [20]. Control points are added in the segment presenting a large estimation error, due to a degree of freedom insufficient for a good fit of the curve. The procedure is repeated until the error is lower than a fixed threshold. Redundant control points are then removed, as long as the error remains lower than the threshold.

2 Problem Formulation

Given a set of images of an object, together with its CAD model, our goal is to reconstruct in 3D the curves observed in the images. The reconstruction is performed by minimizing an energy functional. In order to obtain a 3D curve that meets our requirements regarding regularity, rather than reconstructing a point cloud, we estimate a NURBS curve. Since the regularity aspects are thereby taken care of, the energy functional is defined solely based on image data. The minimization problem is formulated for a set of M images and N sample points by

$$\mathbf{C}(\mathcal{P}) = \arg\min_{\mathcal{P}} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} E(T_i(\mathbf{C}(\mathcal{P}, t_j))), \tag{1}$$

where E is the external energy functional, T_i is the projective operator for image i and $\mathcal{P}$ is the set of control points.

Our choice to use NURBS curves is justified by several reasons. First, NURBS curves have interesting geometrical properties, namely concerning regularity and continuity. An important geometrical property that will be of particular interest is the invariance under projective transformations.

3 Properties of NURBS curves

Let $U = \{u_0, \cdots, u_m\}$ be an increasing vector, called the knot vector. A NURBS curve is a vector valued, piecewise rational polynomial over U, defined by

$$\mathbf{C}(t) = \sum_{i=0}^{n} \mathbf{P}_i R_{i,k}(t) \quad \text{with} \quad R_{i,k}(t) = \frac{w_i B_{i,k}(t)}{\sum_{j=0}^{n} w_j B_{j,k}(t)}, \tag{2}$$

where $\mathbf{P}_i$ are the control points, $B_{i,k}(t)$ the B-spline basis functions defined over U, w_i the associated weights and k the degree.

It is a common choice to take $k = 3$, which has proved to be a good compromise between required smoothness and the problem of oscillation, inherent to high degree polynomials. For our purposes, the parameterization of closed curves, we consider periodic knot vectors, that is, verifying $u_{j+m} = u_j$. Given all these parameters, the set of NURBS defined on U forms, together with the operations of point-wise addition and multiplication with a scalar, a vector space.

For details on NURBS curves and their properties, refer to [14].

3.1 Projective Invariance

According to the pinhole camera model, the perspective projection $T(\cdot)$ that transforms a world point into an image point is expressed in homogeneous coordinates by means of the transformation matrix $\mathbf{T}_{3\times4}$. Using weights associated with the control points, NURBS curves have the important property of being invariant under projective transformations. Indeed, the projection of (2) remains a NURBS, defined by its projected control points and their modified weights. The curve is written

$$\mathbf{c}(t) = T(\mathbf{C})(t) = \frac{\sum_{i=0}^{n} w_i' \, T(\mathbf{P}_i) \, B_{i,k}(t)}{\sum_{i=0}^{n} w_i' \, B_{i,k}(t)} = \sum_{i=0}^{n} T(\mathbf{P}_i) R_{i,k}'(t), \tag{3}$$

where the $R_{i,k}'$ are the basis functions of the projected NURBS. The new weights, w_i', are given by

$$w_i' = (T_{3,1}X_i + T_{3,2}Y_i + T_{3,3}Z_i + T_{3,4}) \, w_i = \mathbf{n} \cdot (\mathbf{C}_\mathrm{O} - \mathbf{P}_i) \, w_i, \tag{4}$$

where $\mathbf{n}$ is a unit vector along the optical axis and $\mathbf{C}_\mathrm{O}$ the optical center of the camera.

3.2 Control Point Insertion

One of the fundamental geometric algorithms available for NURBS curves is the control point insertion. The key is the knot insertion, which is equivalent to adding one dimension to the vector space, consequently adapting the basis. Since the original vector space is included in the new one, there is a set of control points such that the curve remains unchanged.

Let $\bar{u} \in [u_j, u_{j+1})$. We insert $\bar{u}$ in U, forming the new knot vector $\bar{U} = \{\bar{u}_0 = u_0, \cdots, \bar{u}_j = u_j, \bar{u}_{j+1} = \bar{u}, \bar{u}_{j+2} = u_{j+1}, \cdots, \bar{u}_{m+1} = u_m\}$. The new control points $\bar{\mathbf{P}}_i$ are given by the linear system

$$\sum_{i=0}^{n} \mathbf{P}_i R_{i,k}(t) = \sum_{i=0}^{n+1} \bar{\mathbf{P}}_i \bar{R}_{i,k}(t). \tag{5}$$

We present the solution without proof. The new control points are written [14]

$$\bar{\mathbf{P}}_i = \alpha_i \, \mathbf{P}_i + (1 - \alpha_i) \, \mathbf{P}_{i-1}, \tag{6}$$

with

$$\alpha_i = \begin{cases} 1 & i \leq j - k \\ \dfrac{\bar{u} - u_i}{u_{i+k} - u_i} & \text{if } \; j - k + 1 \leq i \leq j \\ 0 & i \geq j + 1 \end{cases} . \tag{7}$$

Note that only k new control points need to be computed, due to the local influence of splines.

4 Optimization

When treating NURBS curves, the regularity aspects are taken care of implicitly by the parameterization and the energy functional can be reduced to its external energy part. We will consider two forms of energy functionals, one based on the distance from the curve to the image contours and another one based on the gradient intensity. The optimization will in both cases operate on the control points of the 3D NURBS curve.

4.1 Distance Minimization

Using a distance formulation and the properties of NURBS curves, the minimization problem (1) is written

$$\min_{\{\hat{\mathbf{P}}_l\}} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \left(\mathbf{q}_{ij} - \sum_{l=0}^{n} T_i(\hat{\mathbf{P}}_l) R_{l,k}^{(i)}(t_j) \right)^2 , \tag{8}$$

where $\mathbf{q}_{ij}$ is a contour point associated with the curve point of parameter t_j in image i and T_i is the projective operator for image i. The search for candidate contour points is carried out independently in the images using a method inspired by the one used by Drummond and Cipolla in [7].

Search for Image Contours We sample the NURBS curve projected in the image, to use as starting points in the search for matching contour points. A line-search is performed in order to find the new position of the curve, ideally corresponding to an edge. Our approach is based solely on the contours. Due

to the aperture problem, the component of motion of an edge, tangent to itself, is not detectable locally and we therefore restrict the search for the new edge position to the edge normal at each sample point. As we expect the motion to be small, we define a search range (typically in the order of 20 pixels) so as to limit computational cost. In order to find the new position of a sample point, for each point belonging to the normal within the range, we evaluate the gradient and compute a weight based on the intensity and the orientation of the gradient and the distance from the sample point. The weight function v_j for a sample point p_j and the candidate point p_ξ will be of the form

$$v_j(p_\xi) = \varphi_1(|\nabla I_\xi|) \cdot \varphi_2\left(\frac{\hat{n}_j \cdot \nabla I_\xi}{|\nabla I_\xi|}\right) \cdot \varphi_3(|p_j - p_\xi|),$$

where $\hat{n}_j$ is the normal of the projected curve at sample point j, ∇I_ξ is the gradient at the candidate point and the φ_k are functions to define. The weight function will be evaluated for each candidate p_ξ and the point p'_j with the highest weight, identified by its distance from the original point $d_j = |p_j - p'_j|$, will be retained as the candidate for the new position of the point.

The bounded search range and the weighting of the point based on their distance from the curve yield a robust behavior, close to that of an M-estimator.

4.2 Gradient Energy Minimization

Using the classical energy formulation and the properties of NURBS curves, the minimization problem (1) is written

$$\min_{\{\hat{\mathbf{P}}_l\}} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} E\left(\sum_{l=0}^{n} T_i(\hat{\mathbf{P}}_l) R_{l,k}^{(i)}(t_j)\right), \tag{9}$$

where $R_{l,k}^{(i)}$ are the basis functions for the projected NURBS curve in image i. The energy functional E can, as already mentioned, be restricted to its external part, due to the use of NURBS. A common choice is to use the gradient intensity. We will however include local information on the curve, namely its normal direction, using the intensity of the gradient projected onto the curve normal.

4.3 Distance versus Gradient Energy

For comparison, we have implemented the two methods in the iterative setting that will be introduced in the following section. Both methods yielded similar results and converge after a number of iterations to an asymptotic lower limit. The 3D error with respect to the true curve is however somewhat lower for the gradient-based method. The results are given in Fig. 2. The difference is partly explained by the noise and the parallel structures perturbing the edge tracking algorithm. An example of candidate points located on a parallel image contour, due to specularities, is given in Fig. 1. Although the gradient intensity method outperforms the distance method, the distance-based cost function will prove to be useful in the iterative framework that will embed the curve optimization.

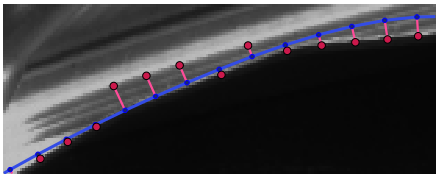


Fig. 1. Problems related to specularities and to the search for candidate points. Starting at the projection of the initial curve (in blue), some candidate points (in magenta) belong to a parasite edge.

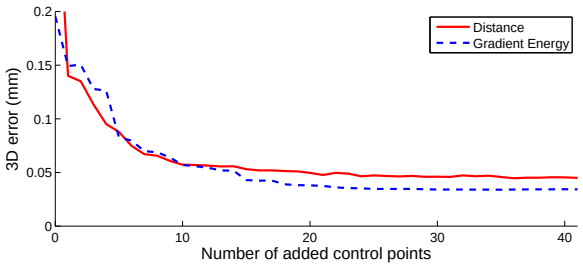


Fig. 2. The evolution of the error, with respect to the true 3D curve, for an optimization using cost functions based on the distance to the image contours and on the gradient intensity respectively. Whereas the distance method seems to yield good results in the start, its asymptotic limit is somewhat higher than that of the gradient intensity method.

5 Curve Estimation

The problem has two parts. First, the optimization of the 3D NURBS curve by energy minimization on a fixed number of control points, then the control point insertion procedure. For the fixed size optimization problem, we use the non-linear Levenberg-Marquardt minimization method. This step allows the control points to move in 3D, but does not change their number. In order to obtain an optimal reconstruction of the observed curve, we iteratively perform control point insertion. So as to avoid over-parameterization for stability reasons, the first optimization is carried out on a limited number of control points. Their number is then increased by iterative insertion, so that the estimated 3D curve fits correctly also in high curvature regions. As mentioned earlier, the insertion of a control point is done without influence on the curve and a second optimization is thus necessary in order to take advantage from the increased number of degrees of freedom.

5.1 Optimization on the Control Points

The first step of the optimization consists in projecting the curve in the images. Since the surface model is known, we can identify the visible parts of the curve

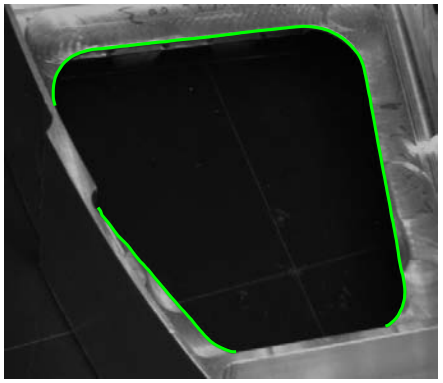


Fig. 3. Due to the use of 3D objects, auto-occlusions cause parts of the curve to be invisible from some viewpoints.

in each image and retain only the sample points corresponding to visible parts. During the iterations, to keep the same cost function, the residual error must be evaluated in the same points in each iteration. Supposing small displacements, we can consider that visible pieces will remain visible throughout the optimization. See Fig. 3 for an example of occlusions due to the 3D structure of the objects.

The optimization of (9) is done on the 3D control point coordinates, leaving the remaining parameters of the NURBS curve constant. The weights associated with the control points are modified by the projection giving 2D weights varying with the depth of each control point, according to the formula (4), but they are not subject to the optimization.

5.2 Control Point Insertion

Due to the use of NURBS, we have a method to insert control points. What remains is to decide where to place them. We also need a criterion to decide when to stop the control point insertion procedure.

Position of the New Control Point Several strategies have been used. Cham and Cipolla consider in [5] the dual problem of knot insertion. They define an error energy reduction potential and propose to place the knot point so as to maximize this potential. The control point is placed using the method described earlier. In our algorithm, since every insertion is followed by an optimization that adjusts the control points, we settle for choosing the interval where to place the point. Since the exact location within the interval is not critical, the point is placed at its midpoint. Dierckx suggests in [6] to place the new point at the interval that presents the highest error. This is consistent with an interpretation of the error as the result of a lack of degrees of freedom that inhibits a good description of the curve. If, however, the error derives from other sources, this solution is not always optimal.

In our case, a significant mean error could also indicate the presence of parasite edges or that of a parallel structure close to the target curve. We will therefore choose the interval with the highest median error, over all images. The error is defined as the distance from a sample point to its corresponding contour point in the image. The search for candidate contour points is carried out using the method described in 4.1.

Stopping Criterion One of the motives for introducing parametric curves was to avoid treating all curve points, as only the control points are modified during the optimization. If the number of control points is close to the number of samples, the benefit is limited. Too many control points could also cause numerical instabilities, due to an over-parameterization of the curve on the one hand and the size of the non-linear minimization problem on the other hand. It is thus necessary to define a criterion that decides when to stop the control point insertion.

A strategy that aims to avoid the over-parameterization is the use of statistical methods inspired by the information theory. Based in a Maximum Likelihood environment, these methods combine a term equivalent, in the case of a normal distributed errors, to the sum of squares of the residual errors with a term penalizing the model complexity. Given two estimated models, in our case differentiated by their number of control points, the one with the lowest criterion will be retained. The first criterion of this type, called AIC (Akaike Information Criterion), was introduced by Akaike in [1] and is written, in the case of normally distributed errors,

$$AIC = 2k + n \ln \frac{RSS}{n}, \tag{10}$$

where k is the number of control points, n is the number of observations and RSS is the sum of the squared residual errors. Another criterion, based on a bayesian formalism, is the BIC (Bayesian Information Criterion) presented by Schwarz [17]. It stresses the number of data points n, so as to ensure an asymptotic consistency and is written, also in the case of normally distributed errors,

$$BIC = 2k \ln n + n \ln \frac{RSS}{n}. \tag{11}$$

Another family of methods uses the MDL [15] formulation, which consists in associating a cost with the quantity of information necessary to describe the curve. Different criteria follow, depending on the formulation of the estimation problem. In the iterative control point insertion procedure of Cham et Cipolla [5], the stopping criterion is defined by means of MDL. The criterion depends, on the one hand on the number of control points and on the residual errors, on the other hand on the number of samples and on the covariance.

Yet another way of choosing an appropriate model complexity is the classical method of cross-validation. The models are evaluated based on their capacity to describe the data. A subset of the data is used to define a fixed complexity

model, while the rest serve to validate it. The process is repeated and a model is retained if its performance is considered good enough.

We have chosen to use the BIC, computed using the contour points found with the method presented in 4.1, for this first version of our algorithm. A more thorough study of the influence of the stopping criterion in our setting will be performed at a later stage.

5.3 Algorithm

The algorithm we implemented has two layers. The optimization of a curve using a fixed complexity model is embedded in an iterative structure that aims to increase the number of control points. The non-linear optimization of the 3D curve is performed by the Levenberg-Marquardt algorithm, using a cost function based on an energy formulation. The control point insertion procedure uses a search for contour points in the images in order to compute the median as well as the RSS error of the projected curve. The mechanism of our method is outlined in Table 1.

6 Experimental Evaluation

6.1 Virtual Images

In order to validate our algorithms for image data extraction and for curve reconstruction, we have performed a number of tests on virtual images. The virtual setting also allows us to simulate deformations of the target object.

We construct a simplified model of an object, based on a single target curve. We then apply our 3D reconstruction algorithm, starting at a modified "model curve", on a set of virtual views, see Fig. 4. The image size is 1284×1002 pixels. The starting curve has 10 control points, to which 45 new points are added. The sampling used for the computations is of 200 points. To fix the scale, note that at the mean distance from the object curve, one pixel corresponds roughly to 0.22 mm. The evaluation of the results is done by measuring the distance from a set of sampled points from the estimated curve to the target model curve. The distances from the target curve are shown in Fig. 4. We obtain the following results:

Mean error	0.0336 mm
Median error	0.0228 mm
Standard deviation	0.0485 mm

We note that the error corresponds to less than a pixel in the images, which indicates a sub-pixel image precision.

Using a series of virtual images of an object presenting a minor anomaly, we have also tested the capacity of our system to detect nonconformities, see Fig. 5. Based on 27 images and starting at the model curve, our algorithm manages to reconstruct the curve and its anomaly with a mean error of 0.0765 mm. Although the reconstruction is good, the error is concentrated around the anomaly, which is somewhat smoothed out.

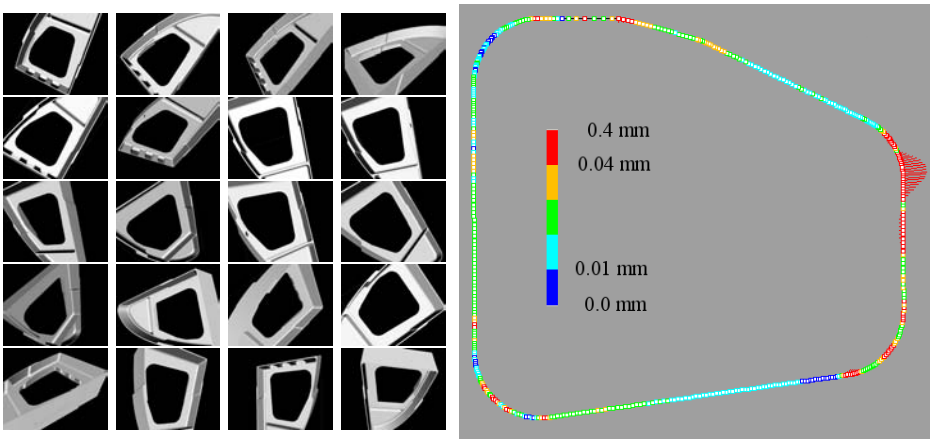


Fig. 4. *Left:* Some of the 32 virtual images used for the reconstruction of the central curve. *Right:* The distances from the sampled points from the reconstructed 3D curve to the model curve. The cloud of sample points from the estimated curve is shown together with the target curve. The starting curve is shown in black. The differences are represented by lines with length proportional to the distance between the curve and the target, using a scale factor of 20.

6.2 Real Images

We also consider a set of real images, see Fig. 6, with the same target curve, using the same starting "model curve" as in the virtual case. We now need to face the problem of noisy image data, multiple parallel structures and imprecision in the localization and the calibration of the views. The image size is 1392×1040 pixels. The starting curve has 10 control points, to which 48 new points are added. The sampling used for the computations is of 200 points. At the mean distance from the object curve, one pixel corresponds roughly to 0.28 mm. The distances from the target curve are shown in Fig. 6. We obtain the following results:

Mean error	0.137 mm
Median error	0.125 mm
Standard deviation	0.074 mm

Even if the errors are higher than in the case of virtual images, we note that they still correspond to less than a pixel in the images. The difference is explained by the noise and to some extent by specularities, causing parallel structures perturbing the minimization algorithm, see Fig. 7(b).

The evolution of the control points is demonstrated in Fig. 7(a), where the set of initial control points is shown, together with the final curve and its control points. As expected, the control points inserted are concentrated in the regions of high curvature, such as the corners.

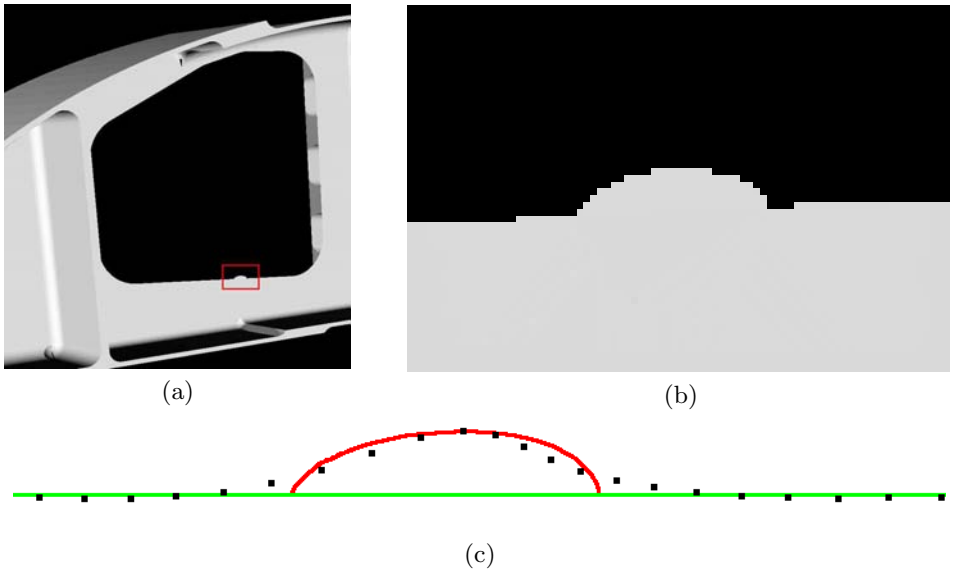


Fig. 5. Reconstruction of a nonconformity based on a series of virtual images of an object with an anomaly. The object is shown in (a) with the anomaly marked in red, with a close-up in (b). The result of the reconstruction around the anomaly is shown in (c), with the original curve in green, the anomaly in red and the reconstructed points in black.

7 Conclusions

We have presented an adaptive 3D reconstruction method using parametric curves, limiting the degrees of freedom of the problem. An algorithm for 3D reconstruction of curves using a fixed complexity model is embedded in an iterative framework, allowing an enhanced approximation by control point insertion. The optimization of the curve with respect to the control points is performed by means of a minimization of an gradient-based energy functional, whereas the insertion procedure is based on the distance from the curve to the observed image contours. An experimental evaluation of the method, using virtual as well as real images, has let us validate its performance in some simple, nevertheless realistic, cases with specular objects subject to occlusions and noise.

Future work will be devoted to the integration of knowledge of the CAD model in the image based edge tracking. Considering the expected neighborhood of a sample point, the problem of parasite contours should be controlled and has limited impact on the obtained precision.

We also plan to do a deeper study around the stopping criterion used in the control point insertion process, using cross-validation.

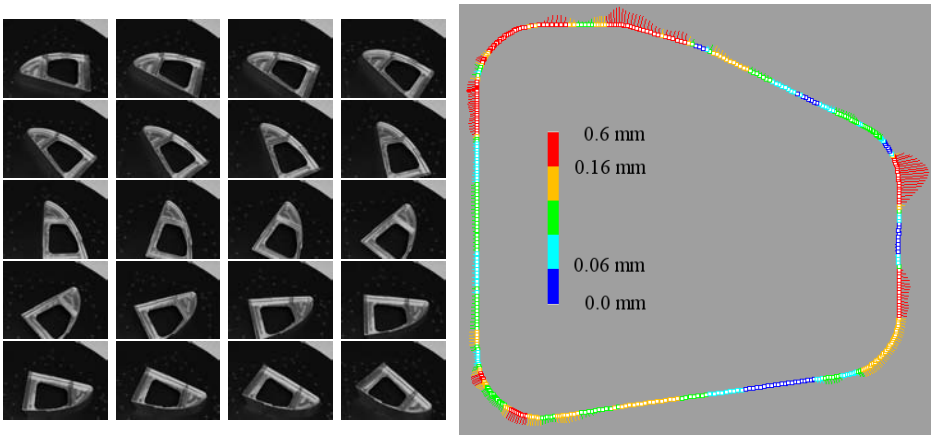
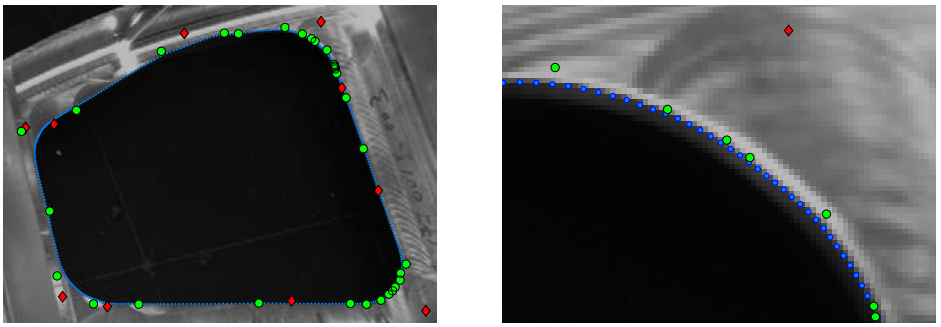


Fig. 6. *Left:* Some of the 36 real images used for the reconstruction of the curve describing the central hole. *Right:* The distances from the sampled points from the reconstructed 3D curve to the model curve. The differences are represented by lines with length proportional to the distance between the curve and the target, using a scale factor of 20.

References

1. H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automated Control*, 19(6):716–723, 1974.
2. P. Brigger, J. Hoeg, and M. Unser. B-spline snakes: A flexible tool for parametric contour detection. *IEEE Trans. on Image Processing*, 9(9):1484–1496, July 2000.
3. C. Canero, P. Radeva, R. Toledo, J.J. Villanueva, and J. Mauri. 3D curve reconstruction by biplane snakes. In *15th International Conference on Pattern Recognition (ICPR'00)*, volume 4, pages 563–566, 2000.
4. T.-J. Cham and R. Cipolla. Stereo coupled active contours. In *Conference on Computer Vision and Pattern Recognition*, pages 1094–1099. IEEE Computer Society, 1997.
5. T.-J. Cham and R. Cipolla. Automated B-spline curve representation incorporating MDL and error-minimizing control point insertion strategies. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(1):49–53, January 1999.
6. P. Dierckx. *Curve and Surface Fitting with Splines*. Oxford University Press, Inc., New York, NY, USA, 1993.
7. T. Drummond and R. Cipolla. Real-time visual tracking of complex structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7:932–946, 2002.
8. M. Figueiredo, J. Leitão, and A.K. Jain. Unsupervised contour representation and estimation using B-splines and a minimum description length criterion. *IEEE Transactions on Image Processing*, 9(6):1075–1087, June 2000.
9. M. H. Hansen and B. Yu. Model selection and the principle of minimum description length. *Journal of the American Statistical Association*, 96(454):746–774, 2001.
10. F. Kahl and J. August. Multiview reconstruction of space curves. In *9th International Conference on Computer Vision*, volume 2, pages 1017–1024, 2003.
11. M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 4(1):321–331, 1987.



(a) (b)

Fig. 7. (a) Evolution of the control points. In red the 10 initial control points, in green the 30 control points after optimization and in blue the final curve. (b) A detail, showing the curve between two similar contours, the parallel model curve to the lower left and a false edge caused by specularities to the upper right. Thanks to the multi-view setting, the reconstructed curve corresponds to the true 3D curve.

12. H. Martinsson, F. Gaspard, A. Bartoli, and J-M. Lavest. Reconstruction of 3d curves for quality control. In *15th Scandinavian Conference on Image Analysis*, 2007 (to appear).
13. R.G.N. Meegama and J. C. Rajapakse. NURBS snakes. *Image and Vision Computing*, 21:551–562, 2003.
14. L. Piegl and W. Tiller. *The NURBS book*. Monographs in visual communication. Springer Verlag, 2nd edition, 1997.
15. J. Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.
16. C. Sbert and A.F. Solé. Stereo reconstruction of 3d curves. In *15th International Conference on Pattern Recognition (ICPR'00)*, volume 1, 2000.
17. G. Schwarz. Estimating the dimension of a model. *Ann. of Stat.*, 6:461–464, 1978.
18. M. Siddiqui and S. Sclaroff. Surface reconstruction from multiple views using rational B-splines and knot insertion. In *First International Symposium on 3D Data Processing Visualization and Transmission*, pages 372–378, 2002.
19. Y.J. Xiao and Y.F. Li. Stereo vision based on perspective invariance of NURBS curves. In *IEEE International Conference on Mechatronics and Machine Vision in Practice*, volume 2, pages 51–56, 2001.
20. H. Yang, W. Wang, and J. Sun. Control point adjustment for B-spline curve approximation. *Computer-Aided Design*, 36:639–652, 2004.

OBJECTIVE

Given a 3D NURBS curve extracted from the CAD model, (partially) seen in M images, sampled in N points. We want to reconstruct the 3D curve observed in the images in order to compare it to the model.

ALGORITHM

- **Visibility Check** Identification of the visible parts

$$\chi_{ij} = \begin{cases} 1 \text{ if } \sum_{l=0}^{n-1} T_i(\mathbf{P}_l)\mathbf{R}_{l,k}^{(i)}(t_j) \text{ visible} \\ 0 \text{ sinon} \end{cases}$$

- **Optimization** on the control points

$$\min_{\{\hat{\mathbf{P}}_l\}} \sum_{i=0}^{M-1}\sum_{j=0}^{N-1} \chi_{ij}\, E\left(\sum_{l=0}^{n-1} T_i(\hat{\mathbf{P}}_l)\mathbf{R}_{l,k}^{(i)}(t_j)\right)$$

- **Line-search** for contour points $\mathbf{q}_{ij}$ matching $\mathbf{p}_{ij} = \sum_{l=0}^{n-1} T_i(\mathbf{P}_l)\mathbf{R}_{l,k}^{(i)}(t_j)$

$$\mathbf{q}_{ij} = \arg_{\mathbf{p}_{ij}^m} \max_{-d\le m\le d} v_j(\mathbf{p}_{ij}^m,) \text{ where } \mathbf{p}_{ij}^m = \mathbf{p}_{ij} + m\cdot\hat{\mathbf{n}}_{ij}$$

- **Computation of the** BIC

$$BIC_0 = k\ln(N\cdot M) + N\cdot M\cdot\ln\left(\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\chi_{ij}\left(\mathbf{q}_{ij} - \sum_{l=0}^{n-1} T_i(\mathbf{P'}_l)\mathbf{R}_{l,k}^{(i)}(t_j)\right)^2/(N\cdot M)\right)$$

- **do** (control point insertion)
  - **Line-search** for contour points $\mathbf{q}_{ij}$
  - **Computation of the median error** for each interval I_K.

$$m_K = \operatorname*{med}_{E_K} |\mathbf{q}_{ij} - \sum_{l=0}^{n-1} T_i(\mathbf{P'}_l)\mathbf{R}_{l,k}^{(i)}(t_j)|$$

    where $E_K = \{(i,j)\,|\,0\le i<M, t_j\in I_K, \chi_{ij}\neq 0\}$.
  - **Knot point insertion** at the midpoint of interval $I = \arg\min_K m_K$.
  - **Visibility Check** Identification of the visible parts
  - **Optimization** on the control points
  - **Computation of the** BIC_J
- **while** $(BIC_J < BIC_{J-1})$

Table 1. Reconstruction algorithm presented.

CHAPTER

9

OTHER WORKS

9.1 Active Appearance Models

I41	**Segmented AAMs Improve Person-Independent Face Fitting** J. Peyras, A. Bartoli, H. Mercier and P. Dalle BMVC'07 - *British Machine Vision Conf.*, Warwick, UK, September 2007	§9.1.1

I49	**Light-Invariant Fitting of Active Appearance Models** D. Pizarro, J. Peyras and A. Bartoli CVPR'08 - IEEE *Int'l Conf. on Computer Vision and Pattern Recognition*, Anchorage, Alaska, USA, June 2008	§9.1.2

9.1.1 Paper (BMVC'07) – *Segmented AAMs Improve Person-Independent Face Fitting*

Segmented AAMs Improve Person-Independent Face Fitting

Julien Peyras[1] Adrien Bartoli[2] Hugo Mercier[3] Patrice Dalle[3]

[1] Dipartimento di Scienze dell'Informazione, Milano, Italy
[2] LASMEA, Clermont-Ferrand, France
[3] IRIT, Toulouse, France

{Peyras,Mercier,Dalle}@irit.fr Adrien.Bartoli@gmail.com

Abstract

An Active Appearance Model (AAM) is a variable shape and appearance model built from annotated training images. It has been largely used to synthesize or fit face images. Person-independent face AAM fitting is a challenging open issue. For standard AAMs, fitting a face image for an individual which is not in the training set is often limited in accuracy, thereby restricting the range of application.

As a first contribution, we show that the limitation mainly comes from the inability of the AAM appearance counterpart to generalize, *i.e.* to accurately generate previously unseen visual data. As a second contribution, we propose an efficient person-independent face fitting framework based on what we call multi-level segmented AAMs. Each segment encodes a physically meaningful part of the face, such as an eye. A coarse-to-fine fitting strategy with a gradually increasing number of segments is used in order to ensure a large convergence basin.

Fitting accuracy is assessed by comparison with manual labelling statistics constructed from multiple data annotations. Experimental results support the claim that standard AAMs are well-adapted to person-specific fitting while segmented AAMs outperform the classical AAMs in a person-independent context in terms of accuracy, and ability to generate new faces.

1 Introduction

The Active Appearance Model (AAM) paradigm was introduced in 1998 by Cootes *et al.* [3] and since then it has had a great success. An AAM learns the shape and the appearance of a labelled set of images showing some class of objects. AAMs are widely used for face fitting, see *e.g.* [3, 8] and face synthesis, see *e.g.* [4]. Most of the applications – in the medical, psychological and linguistic fields, cognitive studies, expression transfer on an avatar, *etc.* – require highly accurate fitting. In other words, the AAM parameters must be recovered such that the synthesized image closely matches the input image.

Most of the previous work uses a single AAM modeling the face as a whole. Accurate fitting is achieved in a person-specific context. For instance, [8] uses AAMs for facial

deformation analysis. The standard AAM usually fails to achieve high accuracy for an image of a previously unseen face, *i.e.* for an individual not in the training set. Person-independent face fitting is however a very important problem since a training image set might not be available for an individual whose face needs to be accurately tracked in a video.

The closest work to ours is probably by Gross *et al.* [7]. They tackle the problem of constructing and fitting person-independent AAMs. They show that this is a difficult problem, even for frontal pose and neutral expression, and that the difficulties come from the inability of standard AAMs to generate new faces. A solution based on training, iteratively refitting the data with the AAM and re-training, is shown to improve the performances compared to traditional single step AAM training. Cristinacce *et al.* [5] recently proposed a paradigm called Constrained Local Model (CLM). It is shown to be effective at fitting a local face model based on measuring the image response around vertices and with a shape prior learnt from training images.

This paper tackles the important issue of person-independent face fitting with AAMs. We bring several statements and technical contributions:

- First, §3, we propose a means to assess fitting accuracy: the SSE (Statistical Shape Error). It is based on using several manual labellings of the input images by different users, from which gaussian statistics are computed for each label. The quality of an AAM fit is assessed by using the Mahalanobis distance with manual labelling statistics. This is an essential tool for the subsequent experimental analysis.

- Second, §4, we experimentally investigate the behavior of standard AAMs on unseen faces, and show that the lack of accuracy is mainly due to the inability of the appearance component to generate unseen faces. We state that standard AAMs are accurate in a person-specific context but not in a person-independent one.

- Third, §5, we show that segmented AAMs outperform standard ones in the person-independent context and achieve very accurate fitting, of the same order as the accuracy reached with manual labelling statistics. Segmented AAMs consist of several portions, each of which modeling a region of the face such as the mouth. Directly fitting each segment would reduce the convergence basin compared to fitting a standard AAM. As a remedy, we propose a coarse-to-fine fitting strategy which gradually splits a standard AAM into pre-defined segments. This *multi-level segmented AAM* we propose thus is able to generate new faces and can be effectively fit to images. Experimental results show that this outperforms the refitting solution of [7].

We give some background on AAMs below and our conclusions in §6.

2 Background on AAMs: Training and Fitting

An AAM combines two linear subspaces, one for the shape and one for the appearance, which are learnt from a previously labelled set of training images [3].

Principal Component Analysis (PCA) is applied on shape training data to retrieve a set of shape eigencomponents s_i expressing the shape model variation, and their associated eigenvalues proportional to the variance of the training data the s_i enclose. Four

extra components s_i^* are added to allow the 2D similarity transform, see [10]. Let $B_s = [s_1, \cdots, s_i, \cdots, s_1^*, \cdots, s_4^*]$ be the shape subspace basis. An instance of shape is defined as a linear expression: $s = B_s p_s$ with p_s the shape deformation parameters.

PCA is applied on the shape-corrected appearance data to retrieve a set of appearance eigencomponents A_i, allowing variations on the model appearance, and their associated eigenvalues proportional to the variance of the training data the A_i enclose. Two extra components A_0 for gain and A_I for bias are added, see [1]. Let $B_a = [A_1, \cdots, A_i, \cdots, A_0, A_I]$ be the appearance subspace basis. An instance of appearance is defined as a linear expression: $A = B_a p_a$ with p_a the appearance variation parameters.

Fitting an AAM consists to find the shape and appearance parameters that make it match the input image as best as possible. This is done by an iterative, nonlinear optimization process. We use the inverse compositional optimization scheme presented by Baker and Matthews in [10]. The Jacobian and Hessian matrices are derived analytically. Two versions of this algorithm were proposed and compared in [7]. Our implementation relies on the most accurate one called the *simultaneous inverse compositional algorithm*, originally described in [1].

We adapt this algorithm to our multi-level segmented AAM.

3 Assessing Fitting Accuracy

Fitting accuracy on unseen face images is generally assessed based on a single manual annotation of each image, considered as the absolute shape reference. The assumption behind this accuracy evaluation method is that the manual label is correct at the pixel level.

This assumption is often violated in practice: a vertex on a face image gets significantly different manual annotations, even from the same user. It is also incorrect to consider that one manual annotation is better than the others. It might also happen that a well performing automatic process is more accurate than manual labellers.

To address this improper accuracy assessment problem, Mercier *et al.* [11] suggest to annotate a face several times and build statistics for each vertex. It is then possible to set up a fitting error measure that takes the imprecision of manual annotation into account. The fitting accuracy score is given strong weight for those vertices that manual labellers have localised accurately, and light weight for badly localised vertices.

We use the multiple label data available from [11] to define the ground truth shape and the fitting error function. A set of $n_I = 40$ images were labelled $n_L = 10$ times each (labels describe the $n_V = 68$ vertices of the model mesh used in [10]). These frontal pose, neutral expression, homogeneous illumination, face images are extracted from the AR database [9]. Each image shows a different individual. A probability distribution is computed for each image i and vertex v, as the mean $\mu_{i,v}$ over its n_L labels $x_{i,v,l}$ and a (2×2) covariance matrix $\Sigma_{i,v}$ as:

$$\mu_{i,v} = \frac{1}{n_L} \sum_{l=1}^{n_L} x_{i,v,l} \quad \text{and} \quad \Sigma_{i,v} = \frac{1}{n_L - 1} \sum_{l=1}^{n_L} (x_{i,v,l} - \mu_{i,v})^T (x_{i,v,l} - \mu_{i,v}).$$

These define what we dub 'manual labelling statistics'. Figure 1 shows face images overlaid with their manual labelling statistics, with each vertex represented by an ellipse showing its mean position and uncertainty. This methodology is in contrast to [11] in which a

single covariance matrix is computed for each vertex over all the n_I images. We believe that keeping a single covariance matrix for each vertex in each image makes sense since the visibility conditions may substantially differ from one image to the others for the same vertex. We want to preserve this information in the statitics.

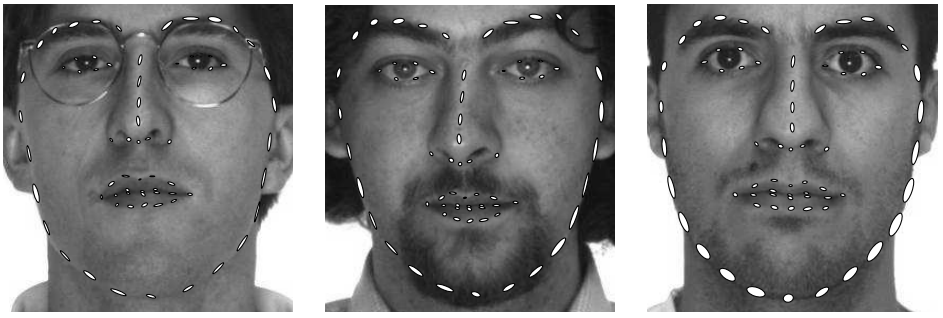


Figure 1: Faces number 1, 3 and 6 from the 40 faces that were annotated 10 times. Covariance ellipses represent the distribution of the 10 labels around mean vertices.

We propose the *Statistical Shape Error* (SSE) for a shape s on an image i that we define by the average of the Mahalanobis distances:

$$SSE_i(s) = \frac{1}{n_V} \sum_{v=1}^{n_V} \sqrt{(s_v - \mu_{i,v})^T \Sigma_{i,v}^{-1} (s_v - \mu_{i,v})}, \quad (1)$$

where s_v is the v-th vertex of shape s. The lower the SSE, the better the fitting accuracy. This error is strongly related to the negative log-likelihood of the parameters with respect to gaussian noise contamined labels. It scores automatic fits and can also be used to score manual fitting accuracy. In particular, we compute the SSE obtained by the 10 labellings on each image. From the 10 error scores on each image we retain the maximum and minimum scores, and compute the average score. This allows us to compare automatic fitting accuracy to manual fitting accuracy in §5.3, which gives a concrete idea of the accuracy that is reached.

4 Issues in Fitting AAMs to Unseen Faces

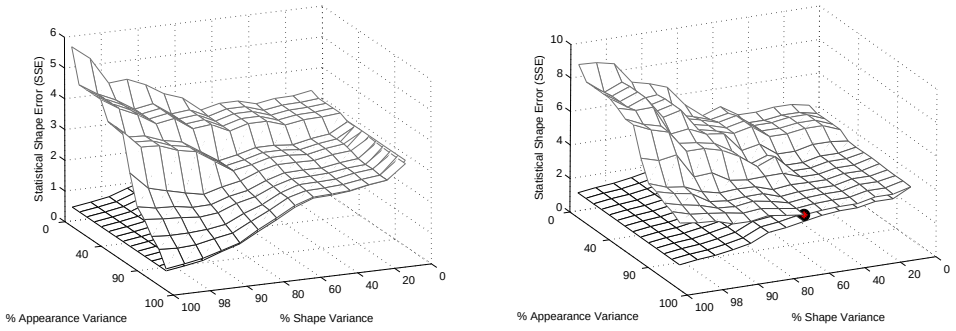
In the literature, AAMs are usually built by retaining 95 to 98% of the total shape and appearance variance contained in the training data without justifying this choice. Few works study the influence of the quantity of variance on the fitting performances. Gross *et al.* [7] recently investigated the effect of shape and appearance variance on the convergence of a fitting algorithm for unseen faces. They estimate the quantities of shape and appearance variance that maximize the number of successful trials. However, they do not explain why the convergence is limited for certain faces.

The experiment we report allows to highlight the fitting accuracy behavior for a range of shape and appearance variances. We identify the combination that maximizes the overall fitting accuracy on all trials and explain why this accuracy is limited and the fitting behaviour for various shape and appearance variance combinations. The experimental setup has similarities with the one in [7].

4.1 Fitting Seen Faces, *i.e.* Images in the Training Set

We use all the 40 images to train the AAM with different amounts of shape and appearance variance. We fit it to the 40 face images on turn. Each fitting trial lasts a number of iterations that allows to reach a clear final state, should it be convergence or divergence.

As in [7], we initialize the fitting process as close as possible to the optimal parameterization: we project the test face shape into the shape subspace to retrieve the initial shape parameters, and its appearance into the appearance subspace to retrieve the appearance parameters. In this way, we ensure that if the model diverges, it is not due to potential local minima but to the model inability to fit the test image. Figure 2 (a) shows the average SSE on all the 40 face images. The bottom curve shows the model SSE in its initial position, which is also the lowest error it can reach.



(a) Accuracy for **seen faces** (images in the training set)

(b) Accuracy for **unseen faces** (individuals not in the training set)

Figure 2: *Seen* and *unseen* contexts analysis. The bottom curves represent the initial SSE of equation (1) averaged over all the 40 images. The top curves show the average SSE after the algorithm has ran. Various amounts of shape and appearance variances are tested. The black dot in (b) represents the point of best average fitting accuracy on unseen faces that stands for 60% of the shape variance and 100% of the appearance variance.

We observe that for full appearance (100% of the variance retained), the fit remains in the best, initial position for any amount of shape variance retained. The characteristics of the full appearance AAM is that, up to appearance sampling artefacts, the test image can be completely reconstructed in appearance.

The second observation holds for any given fixed amount of shape variance: when less than 100% of the appearance variance is retained, the fitting accuracy decreases. The less the appearance variance, the worse the accuracy. For less than 100% of the training data, the AAM appearance space cannot totally reproduce the face appearance of the input image although this face is in the training set. This reluctant intensity discrepancy between the test image and the model causes the drop in the fitting accuracy. This test highlights the following property: the ability of the AAM appearance component to fully generate the face appearance of the test image is a necessary condition to obtain the best possible fitting accuracy. A natural question to answer is whether this also holds when the test image is not in the training set.

4.2 Fitting Unseen Faces, *i.e.* Individuals Not in the Training Set

This test is different from the one in §4.1 in that it is done in a leave-one-out manner: we train the AAM on 39 images and use the 40-th image as a test image of an unseen face. The test is performed for all the 40 face pictures, and for variable amounts of shape and appearance variance.

Figure 2 (b) shows the average SSE over the 40 face images. It is pretty similar to figure 2 (a) but a main difference is however observed. There is no combination of shape and appearance variances that makes the AAM remains into the initial, best position. Indeed, there is no junction between the SSE curves for the initial and fit curves. In contrast with the test on seen faces, even for full appearance AAMs, the fitting process shifts the AAM away from the initial solution. The AAM never remains on the best possible accuracy position, which results in limited accuracy capabilities.

4.3 Discussion

As an observation on the test for seen faces in §4.1, we saw that when the model can fully express the image in terms of appearance (the error in intensity between the model and the image are due to the model misplacement and/or non-optimal appearance parameterization). The fitting optimization process uses the error in intensity to iteratively update the model to a position where this error is minimized, and ideally equals zero. It is assumed that the model parameterization that minimizes the error in intensity correctly aligns the model to the face image. In practice, this is what happens when the model explicitly learnt the image it fits (and when the global minimum is reached). This explains the high fitting accuracy obtained in this context.

When the model appearance cannot fully express the face on the test image, the error in intensity due to this lack of expressivity is considered as being due to the model misplacement. The optimization process tunes the model parameters to minimize the residual error though it does not come from a misplacement. In this case, the minimum error usually does not correspond to the best placement of the model vertices. Indeed, the process bends the model in order to spread out the remaining error in intensity as much as it can to minimize the global error. This makes the model drift away from the sought after shape used as its initial position, *i.e.* fitting accuracy is spoiled. The more deformable the model the more the fitting process can bend it to further minimize error in intensity. For very high deformability the model can even diverge. In the same way for a given deformability (fixed amount of shape variance), the less the appearance variance, the less the model can express the test image data and the worse the fitting accuracy. In the case of fitting on seen faces, this happens when appearance variance is not fully retained (less than 100% of the variance is retained). In the case of fitting on unseen faces, a new face always presents visual aspects that are unknown from the model appearance component and the model always drifts away from the best possible position even when appearance is fully retained. As seen on the curves of figure 2 (b), the best overall accuracy for fitting on unseen faces is obtained for full appearance and 60% shape variance, making the model rigid enough not to bend too much, then minimizing the loss in fitting accuracy.

5 Segmented AAMs and Unseen Individuals

5.1 Motivations for Using Segmented AAMs

The AAM appearance space is unable to completely generate the appearance information. In other words, an unseen face added to the training set would bring new visual information. We saw that the limited ability to generalize the appearance component limits the fitting performance in terms of accuracy. One obvious solution to better generalize to any new face appearance would be to train the AAM on thousands of training images. This is difficult in practice for two reasons: first, this number of training data is hard to gather up, and second, this implies to retain a very high number of appearance components to explain as much of the variance as possible, which makes the optimization process computationally heavy and increases the possibility of getting stuck into local minima.

The solution we propose is to reduce the appearance space dimensionality. This makes more expressive the data coming from our reasonable size training set. To achieve a better fitting accuracy we rely on local models defined over a smaller face area. This approach is somehow similar to the concept of *segmented morphable models* briefly presented by Blanz and Vetter in [2].

5.2 Multi-Level Segmented AAMs and Coarse-to-Fine Fitting

The ability of local models to generalize their shape and appearance is better than for larger models. This makes them potentially more accurate for the same amount of training data. However, their reduced dimension penalizes their robustness to bad initialization: local models must be well initialized. To ensure this, we use a three stage coarse-to-fine strategy, illustrated on figure 3, where a global AAM is used to initialize intermediary AAMs, themselves used to initialize local AAMs.

Intermediary and local models represent a subgroup of the global model vertices. Models concerned with eyebrows also describe some extra vertices on lower eyebrows. A layer of *supporting points* is added to local and intermediary models in order to define visual gradients at 360° around all vertices.

The model is automatically initialized. We use a face and eye center detector available online[1] [6]. The rigid global model is transformed with a 2D similarity and is placed on the image such that its eye centers match the corresponding estimate given by the detector. From this initial position the fitting is launched until it converges.

Global model position is used to initialize each intermediary model: we keep the vertices the global model has in common with an intermediary model, and we find the intermediary model instance that best matches those vertices, as follows.

Let B_{succ} be the shape generating matrix of one intermediary model: the columns of B_{succ} are the n_C long deformation vectors s_i plus the four similarity transform vectors. Vector s_{curr} represents the vertex coordinates of the global model that are in common with the intermediary model. To this vector we add extra null coordinates for intermediary model vertices that are not in common with global model. s_{curr} thus becomes n_C long. We sort the coordinates in s_{curr} in a way such that they correspond to vertex coordinates defined in the vectors s_i. The instance of intermediary model that best matches its common vertices to those of the global model is found by solving the following optimization

[1] http://kolmogorov.sourceforge.net

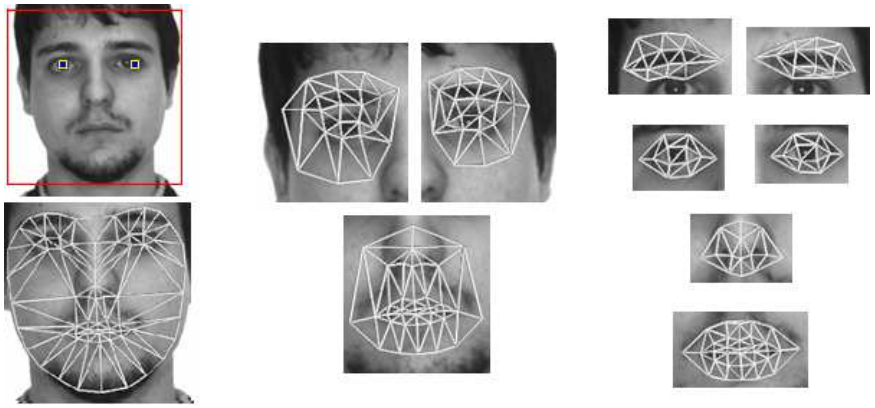


Figure 3: Illustration of the models used to fit the face. A global model is initialized with help of an eye center detector and is fitted on the image giving a first fitting result (left column). From this initialization a set of intermediary models are launched to further refine the fitting (center column). Eventually, the local models dedicated to each facial feature are launched to fit these features more accurately (right column).

problem:

$$\arg\min_{p} \sum_{c=1}^{n_C} Q(c)\left(s_{curr}(c) - B^c_{succ}p\right)^2, \tag{2}$$

where B^c_{succ} is the c^{th} row of matrix B_{succ}. Q is an n_C long vector of weights set to one for the coordinates of vertices that are common between the models, and to zero for the others. A closed form solution can be computed to find the optimal $p^\dagger$ (details are omitted due to lack of space):

$$p^\dagger = (K^T K)^{-1} K^T B^T_{succ} diag(Q) s_{curr}, \tag{3}$$

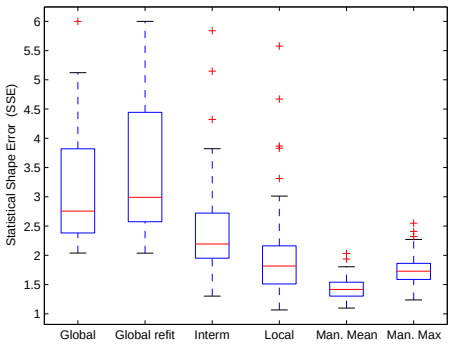
where $K = B^T_{succ} diag(Q) B_{succ}$ and $diag(Q)$ is a diagonal matrix, null everywhere excepted on diagonal where the Q vector coefficients are represented. The result $p^\dagger$ of this minimization can be used to instantiate the shape of the intermediary model: $s_{succ} = B_{succ} p^\dagger$. The process is applied to initialize all intermediary models that are then fitted to the image. Following the same strategy, we use (converged) intermediary model vertices to initialize local models that are in turn fitted to the image. Once each model is initialised in position, its appearance component is initialised by projection of the area underlying on the image onto the appearance subspace, in order to retrieve the initial appearance parameters.

5.3 Experimental Results

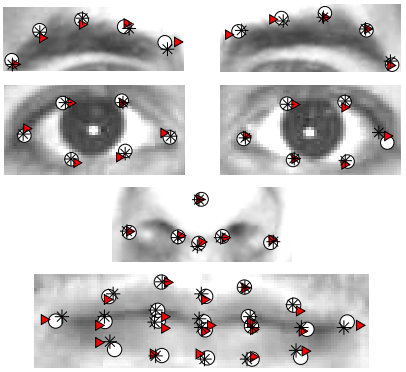
Fitting with intermediary and local models leads to improved fitting accuracy. On figure 4 (a), boxes of whiskers compare the SSE on 40 trials obtained respectively with global, global refitted, intermediary, local models and mean manual error as well as maximum manual error (see §3). The global refitted model is obtained by training the global model

onto refitted data: the used face data is learnt by an AAM retaining 99% variance of shape and appearance, and is fitted again on the same faces in order to increase the vertices correspondences among training data. Introduced in [7], this operation seems to improve the fitting results on unseen faces with respect to the results obtained when training the AAM on once-labelled data. Since we use multiply labelled data for training (the mean of 10 labels to define each vertex), their semantical position on the face is high and should naturally improve the correspondence among data.

A leave-one-out procedure is used to train and fit the global, global-refitted, intermediary and local models. Models are built with the shape and appearance variances that maximize their overall accuracy on unseen faces (*e.g.* 60% shape and 100% appearance for the global model). All intermediary models are gathered. The same is done for the local models. The SSE is computed using equation (1) only on vertices that are common between the models. We see the accuracy improvement allowed by intermediary and local models with respect to the global model, and we see that the accuracy is globally comparable to manual label accuracy evaluated with the statistics. The relative improvement obtained from global to local model fitting is 36% on average. The SSE obtained for refitted data is higher than for the global model trained with multiply labelled data. We believe that the higher semantical meaning obtained with label statistics is mainly responsible for the improvement. Indeed, these labels have higher semantical meaning since human labellers attempted several times to accurately set them into a given position on each face. The refitting process will displace once-labelled vertices to maximize their cohesion, but it is improbable that these new positions are semantically the very desired ones (although they might usually be improved). Multiply labelled data then constitute a maximum bound to accuracy, which explains the improved results obtained when we train an AAM with these data.



(a) Comparison results



(b) Example of fitting results

Figure 4: (a) Comparison between the 40 fitting scores obtained with a global model, a refitted global model, intermediary models, local models, and manual labellings, both maximum and average SSE. Local models often reach a SSE comparable to manual labellings. (b) Example of fitting results on face number 6. The circles represent the ground truth shape vertices (centers of the covariance ellipses), the triangles the vertices of the fitted global model (the SSE equals 2.46), and the stars represent the vertices of the fitted local models (the SSE equals 1.46).

6 Conclusion and Future Work

The AAM paradigm is often used without precisely understanding the influence of the *quantity and nature of training data* and of the *retained quantity of shape and appearance variance* on fitting performances. As a step towards such an understanding this work studies fitting accuracy on unseen frontal and neutral face data through the *Statistical Shape Error* we propose. We showed and explained the fitting accuracy limitations in this case. We propose a solution based on local models, namely the *multi-level segmented AAM*, that overcomes this limitation and reaches very high accuracy benchmarked by manual fitting accuracy with a large convergence basin. To summarize, standard and segmented AAMs are respectively well-adapted to person-specific and person independent face fitting.

We wish to extend these results to varying pose and expression: we will train one set of global, intermediary and local models for each possible pose and expression and set up a strategy to select the set that best suits for fitting the current face image.

References

[1] S. Baker, R. Gross, and I. Matthews. Lucas-Kanade 20 years on: A unifying framework: Part 3. Technical Report CMU-RI-TR-03-35, November 2003.

[2] V. Blanz and T. Vetter. A morphable model for the synthesis of 3D faces. In *Proceedings of* SIGGRAPH, 1999.

[3] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 23(6):681–685, 2001.

[4] D. Cosker, D. Marshall, P. Rosin, and Y.A. Hicks. Speech driven facial animation using a hierarchical model. IEE *VISP*, 151(4):314–321, 2004.

[5] D. Cristinacce and T. F. Cootes. Feature detection and tracking with constrained local models. In *Proceedings of the British Machine Vision Conference*, 2006.

[6] I. Fasel, B. Fortenberry, and J. R. Movellan. Generative framework for real-time object detection and classification. *CVIU*, 98(1):182–210, April 2005.

[7] R. Gross, I. Matthews, and S. Baker. Generic vs. person specific active appearance models. *Image and Vision Computing*, 23(11):1080–1093, 2006.

[8] S. Lucey, I. Matthews, C. Hu, Z. Ambadar, F. de la Torre, and J. Cohn. AAM derived face representation for robust facial action recognition. *FGR*, 2006.

[9] A.M. Martinez and R. Benavente. The AR face database. Technical Report 24, CVC, June 1998.

[10] I. Matthews and S. Baker. Active appearance models revisited. *International Journal of Computer Vision*, 60(2):135–164, November 2004.

[11] H. Mercier, J. Peyras, and P. Dalle. Toward an efficient and accurate AAM fitting on appearance varying faces. *FGR*, 2006.

9.1.2 Paper (CVPR'08) – *Light-Invariant Fitting of Active Appearance Models*

Light-Invariant Fitting of Active Appearance Models

Daniel Pizarro
Alcalà University - Madrid

Julien Peyras
LASMEA - Clermont-Ferrand

Adrien Bartoli
LASMEA - Clermont-Ferrand

Abstract

This paper deals with shading and AAMs. Shading is created by lighting change. It can be of two types: self-shading and external shading. The effect of self-shading can be explicitly learned and handled by AAMs. This is not however possible for external shading, which is usually dealt with by robustifying the cost function.

We take a different approach: we measure the fitting cost in a so-called Light-Invariant space. This approach naturally handles self-shading and external shading. The framework is based on mild assumptions on the scene reflectance and the cameras. Some photometric camera response parameters are required. We propose to estimate these while fitting an existing color AAM in a photometric 'self-calibration' manner.

We report successful results with a face AAM with test images taken indoor under simple lighting change.

1. Introduction

Active Appearance Models (AAMs) were introduced in [4] and since then have been the topic of many studies such as [2, 9]. They model the visual shape and appearance of an object or an object class and have been particularly successful to model faces. Face AAMs are used in this paper without loss of generality.

One of the main weaknesses of the AAMs is that there reliability generally degrades while the amount of variability increases in the training data. Sources of variability include person identity, expression, pose and shading. An AAM trained for several such sources will generally be prone to fall into local minima.

On the other hand, an AAM which has not been specifically trained to handle shading drifts away from the expected solution when the lighting changes. This makes AAMs useless for most of the applications where lighting conditions cannot be kept under control. Several recent papers consider this problem. Global illumination changes are modeled thanks to an affine normalization in the appearance basis [1]. Shaded areas are rejected as outliers thanks to a robust cost function [12]. These solutions are not fully satisfactory in an unconstrained lighting context.

Another approach is to build a rich training database including a large amount of lighting variations [11]. However, for many applications, we cannot reasonably pretend to be able to collect sufficiently many such training data. In addition, it is useful not to overload the appearance statistics but rather to separate the different sources of variability. Keeping the model complexity as low as possible usually results in improved fitting performances and better accuracy. A paper which recently followed this direction is [6]. It uses an Active Illumination Appearance (AIA) that models self-shading by additively combining two appearance bases, one for identity and one for self-shading.

We tackle the AAM fitting problem in the unconstrained illumination context. We do not explicitly model the appearance variations due to shading. This is based on the Light-Invariant transformation of [5]. From color training data acquired under any canonical illumination condition (which is also homogeneous in most cases) with a single camera, the *Light-Invariant AAM* fitting procedure we propose fits faces taken under uncontrolled illumination conditions. The only assumptions are a simplified model for the scene BRDF (Bidirectional Reflectance Distribution Function) and the photometric camera response. This is partly inspired by [10] in which direct Light-Invariant homography computation is successfully demonstrated.

Combining the Light-Invariant theory with AAMs not only allows us to efficiently fit AAMs to face images for which the lighting conditions are uncontrolled, it further allows synthesizing the canonical illumination appearance of the face. Thereafter, we assume that the canonical illumination is homogeneous. It has been shown that shadow free appearance reconstruction is in general an ill-posed problem [5]. We benefit from the strong prior knowledge that the observed object is known. We show successful shadow free face reconstruction. An overview of the proposed method is presented in Figure 1.

The paper is organized as follows. In §2 we describe the original AAM formulation and the Light-Invariant space. We discuss the possibility of combining the AAM frame-

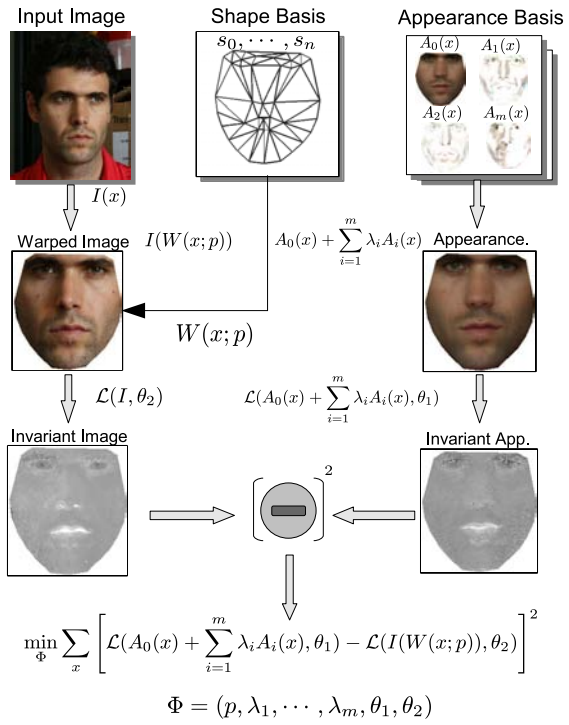


Figure 1. Overview of the proposed method.

work with the Light-Invariant image theory in §3. Two algorithms are introduced. The basic Light-Invariant AAM (LI-AAM) fitting, with its mathematical development, are given in §4. The photometric 'self-calibration' of a regular AAM is explained in §5. We consider the possibility to handle different training and test cameras. This allows us to reconstruct a shadow free RGB appearance, as described in §6. In §7, we experimentally compare the fitting performance obtained with classical fitting of a greylevel AAM, a color AAM, and various LI-AAM fittings. We conclude and give our perspectives in §8.

2. Background and General Points

2.1. Active Appearance Models

An AAM combines two linear subspaces, one for the shape and one for the appearance. They are both learnt from a labelled set of training images [4].

2.1.1 The Shape Counterpart

The shape of an AAM is defined by the N vertex coordinates of a mesh s describing the object boundaries and deformations:

$$s = \left(u_1, v_1, u_2, v_2, \cdots, u_N, v_N\right), \tag{1}$$

where u_i, v_i are the coordinates of vertex i. Principal Component Analysis (PCA) is applied to training shapes, centered on the mean shape s_0. A shape subspace $\boldsymbol{B}_s = [s_1, \cdots s_n]$ of $n-4$ shape eigencomponents is obtained, reducing the dimensionality of the training set. Four eigenvectors $s_{n-3}, \cdots, s_n$ are added to model 2D similarity transformations [9].

An instance of shape $s(p)$ is defined as a linear combination of shape eigenvectors with weights $p = (p_1, \cdots, p_n)$:

$$s(p) = s_0 + \sum_{i=1}^{n} p_i s_i. \tag{2}$$

Using the generated set of meshes parameterized by p, a warp function $W(x;p)$ is defined as a piecewise affine transformation from the base shape s_0 to the transformed mesh $s(p)$:

$$x' = W(x;p) \quad x = \begin{pmatrix} u \\ v \end{pmatrix} \quad x' = \begin{pmatrix} u' \\ v' \end{pmatrix}, \tag{3}$$

where $(u', v')^T$ is the warped coordinates.

2.1.2 The Appearance Counterpart

The training images are warped using $W(x;p)$ to normalize these in size. A photometric normalization is then applied to get the appearance training data. PCA is applied on these data in order to build a linear appearance subspace $\boldsymbol{B}_a = [A_1, \cdots, A_m]$ of $m-2$ image eigencomponents centered on the mean appearance image A_0. To compensate for bias and gain in the image intensity, the mean vector A_0 and a constant image A_I are added (corresponding to the $m-1^{th}$ and m^{th} components). This is the same process for color and greylevel images: the number of channels in the appearance images matches that of the training images.

An instance of appearance is defined as a linear combination of the $A_i(x)$ weighted by a set of parameters $\lambda = (\lambda_1, \cdots, \lambda_m)$:

$$A(x) = A_0(x) + \sum_{i=1}^{m} \lambda_i A_i(x). \tag{4}$$

2.1.3 Fitting

Fitting an AAM consists to find the shape and appearance parameters that make it best match the input image. A cost function is minimized with respect to the shape and appearance parameters:

$$\sum_x \left[A_0(x) + \sum_{i=1}^{m} \lambda_i A_i(x) - I(W(x;p)) \right]^2. \tag{5}$$

The cost function (5) evaluates the pixel value discrepancy between the warped input image and an instance of appearance of the model. An iterative Gauss-Newton minimization process is used to retrieve parameters p and λ.

2.2. Light-Invariant Image Theory

The transformation for Light-Invariant image formation is based on [5], which proposes a method for a single color image. This method relies on a simplified photometric image formation model where all surface materials follow a lambertian model, the lights are modeled as planckian sources and the camera sensor is narrowband. We consider faces to be lambertian to some extent. This assumption suffices for a range of lightings but might break down in cases such as specularities.

A more complex model of the camera is proposed in [8, 7], which takes into account nonlinearities in the image formation process (*i.e.* vignetting and the radiometric response function of the camera), and its usage for photometric alignment of images. Such a model allows to compensate for the camera response, which improves the applicability of a simplified image formation in general purpose cameras.

Given the three color components $\rho = (\rho_1, \rho_2, \rho_3)$, log chromaticity ratios are formed:

$$\mathcal{X}_1 = \log\left(\frac{\rho_1}{\rho_3}\right), \qquad \mathcal{X}_2 = \log\left(\frac{\rho_2}{\rho_3}\right). \tag{6}$$

An illumination invariant quantity $\mathcal{L}$ can be found by projecting $(\mathcal{X}_1, \mathcal{X}_2)$ along direction $\bar{e}^{\perp} = (\cos(\theta), \sin(\theta))$ parameterized by its angle θ from which:

$$\mathcal{L}(\rho, \theta) = \mathcal{X}_1(\rho)\cos(\theta) + \mathcal{X}_2(\rho)\sin(\theta). \tag{7}$$

This projection equates two colors corresponding to point viewed under different illuminations. It thus maps a color ρ to its corresponding Light-Invariant representation.

Transforming the value of each pixel of the color image $\mathcal{S}$ results in $\mathcal{L}(\mathcal{S}, \theta)$, a 1D shadow invariant image. This image transformation is global in that it does not depend on the pixel position $q \in \mathbb{R}^2$, but only on its color value.

The only relevant parameter governing the transformation is the angle θ of the invariant line, which only depends on the camera spectral properties. We elevate the RGB data to a $\log$, obtaining what we call the *logRGB space*. The transformation opportunely becomes linear:

$$\mathcal{L}(\rho, \theta) = L(\theta) \begin{pmatrix} \log(\rho_1) \\ \log(\rho_2) \\ \log(\rho_3) \end{pmatrix}, \tag{8}$$

$$\text{with } L(\theta) = \big(cos(\theta), sin(\theta)\big) \begin{pmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \end{pmatrix}.$$

Several solutions to estimate the angle θ are proposed in the literature. In [5] an off-line calibration step is proposed. It uses a color pattern or a set of preregistered images showing illumination changes. A 'self-calibration' approach is also presented. The strategy consists to find the angle for which the entropy of the invariant image is minimum. The later method is proved to be capable of finding the correct angle by using a single image where remarkable shaded areas are present. Despite its effectiveness in some cases, there are situations where it fails (*e.g.* in the presence of global illumination changes or soft shadows).

3. Combining AAMs and Light-Invariance

There are several possible ways of combining the AAMs with the Light-Invariant transformation $\mathcal{L}$. The general idea is to project the generated color image and the test color image to the Light-Invariant space, and to match them within that space:

$$\sum_x \left[A_0^{LI}(x) + \sum_{i=1}^{m} \lambda_i A_i^{LI}(x) - I^{LI}(W(x;p)) \right]^2, \tag{9}$$

where A_0^{LI}, all A_i^{LI} and I^{LI} are Light-Invariant images, pre-transformed from the 3D color space to the 1D Light-Invariant space. We therefore get a classical 1D dataset, and the problem resembles the one in (5) for greylevel images.

The main drawback of this approach, which limits the system usability, is that we need to know in advance the angle for the test data. As we explained above, there exist calibration methods but their applicability is restricted in several practical cases. The best thing would be to set up an approach which does not require a specific off-line calibration step.

Following the same philosophy as in [10], a more general solution is proposed by embedding the angles within the cost function. Under this approach, the angles can be estimated together with the shape and appearance parameters of the AAM.

To do so, the training of appearance is done in the logRGB space described in §2. It is linearly mapped to the invariant space with (8) and, as described in §6, allows us to reconstruct the appearance in RGB.

The resulting cost function is:

$$\sum_x \left[L(\theta_1) \left(\tilde{A}_0(x) + \sum_{i=1}^{m} \lambda_i \tilde{A}_i(x) \right) - L(\theta_2)(I(W(x;p)) \right]^2, \tag{10}$$

where I represents the logRGB test image, $\tilde{A}_i$s are the logRGB appearance components, θ_1 and θ_2 are the angle

parameters for the training sequence and the test image respectively. Using two different angles allows the system to work with different cameras (or the same camera with different photometric adjustments) for training and testing.

We examine two scenarios:

Case 1: Photometric 'self-calibration' of the training camera. The AAM can be 'self-calibrated' while fitting. If the training and test images are taken with the same camera $\theta_1 = \theta_2$. If two different cameras are used, we generally have $\theta_1 \neq \theta_2$. It is strictly necessary in both cases that changes in illumination arise between the training and the test images. If not, the angle or angles cannot be obtained and move randomly during optimization.

Case 2: Basic Light-Invariant AAM (LI-AAM) fitting. We assume that θ_1 is known (the AAM is photometrically calibrated). In practice, θ_2 is rarely known. We estimate it while fitting the AAM in the Light-Invariant space.

4. Basic Light-Invariant AAM fitting

The basic Light-Invariant AAM (LI-AAM) fitting includes the angle θ_2, needed to convert the test images to the invariant space. As we stated before the training is performed in logRGB so a linear appearance basis is obtained within that space.

The appearance basis is made of an average vector $\tilde{A}_0$ and the set of m appearance vectors $\tilde{A}_i$. The cost function is (10). It is minimized over p, λ and θ_2 using an additive Gauss-Newton algorithm. We refer the reader to [3] for more details.

5. Photometric Calibration of a Regular AAM

There are two possible situations:

- The same camera took the testing and training images.
  The cost function is minimized over p, λ and $\theta_1 = \theta_2$
- Different cameras took the testing and training images.
  The cost function is minimized over p, λ, θ_1 and θ_2

6. Shadow-Free Appearance Reconstruction

Fitting the AAM in the Light-Invariant space not only allows to handle uncontrolled illuminations but it also permits to retrieve the face appearance under the canonical illumination (the one in the training set). We conveniently choose to transform the RGB data to the logRGB since it makes the transformation 'more linear'. The second advantage of our choice is that when the AAM correctly fits the face on the input image, the synthesized logRGB appearance can easily be back-converted to the original RGB space.

Assuming that the color components are strictly positive, the logRGB space is invertible to RGB. Given the set of appearance parameters λ and the appearance basis in logRGB $\tilde{A}_i, \cdots, \tilde{A}_m$, the reconstructed RGB is:

$$A(x) = e^{\left(\tilde{A}_0(x) + \sum_{i=0}^{m} \lambda_i \tilde{A}_i(x)\right)} \tag{11}$$

where e is the element-wise exponential.

Such an operation is usually ill-posed when no prior information is provided on the image content [5]. In our case the AAM gives a unique solution.

Shadow-free appearance reconstruction can be useful for face recognition systems usually, exhibiting better performances under controlled illumination, and for image compression applications, where it is useful to encode facial and illumination data separately.

7. Experimental Results

To evaluate the performance of the above presented algorithms, we tested their robustness to noise and their resistance to initial geometric shape displacement. We measured a geometric fitting error using manually labelled images and the angle error to the ground-truth. So as to allow the reader to compare with various existing methods, we study the results of RGB, greylevel, and different Light-Invariant (LI) AAM fitting procedures:

- **LI-AAM:** Basic Light Invariant AAM fitting, proposed in §4, where only the input image angle θ_2 is unknown.
- **LI-AAM(LIT):** Basic Light Invariant AAM fitting, proposed in §4, where only the input image angle θ_2 is unknown but performing the training in LI space.
- **LI-AAM(PC):** Photometric calibration of the AAM proposed in §5. Both θ_1 and θ_2 are obtained.
- **LI-AAM(PC):** Photometric calibration of the AAM proposed in §5, imposing the constraint $\theta_1 = \theta_2$.
- **RGB-AAM:** A regular AAM fitted in color space.
- **Grey-AAM:** A regular AAM fitted in greylevel space.

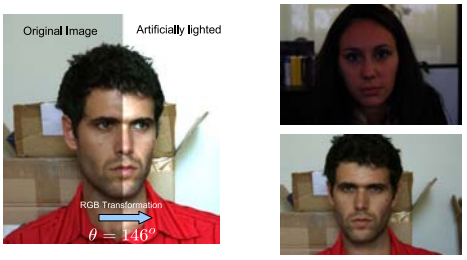
7.1. Simulated Perturbations

The whole set of algorithms are tested against image intensity noise and initial geometric displacement.

A person-specific AAM is trained using 6 face images taken under a mainly homogeneous illumination, and different poses.

The original images of the database are corrupted by noise with variance σ_n over the range $[0 - 255]$ and on each color channel. One of the images used for training is corrupted by noise and an artificial global change of illumination is created. The shadow area modifies the RGB color values of the image to simulate the response of a camera with angle $\theta = 146$ degrees perturbed by a light source

with a given color temperature T (See Figure 2.a). The initial position of the model mesh is obtained by displacing its points from the hand labelled solution by a random distance, the variance of which denoted γ.



(a) (b)

Figure 2. (a) Artificial perturbation on an original RGB image. (b) Extract of the homogeneously illuminated face images used for training.

We compare the performance of all algorithms for a range of perturbation parameters γ and σ_n. Figure 3.a shows the geometric error measured after a sufficient number of iterations to let each AAM converge (we used 50 iterations). On Figure 3.b and Figure 3.c, the geometric error and the angle error are tested against the noise (also measured after 50 iterations).

We observed that:

- LI-AAM and LI-AAM(LIT) perform exactly the same in all tests, so the use of logRGB for training appearance do as good as the direct use of the LI space.
- The image noise substantially affects all the proposed Light-Invariant methods. However LI-AAM shows to be the most robust.
- The initial geometric displacement substantially and equivalently affects all the Light-Invariant methods.

7.2. Real Illumination Changes

We tested the proposed solution on two sets of real data. Each set is composed of two sequences, both presenting one character changing head pose under neutral expression and finally smiles. On one sequence, the face is taken under homogeneous illumination as shown on Figure 2.b.

The presented color images are used to train the various AAMs. On the other sequence, the face is illuminated laterally as shown on Figure 4, and we compare the AAMs performance on this sequence. Figure 5 shows the evolution of the geometric error against the iteration number. As an observation, both LI-AAM and LI-AAM(LIT) present (equivalently) better convergence characteristics than the other LI methods. Also it clearly appears how the Grey-AAM and RGB-AAM quickly diverge.

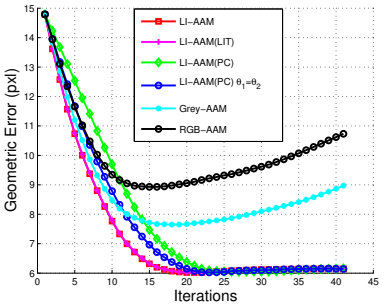


Figure 5. Geometric Error versus iteration number for the test image shown in Figure 4.a

8. Discussion

We tackled the difficult problem of Active Appearance Model fitting in an unconstrained illumination context. An AAM in its basic formulation is very sensitive to an unknown illumination.

Instead of classically matching the model appearance to the input image in the color space, we project the images, both from the synthesized and the input ones into a Light-Invariant space where the effect of shading are canceled. This only requires to tune a camera dependent angle θ, for the cameras used for shooting the training and the test datasets. The calibration of these parameters, necessary to the Light-Invariant transformation, is done jointly with the appearance and shape parameters. This is what we called the Light-Invariant AAM fitting.

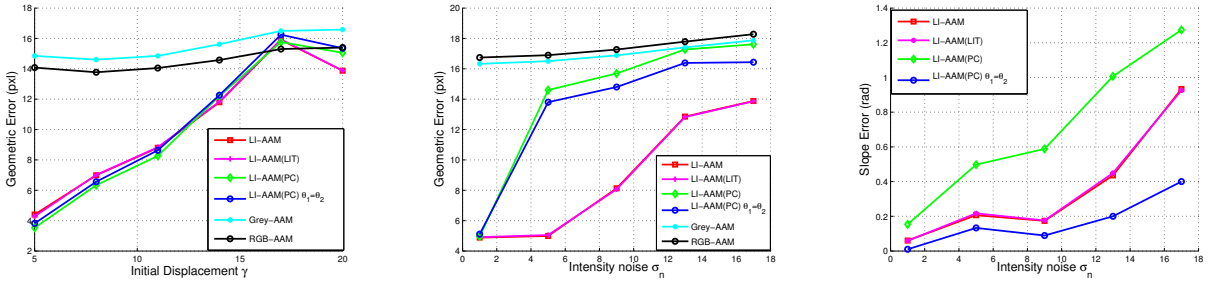
Two different scenarios are proposed: in the first acts the basic LI-AAM algorithm for which only the angle corresponding to the input image is unknown, and in the second is involved the photometric calibration of an existing AAM (LI-AAM(PC)) for which the angles corresponding to both the training and the test data (the cameras used to take them) are sought after.

Our approach does not need that the camera is precalibrated and makes the system useful to unsourced (color) image databases. In addition, we show how training the AAM in logRGB allows to reconstruct the RGB shadow-free appearance of a face after the fitting is performed.

Experimental results show that our proposal outperforms the regular AAM approaches (using RGB or greylevel values), when the input images present different illumination conditions.

References

[1] S. Baker, R. Gross, I. Matthews, and T. Ishikawa. Lucas-kanade 20 years on: A unifying framework: Part 2. Technical Report CMU-RI-TR-03-35, Robotics Institute. Carnegie Mellon University, Pittsburgh PA, 2003.



a) Geometric error versus γ b) Geometric error versus σ_n c) Angle error versus σ_n

Figure 3. Experiments using synthetic perturbations

a) LI-AAM b) RGB c) LI-AAM d) RGB e) LI-AAM f) RGB

g) LI-AAM h) RGB i) LI-AAM j) RGB k) LI-AAM m) RGB

Figure 4. Fitting on real data

[2] S. Baker and I. Matthews. Equivalence and efficiency of image alignment algorithms. *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, 1:1090–1097, 2001.

[3] S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221–255, 2004.

[4] T. Cootes, G. Edwards, and C.J. Taylor. Active appearance models. *Transactions on Pattern Analysis and Machine Intelligence*, 23(6):681–685, june 2001.

[5] Graham D. Finlayson, Steven D. Hordley, Cheng Lu, and Mark S. Drew. On the removal of shadows from images. *Transactions on Pattern Analysis and Machine Intelligence*, 28:59 – 68, January 2006.

[6] F. Kahraman, M. Gokmen, S. Darkner, and R. Larsen. An active illumination and appearance (aia) model for face alignment. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, June 2007.

[7] S. Kim and M. Pollefeys. Robust radiometric calibration and vignetting correction. *Transactions on Pattern Analysis and Machine Intelligence*, 30(4):562–576, 2008.

[8] S.J. Kim and M. Pollefeys. Radiometric self-alignment of image sequences. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, 2004.

[9] I. Matthews and S. Baker. Active appearance models revisited. *International Journal on Computer Vision*, 60(2):135–164, November 2004.

[10] D. Pizarro and A. Bartoli. Shadow resistant direct image registration. In *Scandinavian Conference on Image Analysis*, Aalborg, Denmark, June 2007.

[11] T. Sim and T. Kanade. Combining models and exemplars for face recognition: An illuminating example. *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, 2001.

[12] S. Yan, C. Liu, S.Z. Li, H. Zhang, H.Y. Shum, and Q. Cheng. Face alignment using texture-constrained active shape models. *Image and Vision Computing*, 21(1):69–75, 2003.

9.2 The Prediction Sum of Squares Statistic and Leave-One-Out Cross-Validation

On Computing the Prediction Sum of Squares Statistic in Linear Least Squares Problems with Multiple Parameter or Measurement Sets §9.1.1
A. Bartoli
Submitted to IEEE *Transactions on Neural Networks*, December 2007

N15 **Reconstruction de surface par validation croisée** §9.1.2
F. Brunet, A. Bartoli, R. Malgouyres et N. Navab
ROADEF'08 - *Journées de recherche opérationnelle et d'aide à la décision*, Clermont-Ferrand, France, février 2008

9.2.1 Paper – *On Computing the Prediction Sum of Squares Statistic in Linear Least Squares Problems with Multiple Parameter or Measurement Sets*

On Computing the Prediction Sum of Squares Statistic in Linear Least Squares Problems with Multiple Parameter or Measurement Sets

Adrien Bartoli
LASMEA (CNRS / Université Blaise Pascal), Clermont-Ferrand, France
Adrien.Bartoli@gmail.com
www.lasmea.univ-bpclermont.fr/Personnel/Adrien.Bartoli

Abstract

The prediction sum of squares is a useful statistic for comparing different models. As for linear least squares problems, there is a simple well-known non-iterative formula to compute it without having to refit the model as many times as the number of measurements. We extend this formula to cases where the problem has multiple parameter or measurement sets.

We report experimental results on the fitting of a warp between two images, for which the number of deformation centres is automatically selected, based on the non-iterative formula we derive.

1 Introduction

The prediction sum of squares (PRESS) is a statistic based on the leave-one-out technique. It was proposed by Allen in 1974 [1], and is typically used to compare different models. It is equivalent to the sum of studentized residuals, and can be extended to select parameters such as the regularization weight in smoothing splines, as shown by Wahba *et al.* [8] through leave-one-out cross-validation. For the case of regular linear least squares, it is well-known that there is a simple non-iterative formula giving the PRESS without having to solve as many problems as there are measurements[1] [4]. This has been derived for several variants of the basic linear least squares problem. For instance, Tarpey [6] examines the case of restricted least squares.

We derive non-iterative PRESS formulaes for those cases with multiple parameter or measurement sets. We report experimental results showing how one of the proposed non-iterative PRESS formulaes can be used to assess the fit of an inter-image warp borrowed from [2], and to automatically select the number of deformation centres for this warp.

In general, we write vectors in bold fonts, *e.g.* $\mathbf{x}$, matrices in sans-serif and calligraphic fonts, *e.g.* A, $\mathcal{W}$, and scalars in italics, *e.g.* m.

2 Background: Standard Linear Least Squares

Let $\mathbf{x}$ be the parameter vector, A the design matrix with m rows $\mathbf{a}_j$, $j = 1, \ldots, m$ and $\mathbf{b}$ the m measurement vector. Consider a regular linear least squares problem with the cost function:

$$\mathcal{E}^2_{\text{STD}}(\mathbf{x}) \stackrel{\text{def}}{=} \frac{1}{m}\sum_{j=1}^{m}\left(\mathbf{a}_j^\mathsf{T}\mathbf{x} - b_j\right)^2 = \frac{1}{m}\|\mathsf{A}\mathbf{x} - \mathbf{b}\|_2^2,$$

where $\|\mathbf{u}\|_2$ is vector two-norm, *i.e.* $\|\mathbf{u}\|_2 = \sqrt{\mathbf{u}^\mathsf{T}\mathbf{u}}$. The solution to this problem is:

$$\bar{\mathbf{x}} \stackrel{\text{def}}{=} \mathsf{A}^\dagger\mathbf{b},$$

with $\mathsf{A}^\dagger$ the matrix pseudo-inverse. The PRESS is defined by fitting the model without the j-th measurement, giving the parameter vector $\bar{\mathbf{x}}_{(j)}$. This is used to predict the j-th measurement as $\mathbf{a}_j^\mathsf{T}\bar{\mathbf{x}}_{(j)}$. This prediction is compared against the actual measurement b_j. This is averaged over the m measurements, giving:

$$\mathcal{P}^2_{\text{STD}} \stackrel{\text{def}}{=} \frac{1}{m}\sum_{j=1}^{m}\left(\mathbf{a}_j^\mathsf{T}\bar{\mathbf{x}}_{(j)} - b_j\right)^2.$$

[1] 'Non-iterative formula' emphasizes the fact that explicitly training the model with all but one measurement and testing on this

Directly using this formula for estimating the PRESS would be extremely inefficient since the model has to be fitted m times to compute all the $\bar{\mathbf{x}}_{(j)}$. It is well-known that there is a non-iterative formula giving the PRESS as:

$$\mathcal{P}^2_{\text{STD}} \;=\; \frac{1}{m}\left\|\Delta\left(\frac{1}{\mathbf{1}-\Delta\left(\hat{\mathsf{A}}\right)}\right)\left(\hat{\mathsf{A}}-\mathrm{I}\right)\mathbf{b}\right\|_2^2, \tag{1}$$

with $\mathbf{1}$ the $(n\times 1)$ 'all-one' vector, $\hat{\mathsf{A}}=\mathsf{A}\mathsf{A}^\dagger$ the hat matrix, I the identity matrix, and where Δ is the diag operator, similar as the one in MATLAB (*i.e.* it both extracts a matrix diagonal and constructs a diagonal matrix from a vector). Note that $\left(\hat{\mathsf{A}}-\mathrm{I}\right)\mathbf{b}=\mathsf{A}\bar{\mathbf{x}}-\mathbf{b}$, *i.e.* is the residual vector. Formula (1) is proved in *e.g.* [5].

3 Multiple Parameter or Measurement Sets

This section brings our main results in this paper. They are proved in section 4.

Multiple parameter and measurement sets. This kind of linear least squares problems has l sets of parameters and measurements represented in matrix form: L is the parameter matrix and R is the measurement matrix with rows $\mathbf{r}_j$. They both have n columns. Each of them is respectively a parameter and a measurement set, the former linked to the latter through the design matrix A. The cost function is as follows:

$$\mathcal{E}^2_{\text{MPM}}(\mathsf{L}) \;\stackrel{\text{def}}{=}\; \frac{1}{m}\sum_{j=1}^{m}\left\|\mathbf{a}_j^\mathsf{T}\mathsf{L}-\mathbf{r}_j^\mathsf{T}\right\|_2^2 \;=\; \frac{1}{m}\|\mathsf{A}\mathsf{L}-\mathsf{R}\|^2_{\mathcal{F}},$$

where $\|\mathsf{U}\|_{\mathcal{F}}$ is the matrix Frobenius norm, *i.e.* $\|\mathsf{U}\|_{\mathcal{F}}\stackrel{\text{def}}{=}\sqrt{\text{tr}(\mathsf{U}^\mathsf{T}\mathsf{U})}$. The solution to this problem is:

$$\bar{\mathsf{L}} \;\stackrel{\text{def}}{=}\; \mathsf{A}^\dagger\mathsf{R}.$$

The PRESS writes:

$$\mathcal{P}^2_{\text{MPM}} \;\stackrel{\text{def}}{=}\; \frac{1}{m}\sum_{j=1}^{m}\left\|\mathbf{a}_j^\mathsf{T}\bar{\mathsf{L}}_{(j)}-\mathbf{r}_j^\mathsf{T}\right\|_2^2,$$

and can be computed efficiently with the following non-iterative formula:

$$\mathcal{P}^2_{\text{MPM}} \;=\; \frac{1}{m}\left\|\Delta\left(\frac{1}{\mathbf{1}-\Delta\left(\hat{\mathsf{A}}\right)}\right)\left(\hat{\mathsf{A}}-\mathrm{I}\right)\mathsf{R}\right\|^2_{\mathcal{F}}, \tag{2}$$

which is exactly as (1) for the standard linear least squares case, except that the vector two-norm is replaced by the matrix Frobenius norm. This is demonstrated very easily by following the proof in [5], replacing the vector by the matrix norm. The intuition is that each column of R is independent, in the sense that the corresponding parameters lie in a different column in L, and that $\|\mathsf{U}\|^2_{\mathcal{F}}=\|\mathbf{u}_1\|_2^2+\|\mathbf{u}_2\|_2^2+\ldots$, where $\mathbf{u}_1$, $\mathbf{u}_2$, ..., are the columns[2] of matrix U. The problem can thus be split into n standard linear least squares problems, and their PRESS combined together to give (2).

Multiple measurement sets. We investigate the case where there is a single parameter vector with multiple measurement sets. In other words, each model prediction matches several measurements. This is modeled by the following cost function:

$$\mathcal{E}^2_{\text{MM}}(\mathbf{x}) \;\stackrel{\text{def}}{=}\; \frac{1}{m}\|\mathsf{C}\mathsf{L}-\mathsf{R}\|^2_{\mathcal{F}} \qquad \text{with} \qquad \mathsf{L} \;=\; \mathbf{x}\mathbf{1}^\mathsf{T},$$

where C is the m row design matrix. This is a particular case of the multiple scaled measurement sets described below. We can obviously not apply the standard PRESS formula since n linked measurements must be removed jointly. Holding out only one measurement at a time underestimates the PRESS.

[2] This obviously also holds with the rows.

The solution is obtained from (5) with $\boldsymbol{\omega} = \mathbf{1}$ as:

$$\bar{\mathbf{x}} \;=\; \mathsf{C}^{\dagger}\mathsf{R}\mathbf{1}.$$

The PRESS is defined by:

$$\mathcal{P}^2_{\mathrm{MM}} \;\stackrel{\text{def}}{=}\; \frac{1}{m}\sum_{j=1}^{m}\left\|\mathbf{c}_j^{\mathsf{T}}\bar{\mathbf{x}}_{(j)}\mathbf{1}^{\mathsf{T}} - \mathbf{r}_j^{\mathsf{T}}\right\|^2,$$

with $\mathbf{c}_j$ and $\mathbf{r}_j$ the rows of C and R respectively. The non-iterative PRESS formula we derive is:

$$\mathcal{P}^2_{\mathrm{MM}} \;=\; \frac{1}{m}\left\|\Delta\left(\frac{1}{\mathbf{1}-\Delta\left(\hat{\mathsf{C}}\right)}\right)\left(\hat{\mathsf{C}}\mathsf{R}1 - \mathsf{R} + \Delta\left(\Delta\left(\hat{\mathsf{C}}\right)\right)\mathsf{R}\left(\mathrm{I}-1\right)\right)\right\|^2_{\mathcal{F}}, \tag{3}$$

with $1 = \mathbf{1}\mathbf{1}^{\mathsf{T}}$ the 'all-one' $(n \times n)$ matrix. Specializing equation (7) with $\boldsymbol{\omega} = \mathbf{1}$ to get (3) is straightforward.

Multiple scaled measurement sets. This case generalizes the previous one by incorporating a different scale for each of the measurement sets, *i.e.* for each column in R, through an $(n \times 1)$ scaling vector $\boldsymbol{\omega}$:

$$\mathcal{E}^2_{\mathrm{MSM}}(\mathbf{x}) \;\stackrel{\text{def}}{=}\; \frac{1}{m}\|\mathsf{C}\mathsf{L} - \mathsf{R}\|^2_{\mathcal{F}} \qquad \text{with} \qquad \mathsf{L} \;=\; \mathbf{x}\boldsymbol{\omega}^{\mathsf{T}}. \tag{4}$$

The solution is:

$$\bar{\mathbf{x}} \;=\; \mathsf{C}^{\dagger}\mathsf{R}\boldsymbol{\omega}. \tag{5}$$

The PRESS is defined by:

$$\mathcal{P}^2_{\mathrm{MSM}} \;\stackrel{\text{def}}{=}\; \frac{1}{m}\sum_{j=1}^{m}\left\|\mathbf{c}_j^{\mathsf{T}}\bar{\mathbf{x}}_{(j)}\boldsymbol{\omega}^{\mathsf{T}} - \mathbf{r}_j^{\mathsf{T}}\right\|^2, \tag{6}$$

and we demonstrate below that the non-iterative PRESS formula is:

$$\mathcal{P}^2_{\mathrm{MSM}} \;=\; \frac{1}{m}\left\|\Delta\left(\frac{1}{\mathbf{1}-\Delta\left(\hat{\mathsf{C}}\right)}\right)\left(\hat{\mathsf{C}}\mathsf{R}\boldsymbol{\omega}\boldsymbol{\omega}^{\mathsf{T}} - \mathsf{R} + \Delta\left(\Delta\left(\hat{\mathsf{C}}\right)\right)\mathsf{R}\left(\mathrm{I}-\boldsymbol{\omega}\boldsymbol{\omega}^{\mathsf{T}}\right)\right)\right\|^2_{\mathcal{F}}. \tag{7}$$

This looks like the usual direct solution except that an extra term $\Delta\left(\Delta\left(\hat{\mathsf{C}}\right)\right)\mathsf{R}\left(\mathrm{I}-\boldsymbol{\omega}\boldsymbol{\omega}^{\mathsf{T}}\right)$ is added to the residual matrix $\hat{\mathsf{C}}\mathsf{R}\boldsymbol{\omega}\boldsymbol{\omega}^{\mathsf{T}} - \mathsf{R}$.

4 Proofs for the Multiple Scaled Measurement Sets Case

The solution in equation (5). We start by deriving the solution $\bar{\mathbf{x}}$ in equation (5). This equation means that $\bar{\mathbf{x}}$ is the $\boldsymbol{\omega}$ weighted sum of the solution for each set of measurements. It is derived by rewriting the cost function (4) as:

$$\mathcal{E}^2_{\mathrm{MSM}}(\mathbf{x}) \;=\; \frac{1}{m}\|\boldsymbol{\omega} \otimes \mathsf{C}\mathbf{x} - \mathbf{r}\|^2_2,$$

where $\mathbf{r} = \mathrm{vect}(\mathsf{R})$ is the column-wise vectorization of R, and $\otimes$ is the Kronecker product. From this rewriting, we obtain:

$$\bar{\mathbf{x}} \;=\; (\boldsymbol{\omega} \otimes \mathsf{C})^{\dagger}\mathbf{r},$$

which rewrites as:

$$\bar{\mathbf{x}} \;=\; \left(\boldsymbol{\omega}^{\mathsf{T}} \otimes \mathsf{C}^{\dagger}\right)\mathbf{r} \;=\; \mathsf{C}^{\dagger}\mathsf{R}\boldsymbol{\omega}.$$

The non-iterative PRESS formula (7). The next step is to derive the non-iterative PRESS formula (7). The proof follows similar steps as the proof for the basic non-iterative PRESS formula in [5]. We start from the PRESS definition (6). Defining $\mathbf{e}_j$ as a zero vector with one at the j-th element and $\mathsf{D}_j \stackrel{\text{def}}{=} \mathrm{I} - \Delta(\mathbf{e}_j)$, we have:

$$\bar{\mathbf{x}}_{(j)} \stackrel{\text{def}}{=} \arg\min_{\mathbf{x}} \|\mathsf{D}_j\mathsf{C}\mathbf{x}\boldsymbol{\omega}^\mathsf{T} - \mathsf{D}_j\mathsf{R}\|_\mathcal{F}^2.$$

Note that matrix D_j has the properties:

$$\begin{aligned} \mathsf{D}_j\mathsf{D}_j &= \mathsf{D}_j \\ \mathsf{D}_j^\mathsf{T} &= \mathsf{D}_j \\ \mathrm{I} - \mathsf{D}_j &= \Delta(\mathbf{e}_j). \end{aligned}$$

Similarly as for the solution (5), we get:

$$\bar{\mathbf{x}}_{(j)} = (\mathsf{D}_j\mathsf{C})^\dagger\mathsf{D}_j\mathsf{R}\boldsymbol{\omega}. \tag{8}$$

The lemma we demonstrate in the next paragraph states that:

$$\bar{\mathbf{x}}_{(j)} = \mathsf{C}^\dagger\tilde{\mathsf{R}}_j\boldsymbol{\omega}, \tag{9}$$

with $\tilde{\mathsf{R}}_j$ the measurement matrix R with the j-th row replaced by its prediction $\mathsf{C}\bar{\mathbf{x}}_{(j)}\boldsymbol{\omega}^\mathsf{T}$ with the model parameters $\bar{\mathbf{x}}_{(j)}$ rescaled by $\frac{1}{\|\boldsymbol{\omega}\|_2^2}$, *i.e.*:

$$\tilde{\mathsf{R}}_j \stackrel{\text{def}}{=} \mathsf{D}_j\mathsf{R} + \frac{1}{\|\boldsymbol{\omega}\|_2^2}(\mathrm{I} - \mathsf{D}_j)\mathsf{C}\bar{\mathbf{x}}_{(j)}\boldsymbol{\omega}^\mathsf{T}. \tag{10}$$

We note that:

$$(\mathrm{I} - \mathsf{D}_j)\mathsf{C}\bar{\mathbf{x}}_{(j)}\boldsymbol{\omega}^\mathsf{T} = \Delta(\mathbf{e}_j)\mathsf{C}\bar{\mathbf{x}}_{(j)}\boldsymbol{\omega}^\mathsf{T} = \mathbf{e}_j\mathbf{c}_j^\mathsf{T}\bar{\mathbf{x}}_{(j)}\boldsymbol{\omega}^\mathsf{T}.$$

The prediction of the j-th data with the global model $\bar{\mathbf{x}}$ is $\mathbf{c}_j^\mathsf{T}\bar{\mathbf{x}}\boldsymbol{\omega}^\mathsf{T}$. By substituting $\bar{\mathbf{x}}$ from equation (5), we get:

$$\mathbf{c}_j^\mathsf{T}\bar{\mathbf{x}}\boldsymbol{\omega}^\mathsf{T} = \mathbf{c}_j^\mathsf{T}\mathsf{C}^\dagger\mathsf{R}\boldsymbol{\omega}\boldsymbol{\omega}^\mathsf{T} = \hat{\mathbf{c}}_j^\mathsf{T}\mathsf{R}\boldsymbol{\omega}\boldsymbol{\omega}^\mathsf{T}, \tag{11}$$

where $\hat{\mathbf{c}}_j$ is the j-th row of the hat matrix $\hat{\mathsf{C}} = \mathsf{C}\mathsf{C}^\dagger$. Similarly, we rewrite the prediction of the j-th data with the partial model $\bar{\mathbf{x}}_{(j)}$ from equation (9) as:

$$\mathbf{c}_j^\mathsf{T}\bar{\mathbf{x}}_{(j)}\boldsymbol{\omega}^\mathsf{T} = \hat{\mathbf{c}}_j^\mathsf{T}\tilde{\mathsf{R}}_{(j)}\boldsymbol{\omega}\boldsymbol{\omega}^\mathsf{T}. \tag{12}$$

Taking the difference between the two predictions as rewritten in equations (11) and (12), and factorizing gives:

$$\hat{\mathbf{c}}_j^\mathsf{T}\mathsf{R}\boldsymbol{\omega}\boldsymbol{\omega}^\mathsf{T} - \hat{\mathbf{c}}_j^\mathsf{T}\tilde{\mathsf{R}}_{(j)}\boldsymbol{\omega}\boldsymbol{\omega}^\mathsf{T} = \hat{\mathbf{c}}_j^\mathsf{T}\left(\mathsf{R} - \tilde{\mathsf{R}}_{(j)}\right)\boldsymbol{\omega}\boldsymbol{\omega}^\mathsf{T}.$$

Using (10), we substitute $\tilde{\mathsf{R}}_j = \mathsf{D}_j\mathsf{R} + \frac{1}{\|\boldsymbol{\omega}\|_2^2}\mathbf{e}_j\mathbf{c}_j^\mathsf{T}\bar{\mathbf{x}}_{(j)}\boldsymbol{\omega}^\mathsf{T}$ which gives:

$$\hat{\mathbf{c}}_j^\mathsf{T}\left(\mathsf{R} - \mathsf{D}_j\mathsf{R} - \frac{1}{\|\boldsymbol{\omega}\|_2^2}\mathbf{e}_j\mathbf{c}_j^\mathsf{T}\bar{\mathbf{x}}_{(j)}\boldsymbol{\omega}^\mathsf{T}\right)\boldsymbol{\omega}\boldsymbol{\omega}^\mathsf{T},$$

which, simplifying $\mathsf{R} - \mathsf{D}_j\mathsf{R} = \mathbf{e}_j\mathbf{r}_j^\mathsf{T}$ and since $\boldsymbol{\omega}^\mathsf{T}\boldsymbol{\omega} = \|\boldsymbol{\omega}\|_2^2$, transforms to:

$$\hat{\mathbf{c}}_j^\mathsf{T}\left(\mathbf{e}_j\mathbf{r}_j^\mathsf{T}\boldsymbol{\omega} - \mathbf{e}_j\mathbf{c}_j^\mathsf{T}\bar{\mathbf{x}}_{(j)}\right)\boldsymbol{\omega}^\mathsf{T} = \hat{c}_{j,j}\left(\mathbf{r}_j^\mathsf{T}\boldsymbol{\omega} - \mathbf{c}_j^\mathsf{T}\bar{\mathbf{x}}_{(j)}\right)\boldsymbol{\omega}^\mathsf{T},$$

where $\hat{c}_{j,j}$ is the j-th diagonal element of the hat matrix $\hat{\mathsf{C}}$. We thus have rewritten the original prediction difference as follows:

$$\mathbf{c}_j^\mathsf{T}\bar{\mathbf{x}}\boldsymbol{\omega}^\mathsf{T} - \mathbf{c}_j^\mathsf{T}\bar{\mathbf{x}}_{(j)}\boldsymbol{\omega}^\mathsf{T} = \hat{c}_{j,j}\left(\mathbf{r}_j^\mathsf{T}\boldsymbol{\omega} - \mathbf{c}_j^\mathsf{T}\bar{\mathbf{x}}_{(j)}\right)\boldsymbol{\omega}^\mathsf{T}.$$

Rearranging the terms gives:

$$\mathbf{c}_j^\mathsf{T}\bar{\mathbf{x}}\boldsymbol{\omega}^\mathsf{T} = \hat{c}_{j,j}\mathbf{r}_j^\mathsf{T}\boldsymbol{\omega}\boldsymbol{\omega}^\mathsf{T} + (1 - \hat{c}_{j,j})\,\mathbf{c}_j^\mathsf{T}\bar{\mathbf{x}}_{(j)}\boldsymbol{\omega}^\mathsf{T}.$$

Adding $\mathbf{r}_j^\mathsf{T}\left(\hat{c}_{j,j}\mathrm{I} - \mathrm{I} - \hat{c}_{j,j}\boldsymbol{\omega}\boldsymbol{\omega}^\mathsf{T}\right)$ on both sides gives:

$$\mathbf{c}_j^\mathsf{T}\bar{\mathbf{x}}\boldsymbol{\omega}^\mathsf{T} + \mathbf{r}_j^\mathsf{T}\left(\hat{c}_{j,j}\mathrm{I} - \mathrm{I} - \hat{c}_{j,j}\boldsymbol{\omega}\boldsymbol{\omega}^\mathsf{T}\right) = (1 - \hat{c}_{j,j})\left(\mathbf{c}_j^\mathsf{T}\bar{\mathbf{x}}_{(j)}\boldsymbol{\omega}^\mathsf{T} - \mathbf{r}_j^\mathsf{T}\right)$$

from which:

$$\mathbf{c}_j^\mathsf{T}\bar{\mathbf{x}}_{(j)}\boldsymbol{\omega}^\mathsf{T} - \mathbf{r}_j^\mathsf{T} = \frac{1}{1-\hat{c}_{j,j}}\left(\mathbf{c}_j^\mathsf{T}\bar{\mathbf{x}}\boldsymbol{\omega}^\mathsf{T} - \mathbf{r}_j^\mathsf{T} + \mathbf{r}_j^\mathsf{T}\hat{c}_{j,j}\left(\mathrm{I} - \boldsymbol{\omega}\boldsymbol{\omega}^\mathsf{T}\right)\right).$$

We observe that $\mathbf{c}_j^\mathsf{T}\bar{\mathbf{x}}\boldsymbol{\omega}^\mathsf{T} - \mathbf{r}_j^\mathsf{T}$ is the residual vector for the j-th measurement. Replacing $\bar{\mathbf{x}}$ from equation (5) and summing the squared norm over j, we get the non-iterative PRESS formula:

$$\mathcal{P}^2_{\mathrm{MSM}} = \frac{1}{m}\left\|\Delta\left(\frac{1}{\mathbf{1}-\Delta\left(\hat{\mathsf{C}}\right)}\right)\left(\hat{\mathsf{C}}\mathsf{R}\boldsymbol{\omega}\boldsymbol{\omega}^\mathsf{T} - \mathsf{R} + \Delta\left(\Delta\left(\hat{\mathsf{C}}\right)\right)\mathsf{R}\left(\mathrm{I} - \boldsymbol{\omega}\boldsymbol{\omega}^\mathsf{T}\right)\right)\right\|^2_{\mathcal{F}}.$$

The lemma. We want to show that $\bar{\mathbf{x}}_{(j)}$ as defined by equation (8) is given by equation (9). We start by expanding the right-hand side of equation (9) by substituting $\tilde{\mathsf{R}}_j$ from (10), giving:

$$\begin{aligned}\mathsf{C}^\dagger\tilde{\mathsf{R}}_j\boldsymbol{\omega} &= \mathsf{C}^\dagger\mathsf{D}_j\mathsf{R}\boldsymbol{\omega} + \frac{1}{\|\boldsymbol{\omega}\|_2^2}\mathsf{C}^\dagger(\mathrm{I}-\mathsf{D}_j)\mathsf{C}\bar{\mathbf{x}}_{(j)}\boldsymbol{\omega}^\mathsf{T}\boldsymbol{\omega}\\ &= \mathsf{C}^\dagger\mathsf{D}_j\mathsf{R}\boldsymbol{\omega} + \mathsf{C}^\dagger\mathsf{C}\bar{\mathbf{x}}_{(j)} - \mathsf{C}^\dagger\mathsf{D}_j\mathsf{C}\bar{\mathbf{x}}_{(j)}.\end{aligned}$$

The second term reduces to $\bar{\mathbf{x}}_{(j)}$ since $\mathsf{C}^\dagger\mathsf{C} = \mathrm{I}$. By replacing $\bar{\mathbf{x}}_{(j)}$ by its expression (8), the third term expands as:

$$\begin{aligned}\mathsf{C}^\dagger\mathsf{D}_j\mathsf{C}\bar{\mathbf{x}}_{(j)} &= \left(\mathsf{C}^\mathsf{T}\mathsf{C}\right)^{-1}\mathsf{C}^\mathsf{T}\mathsf{D}_j\mathsf{C}\left(\mathsf{C}^\mathsf{T}\mathsf{D}_j\mathsf{C}\right)^{-1}\mathsf{C}^\mathsf{T}\mathsf{D}_j\mathsf{R}\boldsymbol{\omega}\\ &= \mathsf{C}^\dagger\mathsf{D}_j\mathsf{R}\boldsymbol{\omega},\end{aligned}$$

and the overall expression simplifies to:

$$\mathsf{C}^\dagger\tilde{\mathsf{R}}_j\boldsymbol{\omega} = \bar{\mathbf{x}}_{(j)}.$$

5 Application to Estimating Rigid Affine Thin-Plate Spline Warps

A warp is a geometric $\mathbb{R}^2 \to \mathbb{R}^2$ transformation matching corresponding pixels between two images. Estimating a warp from point correspondences in two images is one of the most important problems in fields such as computer vision, medical image analysis, photogrammetry and augmented reality. The warp is often parameterized by some smooth transformation driven by deformation centres. This section shows an application of the non-iterative PRESS formula for multiple scaled measurement sets to the problem of assessing the quality of an image warp called the Rigid Affine Thin-Plate Spline Warp (RA-Warp) and originally proposed in [2]. We show that the number of deformation centres can be selected by minimizing the PRESS, which corresponds to maximizing the predictivity of the warp.

5.1 The RA-Warp and its PRESS

A warp is an $\mathbb{R}^2$ to $\mathbb{R}^2$ function that models the deformation between two images. The RA-Warp is dedicated to the case of two images of a rigid smooth surface, and models the cameras as affine, *i.e.* parallel, projections. An example is shown on figure 1.

Let $\mathbf{q} \in \mathbb{R}^2$ be the coordinates of a point in the first image. The RA-Warp depends on a parameter vector $\boldsymbol{\delta} \in \mathbb{R}^l$ and writes as:

$$\mathcal{W}(\mathbf{q};\boldsymbol{\delta}) \stackrel{\text{def}}{=} \mathcal{S}\tilde{\mathbf{q}} + \tau(\mathbf{q};\boldsymbol{\delta})\mathbf{s} \qquad \text{with} \qquad \tilde{\mathbf{q}}^\mathsf{T} = \left(\mathbf{q}^\mathsf{T}\ \ 1\right),$$

where τ is an $\mathbb{R}^2$ to $\mathbb{R}$ Thin-Plate Spline as derived by Duchon [3], and $(\mathcal{S};\mathbf{s})$ are camera parameters[3] that we estimate from image point correspondences as described in *e.g.* [7]. This warp guarantees that the points move along epipolar lines.

The parameter vector $\boldsymbol{\delta}$ of the Thin-Plate Spline τ contains the depth of some l deformation centres. The Thin-Plate Spline writes $\tau(\mathbf{q};\boldsymbol{\delta}) = \ell_\mathbf{q}^\mathsf{T}\mathcal{K}\boldsymbol{\delta}$, where $\mathcal{K}$ is a constant matrix depending on the deformation centres in the first image, and $\ell_\mathbf{q}$ is a nonlinear lifting function depending on point $\mathbf{q}$.

[3]These are the parameters of the second camera projection matrix in the canonical basis of the 3D space for which the first camera

Figure 1: The two example images we use, overlaid with the $m = 70$ points correspondences and corresponding epipolar lines. These two images show a bed sheet wraping a chair. They were taken while strongly zooming, making the camera close to affine. The bed sheet remained still between the two snapshots and thus only the relative position and orientation of the camera changed. The epipolar lines depend on this relative camera motion which is computed from the point correspondences.

Given m point correspondences $\mathbf{q}_j \leftrightarrow \mathbf{q}'_j$, the camera parameters $(\mathcal{S}; \mathbf{s})$ and the centres in $\mathcal{K}$, we estimate the parameter vector $\boldsymbol{\delta}$ by minimizing the following cost function, measuring the euclidean distance between the points in the second image and those transferred by the warp from the first image:

$$\mathcal{E}^2_{\mathrm{RA}}(\boldsymbol{\delta}) \stackrel{\mathrm{def}}{=} \sum_{j=1}^{m} \|\mathcal{W}(\mathbf{q}_j, \boldsymbol{\delta}) - \mathbf{q}'_j\|_2^2 = \sum_{j=1}^{m} \|\mathcal{S}\tilde{\mathbf{q}}_j + \tau(\mathbf{q}_j; \boldsymbol{\delta})\mathbf{s} - \mathbf{q}'_j\|_2^2.$$

Substituting the expression of the Thin-Plate Spline τ, we get:

$$\mathcal{E}^2_{\mathrm{RA}}(\boldsymbol{\delta}) = \sum_{j=1}^{m} \left\| \mathcal{S}\tilde{\mathbf{q}}_j + \left(\ell_{\mathbf{q}_j}^{\mathsf{T}} \mathcal{K}\boldsymbol{\delta}\right)\mathbf{s} - \mathbf{q}'_j \right\|_2^2.$$

Transposing and gathering all the measurements in matrices $\tilde{\mathcal{Q}}$ and $\mathcal{Q}'$ which rows are $\tilde{\mathbf{q}}_j$ and $\mathbf{q}'_j$ respectively, and the vectors $\ell_{\mathbf{q}_j}$ as the rows of $\mathcal{M}$, gives:

$$\mathcal{E}^2_{\mathrm{RA}}(\boldsymbol{\delta}) = \left\| \tilde{\mathcal{Q}}\mathcal{S}^{\mathsf{T}} + (\mathcal{M}\mathcal{K}\boldsymbol{\delta})\mathbf{s}^{\mathsf{T}} - \mathcal{Q}' \right\|_{\mathcal{F}}^2.$$

We identifie this as a multiple scaled measurement sets linear least squares problem, as given by equation (4). We get gives the solution from equation (5) as:

$$\bar{\boldsymbol{\delta}} = (\mathcal{M}\mathcal{K})^{\dagger} \left(\mathcal{Q}' - \tilde{\mathcal{Q}}\mathcal{S}^{\mathsf{T}}\right)\mathbf{s}. \tag{13}$$

Using the non-iterative PRESS formula (7), we obtain the PRESS statistic for the RA-Warp as:

$$\mathcal{P}^2_{\mathrm{RA}} \stackrel{\mathrm{def}}{=} \frac{1}{m} \left\| \Delta\left(\frac{1}{1 - \Delta\left(\widehat{\mathcal{M}\mathcal{K}}\right)}\right) \left(\widehat{\mathcal{M}\mathcal{K}}\left(\mathcal{Q}' - \tilde{\mathcal{Q}}\mathcal{S}^{\mathsf{T}}\right)\mathbf{s}\mathbf{s}^{\mathsf{T}} - \mathcal{Q}' + \tilde{\mathcal{Q}}\mathcal{S}^{\mathsf{T}} + \Delta\left(\Delta\left(\widehat{\mathcal{M}\mathcal{K}}\right)\right)\left(\mathcal{Q}' - \tilde{\mathcal{Q}}\mathcal{S}^{\mathsf{T}}\right)\left(\mathrm{I} - \mathbf{s}\mathbf{s}^{\mathsf{T}}\right)\right) \right\|_{\mathcal{F}}^2. \tag{14}$$

5.2 Estimation Algorithm

In practice, we are given a set of point correspondences from which we can estimate the camera parameters in $(\mathcal{S}; \mathbf{s})$. We do not however know how many deformation centres l are needed, and where to place them in the first image. A sensible choice, though heuristic, is to choose as deformation centres the vertices of a regular, square grid located in the vicinity of the data points.

Choosing the number of centres is more critical: underestimating l makes the warp too constrained to explain the measurements, while overestimating l makes it too flexible, possibly ill-conditioned, with bad predictivity. We propose to use the PRESS statistic in order to choose the number of centres. We start with a coarse square control grid of, say, $l_{\min} = 2^2 = 4$ centres, and subdivide it while monitoring the PRESS, until the maximum possible number of centres $l_{\max} = \lfloor \sqrt{m} \rfloor^2$ is reached.

5.3 Experimental Results

For the example data shown in figure 1, we have $m = 70$ point correspondences. We thus try to fit the warp driven by grids of $l_{\min} = 4, 9, 16, 25, 36, 49, 64 = l_{\max}$ deformation centres. The fitting Root Mean Square Residual (RMSR) and PRESS as functions of the number of deformation centres are shown in figure 2.

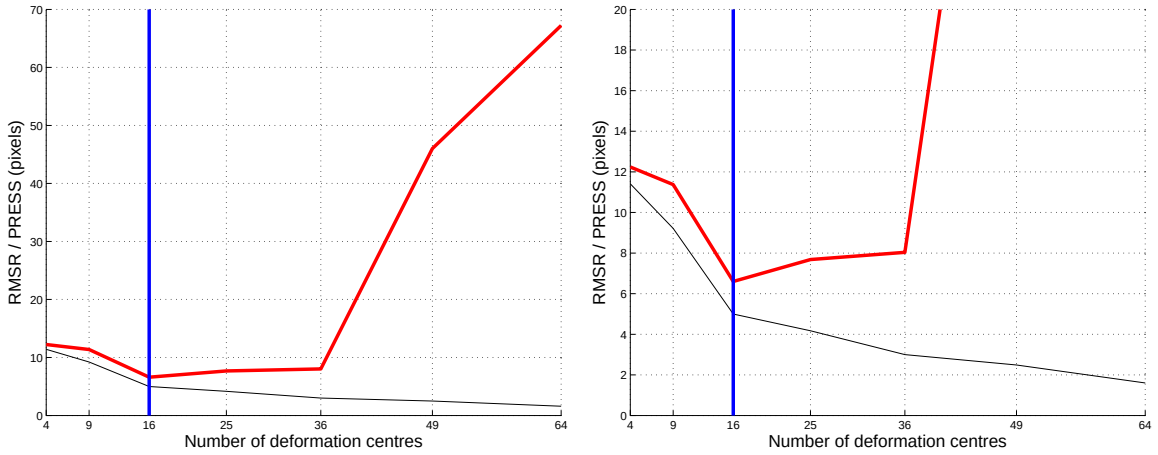
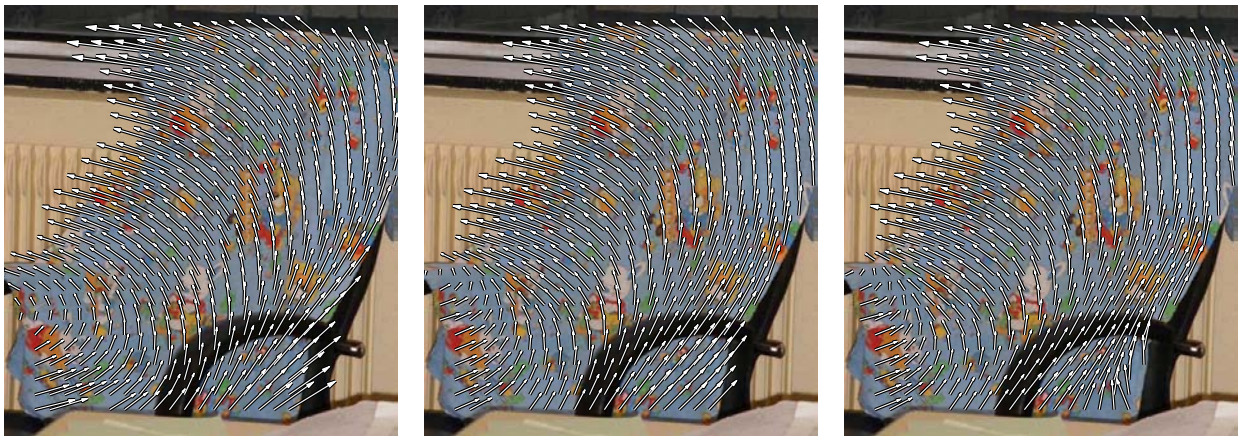


Figure 2: The fitting residual (RMSR, thin curve) and prediction sum of squares (PRESS, thick curve) as functions of the number of deformation centres. The vertical line shows the selected number of centres corresponding to the minimum PRESS. The two graphs are similar: the right one is a zoom on the left one.

We observe that the residual RMSR decreases as the number of deformation centres increases. This is to be expected since the most parameters in the model the least the fitting error. The PRESS decreases until 16 deformation centres are reached, and then grows as the number of deformation centres increases beyond 16.

This is explained as follows: for less than 16 deformation centres, the warp is not flexible enough to model the actual image deformations, while for more than 16 deformation centres, the warp is too flexible and is less and less well constrained by the point correspondences. In both cases, it fails to accurately capture the deformation, and thus does not interpolate well the data leading to a bad predictivity.

The so-called flow field is the set of displacement vectors at each pixel in one of the two images. The flow field sub-sampled to $\frac{1}{20^2}$ pixels is shown for different numbers of deformation centres in figure 3. Noticeable differences can be seen, though it is difficult to visually figure out which solution is the best one. This is better seen by observing the color discrepancy image, computed by warping the second image onto the first one and taking the absolute value of their difference. This is shown on figure 4 over a region of interest defined as the area covered by the bed sheet in the source image. The color discrepancy, computed as the Root Mean Square of the color discrepancy image, is lower for 16 deformation centres than for 9 and 25. It also is visually seen that for 16 deformation centres, only the area under the arm of the chair shows high discrepancy, while the rest of the bed sheet area is correctly registered, which is not the case for 9 and 25 deformation centres.



9 deformation centres | 16 deformation centres | 25 deformation centres

Figure 3: The estimated flow field for different numbers l of deformation centres. The solution minimizing the PRESS is with $l = 16$ deformation centres.

9 deformation centres
color discrepancy: 67.02

16 deformation centres
color discrepancy: 59.90

25 deformation centres
color discrepancy: 63.84

Figure 4: The color discrepancy images for different numbers l of deformation centres. Black indicates low discrepancy, and white indicates high discrepancy. The solution minimizing the PRESS is with $l = 16$ deformation centres.

6 Conclusion

We derived non-iterative formulaes for the Prediction Sum of Squares (PRESS) statistic for linear least squares with multiple parameter or measurement sets. As an application, we showed that this can be used to assess the quality of an image warp. It also allows selecting its complexity by varying the number of deformation centres and shows to be consistent with the color discrepancy image.

There are some open research directions with the formulaes we propose. The first one is to extend to the case of restricted least squares, as was done in [6] for standard linear least squares. The second one is to investigate how it can be used to select a regularization parameter in damped least squares, by replacing the hat matrix by the influence matrix, which is related to cross-validation methods.

References

[1] D. M. Allen. The relationship between variable selection and data augmentation and a method for prediction. *Technometrics*, 16:125–127, 1974.

[2] A. Bartoli, M. Perriollat, and S. Chambon. Generalized Thin-Plate Spline warps. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, 2007.

[3] J. Duchon. Interpolation des fonctions de deux variables suivant le principe de la flexion des plaques minces. *RAIRO Analyse Numrique*, 10:5–12, 1976.

[4] J. E. Gentle, W. Hardle, and Y. Mori. *Handbook of Computational Statistics*. Springer-Verlag, 2004.

[5] D. Montgomery and E. Peck. *Introduction to Linear Regression Analysis*. Wiley, New York, USA, 1992.

[6] T. Tarpey. A note on the prediction sum of squares statistic for restricted least squares. *The American Statistician*, 54(2):116–118, May 2000.

[7] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *International Journal of Computer Vision*, 9(2):137–154, November 1992.

[8] G. Wahba and S. Wold. A completely automatic french curve: Fitting spline functions by cross-validation. *Communications in Statistics*, 4:1–17, 1975.

9.2.2 Paper (ROADEF'08) – *Reconstruction de surface par validation croisée*

Reconstruction de surface par validation croisée

F. Brunet[1,2,3], A. Bartoli[1], R. Malgouyres[2], et N. Navab[3]

[1] LASMEA UMR 6602, Clermont-Ferrand, France
adrien.bartoli@lasmea.univ-bpclermont.fr
[2] LAIC EA 2446, Clermont-Ferrand, France
{brunet,malgouyres}@laic.u-clermont1.fr
[3] CAMP, Technische Universität München, Allemagne
navab@cs.tum.edu

1 Introduction

Nous nous intéressons à la reconstruction d'une surface à partir d'un nuage de points. C'est un problème ayant de nombreuses applications dans des domaines variés tels que l'étude de données statistiques, l'analyse d'images médicales ou encore la CAO. Nous abordons ici plus particulièrement le choix du compromis entre l'attache aux données et la régularité de la surface. L'approche considérée pour résoudre ce problème consiste à minimiser le score de validation croisée. Bien que de nombreux travaux aient déjà été menés à ce sujet, les méthodes utilisées pour optimiser le critère de la validation croisée ne sont que rarement explicités. De plus les méthodes habituellement décrites sont peu satisfaisantes.

2 La reconstruction de surface

Soit $X = \{(x_i \leftrightarrow y_i) \mid i \in \{1,\dots,n\},\ x_i \in \mathbb{R}^N,\ y_i \in \mathbb{R}^M\}$ un ensemble de données. En pratique, nous considérons principalement le cas $N = 2$ et $M = 1$, c'est-à-dire le cas où y_i représente une élévation par rapport au plan contenant l'ensemble des points x_i. Considérons le modèle de données suivant :

$$\mathbf{y} = f(\mathbf{x};\mathbf{p}) + \epsilon \tag{1}$$

où $\mathbf{x} = \begin{bmatrix} x_1 \dots x_n \end{bmatrix}^\mathsf{T}$, $\mathbf{y} = \begin{bmatrix} y_1 \dots y_n \end{bmatrix}^\mathsf{T}$ et $\mathbf{p}$ est un vecteur donnant les paramètres du modèle. Le problème d'approximation surfacique consiste à trouver le vecteur $\hat{\mathbf{p}}_\lambda$ tel que :

$$\hat{\mathbf{p}}_\lambda = \arg\min_{\mathbf{p}} \mathcal{E}(\mathbf{p},\lambda) \qquad \text{avec} \qquad \mathcal{E}(\mathbf{p},\lambda) = \mathcal{E}_d(\mathbf{p}) + \lambda \mathcal{E}_r(\mathbf{p}) \tag{2}$$

où $\mathcal{E}_d$ est un terme donnant l'énergie d'attache aux données, $\mathcal{E}_r$ quantifie la régularité du modèle et λ contrôle le compromis entre ces deux aspects. Nous ne considérons ici que les modèles linéaires en $\mathbf{p}$ (où A_λ est une matrice dépendant de λ, de $\mathbf{x}$ et du modèle f et où $\tilde{\mathbf{y}}$ contient $\mathbf{y}$) :

$$A_\lambda \mathbf{p} = \tilde{\mathbf{y}} \tag{3}$$

Les termes d'attache aux données et de régularisation sont souvent choisis comme étant respectivement la moyenne des résidus au carré et l'énergie de torsion [2,5,7] :

$$\mathcal{E}_d(\mathbf{p}) = \tfrac{1}{n}\textstyle\sum_{i=1}^n \left(f(x_i;\mathbf{p}) - y_i\right)^2 \qquad \text{et} \qquad \mathcal{E}_r(\mathbf{p}) = \iint_{\mathbb{R}^2} \left\| \tfrac{\partial f}{\partial \mathbf{x}}(\mathbf{x};\mathbf{p}) \right\|^2 \mathrm{d}\mathbf{x} \tag{4}$$

Il a été montré [7] qu'en utilisant de telles énergies, le problème peut être écrit sous la forme de l'équation (3) si l'on utilise comme modèle les *Thin Plate Splines* [1] :

$$f\left(\mathbf{x} = \begin{bmatrix}\alpha\ \beta\end{bmatrix}^\mathsf{T}; \mathbf{p} = \begin{bmatrix} w_1 \dots w_n\ a\ b\ c \end{bmatrix}^\mathsf{T}\right) = a\alpha + b\beta + c + \textstyle\sum_{i=1}^n w_i \rho(x, x_i) \tag{5}$$

$$\text{où } \rho(\mathbf{a},\mathbf{b}) = \|\mathbf{a}-\mathbf{b}\|^2 \log\|\mathbf{a}-\mathbf{b}\| \tag{6}$$

Pour un paramètre λ fixé, la solution du problème est donnée par [6,7,3] :

$$\hat{\mathbf{p}}_\lambda = \arg\min_{\mathbf{p}} \|A_\lambda \mathbf{p} - \mathbf{y}\|_2^2 \qquad \Leftrightarrow \qquad \hat{\mathbf{p}}_\lambda = A_\lambda^\dagger \mathbf{y} \tag{7}$$

Le paramètre λ ne peut évidemment pas être choisi en minimisant directement $\mathcal{E}(\mathbf{p},\lambda)$. En effet, cette approche conduirait systématiquement à la solution $\lambda = 0$. Nous sélectionnons λ avec une autre technique : la validation croisée [8].

3 La validation croisée et son optimisation

Soit $\hat{\mathbf{p}}_\lambda^{[i]}$ le paramètre minimisant l'énergie du modèle pour λ fixé en considérant le sous-ensemble de données $X^{[i]} = X \setminus \{(x_i \leftrightarrow y_i)\}$. La fonction de validation croisée V est alors donnée par :

$$V(\lambda) = \frac{1}{n} \sum_{i=1}^{n} \|f(\mathbf{x}; \hat{\mathbf{p}}_\lambda^{[i]}) - y_i\|^2 \tag{8}$$

La sélection automatique du compromis entre l'attache aux données et la régularisation consiste alors à déterminer le paramètre $\hat{\lambda} \in [0, \infty[$ minimisant la fonction V.

Bien qu'expérimentalement l'allure de la fonction V semble être adéquate (V n'est pas convexe mais a un seul minimum, voir figure 1), sa minimisation se heurte à plusieurs problèmes. L'utilisation directe de la formule (8) entrainerait le calcul de n ajustements de surfaces pour une seule valeur de λ ce qui serait bien trop coûteux. Il est possible de montrer [7] que l'expression de V peut être ramenée, demanière équivalente, à :

$$V(\lambda) = \frac{1}{n} \left\| \mathrm{diag}\left(\frac{\mathbf{1}}{\mathbf{1} - \mathrm{diag}(A_\lambda A_\lambda^+)} \right) (A\hat{\mathbf{p}}_\lambda - \mathbf{y}) \right\|^2 \tag{9}$$

La complexité de l'évaluation ponstuelle de V est alors équivalente à celle du calcul d'un seul ajustement. Cependant, cette complexité reste trop élevée pour que le minimum puisse être déterminé en échantillonant suffisamment finement la fonction V. De plus, la précision limitée des ordinateurs entraîne des erreurs de calcul lors de l'évaluation directe de la formule (9). Ces erreurs nous empêchent de mettre en œuvre des approches simples basées, par exemple, sur une dichotomie (les erreurs pouvant donner V comme décroissante pour des valeurs élevées de λ alors qu'elle devrait être croissante). Nous nous intéressons donc à d'autres techniques d'optimisation pour résoudre ce problème. De plus, nous souhaitons prendre en compte des énergies comportant plusieurs paramètres de régularisation. Plusieurs approches sont considérées : approximation de la fonction de validation croisée, factorisation de la matrice A_λ, analyse par intervalles [4].

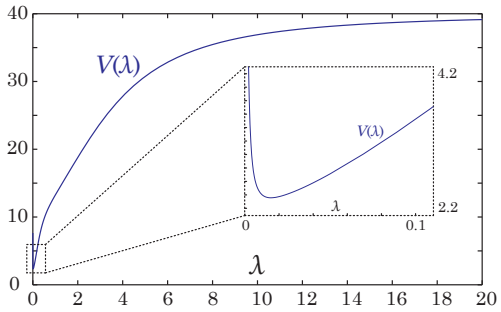


FIG. 1. Fonction de validation croisée.

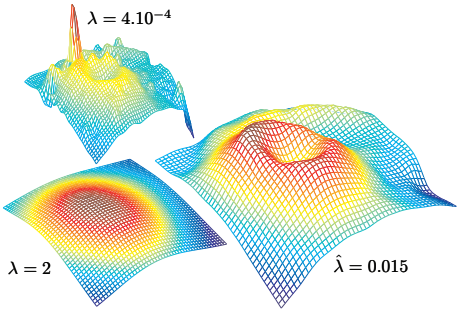


FIG. 2. Ajustement de surface sur le puy Pariou (Auvergne).

Références

1. F. L. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE Trans. Pattern Anal. Mach. Intell.*, 11(6):567–585, 1989.
2. G. Donato and S. Belongie. Approximate Thin-Plate Spline mappings. In *ECCV (3)*, pages 21–31, 2002.
3. M. Fornefett, K. Rohr, and H. S. Stiel. Radial Basis Functions with compact support for elastic registration of medical images. *Image and vision computing*, 19:87–96, 2001.
4. L. Jaulin, M. Kieffer, O. Didrit, and É. Walter. *Applied Interval Analysis*. Springer, 2001.
5. R. Koenker and I. Mizera. Penalized triograms: Total variation regularization for bivariate smoothing. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 66:145, februray 2004.
6. T. Tarpey. A note on the prediction sum of squares statistic for restricted least squares. *The American Statistician*, 54(2):116–118, May 2000.
7. G. Wahba. *Spline Models for Observational Data*, pages 10–14. SIAM, 1990.
8. G. Wahba and S. Wold. A completely automatic French curve: Fitting spline functions by cross validation. *Communications in Statistics*, 4:1–18, 1975.