Available online at www.sciencedirect.com
ScienceDirect
Computer Vision and Image Understanding 109 (2008) 244–259
Computer Vision and Image Understanding
www.elsevier.com/locate/cviu

Global parametric image alignment via high-order approximation

Y. Keller [a,*], A. Averbuch [b]

[a] Electrical & Computer Engineering Department, Ben-Gurion University of the Negev, Israel
[b] School of Computer Science, Tel Aviv University, Aviv 69978, Israel

Received 11 February 2006; accepted 8 May 2007
Available online 29 May 2007

Abstract

The estimation of parametric global motion is one of the cornerstones of computer vision. Such schemes are able to estimate various motion models (translation, rotation, affine, projective) with subpixel accuracy. The parametric motion is computed using a first order Taylor expansions of the registered images. But, it is limited to the estimation of small motions, and while large translations and rotations can be coarsely estimated by Fourier domain algorithms, no such techniques exist for affine and projective motions. This paper offers two contributions: first, we improve both the convergence range and rate using a second order Taylor expansion and show first order methods to be a degenerate case of the proposed scheme. Second, we extend the scheme using a symmetrical formulation which further improves the convergence properties. The results are verified by rigorous analysis and experimental trials.

Keywords: Motion estimation; Non-linear optimization; Large motions

1. Introduction

Image registration plays a vital role in many image processing and computer vision applications such as optical flow computation [1,2], tracking [3–5], video compression [6,7], layered motion estimation [8], Nonparametric motion recognition [9], and 3D reconstruction [10] to name a few. A comprehensive comparative survey by Barron et. al. [11] found the family of gradient-based motion estimation methods (GM), originally proposed by Horn and Schunck [12] and Lucas and Kanade [1], to perform especially well. The aim of the GM scheme is to estimate the parameters vector **p** associated with the *parametric image registration* problem: starting from pure global translation, rotation, affine, and projective motions. These models have been used extensively and are directly computed using image spatio-temporal derivatives and coarse-to-fine estimation. They search for the parametric geometric transform that best minimizes the square of differences between image intensities over the whole image. Several formulations of the gradient methods were suggested. They differ in the way the motion parameters are updated [13]: either by incrementing the motion parameters [1] or the warp matrix [14]. An updated comprehensive description of these methods was given in [15].

The registration is computed by relating a pair of images having some overlap using a first order Taylor series expansion. Each pixel in the common support contributes a linear constraint, denoted the *constant brightness constraint*. Thus, an over-constrained linear system is formulated yielding a robust estimate. Gathering and solving all the equations associated with pixels in the common support, estimates the *global motion* between the images [14].

Local motions denoted Optical flow, are estimated by computing a vector field $\{\mathbf{p}_{i,j}\}$ of motion parameters, where each motion vector $\mathbf{p}_{i,j}$ corresponds to the motion of a small image patch. Such schemes enforce a smoothness constraint on the computed motion field, and the variational framework was shown to be the state-of-the-art. Variational schemes formulate the Optical flow computation as a variational minimization problem, which is solved via the solution of partial differential equations. Such

* Corresponding author.
E-mail address: yosi.keller@gmail.com (Y. Keller).

1077-3142/$ - see front matter © 2007 Elsevier Inc. All rights reserved.
doi:10.1016/j.cviu.2007.05.003

techniques were applied in [16] to the computation of piecewise parametric motion, where each image patch is in the order of 50–100 pixels. Bronx et al. [2] applied a second order variational approach to improve the accuracy and robustness.

The smoothness constraint enforced by the variational schemes is inappropriate when the estimated motion field is discontinuous, such as the one resulting form the motion of multiple objects and their occlusions. A possible solution is to apply robust statistical measures. Odobez and Bouthemy [17] implemented such schemes using M-Estimators for correlation and differential based optic flow computation. Instead of minimizing the L_2 norm of the intensity discrepancies between the registered images, they minimize the M-Estimator functional applied to the discrepancies. Sim and Park [18] applied both the least median squares (LMS) the reweighed least squares (RLS) to the discrepancies and the estimated motion field. Thus, they were able to analyze noisy image sequences containing several (spatially) discontinuous motions.

The estimation of parametric global motion is the focal point of this work and has gained significant attention in the computer vision community. As such schemes estimate a relatively small number of parameters (usually up to 8) based on a least squares formulation of the entire images, such schemes are robust and can be applied to a wide range of applications related to image stitching [14] and mosaicing [19]. This robustness was further used to estimate higher order parametric models that are able to register quadratic surfaces [20] (12 parameters), estimate global non-linear illumination changes [21] and barrel lenses distortions [22]. A limited class of parametric motions was also applied to video compression within the MPEG4 video compression standard [7,23]. The focal point in such works is to derive computationally efficient schemes, where due to the high frame-rate used (15–25 frames per second) the estimated motion is assumed to be small.

A critical implementation issue concerning the GM is their convergence when estimating large motions. As the estimated motion grows, the convergence rate decreases and the GM may converge to a local minima. Hence, GM algorithms are unable to estimate large motions and have to be bootstrapped [24,25], for instance, by the coarser and more robust Fourier schemes [26,27]. But, while the estimation of translations and rotations can be bootstrapped by Fourier based methods, there are no such reliable solutions for affine and projective motions. For instance, [28,29] use affine invariant texture descriptors to bootstrap wide basis stereo.

We propose to improve the convergence properties of the GM algorithm using a second order Taylor expansion of the registered images. We show that the convergence properties of the proposed algorithm are superior to those of the regular GM for large and small motions. Convergence rates of 3 and $\frac{3}{2}$ are achieved for small and large motions, respectively, compared to 2 and 1 for the regular GM. The improvement is rigorously proven in two ways, first we analyze the convergence properties of non-linear optimization schemes, proving that higher order schemes provide better convergence properties, and apply the results to image registration. Second, we show that the GM is a degenerate, sub-optimal case of the proposed scheme.

Further improvement is achieved by combining the second order expansion with a symmetrical formulation [30]. While no gradient based scheme (including ours) can guarantee global convergence, we show rigorously and experimentally that the proposed scheme achieves better convergence ranges than the standard GM.

The paper is organized as follows: the optimization based GM approach to image registration is presented in Section 2. We then introduce the third order (O3GM) and symmetric third order (O3GM) schemes in Section 3. The convergence properties of these schemes are rigorously analyzed in Section 5, while experimental validation is given in Section 6. Concluding remarks and future work are discussed in Section 7.

2. Gradient methods based motion estimation

GM methodology [15] estimates the motion parameters $\mathbf{p}$ by minimizing the intensity discrepancies between I_1 and I_2

$$\mathbf{p}^* = \arg\min_{\mathbf{p}} \sum_{(x_i^1,y_i^1)\in S} \left(I_1(x_i^1,y_i^1) - I_2(x_i^2,y_i^2)\right)^2, \tag{2.1}$$

where

$$\begin{aligned} x_i^2 &= f(x_i^1,y_i^1,\mathbf{p}), \\ y_i^2 &= g(x_i^1,y_i^1,\mathbf{p}), \end{aligned} \tag{2.2}$$

S is the set of coordinates of pixels common to I_1 and I_2 in I_1's coordinates, $\mathbf{p}$ is the estimated parameters vector and f and g represent the motion model. For instance, affine motion is given by

$$\begin{aligned} x_i^2 &= p_1x_i^1 + p_2y_i^1 + p_3, \\ y_i^2 &= p_4x_i^1 + p_5y_i^1 + p_6, \end{aligned} \tag{2.3}$$

and the projective model is given by

$$\begin{aligned} x_i^2 &= \frac{p_1x_i^1 + p_2y_i^1 + p_3}{p_7x_i^1 + p_8y_i^1 + 1}, \\ y_i^2 &= \frac{p_4x_i^1 + p_5y_i^1 + p_6}{p_7x_i^1 + p_8y_i^1 + 1}. \end{aligned} \tag{2.4}$$

In practice, solving Eq. (2.1) does not result in perfect intensity alignment due to relative intensity changes, and non-corresponding pixels within the registered images. Next we follow the formulation of [14,31] and solve Eq. (2.1). The basic GM formulation and its iterative refinement step are described in Sections 2.1 and 2.2, respectively. These are embedded in the multi-resolution, coarse-to-fine scheme, which improves the convergence properties.

2.1. Basic GM formulation

Eq. (2.1) is solved via a linearization scheme, based on a pixel-wise first order Taylor expansion of I_1 in terms of I_2 as a function of the parameter vector $\mathbf{p}$ around $\mathbf{p}=0$

$$I_1\left(x_i^1,y_i^1\right)=I_2\left(x_i^1,y_i^1\right)+\sum_{k=1}^{N_P}\frac{\partial I_2\left(x_i^1,y_i^1\right)}{\partial p_k}\mathbf{p}_k+\frac{1}{2}\sum_{k=1,m=1}^{N_P}\frac{\partial I_2\left(x_i^1,y_i^1,\widetilde{\mathbf{p}}\right)}{\partial p_k\partial p_m}\mathbf{p}_k\mathbf{p}_m,\quad \widetilde{\mathbf{p}}\in[\mathbf{0},\mathbf{p}]. \tag{2.5}$$

$I_1(x_i^1,y_i^1)$ and $I_2(x_i^1,y_i^1)$ are the ith corresponding pixel in I_1 and I_2. $\frac{\partial I_2(x_i^1,y_i^1)}{\partial p_k}$ is the partial derivative with respect to the motion parameters, whose computation for the affine motion model is described in Appendix B. The second term in the r.h.s of Eq. (2.5) is the Lagrange Remainder [32] of the first order Taylor approximation. As $\widetilde{\mathbf{p}}$ is an element of a vectorial space, $[\mathbf{0},\mathbf{p}]$ is a segment and not an interval, where $\mathbf{0}$ is the zero vector.

As $\widetilde{\mathbf{p}}$ is unknown, Eq. (2.5) cannot be solved for $\mathbf{p}$. Instead, we neglect the Lagrange Remainder term and solve for $\delta\mathbf{p}$

$$I_1(x_i^1,y_i^1)=I_2(x_i^1,y_i^1)+\sum_{k=1}^{N_P}\frac{\partial I_2(x_i^1,y_i^1)}{\partial p_k}\delta\mathbf{p}_k. \tag{2.6}$$

By gathering the pixel-wise equations we get the equation set

$$\mathbf{H}\,\delta\mathbf{p}=\mathbf{I_t}, \tag{2.7}$$

where $\mathbf{H}_{i,j}=\frac{\partial I_2(x_i^1,y_i^1)}{\partial p_j}$ and $\underline{I_t}_i=I_1(x_i^1,y_i^1)-I_2(x_i^1,y_i^1)$. Eq. (2.7) is solved using least squares

$$\delta\mathbf{p}=(\mathbf{H}^{\mathrm{T}}\mathbf{H})^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{I_t}. \tag{2.8}$$

2.2. Iterative solution of gradient methods

In general, due to the omission of the error term in Eq. (2.5) $\delta\mathbf{p}\neq\mathbf{p}$ and Eq. (2.1) is solved iteratively. At each iteration Eq. (2.6) is formulated based on the current estimate of the motion parameters $\mathbf{p_n}$.

Denote:

$\mathbf{p_0}$ an initial estimated solution of Eq. (2.1) given as input
$\mathbf{p_n}$ the estimated solution after n iterations

The nth iteration of the iterative GM is given in Algorithm 1.

Algorithm 1.

The nth iteration of the iterative gradient methods

1: The input image I_2 is warped towards I_1 using the current estimate $\mathbf{p_n}$ $n\geqslant 0$ and it is stored in $\widetilde{I}_2$. $\mathbf{p_0}$ is a given input.
2: I_1 and $\widetilde{I}_2$ are used as input images to the procedure described in Section 2.1.
3: $\delta\mathbf{p}$, the result of step 2 is used to update the solution

$$\mathbf{p_{n+1}}=\delta\mathbf{p}+\mathbf{p_n}\quad n\geqslant 0. \tag{2.9}$$

4: The iterations are performed until at most $N_{\max}$ iterations are completed or the translation parameters (within $\delta\mathbf{p}$) are smaller than a predetermined threshold.

In order to improve the convergence properties of the GM, the iterative process is embedded in a coarse-to-fine multiscale formulation [8,15].

3. Third order gradient methods

This section introduces the third order GM formulation (O3GM) which is integrated into the regular GM scheme presented in Section 2. The O3GM replaces *only* the basic GM step given in Section 2.1. The iterative refinement (Section 2.2) and multiscale steps are left intact. We rigorously show in Section 6 that for non-linear optimization problems, such as the GM, the higher the order of approximation, the better the convergence rate and range. In particular, the registration of images related by large motions, results in large parameters deviations. Thus, the approximation error grows, lowering the convergence rate down to the point of divergence.

The focal point of the proposed approach is to improve the estimation of large motions, which might otherwise be impossible, by lowering the approximation error, and reduce the number of iterations needed for convergence.

While one can use a higher order approximation than the second order one used by our approach, such schemes will result in an increased computational complexity that might prove exhaustive.

The second order approximation results in a set of quadratic equations solved iteratively, whose solution yields the refinement term δp. Thus, the proposed algorithm applies two iterative cycles: the first is identical to Algorithm 1, while the second solves the quadratic equation set for δp. We show in Section 3.1 that the GM corresponds to a degenerate and sub-optimal instance of the proposed O3GM.

Similar to the basic GM step (Section 2.1), the O3GM accepts as input two images, I_1 and I_2, that are aligned as best as possible. It outputs δp, that is an estimate of the motion between I_1 and I_2. The accumulation of the iterative refinements is handles by the iterative refinement (Section 2.2), the same way as in the regular GM scheme.

Eq. (2.1) is solved by expanding it in a second order Taylor series expansion

$$I_1(x_i^1,y_i^1)=I_2(x_i^1,y_i^1)+\sum_{j=1}^{N_p}\delta p_j\frac{\partial I_2(x_i^1,y_i^1)}{\partial p_j}+\frac{1}{2}\sum_{j,s=1}^{N_p}\delta p_j\delta p_s\frac{\partial^2 I_2(x_i^1,y_i^1)}{\partial p_j\partial p_s}, \tag{3.1}$$

and solving for δp. δp is a refinement of the current estimate of the motion. Thus, δp is substituted into Eq. (2.9). N_p is the number of motion parameters. For instance, for the translation motion model $\delta\mathbf{p}=(\delta x,\delta y)^{\mathrm{T}}$, $N_p=2$, and we get

$$\sum_{j,s=1}^{2}\delta p_j\delta p_s\frac{\partial^2 I_2(x_i^1,y_i^1)}{\partial p_j\partial p_s}=\frac{\partial^2 I_2(x_i^1,y_i^1)}{\partial x^2}\delta x^2+2\frac{\partial^2 I_2(x_i^1,y_i^1)}{\partial y\partial x}\delta y\delta x+\frac{\partial^2 I_2(x_i^1,y_i^1)}{\partial y^2}\delta y^2. \tag{3.2}$$

The first and second partial derivatives with respect to the motion parameters are computed using the chain rule and their computation for the affine motion model is described in Appendix B. Eq. (3.1) is evaluated at each pixel common to the images I_1 and I_2, forming the quadratic equation set $\{r_i\}_{i=1,\dots,N}$ where N is the size of the common area. Thus, r_i is the ith equation computed by substituting $I_1(x_i^1,y_i^1)$ and $I_2(x_i^1,y_i^1)$ into Eq. (3.1).

$$r_1(\delta p)=\sum_{j=1}^{N_p}\delta p_j\frac{\partial I_2(x_1^1,y_1^1)}{\partial p_j}+\tfrac{1}{2}\sum_{j,s=1}^{N_p}\delta p_j\delta p_s\times\frac{\partial^2 I_2(x_1^1,y_1^1)}{\partial p_j\partial p_s}-I_t(x_1^1,y_1^1), \tag{3.3}$$

$$\vdots$$

$$r_N(\delta p)=\sum_{j=1}^{N_p}\delta p_j\frac{\partial I_2(x_N^1,y_N^1)}{\partial p_j}+\tfrac{1}{2}\sum_{j,s=1}^{N_p}\delta p_j\delta p_s\times\frac{\partial^2 I_2(x_N^1,y_N^1)}{\partial p_j\partial p_s}-I_t(x_N^1,y_N^1), \tag{3.4}$$

where

$$I_t(x_i^1,y_i^1)=I_1(x_i^1,y_i^1)-I_2(x_i^1,y_i^1).$$

Appendix A details the derivation of the equation set for translation estimation.

In first order GM schemes (Section 2.1), the image I_2 is approximated by a linear model, and the set $\{r_i\}_{i=1,\dots,N}$ is a set of linear equations. Thus, the refinement term of the motion parameters, δp, is computed by *linear least squares* [Eq. (2.8)]. In contrast, the O3GM derives a quadratic set of [Eq. (3.3)] and the refinement of the motion parameters, δp is computed by *non-linear least squares*

$$\delta p=\arg\max_{\delta p}\sum_i r_i(\delta p)^2. \tag{3.5}$$

Eq. (3.5) is solved by iterative Newtonian methods [33,34]. We applied the Gauss–Newton and Newton's schemes that differ on the formulation of the Newtonian iterations. A single iteration of the Gauss–Newton algorithm is given by

$$\delta p^{(k+1)}=\delta p^{(k)}-(\mathbf{J}^{\mathrm{T}}\mathbf{J})^{-1}\mathbf{J}^{\mathrm{T}}\mathbf{r},\quad k=0,\dots, \tag{3.6}$$

while the Newton's schemes iteration is given by

$$\delta p^{(k+1)}=\delta p^{(k)}-(\mathbf{J}^{\mathrm{T}}\mathbf{J}+\sum_i^N\mathbf{H}_i r_i(\delta p^{(k)}))^{-1}\mathbf{J}^{\mathrm{T}}\mathbf{r},\quad k=0,\dots, \tag{3.7}$$

$r=(r_1,\dots,r_N)^{\mathrm{T}}$, $\mathbf{J}$ is the Jacobian, $\mathbf{H}_i$ is the Hessian of an equation r_i and $\delta p^{(k)}$ is the solution at the Newtonian iteration k, where $\delta p^{(0)}=0$.

The Jacobian J of $\{r_i\}_{i=1,\dots,N}$ is given by

$$J=\begin{bmatrix}\frac{\partial I_2(x_1^1,y_1^1)}{\partial p_1}+\sum_{s=1}^{N_p}\delta p_s\frac{\partial^2 I_2(x_1^1,y_1^1)}{\partial p_1\partial p_s} & \cdots & \frac{\partial I_2(x_1^1,y_1^1)}{\partial p_{N_p}}+\sum_{s=1}^{N_p}\delta p_s\frac{\partial^2 I_2(x_1^1,y_1^1)}{\partial p_{N_p}\partial p_s}\\ \vdots & & \vdots\\ \frac{\partial I_2(x_N^1,y_N^1)}{\partial p_1}+\sum_{s=1}^{N_p}\delta p_s\frac{\partial^2 I_2(x_N^1,y_N^1)}{\partial p_j\partial p_s} & \cdots & \frac{\partial I_2(x_N^1,y_N^1)}{\partial p_{N_p}}+\sum_{s=1}^{N_p}\delta p_s\frac{\partial^2 I_2(x_N^1,y_N^1)}{\partial p_{N_p}\partial p_s}\end{bmatrix}. \tag{3.8}$$

The first and second order partial derivatives of I_2 with respect to the motion parameters $\mathbf{p}$, are computed using the chain rule. Appendix B details their computation for the affine motion model. The Hessian $\mathbf{H}_i$ is the Hessian matrix of the single equation r_i [see Eq. (3.3)]

$$H_i=\begin{bmatrix}\frac{\partial^2 I_2(x_1^1,y_1^1)}{\partial p_1^2} & \frac{\partial^2 I_2(x_1^1,y_1^1)}{\partial p_1\partial p_2} & \cdots & \frac{\partial^2 I_2(x_1^1,y_1^1)}{\partial p_1\partial p_{N_p}}\\ \frac{\partial^2 I_2(x_2^1,y_2^1)}{\partial p_2\partial p_1} & \frac{\partial^2 I_2(x_2^1,y_2^1)}{\partial p_2^2} & & \frac{\partial^2 I_2(x_2^1,y_2^1)}{\partial p_2\partial p_{N_p}}\\ \vdots & & \ddots & \vdots\\ \frac{\partial^2 I_2(x_N^1,y_N^1)}{\partial p_{N_p}\partial p_1} & \frac{\partial^2 I_2(x_N^1,y_N^1)}{\partial p_{N_p}\partial p_2} & \cdots & \frac{\partial^2 I_2(x_N^1,y_N^1)}{\partial p_{N_p}^2}\end{bmatrix}. \tag{3.9}$$

Since the quadratic equations set is given explicitly, the computation of $\mathbf{J}$ and $\mathbf{H}_i$ is fast and accurate up to machine precision. The scheme is summarized in Algorithm 2.

Algorithm 2.

Third order gradient methods

1: Compute the first and second partial derivatives of I_2 with respect to the motion parameters.
2: Form the second order polynomial equation set $\{r_i\}_{i=1,\dots,N}$.
3: Compute the Hessian $\{H_i\}_{i=1,\dots,N}$ of $\{r_i\}_{i=1,\dots,N}$ if Newton's algorithm is to be used.
4: Set $\delta p_0=0$.
5: **for** $n=0$ to $N_{\max}$ **do**
6: Compute the Jacobian $\mathbf{J}$ and evaluate $r_i(\delta p_n)$.
7: Either Eq. (3.6) or Eq. (3.7) are applied according to the optimization scheme used.
8: Stop the iterations if the translation parameters reach a predetermined threshold.
9: **end for**
10: Return δp_n.

3.1. GM as a degenerate case of the O3GM

In this section we prove that the GM is a degenerate and sub-optimal instance of the O3GM. It is equivalent to applying the O3GM using a single Gauss–Newton iteration ($N_{\max}=1$ in Algorithm 2) with a zero initial motion estimate.

We start by considering the GM formulation in Eq. (2.6), whose Jacobian is given by

$$J_{\mathrm{GM}}=\begin{bmatrix}\frac{\partial I_2(x_1^1,y_1^1)}{\partial p_1} & \cdots & \frac{\partial I_2(x_1^1,y_1^1)}{\partial p_{N_p}}\\ \vdots & & \vdots\\ \frac{\partial I_2(x_N^1,y_N^1)}{\partial p_1} & \cdots & \frac{\partial I_2(x_N^1,y_N^1)}{\partial p_{N_p}}\end{bmatrix} \tag{3.10}$$

Comparing Eq. (3.10) to J_{O3GM}, the Jacobian of the O3GM [Eq. (3.8)], we note that by setting $\delta p = 0$ we get $J_{\mathrm{GM}} = J_{\mathrm{O3GM}}$.

Thus, applying Eq. (3.6) once to solve the quadratic set with an initial estimate $\delta p = 0$, yields the same result as solving Eq. (2.8) within the GM scheme. As one can apply more iterations and use Newton's scheme [Eq. (3.7)] to solve the quadratic equation, the GM is a degenerate and sub-optimal case of the O3GM scheme.

4. Symmetric third order GM formulations

The symmetric GM (SGM) formulation was introduced in [30] to improve the convergence properties of GM schemes. It utilizes the symmetry of the image registration problem with respect to the motion parameters $\mathbf{p}$, to reduce the approximation error. The error reductions achieved by the O3GM and SGM are complementary, thus, we integrate both in a unified framework we call the Symmetric third order GM (SO3GM). The image registration problem is symmetrically formulated using a parametric motion model defined by $\mathbf{p}$

$$I_2\left(x_i^1, y_i^1, \frac{\mathbf{p}}{2}\right) = I_1\left(x_i^1, y_i^1, -\frac{\mathbf{p}}{2}\right), \tag{4.1}$$

Both sides of Eq. (4.1) are expanded using a second order Taylor expansion and neglecting the third order error term

$$\begin{aligned} I_1(x_i^1, y_i^1) - \sum_{j=1}^{N_p} \frac{\varepsilon_j}{2} \frac{\partial I_1(x_i^1, y_i^1)}{\partial p_j} + \frac{1}{2} \sum_{j,s=1}^{N_p} \frac{\varepsilon_j}{2} \frac{\varepsilon_s}{2} \frac{\partial^2 I_1(x_i^1, y_i^1)}{\partial p_j \partial p_s} \\ = I_2(x_i^1, y_i^1) + \sum_{j=1}^{N_p} \frac{\varepsilon_j}{2} \frac{\partial I_2(x_i^1, y_i^1)}{\partial p_j} + \frac{1}{2} \sum_{j,s=1}^{N_p} \frac{\varepsilon_j}{2} \frac{\varepsilon_s}{2} \frac{\partial^2 I_2(x_i^1, y_i^1)}{\partial p_j \partial p_s}, \end{aligned} \tag{4.2}$$

where N_p is the number of motion parameters and ε is the vector of motion parameters to be estimated.

The focal point of this formulation is that given the images I_1 and I_2 related by an unknown motion $\mathbf{p}$, both sides of Eq. (4.2) approximate an image corresponding to the middle point (in the parameters space), which is $\delta_1 = \pm\frac{\mathbf{p}}{2}$ apart from both images. In contrast, the O3GM scheme used a single approximation over the interval $\delta_2 = \mathbf{p}$. Recalling that the approximation error is related to $\|\delta_i\|^3$, we get that both sides of Eq. (4.2) are associated with an error of $\left(\frac{\|\mathbf{p}\|}{2}\right)^3 = \frac{\|\mathbf{p}\|^3}{8}$ and the overall error is bounded by $\frac{\|\mathbf{p}\|^3}{4}$ compared to $\|\mathbf{p}\|^3$ for the O3GM. Similar results were derived for the GM scheme in [30].

By reformulating Eq. (4.2) we get

$$\begin{aligned} I_1(x_i^1, y_i^1) = I_2(x_i^1, y_i^1) + \frac{1}{2} \sum_{j=1}^{N_p} \delta p_j \left(\frac{\partial I_2(x_i^1, y_i^1)}{\partial p_j} + \frac{\partial I_1(x_i^1, y_i^1)}{\partial p_j} \right) \\ + \frac{1}{8} \sum_{j,s=1}^{N_p} \delta p_j \delta p_s \left(\frac{\partial^2 I_2(x_i^1, y_i^1)}{\partial p_j \partial p_s} - \frac{\partial^2 I_1(x_i^1, y_i^1)}{\partial p_j \partial p_s} \right). \end{aligned} \tag{4.3}$$

By constructing the above equation for all of the pixels in the common support between I_1 and I_2, we derive a quadratic equation set that is solved by Algorithm 2 for $\delta\mathbf{p}$.

5. Third order convergence analysis

In this section we analyze the convergence properties of the non-linear least squares scheme used in GM based image registration. We formulate the optimization as a zero crossing problem solved by Taylor series based approximations, and derive the corresponding convergence rate and range.

Definition 5.1 (*Least squares minimization*)**.** A vector function $f(\mathbf{p}) : R^n \to R^m$ is minimized in the least squares sense with respect to the vector P if

$$\mathbf{p}^* = \arg\min_{\mathbf{P}} f(\mathbf{p}) = \arg\min_{\mathbf{P}} \sum_{i=1}^{m} (r_i(\mathbf{p}))^2. \tag{5.1}$$

If $r_i(\mathbf{p})$ is non-linear with respect to p, Eq. (5.1) is denoted the non-linear minimization of f with respect to p.

Definition 5.2 (*Iterative solution*)**.** An iterative solution of an optimization problem is given by

$$\mathbf{p}_{n+1} = \mathbf{p}_n - \delta\mathbf{p}_n \tag{5.2}$$

where $p_{\mathbf{n}}$ and δp_n are the vector of parameters and the update term, respectively, computed at iteration n while

$$\varepsilon_n = \mathbf{p}^* - \mathbf{p}_n \tag{5.3}$$

is the parameters' error at iteration n.

Definition 5.3 (*Range of convergence*)**.** For an iterative optimization scheme, the range of convergence is given by a scalar $\varepsilon_{\max} > 0$, such that for any vector ε such that $\varepsilon_{\max} > \|\varepsilon\|$, applying the iterative optimization scheme decreases the value of the cost function given in Eq. (5.1), $|\varepsilon_n\| > \|\varepsilon_{n+1}\|$ and the scheme converges to its global minimum.

Definition 5.4 (*Small and Larger deviation*)**.** For an iterative minimization problem, a large deviation $\mathbf{p}$ is characterized by $\|f(\mathbf{p})\| \gg \|f(\mathbf{p}^*)\|$. A small deviation $\mathbf{p}$ is in the vicinity of the solution $\mathbf{p}^*$, and we have $\|f(\mathbf{p})\| = \mathrm{O}(\|f(\mathbf{p}^*)\|)$.

Eq. (5.1) is solved iteratively by approximating $f(\mathbf{p})$ at the zero crossing point p^* using a second order Taylor series expansion around $\mathbf{p}_n$

$$\begin{aligned} f(\mathbf{p}^*) = f(\mathbf{p}_n) + \sum_{k=\mathbf{1}}^{N_P} \frac{\partial f(\mathbf{p}_n)}{\partial p_k} (\varepsilon_{\mathbf{n}})_k \\ + \frac{1}{2} \sum_{k_1=1,k_2=1}^{N_P} \frac{\partial^2 f(\mathbf{p}_n)}{\partial p_{k_1} \partial p_{k_2}} (\varepsilon_{\mathbf{n}})_{k_1} (\varepsilon_{\mathbf{n}})_{k_2} + R_2(\mathbf{p}_n, \widetilde{\mathbf{p}}, \varepsilon_{\mathbf{n}}). \end{aligned} \tag{5.4}$$

where $\varepsilon_{\mathbf{n}}$ is the estimation error in iteration n, N_P is the dimension of $\mathbf{p}$ and $R_2(\mathbf{p}_k, \widetilde{\mathbf{p}})$ is the *Lagrange Remainder* [32] of the Taylor series approximation given by

$$R_2(\mathbf{p}_n, \widetilde{\mathbf{p}}, \varepsilon_{\mathbf{n}}) = \frac{1}{6} \sum_{k_1=1,k_2=1,k_3=1}^{N_P} \frac{\partial^3 f(\widetilde{\mathbf{p}})}{\partial p_{k_1} \partial p_{k_2} \partial p_{k_3}} (\varepsilon_{\mathbf{n}})_{k_1} (\varepsilon_{\mathbf{n}})_{k2} (\varepsilon_{\mathbf{n}})_{k_3}, \quad \widetilde{\mathbf{p}} \in [0, \varepsilon_{\mathbf{n}}]. \tag{5.5}$$

As both $f(\mathbf{p}^*)$ and R_2 are unknown ($\mathbf{p}^*$ is a minimum point, but one cannot assume that $f(\mathbf{p}^*) = 0$), Eq. (5.4) cannot be solved directly. Hence we neglect R_2 and $f(\mathbf{p}^*)$ and solve

$$0 = H(\delta \mathbf{p}_n) \triangleq f(\mathbf{p}_n) + \sum_{k=\mathbf{1}}^{N_P} \frac{\partial f(\mathbf{p}_n)}{\partial p_k} (\delta \mathbf{p}_n)_k + \frac{1}{2} \sum_{k_1=\mathbf{1},k_2=\mathbf{1}}^{N_P} \frac{\partial^2 f(\mathbf{p}_n)}{\partial p_{k_1} \partial p_{k_2}} (\delta \mathbf{p}_n)_{k_1} (\delta \mathbf{p}_n)_{k_2}, \tag{5.6}$$

in the least squares sense. δp_n is the refinement term computed by the O3GM.

Recalling that $\mathbf{p}_{n+1} = \mathbf{p}_n - \delta \mathbf{p}_n$, the right-hand-side of Eq. (5.6) is a truncated second order Taylor approximation of $f(\mathbf{p}_{n+1})$ over the interval $[\mathbf{p}_n, \mathbf{p}_{n+1}]$. Thus,

$$f(\mathbf{p}_{n+1}) = H(\delta \mathbf{p}_n) + R_2(\mathbf{p}_n, \widetilde{\mathbf{p}}, \delta \mathbf{p}_n), \quad \widetilde{\mathbf{p}} \in [\mathbf{0}, \delta \mathbf{p}_n], \tag{5.7}$$

and as $H(\delta p_n) = 0$ [Eq. (5.6)], we get

$$\|f(\mathbf{p}_{n+1})\| = C_3 \|\delta \mathbf{p}_n\|^3, \tag{5.8}$$

where

$$C_3 = \frac{1}{6} \left\| \sum_{k_1=1,\ldots,k_{T+1}=1}^{N_P} \frac{\partial^3 f(\widetilde{\mathbf{p}})}{\partial p_{k_1} \partial p_{k_2} \partial p_{k_3}} \right\| = \mathrm{O}\left(\left\| \frac{\partial^3 f}{\partial p^3} \right\| \right), \quad \widetilde{\mathbf{p}} \in [\mathbf{p}_n, \mathbf{p}_{n+1}].$$

We continue to study the convergence properties for large and small deviations.

Lemma 5.1. *The convergence rate of the second order approximation scheme for large deviations is given by* $\|\varepsilon_{\mathbf{n+1}}\| = \sqrt{\frac{C_3}{C_2}} \|\varepsilon_{\mathbf{n}}\|^{3/2}$, *where* C_2 *and* C_3 *are constants and large deviations are characterized by* $\|f(\mathbf{p})\| \gg \|f(\mathbf{p}^*)\|$ *and* $\|\varepsilon_{\mathbf{n}}\| \gg 0$

Proof. For large deviations $\|f(\mathbf{p})\| \gg \|f(\mathbf{p}^*)\|$, hence we can neglect the term $\|f(\mathbf{p}^*)\|$ in Eq. (5.4). Moreover, since $\|\varepsilon_{\mathbf{n}}\| \gg 0$

$$\left\| \sum_{k_1=1,k_2=1}^{N_P} \frac{\partial^2 f(\widetilde{\mathbf{p}})}{\partial p_{k_1} \partial p_{k_2}} (\varepsilon_{\mathbf{n}})_{k_1} (\varepsilon_{\mathbf{n}})_{k_2} \right\| \gg \left\| \sum_{k=\mathbf{1}}^{N_P} \frac{\partial f(\mathbf{p}_n)}{\partial p_k} (\varepsilon_{\mathbf{n}})_k \right\|, \tag{5.9}$$

and thus, Eq. (5.4) is reduced to

$$|f(\mathbf{p}_n)| = \left\| \frac{1}{2} \sum_{k_1=1,k_2=1}^{N_P} \frac{\partial^2 f(\widetilde{\mathbf{p}})}{\partial p_{k_1} \partial p_{k_2}} (\varepsilon_{\mathbf{n}})_{k_1} (\varepsilon_{\mathbf{n}})_{k_2} \right\| = C_2 \|\varepsilon_{\mathbf{n}}\|^2, \quad \widetilde{\mathbf{p}} \in [\mathbf{0}, \varepsilon_n], \tag{5.10}$$

where

$$C_2 = \left\| \frac{1}{2} \sum_{k_1=1,k_2=1}^{N_P} \frac{\partial^2 f(\widetilde{\mathbf{p}})}{\partial p_{k_1} \partial p_{k_2}} \right\| = \mathrm{O}\left(\left\| \frac{\partial^2 f}{\partial p^2} \right\| \right). \tag{5.11}$$

Substituting Eqs. (5.8) and (5.10)

$$\|\delta \mathbf{p}_n\| = \sqrt[3]{\frac{|f(\mathbf{p}_{n+1})|}{C_3}} = \sqrt[3]{\frac{C_2 \|\varepsilon_{\mathbf{n+1}}\|^2}{C_3}} = \sqrt[3]{\frac{C_2}{C_3}} \|\varepsilon_{\mathbf{n+1}}\|^{\frac{2}{3}}. \tag{5.12}$$

By Eq. (5.3), $\varepsilon_{\mathbf{n}+1} = \varepsilon_{\mathbf{n}} - \delta p_n$, and as we are away from the solution $\|\varepsilon_{\mathbf{n}+1}\| \gg 0$, thus

$$\sqrt[3]{\frac{C_2}{C_3}} \|\varepsilon_{\mathbf{n+1}}\|^{\frac{2}{3}} = \|\delta \mathbf{p}_n\| \leqslant \|\varepsilon_{\mathbf{n}}\|,$$

and

$$\|\varepsilon_{\mathbf{n+1}}\| \leqslant \sqrt{\frac{C_3}{C_2}} \|\varepsilon_{\mathbf{n}}\|^{\frac{3}{2}}. \tag{5.13}$$

The ratio $\frac{C_3}{C_2}$ is given by

$$\frac{C_3}{C_2} = \mathrm{O}\left(\left\| \frac{\partial^3 f}{\partial p^3} \right\| \Big/ \left\| \frac{\partial^2 f}{\partial p^2} \right\| \right). \tag{5.14}$$

The smaller the ratio $\frac{C_3}{C_2}$, the faster the convergence. $C_3 = \|\frac{\partial^3 f}{\partial p^3}\|$ is related to the approximation error of the second order Taylor expansion. Thus, the more accurate the approximation the faster the convergence.

Lemma 5.2. *The convergence rate of the second order approximation based scheme for small deviation is given by* $\|\varepsilon_{\mathbf{n+1}}\| \leqslant \frac{C_3}{C1} \|\varepsilon_{\mathbf{n}}\|^3$, *where* C_1 *and* C_3 *are constants.*

Proof. For small deviations $\|f(\mathbf{p})\| \to \|f(\mathbf{p}^*)\|$ and $\|\varepsilon_{\mathbf{n}}\| \to 0$, thus

$$\|\varepsilon_{\mathbf{n}}\| \gg \|\varepsilon_{\mathbf{n}}\|^2, \quad \left\| \sum_{k=\mathbf{1}}^{N_P} \frac{\partial f(\mathbf{p}_n)}{\partial p_k} (\varepsilon_{\mathbf{n}})_k \right\| \gg \left\| \sum_{k_1=1,k_2=1}^{N_P} \frac{\partial^2 f(\widetilde{\mathbf{p}})}{\partial p_{k_1} \partial p_{k_2}} (\varepsilon_{\mathbf{n}})_{k_1} (\varepsilon_{\mathbf{n}})_{k_2} \right\|, \tag{5.15}$$

and Eq. (5.4) is reduced to

$$|f(\mathbf{p}_n)| = \left\| \sum_{k=\mathbf{1}}^{N_P} \frac{\partial f(\mathbf{p}_n)}{\partial p_k} \right\| \|\varepsilon_{\mathbf{n}}\| = C_1 \|\varepsilon_{\mathbf{n}}\|, \quad \widetilde{\mathbf{p}} \in (\mathbf{0}, \varepsilon_n), \tag{5.16}$$

where

$$C_1 = \left\| \sum_{k=\mathbf{1}}^{N_P} \frac{\partial f(\mathbf{p}_n)}{\partial p_k} \right\| = \mathrm{O}\left(\left\| \frac{\partial f}{\partial p} \right\| \right). \tag{5.17}$$

Substituting Eqs. (5.8) and (5.16), we have that

$$\|\delta \mathbf{p}_n\| \leqslant \sqrt[3]{\frac{|f(\mathbf{p}_{n+1})|}{C_3}} = \sqrt[3]{\frac{C_1 \|\varepsilon_{\mathbf{n+1}}\|}{C_3}} = \sqrt[3]{\frac{C_1}{C_3}} \|\varepsilon_{\mathbf{n}+1}\|^{\frac{1}{3}}. \tag{5.18}$$

By Eq. (5.3), $\varepsilon_{\mathbf{n}+1} = \varepsilon_{\mathbf{n}} + \delta p_n$, hence

$$0 \leqslant \|\varepsilon_{\mathbf{n+1}}\| = \|\varepsilon_{\mathbf{n}}\| - \|\delta \mathbf{p}_n\| = \|\varepsilon_{\mathbf{n}}\| - \sqrt[3]{\frac{C_1}{C_3}} \|\varepsilon_{\mathbf{n+1}}\|^{\frac{1}{3}}, \tag{5.19}$$

$$\|\varepsilon_{\mathbf{n}}\| \geqslant \sqrt[3]{\frac{C_1}{C_3}} \|\varepsilon_{\mathbf{n+1}}\|^{\frac{1}{3}},$$

and

$$\frac{C_3}{C_1}\|\varepsilon_{\mathbf{n}}\|^3 \geqslant \|\varepsilon_{\mathbf{n+1}}\|. \tag{5.20}$$

The ratio $\frac{C_3}{C_1}$ is given by

$$\frac{C_3}{C_1} = \mathrm{O}\left(\left\|\frac{\partial^3 f}{\partial p^3}\right\| \Big/ \left\|\frac{\partial f}{\partial p}\right\|\right). \tag{5.21}$$

Recalling that the error of the second order Taylor approximation is bounded by $\|\frac{\partial^3 f}{\partial p^3}\|$ [see Eqs. (5.4), and (5.5)], Eq. (5.21) takes an intuitive form—in the minimization of functions that are well approximated by a second order expansion, we have $\|\frac{\partial^3 f}{\partial p^3}\| \to 0$ and $\frac{C_3}{C_1} \to 0$, thus speeding up the convergence.

Lemma 5.3. *The convergence range of the second order approximation based scheme is related to the energy of the high-order derivatives of the objective function f and therefore to the decay of its spectra in the Fourier domain. This provides a measure of the performance of the second order scheme given an objective function f.*

Proof. The convergence range $\varepsilon_{\max}$ is given by computing the error $\varepsilon_{\mathbf{n}}$, such that for any $\|\varepsilon_n\| > \|\varepsilon_{\max}\|$ applying the iterative step given, using Eq. (5.13), will result in an increase of the parameters error, instead of its decrease

$$\varepsilon_{\max} \leqslant \|\varepsilon_{\mathbf{n}+1}\| = \sqrt{\frac{C_3}{C_2}}\|\varepsilon_{\max}\|^{\frac{3}{2}},$$

and we get

$$\frac{C_2}{C_3} \leqslant \|\varepsilon_{\max}\|. \tag{5.22}$$

Eq. (5.22) implies that $\varepsilon_{\max}$ is maximized as $C_3 \to 0$. This corresponds to using an approximation whose order is higher than the order of the function f. For instance, the solution of a quadratic equation set using the second order approximation, will converge in a single iteration, using *any* initial estimate.

Further insight can be derived by applying Parseval's Theorem to C_2 and C_3. Thus,

$$C_2 = \left\|\frac{\partial^2 f}{\partial p^2}\right\| = \|F(\omega)\omega^2\|, \tag{5.23}$$

and

$$C_3 = \left\|\frac{\partial^3 f}{\partial p^3}\right\| = \|F(\omega)\omega^3\|, \tag{5.24}$$

and the ratio $\frac{C_2}{C_3}$ measures to the decay of the magnitude of Fourier transform of the image. Hence, a smoother images will tend to have a smaller C_3 coefficient, and a larger convergence range. In particular, this ratio can be efficiently computed for a given image and asses the convergence range $\varepsilon_{\max}$. This analysis also provides a justification to the common practice of smoothing the input images before applying the GM registration.

For the sake of completeness, we summarize the convergence properties of the first order GM scheme. Such schemes apply a first order Taylor expansion of $f(\mathbf{p})$, and are commonly known as the Gauss–Newton optimization algorithm [33,34]. Its convergence properties are given in the following Lemma, while details and rigorous analysis can be found in [30].

Lemma 5.4. *The convergence process of the GM and SGM schemes can be divided to two distinct phases, characterized by the deviation of the parameters* $\mathbf{p}$ *from their optimal value* $\mathbf{p}^*$ *and the convergence rate of the optimization scheme. Near the minimum* $\mathbf{p} \to \mathbf{p}^*$, $\|\varepsilon_n\| \to 0$, *a quadratic convergence rate is achieved. Away from the minimum, a slow linear convergence rate is achieved.*

$$\|\varepsilon_{\mathbf{n}+1}\| \leqslant C^L_{\mathrm{GM}} \cdot \|\varepsilon_n\| + C^S_{\mathrm{GM}} \cdot \|\varepsilon_n\|^2,$$

where C^S_{GM} and C^L_{GM} are constants.

Proof. The proof of the above Lemma was given in [30]. Similar results were derived in [34] for unconstrained Newton's methods, where small and large deviations are referred to as the *quadratically convergent* and *damped* convergence phases, respectively.

5.1. Symmetric third order GM convergence

The convergence properties of the SO3GM can be derived by considering Eq. (4.3), which defines the derivatives of $\widehat{f}$, the objective function minimized by the SO3GM. Since the SO3GM uses the same order of approximation as the O3GM, we get the same orders of convergence for both small and large deviations.

The difference lies in the ratio of C_3 and C_2, the overall energy of the derivatives. Denote by $\widetilde{C}_3$, $\widetilde{C}_2$, and $\widetilde{C}_1$ the corresponding derivatives norms for the SO3GM. We compare their values to C_3, C_2 and C_1 related by the O3GM, and show that the convergence rate is improved.

Using Eq. (4.3) we get that the norm of $\widetilde{C}_2$ is bounded by

$$\begin{aligned}\widetilde{C}_2 &= \frac{1}{4}\left\|\frac{\partial^2 I_2(x_i^1, y_i^1)}{\partial p_j \partial p_s} - \frac{\partial^2 I_1(x_i^1, y_i^1)}{\partial p_j \partial p_s}\right\| \\ &\leqslant \frac{1}{4}\left\|\frac{\partial^2 I_2}{\partial p^2}\right\| + \frac{1}{4}\left\|\frac{\partial^2 I_1}{\partial p^2}\right\|, \\ &= \frac{1}{2}\left\|\frac{\partial^2 I_2}{\partial p^2}\right\| = \frac{C_2}{2}.\end{aligned}$$

For $\widetilde{C}_3$ and $\widetilde{C}_1$ we have

$$\begin{aligned}\widetilde{C}_3 &= \frac{1}{8}\left\|\frac{\partial^3 I_2(x_i^1,y_i^1)}{\partial p_j \partial p_s \partial p_t}+\frac{\partial^3 I_1(x_i^1,y_i^1)}{\partial p_j \partial p_s \partial p_t}\right\| \\ &\leqslant \frac{1}{8}\left\|\frac{\partial^3 I_2}{\partial p^3}\right\|+\frac{1}{8}\left\|\frac{\partial^3 I_1}{\partial p^3}\right\|, \\ &= \frac{1}{4}\left\|\frac{\partial^3 I_2}{\partial p^3}\right\| = \frac{C_3}{4},\end{aligned}$$

and

$$\begin{aligned}\widetilde{C}_1 &= \frac{1}{2}\left\|\frac{\partial I_2\left(x_i^{(2)},y_i^{(2)},\mathbf{p}\right)}{\partial p_j}+\frac{\partial I_1(x_i^{(1)},y_i^{(1)})}{\partial p_j}\right\| \\ &\leqslant \frac{1}{2}\left\|\frac{\partial I_2}{\partial p}\right\|+\frac{1}{2}\left\|\frac{\partial I_2}{\partial p}\right\|, \\ &= \left\|\frac{\partial I_2}{\partial p_j}\right\| = C_1.\end{aligned}$$

Hence, we compute the convergence rates and ranges for the SO3GM. For large deviations we get

$$\|\varepsilon_{\mathbf{n+1}}\| \leqslant \frac{\widetilde{C}_3}{\widetilde{C}_2}\|\varepsilon_{\mathbf{n}}\|^{\frac{3}{2}} = \frac{C_3/4}{C_2/2}\|\varepsilon_{\mathbf{n}}\|^{\frac{3}{2}} = \frac{1}{2}\frac{C_3}{C_2}\|\varepsilon_{\mathbf{n}}\|^{\frac{3}{2}} \tag{5.25}$$

and for small deviations

$$\|\varepsilon_{\mathbf{n+1}}\| \leqslant \frac{\widetilde{C}_3}{\widetilde{C}_1}\|\varepsilon_{\mathbf{n}}\|^{3} = \frac{C_3/4}{C_1}\|\varepsilon_{\mathbf{n}}\|^{3} = \frac{1}{4}\frac{C_3}{C_1}\|\varepsilon_{\mathbf{n}}\|^{3} \tag{5.26}$$

Comparing these results to Eqs. (5.13) and (5.20) we see that the SO3GM allows better convergence rates, especially for low deviations.

Next we show that the convergence range $\varepsilon_{\max}^{\mathrm{SO3GM}}$ is also improved and by substituting $\widetilde{C}_2$ and $\widetilde{C}_3$ in Eq. (5.22) we have

$$\varepsilon_{\max}^{\mathrm{SO3GM}} = \frac{\widetilde{C}_2}{\widetilde{C}_3} = \frac{C_2/2}{C_3/4} = 2\frac{C_2}{C_3} = 2\varepsilon_{\max}^{\mathrm{O3GM}}. \tag{5.27}$$

6. Experimental results

This section describes the performance of the proposed algorithms and verifies the convergence analysis given in Section 5. The same implementations of the *iterative refinement* and *multiscale embedding* were used for the O3GM, SO3GM, SGM, and GM algorithms. Thus, the only difference between the schemes is the *single iteration* module. The translation and rotation simulations were conducted using the *Lena* and *Airfield* images transformed by bilinear interpolation, while the affine and projective motion were tested using real images. The images in Fig. 5 were taken by a photogrammetric aerial camera, while the images in Fig. 7 were taken by a hand held 35 mm camera. The GM algorithm was implemented according to [15,35] that are considered state-of-the-art. The first and second order derivatives were computed using the following central difference approximations

$$\begin{aligned}\frac{\partial I(i,j)}{\partial x} &= \frac{1}{2}(I(i,j+1)-I(i,j-1)), \\ \frac{\partial I(i,j)}{\partial y} &= \frac{1}{2}(I(i+1,j)-I(i-1,j)), \\ \frac{\partial^2 I(i,j)}{\partial x^2} &= \frac{1}{4}(I(i,j+2)-2I(i,j)+I(i,j-2)), \\ \frac{\partial^2 I(i,j)}{\partial y^2} &= \frac{1}{4}(I(i+2,j)-2I(i,j)+I(i-2,j)), \\ \frac{\partial^2 I(i,j)}{\partial y \partial x} &= \frac{\partial^2 I(i,j)}{\partial y \partial x} = \frac{1}{4}(I(i-1,j-1)-I(i-1,j+1) \\ &\quad +I(i+1,j-1)+I(i+1,j+1)).\end{aligned}$$

In order to avoid spatiotemporal aliasing and allow accurate computation of the spatial derivatives, the images were initially smoothed by a Gaussian filter with a bandwidth of $\sigma = 11$. The same filter was used to construct the multiscale pyramid whose scales were 1 (original scale) and 1/3. The support of the filters is larger than the one used in prior works [15,35]. This allows us to better estimate the second order image derivatives and also improves the convergence range of the regular GM scheme such as in [15,35].

We used the upper-left image corner as the origin of axis. The common support of I_1 and I_2 was computed in each iteration by applying the motion estimate to the bounding rectangular of I_2. Next, we identify the intersection points between the bounding rectangles of I_1 and (the transformed) I_2 and compute the convex hull of their mutual support.

The focal point of the experiments in this section is to illustrate the improved convergence properties of the proposed schemes. In order to asses the convergence properties (rate and range), the figures show the alignment error [Eq. (6.1)] vs. the number of iterations. It was shown in [30] that the convergence rate of the alignment error and the motion parameters are the same.

The alignment error is given in terms of the mean squared error

$$\text{Alignment error} = \sqrt{\frac{1}{N_m}\sum\nolimits_{(x_i^1,y_i^1)\in S}\left(I_1(x_i^1,y_i^1)-I_2(x_i^1,y_i^1)\right)^2} \tag{6.1}$$

where $N_m = \|S\|$ is the number of common pixels.

The dimensions of the images used in the simulations is in the order of 512^2 pixels. In least squares based optimization schemes, the larger the equation set, the better the robustness to noise and computational accuracy. Thus, the larger the registered images, the more robust their registration by the GM and O3GM schemes.

We empirically study the convergence properties in Section 6.3, and compare the results to the rigorous analysis presented in Section 5. Finally, as higher order derivatives are known to be sensitive to noise, we asses the convergence of the proposed approaches in the presence of noise, and different interpolation schemes in Section 6.4.

6.1. Rotation and translation estimation

We start by applying the proposed scheme to the estimation of translations and rotations. For that we used the Lena and Airfield images shown in Fig. 1. These images were rotated around their center and translated. Different registration schemes were applied to align the transformed replicas to the original images, using a zero initial estimate of the motion.

First, we recovered small motions for which all of the alignment schemes converged. This allows us to compare the convergence rates over the different schemes, and verify the analysis given in Lemmas 5.1 and 5.2. All of the schemes were applied using a single resolution scale. These results are presented in Figs. 2a and 3a. For both images, the third order schemes (O3GM and SO3GM), outperformed the corresponding standard schemes, GM and SGM, respectively.

We then studied the estimation larger motions in Figs. 2b and 3b. There, we applied a resolution pyramid to all schemes, as such pyramids are the common approach to improving the registration range of gradient methods. In all of these instances, the O3GM converged significantly faster than the first order formulations (GM, SGM), and was outperformed by the SO3GM, while being five times faster than the regular GM.

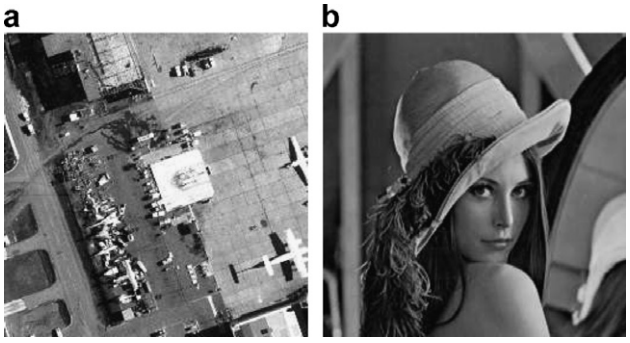

Fig. 1. The test images used for evaluating the proposed scheme by recovering rotations and translations. (a) Airfield. (b) Lena.

The convergence of larger motions is presented in Fig. 4, where we significantly enlarged the motion magnitudes compared to Figs. 3 and 2. The O3GM and SO3GM schemes converged, while the first order GM and SGM diverged. This exemplifies the superiority of the second order approach when dealing with large motions. The registration accuracy of the O3GM and SO3GM that was of $O(10^{-15})$, for all motion parameters.

6.2. Affine and projective motion estimation

The registration results of real images using the affine and projective motion models are given in Figs. 5 and 7, respectively. The initial estimate of the motion was given as a translation, computed by aligning the X marks in both images. We intentionally chose an inaccurate initial estimate, making the residual motion (estimated by the various GM schemes) large. The same initial motion was used by all the different schemes. For these real images, the final alignment error results from the lack of perfect matching, and the existence of non corresponding (outlier) objects in both images.

In the affine case (Fig. 5), the O3GM outperformed the GM by converging twice as fast. Better convergence was achieved by the symmetric motion models (SGM and SO3GM) and the best convergence was achieved by the SO3GM. The initial estimate (based on the translation between the X signs in Fig. 5) was $(\delta x, \delta y) = (231, -17)$ and the computed motion was

$$x_2 = 1.0516x_1 + 0.0708y_1 + 217.5,$$
$$y2 = 0.0058x_1 + 0.9387y_1 - 9.23.$$

In order to statistically asses the improvement achieved by the O3GM, we computed a set of 1000 random affine motions and applied them to the Lena and Airfield images. The images were then registered using the GM and O3GM schemes. As different affine motions result in different alignment errors and convergence curves, we normalized these curves by computing the number of iterations needed

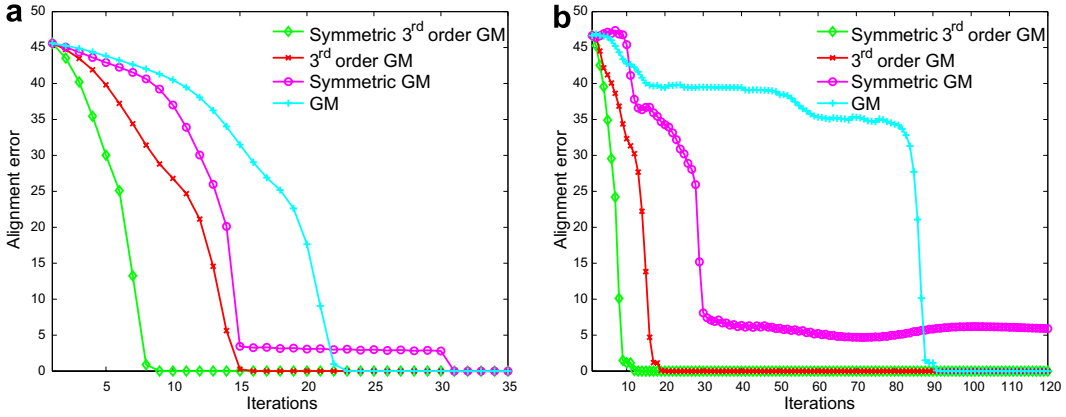

Fig. 2. Rotation and translation estimation results for the *Airfield* image. The image was rotated and translated by (a) $(\theta = 5°, \delta x = 20, \delta y = 20)$. (b) $(\theta = 10°, \delta x = 35, \delta\ y = 35)$. A zero initial estimate of the motion was used. The second order formulations converged significantly faster than the first order formulations. The SO3GM algorithm converged twice as fast as the O3GM algorithm.

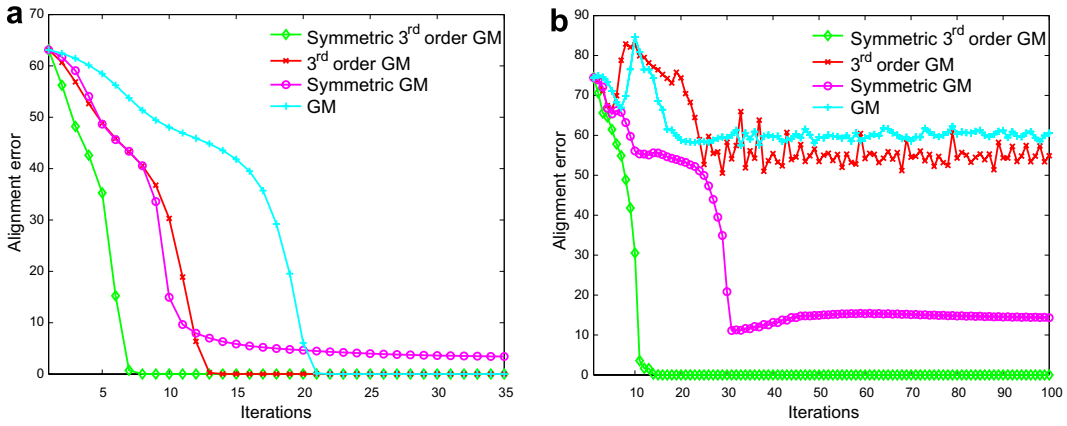

Fig. 3. Registration results for the *Lena* image. The image was rotated and translated by (a) ($\theta = 5°, \delta x = 20, \delta y = 20$). (b) ($\theta = 10°, \delta\ x = 20, \delta y = 20$). A zero initial estimate of the motion was used. The SO3GM converged significantly faster than the SGM. The other algorithms diverged.

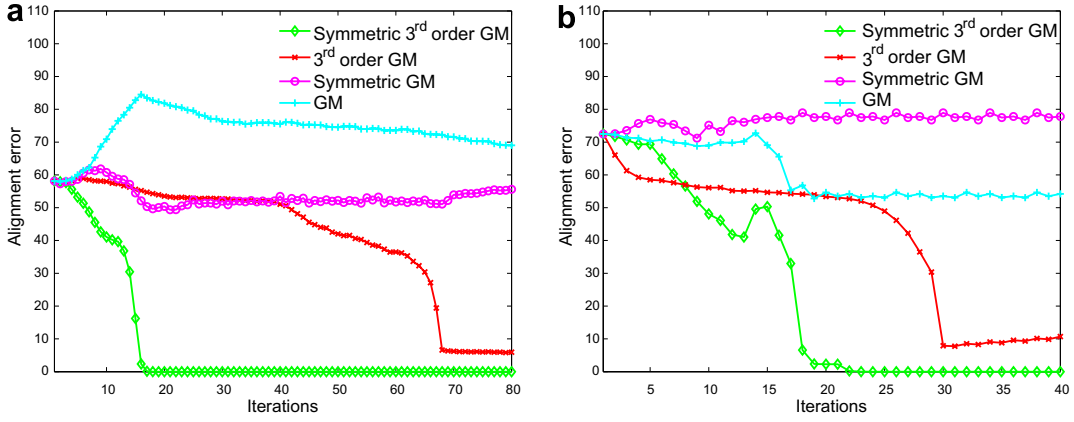

Fig. 4. Registration results for large motions. (a) *Airfield* image rotated and translated by ($\theta = 15°, \delta x = 0, \delta y = 0$). (b) *Lena* image rotated and translated by ($\theta = 20°, \delta x = 0, \delta y = 0$). A zero initial estimate of the motion was used. In both cases the SO3GM converged significantly faster than the O3GM, while the first order schemes diverged.

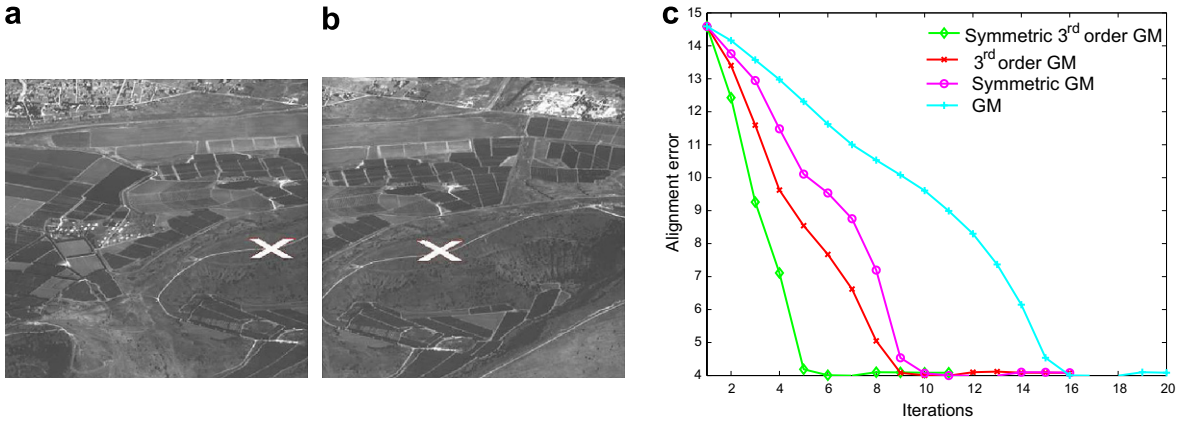

Fig. 5. Registration results for affine motion. The initial estimate of the motion was given by marked by the red X. The SO3GM converged 4 times faster than the GM and twice as fast as the O3GM and SGM. (For interpretation of color mentioned in this figure the reader is referred to the web version of the article.)

to reach certain error alignment ratios. These are the number of iterations needed to reduce the alignment error to a particular percentage of the initial error. The error ratio is depicted by the x-axis of Fig. 6. Then, in order to compare the GM and O3GM, we present in the average ratio of GM to O3GM iterations (y-axis) needed to reach corresponding relative alignment errors. Similar to Fig. 5, the O3GM required less iterations than the GM and the standard deviation was of $O(10^{-2})$, proving that the improvement achieved by the O3GM is statistically stable.

The results of registering the panoramic images using a projective motion model are presented in Fig. 7. These

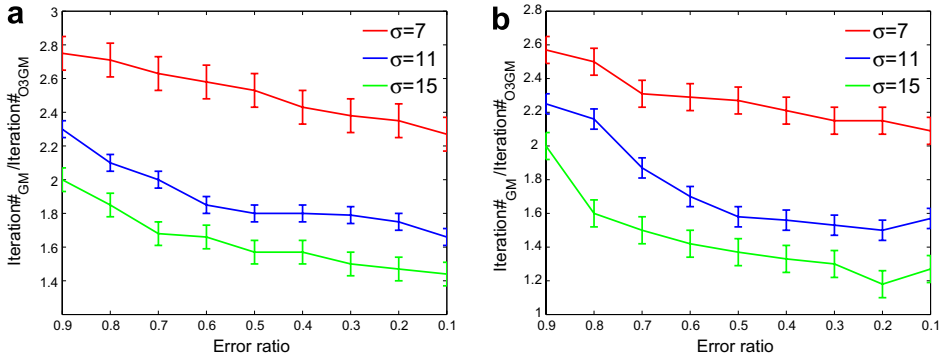

Fig. 6. Registration statistics for 1000 randomly generated affine motions. Random affine motions were applied to the Lena (a) and Airfield (b) images. We compare the number of GM and O3GM iterations needed to decrease the alignment error to a particular ratio, and the corresponding standard deviation. The x-axis shows the alignment error ratio, while the y-axis depicts the ratio of GM iterations to O3GM iterations needed to achieve a particular error ratio.

images have a significantly different brightness due to the auto-exposure of the camera. We left the brightness as is, to make the registration more difficult. The initial estimate was $(\delta x, \delta y) = (583, -43)$ and the computed motion was

$$x_2 = \frac{0.6295x_1 + 0.0021y_1 + 596.6587}{-4\cdot 10^{-4}x_1 + 1.1\times 10^{-5}y_1 + 1},$$
$$y_2 = \frac{-0.1038x_1 + 0.9324y_1 - 25.3759}{-4\cdot 10^{-4}x_1 + 1.1\times 10^{-5}y_1 + 1}.$$

The timing results for the affine case are given in Table 1, where the measurement were taken on a 2.8 GHz PC computer and the algorithms were implemented in non-optimized C++. For these high-order models, the complexity of the proposed algorithms is higher than the computational complexity of the GM and SGM. Yet, it is useful for estimating large motions where the GM and SGM might diverge.

6.3. Convergence analysis verification

The convergence analysis given in Lemmas 5.1 and 5.2, suggests that the convergence ratios, $\frac{C_3}{C_2}$ and $\frac{C_3}{C_1}$, for the large and small motions, respectively, are related to the energy of the image derivatives. Equivalently, $\frac{C_3}{C_2}$ and $\frac{C_3}{C_1}$ are related to the spectral content of the input images (Lemma 5.3).

In this section, we study this relationship empirically, by varying the derivatives/spectral content of the images in Fig. 1, by smoothing them with Gaussian low-pass filters. For each image, several smoothed replicas, corresponding to standard deviations of $\sigma = 7, 11, 15$, were computed. These images were then aligned using the O3GM and GM schemes. Fig. 8a and b show the convergence curves over the different image smoothing factors, using a *logarithmic* y-axis. Thus, Eqs. (5.13) and (5.20) become

$$\log\|\varepsilon_{\mathbf{n+1}}\| = \log\sqrt{\frac{C_3}{C_2}} + \frac{3}{2}\log\|\varepsilon_{\mathbf{n}}\|, \tag{6.2}$$

and

$$\log\|\varepsilon_{\mathbf{n}+1}\| = \log\frac{C_3}{C_1} + 3\log\|\varepsilon_{\mathbf{n}}\|, \tag{6.3}$$

respectively. Thus, we are able to study the different convergence phases, and the dependence if the coefficients $\frac{C_3}{C_2}$ and $\frac{C_3}{C_1}$ on the smoothness of the input images.

A few conclusions can be drawn from Fig. 8: first, the convergence is indeed made of two phases, starting with a low slope ($\frac{3}{2}$ and 1 for the O3GM and GM, respectively)

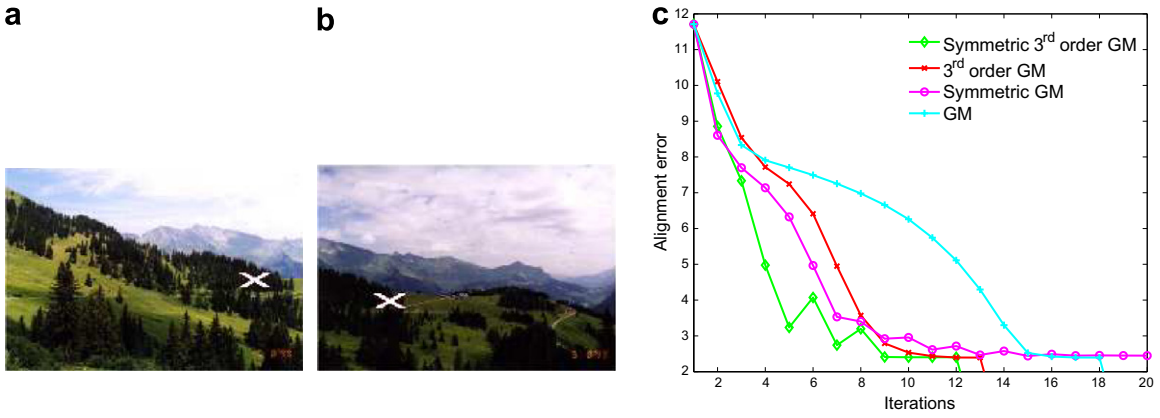

Fig. 7. Registration results for projective motion. The initial estimate of the motion was given by the red X. The SGM converged better than the O3GM while the SO3GM showed the best convergence properties. (For interpretation of color mentioned in this figure the reader is referred to the web version of the article.)

Table 1
Timing results for the affine registration given in Fig. 5

	Iteration (#)	Total timing (s)	Average iteration timing (s)
GM	22	0.8	0.036
SGM	17	1.1	0.064
O3GM	17	1.2	0.070
SO3GM	12	1.4	0.117

and then turning into a steeper slope (3 and 2 for the O3GM and GM, respectively). Second, by comparing the convergence curves for a given image and scheme (say the O3GM), over the different smoothing factors, we notice that the curves only differ by a vertical shift, while having the same slopes. This corroborates Lemma 5.3, that related the spectral content of the input images to the coefficients $\frac{C_3}{C_2}$ and $\frac{C_3}{C_1}$, and not to the convergence rates ($\frac{3}{2}$ and 1). Last, we note that the O3GM and SGM exhibit similar slopes. Indeed, it was shown in [30], that the SGM can achieve O3GM-like convergence for small motions.

6.4. Sensitivity to noise and interpolation errors

As higher order schemes are known to be sensitive to noise, we tested the robustness of the proposed scheme to two typical noise sources in image registration. First, we considered the influence of White Gaussian noise (WGN). WGN was added to both images in Fig. 5 with $\sigma = 0, 30, 60$. The GM and SGM schemes were applied and the results are depicted in Fig. 9. The same initial motion was assumes as in the prior section. Fig. 9a and b shows Fig. 5a after adding WGN with $\sigma = 30$ and $\sigma = 60$, respectively. The initial estimate of the motion, was chosen such that the residual motion was large. The convergence results depicted in Fig. 9c, shows that the second order approaches were more stable than the GM in the presence of noise for all noise levels. As before the SO3GM outperformed the other schemes.

Next we considered the errors induced by using different image interpolation schemes. Interpolation is used in Step #1 of Algorithm 1, where at each iteration the current estimate of the motion is used to align the images. Fig. 10 depicts the results of applying the GM and O3GM to the same registration problem as in Fig. 5, using bilinear, cubic, and spline interpolation. The same registration parameters such as the initial estimate, iterations number and multiscale pyramid were used in all of the simulations.

The results for the GM and O3GM are shown in Fig. 10a and b, respectively. For both schemes the convergence properties are unchanged by the use of different interpolation schemes. We attribute that to the following issues:

1. It is custom to smooth the images before applying a gradient based registration scheme. This attenuates the energy of the high frequencies in the image, thus, reducing the Taylor series approximation error. Note the discussion in Section 6.3. Hence, in practice, the images we register, are quite smooth and there is no significant difference wether they are interpolated by a bilinear or a higher order scheme. Namely, a higher order interpolation scheme, provides sharper result images, but if the input image is smooth to begin with, it makes no difference which interpolation scheme is used.
2. Suppose we aim to register the images $I_1(x,y)$ and $I_2(x+\Delta x, y+\Delta y)$, ($\Delta x$ and Δy are unknown), and that after n iterations the current estimate of the motion is $(\widetilde{\Delta x}, \widetilde{\Delta y})$. Image interpolation is used to compute the image $\widetilde{I}_2(x,y) = I_2(x+\widetilde{\Delta x}, y+\widetilde{\Delta y})$ before applying the Taylor series approximation. Hence, using a low-order interpolation introduces the noise term $N(x_i^1, y_i^1)$ into $\widetilde{I}_2(x,y)$ and Eq. (2.5) becomes

$$I_1(x_i^1, y_i^1) = I_2(x_i^1, y_i^1) + \sum_{k=1}^{N_P} \frac{\partial I_2(x_i^1, y_i^1)}{\partial p_k} \varepsilon_k + \frac{1}{2} \sum_{k=1,m=1}^{N_P} \frac{\partial I_2(x_i^1, y_i^1)}{\partial p_k \partial p_m} (\widetilde{\varepsilon}_{\mathbf{n}})_k (\widetilde{\varepsilon}_{\mathbf{n}})_m + N(x_i^1, y_i^1), \quad \widetilde{\varepsilon} \in [0, \varepsilon].$$

First, the intensity of this noise term is negligible, due to the smoothness of the registered image. Second, Eq. (2.5)

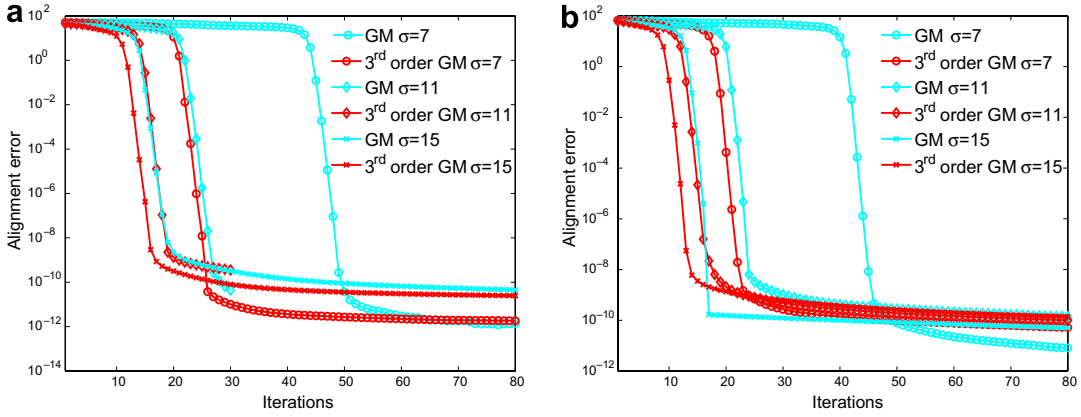

Fig. 8. The convergence of the O3GM and GM schemes with respect to the smoothness of the input images. The images were smoothed with a Gaussian low-pass filter with a varying width σ. (a) Registration of the Airfield image ($\theta = 5°, \delta x = 20, \delta y = 20$). (b) Registration of the Lena image ($\theta = 5°, \delta x = 20, \delta y = 20$). These images are shown in Fig. 1.

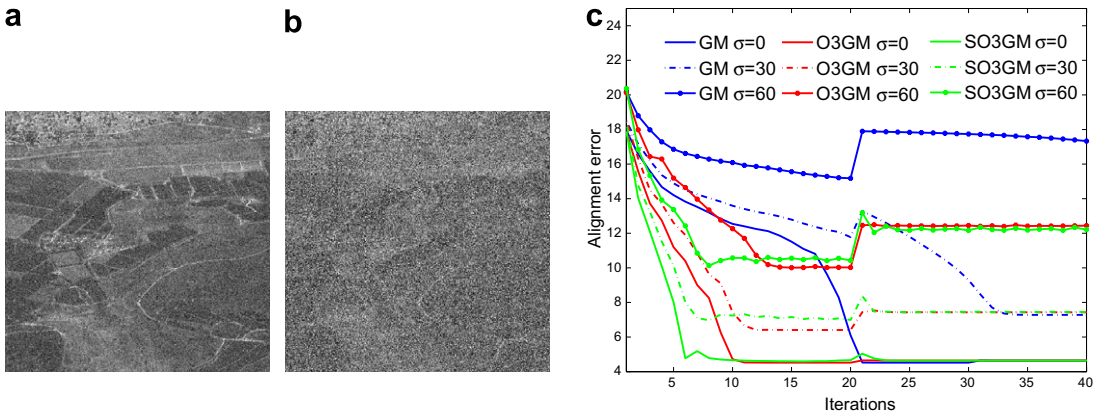

Fig. 9. Estimating affine motion of noisy images. The O3GM and SO3GM allow better registration of noisy images, compared to the regular GM. The initial estimate is the same as in Fig. 5 and σ is the standard deviation of the added noise.

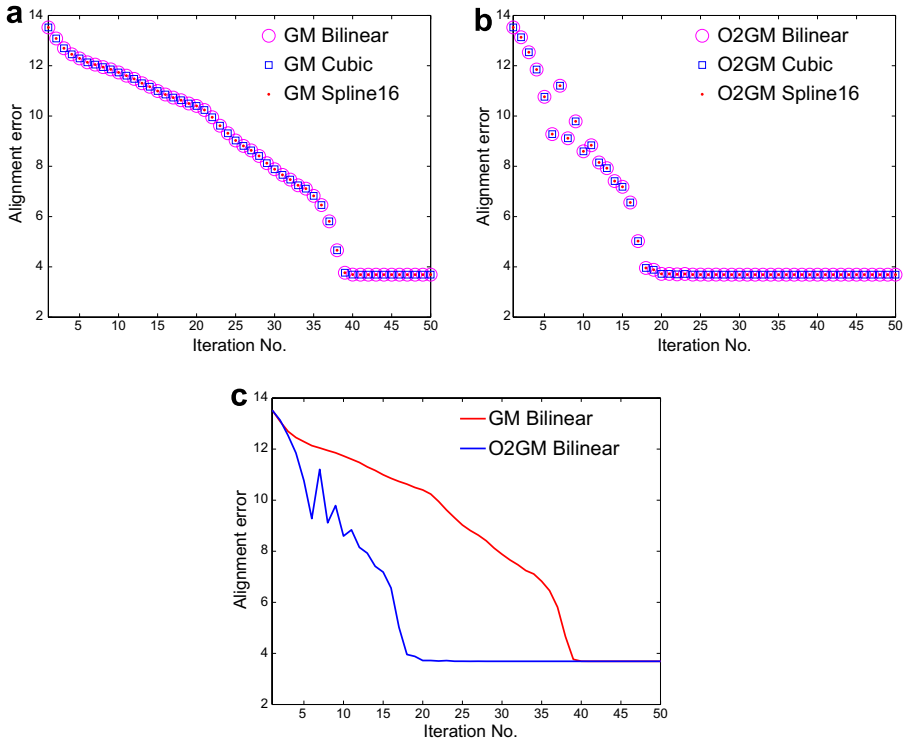

Fig. 10. Image registration using different image interpolation schemes: bilinear, cubic, and spline of a support of [16]. All of the simulations were run with the same parameters: initial estimate of the motion, number of iterations, resolution scales, etc. (a) GM. (b) O3GM. (c) A comparison between the GM and the O3GM that prevails by converging in fewer iterations.

is minimized as part of a least squares solution, and the noise N is averaged over all of the equations. For instance, for the first order schemes, we formulate a set of linear equations and the standard deviation of the solution reduces as $\frac{1}{N_m^2}, N_m$ being the number of common pixels.

7. Conclusions and future work

In this work we presented the O3GM and SO3GM image registration algorithms which enhance the performance of gradient based registration methods. These algorithms extend the current state-of-the-art schemes and were shown to have superior convergence properties. They are especially suitable for the estimation of large affine and projective motions that cannot be bootstrapped by Fourier domain methods. Future work includes the application of the O3GM and SO3GM to numerically ill-posed computer vision problems that are based on the gradient methods, such as wide baseline stereo [36] and 3D reconstruction [10].

Appendix A. Third order translation estimation

This section presents a simplified derivation of the O3GM and SO3GM formulations for the 2D translation motion model. For this motion model $\mathbf{p} = (\Delta x, \Delta y)$ and

$$x_i^2(\mathbf{p}) = x_i^1 + \Delta x,$$
$$y_i^2(\mathbf{p}) = y_i^1 + \Delta y. \tag{A.1}$$

A.1. Third order translation estimation

We start with the O3GM, where given the input images I_1 and I_2 we approximate I_1 by a second order expansion of I_2

$$I_1(x_i^1, y_i^1) = I_2(x_i^1, y_i^1) + \frac{\partial I_2(x_i^1, y_i^1)}{\partial x}\Delta x + \frac{\partial I_2(x_i^1, y_i^1)}{\partial y}\Delta y + \frac{1}{2}\frac{\partial^2 I_2(x_i^1, y_i^1)}{\partial x^2}\Delta x^2 + \frac{1}{2}\frac{\partial^2 I_2(x_i^1, y_i^1)}{\partial y^2}\Delta y^2 + \frac{\partial^2 I_2(x_i^1, y_i^1)}{\partial y \partial x}\Delta x \Delta y. \tag{A.2}$$

Eq. (A.2) is formulated for the pixels common to I_1 and I_2, and

$$r_i = \frac{\partial I_2(x_i^1, y_i^1)}{\partial x}\Delta x + \frac{\partial I_2(x_i^1, y_i^1)}{\partial y}\Delta y + \frac{1}{2}\frac{\partial^2 I_2(x_i^1, y_i^1)}{\partial x^2}\Delta x^2 + \frac{1}{2}\frac{\partial^2 I_2(x_i^1, y_i^1)}{\partial y^2}\Delta y^2 + \frac{\partial^2 I_2(x_i^1, y_i^1)}{\partial y \partial x}\Delta x \Delta y - I_t(x_1^1, y_1^1) \tag{A.3}$$

In order to compute the Hessian and Jacobian, we use the chain rule derivatives

$$\frac{\partial I_2}{\partial \Delta y} = \frac{\partial I_2}{\partial x}\underbrace{\frac{\partial \Delta y}{\partial x}}_{0} + \frac{\partial I_2}{\partial y}\underbrace{\frac{\partial \Delta y}{\partial y}}_{1} = \frac{\partial I_2}{\partial y},$$

and

$$\frac{\partial I_2}{\partial \Delta x} = \frac{\partial I_2}{\partial x}\underbrace{\frac{\partial \Delta x}{\partial x}}_{1} + \frac{\partial I_2}{\partial y}\underbrace{\frac{\partial \Delta x}{\partial y}}_{0} = \frac{\partial I_2}{\partial x}.$$

Thus, the Jacobian is given by

$$J = \begin{bmatrix} \frac{\partial I_2(x_1^1, y_1^1)}{\partial x} + \frac{\partial^2 I_2(x_1^1, y_1^1)}{\partial x^2}\Delta x + \frac{\partial^2 I_2(x_1^1, y_1^1)}{\partial y \partial x}\Delta y & \frac{\partial I_2(x_1^1, y_1^1)}{\partial y} + \frac{\partial^2 I_2(x_1^1, y_1^1)}{\partial y^2}\Delta y + \frac{\partial^2 I_2(x_1^1, y_1^1)}{\partial y \partial x}\Delta x \\ \vdots & \vdots \end{bmatrix}. \tag{A.4}$$

It is straight forward to verify that $\frac{\partial^2 I_2}{\partial \Delta y^2} = \frac{\partial^2 I_2}{\partial y^2}$ and $\frac{\partial^2 I_2}{\partial \Delta x^2} = \frac{\partial^2 I_2}{\partial x^2}$. Thus, the Hessian $\mathbf{H}_i$ of Eq. (A.2) is given by

$$H_i = \begin{bmatrix} \frac{\partial^2 I_2(x_1^1, y_1^1)}{\partial x^2} & \frac{\partial^2 I_2(x_1^1, y_1^1)}{\partial x \partial y} \\ \frac{\partial^2 I_2(x_2^1, y_2^1)}{\partial y \partial x} & \frac{\partial^2 I_2(x_2^1, y_2^1)}{\partial y^2} \end{bmatrix}. \tag{A.5}$$

Since it is common to smooth the input images, one can apply Clairaut's theorem

$$\frac{\partial^2 I_2(x_1^1, y_1^1)}{\partial x \partial y} = \frac{\partial^2 I_2(x_1^1, y_1^1)}{\partial y \partial x}.$$

A.2. Symmetric third order translation estimation

In the symmetric formulation (SO3GM) we solve

$$I_1\left(x_i^1 + \frac{\Delta x}{2}, y_i^1 + \frac{\delta y}{2}\right) = I_2\left(x_i^2 - \frac{\Delta x}{2}, y_i^2 - \frac{\Delta y}{2}\right). \tag{A.6}$$

We expand both sides of Eq. (A.6) in a second order Taylor series expansion

$$I_1\left(x_i^1 + \frac{\Delta x}{2}, y_i^1 + \frac{\Delta y}{2}\right) = I_1(x_i^1, y_i^1) + \frac{\partial I_1(x_i^1, y_i^1)}{\partial x}\frac{\Delta x}{2} + \frac{\partial I_1(x_i^1, y_i^1)}{\partial y}\frac{\Delta y}{2} + \frac{1}{8}\Delta x^2 \times \frac{\partial^2 I_1(x_i^1, y_i^1)}{\partial x^2} + \frac{1}{8}\Delta y^2 \frac{\partial^2 I_1(x_i^1, y_i^1)}{\partial y^2} + \frac{1}{4}\Delta x \Delta y \frac{\partial^2 I_1(x_i^1, y_i^1)}{\partial y \partial x}.$$

and r_i is given by

$$r_i = \frac{\Delta x}{2}\left(\frac{\partial I_1(x_i^1, y_i^1)}{\partial x} + \frac{\partial I_2(x_i^1, y_i^1)}{\partial x}\right) + \frac{\Delta y}{2}\left(\frac{\partial I_1(x_i^1, y_i^1)}{\partial y} + \frac{\partial I_2(x_i^1, y_i^1)}{\partial y}\right) + \frac{1}{8}\Delta x^2\left(\frac{\partial^2 I_1(x_i^1, y_i^1)}{\partial x^2} - \frac{\partial^2 I_2(x_i^1, y_i^1)}{\partial x^2}\right) + \frac{1}{8}\Delta y^2\left(\frac{\partial^2 I_1(x_i^1, y_i^1)}{\partial y^2} - \frac{\partial^2 I_2(x_i^1, y_i^1)}{\partial y^2}\right) + \frac{1}{4}\Delta x \Delta y\left(\frac{\partial^2 I_1(x_i^1, y_i^1)}{\partial y \partial x} - \frac{\partial^2 I_2(x_i^1, y_i^1)}{\partial y \partial x}\right) - I_t(x_i^1, y_i^1) \tag{A.7}$$

Comparing Eqs. (A.7) and (A.3), we notice that the coefficients of the powers of Δx or Δy, in Eq. (A.7), are the scalings of the sum or difference of the corresponding entries in Eq. (A.3). Thus, in order to compute Eq. (A.7) it suffices to compute Eq. (A.3) for both input images, store the entries in a matrix, and compute Eq. (A.7) by adding and subtracting the columns of that matrix. The Jacobian and Hessian can then be numerically computed the same way as in Section A.1.

Appendix B. Affine third order formulation

The affine motion model is given in Eq. (2.3) and the derivatives are computed using the derivative chain rule, where for any of the motion parameters $v \in \{a, b, c, d, e, f\}$ the first order partial derivative is given by

$$\frac{\partial I}{\partial v} = \frac{\partial I}{\partial x}\frac{\partial x}{\partial v},$$

and we get

$$\frac{\partial I}{\partial a} = I_x x_i^1, \quad \frac{\partial I}{\partial b} = I_x y_i^1, \quad \frac{\partial I}{\partial c} = I_x,$$
$$\frac{\partial I}{\partial d} = I_y x_i^1, \quad \frac{\partial I}{\partial e} = I_y y_i^1, \quad \frac{\partial I}{\partial f} = I_y. \tag{9.8}$$

The second order derivatives are computed by applying the chain rule to Eq. (9.8) and using Clairaut's theorem [37] to compute the mixed partial derivatives

$$\frac{\partial^2 I}{\partial a^2} = \frac{\partial}{\partial a}\left(\frac{\partial I}{\partial a}\right) = \frac{\partial}{\partial a}\left(I_x x_i^1\right) = I_{xa} x_i^1,$$
$$\frac{\partial^2 I}{\partial a \partial b} = \frac{\partial}{\partial b}\left(\frac{\partial I}{\partial a}\right) = \frac{\partial}{\partial b}\left(I_x x_i^1\right) = I_{xb} x_i^1,$$
$$\frac{\partial^2 I}{\partial a \partial d} = \frac{\partial}{\partial d}\left(\frac{\partial I}{\partial a}\right) = \frac{\partial}{\partial d}\left(I_x x_i^1\right) = I_{xd} x_i^1,$$
$$\frac{\partial^2 I}{\partial a \partial d} = \frac{\partial}{\partial e}\left(\frac{\partial I}{\partial a}\right) = \frac{\partial}{\partial e}\left(I_x x_i^1\right) = I_{xe} x_i^1,$$
$$\frac{\partial^2 I}{\partial a \partial c} = \frac{\partial}{\partial c}\left(\frac{\partial I}{\partial a}\right) = \frac{\partial}{\partial c}\left(I_x x_i^1\right) = I_{xc} x_i^1,$$
$$\frac{\partial^2 I}{\partial a \partial f} = \frac{\partial}{\partial f}\left(\frac{\partial I}{\partial a}\right) = \frac{\partial}{\partial f}\left(I_x x_i^1\right) = I_{xf} x_i^1,$$
$$\frac{\partial^2 I}{\partial b^2} = \frac{\partial}{\partial b}\left(\frac{\partial I}{\partial b}\right) = \frac{\partial}{\partial b}\left(I_x y_i^1\right) = I_{xb} y_i^1,$$
$$\frac{\partial^2 I}{\partial b \partial d} = \frac{\partial}{\partial d}\left(\frac{\partial I}{\partial b}\right) = \frac{\partial}{\partial d}\left(I_x y_i^1\right) = I_{xd} y_i^1,$$
$$\frac{\partial^2 I}{\partial b \partial c} = \frac{\partial}{\partial c}\left(\frac{\partial I}{\partial b}\right) = \frac{\partial}{\partial c}\left(I_x y_i^1\right) = I_{xc} y_i^1,$$
$$\frac{\partial^2 I}{\partial b \partial e} = \frac{\partial}{\partial e}\left(\frac{\partial I}{\partial b}\right) = \frac{\partial}{\partial e}\left(I_x y_i^1\right) = I_{xe} y_i^1,$$
$$\frac{\partial^2 I}{\partial b \partial f} = \frac{\partial}{\partial f}\left(\frac{\partial I}{\partial b}\right) = \frac{\partial}{\partial f}\left(I_x y_i^1\right) = I_{xf} y_i^1,$$
$$\frac{\partial^2 I}{\partial c^2} = \frac{\partial}{\partial c}\left(\frac{\partial I}{\partial c}\right) = \frac{\partial}{\partial c}\left(I_x\right) = I_{xc},$$
$$\frac{\partial^2 I}{\partial c \partial d} = \frac{\partial}{\partial d}\left(\frac{\partial I}{\partial c}\right) = \frac{\partial}{\partial d}\left(I_x\right) = I_{xd},$$
$$\frac{\partial^2 I}{\partial c \partial e} = \frac{\partial}{\partial e}\left(\frac{\partial I}{\partial c}\right) = \frac{\partial}{\partial e}\left(I_x\right) = I_{xe},$$
$$\frac{\partial^2 I}{\partial c \partial f} = \frac{\partial}{\partial f}\left(\frac{\partial I}{\partial c}\right) = \frac{\partial}{\partial f}\left(I_x\right) = I_{xf},$$
$$\frac{\partial^2 I}{\partial d^2} = \frac{\partial}{\partial d}\left(\frac{\partial I}{\partial d}\right) = \frac{\partial}{\partial d}\left(I_y x_i^1\right) = I_{yd} x_i^1,$$
$$\frac{\partial^2 I}{\partial d \partial e} = \frac{\partial}{\partial e}\left(\frac{\partial I}{\partial d}\right) = \frac{\partial}{\partial e}\left(I_y x_i^1\right) = I_{ye} x_i^1,$$
$$\frac{\partial^2 I}{\partial d \partial f} = \frac{\partial}{\partial f}\left(\frac{\partial I}{\partial d}\right) = \frac{\partial}{\partial f}\left(I_y x_i^1\right) = I_{yf} x_i^1,$$
$$\frac{\partial^2 I}{\partial e \partial e} = \frac{\partial}{\partial e}\left(\frac{\partial I}{\partial e}\right) = \frac{\partial}{\partial e}\left(I_y y_i^1\right) = I_{ye} y_i^1,$$
$$\frac{\partial^2 I}{\partial e \partial f} = \frac{\partial}{\partial f}\left(\frac{\partial I}{\partial e}\right) = \frac{\partial}{\partial f}\left(I_y y_i^1\right) = I_{yf} y_i^1,$$
$$\frac{\partial^2 I}{\partial f \partial f} = \frac{\partial}{\partial f}\left(\frac{\partial I}{\partial f}\right) = \frac{\partial}{\partial f}\left(I_y\right) = I_{yf},$$

where

$$I_{xa} = \frac{\partial}{\partial a} I_x = I_{xx} x_1, \quad I_{yb} = \frac{\partial}{\partial a} I_y = I_{yx} x_1, \quad I_{xd} = \frac{\partial}{\partial d} I_x = I_{xy} x_1$$
$$I_{yd} = \frac{\partial}{\partial d} I_y = I_{yy} x_1, \quad I_{xb} = \frac{\partial}{\partial b} I_x = I_{xx} y_1, \quad I_{yb} = \frac{\partial}{\partial b} I_y = I_{xy} y_1,$$
$$I_{xe} = \frac{\partial}{\partial e} I_x = I_{xy} y_1, \quad I_{ye} = \frac{\partial}{\partial e} I_y = I_{yy} y_1, \quad I_{xc} = \frac{\partial}{\partial c} I_x = I_{xx},$$
$$I_{yc} = \frac{\partial}{\partial c} I_y = I_{xy}, \quad I_{xf} = \frac{\partial}{\partial f} I_x = I_{xy}, \quad I_{yf} = \frac{\partial}{\partial f} I_y = I_{yy}.$$

References

[1] B.D. Lucas, T. Kanade, An iterative image registration technique with an application to stereo vision, in: International Joint Conference on Artificial Intelligence, 1981, pp. 674–679.

[2] T. Brox, A. Bruhn, N. Papenberg, J. Weickert, High accuracy optical flow estimation based on a theory for warping, in: Proceedings of the ECCV'04, vol. 3024, 2004, pp. 25–36.

[3] M. Black, A. Jepson, Eigentracking: robust matching and tracking of articulated objects using a view-based representation, in: European Conference on Computer Vision, 1996, pp. 329–342.

[4] M. LaCascia, S. Sclaroff, Fast, reliable head tracking under varying illumination, in: Proceedings, IEEE Conference on Computer Vision and Pattern Recognition, 1999, pp. 604–610.

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

[6] A. Tekalp, Digital Video Processing, Prentice Hall, 1995.

[7] F. Dufaux, J. Konrad, Efficient, robust, and fast global motion estimation for video coding, IEEE Transactions on Image Processing 9 (3) (2000) 497–501.

[8] J. Bergen, P. Anandan, K. Hanna, R. Hingorani, Hierarchical model-based motion estimation, in: European Conference on Computer Vision, 1992, pp. 237–252.

[9] R. Fablet, P. Bouthemy, Motion recognition using non parametric image motion models estimated from temporal and multiscale cooccurrence statistics, IEEE Transactions on Pattern Analysis and Machine Intelligence 25 (12) (2003) 1619–1624.

[10] G. Stein, A. Shashua, Model-based brightness constraints: on direct estimation of structure and motion, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (9) (2000) 992–1015.

[11] J. Barron, D. Fleet, S. Beauchemin, Performance of optical flow techniques, International Journal of Computer Vision 12 (1) (1994) 43–77.

[12] B. Horn, B. Schunck, Determining optical flow, Artificial Intelligence 17 (1981) 185–203.

[13] S. Baker and I. Matthews, Equivalence and efficiency of image alignment algorithms, in: Proceedings, IEEE Conference on Computer Vision and Pattern Recognition, 2001.

[14] R. Szeliski, Image mosaicking for tele-reality applications, in: Proceedings of IEEE Workshop on Applications of Computer Vision, 1994, pp. 44–53.

[15] M. Irani, P. Anandan, Vision Algorithms: Theory and Practice: International Workshop on Vision Algorithms, Corfu, Greece, September 1999, in: Proceedings. Springer–Verlag, 1999, Chapter. All about direct methods, pp. 267–278.

[16] D. Cremers, S. Soatto, Variational space-time motion segmentation, in: ICCV, 2003, pp. 886–893.

[17] J.-M. Odobez, P. Bouthemy, Robust multiresolution estimation of parametric motion models, Journal of Visual Communication and Image Representation 6 (4) (1995) 348–365.

[18] D.-G. Sim, R.-H. Park, Robust reweighted map motion estimation, IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (4) (1998) 353–365.
[19] S. Peleg, B. Rousso, A. Rav-Acha, A. Zomet, Mosaicing on adaptive manifolds, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (10) (2000) 1144–1154.
[20] A. Shashua, Y. Wexler, Q-warping: direct computation of quadratic reference surfaces, IEEE Transactions on Pattern Analysis and Machine Intelligence 23 (8) (2001) 920–925.
[21] F. Candocia, Simultaneous homographic and comparametric alignment of multiple exposure-adjusted pictures of the same scene, IEEE Transactions on Image Processing 12 (12) (2003) 1485–1494.
[22] Y. Altunbasak, R. Mersereau, A. Patti, A fast parametric motion estimation algorithm with illumination and lens distortion correction, IEEE Transactions on Image Processing 12 (4) (2003) 395–408.
[23] Y. Keller, A. Averbuch, Fast gradient methods based on global motion estimation for video compression, IEEE Transactions Circuits and Systems for Video Technology 13 (4) (2003) 300–309.
[24] F.M. Candocia, Jointly registering images in domain and range by piecewise linear comparametric analysis, IEEE Transactions on Image Processing 12 (4) (2003) 409–419.
[25] R. Szeliski, Video mosaics for virtual environments, IEEE Computer Graphics and Applications 16 (1996) 22–30.
[26] C.D. Kuglin, D.C. Hines, The phase correlation image alignment method, IEEE Conference on Cybernetics and Society (1975) 163–165.
[27] S. Reddy, B.N. Chatterji, An FFT-based technique for translation, rotation, and scale-invariant image registration, IEEE Transactions on Image Processing 3 (8) (1996) 1266–1270.
[28] V. Ferrari, T. Tuytelaars, L.V. Gool, Wide-baseline multiple-view correspondences, in: Proceedings, IEEE Conference on Computer Vision and Pattern Recognition, vol. 1, 2003, pp. 1–718 to 1–725.
[29] F. Schaffalitzky, A. Zisserman, Viewpoint invariant texture matching and wide baseline stereo, in: Proceedings of the 8th International Conference on Computer Vision, Vancouver, Canada, 2001, pp. 636–643.
[30] Y. Keller, A. Averbuch, Fast motion estimation using bi-directional gradient methods, IEEE Transactions on Image Processing 13 (8) (2004) 1042–1054.
[31] S. Mann, R. Picard, Virtual bellows: constructing high quality stills from video, IEEE International Conference Image Processing, Austin, TX, 1994.
[32] [Online]. Available from: <http://mathworld.wolfram.com/Lagrange-Remainder.html/>.
[33] P. Gill, Practical Optimization, Academic Press, 1982.
[34] S. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University Press, 2004.
[35] M. Irani, S. Peleg, Motion analysis for image enhancement: resolution, occlusion and transparency, Journal of Visual Communication and Image Representation 4 (4) (1993) 324–335.
[36] J. Matas, 0. Chum, M. Urban, T. Pajdla, Robust wide baseline stereo from maximally stable extremal regions, in: Proceedings of the British Machine Vision Conference (BMVC), London, GB, 2002, pp. 384–393.
[37] J. Stewart, Multivariable Calculus, fifth ed., Brooks/Cole Pub Co., 2003.