Fast motion estimation using bidirectional gradient methods

A. Averbuch[1], Y. Keller[1,2]

[1]School of Computer Science, Tel Aviv University, Tel Aviv 69978
Israel

[2]Dept. of Electrical Engineering Systems
Tel-Aviv University
Tel-Aviv 69978, Israel

Abstract

Gradient based motion estimation techniques (GM) are considered to be in the heart of state-of-the-art registration algorithms [3], being able to account for both pixel and subpixel registration and to handle various motion models (translation, rotation, affine, projective). These methods estimate the motion between two images based on the local changes in the image intensities while assuming image smoothness. This paper offers two main contributions: (i) Enhancement of the GM technique by introducing two new bidirectional formulations of the GM. This improves the convergence properties for large motions. (ii) We present an analytical convergence analysis of the GM and its properties. Experimental results demonstrate the applicability of these algorithms to real images.

Keywords: Global motion estimation, Sub-pixel registration, Gradient methods, image alignment

EDICS Category={2-ANAL, 2-MOTD}

1 Introduction

Image registration plays a vital role in many image processing applications such as video compression [12, 15], video enhancement [10] and scene representation [1, 4, 11]. It has drawn a significant research attention. A comprehensive comparative survey by Barron et. al. [2] found the family of gradient-based motion estimation methods (GM), originally proposed by Horn and Schunck [3], to perform especially well. The purpose of the GM algorithm is to estimate the parameters vector $\underline{P}$ associated with the *parametric image registration* problem: starting from pure global translation, image plane rotation, 2D affine, and pseudo-projective (8-parameter flow). These models have been

used extensively and are estimated directly from image spatio-temporal derivatives using coarse-to-fine estimation via Gaussian pyramids (multiscale). These methods search for the best parametric geometric transform that minimizes the square of changes between image intensities (SSD) over the whole image. Several formulations of the gradient methods which differ on the way the motion parametrs are updated, either by incrementing the motion parmeters [13] or incrementing the warp matrix [4]. An updated comprehensive description of these methods was given in [17].

Let $I_1(x,y)$ and $I_2(x,y)$ be the images, which have some common overlap as described in Fig. 1. Then each pixel in their common support satisfies:

$$I_1(x,y) = I_2\left(\widetilde{x}(x,y,\underline{P}),\widetilde{y}(x,y,\underline{P})\right) \tag{1.1}$$

where the structure of the parameters vector $\underline{P}$ depends on the type of the estimated motion model.

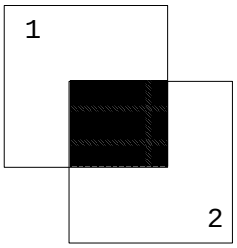


Figure 1: Image translation

Gathering and solving all the equations associated with pixels in the mutual support ($\underline{P}$ is assumed to be constant over the whole mutual area), estimates the *global motion* between the images [4], thus gaining robustness due to very highly over-constrained linear systems (each pixel contributes a linear constraint). Gathering the equations related to small image patches estimates *local motion* [13]. Equation [1.1] is solved using non-linear iterative optimization techniques such as Gauss-Newton [6] and Levenberg-Marquardt (LM) [4] described in section 2. A critical implementation issue concerning the GM is the convergence range and the rate of convergence while estimating large image motions: as the estimated motion becomes larger, the convergence rate decreases and the GM may diverge to a local minima. A possible solution is to bootstrap the motion estimation process with a different motion estimation algorithm [14, 15] which is robust to large motions. In order to improve the convergence of the GM we analyze it using optimization methodology in section 3. The analysis of the GM convergence leads to a new robust constructive algorithm that achieves faster convergence through symmetric and non-symmetric bidirectional formulation, presented in section 4. These properties are experimentally verified in section 5.

2 Gradient method based motion estimation

GM methodology [17] estimates the motion parameters $\underline{P}$ by minimizing the intensity discrepancies between the images $I_1(x,y)$ and $I_2(x,y)$ described in Fig. 1:

$$\underline{P}^* = \arg\min_{\underline{P}} \left\{ \sum_{(x_1,y_1)\in S} \left(I_1\left(x_i^{(1)},y_i^{(1)}\right) - I_2\left(f\left(x_i^{(1)},y_i^{(1)},\underline{P}\right), g\left(x_i^{(1)},y_i^{(1)},\underline{P}\right)\right)\right)^2 \right\} \tag{2.1}$$

where

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

and S is the set of coordinates of pixels that are common to I_1 and I_2 in I_1's coordinates and $\underline{P}$ is the estimated parameter vector. Next we follow the formulation of [1, 4] by solving Eq. [2.1] using non-linear iterative optimization techniques such as Gauss-Newton [6] and Levenberg-Marquardt (LM) [4, 6]. The basic GM formulation and the iterative refinement stage are described in sections 2.1 and 2.2, respectively. These are embedded in a multi-resolution scheme described in section 2.3.

2.1 Basic GM formulation

The non-linear optimization of Eq. [2.1], is conducted via a linearization procedure, which is based on a pixel-wise first order Taylor expansion of I_1 in terms of I_2 as a function of the parameters vector $\underline{P}$:

$$I_1(x_i^{(1)}, y_i^{(1)}) = I_2(x_i^{(2)}, y_i^{(2)}) + \sum_{P_i\in\underline{P}} \frac{\partial I_2(x_i^{(2)}, y_i^{(2)})}{\partial P_i} P_i \tag{2.3}$$

where $\frac{\partial I_2(x_i^{(2)}, y_i^{(2)})}{\partial P_i}$ is the partial spatial derivative, $(x_i^{(1)}, y_i^{(1)})$ and $(x_i^{(2)}, y_i^{(2)})$ are the ith common pixel in I_1 and I_2, respectively.

By gathering the pixel-wise equations we formulate the system

$$\underline{\underline{H}}\,\underline{P} = \underline{I_t} \tag{2.4}$$

where

$$\begin{aligned} \underline{\underline{H}}_{i,j} &= \frac{\partial I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial P_j} \\ &= \frac{\partial I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial x_2} \frac{\partial f\left(x_i^{(1)},y_i^{(1)},\underline{P}\right)}{\partial P_j} + \frac{\partial I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial y_2} \frac{\partial g\left(x_i^{(1)},y_i^{(1)},\underline{P}\right)}{\partial P_j} \end{aligned} \tag{2.5}$$

$$\underline{I_t}_i = I_1\left(x_i^{(1)},y_i^{(1)}\right) - I_2\left(x_i^{(2)},y_i^{(2)}\right). \tag{2.6}$$

Equation [2.5] is formulated using the chain rule. Equation [2.4] can be solved using regular least square [6]:

$$\underline{P} = \left(\underline{\underline{H}}^t\underline{\underline{H}}\right)^{-1}\underline{\underline{H}}^t\underline{I_t} \tag{2.7}$$

where $\underline{\underline{H}}^t$ is the transpose of $\underline{\underline{H}}$. The expressions for $\underline{\underline{H}}^t\underline{\underline{H}}$ and $\underline{\underline{H}}^t\underline{I_t}$ can be derived analytically by direct calculation:

$$\left(\underline{\underline{H}}^t\underline{\underline{H}}\right)_{k,j} = \sum_i \frac{\partial I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial P_k}\frac{\partial I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial P_j} \tag{2.8}$$

$$\left(\underline{\underline{H}}^t\underline{I_t}\right)_k = \sum_i \frac{\partial I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial P_k}\underline{I_t}_i. \tag{2.9}$$

2.1.1 Algorithm flow

The basic GM iteration, which is marked as "*Single Iteration*" in Fig. 2, is as follows:

1. The matrix $\underline{\underline{H}}^t\underline{\underline{H}}$ and vector $\underline{\underline{H}}^t\underline{I_t}$ are computed using Eq. 2.8 and Eq. 2.9, respectively.
2. Eq. 2.7 is solved using singular value decomposition [6].
3. The GM returns $\underline{\mathrm{P}}$ as its output (result).

2.2 Iterative solution of the gradient methods

Denote:

$\underline{P}_0$- an initial estimated solution of Eq. [2.1] given as input, such that $Warp\left(I_2,\underline{P}_0\right) \approx I_1$

$\underline{P}_n$- the estimated solution after $n = 1,\ldots$ iterations

Then, the nth iteration of the motion estimation algorithm becomes:

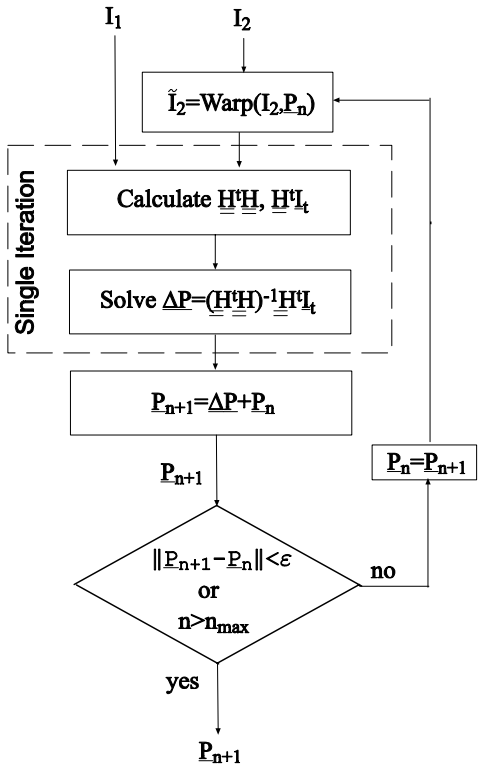


Figure 2: Block diagram of the basic and iterative GM formulations. For $n = 0$, $\underline{P}_0$ is given as an initial guess and $\underline{\Delta P}$ is the iterative update after each iteration.

1. The input image I_2 is wrapped towards I_1 using the current estimate $\underline{P}_n$ and it is stored in $\widetilde{I}_2$ $n \geq 0$. For $n = 0$ $\underline{P}_0$ is given as input.

2. I_1 and $\widetilde{I}_2$ are used as input images to the procedure described in section 2.1.

3. The result of step 2 - $\underline{\Delta P}$, is used to update the solution:

$$\underline{P}_{n+1} = \underline{\Delta P} + \underline{P}_n \qquad n \geq 0$$

4. Go back to step 1 until one of the following stopping criteria is met:

   (a) At most $N_{\max}$ iterations are performed

   or

   (b) The process is stopped if the translation parameters within the updated term $\underline{\Delta P}$ reaches a predetermined threshold.

2.3 Gradient methods with multiscale scheme

In order to improve the robustness and reduce the complexity of the algorithm, the iterative process is embedded in a coarse-to-fine multiscale formulation. The robustness analysis is given in section 3.2. Next we describe the coarse-to-fine formulation. A thorough description can be found in [1]:

1. The input images I_1 and I_2 are smoothed. Our experience shows that a separable averaging filter is suitable for this task.
2. The input images I_1 and I_2 are downsampled through multiscale decomposition, until a minimal size of their mutual area is reached. The minimal mutual area size depends upon the estimated motion model, while the resolution step depends upon the motion estimation accuracy at each resolution level.
3. Starting with the coarsest scale, the initial estimate $\underline{P}_0$ is used to bootstrap the iterative refinement algorithm described in section 2.2.
4. The result of the iterative refinement from coarse-to-fine (step 2) is recursively repeated until the original image size is reached.

3 Convergence analysis of gradient methods

In order to analyze the convergence properties of the GM algorithm, we examine the convergence properties of the general Gauss-Newton algorithm in Appendix A. These results are interpreted in the context of the GM algorithm in section 3.1.

3.1 General convergence analysis of the Gauss-Newton algorithm

The analysis in Appendix A shows that the convergence of the Gauss-Newton algorithm can be divided into two distinct phases as it is described by Eq. [7.14] in Appendix A:

$$\|\varepsilon_{k+1}\| \leq C_1 \cdot \|\varepsilon_k\| + C_2 \cdot \|\varepsilon_k\|^2 \tag{3.1}$$

where:

$$\begin{aligned} C_1 &= \left\| \left[A\left(\underline{x}_k\right) A^T\left(\underline{x}_k\right) \right]^{-1} \sum_{n=1}^{m} \frac{\partial r_n^2\left(x_k\right)}{\partial x} r_n\left(\underline{x}_k\right) \right\| \\ C_2 &= \left\| \left[A\left(\underline{x}_k\right) A^T\left(\underline{x}_k\right) \right]^{-1} \right\| \end{aligned}$$

ε_k	is the parameters estimation error after iteration k
$r_n(\underline{x}_k)$	is the error associated with the nth equation at iteration k
$A\left(\underline{x}_k\right)$	is the Jacobian matrix at iteration k.

By rearranging the GM formulations developed in section 2, we interpret these expressions in the context of the GM formulation. C_1 and C_2 will be denoted C_1^{GM} and C_2^{GM} respectively.

$$r_n(\underline{x}_k) = I_1\left(x_i^{(1)},y_i^{(1)}\right) - I_2\left(f\left(x_i^{(1)},y_i^{(1)},\underline{P}\right), g\left(x_i^{(1)},y_i^{(1)},\underline{P}\right)\right) \tag{3.2}$$

where the vector parameters $\underline{x}$ identifies with $\underline{P}$:

$$A_{i,k} = \frac{\partial r_n(\underline{x}_k)}{\partial x_k} = \frac{\partial I_2(x_i^{(2)},y_i^{(2)})}{\partial P_k} \tag{3.3}$$

and the equation index n is identified with the GM index i, since we have one equation per common pixel. Thus, $A(\underline{x}_k)$ is identified with the matrix $\underline{\underline{H}}$ defined in Eq. [2.5] and the second derivative $\frac{\partial^2 r_n(\underline{x}_k)}{\partial x_k^2}$ is given by:

$$\frac{\partial^2 r_n(\underline{x}_k)}{\partial x_k^2} = \frac{\partial^2 I_2(x_i^{(2)},y_i^{(2)})}{\partial {P_k}^2}. \tag{3.4}$$

Therefore, we have

$$C_1^{GM} = \left\| \left[\frac{\partial I_2}{\partial \underline{P}} \cdot \frac{\partial I_2}{\partial \underline{P}}^T \right]^{-1} \cdot \sum_{i=1}^{m} \frac{\partial^2 I_2(x_i^{(2)},y_i^{(2)})}{\partial {P_k}^2} r_n\left(\underline{x}_k\right) \right\| \tag{3.5}$$

$$C_2^{GM} = \left\| \left[\frac{\partial I_2}{\partial \underline{P}} \cdot \frac{\partial I_2}{\partial \underline{P}}^T \right]^{-1} \right\| \tag{3.6}$$

The basic GM equation (Eq. [2.3]) is solved by the Gauss-Newton algorithm using the LS formulation given in Eq. [3.2]. The error associated with each equation is the truncation error of the first order Taylor expansion [6]:

$$\varepsilon_{GM}\left(x_i^{(2)}, y_i^{(2)}, \underline{P}\right) = \frac{1}{2} \sum_{P_i \in \underline{P}} \frac{\partial^2 I_2(x_i^{(2)},y_i^{(2)})}{\partial P_i^2} \left(P_i - P_i^*\right)^2 \tag{3.7}$$

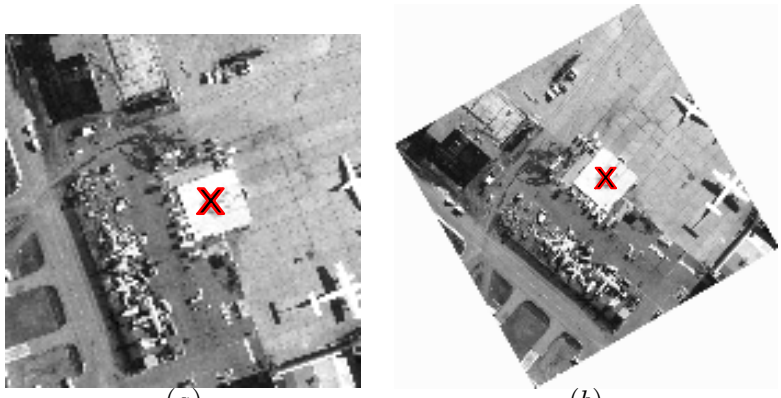
where $\underline{\mathrm{P}}^*$ is the optimal solution of the optimization problem.

In the GM setup the sum of second partial derivatives $\frac{\partial^2 I_2(x_i^{(2)},y_i^{(2)})}{\partial P_i^2}$ does not strongly depend on the motion parameters vector $\underline{P}$ and the magnitude of C_1^{GM} is dominated by $\|\underline{P}-\underline{P}^*\|$, hence, for large motions $\|\underline{P}-\underline{P}^*\| \gg 0$ and $C_1^{GM} \gg 0$ and the error decay rate becomes linear rather than quadratic. Therefore, the convergence of the GM algorithm can be divided into two phases:

Initialization phase: In the first iterations we have $\|\underline{P}-\underline{P}^*\| \gg 0$ and $C_1^{GM} \gg 0$, therefore the convergence rate is linear.

Convergence phase: near the solution $\|\underline{P}-\underline{P}^*\| \to 0$ we have $C_1^{GM} \to 0$, and the convergence rate is quadratic according to C_2^{GM}, where C_2^{GM} is a function of the image properties.

This analysis was experimentally verified by registering the images which are shown in Fig. 3: the "Airfield" image and a 30° rotated version of it. The registration results are presented in Fig. 4. We have two distinct convergence phases. We start with a low-rate convergence corresponding to the linear convergence , after a cross-over point located at $n = 170$, we encounter the quadratic convergence phase. Using Eq. [7.18] we present the image alignment error instead of the parameters' error.



(a) (b)

Figure 3: Test of the Gauss-Newton convergence - (a) Original "Airfield" image (b) The "Airfield" image which was rotated by 30° using bilinear interpolation. The red X marks the initial estimate of the motion given as translation.

3.2 Multiresolution GM scheme

The relation between the pyramidal GM scheme and the convergence properties of the GM was studied by Burt et-al [18] in the frequency domain for pure translations. By examining the error associated with the translation coefficients, the multiscale scheme was proved to decrease the error term in Eq. [3.7] and improve the convergence rate.

Denote by $\underline{\varepsilon_{trans}}\,(s)$ - the error associated with the translation parameters (dx_s, dy_s) at a resolution scale s:

$$\underline{\varepsilon_{trans}}\,(s) = \begin{bmatrix} dx_s - dx_s^* \\ dy_s - dy_s^* \end{bmatrix} \tag{3.8}$$

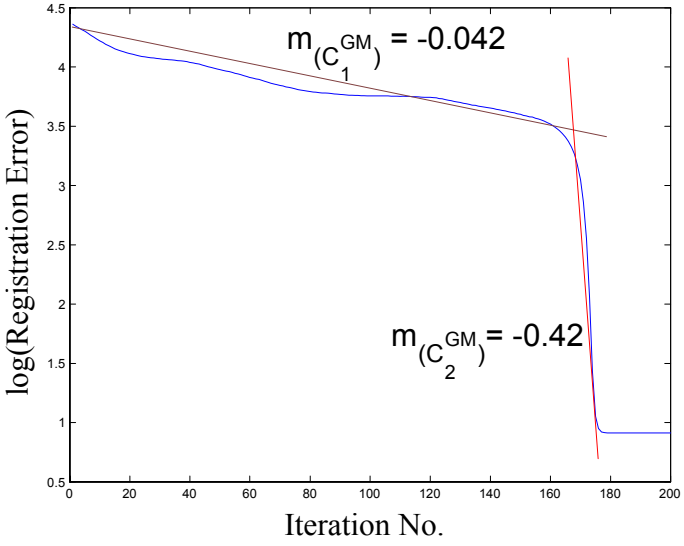


Figure 4: The convergence process is divided into two phases: the first is related to large motion estimation characterized by a low convergence rate $m_{(C_1)}$, the second is related to small motion having a high convergence rate $m_{(C_2)}$. The cross-over region is located after iteration $n \approx 170$.

where s is the image scaling factor, dx_s^* and dy_s^* are the optimal values of the translation parameters in scale s. Then, by scaling down the images from scale s_1 to scale s_2 ($s_2 > s_1$) we get:

$$\begin{aligned} dx_{s_2} &= dx_{s_1} \cdot \tfrac{s_1}{s_2} \\ dy_{s_2} &= dy_{s_1} \cdot \tfrac{s_1}{s_2} \end{aligned} \tag{3.9}$$

and the associated error becomes

$$\underline{\varepsilon_{trans}}\,(s_2)^2 = \begin{pmatrix} dx_{s_2} - dx_{s_2}^* \\ dy_{s_2} - dy_{s_2}^* \end{pmatrix}^2 = \begin{pmatrix} dx_{s_1} \cdot \frac{s_1}{s_2} - dx_{s_1}^* \cdot \frac{s_1}{s_2} \\ dy_{s_1} \cdot \frac{s_1}{s_2} - dy_{s_1}^* \cdot \frac{s_1}{s_2} \end{pmatrix}^2 = \underline{\varepsilon_{trans}}\,(s_1)^2 \cdot \underbrace{\left(\frac{s_1}{s_2}\right)^2}_{<1}. \tag{3.10}$$

Hence, the error associated with the translation error is decreased by a factor of $\frac{s_1}{s_2} < 1$. Inserting Eq. [3.10] into Eq. [3.7] we get:

$$\begin{aligned} \varepsilon_{GM}\left(x_i^{(2)}, y_i^{(2)}, [dx, dy], s_2\right) &= \frac{1}{2} \sum_{P_i \in [dx,dy]} \frac{\partial^2 I_2(x_i^{(2)}, y_i^{(2)})}{\partial P_i^2} \underline{\varepsilon_{trans}}\,(s_2)^2 \\ &= \frac{1}{2} \sum_{P_i \in [dx,dy]} \frac{\partial^2 I_2(x_i^{(2)}, y_i^{(2)})}{\partial P_i^2} \left(\underline{\varepsilon_{trans}}\,(s_1)^2 \cdot \left(\frac{s_1}{s_2}\right)^2 \right) \\ &= \varepsilon_{GM}\left(x_i^{(2)}, y_i^{(2)}, [dx, dy], s_1\right) \cdot \underbrace{\left(\frac{s_1}{s_2}\right)^2}_{<1} \end{aligned} \tag{3.11}$$

Therefore, by using a dyadic pyramid, the truncation error of the Taylor approximation, related to the translation parameters (Eq. [3.11]) is decreased by a factor of 4 in each increase of the pyramid's scale. While the approximation error related to the motion parameters which are scale-invariant, such as scale and shear, is not reduced since relative scale changes are invariant to identical scale changes of both images. Hence, multiresolution schemes do not improve the convergence properties when the motion is dominated by scale and shear. This method can achieve higher convergence rate if instead of dyadic division, we use bigger scale factors.

3.3 Dominant motion locking

Burt et. al. [18] used a frequency analysis to show that the coarse-to-fine refinement process allows the GM to lock on a single dominant motion even when multiple motions are present. This property is essential for most applications which are based on image registration [4, 10, 11]. We utilize the method presented in section 3 to provide an optimization based analysis of this property by studying the error associated with objects which perform dominant and non-dominant motions. This analysis extends the results of [18] by being applicable to general motion models - parametric and non-parametric.

Notation:

S	The set of pixels that are common to I_1 and I_2
S_{Dom}	A subset of S. This set of pixels that are common to I_1 and I_2, whose motion is the *dominant motion*, was defined above
S_{NonD}	A subset of S. This set of pixels that are common to I_1 and I_2, whose motion is not the *dominant motion*

By permuting the rows of Eq. [2.4] according to the ith pixel's relation to either S_{Dom} or S_{NonD} we get:

$$\underline{\underline{\widetilde{H}}}\,\underline{P} = \begin{bmatrix} \underline{\underline{\widetilde{H}}}_{Dom} \\ \underline{\underline{\widetilde{H}}}_{NonD} \end{bmatrix} \underline{P} = \begin{bmatrix} \underline{I}_t^{Dom} \\ \underline{I}_t^{NonD} \end{bmatrix} = \underline{\widetilde{I}_t} \tag{3.12}$$

where

$$\underline{\underline{\widetilde{H}}}_{Dom}\underline{P} = \underline{I_t}^{Dom} \tag{3.13}$$

are the equations related to dominant motion, and

$$\underline{\underline{\widetilde{H}}}_{NonD}\underline{P} = \underline{I_t}^{NonD} \tag{3.14}$$

are the equations related to non-dominant motion. As the GM algorithm converges to the dominant motion, the term $\underline{I}_t^{NonD}$ becomes the difference of uncorrelated pixels. Therefore, $\underline{I}_t^{NonD}$ can be

modelled as a uniformly distributed random variable with zero mean

$$E\left\{\underline{I_t}^{NonD}\right\}=0 \tag{3.15}$$

This is a *landslide* type phenomenon: as the iterative solution $\underline{P}_n$ gets closer to the dominant motion's true parameters $\underline{P}^{Dom}$, the non-dominant pixels become more and more uncorrelated. Inserting Eq. [3.15] into Eq. [2.7] we have

$$\begin{aligned} E\left\{\underline{\Delta P}\right\} &= E\left\{\left(\underline{\underline{\widetilde{H}}}^t\underline{\underline{\widetilde{H}}}\right)^{-1}\underline{\underline{\widetilde{H}}}^t\underline{\widetilde{I}_t}\right\} \\ &= E\left\{\left(\underline{\underline{\widetilde{H}}}^t\underline{\underline{\widetilde{H}}}\right)^{-1}\underline{\underline{\widetilde{H}}}^t\begin{bmatrix}\underline{I_t}^{Dom}\\ \underline{0}\end{bmatrix}\right\}+\underbrace{E\left\{\left(\underline{\underline{\widetilde{H}}}^t\underline{\underline{\widetilde{H}}}\right)^{-1}\underline{\underline{\widetilde{H}}}^t\begin{bmatrix}\underline{0}\\ \underline{I_t}^{NonD}\end{bmatrix}\right\}}_{=0} \\ &= \underline{P}^{Dom}. \end{aligned} \tag{3.16}$$

Thus, the non-dominant outliers are automatically rejected. We conclude that the GM is a non-biased estimator of the dominant motion parameters $\underline{P}^{Dom}$, where the variance of the estimation $Var\left(\underline{P}^{Dom}\right)$ depends on the ratio between dominant and non-dominant pixels.

4 Improved GM convergence using bidirectional formulations

In order to improve the convergence properties of the GM algorithm, we consider the registration of two one-dimensional signals $I_1(x)$ and $I_2(x)$ using the translation motion model:

$$I_1\left(x_i^{(1)}\right)=I_2\left(x_i^{(2)}\right) \tag{4.1}$$

w here $x_i^{(1)}$ and $x_i^{(2)}$ are the current estimates of the coordinate of the ith sample common to I_1 and I_2 satisfying:

$$x_i^{(1)}=x_i^{(2)}+\Delta x. \tag{4.2}$$

Similar to section 2, Eq. [4.1] is solved by expanding I_2 in a first order Taylor expansion and solving for Δx:

$$I_1\left(x_i^{(1)}\right)=I_2\left(x_i^{(2)}\right)+\frac{\partial I_2\left(x_i^{(2)}\right)}{\partial x}\cdot\Delta x. \tag{4.3}$$

as illustrated in Fig. 5(a). This point-wise expansion causes an error which is estimated by Eq. [3.7]:

$$\varepsilon_{GM}\left(x_i^{(2)},\Delta x\right)=\frac{1}{2}\frac{\partial^2 I_2\left(\widetilde{x}^{(2)}\right)}{\partial x^2}\Delta x_i^2,\ \ \widetilde{x}\in[0,\Delta x]\,. \tag{4.4}$$

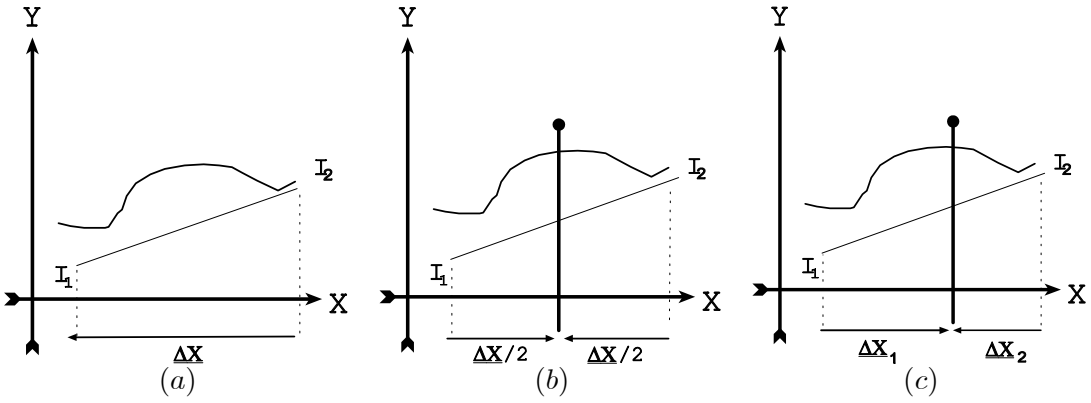


Figure 5: 1D illustration of various GM techniques: (a) Regular GM: pixels in I_1 are approximate by pixels in I_2 over the interval Δx. (b) Symmetric GM (SGM): pixels in the middle of the interval between I_1 and I_2 ($^{\Delta X}/_2$) are approximated by common pixels in I_1 and I_2. (c) Bidirectional GM (BDGM): pixels in the interval between I_1 and I_2 are approximated by common pixels in I_1 and I_2. The equilibrium point is chosen optimally to minimize the approximation error.

In order to lower the estimation error $\varepsilon_{GM}\left(x_i^{(2)}, \Delta x\right)$, Eq. [4.1] is reformulated symmetrically in respect to Δx according to Fig. 5(b):

$$I_1\left(x_i^{(1)} - {}^{\Delta X}/_2\right) = I_2\left(x_i^{(2)} + {}^{\Delta X}/_2\right) \tag{4.5}$$

expanding both sides using Taylor expansion we have

$$I_1\left(x_i^{(1)}\right) - \frac{\partial I_1\left(x_i^{(1)}\right)}{\partial x}\frac{\Delta x}{2} + \varepsilon_{GM}\left(x_i^{(1)}, {}^{\Delta X}/_2\right) = I_2\left(x_i^{(2)}\right) + \frac{\partial I_2\left(x_i^{(2)}\right)}{\partial x}\frac{\Delta x}{2} + \varepsilon_{GM}\left(x_i^{(2)}, {}^{\Delta X}/_2\right). \tag{4.6}$$

$$\underbrace{I_2\left(x_i^{(2)}\right) - I_1\left(x_i^{(1)}\right)}_{=-I_t} + \left(\frac{\partial I_2\left(x_i^{(2)}\right)}{\partial x} + \frac{\partial I_1\left(x_i^{(1)}\right)}{\partial x}\right)\frac{\Delta x}{2} + \underbrace{\varepsilon_{GM}\left(x_i^{(2)}, {}^{\Delta X}/_2\right) - \varepsilon_{GM}\left(x_i^{(1)}, {}^{\Delta X}/_2\right)}_{=\varepsilon_{SGM}\left(x_i^{(2)}, \Delta x\right)} = 0. \tag{4.7}$$

Next we derive an upper bound of the error associated with Eq. [4.6]:

$$\varepsilon_{SGM}\left(x_i^{(2)}, \Delta x\right) \leq 2\varepsilon_{GM}\left(x_i^{(2)}, {}^{\Delta X}/_2\right). \tag{4.8}$$

The error term is quadratic in Δx, then by comparing ε_{SGM} to the regular GM error ε_{GM} of Eq. [4.4] we get:

$$\varepsilon_{GM}\left(x_i^{(2)},{}^{\Delta X}\!/_2\right) = \frac{1}{4}\varepsilon_{GM}\left(x_i^{(2)},\Delta x\right). \tag{4.9}$$

$$\begin{aligned}\varepsilon_{SGM}\left(x_i^{(2)},\Delta x\right) &= 2\varepsilon_{GM}\left(x_i^{(2)},{}^{\Delta X}\!/_2\right)\\ &= \frac{1}{2}\varepsilon_{GM}\left(x_i^{(2)},\Delta x\right).\end{aligned}$$

A better approximation error analysis can be derived by expanding $\varepsilon_{SGM}\left(x_i^{(2)},\Delta x\right)$:

$$\begin{aligned}\varepsilon_{GM}\left(x_i^{(2)},{}^{\Delta X}\!/_2\right) - \varepsilon_{GM}\left(x_i^{(1)},{}^{\Delta X}\!/_2\right) &= \frac{1}{4}\varepsilon_{GM}\left(x_i^{(2)},\Delta x\right) - \frac{1}{4}\varepsilon_{GM}\left(x_i^{(1)},\Delta x\right)\\ &= \frac{1}{8}\Delta x_i^2\left(\frac{\partial^2 I_2(\widetilde{x}^{(2)})}{\partial x^2} - \frac{\partial^2 I_1(\widetilde{x}^{(1)})}{\partial x^2}\right)\\ &\approx \frac{1}{8}\Delta x_i^2\left(\frac{\partial^3 I_2(\widetilde{x})}{\partial x^3}\Delta x\right)\\ &= \frac{\partial^3 I_2(\widetilde{x})}{\partial x^3}\frac{\Delta x_i^3}{8}.\end{aligned} \tag{4.10}$$

The smaller the linearization error ε_{SGM}, the better is the convergence rate. If $\varepsilon_{SGM} = 0$ Eq. [4.3] converges in a single iteration.

In order to further decrease the linearization error we allow the interval $[0\ldots\Delta x]$ to be partitioned optimally using the following formulation that is based on Fig. 5(c):

$$I_1(x_i^{(1)} - \Delta x_1) = I_2(x_i^{(2)} - \Delta x_2) \tag{4.11}$$

Using Eq. [4.4] and a Taylor expansion of Eq. [4.11] we have

$$\begin{aligned}I_1\left(x_i^{(1)}\right) - \frac{\partial I_1\left(x_i^{(1)}\right)}{\partial x}\Delta x_1 + \varepsilon_{GM}\left(x_i^{(1)},\Delta x_1\right) =\\ I_2\left(x_i^{(2)}\right) + \frac{\partial I_2(x_i^{(2)})}{\partial x}\Delta x_2 + \varepsilon_{GM}\left(x_i^{(2)},\Delta x_2\right).\end{aligned} \tag{4.12}$$

$$\begin{aligned}I_2\left(x_i^{(2)}\right) - I_1\left(x_i^{(1)}\right) + \frac{\partial I_2\left(x_i^{(2)}\right)}{\partial x}\Delta x_1 + \frac{\partial I_1\left(x_i^{(1)}\right)}{\partial x}\Delta x_2\\ \underbrace{+\,\varepsilon_{GM}\left(x_i^{(2)},\Delta x_2\right) - \varepsilon_{GM}\left(x_i^{(1)},\Delta x_1\right)}_{=\varepsilon_{BDGM}\left(x_i^{(1)},x_i^{(2)},\Delta x_1,\Delta x_2\right)} = 0\end{aligned} \tag{4.13}$$

where the motion between $I_2$and I_1 is given by:

$$\Delta x = \Delta x_1 + \Delta x_2 \tag{4.14}$$

and ε_{BDGM} is the error of the Bidirectional Gradient Methods.

Since the solution of Eq. [4.13] minimizes ε_{BDGM} directly, we expect to achieve superior convergence results. Following the analysis presented in Eq. [4.10], we analyze the error term of Eq. [4.13]:

$$\begin{aligned}
2\cdot\varepsilon_{BDGM}\left(x_i^{(1)},x_i^{(2)},\Delta x_1,\Delta x_2\right) &= 2\left(\varepsilon_{GM}\left(x_i^{(2)},\Delta x_2\right)-\varepsilon_{GM}\left(x_i^{(1)},\Delta x_1\right)\right) \qquad (4.15)\\
&= \frac{\partial^2 I_2\left(\widetilde{x}^{(2)}\right)}{\partial x^2}\Delta x_2^2-\frac{\partial^2 I_1\left(\widetilde{x}^{(1)}\right)}{\partial x^2}\Delta x_1^2\\
&= \frac{\partial^2 I_2\left(\widetilde{x}^{(2)}\right)}{\partial x^2}\Delta x_2^2\underbrace{-\frac{\partial^2 I_2\left(\widetilde{x}^{(2)}\right)}{\partial x^2}\Delta x_1^2+\frac{\partial^2 I_2\left(\widetilde{x}^{(2)}\right)}{\partial x^2}\Delta x_1^2}_{=0}+\frac{\partial^2 I_1\left(\widetilde{x}^{(1)}\right)}{\partial x^2}\Delta x_1^2\\
&= \frac{\partial^2 I_2\left(\widetilde{x}^{(2)}\right)}{\partial x^2}\left(\Delta x_2^2-\Delta x_1^2\right)+\Delta x_1^2\left(\frac{\partial^2 I_2\left(\widetilde{x}^{(2)}\right)}{\partial x^2}-\frac{\partial^2 I_1\left(\widetilde{x}^{(1)}\right)}{\partial x^2}\right)\\
&= \frac{\partial^2 I_2\left(\widetilde{x}^{(2)}\right)}{\partial x^2}\left(\Delta x_2-\Delta x_1\right)\left(\Delta x_2+\Delta x_1\right)+\Delta x_1^2\left(\frac{\partial^3 I_2(\widetilde{x})}{\partial^3 x}\Delta x\right)
\end{aligned}$$

Substituting Eq. [4.14] into Eq. [4.15] we have

$$\varepsilon_{BDGM}\left(x_i^{(1)},x_i^{(2)},\Delta x_1,\Delta x_2\right)=\frac{\Delta x}{2}\left(\frac{\partial^2 I_2\left(\widetilde{x}^{(2)}\right)}{\partial x^2}\cdot\left(\Delta x_2-\Delta x_1\right)+\Delta x_1^2\frac{\partial^3 I_2(\widetilde{x})}{\partial x^3}\right) \qquad (4.16)$$

For the symmetric case where $\Delta x_2=\Delta x_1=\frac{\Delta x}{2}$, the first term of Eq. [4.16] vanishes and ε_{BDGM} identifies with ε_{SGM}:

$$\varepsilon_{BDGM}\left(x_i^{(1)},x_i^{(2)},\Delta x_1,\Delta x_2\right)=\varepsilon_{SGM}\left(x_i^{(2)},\Delta x\right)=\frac{\partial^3 I_2(\widetilde{x})}{\partial x^3}\frac{\Delta x^3}{8} \qquad (4.17)$$

An extension into two dimensions with general motion models is given in sections 4.1 and 4.2 for the SGM and BDGM algorithms respectively.

4.1 Symmetric GM (SGM)

In the general 2D case, the SGM is formulated using the motion parameters vector $\underline{P}$ (see Fig. 5(b)):

$$\underline{P}^*=\arg\min_{\underline{P}}\left\{\sum_{(x_1,y_1)\in S}\left(I_2\left(x_i^{(2)},y_i^{(2)},\underline{P}/2\right)-I_1\left(x_i^{(1)},y_i^{(1)},-\underline{P}/2\right)\right)^2\right\}. \qquad (4.18)$$

$$r_i\left(x_i^{(1)},y_i^{(1)},\underline{P}\right)=I_2\left(x_i^{(2)},y_i^{(2)},\underline{P}/2\right)-I_1\left(x_i^{(1)},y_i^{(1)},-\underline{P}/2\right). \qquad (4.19)$$

From Eq. [4.19] we have

$$\frac{\partial r_i\left(x_i^{(1)},y_i^{(1)},\underline{P}\right)}{\partial\underline{P}}=\frac{\partial I_2\left(x_i^{(2)},y_i^{(2)},\underline{P}/2\right)}{\partial\underline{P}}-\frac{\partial I_1\left(x_i^{(1)},y_i^{(1)},-\underline{P}/2\right)}{\partial\underline{P}}. \qquad (4.20)$$

Using the chain rule $\frac{\partial I_1(x_i^{(1)},y_i^{(1)},\underline{P}/2)}{\partial \underline{P}}$ is expressed in terms of $\frac{\partial I_1(x_i^{(1)},y_i^{(1)},\underline{P})}{\partial \underline{P}}$:

$$\begin{aligned}\frac{\partial I_1\left(x_i^{(1)},y_i^{(1)},\underline{P}/2\right)}{\partial \underline{P}} &= \frac{\partial I_1\left(x_i^{(1)},y_i^{(1)},\underline{P}/2\right)}{\partial\left(\underline{P}/2\right)}\cdot\frac{\partial \underline{P}/2}{\partial \underline{P}} \\ &= \frac{1}{2}\cdot\frac{\partial I_1\left(x_i^{(1)},y_i^{(1)},\underline{P}\right)}{\partial \underline{P}}.\end{aligned} \tag{4.21}$$

Therefore, we have

$$\frac{\partial r_i\left(x_i^{(1)},y_i^{(1)},\underline{P}/2\right)}{\partial \underline{P}}=\frac{1}{2}\left(\frac{\partial I_2\left(x_i^{(2)},y_i^{(2)},\underline{P}/2\right)}{\partial \underline{P}}+\frac{\partial I_1\left(x_i^{(1)},y_i^{(1)},\underline{P}/2\right)}{\partial \underline{P}}\right). \tag{4.22}$$

Assuming

$$\frac{\partial I_2\left(x_i^{(2)},y_i^{(2)},\underline{P}/2\right)}{\partial \underline{P}}\approx\frac{\partial I_1\left(x_i^{(1)},y_i^{(1)},\underline{P}/2\right)}{\partial \underline{P}} \tag{4.23}$$

we get

$$\frac{\partial r_i\left(x_i^{(1)},y_i^{(1)},\underline{P}/2\right)}{\partial \underline{P}}\approx\frac{\partial I_2\left(x_i^{(2)},y_i^{(2)},\underline{P}/2\right)}{\partial \underline{P}}. \tag{4.24}$$

Taking the second derivative and using Eq.[4.23] we have

$$\begin{aligned}\frac{\partial^2 r_i\left(x_i^{(1)},y_i^{(1)},\underline{P}/2\right)}{\partial \underline{P}^2} &= \frac{1}{2}\left(\frac{1}{2}\cdot\frac{\partial^2 I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial \underline{P}^2}-\frac{1}{2}\cdot\frac{\partial^2 I_1\left(x_i^{(1)},y_i^{(1)}\right)}{\partial \underline{P}^2}\right) \\ &= \frac{\partial^2 I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial \underline{P}^2}-\frac{\partial^2 I_1\left(x_i^{(1)},y_i^{(1)}\right)}{\partial \underline{P}^2} \\ &\approx 0.\end{aligned} \tag{4.25}$$

Comparing Eq. [4.25] to Eqs. [3.1] and [3.5] we get:

$$C_1^{SGM}=\frac{C_1^{GM}}{2} \tag{4.26}$$

$$C_2^{SGM}=C_2^{GM} \tag{4.27}$$

Therefore, the SGM is expected to outperform the regular GM algorithm due to its reduced linearization error.

4.1.1 Algorithm flow

The Symmetric-GM replaces **only** the *single iteration* phase described in section 2.1 and Fig. 2. The *iterative refinement* and *multiscale schemes* described in sections 2.2 and 2.3 respectively, are left unchanged.

1. The matrix $\left(\underline{\underline{H}}^t\underline{\underline{H}}\right)$ is calculated separately for I_2 and I_1 according to Eq. 2.8:

$$\left(\underline{\underline{H}}^t\underline{\underline{H}}\right)^{I_1}_{k,j} = \sum_i \frac{\partial I_1\left(x_i^{(1)}, y_i^{(1)}\right)}{\partial P_k}\frac{\partial I_1\left(x_i^{(1)}, y_i^{(1)}\right)}{\partial P_j} \tag{4.28}$$

$$\left(\underline{\underline{H}}^t\underline{\underline{H}}\right)^{I_2}_{k,j} = \sum_i \frac{\partial I_2\left(x_i^{(2)}, y_i^{(2)}\right)}{\partial P_k}\frac{\partial I_2\left(x_i^{(2)}, y_i^{(2)}\right)}{\partial P_j} \tag{4.29}$$

2. We solve the equation

$$\left(\underline{\underline{H}}^t\underline{\underline{H}}\right)^{SGM}\underline{P}^{SGM} = \underline{\underline{H}}^t\underline{I_t} \tag{4.30}$$

where $\left(\underline{\underline{H}}^t\underline{\underline{H}}\right)^{SGM}$ is given by:

$$\left(\underline{\underline{H}}^t\underline{\underline{H}}\right)^{SGM}_{k,j} = \frac{1}{2}\left(\left(\underline{\underline{H}}^t\underline{\underline{H}}\right)^{I_1}_{k,j} + \left(\underline{\underline{H}}^t\underline{\underline{H}}\right)^{I_2}_{k,j}\right) \tag{4.31}$$

and $\left(\underline{\underline{H}}^t\underline{I_t}\right)$ is calculated according to Eq. 2.6.

3. The SGM returns $\underline{P}^{SGM}$ as the result.

4.2 Bidirectional Gradient Methods (BDGM)

The BDGM uses a different formulation than the GM and the SGM of Eq. [2.1] (see Fig. 5(c)):

$$\underline{P}^* = \arg\min_{\underline{P}}\left\{\sum_{(x_1,y_1)\in S}\left(I_2\left(x_i^{(2)},y_i^{(2)},\underline{P}_2\right) - I_1\left(x_i^{(1)},y_i^{(1)},-\underline{P}_1\right)\right)^2\right\} \tag{4.32}$$

where $\underline{P}_1$and $\underline{P}_2$ have the same dimensions as the motion parameters vector used in the GM and SGM formulations. The overall motion is given by

$$\underline{P} = \underline{P}_1 + \underline{P}_2. \tag{4.33}$$

Let $k, m \in [0 \ldots 1]$, $k + m = 1$, then:

$$\begin{aligned}\underline{P}_1 &= k\cdot\underline{P}\\ \underline{P}_2 &= m\cdot\underline{P}\end{aligned} \tag{4.34}$$

$$\begin{aligned} r_i\left(x_i^{(1)},y_i^{(1)},\underline{P}\right) &= I_2\left(x_i^{(1)},y_i^{(1)},\underline{P}_2\right) - I_1\left(x_i^{(2)},y_i^{(2)},-\underline{P}_1\right)\\ &= I_2\left(x_i^{(1)},y_i^{(1)},k\cdot\underline{P}\right) - I_1\left(x_i^{(2)},y_i^{(2)},-m\cdot\underline{P}\right). \end{aligned} \tag{4.35}$$

$$\frac{\partial r_i\left(x_i^{(1)},y_i^{(1)},\underline{P}\right)}{\partial \underline{P}}=\frac{\partial I_2\left(x_i^{(2)},y_i^{(2)},k\cdot\underline{P}\right)}{\partial \underline{P}}-\frac{\partial I_1\left(x_i^{(1)},y_i^{(1)},-m\cdot\underline{P}\right)}{\partial \underline{P}} \tag{4.36}$$
$$=k\frac{\partial I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial \underline{P}}+m\frac{\partial I_1\left(x_i^{(1)},y_i^{(1)}\right)}{\partial \underline{P}}$$

$$\frac{\partial^2 r_i\left(x_i^{(1)},y_i^{(1)},\underline{P}\right)}{\partial \underline{P}^2}=k^2\frac{\partial^2 I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial \underline{P}^2}+m^2\frac{\partial^2 I_1\left(x_i^{(1)},y_i^{(1)}\right)}{\partial \underline{P}^2}. \tag{4.37}$$

By assuming symmetry in Eq. [4.23] we get

$$\frac{\partial r_i\left(x_i^{(1)},y_i^{(1)},\underline{P}\right)}{\partial \underline{P}}=(k+m)\frac{\partial I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial \underline{P}}=\frac{\partial I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial \underline{P}} \tag{4.38}$$

$$\frac{\partial^2 r\left(x_i^{(1)},y_i^{(1)},\underline{P}\right)}{\partial \underline{P}^2}=\left(k^2+m^2\right)\frac{\partial^2 I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial \underline{P}^2} \tag{4.39}$$
$$=\left(k^2-(k-1)^2\right)\frac{\partial^2 I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial \underline{P}^2}=(2k-1)\frac{\partial I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial \underline{P}}$$

Similar to Eq. [4.27] we have:

$$C_2^{BDGM}=C_2^{GM} \tag{4.40}$$

and the optimal partitioning of the interval $[0\ldots\underline{P}]$, which minimizes $\left|\frac{\partial^2 r\left(x_i^{(1)},y_i^{(1)},\underline{P}\right)}{\partial \underline{P}^2}\right|$, is the symmetric approach $\left(k=\frac{1}{2}\right)$, which was described in section 4.1. Furthermore, the partitioning used by the regular GM is worse than any of the bidirectional formulation, since $\left|\frac{\partial^2 r\left(x_i^{(1)},y_i^{(1)},\underline{P}\right)}{\partial \underline{P}^2}\right|$ is maximized for $k=0,m=1$ or $k=1,m=0$. Although it seems that the BDGM is inferior to the SGM, experimental results show the opposite. The reason is the violation of the symmetry assumption

$$I_2\left(x_i^{(2)},y_i^{(2)},\underline{P}/2\right)\neq I_1\left(x_i^{(1)},y_i^{(1)},-\underline{P}/2\right)\Longrightarrow\frac{\partial I_2\left(x_i^{(2)},y_i^{(2)},\underline{P}/2\right)}{\partial \underline{P}}\neq\frac{\partial I_1\left(x_i^{(1)},y_i^{(1)},-\underline{P}/2\right)}{\partial \underline{P}}. \tag{4.41}$$

Then we get:

$$C_1^{BDGM}\leq C_1^{SGM}=\frac{1}{2}C_1^{GM}. \tag{4.42}$$

4.2.1 Algorithm flow

Similar to the SGM, the BDGM replaces **only** the *single iteration* phase, as follows:

1. The matrix $\underline{\underline{H}}$ is calculated separately for I_2 and I_1 according to Eq.2.5:

$$\underline{\underline{H}}^{I_1}_{i,j} = \frac{\partial I_1\left(x_i^{(1)}, y_i^{(1)}\right)}{\partial P_j} \tag{4.43}$$

$$\underline{\underline{H}}^{I_2}_{i,j} = \frac{\partial I_2\left(x_i^{(2)}, y_i^{(2)}\right)}{\partial P_j} \tag{4.44}$$

2. $\underline{\underline{H}}^{BDGM}$ is formed by:

$$\underline{\underline{H}}^{BDGM} = \left[\begin{array}{cc} \underline{\underline{H}}^{I_1} & \underline{\underline{H}}^{I_2} \end{array}\right] \tag{4.45}$$

   $\underline{\underline{H}}^{BDGM}$ is a matrix of dimensions $(n_{pixels} \times 2 \cdot n_{param})$, where n_{param} is the number of motion parameters and n_{pixels} is the number of pixels common to I_2 and I_1.

3. Denote by $\underline{P}^{BDGM}$ the BDGM parameters vector, then

$$\underline{P}^{BDGM} = \left[\begin{array}{c} \underline{P}^1 \\ \underline{P}^2 \end{array}\right] \tag{4.46}$$

   $\underline{P}^1$ and $\underline{P}^2$ are vectors of dimension $(n_{param} \times 1)$.

4. We solve the equation

$$\left(\left(\underline{\underline{H}}^{BDGM}\right)^t \underline{\underline{H}}^{BDGM}\right) \underline{P}^{BDGM} = \left(\underline{\underline{H}}^{BDGM}\right)^t \underline{I_t} \tag{4.47}$$

   where $\underline{I_t}$ is similar to the one used in section 2.1.

5. After solving Eq. 4.47, the solution $\underline{P}^{BDGM}$ is given by:

$$\underline{P}^{BDGM} = \underline{P}^1 + \underline{P}^2 \tag{4.48}$$

5 Experimental Results

This section describes the performance of the proposed new techniques. The numeric results are expressed in terms of alignment error vs. the number of iterations needed for convergence as the total computation time is linearly dependent on the number of iterations. Each simulation was conducted using a set of 50 inital estimates of the motion parameters, where the estimates were

given as relative translation values and are displayed in the results figures as an overlay of red dots in both images. Thus, the alignment error value at each iteration the avergae of all the simulations. the Real and simulated image pairs were used. The same implementations of the *iterative refinement* (section 2.2) and *multiscale embedding* (section 2.3) were used for the SGM, BDGM and GM algorithms. Thus, the only difference was the *single iteration module*, which was replaced by the algorithms described in sections 4.1 and 4.2 for the SGM and BDGM, respectively. The pyramid has been constructed using a three-tap filter $\begin{bmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{bmatrix}$ and the derivative was approximated using $\begin{bmatrix} \frac{1}{2} & 0 & -\frac{1}{2} \end{bmatrix}$. Following [8, 9], other filters were tested with no significant improvements. The initial estimate was an estimate of the translation parameters. The SGM and BDGM were tested by estimating large and small motions using several motion models: rotation, affine and pseudo-projective.

5.1 Estimation of large and very large rotations

The image presented in Fig. 3 was rotated using bilinear interpolation, while the background areas created by the rotation were padded with zeros. The registration was calculated using a linearized rotation model:

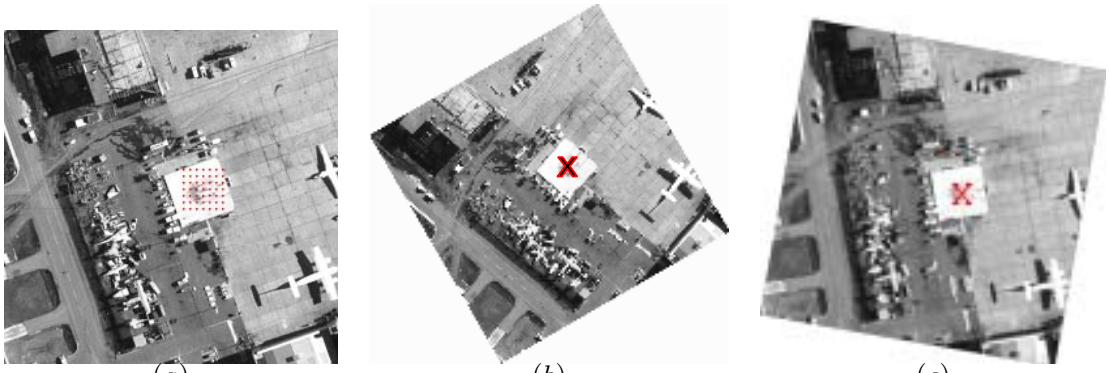
$$\begin{aligned} x_1 &= a \cdot x_2 + b \cdot y_2 + c \\ y_1 &= -b \cdot x_2 + a \cdot y_2 + d \end{aligned} \tag{5.1}$$

Figure 7(a) shows the performance of registering an image rotated by 10°, which is considered to be a large rotation. The BDGM converged twice as fast in comparison to the GM: 4 iterations compared to 7 iterations. The SGM converged in 5 iterations but to a higher alignment error. This instability of the SGM is more evident in the 30° registration results, presented in Fig. 7(b): the SGM diverged while the BDGM significantly outperforms the GM by converging in 25 iterations compared to the GM's 37 iterations. We attribute the instability of the SGM to the violation of the symmetry assumption (Eq. [4.23]) due to the zero padding.

5.2 Estimation of small affine motion

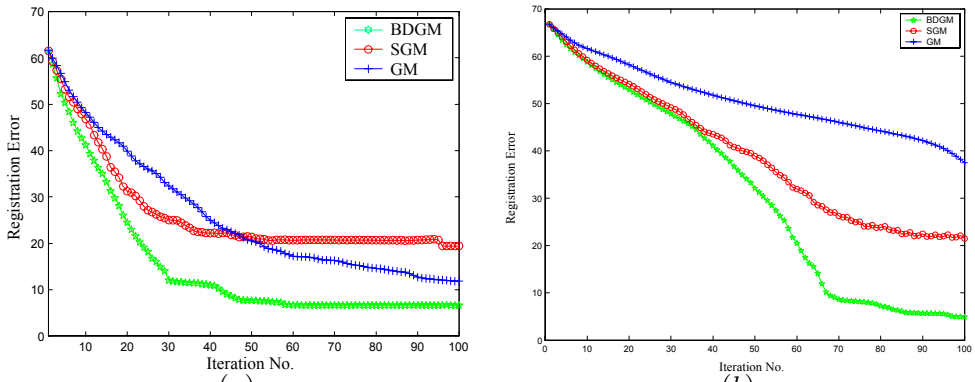
According to Eqs. [4.26] and [4.42] the BDGM and SGM, respectively, are expected to perform similarly to the GM when registering small motions. In order to verify it experimentally, we registered the images in Fig. 8 using the affine motion model:

$$\begin{aligned} x_1 &= a \cdot x_2 + b \cdot y_2 + c \\ y_1 &= d \cdot x_2 + e \cdot y_2 + f \end{aligned} \tag{5.2}$$



(a) (b) (c)

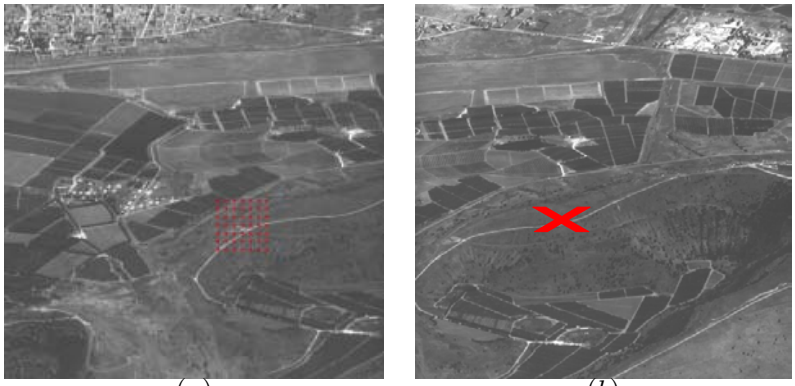
Figure 6: Test images for rotation registration. The red dots in image (a) are the initial estimates of the red X in image (b). X marks the initial motion (a) original airport image. (b) airport image rotated by 10°. (c) airport image rotated by 30°.



(a) (b)

Figure 7: Registration results of rotated images: (a) 10° rotation (b) 30° rotation. In both case the BDGM and SGM converged faster than the regular GM.

The results presented in Fig. 9 show that all the algorithms converged similarly. However, the BDGM suffers from numerical instability which can be attributed to a larger number of unknowns solved in each iteration (12 unknowns used by the BDGM compared to 6 unknowns used by the SGM and the GM).



(a) (b)

Figure 8: Test images for affine registration with small motion. The red dots in image (a) are initial estimates of the red X in image (b) which were used in the simulations.

5.3 Registration of images with low contrast

Instability in the registration process can also be attributed to images which have low contrast. In this type of images the spatial derivatives are very small:

$$\frac{\partial r\left(x_i^{(1)},y_i^{(1)},\underline{P}\right)}{\partial \underline{P}} \longrightarrow 0$$
$$\frac{\partial^2 r\left(x_i^{(1)},y_i^{(1)},\underline{P}\right)}{\partial \underline{P}^2} \longrightarrow 0 \quad .$$

Therefore, according to Eqs. [3.5] and [3.6] the convergence rate of the GM deteriorates. The images presented in 10 are real airborne images, which are registered using the affine motion model defined in Eq. [5.2].

The results, presented in Fig. 11, show that both the BDGM and SGM were able to converge to the solution while the GM completely diverged. However, the numerical instability of the BDGM in proximity of the solution is evident, similar to the result of section 5.2. This phenomenon could have been avoided, by switching from the BDGM to either the SGM or GM near the proximity of the solution.

5.4 Estimation of large and very large panoramic motion

The registration of panoramic images is of special importance, since it is the basis for most mosaic based applications discussed in section 1. The motion model used for panoramic image registration is the pseudo-projective model [1, 4]

$$\begin{aligned} x_1 &= \frac{a\cdot x_2+b\cdot y_2+c}{g\cdot x_2+h\cdot y_2+1} \\ y_1 &= \frac{d\cdot x_2+e\cdot y_2+f}{g\cdot x_2+h\cdot y_2+1} \end{aligned} \tag{5.3}$$

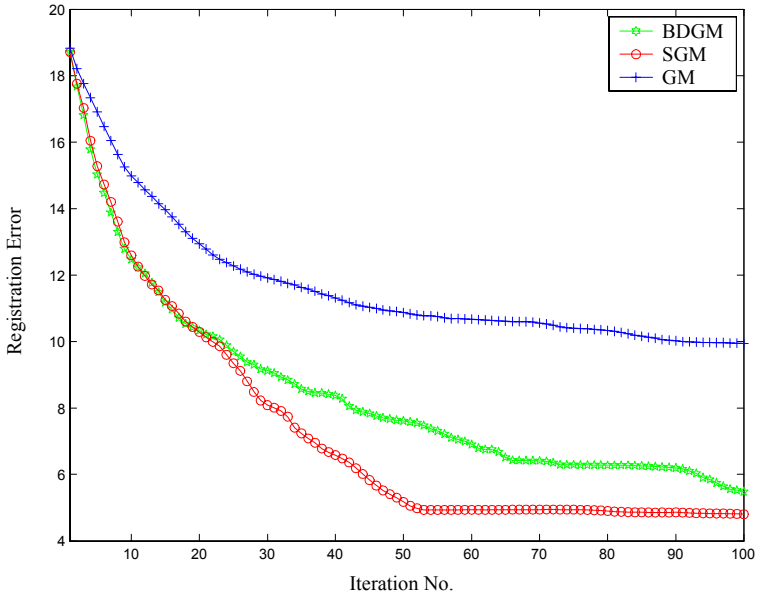


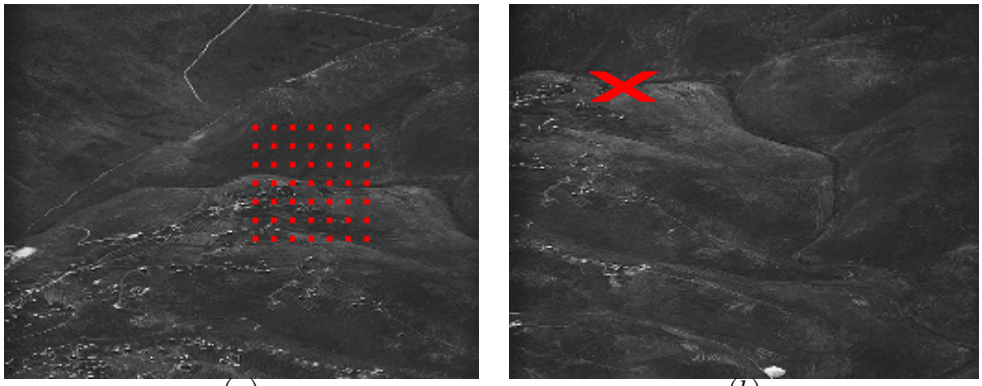
Figure 9: Performance of small motion registration using the affine motion model.

Due to large number of unknowns and the non-linear nature of Eq. [5.3], the GM based registration becomes slow and unstable. Two sets of images photographed by a regular 35mm cameras were used to compare between the performance of the registration algorithms: large panoramic transformation is presented in Fig. 12 while small panoramic motion is shown in Fig. 14.

The results shown in Fig. 13 demonstrate the superior convergence of the BDGM (13 iterations) and SGM (17 iterations) compared to the GM algorithm (23 iterations). Estimation of small projective motions is presented in figure 15. The results are similar to those obtained in section 5.2 where the BDGM and SGM coincide and converge twice as fast (5 iterations) as the GM (9 iterations).

6 Conclusions and future work

In this paper we proposed two new formulations which enhance the performance of gradient based image registration methods. These algorithms extend the current state-of-the-art image registration algorithms and were proven to possess superior convergence range and rate. By analyzing the convergence properties using non-linear optimization algorithms, we derived explicit expressions for the convergence of the GM. The experimental results verify the theoretical analysis. Future work includes the application of the BDGM and SGM to other GM based algorithms such as direct



(a) (b)

Figure 10: Images used to test the registration under poor illumination conditions. The red dots in image (a) are the initial estimates of the red X in image (b) used in the simulations.

estimation of 3D structure [16]. Furthermore, the improved convergence rate of the DBGM and SGM is vital for advanced video compression standards such as the MPEG4 [15], when implemented on low-power mobile devices. In order to further reduce the computational complexity, we intend to integrate the proposed algorithm with the WarpFree formulation presented in [7].

7 Appendix A: Convergence properties of the Gauss-Newton optimization algorithm

7.1 Definitions

The general least square problem (LS) is defined as

$$x^* = \min_{\underline{x}} \{f(x)\} \tag{7.1}$$

where $f(x)$ is the sum of squares

$$f(x) = \sum_{n=1}^{m} [r_n(x)]^2 = [r_1(x) \dots r_m(x)] \begin{bmatrix} r_1(x) \\ r_2(x) \\ \vdots \\ r_m(x) \end{bmatrix} = R^T(x) \cdot R(x). \tag{7.2}$$

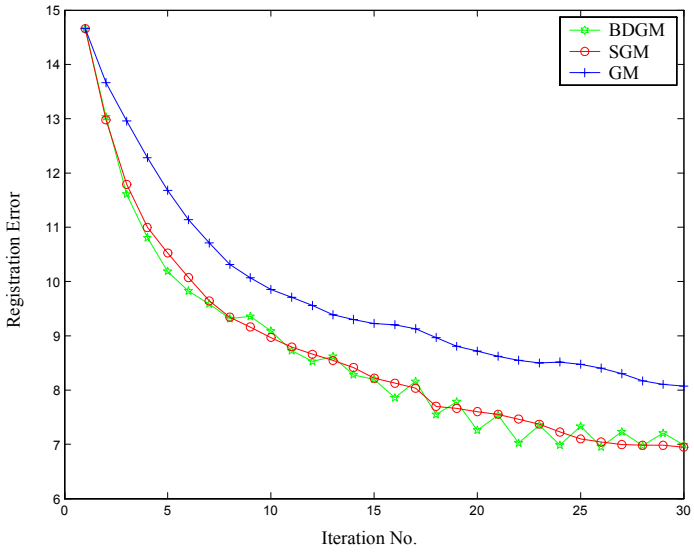


Figure 11: Registration performance under poor illumination conditions: The regular GM diverges, while the SGM and BDGM converge. Due to the small motion, the SGM converges better than the BDGM, which is unstable due to its larger number of unknowns.

We start by calculating the first and second derivatives of the objective function $f(x)$

$$\frac{\partial f(x)}{\partial x} = 2\sum_{n=1}^{m} \frac{\partial r_n(x)}{\partial x} r_k(x) = 2\left[\frac{\partial r_1(x)}{\partial x} \quad \frac{\partial r_2(x)}{\partial x} \quad \cdots \quad \frac{\partial r_m(x)}{\partial x}\right] \cdot R(x) = 2A(x) R(x) \quad (7.3)$$

$$\frac{\partial^2 f(x)}{\partial x_i^{(2)}} = 2\sum_{n=1}^{m} \frac{\partial r_n(x)}{\partial x} \frac{\partial r_n^T(x)}{\partial x} + 2\sum_{n=1}^{m} \frac{\partial^2 r_n(x)}{\partial x^2} r_n(x) = 2A(x) A^T(x) + 2\sum_{n=1}^{m} \frac{\partial^2 r_n(x)}{\partial x^2} r_n(x). \quad (7.4)$$

The Gauss-Newton iterative optimization algorithm for the LS problem [5] is

$$x_{k+1} = x_k - \left[A(x_k) A^T(x_k)\right]^{-1} A(x_k) R(x_k) \quad (7.5)$$

where x_k is the parameters vector estimated at iteration k and

$$\varepsilon_k = x_k - x^* \quad (7.6)$$

and ε_k is the estimation error at iteration k.

(a) (b)

Figure 12: Panoramic images with large motion. The red dots in image (a) are the initial estimates of the red X in image (b) used in the simulations.

7.2 Convergence analysis

We approximate $\frac{\partial f(\tilde{x})}{\partial x}$ using a first order Taylor expansion around x

$$\frac{\partial f(\tilde{x})}{\partial x} \approx \frac{\partial f(x)}{\partial x} + \frac{\partial^2 f(x)}{\partial x^2} \cdot (\hat{x} - x) + \frac{1}{2}\frac{\partial^3 f(\tilde{x})}{\partial x^3} \cdot (\hat{x} - x)^2, \tilde{x} \in [\hat{x}, x]. \tag{7.7}$$

Inserting Eqs. [7.3] and [7.4] into Eq. [7.7] we get

$$\frac{\partial f(\tilde{x})}{\partial x} \approx 2A(x) R(x) + \left\{ 2A(x) A^T(x) + 2\sum_{n=1}^{m} \frac{\partial^2 r_n(x)}{\partial x^2} r_n(x) \right\} \cdot (\hat{x} - x) + \frac{f^{(3)}(\tilde{x})}{2} \cdot (\hat{x} - x)^2. \tag{7.8}$$

Using Eq. [7.8] we estimate the gradient at the minimum point x^* using a Taylor approximation around x_k:

$$A(x^*) R(x^*) = A(x_k) R(x_k) - \left[A(x_k) A^T(x_k) + \sum_{n=1}^{m} \frac{\partial r_n^2(x)}{\partial x} r_n(x) \right] \cdot \varepsilon_k + O\left({\varepsilon_k}^T \varepsilon_k\right). \tag{7.9}$$

Rewriting Eq. [7.5] for ε_k we get:

$$\varepsilon_{k+1} = \varepsilon_k - \left[A(x_k) A^T(x_k)\right]^{-1} A(x_k) R(x_k). \tag{7.10}$$

Since $A(x^*) = 0$ we get the identity:

$$\varepsilon_k = \left[A(x_k) A^T(x_k)\right]^{-1} \left[A(x_k) A^T(x_k) \cdot \varepsilon_k + \underbrace{A(x^*) R(x^*)}_{=0} \right]. \tag{7.11}$$

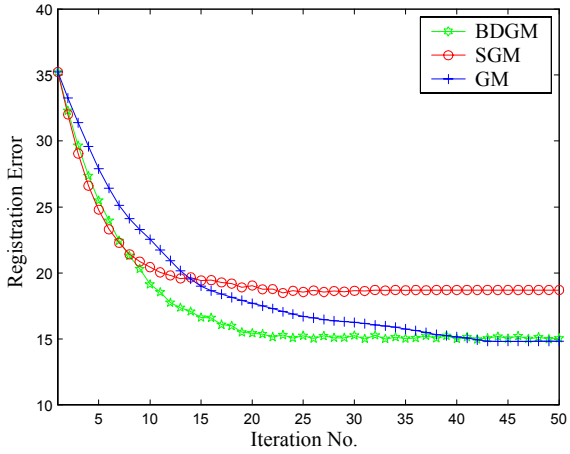


Figure 13: Registration results for large panoramic motion: the SGM and BDGM converge twice as fast as the regular GM. Due to the large motion the symmetric assumption (Eq. 4.23) is violated and the BDGM converges better than the SGM.

inserting Eq. [7.11] into Eq. [7.10] we get

$$\begin{aligned}\varepsilon_{k+1} = &\left[A\left(x_k\right)A^T\left(x_k\right)\right]^{-1}\left[A\left(x_k\right)A^T\left(x_k\right)\cdot\varepsilon_k + A\left(x^*\right)R\left(x^*\right)\right] \\ &- \left[A\left(x_k\right)A^T\left(x_k\right)\right]^{-1}A\left(x_k\right)R\left(x_k\right) \\ = &\left[A\left(x_k\right)A^T\left(x_k\right)\right]^{-1}\left[A\left(x_k\right)A^T\left(x_k\right)\cdot\varepsilon_k + A\left(x^*\right)R\left(x^*\right) - A\left(x_k\right)R\left(x_k\right)\right]. \quad (7.12)\end{aligned}$$

Inserting Eq. [7.9] into Eq. [7.12] we get

$$\begin{aligned}\varepsilon_{k+1} = &-\left[A\left(x_k\right)A^T\left(x_k\right)\right]^{-1}\left(\sum_{n=1}^{m}\frac{\partial r_n^2\left(x_k\right)}{\partial x}r_n\left(x_k\right)\right)\cdot\varepsilon_k - \\ &\left[A\left(x_k\right)A^T\left(x_k\right)\right]^{-1}\cdot O\left({\varepsilon_k}^T\varepsilon_k\right). \quad (7.13)\end{aligned}$$

Taking a norm on both sides and using the Cauchy-Schwartz inequality we get

$$\begin{aligned}\left\|\varepsilon_{k+1}\right\| \le &\left\|\left[A\left(x_k\right)A^T\left(x_k\right)\right]^{-1}\sum_{n=1}^{m}\frac{\partial r_n^2\left(x_k\right)}{\partial x}r_n\left(x_k\right)\right\|\cdot\left\|\varepsilon_k\right\| \\ &+\left\|\left[A\left(x_k\right)A^T\left(x_k\right)\right]^{-1}\right\|\cdot O\left(\left\|\varepsilon_k\right\|^2\right). \quad (7.14)\end{aligned}$$

In other words,

$$\left\|\varepsilon_{k+1}\right\| \le C_1\cdot\left\|\varepsilon_k\right\| + C_2\cdot\left\|\varepsilon_k\right\|^2. \quad (7.15)$$

7.3 Convergence analysis of the objective function

In the case of the motion estimation problem, the *natural* norm related to the problem is the L_2 norm of the image intensity alignment error, rather then the norm of the motion parameters error.

(a) (b)

Figure 14: Panoramic images with small motion. The red dots in image (a) are the initial estimates of the red X in image (b) used in the simulations.

Therefore, we relate the convergence of the estimated motion parameters to the convergence of the objective function $f(\underline{x})$. We approximate $r(\underline{x}_k)$ by a first order Taylor expansion around the minimum point $\underline{x}^*$:

$$r(\underline{x}^*) = r(\underline{x}_k) + A(\underline{x}_k) \cdot (\underline{x}^* - \underline{x}_k).$$

Then, by taking the L_2 norm we get:

$$\|\Delta r_k\| = \|r(\underline{x}^*) - r(\underline{x}_k)\| = A(\underline{x}_k) \|\underline{\varepsilon}_k\| \tag{7.16}$$

where ε_k is defined as in [7.6].

Rewriting Eq. [7.16] for $\|\Delta r_{k-1}\|$:

$$\|\Delta r_{k+1}\| = \left\|r(\underline{x}^*) - r\left(\underline{x}_{k+1}\right)\right\| = A(\underline{x}_{k+1}) \left\|\underline{\varepsilon}_{k+1}\right\| \tag{7.17}$$

and assuming a small refinement step: $A(\underline{x}_{k+1}) \approx A(\underline{x}_k)$, then

$$\|\Delta r_{k+1}\| = \frac{\left\|\underline{\varepsilon}_{k+1}\right\|}{\|\underline{\varepsilon}_k\|} \cdot \|\Delta r_k\| \tag{7.18}$$

We conclude that the parameters error $\underline{\varepsilon}_k$ and the objective function Δr_k have a similar convergence rate.

7.4 Conclusion

Using Eq. [7.14] the convergence of the Gauss-Newton algorithm can be divided into linear and quadratic convergence phases depending on the properties of the objective function $f(x)$.

Linear convergence phase

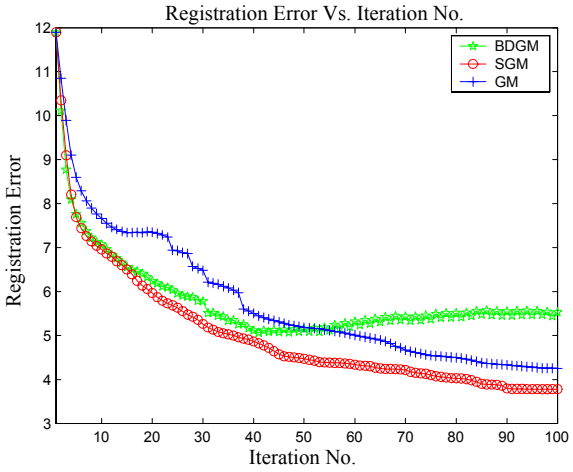


Figure 15: Registration results for small panoramic motion: the SGM and BDGM converge twice as fast as the regular GM. Due to the small motion the symmetric assumption (Eq. 4.23) is valid and the SGM and BDGM converge similarly.

In this phase the convergence is dominated by the linear convergence term C_1 (Eq. [7.15]).

$$\left\|\underline{\varepsilon}_{k+1}\right\| \leq C_1 \cdot \left\|\underline{\varepsilon}_k\right\| \tag{7.19}$$

Therefore we have $C_1 \gg C_2$:

$$\left\|\sum_{k=1}^{m} \frac{\partial r_k^2\left(\underline{x}_k\right)}{\partial x} r_k(\underline{x}_k)\right\| \gg \left\|\sum_{k=1}^{m} \frac{\partial r_k^2\left(\underline{x}_k\right)}{\partial x}\right\| \tag{7.20}$$

and the observation error term $r_k(\underline{x}_k)$ satisfies

$$r_k(\underline{x}_k) \gg 1 \tag{7.21}$$

Equation [7.21] characterizes situations in which there is a significant discrepancy in the minimization model defined in Eq. [7.2] due to large deviations of the estimated parameters $\underline{x}_k$ from the true parameters $\underline{x}^*$. For $\|C_1\| > 1$ the process diverges.

Close range phase

In regions in proximity of the solution $r_k\left(\underline{x}_k\right) \to 0$ and $C_1 \to 0$. The second term C_2 in Eq. [7.15] becomes dominant, making the convergence rate quadratic.

References

[1] S .Man and R. W. Picard, "Virtual Bellows: constructing high quality stills from video", Proc. IEEE Int. Conf. Image Processing Austin, TX, Nov. 13-16, 1994.

[2] L. Barron, D. J. Fleet and S. S. Beauchemin, "Performance of Optical flow Techniques", Int. J. Computational Vision, Vol. 12, pp. 43-77, 1994.

[3] B. K. P. Horn and B. G. Schunck, "Determining optical Flow," Artificial Intelligence,vol. 17, pp. 185-203, 1981.

[4] R. Szeliski, "Image Mosaicking for Tele-Reality Applications",WACV94, pp. 44-53, 1994..

[5] P. Gill , "Practical Optimization", Academic Press, 1982.

[6] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, "Numerical Recipes in C: The Art of Scientific Computing", Cambridge Univ. Press, 1988.

[7] A. Averbuch, Y. Keller, "Warp free Gradient methods based image registration", in preperation.

[8] M. Elad, P. Teo, Y. Hel-Or, "Optimal Filters For Gradient-Based Motion Estimation", Technical Report # 111, HP Lab, 1997.

[9] E. P. Simoncelli, "Design of multi-dimensional derivatives filters," IEEE International Conf. on Image Processing, Austin Tx, pp. 790-794, 1994.

[10] M. Irani and S. Peleg, "Motion Analysis for Image Enhancement: Resolution, Occlusion and Transparency". Journal of Visual Communication and Image Representation, Vol. 4, No. 4, pp.324-335, December 1993.

[11] M. Irani, P. Anandan, and S. Hsu. "Mosaic based representations of video sequences and their applications", International Conference on Computer Vision, pages 605-611, 1995.

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

[13] B. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in Proc. DARPA, Image Understanding Workshop, pp. 121-130, 1981.

[14] C. D. Kuglin and D. C. Hines. "The phase correlation image alignment method", IEEE Conference on Cybernetics and Society, pp. 163–165, September 1975.

[15] F. Dufaux and J. Konrad, "Efficient, Robust, and Fast Global Motion Estimation for Video Coding", IEEE Transactions on Image Processing, vol. 9, no. 3, pp. 497-501, March 2000.

[16] G. Stein and A. Shashua. "Model-Based Brightness Constraints: On Direct Estimation of Structure and Motion", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 22, No. 9, September 2000.

[17] M. Irani and P. Anandan. "All About Direct Methods", ftp://ftp.robots.ox.ac.uk/pub/outgoing/phst/summary.ps.gz .

[18] P. J. Burt, R. Hingorani and R. J. Kolezynski, "Mechanism for isolating component patterns in the sequential analysis of multiple motion", IEEE Workshop on Visual Motion, pp. 187-193, 1991.