Fast motion estimation using bidirectional gradient methods

Yosi Keller, Amir Averbuch,

***Abstract*— Gradient based motion estimation techniques (GM) are considered to be in the heart of state-of-the-art registration algorithms, 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. These improve 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.**

***Index Terms*— Global motion estimation, Sub-pixel registration, Gradient methods, image alignment**

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

I. 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 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. 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 differ on the way the motion parameters are updated, either by incrementing the motion parameters [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:

$$I_1(x,y) = I_2\left(\widetilde{x}(x,y,p), \widetilde{y}(x,y,p)\right) \tag{1}$$

where the structure of the parameters vector p depends on the type of the estimated motion model. Gathering and solving all the equations associated with pixels in the mutual support (p is assumed to be constant over the mutual area), estimates the *global motion* between the images [4], thus gaining robustness due to a highly over-constrained linear system of equations (each pixel contributes a linear constraint). Gathering the equations related to small image patches estimates *local motion* [13]. Equation 1 is solved by non-linear iterative optimization techniques such as Gauss-Newton [6] and Levenberg-Marquardt (LM) [4] described in section II. 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 algorithms which are robust to large motions [14], [15]. Further improvements were achieved by introducing robust statistical measures such as iteratively weighted least-squares [19], [20] and Total least-squares [21] within the GM formulation.

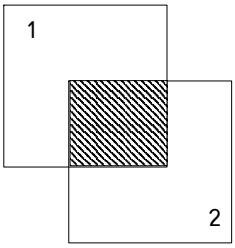


Fig. 1. Image translation.

In order to improve the convergence of the GM it is analyzed using optimization theory methodology in section III. The analysis of the GM convergence leads to a new robust constructive algorithm that achieves faster convergence through symmetric and non-symmetric bidirectional formulations, presented in section IV. These properties are experimentally verified in section V.

II. Gradient method based motion estimation

GM methodology [17] estimates the motion parameters p by minimizing the intensity discrepancies between the input images $I_1(x,y)$ and $I_2(x,y)$ described in Fig. 1

$$p^* = \arg\min_p \sum_{(x_1,y_1)\in S} \left(I_1\left(x_i^{(1)}, y_i^{(1)}\right) - I_2\left(x_i^{(2)}, y_i^{(2)}\right)\right)^2 \tag{2}$$

Yosi Keller is with the Department of Mathematics, Yale University, 06520 USA. (e-mail: yosi.keller@yale.edu).

Amir Averbuch is with the Department of Computer Science, School of Mathematical Sciences, Tel Aviv University, Tel Aviv, Israel. (e-mail: amir@math.tau.ac.il).

where

$$\begin{aligned} x_i^{(2)} &= f(x_i^{(1)}, y_i^{(1)}, p) \\ y_i^{(2)} &= g(x_i^{(1)}, y_i^{(1)}, p) \end{aligned} \tag{3}$$

and S is the set of coordinates of pixels that are common to I_1 and I_2 in I_1's coordinates and p is the estimated parameter vector. Next we follow the formulation of [1], [4] by solving Eq. 2 using non-linear optimization techniques. The basic GM formulation and the iterative refinement step are described in sections II-A and II-B, respectively. These are embedded in a multi-resolution scheme in section II-C.

A. Basic GM formulation

The non-linear optimization of Eq. 2, 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 p

$$I_1(x_i^{(1)}, y_i^{(1)}) = I_2(x_i^{(2)}, y_i^{(2)}) + \sum_{p_j \in p} \frac{\partial I_2(x_i^{(2)}, y_i^{(2)})}{\partial p_j} p_j \tag{4}$$

where $\frac{\partial I_2}{\partial p_j}$ is the partial 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

$$Hp = I_t \tag{5}$$

where

$$\begin{aligned} H_{i,j} &= \frac{\partial I_2\left(x_i^{(2)}, y_i^{(2)}\right)}{\partial p_j} \\ &= \frac{\partial I_2\left(x_i^{(2)}, y_i^{(2)}\right)}{\partial x_2} \frac{\partial f\left(x_i^{(1)}, y_i^{(1)}, p\right)}{\partial p_j} \\ &+ \frac{\partial I_2\left(x_i^{(2)}, y_i^{(2)}\right)}{\partial y_2} \frac{\partial g\left(x_i^{(1)}, y_i^{(1)}, p\right)}{\partial p_j} \end{aligned} \tag{6}$$

$$(I_t)_i = I_1\left(x_i^{(1)}, y_i^{(1)}\right) - I_2\left(x_i^{(2)}, y_i^{(2)}\right). \tag{7}$$

Equation 5 is solved using least squares [6]

$$p = \left(H^T H\right)^{-1} H^T I_t \tag{8}$$

where H^T is the transpose of H. The expressions for $H^T H$ and $H^T I_t$ are derived analytically and computed directly

$$\left(H^T H\right)_{k,j} = \sum_i \frac{\partial I_2\left(x_i^{(2)}, y_i^{(2)}\right)}{\partial p_k} \frac{\partial I_2\left(x_i^{(2)}, y_i^{(2)}\right)}{\partial p_j} \tag{9}$$

$$\left(H^T I_t\right)_k = \sum_i \frac{\partial I_2\left(x_i^{(2)}, y_i^{(2)}\right)}{\partial p_k} (I_t)_i. \tag{10}$$

Algorithm flow

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

1) The matrix $H^T H$ and vector $H^T I_t$ are computed using Eq. 9 and Eq. 10, respectively.
2) Eq. 8 is solved using singular value decomposition [6].
3) The GM returns p as its output (result).

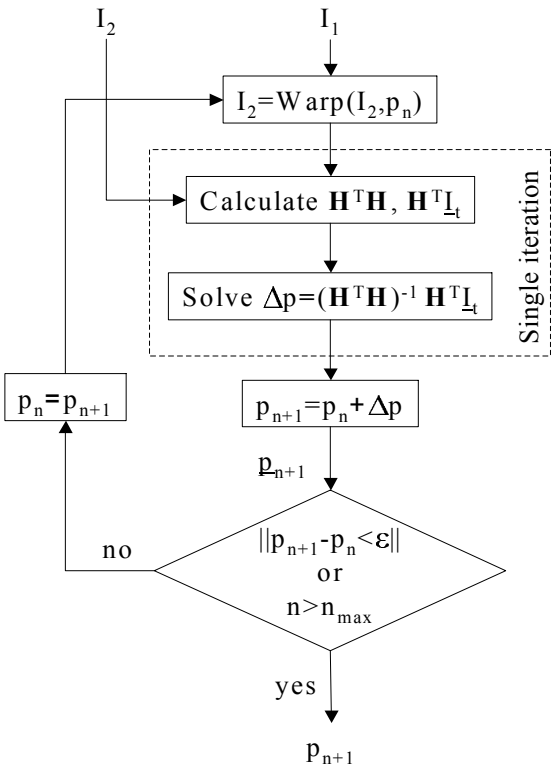


Fig. 2. Block diagram of the basic and iterative GM formulations. For $n = 0$, p_0 is given as an initial guess and ΔP is the iterative update after each iteration.

B. Iterative solution of the gradient methods

Denote:

p_0- an initial estimated solution of Eq. 2 given as input, such that $Warp\,(I_2, p_0) \approx I_1$.

p_n- the estimated solution after $n = 1, \ldots$ iterations.

Then, the nth iteration is given by:

1) The input image I_2 is wrapped towards I_1 using the current estimate p_n and it is stored in $\widetilde{I}_2$ $n \geq 0$. For $n = 0$ p_0 is given as input.
2) I_1 and $\widetilde{I}_2$ are used as input images to the procedure described in section II-A.
3) Δp, the result of step 2, is used to update the solution:

$$p_{n+1} = \Delta p + p_n \qquad n \geq 0$$

4) Go back to step 1 until one of the following stopping criteria is met:
   a) At most $N_{\max}$ iterations are performed.
   b) The process is stopped if the translation parameters within the updated term Δp reaches a predetermined threshold.

C. 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, whose analysis is given in section III-B. Next we describe the coarse-to-fine formulation. A thorough description can be found in [1]:

1) 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 common area is reached. The minimal size of the common area depends upon the estimated motion model, while the resolution step depends upon the motion estimation's accuracy at each resolution level.
3) Starting with the coarsest scale, the initial estimate p_0 is used to bootstrap the iterative refinement algorithm described in section II-B.
4) The result of the iterative refinement from coarse-to-fine (step 2) is recursively repeated until the original image size is reached.

III. Convergence analysis of gradient methods

In order to analyze the convergence properties of the GM algorithm, we study the convergence of the Gauss-Newton algorithm in Appendix A and apply the results to motion estimation in section III-A.

A. General convergence analysis of the Gauss-Newton algorithm

The analysis given in Appendix A shows that the convergence of the Gauss-Newton algorithm can be divided into two distinct phases defined by Eq. 89

$$\|\varepsilon_{k+1}\| \leq C_1 \cdot \|\varepsilon_k\| + C_2 \cdot \|\varepsilon_k\|^2 \tag{11}$$

where

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

ε_k is the parameters estimation error after iteration k

$r_n(x_k)$ is the error associated with the nth equation at iteration k

$A(x_k)$ is the Jacobian matrix at iteration k.

By rearranging the GM formulations developed in section II, 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.

$$r_n(x_k) = I_2\left(f\left(x_i^{(1)}, y_i^{(1)}, p\right), g\left(x_i^{(1)}, y_i^{(1)}, p\right)\right) - I_1\left(x_i^{(1)}, y_i^{(1)}\right) \tag{12}$$

where the vector parameters x identifies with p:

$$A_{i,k} = \frac{\partial r_n(x_k)}{\partial x_k} = \frac{\partial I_2(x_i^{(2)}, y_i^{(2)})}{\partial p_k} \tag{13}$$

and the equation index n identifies with the GM index i. $A(x_k)$ identifies with the matrix H defined in Eq. 6 and the second derivative $\frac{\partial^2 r_n(x_k)}{\partial x_k^2}$ is given by

$$\frac{\partial^2 r_n(x_k)}{\partial x_k^2} = \frac{\partial^2 I_2(x_i^{(2)}, y_i^{(2)})}{\partial p_k^2}. \tag{14}$$

Therefore, we have

$$C_1^{GM} = \left\| \left[\frac{\partial I_2}{\partial p} \cdot \frac{\partial I_2}{\partial p}^T \right]^{-1} \cdot \sum_{i=1}^{m} \frac{\partial^2 I_2(x_i^{(2)}, y_i^{(2)})}{\partial p_k^2} r_n(x_k) \right\| \tag{15}$$

$$C_2^{GM} = \left\| \left[\frac{\partial I_2}{\partial p} \cdot \frac{\partial I_2}{\partial p}^T \right]^{-1} \right\| \tag{16}$$

and the error associated with each equation is the truncation error of the first order Taylor expansion [6]

$$\varepsilon_{GM}\left(x_i^{(2)}, y_i^{(2)}, p\right) = \frac{1}{2} \sum_{p_j \in p} \frac{\partial^2 I_2(x_i^{(2)}, y_i^{(2)})}{\partial p_j} \left(p_j - p_j^*\right)^2 \tag{17}$$

where p_j^* is the optimal value of p_j.

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

Initialization phase:
During the initial iterations we have $\|p - p^*\| \gg 0$ and $C_1^{GM} \gg 0$, therefore the convergence rate is linear.

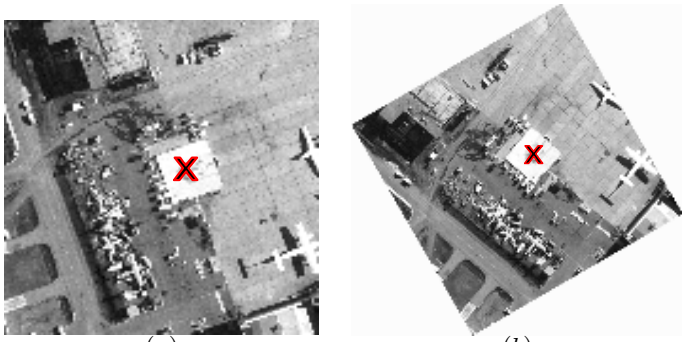
Convergence phase:
In the proximity of the solution $\|p - p^*\| \to 0$, $C_1^{GM} \to 0$ and the convergence is quadratic.

This analysis was experimentally verified by registering the "Airfield" image and a 30° rotated replica of it shown in Fig. 3. Two distinct convergence phases are observed in the registration results presented in Fig. 4. Starting with a low-rate linear convergence and switching to the quadratic convergence phase after a cross-over point located at $n = 170$ iterations.

B. 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 proven to decrease the error term in Eq. 17 and improve the convergence rate.
Denote by $\varepsilon_{trans}(s)$, the error associated with the translation parameters (dx_s, dy_s) at a resolution scale s:

$$\varepsilon_{trans}(s) = \begin{bmatrix} dx_s - dx_s^* \\ dy_s - dy_s^* \end{bmatrix} \tag{18}$$



(a) (b)

Fig. 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.

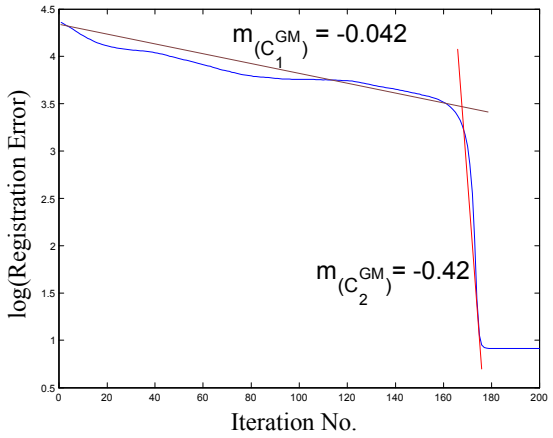


Fig. 4. The convergence process is divided into two phases: the first is related to large motion estimation characterized by a low convergence rate $m_{(C_1)}$, the second is related to small motion having a high convergence rate $m_{(C_2)}$. The cross-over region is located after iteration $n \approx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

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:

$$\begin{aligned} dx_{s_2} &= dx_{s_1} \cdot \tfrac{s_1}{s_2} \\ dy_{s_2} &= dy_{s_1} \cdot \tfrac{s_1}{s_2} \end{aligned} \tag{19}$$

and the associated error becomes

$$\begin{aligned} \varepsilon_{trans}(s_2)^2 &= \begin{pmatrix} dx_{s_2} - dx_{s_2}^* \\ dy_{s_2} - dy_{s_2}^* \end{pmatrix}^2 \\ &= \begin{pmatrix} dx_{s_1} \cdot \frac{s_1}{s_2} - dx_{s_1}^* \cdot \frac{s_1}{s_2} \\ dy_{s_1} \cdot \frac{s_1}{s_2} - dy_{s_1}^* \cdot \frac{s_1}{s_2} \end{pmatrix}^2 \\ &= \varepsilon_{trans}(s_1)^2 \cdot \underbrace{\left(\frac{s_1}{s_2}\right)^2}_{<1}. \end{aligned} \tag{20}$$

Hence, the error associated with the translation error is decreased by a factor of $\frac{s_1}{s_2} < 1$.

Inserting Eq. 20 into Eq. 17 we get:

$$\begin{aligned} &\varepsilon_{GM}\left(x_i^{(2)}, y_i^{(2)}, [dx, dy], s_2\right) \\ &= \frac{1}{2} \sum_{p_j \in [dx,dy]} \frac{\partial^2 I_2(x_i^{(2)}, y_i^{(2)})}{\partial p_j} \varepsilon_{trans}(s_2)^2 \\ &= \frac{1}{2} \sum_{p_j \in [dx,dy]} \frac{\partial^2 I_2(x_i^{(2)}, y_i^{(2)})}{\partial p_j} \left(\varepsilon_{trans}(s_1)^2 \cdot \left(\frac{s_1}{s_2}\right)^2\right) \\ &= \varepsilon_{GM}\left(x_i^{(2)}, y_i^{(2)}, [dx, dy], s_1\right) \cdot \underbrace{\left(\frac{s_1}{s_2}\right)^2}_{<1} \end{aligned} \tag{21}$$

Therefore, by using a dyadic pyramid, the truncation error of the Taylor approximation, related to the translation parameters (Eq. 21) is decreased by a factor of 4 in each increase of the pyramid's scale. The approximation error related to the motion parameters which are scale-invariant, such as scale and shear is not reduced.

C. Dominant motion locking

Burt et. al. [18] used a frequency domain analysis to show that the coarse-to-fine refinement process allows the GM to lock onto a single dominant motion even when multiple motions are present. This property is essential for most image registration based applications [4], [10], [11]. We utilize the method presented in section III 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 common to I_1 and I_2
S_{Dom}	A subset of S. The set of pixels that are common to I_1 and I_2, whose motion is the *dominant motion.*
S_{NonD}	A subset of S. The 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. 5 according to the ith pixel's relation to either S_{Dom} or S_{NonD} we get:

$$\widetilde{H}p = \left[\begin{array}{c} \widetilde{H}_{Dom} \\ \widetilde{H}_{NonD} \end{array}\right] p = \left[\begin{array}{c} I_t^{Dom} \\ I_t^{NonD} \end{array}\right] = \widetilde{I}_t \tag{22}$$

where

$$\widetilde{H}_{Dom} p = I_t^{Dom} \tag{23}$$

are the equations related to dominant motion, and

$$\widetilde{H}_{NonD} p = I_t^{NonD} \tag{24}$$

are the equations related to non-dominant motion. As the GM algorithm converges to the dominant motion, the term I_t^{NonD} becomes the difference of uncorrelated pixels. Therefore, I_t^{NonD} can be modelled as a uniformly distributed random variable with zero mean

$$E\left\{I_t^{NonD}\right\} = 0 \tag{25}$$

This is a *landslide* type phenomenon: as the iterative solution p_n gets closer to the dominant motion's true parameters p^{Dom}, the non-dominant pixels become more and more uncorrelated. Inserting Eq. 25 into Eq. 8 we have

$$\begin{aligned} E\left\{\Delta p\right\} &= E\left\{\left(\widetilde{H}^T\widetilde{H}\right)^{-1}\widetilde{H}^T\widetilde{I}_t\right\} \\ &= E\left\{\left(\widetilde{H}^T\widetilde{H}\right)^{-1}\widetilde{H}^T\left[\begin{array}{c} I_t^{Dom} \\ \underline{0} \end{array}\right]\right\} \\ &+ \underbrace{E\left\{\left(\widetilde{H}^T\widetilde{H}\right)^{-1}\widetilde{H}^T\left[\begin{array}{c} \underline{0} \\ I_t^{NonD} \end{array}\right]\right\}}_{=0} \\ &= p^{Dom}. \end{aligned} \tag{26}$$

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

IV. Improved GM convergence using bidirectional formulations

In order to improve the convergence properties of the GM algorithm, we consider the registration of the translated one-dimensional signals $I_1(x)$ and $I_2(x)$

$$I_1\left(x_i^{(1)}\right) = I_2\left(x_i^{(2)}\right) \tag{27}$$

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

$$x_i^{(1)} = x_i^{(2)} + \Delta x. \tag{28}$$

Similar to section II, Eq. 27 is solved by expanding I_2 in a first order Taylor expansion (see Fig. 5(a)) and solving for Δx

$$I_1\left(x_i^{(1)}\right) = I_2\left(x_i^{(2)}\right) + \frac{\partial I_2\left(x_i^{(2)}\right)}{\partial x}\cdot\Delta x. \tag{29}$$

This point-wise expansion causes an error given in Eq. 17

$$\varepsilon_{GM}\left(x_i^{(2)}, \Delta x\right) = \frac{1}{2}\frac{\partial^2 I_2\left(\widetilde{x}^{(2)}\right)}{\partial x^2}\Delta x_i^2, \;\; \widetilde{x} \in [0, \Delta x]. \tag{30}$$

In order to lower the estimation error $\varepsilon_{GM}\left(x_i^{(2)}, \Delta x\right)$, Eq. 27 is reformulated symmetrically in respect to Δx according to Fig. 5(b)

$$I_1\left(x_i^{(1)} - \Delta X/2\right) = I_2\left(x_i^{(2)} + \Delta X/2\right). \tag{31}$$

Expanding both sides using Taylor expansion we have

$$\begin{aligned} &I_1\left(x_i^{(1)}\right) - \frac{\partial I_1\left(x_i^{(1)}\right)}{\partial x}\frac{\Delta x}{2} + \varepsilon_{GM}\left(x_i^{(1)}, \Delta X/2\right) \\ &= I_2\left(x_i^{(2)}\right) + \frac{\partial I_2\left(x_i^{(2)}\right)}{\partial x}\frac{\Delta x}{2} + \varepsilon_{GM}\left(x_i^{(2)}, \Delta X/2\right). \end{aligned} \tag{32}$$

$$\begin{aligned} &\underbrace{I_2\left(x_i^{(2)}\right) - I_1\left(x_i^{(1)}\right)}_{=-I_t} + \left(\frac{\partial I_2\left(x_i^{(2)}\right)}{\partial x} + \frac{\partial I_1\left(x_i^{(1)}\right)}{\partial x}\right)\frac{\Delta x}{2} \\ &+ \underbrace{\varepsilon_{GM}\left(x_i^{(2)}, \Delta X/2\right) - \varepsilon_{GM}\left(x_i^{(1)}, \Delta X/2\right)}_{=\varepsilon_{SGM}\left(x_i^{(2)}, \Delta x\right)} = 0. \end{aligned} \tag{33}$$

The error term is quadratic in Δx and by comparing ε_{SGM} to the regular GM error ε_{GM} of Eq. 30 we get

$$\varepsilon_{GM}\left(x_i^{(2)}, \Delta X/2\right) = \frac{1}{4}\varepsilon_{GM}\left(x_i^{(2)}, \Delta x\right). \tag{34}$$

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

$\varepsilon_{GM}\left(x_i^{(2)}, \Delta X/2\right)$ and $\varepsilon_{GM}\left(x_i^{(1)}, \Delta X/2\right)$, the error terms in Eq. 33 are uncorrelated. Thus an upper bound of the error associated with Eq. 32 is deduced

$$\varepsilon_{SGM}\left(x_i^{(2)}, \Delta x\right) \leq 2\varepsilon_{GM}\left(x_i^{(2)}, \Delta X/2\right). \tag{35}$$

Further analysis can be derived by expanding

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

The smaller the linearization error ε_{SGM}, the better the convergence rate. If $\varepsilon_{SGM} = 0$ Eq. 29 converges in a single iteration. Thus, in order to further decrease the linearization error the interval $[0 \ldots \Delta x]$ is optimally partitioned using the following formulation based on Fig. 5(c)

$$I_1(x_i^{(1)} - \Delta x_1) = I_2(x_i^{(2)} + \Delta x_2). \tag{36}$$

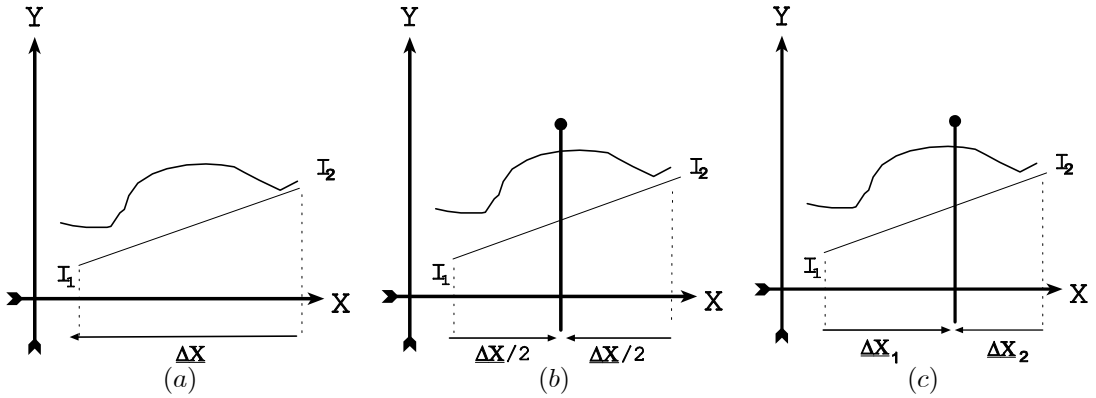


Fig. 5. 1D illustration of GM techniques. (a) Regular GM: pixels in I_1 are approximate by pixels in I_2 over the interval Δx. (b) Symmetric GM (SGM): pixels in the middle of the interval between I_1 and I_2 ($^{\Delta X}/_2$) are approximated by common pixels in I_1 and I_2. (c) Bidirectional GM (BDGM): 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.

Using Eq. 30 and a Taylor expansion of Eq. 36 we have

$$I_1\left(x_i^{(1)}\right)-\frac{\partial I_1\left(x_i^{(1)}\right)}{\partial x}\Delta x_1+\varepsilon_{GM}\left(x_i^{(1)},\Delta x_1\right)= I_2\left(x_i^{(2)}\right)+\frac{\partial I_2(x_i^{(2)})}{\partial x}\Delta x_2+\varepsilon_{GM}\left(x_i^{(2)},\Delta x_2\right). \quad (37)$$

$$I_2\left(x_i^{(2)}\right)-I_1\left(x_i^{(1)}\right)+\frac{\partial I_2\left(x_i^{(2)}\right)}{\partial x}\Delta x_1+\frac{\partial I_1\left(x_i^{(1)}\right)}{\partial x}\Delta x_2 + \underbrace{\varepsilon_{GM}\left(x_i^{(2)},\Delta x_2\right)-\varepsilon_{GM}\left(x_i^{(1)},\Delta x_1\right)}_{=\varepsilon_{BDGM}\left(x_i^{(1)},x_i^{(2)},\Delta x_1,\Delta x_2\right)}=0, \quad (38)$$

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

$$\Delta x=\Delta x_1+\Delta x_2 \quad (39)$$

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

Next, we analyze the error term given in Eq. 38

$$\begin{aligned}
&2\cdot\varepsilon_{BDGM}\left(x_i^{(1)},x_i^{(2)},\Delta x_1,\Delta x_2\right)=\\
&\quad=2\left(\varepsilon_{GM}\left(x_i^{(2)},\Delta x_2\right)-\varepsilon_{GM}\left(x_i^{(1)},\Delta x_1\right)\right)\\
&\quad=\frac{\partial^2 I_2\left(\widetilde{x}^{(2)}\right)}{\partial x^2}\Delta x_2^2-\frac{\partial^2 I_1\left(\widetilde{x}^{(1)}\right)}{\partial x^2}\Delta x_1^2\\
&\quad=\frac{\partial^2 I_2\left(\widetilde{x}^{(2)}\right)}{\partial x^2}\Delta x_2^2+\frac{\partial^2 I_1\left(\widetilde{x}^{(1)}\right)}{\partial x^2}\Delta x_1^2\\
&\quad\underbrace{-\frac{\partial^2 I_2\left(\widetilde{x}^{(2)}\right)}{\partial x^2}\Delta x_1^2+\frac{\partial^2 I_2\left(\widetilde{x}^{(2)}\right)}{\partial x^2}\Delta x_1^2}_{=0}\\
&\quad=\frac{\partial^2 I_2\left(\widetilde{x}^{(2)}\right)}{\partial x^2}\left(\Delta x_2^2-\Delta x_1^2\right)\\
&\quad+\Delta x_1^2\left(\frac{\partial^2 I_2\left(\widetilde{x}^{(2)}\right)}{\partial x^2}-\frac{\partial^2 I_1\left(\widetilde{x}^{(1)}\right)}{\partial x^2}\right)\\
&=\frac{\partial^2 I_2\left(\widetilde{x}^{(2)}\right)}{\partial x^2}\left(\Delta x_2-\Delta x_1\right)\left(\Delta x_2+\Delta x_1\right)\\
&\qquad+\Delta x_1^2\left(\frac{\partial^3 I_2(\widetilde{x})}{\partial^3 x}\Delta x\right) \quad (40)
\end{aligned}$$

Substituting Eq. 39 into Eq. 40 we have

$$\varepsilon_{BDGM}\left(x_i^{(1)},x_i^{(2)},\Delta x_1,\Delta x_2\right)= \frac{\Delta x}{2}\left(\frac{\partial^2 I_2\left(\widetilde{x}^{(2)}\right)}{\partial x^2}\cdot\left(\Delta x_2-\Delta x_1\right)+\Delta x_1^2\frac{\partial^3 I_2(\widetilde{x})}{\partial x^3}\right) \quad (41)$$

For the symmetric case where $\Delta x_2=\Delta x_1=\frac{\Delta x}{2}$, the first term of Eq. 41 vanishes and ε_{BDGM} identifies with ε_{SGM}.

An extension to 2D with general motion models is given in sections IV-A and IV-B for the SGM and BDGM algorithms, respectively.

A. Symmetric GM (SGM)

In the general 2D case, the SGM is formulated using the motion parameters vector p (see Fig. 5(b))

$$p^*=\arg\min_p\sum_{(x_1,y_1)\in S}\left(I_2\left(x_i^{(2)},y_i^{(2)},{}^p/_2\right)-I_1\left(x_i^{(1)},y_i^{(1)},-{}^p/_2\right)\right)^2. \quad (42)$$

and

$$r_i\left(x_i^{(1)},y_i^{(1)},p\right)=I_2\left(x_i^{(2)},y_i^{(2)},{}^p/_2\right)-I_1\left(x_i^{(1)},y_i^{(1)},-{}^p/_2\right). \quad (43)$$

From Eq. 43 we have

$$\frac{\partial r_i\left(x_i^{(1)},y_i^{(1)},p\right)}{\partial p}= \frac{\partial I_2\left(x_i^{(2)},y_i^{(2)},{}^p/_2\right)}{\partial p}-\frac{\partial I_1\left(x_i^{(1)},y_i^{(1)},-{}^p/_2\right)}{\partial p}. \quad (44)$$

Using the chain rule

$$\frac{\partial I_1\left(x_i^{(1)},y_i^{(1)},{}^p/_2\right)}{\partial p}=\frac{1}{2}\cdot\frac{\partial I_1\left(x_i^{(1)},y_i^{(1)},p\right)}{\partial p}. \quad (45)$$

Therefore, we have

$$\frac{\partial r_i\left(x_i^{(1)},y_i^{(1)},p/2\right)}{\partial p}= \frac{1}{2}\left(\frac{\partial I_2\left(x_i^{(2)},y_i^{(2)},p/2\right)}{\partial p}+\frac{\partial I_1\left(x_i^{(1)},y_i^{(1)},p/2\right)}{\partial p}\right) \quad (46)$$

and by assuming

$$\frac{\partial I_2\left(x_i^{(2)},y_i^{(2)},p/2\right)}{\partial p}\approx\frac{\partial I_1\left(x_i^{(1)},y_i^{(1)},p/2\right)}{\partial p} \quad (47)$$

we get

$$\frac{\partial r_i\left(x_i^{(1)},y_i^{(1)},p/2\right)}{\partial p}\approx\frac{\partial I_2\left(x_i^{(2)},y_i^{(2)},p/2\right)}{\partial p}. \quad (48)$$

Taking the second derivative and using Eq. 47 we have

$$\frac{\partial^2 r_i\left(x_i^{(1)},y_i^{(1)},p/2\right)}{\partial p^2}= = \frac{1}{2}\left(\frac{1}{2}\cdot\frac{\partial^2 I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial p^2}+\frac{1}{2}\cdot\frac{\partial^2 I_1\left(x_i^{(1)},y_i^{(1)}\right)}{\partial p^2}\right) \approx \frac{1}{2}\frac{\partial^2 I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial p^2} \quad (49)$$

and by comparing Eq. 49 to Eqs. 11 and 15 we get:

$$C_1^{SGM}=\frac{C_1^{GM}}{2} \quad (50)$$

$$C_2^{SGM}=C_2^{GM} \quad (51)$$

Therefore, the convergence rate of the SGM is twice faster than that of the GM for large motions, due to its reduced linearization error. For small motions (Eq. 51) the SGM and GM show similar convergence rates.

Algorithm flow

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

1) The matrix $\left(H^TH\right)$ is computed separately for I_2 and I_1 according to Eq. 9:

$$\left(H^TH\right)_{k,j}^{I_1}=\sum_i\frac{\partial I_1\left(x_i^{(1)},y_i^{(1)}\right)}{\partial p_k}\frac{\partial I_1\left(x_i^{(1)},y_i^{(1)}\right)}{\partial p_j} \quad (52)$$

$$\left(H^TH\right)_{k,j}^{I_2}=\sum_i\frac{\partial I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial p_k}\frac{\partial I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial p_j} \quad (53)$$

2) We solve the equation

$$\left(H^TH\right)^{SGM}p^{SGM}=H^TI_t \quad (54)$$

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

$$\left(H^TH\right)_{k,j}^{SGM}=\frac{1}{2}\left(\left(H^TH\right)_{k,j}^{I_1}+\left(H^TH\right)_{k,j}^{I_2}\right) \quad (55)$$

and $\left(H^TI_t\right)$ is calculated according to Eq. 7.

3) The SGM returns p^{SGM} as the result.

B. Bidirectional Gradient Methods (BDGM)

The BDGM uses a different formulation of Eq. 2 illustrated in Fig. 5(c)

$$p^*=\arg\min_p\sum_{(x_1,y_1)\in S}\left(I_2\left(x_i^{(2)},y_i^{(2)},p_2\right)-I_1\left(x_i^{(1)},y_i^{(1)},-p_1\right)\right)^2 \quad (56)$$

where p_1 and p_2 have the same dimensions as the motion parameters vector used in the GM and SGM and the overall motion is given by

$$p=p_1+p_2. \quad (57)$$

Let $k,m\in[0\ldots1]$, $k+m=1$, then:

$$\begin{aligned}p_1&=k\cdot p\\ p_2&=m\cdot p,\end{aligned} \quad (58)$$

$$r_i\left(x_i^{(1)},y_i^{(1)},p\right)=I_2\left(x_i^{(1)},y_i^{(1)},p_2\right)-I_1\left(x_i^{(2)},y_i^{(2)},-p_1\right) =I_2\left(x_i^{(1)},y_i^{(1)},k\cdot p\right)-I_1\left(x_i^{(2)},y_i^{(2)},-m\cdot p\right), \quad (59)$$

$$\frac{\partial r_i\left(x_i^{(1)},y_i^{(1)},p\right)}{\partial p}= k\frac{\partial I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial p}+m\frac{\partial I_1\left(x_i^{(1)},y_i^{(1)}\right)}{\partial p}, \quad (60)$$

and

$$\frac{\partial^2 r_i\left(x_i^{(1)},y_i^{(1)},p\right)}{\partial p^2}= k^2\frac{\partial^2 I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial p^2}+m^2\frac{\partial^2 I_1\left(x_i^{(1)},y_i^{(1)}\right)}{\partial p^2}. \quad (61)$$

By assuming symmetry in Eq. 47 we get

$$\frac{\partial r_i\left(x_i^{(1)},y_i^{(1)},p\right)}{\partial p}=(k+m)\frac{\partial I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial p} =\frac{\partial I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial p} \quad (62)$$

$$\frac{\partial^2 r\left(x_i^{(1)},y_i^{(1)},p\right)}{\partial p^2}=\left(k^2+m^2\right)\frac{\partial^2 I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial p^2} =\left(2k^2-2k+1\right)\frac{\partial^2 I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial p^2} \quad (63)$$

For small motions, similar to Eq. 51 we have

$$C_2^{BDGM}=C_2^{GM} \quad (64)$$

and the optimal partitioning of the interval $[0 \dots p]$, which minimizes $\left|\frac{\partial^2 r\left(x_i^{(1)},y_i^{(1)},p\right)}{\partial p^2}\right|$, is the symmetric approach $\left(k=\frac{1}{2}\right)$ described in section IV-A. Furthermore, the partitioning used by the regular GM is worse than any of the bidirectional formulation, since $\left|\frac{\partial^2 r\left(x_i^{(1)},y_i^{(1)},p\right)}{\partial p^2}\right|$ 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. In practice, the symmetry assumption is violated for large motions as non corresponding pixels are aligned

$$I_2\left(x_i^{(2)},y_i^{(2)},{}^{p}\!/_{2}\right) \neq I_1\left(x_i^{(1)},y_i^{(1)},-{}^{p}\!/_{2}\right) \Longrightarrow \frac{\partial I_2\left(x_i^{(2)},y_i^{(2)},{}^{p}\!/_{2}\right)}{\partial p} \neq \frac{\partial I_1\left(x_i^{(1)},y_i^{(1)},-{}^{p}\!/_{2}\right)}{\partial p}. \quad (65)$$

Then we get

$$C_1^{BDGM} \leq C_1^{SGM} = \frac{1}{2} C_1^{GM}. \quad (66)$$

Algorithm flow

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

1) The matrix H is computed separately for I_2 and I_1 according to Eq. 6:

$$H_{i,j}^{I_1} = \frac{\partial I_1\left(x_i^{(1)},y_i^{(1)}\right)}{\partial p_j} \quad (67)$$

$$H_{i,j}^{I_2} = \frac{\partial I_2\left(x_i^{(2)},y_i^{(2)}\right)}{\partial p_j} \quad (68)$$

2) H^{BDGM} is formed by:

$$H^{BDGM} = \left[\begin{array}{cc} H^{I_1} & H^{I_2} \end{array}\right] \quad (69)$$

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 p^{BDGM} the BDGM parameters vector, then

$$p^{BDGM} = \left[\begin{array}{c} p^1 \\ p^2 \end{array}\right] \quad (70)$$

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

4) We solve the equation

$$\left(\left(H^{BDGM}\right)^T H^{BDGM}\right) p^{BDGM} = \left(H^{BDGM}\right)^T I_t \quad (71)$$

where I_t is similar to the one used in section II-A.

5) After solving Eq. 71, the solution p^{BDGM} is given by:

$$p^{BDGM} = p^1 + p^2 \quad (72)$$

V. Experimental Results

This section describes the performance of the proposed new techniques. The numerical 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 initial 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. The mean value of the initial motion parameters was recovered using phase-correlation [14] and the set of initial estimates was constructed around it. Thus, the alignment error at each iteration is the average of all the simulations. The same implementations of the *iterative refinement* (section II-B) and *multiscale embedding* (section II-C) 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 IV-A and IV-B for the SGM and BDGM, respectively. The resolution pyramid was constructed using a three-tap filter $\left[\begin{array}{ccc} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{array}\right]$ and the derivative was approximated using $\left[\begin{array}{ccc} \frac{1}{2} & 0 & -\frac{1}{2} \end{array}\right]$. Following [8], [9], other filters were tested with no significant improvement of the results. The initial estimate of the motion 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.

A. Estimation of rotations

The registration of rotated images was studied using the images in Fig. 6. Figure 6a was rotated and the background areas created by the rotation were padded with zeros. The registration was calculated using a linearized rotation model

$$\begin{array}{l} x_1 = a \cdot x_2 + b \cdot y_2 + c \\ y_1 = -b \cdot x_2 + a \cdot y_2 + d \end{array} \quad (73)$$

Figure 7(a) shows the performance of registering an image rotated by 10°. 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. 47) due to the zero padding.

B. Estimation of small affine motion

According to Eqs. 50 and 66 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:

$$\begin{array}{l} x_1 = a \cdot x_2 + b \cdot y_2 + c \\ y_1 = d \cdot x_2 + e \cdot y_2 + f \end{array} \quad (74)$$

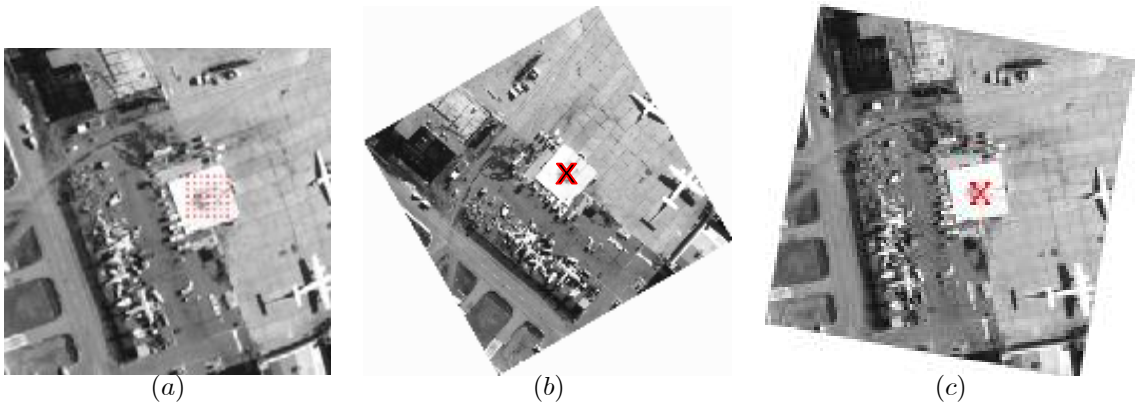


Fig. 6. Test images for rotation registration. The red dots in image (a) are the initial estimates of the red X in image (b). X marks the initial motion (a) original airport image. (b) airport image rotated by 10°. (c) airport image rotated by 30°.

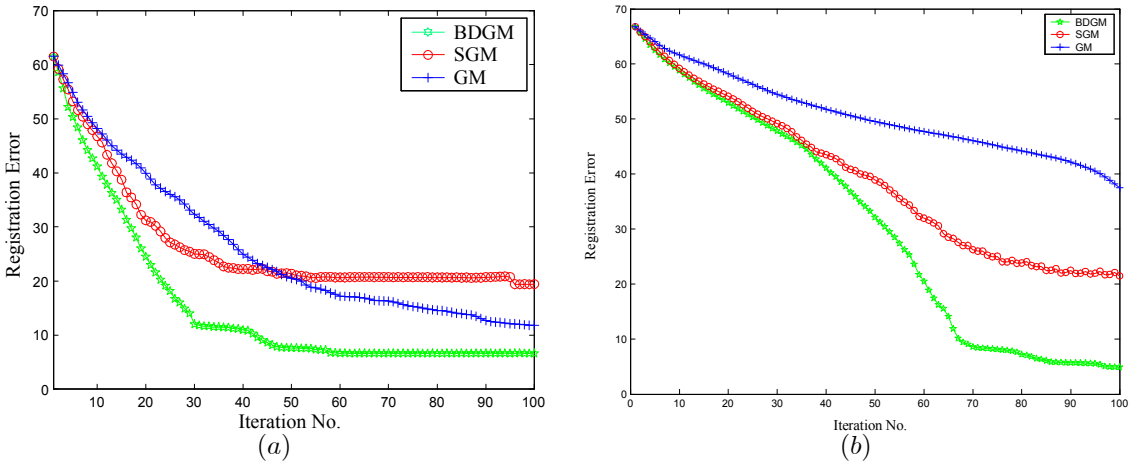


Fig. 7. Registration results of the rotated images: (a) 10° rotation (b) 30° rotation. In both case the BDGM and SGM converged faster than the regular GM.

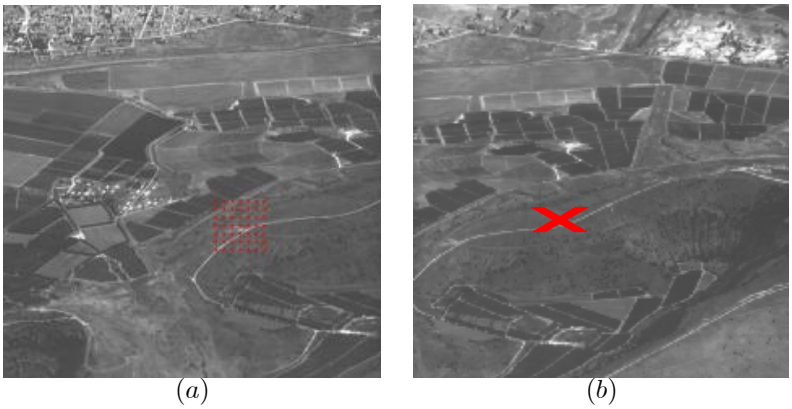


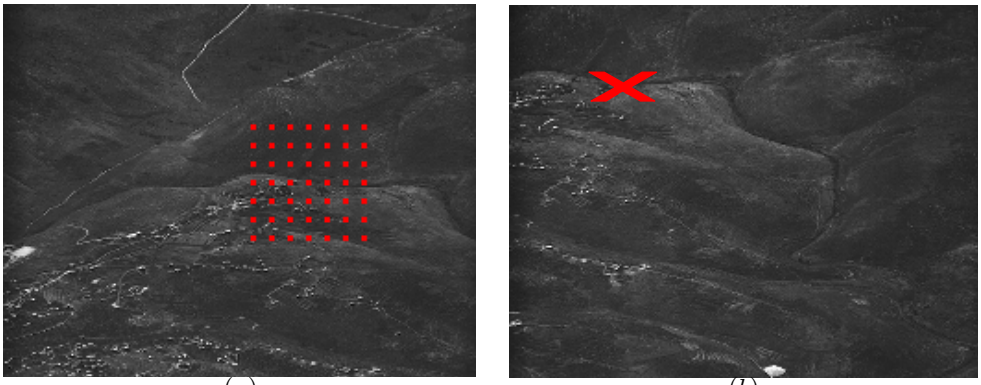
Fig. 8. Test images for affine registration with small motion. The red dots in image (a) are initial estimates of the red X in image (b) which were used in the simulations.

The results presented in Fig. 9 show the improved convergence achieved by the SGM and BDGM compared to the regular GM algorithm. Since Fig. 9 shows the *average* alignment error, the final GM error is higher, as for some of the initial estimates, the GM diverged. 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).

C. Registration of low contrast images

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:

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



(a) (b)

Fig. 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.

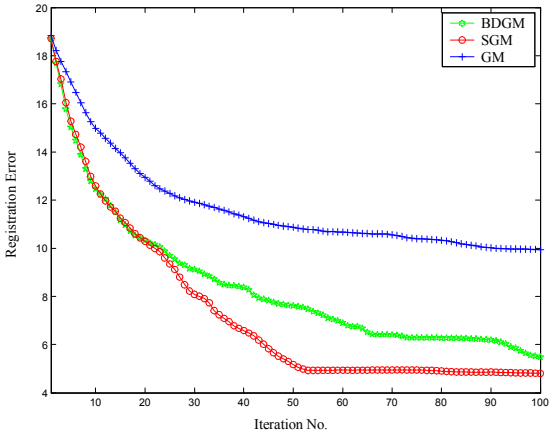


Fig. 9. Registration results of small affine motion.

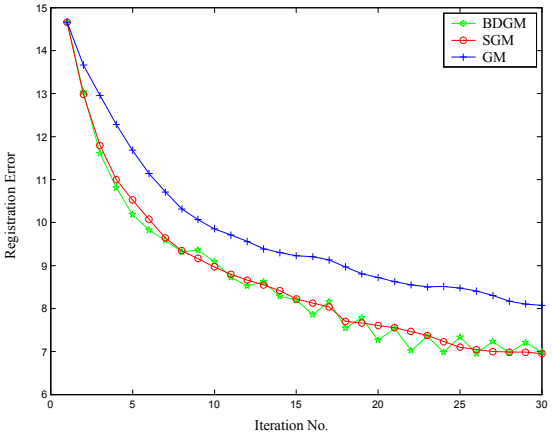


Fig. 11. Registration performance under poor illumination conditions: The regular GM diverges, while the SGM and BDGM converge. Due to the small motion, the SGM converges better than the BDGM, which is unstable due to its larger number of unknowns.

Therefore, according to Eqs. 15 and 16 the convergence rate of the GM deteriorates. The images presented in Fig. 10 are real airborne images, which are registered using the affine motion model defined in Eq. 74.

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 V-B. This phenomenon could have been avoided, by switching from the BDGM to either the SGM or GM near the proximity of the solution.

D. Estimation of panoramic motion

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

$$\begin{aligned} x_1 &= \frac{a\cdot x_2+b\cdot y_2+c}{g\cdot x_2+h\cdot y_2+1} \\ y_1 &= \frac{d\cdot x_2+e\cdot y_2+f}{g\cdot x_2+h\cdot y_2+1} \end{aligned} \tag{75}$$

Due to large number of unknowns and the non-linear nature of Eq. 75, the GM based registration becomes slow and unstable. Two sets of images photographed by a regular 35mm cameras were used to compare the performances 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 V-B where the BDGM and SGM coincide and converge twice as fast (5 iterations) as the GM (9 iterations).

VI. 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 which are verified by the experimental results. Future work includes the application of the BDGM and SGM to other GM

(a) (b)

Fig. 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.

(a) (b)

Fig. 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.

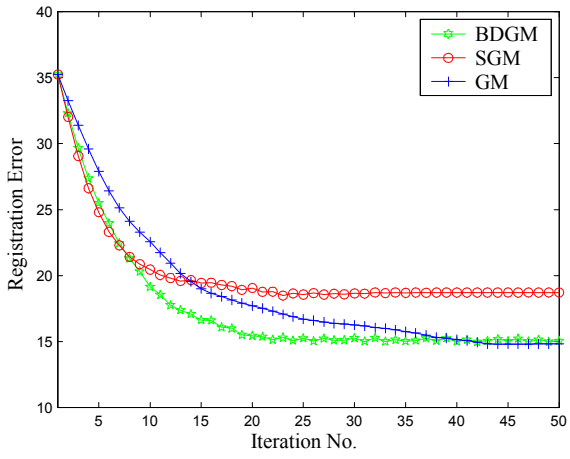


Fig. 13. Registration results for large panoramic motion: the SGM and BDGM converge twice as fast as the regular GM. Due to the large motion the symmetric assumption (Eq. 47) is violated and the BDGM converges better than the SGM.

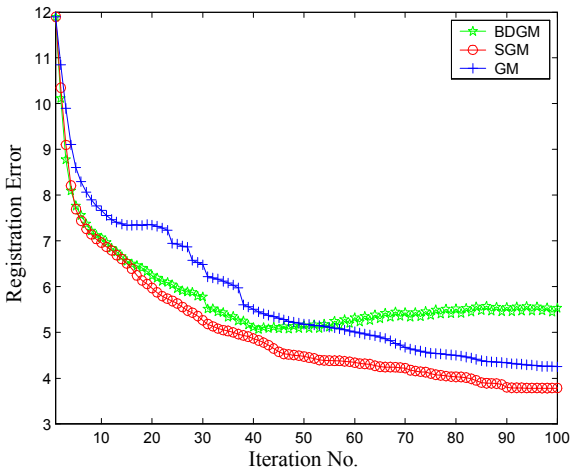


Fig. 15. Registration results for small panoramic motion: the SGM and BDGM converge twice as fast as the regular GM. Due to the small motion the symmetric assumption (Eq. 47) is valid and the SGM and BDGM converge similarly.

based algorithms such as direct estimation of 3D structure [16].

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

A. Definitions

The general least square problem (LS) is defined as

$$x^* = \min_{\underline{x}} \{f(x)\} \tag{76}$$

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

$$\begin{aligned} f(x) &= \sum_{n=1}^{m} [r_n(x)]^2 = [r_1(x) \dots r_m(x)] \begin{bmatrix} r_1(x) \\ r_2(x) \\ \vdots \\ r_m(x) \end{bmatrix} \quad (77) \\ &= R^T(x) \cdot R(x). \end{aligned}$$

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

$$\frac{\partial f(x)}{\partial x} = 2\sum_{n=1}^{m} \frac{\partial r_n(x)}{\partial x} r_n(x) = 2\begin{bmatrix} \frac{\partial r_1(x)}{\partial x} & \frac{\partial r_2(x)}{\partial x} & \dots & \frac{\partial r_m(x)}{\partial x} \end{bmatrix} \cdot R(x) = 2A(x) R(x) \quad (78)$$

$$\begin{aligned} \frac{\partial^2 f(x)}{\partial x_i^{(2)}} &= 2\sum_{n=1}^{m} \frac{\partial r_n(x)}{\partial x} \frac{\partial r_n^T(x)}{\partial x} + 2\sum_{n=1}^{m} \frac{\partial^2 r_n(x)}{\partial x^2} r_n(x) \\ &= 2A(x) A^T(x) + 2\sum_{n=1}^{m} \frac{\partial^2 r_n(x)}{\partial x^2} r_n(x). \quad (79) \end{aligned}$$

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

$$x_{k+1} = x_k - \left[A(x_k) A^T(x_k)\right]^{-1} A(x_k) R(x_k) \quad (80)$$

where x_k is the parameters vector estimated at iteration k and

$$\varepsilon_k = x_k - x^* \quad (81)$$

and ε_k is the estimation error at iteration k.

B. Convergence analysis

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

$$\begin{aligned} \frac{\partial f(\tilde{x})}{\partial x} \approx \frac{\partial f(x)}{\partial x} + \frac{\partial^2 f(x)}{\partial x^2} \cdot (\hat{x} - x) \\ + \frac{1}{2} \frac{\partial^3 f(\tilde{x})}{\partial x^3} \cdot (\hat{x} - x)^2, \tilde{x} \in [\hat{x}, x]. \quad (82) \end{aligned}$$

Inserting Eqs. 78 and 79 into Eq. 82 we get

$$\begin{aligned} \frac{\partial f(\tilde{x})}{\partial x} \approx 2A(x) R(x) + \\ \left\{ 2A(x) A^T(x) + 2\sum_{n=1}^{m} \frac{\partial^2 r_n(x)}{\partial x^2} r_n(x) \right\} \cdot (\hat{x} - x) \\ + \frac{f^{(3)}(\tilde{x})}{2} \cdot (\hat{x} - x)^2. \quad (83) \end{aligned}$$

Using Eq. 83 we estimate the gradient at the minimum point x^* using a Taylor approximation around x_k:

$$\begin{aligned} A(x^*) R(x^*) = A(x_k) R(x_k) \\ - \left[A(x_k) A^T(x_k) + \sum_{n=1}^{m} \frac{\partial r_n^2(x)}{\partial x^2} r_n(x) \right] \cdot \varepsilon_k + O\left({\varepsilon_k}^T \varepsilon_k\right). \end{aligned} \quad (84)$$

Rewriting Eq. 80 for ε_k we get:

$$\varepsilon_{k+1} = \varepsilon_k - \left[A(x_k) A^T(x_k)\right]^{-1} A(x_k) R(x_k). \quad (85)$$

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

$$\varepsilon_k = \left[A(x_k) A^T(x_k)\right]^{-1} \left[A(x_k) A^T(x_k) \cdot \varepsilon_k + \underbrace{A(x^*) R(x^*)}_{=0} \right]. \quad (86)$$

inserting Eq. 86 into Eq. 85 we get

$$\begin{aligned} \varepsilon_{k+1} = \\ = \left[A(x_k) A^T(x_k)\right]^{-1} \left[A(x_k) A^T(x_k) \cdot \varepsilon_k + A(x^*) R(x^*)\right] \\ - \left[A(x_k) A^T(x_k)\right]^{-1} A(x_k) R(x_k) \\ = \left[A(x_k) A^T(x_k)\right]^{-1} \cdot \\ \cdot \left[A(x_k) A^T(x_k) \cdot \varepsilon_k + A(x^*) R(x^*) - A(x_k) R(x_k)\right]. \end{aligned} \quad (87)$$

Inserting Eq. 84 into Eq. 87 we get

$$\begin{aligned} \varepsilon_{k+1} = -\left[A(x_k) A^T(x_k)\right]^{-1} \left(\sum_{n=1}^{m} \frac{\partial r_n^2(x_k)}{\partial x^2} r_n(x_k) \right) \cdot \varepsilon_k - \\ \left[A(x_k) A^T(x_k)\right]^{-1} \cdot O\left({\varepsilon_k}^T \varepsilon_k\right). \end{aligned} \quad (88)$$

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

$$\begin{aligned} \|\varepsilon_{k+1}\| \leq \left\| \left[A(x_k) A^T(x_k)\right]^{-1} \sum_{n=1}^{m} \frac{\partial r_n^2(x_k)}{\partial x^2} r_n(x_k) \right\| \cdot \|\varepsilon_k\| \\ + \left\| \left[A(x_k) A^T(x_k)\right]^{-1} \right\| \cdot O\left(\|\varepsilon_k\|^2\right). \end{aligned} \quad (89)$$

In other words,

$$\|\varepsilon_{k+1}\| \leq C_1 \cdot \|\varepsilon_k\| + C_2 \cdot \|\varepsilon_k\|^2. \quad (90)$$

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

$$r(x^*) = r(x_k) + A(x_k) \cdot (x^* - x_k).$$

Then, by taking the L_2 norm we get:

$$\|\Delta r_k\| = \|r(x^*) - r(x_k)\| = A(x_k) \|\varepsilon_k\| \quad (91)$$

where ε_k is defined as in 81.

Rewriting Eq. 91 for $\|\Delta r_{k+1}\|$:

$$\|\Delta r_{k+1}\| = \|r(x^*) - r(x_{k+1})\| = A(x_{k+1}) \|\varepsilon_{k+1}\| \quad (92)$$

and assuming a small refinement step: $A(x_{k+1}) \approx A(x_k)$, then

$$\|\Delta r_{k+1}\| = \frac{\|\varepsilon_{k+1}\|}{\|\varepsilon_k\|} \cdot \|\Delta r_k\| \quad (93)$$

We conclude that the parameters error ε_k and the objective function Δr_k have a similar convergence rate.

D. Conclusion

Using Eq. 89 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:

In this phase the convergence is dominated by the linear convergence term C_1 (Eq. 90).

$$\|\varepsilon_{k+1}\| \leq C_1 \cdot \|\varepsilon_k\| \tag{94}$$

Therefore we have $C_1 \gg C_2$:

$$\left\| \sum_{k=1}^{m} \frac{\partial r_k^2(x_k)}{\partial x^2} r_k(x_k) \right\| \gg \left\| \sum_{k=1}^{m} \left(\frac{\partial r_k(x_k)}{\partial x} \right)^2 \right\| \tag{95}$$

and the observation error term $r_k(x_k)$ satisfies

$$r_k(x_k) \gg 1 \tag{96}$$

Equation 96 characterizes situations in which there is a significant discrepancy in the minimization model defined in Eq. 77 due to large deviations of the estimated parameters x_k from the true parameters x^*. For $\|C_1\| > 1$ the process diverges.

Close range phase:

In regions in proximity of the solution $r_k(x_k) \to 0$ and $C_1 \to 0$. The second term C_2 in Eq. 90 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, pp. 363–367, 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",Workshop on Application of Computer Vision 1994, 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, pp. 992-1015, September 2000.

[17] M. Irani and P. Anandan. "All About Direct Methods", International Workshop on Vision Algorithms, Corfu, Greece, September 1999.

[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.

[19] A. Fusiello, E. Trucco, T. Tommasini, V. Roberto, "Improving Feature Tracking with Robust Statistics", Pattern Analysis and Applications Vol. 2, No. 4, 1999, pp. 312-320 .

[20] T. Tommasini, A. Fusiello, E. Trucco, and V. Roberto, "Making good features to track better", in Proceedings IEEE Computer Society Conference on Computer Vision Pattern Recognition, 1998, pp. 145–149.

[21] B. Matei and P. Meer, "Optimal Rigid Motion Estimation and Performance Evaluation with Bootstrap", Computer Vision and Pattern Recognition Conference (CVPR), Fort Collins, CO, June 1999, vol. 1, pp. 339-345.

Yosi Keller received the B.Sc. degree in electrical engineering in 1994 from The Technion-Israel Institute of Technology, Haifa. He received the M.Sc and Ph.D degree in electrical engineering from Tel-Aviv University, Tel-Aviv, in 1998 and 2003, respectively. From 1994 to 1998, he was an R&D Officer in the Israeli Intelligence Force. He is a visiting Assistant Professor with the Department of Mathematics, Yale University. His research interests include motion estimation, video analysis, image restoration and statistical pattern analysis.

Amir Averbuch was born in Tel Aviv, Israel. He received the B.Sc and M.Sc degrees in mathematics from the Hebrew University in Jerusalem, Israel in 1971 and 1975, respectively. He received the Ph.D degree in Computer Science from Columbia University, New York, in 1983. From 1966-1970 and 1973-1976 he served in the Israeli Defense Forces. From 1976-1986 he was a Research Staff Member at IBM T.J. Watson Research Center, Yorktown Heights, in NY, Department of Computer Science. In 1987, he joined the Department of Computer Science, School of Mathematical Sciences, Tel Aviv University, where he is now professor of computer science. His research interests include wavelets, signal/image processing, multiresolution analysis, numerical computation for the solutions of PDEs, scientific computing (fast algorithms), parallel and supercomputing (software and algorithms).