Computation of Three-dimentional Motions of Objects by Using A Fine-Grain Parallel GA

Fattaneh Taghiyareh* & Hiroshi Nagahashi**

***ftaghiyar@ut.ac.ir**
Electrical and Computer Engineering Dep.,
Faculty of Engineering,
University of Tehran, Iran
P.O Box : 14395/515
****longb@n.isl.titech.ac.jp Imaging Science and Eng. Laboratory**
Tokyo Institute of Technology
4259 Nagatsuta, Midori-ku,
Yokohama 226-8503 , Japan

ABSTRACT

Motion estimation is a very important problem in dynamic scene analysis. Although it is easier to estimate motion parameters from 3D data than from 2D images, it is not trivial since the 3D data we have are almost always corrupted by noise. We address the problem of computing the three-dimensional motions of objects. This paper proposes a robust approach to position estimation of moving objects by exploiting the only available geometric constraint, namely, the epipolar constraint. The extrinsic parameters of the camera and the motion of the stereo rig is unknown. If we make an exhaustive search for the epipolar geometry, the complexity is prohibitively high. The idea underlying our approach is to use a parallel fine-grain GA as an optimizer. Since the constraint on the rotation matrix is not fully exploited in the analytical method, nonlinear minimization can be used to improve the result.

keywords : parallel genetic algorithms, butterfly network, epipolar geometry, fine-grain GA, position estimation, fundamental matrix

1 Introduction

Evolutionary algorithms have gained a growing popularity in solving many complex problems from various application fields[1]. Considering image processing, which requires robust and fast techniques capable of managing large and noisy data, it would seem that genetic algorithms (GAs) are well suited and they have been successfully applied to various field of image processing[2].

Estimating the position of a moving object remains one of the bottlenecks in computer vision. A large number of work has been carried out, however the results are not satisfactory. The only geometric constraint we know between two images of a single object is *epipolar constraint*. However, when the motion between two images is not known, the epipolar geometry is also unknown. The methods reported in the literature all exploit some heuristics in one form or another, which are not general and can not be applied for all cases.

The classical approach to motion and structure estimation problem from two perspective projections consists of two stages: (i) using the 8-point algorithm to estimate the 9 essential parameters defined up to a scale factor, which is a linear estimation problem; (ii) refining the motion estimation based on some statistically optimal criteria, which is a nonlinear estimation problem on a five-dimensional space. Unfortunately, the results obtained using this approach are often not satisfactory, especially when the motion is small or when the observed points are close to a degenerate surface (e.g. plane). The problem is that the second stage is very sensitive to the initial guess, and that it is very difficult to obtain a precise initial estimate from the first stage. This is because we perform a projection of a set of quantities which are estimated in a space of 8 dimensions, much higher than that of the real space which is five-dimensional.

We propose in this paper a novel approach by introducing a method which applies a fine-grained parallel GA based on "Butterfly" topology. The algorithm deals with several subpopulations distributed in the butterfly network, which leads to overcoming the problem of premature convergence. The "Butterfly" is one of the most versatile and efficient networks yet discovered for parallel computation. Massive communications capability of the butterfly provides a way to disseminate good solutions across the entire population[3].

The organization of the paper is as follows: we begin with a brief review of the epipolar geometry. The simulated system along with its optical and geometrical parameters is then introduced, followed by a description of our fine-grain parallel GA and our measures. A pilot study performed to assess the effectiveness of the algorithm is then discussed. Finally, we present some tentative conclusions and recommendations for further study.

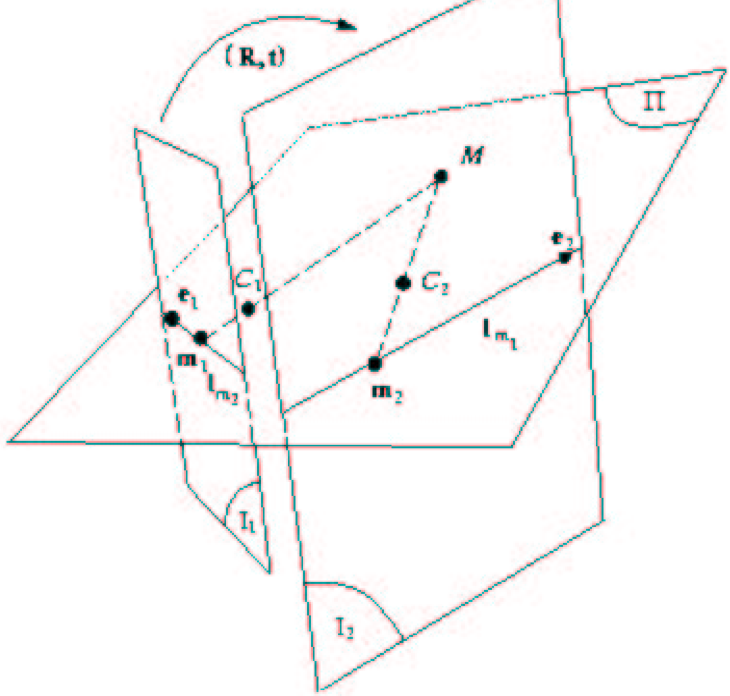

Figure 1: *The epipolar geometry*

2 Epipolar Geometry

Considering the case of two cameras as shown in Fig. 1. Let C_1 and C_2 be the optical centers of the first and second cameras, respectively. Given a point m_1 in the first image, its corresponding point in the second image is constrained to lie on a line called *the epipolar line* of m_1, denoted by l_{m_1}. the line l_{m_1} is the intersection of the plane II, defined by m_1, C_1 and C_2(known as the epipolar plane), with the second image plane I_2. This is because image point m_1 may correspond to an arbitrary point on the semi-line C_1M (M may be at infinity) and that the projection of C_1M on I_2 is the line l_{m_1}.

The corresponding point in the first image of each point m_{2k} lying on $l_{m_{1k}}$ must lie on the epipolar line $l_{m_{2k}}$, which is the intersection of the same plane II^k with the first image plane I_1. All epipolar lines form a pencil containing the epipole e_1, which is the intersection of the line C_1C_2 with the image plane I_1. The symmetry leads to the following observation. if m_1 (a point in I_1) and m_2 (a point in I_2) correspond to a single physical point M in space, then m_1, m_2 , C_1 and C_2 must lie in a single plane. This is the well-known *co-planarity constraint* or *epipolar equation* in solving motion and structure from motion problems when the intrinsic parameters of the cameras are known.

Let the displacement from the first camera to the second be (R, t). Let m_1 and m_2 be the images of a 3-d point M on the cameras . Without loss of generality, we assume that M is expressed in the coordinate frame of the first camera. Under the pinhole model, we have the following two equations:

$$\boldsymbol{s_1\tilde{m}_1 = A_1[I \ \ 0]\begin{bmatrix} M \\ 1 \end{bmatrix}}$$

$$\boldsymbol{s_2\tilde{m}_2 = A_2[R \ \ t]\begin{bmatrix} M \\ 1 \end{bmatrix}}$$

where A_1 and A_2 are the intrinsic matrices of the first and second cameras, respectively. Eliminating M, s_1 and s_2 from the above equations, we obtain the following fundamental equation

$$\boldsymbol{\tilde{m}_2^T A_2^{-T} T R A_1^{-1} \tilde{m}_1 = 0,} \tag{1}$$

which is another form of *epipolar equation*. T stands for an antisymmetric matrix defined by t such that $Tx = t \wedge x$ for all 3-D vector x ($\wedge$ denotes the cross product).

Equation (1) is a fundamental constraint underlying any two images if they are perspective projections of one and the same scene. Let $F = A_2^{-T}TRA_1^{-1}$, F is known as the fundamental matrix of the two images. It has only seven degrees of freedom. Indeed, it is only defined up to a scale factor and its determinant is zero. Geometrically, $F\tilde{m}_1$ defines the epipolar line of point m_1 in the second image.

It can be shown that the fundamental matrix F is related to the essential matrix

$$\boldsymbol{E = t_{\mathrm{x}} R} \tag{2}$$

by

$$F = A_2^{-T} E A_1^{-1}.$$

It is thus clear that if the camera are calibrated, the problem becomes the one of *motion and structure from motion*[4].

3 Geometric Model

We present a method to recover the epipolar geometry between two images from point matches. The intrinsic parameters of the camera are known and therefore the problem is determining the motion. The two images are taken by a fixed camera at two different time instants in a dynamic scene. We assume that the two images are projections of a single moving non deformable object.

Let a point $m_i = [u_i, v_i]^T$ in the first image be matched to a point $m'_i = [u'_i, v'_i]^T$ in the second image. They must satisfy the epipolar equation, i.e. $\tilde{m}_i^T F \tilde{m}'_i = 0$. Since a fundamental matrix F has only 7 degrees of freedom, 7 is the minimum number of point matches required for having a solution of the epipolar geometry. In practice, we are given more than 7 matches and we may try to find F by minimizing :

$$\sum_i (\tilde{m}_i^T \; F\tilde{m}'_i)^2$$

. It is linear criterion and yields an analytic solution. However, it is quite sensitive to noise, even with a large set of data points[5].

To address this problem, there is another criterion, based on Euclidean distance between point m_i and its corresponding epipolar line $l_i = F\tilde{m}'_i \equiv [l_1, l_2, l_3]^T$. Therefore the criterion can be rewritten as :

$$\min_{\mathrm{F}} \sum_i \boldsymbol{d^2(\tilde{m}_i, F\tilde{m}'_i)}, \tag{3}$$

where $d(.,.)$ is given by

$$d(m_i, l_i) = \frac{\tilde{m}_i^T F \tilde{m}'_i}{\sqrt{l_1^2 + l_2^2}}$$

The equation (3) includes only first image and it is necessary to insert the epipolar lines of the second image, too. This leads to the following criterion :

$$\min_{\mathrm{F}} \sum_i \boldsymbol{(d^2(\tilde{m}_i, F\tilde{m}'_i) + d^2(\tilde{m}'_i, F^T\tilde{m}_i))}, \tag{4}$$

which operates simultaneously in the two images.

Our problem is to determine the rotation and translation for a non-deformable moving object. The estimation steps are as follows :

- Extraction of marks in the image plane
- Evaluation of the positions of the marks
- Estimation of the angular position

There are many different ways to define a 3D rotation. One of them is to define an arbitrary 3D rotation by three consecutive rotations around the coordinate axes, that is, a rotation by α around the z-axis first, then a rotation by β around the new y-axis, and finally a rotation by $-\gamma$ around the new z-axis.

$$R = R_z(\alpha) R_y(\beta) R_z(-\gamma)$$

$(\alpha, \beta, -\gamma)$ is the same as *Euler angles.* Representing R by the three angles α, β and $-\gamma$, we have

$$\mathrm{R} = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

We know that if the images are calibrated, then the fundamental matrix, F, is reduced to the essential matrix, E. It can be seen from equation (2) that when R, t are known, E can be obtained. In other words, E may be computed as :

$$\mathrm{E} = \begin{bmatrix} 0 & -t_Z & t_Y \\ t_Z & 0 & -t_X \\ -t_Z & t_X & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{5}$$

Now problem is to find $(\alpha, \beta, \gamma, t_X, t_Y, t_Z)$ that will minimize the fitness function defined by equation (4). This minimization is considered as an optimization problem and has performed by a butterfly-based fine-grain parallel GA introduced in the next section.

4 Butterfly-based Parallel GA : Buf-PGA

Buf-PGA[6], deals with several small subpopulations with equal size each one mapped onto a different node of butterfly network. In other words, a panmictic population is divided into many subpopulations distributed among processors of butterfly. However there is no limitation in the size of subpopulation, it is desired to have a few individuals in each processor. Local subpopulations are initiated inside the node randomly. Each individual which is a solution to a given problem, is represented by a string of values called chromosome. Fitness of all the individuals in the population are evaluated in parallel by each processor locally and there is no need to any communication till this step of the algorithm. Mimicking nature, selection is being done in the neighborhood of individuals which is not fixed and can be expanded with a not-so-high probability. The smallest neighborhood includes the node itself and its first-degree neighbors, which are connected via one edge to this node: this makes four neighbors for most

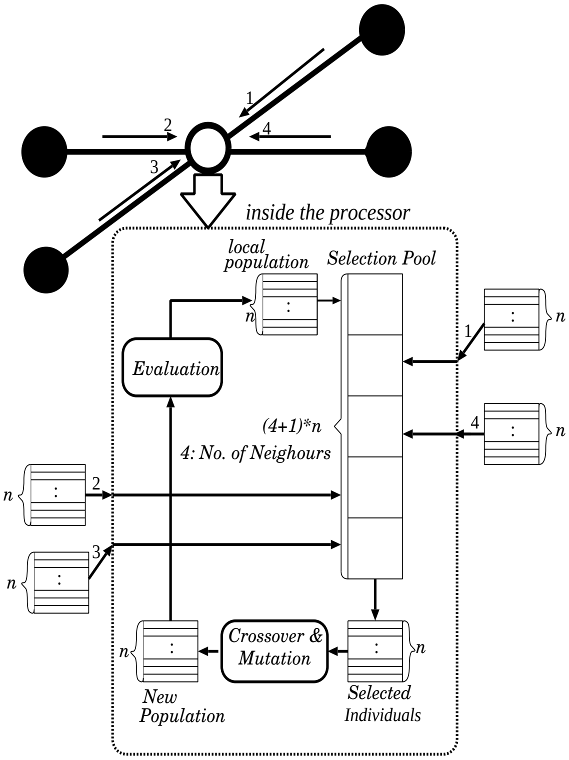

Figure 2: *Details of Evolution in Each Node of Butterfly, Range = 1*

of the processors. The exception is only on the first and the last level of the butterfly, where each node has only two first-degree neighbors. Giving different values to "*Range*" parameter causes changes in the neighborhood. For example for *Range*=1 neighborhood is restricted to the first-degree neighbors while *Range*=2 includes the second-degree neighbors, too. The neighbors of the first-degree neighbors are called as second-degree neighbors.

Figure 2 indicates details of local operation conducted by each processor to evolve its subpopulation. each stage of the algorithm, every node of the butterfly gathers all the individuals from its neighborhood with its own subpopulation into a pool called *selection pool* and then selects n individuals randomly using a roulette wheel method. n individuals are selected from a pool including $(1 + P) * n$ individuals. n denotes the size of each subpopulation and P stands for number of processors in the predetermined neighborhood. Illustrated situation in Fig. 2 includes four processors in the neighborhood of the processor, since *Range* is set to one.

In order to evaluate the algorithm, the standard online and offline were applied as performance measures in our experiments.

$$online(T) = \frac{1}{T}\sum_{1}^{T} mean(t) \tag{6}$$

Architecture	Butterfly
Dimension	2 and 3 dimensional
Generations	20 ... 100
Average over	5 runs
Range of Neighborhood	1 & 2
Sub Population Size	set {1,2,5,10}
Selection Mechanism	Roulette Wheel
Crossover Mechanism	Uniform Crossover
Crossover rate	0.1, ..., 1.0,
Mutation Rate	0.0001, ..., 0.1
No of Input Points	17
range of angles: α, β, γ	0 ... 15
range of t_X, t_Y, t_Z	10 ... 110
Performance Measurements	Online & Offline

Table 1: Specifications of our experiments

$$offline(T) = \frac{1}{T}\sum_{1}^{T} best(t) \tag{7}$$

where $mean(t)$ is the mean fitness of the subpopulation at generation t, and, $best(t)$ is the fitness of the fittest individual of the subpopulation at generation t.

5 Experimental Results

The *Butterfly*-topology fine-grain PGA, *Buf-PGA,* has been simulated and estimating of angular position of a moving object was performed using image. The equation (4) had the role of fitness function and we tried to minimize it by introducing two sets of 17 correspondent points of the first and second images to the system. Table (1) indicates the detail specifications of the implemented algorithm. As it can be seen, α, β, γ vary between 0.00 and 15.00 degrees at interval 1 degree. The results indicate that our algorithm is successful at both online and offline performance, however it shows more convergence at online.

6 Conclusions and Future Work

The method proposed in this paper allows the estimation of the three rotation angles of an object using epipolar geometry and parallel genetic algorithm. Based on the results, there is a good possibility that our proposed algorithm may outperform some well-known linear methods in the motion. Moreover, findings lend support to the assumption that the butterfly is a promising network for implementing fine-grain parallel GAs. They suggest that it may be beneficial to apply butterfly-based fine-grain parallel GA to *"Structure from Motion"* problem.

REFERENCES

[1] Cantu-Paz, E., *A Survey of Parallel Genetic Algorithms* (IlliGAL Report No. 97003). Urbana, IL: University of Illinois at Urbana-Champaign, 1997.

[2] Catherine Bounsaythip, et. all,*Genetic Algorithms in Image Processing- A Review*, in Proc. of the third Nordic Workshop on Genetic Algorithms and their Applications(3NWGA), pp. 173-192, August 1997.

[3] Taghiyareh F., Nagahashi H., *Toward Designing a New Parallel Fine-Grain Genetic Algorithm*, in the Late-Breaking Papers book of Genetic and Evolutionary Computation Conference, GECCO-99, July, Florida, USA, 1999.

[4] T.S. Huang and A.N. Netravali. *Motion and structure from feature correspondences: A review.*, Proc. IEEE, 82(2), pp. 252-268, February 1994.

[5] Xu, Gang and Zhang, Zhengyou, *Epipolar Geometry in Stereo, Motion and Object Recognition* , Kluwer Academic Publishers,1996.

[6] Taghiyareh F., Nagahashi H., *Buf-PGA : A Butterfly-topology Parallel GA as A Function Optimizer*, in the Proceeding of seventh European Congress on Intelligent Techniques and Soft Computing, EUFIT '99 , Germany, September, 1999.