

RICE UNIVERSITY


**DETERMINATION OF MOTION PARAMETERS OF A MOVING OBJECT  
FROM MOVING CAMERA DATA**

by

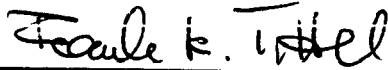
**SUNIL FOTEDAR**

A THESIS SUBMITTED  
IN PARTIAL FULFILLMENT OF THE  
REQUIREMENTS FOR THE DEGREE OF  
**MASTER OF SCIENCE**

**APPROVED, THESIS COMMITTEE :**



Dr. Rui J. P. de Figueiredo, Professor  
of Electrical and Computer Engineering,  
and of Mathematical Sciences; Director



Dr. Frank K. Tittel, Professor of Electrical  
and Computer Engineering



Dr. Paul E. Pfeiffer, Professor of  
Mathematical Sciences

Houston, Texas  
April, 1987

## ABSTRACT

The determination of 3-D motion parameters of an object from its image-sequences is discussed for three types of motion analysis : (1) *monocular vision*, (2) *stereo vision*, and (3) *stereo motion*. These parameters of the object enable one to obtain its attitude, attitude rate, visible surface shape, identification/recognition, and track. Under suitable conditions, these parameters can be estimated from the 2-D image co-ordinates of a set of points on the object's surface in consecutive images, using the *Image Point Correspondence* ( IPC ) algorithm. In this research, a *Generalized Image Point Correspondence* ( GIPC ) algorithm has been developed to enable the computation of motion parameters for a general configuration where both the object and the camera are moving. A detailed error analysis of these algorithms has been carried out. Furthermore, this algorithm was tested on both simulated and video-acquired data, and its accuracy determined.

## TABLE OF CONTENTS

Chapter I : Introduction .....	1
1.1. Background .....	1
1.1.1. Applications of Motion Analysis .....	1
1.1.2. Current Research Thrusts in Motion Analysis .....	2
1.1.3. Approaches to the Solution to the Motion Analysis Problem .....	5
1.1.3.1. Intensity-Based Approach .....	5
1.1.3.2. Feature-Based Approach .....	6
1.2. Description of Research in the Present Work .....	7
1.2.1. Statement of the Research Problem .....	7
1.2.2. Assumptions .....	9
1.2.3. Outline of the Present Work .....	9
Chapter II : Frame Transformations and Rigid-Body Motion .....	11
2.1. Relationship between Co-ordinate Frame Transformation and Rigid-Body Motion .....	11
2.2. Three Cases of Motion Analysis And Their Equivalence .....	14
2.2.1. The Two-View Motion Analysis/Monocular Vision Case .....	14
2.2.2. The Stereo Vision/Binocular Vision Case .....	19
2.2.3. The Stereo Motion Case .....	22

<b>Chapter III : The Image Point Correspondence Algorithm .....</b>	<b>23</b>
<b>3.1. The Algorithm .....</b>	<b>23</b>
<b>3.1.1. The First Method .....</b>	<b>26</b>
<b>3.1.2. The Second Method .....</b>	<b>27</b>
<b>3.1.3. The Third Method .....</b>	<b>30</b>
<b>Chapter IV : The Generalized Image Point Correspondence Algorithm .....</b>	<b>33</b>
<b>4.1. The Algorithm .....</b>	<b>33</b>
<b>4.1.1. The First Case .....</b>	<b>38</b>
<b>4.1.2. The Second Case .....</b>	<b>40</b>
<b>4.2. The Determination of the Motion Parameters .....</b>	<b>41</b>
<b>4.2.1. The Object Surface Shape .....</b>	<b>41</b>
<b>4.2.2. Range/Interframe Range Rate .....</b>	<b>43</b>
<b>4.2.3. Interframe Attitude/Attitude Rates .....</b>	<b>43</b>
<b>4.2.4. Object Tracking .....</b>	<b>44</b>
<b>Chapter V : Error Analysis and Experimental Results .....</b>	<b>46</b>
<b>5.1. Preliminary Remarks .....</b>	<b>46</b>
<b>5.1.1. Sources Of Error .....</b>	<b>46</b>
<b>5.1.2. Definitions .....</b>	<b>47</b>
<b>5.1.2.1. Error, Relative Error .....</b>	<b>47</b>
<b>5.1.2.2. Error Bounds .....</b>	<b>47</b>
<b>5.1.2.3. Errors In Simultaneous Equations .....</b>	<b>48</b>
<b>5.2. Error Propagation In Present Algorithms .....</b>	<b>49</b>

5.2.1. Error In Matrix Q Due To Error In Feature Coordinates .....	49
5.2.2. Error In Rotation Matrix R Due To Error In Matrix Q .....	52
5.2.2.1. Error Analysis Using The First Method .....	52
5.2.2.2. Error Analysis Using The Second Method .....	54
5.2.2.3. Error Analysis Using The Third Method .....	56
5.2.3. Error In Motion Parameters Due To Error In Rotation Matrix, R .....	58
5.2.3.1. Using the first representation of R .....	58
5.2.3.2. Using the second representation of R .....	61
Chapter VI : Experimental Results .....	63
6.1. Using Simulated Data .....	63
6.2. Using Real Data .....	64
6.3. Experimental Error Estimates .....	66
Chapter VII : Conclusions .....	85
Appendix A .....	87
Appendix B .....	94
Appendix C .....	97
Appendix D .....	101
Appendix E .....	102
Appendix F .....	103
Appendix G .....	105
References .....	110

## **CHAPTER I**

### **INTRODUCTION**

The research in motion analysis has evolved over the years as a challenging field in the area of computer vision. Its major contribution is in application to dynamic image-sequence analysis. The information contained in the image-sequences of a moving object aids in the computation of the motion parameters, as well as, the segmentation and shape analysis.

#### **1.1. BACKGROUND**

In this section we provide the motivation for this research and present developments pertinent to it.

##### **1.1.1. Applications of Motion Analysis**

The potential applications of image-sequence analysis, as indicated by Shariat ([1]), are listed below.

(1) *Robotics/Automation* : A camera mounted on the end-effector of a manipulator ( robot arm ) takes images of a moving object or target. The range and the orientation of the object relative to the end-effector are estimated from a knowledge of the displacements of the images of its features. This information, in turn, is used as a feedback to control the manipulator. Robotics and automation find applications in industry and space. For space applications, a robot's task can be, for example, to retrieve a defective satellite. After knowing the position, orientation, and the velocities in all six degrees of freedom of the satellite, the motion of the manipulator under computer control is matched to that of the satellite. It is done so that when the robot grasps the satellite, no excessive forces and torques are produced, which might otherwise damage the satellite or the manipulator. A future use of the robot is in its application to an autonomous vehicle. Such robot-controlled vehicle could be extensively used for space exploration,

using vision for its navigation. One of the functions of the vision system is to prevent collision with obstacles, such as rocks, while moving on and taking pictures of different types of terrains in planetary exploration.

(2) *Medicine and Biological Sciences* : The image-sequences are used to understand the normal functions of an organ as well as its abnormalities. They are also used in the fields of biophysics, biology, and biomedicine to study movement and behavior of fish and micro-organisms.

(3) *Military Applications* : The camera is used to identify and track moving targets in its field of view. From a set of images of the target, its range and orientation can be computed. This process of recognition of the target is essential for some military applications.

(4) *Meteorological Applications* : Cloud displacements are measured from a sequence of satellite-acquired images. From these, the direction of cloud motion and wind velocity are estimated for weather forecasting.

(5) *Traffic Monitoring* : The detection, tracking, and identification of moving vehicles, such as cars, are done from a sequence of images.

(6) *Segmentation and Scene Analysis* : The motion analysis provides useful information regarding the segmentation and the scene analysis in an environment where multiple objects are moving.

### **1.1.2. Current Research Thrusts In Motion Analysis**

The current research in motion analysis for perception is concentrated in the following areas, as indicated in Fig.1 :

(i) Feature extraction and matching of consecutive images.

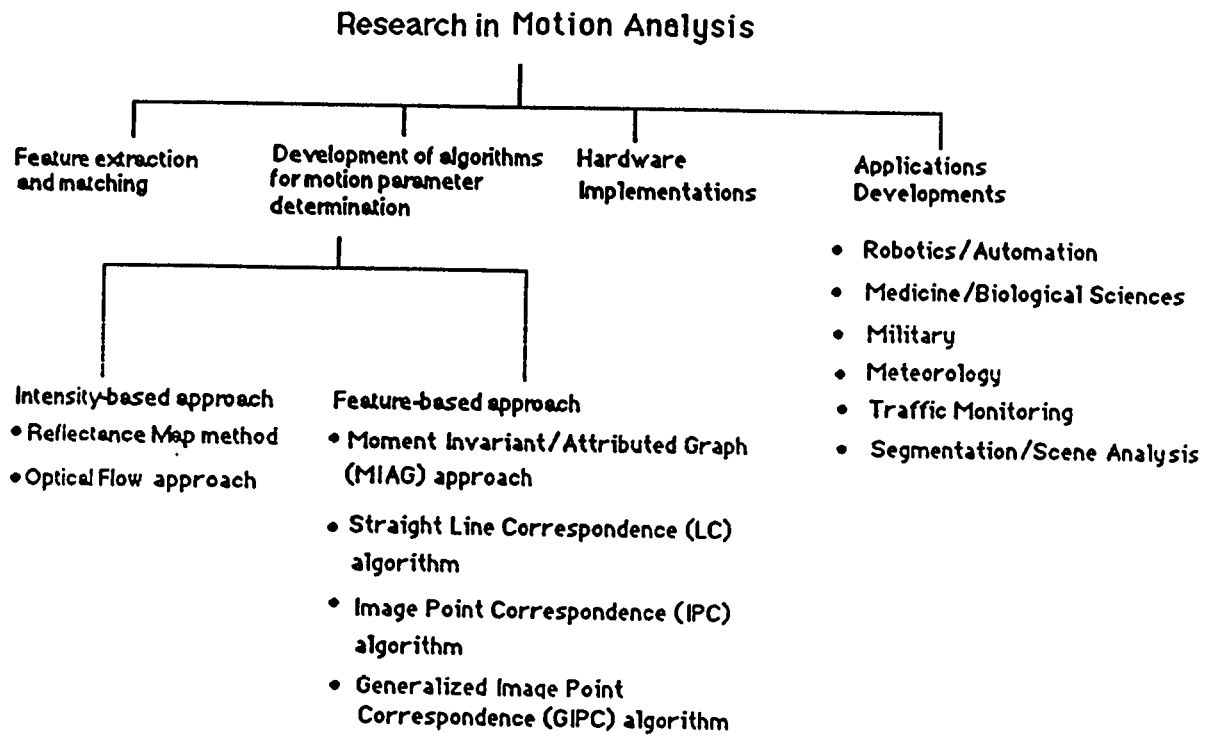


Fig1 : Current Research Thrusts in Motion Analysis



(ii) Efficient algorithms for motion parameter detection.

(iii) Hardware Implementations.

(iv) Applications developments.

Brief comments on the above four thrusts follow.

Techniques are being developed that enable extraction of features and matching these over consecutive frames. Features can also be estimated from images for identification purposes. Furthermore, several images can be matched using a set of features corresponding to a particular object or scene. Another approach to the identification of the object/target is the correlation of the acquired scenes with those stored in the computer. Feature extraction and image matching provide the needed inputs to the algorithms for motion parameter estimation. These algorithms perform unique operations on image sequences. Considerable effort is presently being expended in the area of parameter estimation for applications with relative motion between the camera and the object/scene. Applications of motion parameter estimation in various fields of engineering and science continue to be explored at an accelerated pace. The thrust of automation and robotics to industrial, space, and defense operations has created a need for the motion parameter estimation algorithms. The research efforts in the areas of algorithm development and applications determine requirements for hardware development. Hardware subsystems comprise of cameras, videos, pointing, tracking, and on-board data processing. The technology innovations for space applications include reduction of size and weight, increase in speed and reliability, and automatic operation ([2]). For the medical and ground-based systems, technology is being developed for data processing/recording, and display subsystems.

In the past two decades, a number of approaches have evolved for the development of efficient algorithms for motion perception from a sequence of images. Some use

two frames of the images of the moving object ( [3,4,5,6,7] ), while others use more than two ( [1,8] ), to determine the motion parameters of the second image of the object relative to the first image. These approaches are discussed in the following subsection.

### **1.1.3. Approaches to the Solution for the Motion Analysis Problem**

In order to develop efficient algorithms, one can use either of the two broadly classified approaches :

**1.1.3.1. Intensity-Based Approach :** Here, image processing techniques like subtraction, correlation, convolution, Fourier analysis, or differentiation, are applied to the image of the object to estimate its motion parameters.

The algorithms that fall in this category are explained below :

#### *(1) Reflectance Map Method :*

The dependence of surface reflection on the geometry of incident and reflected rays is given by the bidirectional reflectance distribution function ([9]). The reflectance map, that gives a relationship between the surface orientation and brightness, can be derived from this function and the distribution of the light sources. The photometric stereo method for recovering the orientation of surface patches from a number of images taken under different lighting conditions has been developed. If a single image is available, the shape can also be recovered from the spatial variation of brightness called *shading*, since parts of the surface are oriented differently and thus appear with different brightnesses.

#### *(2) Optical Flow Approach :*

Optical flow is a velocity field that defines motion in an image. A velocity vector is assigned to each point in the image. Brightness patterns in the image move as the object moves. It is the apparent motion of these brightness patterns that gives the optical flow. Movement through the environment maps information onto a pattern. From

this pattern through *inverse mapping*, it is possible to derive information about the environment and the observer's motion ([9,10,11,12,13]). This approach requires iterative searches. The orthographic (parallel) projection is also assumed.

**1.1.3.2. Feature-Based Approach :** In this approach, prominent features are found and then matched over consecutive time-varying frames. The features can be points, line segments, edge fragments, or moment invariants. The correspondence or the matching problem over a set of frames is assumed to be known *a priori*.

The algorithms that fall in this category are explained below :

(1) *Moment Invariant/Attributed Graph Approach* : The 3-D objects are recognized, and the motion parameters determined from their 2-D orthographic projections. The geometric transforms are used instead of the iterative matching techniques ([14,15,16,17,18]). For the identification purposes, a 3-D object is represented by an attributed graph where a node represents a face of the object. Associated with each node is a feature vector containing moment invariants of the face. A link between two nodes means that the two faces are adjoining. Associated with each edge is a scalar which is the angle between two nodes. Any 2-D projection of the object can be similarly represented by a graph, which is a subgraph of the above graph. It is so because only a part of the object is facing the camera. Thus the identification problem becomes a subgraph isomorphism between the observed image and the 3-D object. The moment invariants are found to be invariant under 3-D motion. The attitude parameters of the observed object relative to the 3-D object are determined from the knowledge of 2-D moments of its faces. The orthographic projection is assumed in this analysis. It also considers the object with flat surfaces.

(2) *Straight Line Correspondence Algorithm* : The 3-D motion/structure of a rigid body, containing straight line segments as features, can be determined if a sequence of

three 2-D perspective views is given ([19,20]). The projections of 3D lines over the three consecutive image frames are assumed to be known. A seven line correspondence (LC) involves an iterative search without any constraints on the 3-D line. If the 3-D lines lie on parallel planes, and the orientation of the rotation axis is fixed over the three image frames, an eight LC results in a linear method. The surface of a unit sphere is used in place of the plane of the perspective projection. In this analysis, the projections of moving 3-D lines on this sphere over three frames are studied. A fairly good initial guess is required for the convergence in the iterative search.

### (3) *Image Point Correspondence Algorithm* :

Three different cases of motion analysis have been identified ([4,5,6,7]). They are : (i) Two-view motion analysis ( monocular vision ), (ii) stereo vision, and (iii) stereo motion. A discussion of these cases appears in the following section where the approach used in the present research is described.

## 1.2. DESCRIPTION OF RESEARCH IN THE PRESENT WORK

### 1.2.1. Statement of the Research Problem

In this work, three cases of motion analysis based on vision, namely, monocular vision, stereo vision, and stereo motion, have been investigated, and, in this context, the work done by different authors ([4,5,6,7]) in developing the *Image Point Correspondence* ( IPC ) algorithm in three different ways has been studied. However, the IPC algorithm does not apply to a more general problem of motion analysis. The generalized version of motion analysis involves a situation where both the object and the camera are moving. Industrial and space robots face this situation in locating and tracking of various objects/scenes. The space robot, for example, takes pictures of a satellite for motion analysis. Both the camera/video system and the object move asynchronously. An algorithm for motion parameter estimation is required for this general case of relative motion.

An extension of the IPC algorithm is, therefore, needed to address this requirement.

As mentioned previously, three different types of motion analysis have been identified. They are discussed in the following :

(i) *Two-view motion analysis or monocular vision case* : The pictures of a moving target are taken by a stationary camera at different instants of time. The motion parameters of the target are found from the knowledge of the correspondence of projected images of the 3-D points on it, using the IPC algorithm. The surface of the target is also determined.

(ii) *Stereo vision or binocular vision case* : The pictures of a stationary target are taken by two cameras stationed at different locations in this case. The motion parameters that relate the two camera co-ordinate systems are found by using the IPC algorithm. The surface of the target can also be determined from a minimum of eight data points available.

(iii) *Stereo motion case* : This case is similar to the second case. However, instead of two stationary cameras, one moving camera is used to take the pictures of the stationary target from two different locations at different instants of time.

An extension to the IPC algorithm, termed as *Generalized Image Point Correspondence* ( GIPC ) algorithm, has been used for the general problem of motion analysis in this research. In the generalized version of the motion analysis, both the object/scene and the camera are moving. The other three cases of motion analysis have been determined to be its special cases.

In all the cases mentioned, the image plane is assumed to be at the focal point of the camera with its X- and Y-axes parallel to those of the camera co-ordinate system. The center of the camera rotations coincides with the origin of the camera co-ordinate system, and its z-axis is the line of sight.

### 1.2.2. Assumptions

The assumptions used in the development and testing of the algorithm are as follows :

(a) The object undergoes a *rigid-body* motion which, in other words, means that the object does not change its shape while in motion.

(b) The motion of the object is constant in any two consecutive frames. In other words, it means that there are no discontinuities in the motion parameters, and the axis of rotation remains fixed.

(c) For using the GIPC algorithm, the direction of motion of the object is known.

(d) For the GIPC algorithm, the motion parameters of the camera are known so as to compute those of the object.

(e) There are certain restrictions on the spatial distribution of the data points on the surface of the object in order to run the IPC algorithm, and are explained in Appendix B.

### 1.2.3. Outline of the Present Work

In chapter II, the relationship between the co-ordinate frame transformation and the rigid-body motion of an object is determined. The three cases of motion analysis are shown to be equivalent for the determination of motion parameters.

In chapter III, the three methods for the IPC algorithms, adopted by different authors, are presented as different manifestations of the same approach.

In chapter IV, the GIPC algorithm is developed. Furthermore, the IPC algorithm applied to the three cases of motion analysis is shown to be a special case of the GIPC algorithm applied to the most general motion equation.

In chapter V, error analysis for the three methods of the IPC algorithm has been carried out.

Chapter VI discusses some experimental results, where simulated and real data have been used.

In Appendices A through F, proofs and definitions used in this research, are presented. Appendix G presents implementations of the IPC and the GIPC algorithms in terms of computer programs.

## CHAPTER II

### FRAME TRANSFORMATIONS AND RIGID-BODY MOTION

In this chapter, the relationship between co-ordinate frame transformations and *rigid-body* motion of an object ([6]) is established. Rigid-body motion is represented as a *rotation*, followed by a *translation*. The rotation is expressed in terms of a matrix  $\mathbf{R}$ . Two different representations of the rotation matrix are widely used. The rotation matrix is an orthonormal matrix of the first kind. Other properties, along with the two representations of the matrix  $\mathbf{R}$  just mentioned are discussed in Appendix A. The translation is expressed as a vector  $\mathbf{T}$  with its components representing translations along the three axes.

In what follows, we shall show the three special cases of motion analysis; namely, the *monocular vision*, *stereoscopic vision*, and *stereo motion*, are equivalent. A detailed description of these is also given.

#### 2.1. RELATIONSHIP BETWEEN CO-ORDINATE FRAME TRANSFORMATION AND RIGID-BODY MOTION

In this section, the various developments have been taken from Zhuang *et al.* ([6]). Let there be two right-hand co-ordinate frames in the 3-D Euclidean space  $\mathbf{E}_3$  ( Fig.2 ) :

$$\mathbf{F} = [ \mathbf{o} ; \Phi ] \quad \text{and} \quad \mathbf{F}' = [ \mathbf{o}' ; \Phi' ] ;$$

where  $\mathbf{o}$  and  $\mathbf{o}'$  are the origins, and  $\Phi$  and  $\Phi'$  are the matrices containing the orthonormal basis  $( \phi_1 , \phi_2 , \phi_3 )$  and  $( \phi_1' , \phi_2' , \phi_3' )$  of the frames  $\mathbf{F}$  and  $\mathbf{F}'$  respectively. These basis vectors define the relative orientations of the frames with respect to a *base* or a *standard* frame  $\mathbf{S}$ . Thus the transformation of the frame  $\mathbf{F}'$  to the frame  $\mathbf{F}$  can be expressed in terms of a rotation matrix  $\mathbf{R}$  followed by a translation  $\mathbf{T}$  from the origin  $\mathbf{o}'$  to the origin  $\mathbf{o}$ . In other words, the set  $\Phi$  of the basis vectors of  $\mathbf{F}$  can be expressed in terms of the set  $\Phi'$  of  $\mathbf{F}'$  by the equation



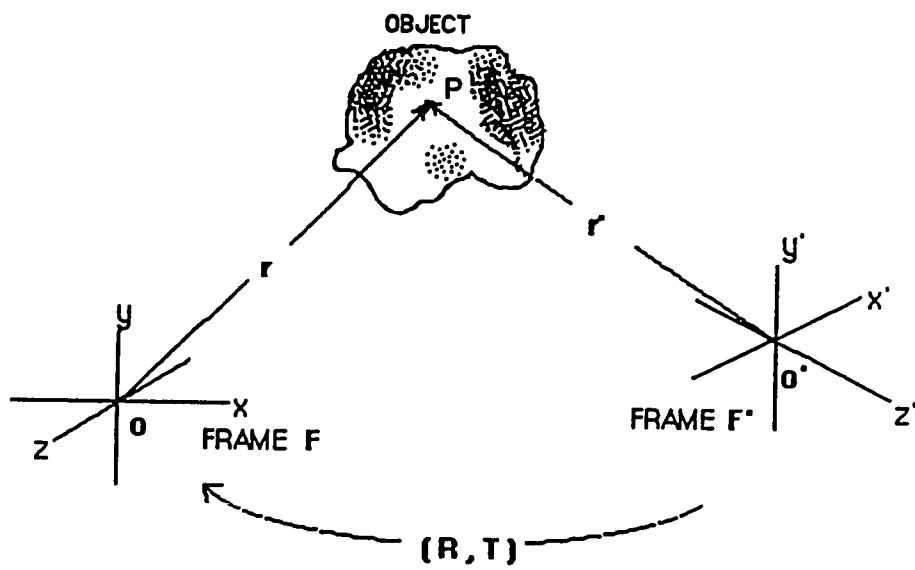


Fig2. Geometry illustrating the relationship between Frame Transformation and Rigid Body Motion.

$$\Phi = \Phi' R \quad (2.1a)$$

and

$$\mathbf{o} = \mathbf{o}' + \mathbf{T} . \quad (2.1b)$$

Any point P (Fig.2) in  $\mathbf{E}_3$  can be represented in both the frames  $\mathbf{F}$  and  $\mathbf{F}'$  respectively as follows

$$P = x \phi_1 + y \phi_2 + z \phi_3 + \mathbf{o} = \Phi \mathbf{r} + \mathbf{o} \quad (2.2a)$$

$$= x' \phi_1' + y' \phi_2' + z' \phi_3' + \mathbf{o}' = \Phi' \mathbf{r}' + \mathbf{o}' ; \quad (2.2b)$$

where  $\mathbf{r} = (x, y, z)^t$ ,  $\mathbf{r}' = (x', y', z')^t$  are the co-ordinates of P with respect to the frames  $\mathbf{F}$  and  $\mathbf{F}'$  respectively, and the superscript 't' indicates the transposition operation.

Substituting eqn.(2.1a) and eqn.(2.1b) in eqn.(2.2), we get

$$\Phi' \mathbf{r}' = \Phi' R \mathbf{r} + \mathbf{T} . \quad (2.3)$$

If the frame  $\mathbf{F}'$  coincides with the frame  $\mathbf{S}$  in  $\mathbf{E}_3$ , then,

$$\mathbf{o}' = (0, 0, 0)^t, \quad \Phi' = \mathbf{I}, \quad \text{or} \quad \phi_1' = (1, 0, 0)^t, \quad \phi_2' = (0, 1, 0)^t \quad \text{and} \quad \phi_3' = (0, 0, 1)^t,$$

and eqn.(2.3) can be simplified as

$$\mathbf{r}' = \mathbf{R} \mathbf{r} + \mathbf{T} . \quad (2.4)$$

This represents the most basic form of the *motion analysis equation*. From eqn.(2.4), we deduce that when the standard frame  $\mathbf{F}'$  in  $\mathbf{E}_3$  transforms into the right-hand frame  $\mathbf{F}$  by the rotation  $\mathbf{R}$  and the translation  $\mathbf{T}$ , the point  $\mathbf{r}$  experiences a rigid-body motion defined by  $(\mathbf{R}, \mathbf{T})$  and becomes the point  $\mathbf{r}'$  where  $\mathbf{r}$  and  $\mathbf{r}'$  are the co-ordinates of the same spatial point P in  $\mathbf{E}_3$  relative to the frames  $\mathbf{F}$  and  $\mathbf{F}'$  respectively.

Conversely, if we consider a point  $P(r)$  undergoing a rigid-body motion to another point  $P'(r')$  in Fig.3, then

$$r' = R r + T . \quad (2.5)$$

Let a new frame  $F'$  be defined as follows :

$$\Phi' = \Phi R^t \quad (2.6a)$$

and

$$o' = o - T \quad (2.6b)$$

and consider  $F'$  as the standard frame. In that case,  $x\phi_1 + y\phi_2 + z\phi_3 + o$  and  $x'\phi_1' + y'\phi_2' + z'\phi_3' + o'$  represent the same point in  $E_3$  and hence  $r$  and  $r'$  become co-ordinate representations of that point with respect to  $F$  and  $F'$  respectively. In other words, the motion of the point  $P(r)$  to the point  $P'(r')$  with a fixed frame  $F'$  is similar to the motion of the frame  $F'$  to the frame  $F$  with respect to a fixed point  $P$ .

## 2.2. THE THREE CASES OF THE MOTION ANALYSIS AND THEIR EQUIVALENCE

We are now in a position to describe the three special classes of motion analysis, find their equivalence ([6]), and discuss the orientation of the camera with a frame. The reader is referred to Tsai ([21]) for the relationships shown in the following sub-sections. The three cases are :

### 2.2.1. The Two-View Motion Analysis/Monocular Vision Case

One stationary camera is used to take images of a moving object at different instants of time ( Fig.3 ). The camera's lens coincides with the origin  $o$  of the frame  $F$ , its line of sight directs along  $z < 0$ , and its image plane is at  $z = f$ , where  $f$  ( assumed known and normalized to 1 ) is the focal length.

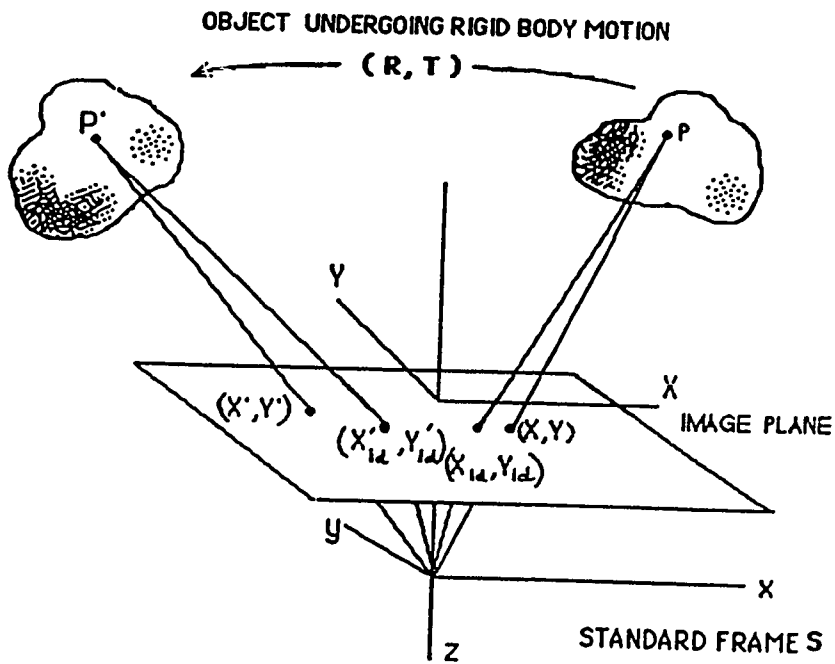


Fig3: Camera Geometry illustrating Perspective Projection and Radial Lens Distortion in Two-View Motion Analysis Case.

The algorithm can be explained by considering any point  $P$  at any instant  $t_1$  on the surface of an object. The point  $P$  moves to another point  $P'$  at instant  $t_2$  ( $t_2 > t_1$ ) as a result of the motion of the object. The geometry of the problem is shown in Fig.3 where  $S$  is a standard or reference co-ordinate system in the Euclidean space  $E_3$  with which the camera co-ordinate system coincides. Let

$\mathbf{r} = (x, y, z)^t$  be the object-space co-ordinates of  $P$  relative to the frame  $S$ ;

$\mathbf{r}' = (x', y', z')^t$  the object-space co-ordinates of  $P'$  relative to the frame  $S$ .

If the object moves with motion parameters  $(\mathbf{R}, \mathbf{T})$ , where  $\mathbf{R}$  and  $\mathbf{T}$  are the rotation matrix and translation vector respectively and given by

$$\mathbf{R} = \begin{bmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{bmatrix} = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix} \quad (2.7a)$$

and

$$\mathbf{T} = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \quad (2.7b)$$

$\mathbf{p}_\alpha$  ( $\alpha = 1, 2, 3$ ) being the  $\alpha$ 'th row of  $\mathbf{R}$ ;  $r_1, r_2, \dots, r_9$  the rotation parameters and  $t_x, t_y$  and  $t_z$  the translations along  $x$ -,  $y$ - and  $z$ -axes respectively; we have

$$\mathbf{r}' = \mathbf{R} \mathbf{r} + \mathbf{T} . \quad (2.8)$$

Let ( [21] )

$(X, Y)$  : the ideal image-space co-ordinates of  $P$  if a perfect pin-hole camera is used;

$(X_{ld}, Y_{ld})$  : the actual image-space co-ordinates of  $P$  which differ from the ideal co-ordinates due to the lens distortion, and

$(X_c, Y_c)$  : the co-ordinates of  $P$  used in the computer which are the number of pixels for the discrete image in the frame memory.

Likewise, let

$(X', Y')$  : the ideal image-space co-ordinates of  $P'$  ;

$(X_{ld}', Y_{ld}')$  : the actual image-space co-ordinates of  $P'$  , and

$(X_c', Y_c')$  : the co-ordinates of  $P'$  used in the computer.

The following steps, shown by Tsai ([21]), give the transformations which arise due to the rigid-body motion of the object:

**Step I.** Rigid-body motion from  $P$  to  $P'$  is given by eqn.(2.8).

**Step II.** Transformations from 3-D object-space co-ordinates of  $P$  and  $P'$  to the ideal 2-D image-space co-ordinates, using perspective projections, are respectively

$$X = f x/z ; \quad Y = f y/z ; \quad (2.9a)$$

$$X' = f x'/z' ; \quad Y' = f y'/z' ; \quad (2.9b)$$

where 'f' is the focal length of the camera lens used.

**Step III.** Transformations due to the lens distortions are given by two kinds of lens distortions : radial and tangential, the former one being more common. Ignoring the tangential component of the lens distortion, we get ([21])

$$X_{ld} = \sigma X ; \quad Y_{ld} = \sigma Y ; \quad (2.10a)$$

$$X_{ld}' = \sigma' X' ; \quad Y_{ld}' = \sigma' Y' ; \quad (2.10b)$$

where

$$\sigma = (1 + \alpha \xi^2 + \beta \xi^4)^{-1} ; \quad \sigma' = (1 + \alpha \xi'^2 + \beta \xi'^4)^{-1} ; \quad (2.11a)$$

$$\xi^2 = X_{ld}^2 + Y_{ld}^2 \quad \text{and} \quad \xi'^2 = X_{ld}'^2 + Y_{ld}'^2 ; \quad (2.11b)$$

$\alpha$  and  $\beta$  being the distortion coefficients.

**Step IV.** This step of transformation results from using TV cameras, particularly solid state CCD, as a visual aid for robots. The transformations from actual image co-ordinates to the computer image co-ordinates are given by

$$X_c = \gamma X_{ld} + C_x ; Y_c = \delta Y_{ld} + C_y ; \quad (2.12a)$$

$$X_c' = \gamma X_{ld}' + C_x \quad \text{and} \quad Y_c' = \delta Y_{ld}' + C_y ; \quad (2.12b)$$

where

$$\gamma = (s_x / l_x) (\kappa_{fx} / \kappa_{cx}) ; \quad (2.12c)$$

$$\delta = 1/l_y ; \quad (2.12d)$$

$l_x$  : center to center distance between adjacent sensor elements along X axis;

$l_y$  : center to center distance between adjacent sensor elements along Y axis;

$s_x$  : uncertainty image scale factor due to slight hardware timing mismatch between image acquisition and camera scanning hardware or the timing of TV scanning not being precise;

$C_x, C_y$  : image origin;

$\kappa_{fx}$  : number of pixels in a line as sampled by the computer and

$\kappa_{cx}$  : number of sensor elements along X axis.

The parameters  $f, \alpha, \beta, \gamma, \delta, \sigma, \sigma', C_x, C_y$  are called the *intrinsic parameters* of the camera which can be found out by the *camera calibration techniques* ([21,22,23]).

Combining all the steps shown above, we get the following relationship between the actual computer co-ordinates of P and P' :

$$X_c' = f \left[ \frac{(r1 X_c + r2 Y_c + r3) z + t_x}{(r7 X_c + r8 Y_c + r9) z + t_z} \right]; \quad (2.13a)$$

$$Y_c' = f \left[ \frac{(r4 X_c + r5 Y_c + r6) z + t_y}{(r7 X_c + r8 Y_c + r9) z + t_z} \right]; \quad (2.13b)$$

For a vidicon-type camera where  $\kappa_{fx} = \kappa_{cx}$ , and  $C_x = 0 = C_y$  with no lens distortion (i.e.,  $\alpha = 0 = \beta$ ), and  $l_x = l_y = s_x = 1$ , the transformation equation (2.13) reduces to the form ([5])

$$X' = f \left[ \frac{(r1 x + r2 y + r3 z + t_x)}{(r7 x + r8 y + r9 z + t_z)} \right] = f \left[ \frac{(r1 X + r2 Y + r3) z + t_x}{(r7 X + r8 Y + r9) z + t_z} \right] \quad (2.14a)$$

and

$$Y' = f \left[ \frac{(r4 x + r5 y + r6 z + t_y)}{(r7 x + r8 y + r9 z + t_z)} \right] = f \left[ \frac{(r4 X + r5 Y + r6) z + t_y}{(r7 X + r8 Y + r9) z + t_z} \right] \quad (2.14b)$$

### 2.2.2. The Stereo Vision/Binocular Vision Case

Two stationary cameras are used to take pictures of a stationary object (Fig.2). The lens of one camera coincides with the origin  $\mathbf{o}$  of the frame  $\mathbf{F}$  and the line of sight is directed along  $z < 0$ . Another camera's lens coincides with the origin  $\mathbf{o}'$  of the frame  $\mathbf{F}'$  and the line of sight directs along  $z' < 0$ . The  $z = f$  and  $z' = f'$  (where  $f$  and  $f'$  are assumed to be 1 without any loss of generality) are two image planes in  $\mathbf{F}$  and  $\mathbf{F}'$  respectively.

One of the important applications of stereoscopic vision is in photogrammetry where the shape of an object is determined from overlapping its pictures or 2D images which are taken by properly calibrated cameras stationed at two places. The transformation between the two camera coordinate frames based on a knowledge of 2D image coordinates of the points on the surface of the object is called the *relative orientation*. If the 3D coordinates of the points are known, it is called *absolute orientation*.

A simple example for stereo vision ([9]) is shown in Fig.4 where the two cameras are placed such that their optical axes are parallel and separated by a distance 'd'. The



camera coordinate frames align with  $F$  and  $F'$ . The line which connects their centers is called the *baseline*. Let the image coordinates of any point  $P(x, y, z)$  on the surface of an object relative to the two frames be  $(X_1, Y_1)$  and  $(X_2, Y_2)$  respectively. It is seen that

$$X_1 / f_1 = \frac{x + d/2}{z} ; X_2 / f_2 = \frac{x - d/2}{z} \text{ and } Y_1 / f_1 = Y_2 / f_2 = y / z ; \quad (2.15)$$

where ' $f_1$ ' and ' $f_2$ ' are the focal lengths of the two camera lenses. Assuming the same focal length ' $f$ ' for the two cameras, the eqn.(2.15) could be solved for  $x, y, z$ , which in turn, determine the shape of the surface of the object. Thus

$$x = \frac{d (X_1 + X_2)}{2 \mu} ; y = \frac{d (Y_1 + Y_2)}{2 \mu} \text{ and } z = d f / \mu ; \quad (2.16)$$

where

$$\mu = X_1 - X_2$$

and is called the *disparity*. Distance is inversely proportional to disparity. The distance to near objects can thus be measured accurately. This is not true for far objects. This arrangement of determining the surface shape encounters other problems and are discussed in [9].

In a practical situation, the two cameras are seldom parallel to each other and are oriented relative to each other as shown in Fig.2. The transformation between the two camera frames  $F$  and  $F'$  can always be regarded as a rigid body motion and thus can be decomposed into a rotation and a translation. If  $r' = (x', y', z')^t$  is the 3D object space coordinates of  $P$  relative to frame  $F'$  and  $r = (x, y, z)^t$  the 3D object space coordinates of  $P$  relative to the frame  $F$ , then eq.(2.8) holds true. The steps that give the transformations which arise due to the rigid-body motion of the camera are similar to the ones discussed in the monocular vision case earlier. Fig.4 is a special case of Fig.2, where

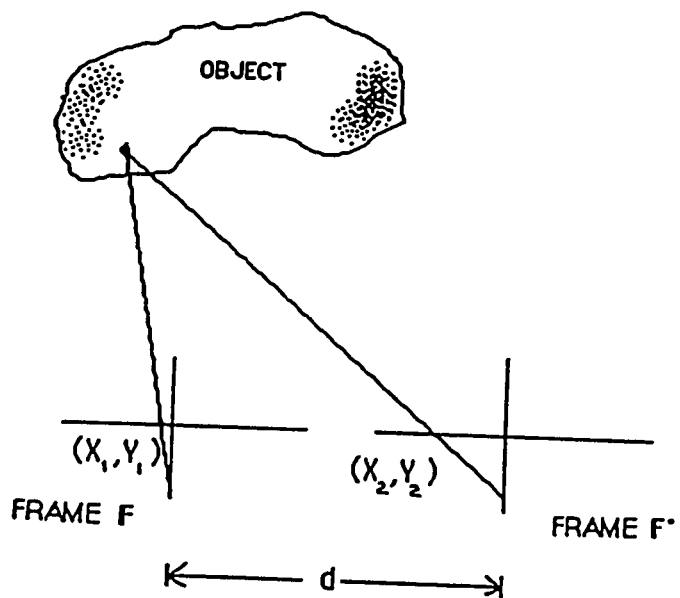


Fig4: Special Case of Stereo Vision where the optical axes of two cameras are parallel.

$$\mathbf{R} = \mathbf{I} \quad \text{and} \quad \mathbf{T} = \begin{bmatrix} -d \\ 0 \\ 0 \end{bmatrix}.$$

The identity matrix  $\mathbf{I}$  indicates that there is no rotation around any axis.

### **2.2.3. The Stereo Motion Case**

This case is similar to the second case in the sense that instead of using two stationary cameras, a moving camera is used. This camera takes pictures of a stationary object from different locations. One location is identified with the frame  $\mathbf{F}$  and the other with the frame  $\mathbf{F}'$ . Thus, as before, the motion analysis equation is represented by eqn.(2.8).

## CHAPTER III

### THE IMAGE POINT CORRESPONDENCE ALGORITHM

The motion parameters of an object from its image-sequences are determined by using the Image Point Correspondence ( IPC ) algorithm. The input to the algorithm is a set of 2-D image co-ordinates of data points in the consecutive images with known correspondence.

Several authors ([4,5,6,7]) have developed special methods for the three classes of motion analysis. In this chapter, we briefly survey these methods separately before the concept of using an extended algorithm to a more general problem of motion analysis is introduced in the next chapter. The detailed analyses and various proofs are presented in Appendices B through F.

#### 3.1. THE ALGORITHM

Solving eqns.(2.14a) and (2.14b), the following expressions for 'z' are obtained respectively,

$$z = \frac{t_x - t_z X' / f}{X' / f (r_7 X / f + r_8 Y / f + r_9) - (r_1 X / f + r_2 Y / f + r_3)} \quad (3.1a)$$

and

$$z = \frac{t_y - t_z Y' / f}{Y' / f (r_7 X / f + r_8 Y / f + r_9) - (r_4 X / f + r_5 Y / f + r_6)} \quad (3.1b)$$

Normalizing 'f' to 1, we get from eqns.(3.1a) and (3.1b)

$$r'{}^t Q r = 0; \quad (3.2a)$$

or

$$V'{}^t Q V = 0; \quad (3.2b)$$

or

$$\mathbf{M} \mathbf{Q}_v = 0; \quad (3.2c)$$

where

$$\mathbf{V} = [X, Y, 1]^t; \quad \mathbf{V}' = [X', Y', 1]^t; \quad (3.2d)$$

$$\mathbf{Q} = \begin{bmatrix} l_1 \\ l_2 \\ l_3 \end{bmatrix} = \begin{bmatrix} q_1 & q_2 & q_3 \\ q_4 & q_5 & q_6 \\ q_7 & q_8 & q_9 \end{bmatrix} = \pm \begin{bmatrix} t_z r_4 - t_y r_7 & t_z r_5 - t_y r_8 & t_z r_6 - t_y r_9 \\ t_x r_7 - t_z r_1 & t_x r_8 - t_z r_2 & t_x r_9 - t_z r_3 \\ t_y r_1 - t_x r_4 & t_y r_2 - t_x r_5 & t_y r_3 - t_x r_6 \end{bmatrix}; \quad (3.2e)$$

$$\mathbf{M} = [XX', YX', X', XY', YY', Y', X, Y, 1] \quad (3.2f)$$

and

$$\mathbf{Q}_v = (q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8, q_9)^t. \quad (3.2g)$$

In eqn.(3.2e),  $l_\alpha$  ( $\alpha = 1,2,3$ ) is the  $\alpha$ 'th row of  $\mathbf{Q}$ .

Eqn.(3.2c) has nine unknowns, as given in eqn.(3.2g), and are called *essential parameters*. These play a vital role in the computation of motion parameters. However, applying these equations to  $n$  ( $n \geq 9$ ) known locations of data points on the object surface and their corresponding image points, called the *conjugate or image point correspondence* pair, the unknown matrix  $\mathbf{Q}$  can be computed. Since the eqn.(3.2c) is homogeneous, the solution for each essential parameter will contain a scale factor  $\lambda$ . This also follows from eqn.(3.2a) where the equality is preserved if we multiply  $\mathbf{Q}$  (or  $\mathbf{Q}_v$ ) with a scalar. Note that the essential parameters are linear in  $t_x$ ,  $t_y$  and  $t_z$ . Thus, when eqn.(3.2c) is solved, we get the scale factor  $\lambda$  that cannot be determined. The scale factor  $\lambda$  influences 'z' in eqn.(3.1) but not the rotation parameters ([4]).

Re-arranging eqn.(3.2c) results in

$$\mathbf{N} \mathbf{Q}' = \mathbf{G}; \quad (3.3a)$$

where

$$\mathbf{G} = \text{col. } [-1, -1, -1, \dots ]^t ; \quad (3.3b)$$

$$\mathbf{N} = \text{col. } (\mathbf{N}_1, \mathbf{N}_2, \mathbf{N}_3 \dots ) ; \quad (3.3c)$$

$$\mathbf{N}_\alpha = [ X_\alpha X'_\alpha, Y_\alpha X'_\alpha, X'_\alpha X_\alpha, Y_\alpha Y'_\alpha, Y'_\alpha Y_\alpha, X_\alpha Y_\alpha ] ; \quad (3.3d)$$

$$\mathbf{Q}' = [ q_1/q_9, q_2/q_9, q_3/q_9, q_4/q_9, q_5/q_9, q_6/q_9, q_7/q_9, q_8/q_9 ]^t ; \quad (3.3e)$$

( $X_\alpha, Y_\alpha$ ) being the 2-D image-space co-ordinates for the  $\alpha$ 'th ( $\alpha = 1, 2, 3, \dots, n; n \geq 8$ ) data point.

Eqn. (3.3a) can be solved uniquely for the essential parameters from the knowledge of  $n$  ( $n \geq 8$ ) image point correspondence pairs. If eight data points are available,

$$\mathbf{Q}' = \mathbf{N}^{-1} \mathbf{G} ; \quad (3.4a)$$

provided  $\mathbf{N}$  is non-singular. If more than eight points are available, then the pseudo-inverse solution

$$\mathbf{Q}' = \mathbf{N}^+ \mathbf{G} = (\mathbf{N}^t \mathbf{N})^{-1} \mathbf{N}^t \mathbf{G} \quad (3.4b)$$

provides the least mean squares error solution.

There are certain *degenerate* eight-point configurations ( [4,5] ) for which the algorithm fails, because for these cases the matrix  $\mathbf{N}$  becomes singular. A configuration is degenerate if

- (i) as many as four of the points lie in a straight line, or
- (ii) more than six points are coplanar, for example, the vertices of a regular hexagon, or
- (iii) seven or more points are traversable by two planes with one plane containing the origin, such as eight points representing the vertices of a cube, or
- (iv) seven or more points lie on the surface of a cone containing the origin.

In general, the IPC algorithm works satisfactorily in almost all cases if the eight data points lie on more than two planes. The proof is given in Appendix B.

### 3.1.1. The First Method

Here, we are presenting the details of the *first method* for the IPC algorithm, where Huang *et al.* ( [5,24,25,26] ) have used the algorithm for the two-view motion analysis case.

We consider the estimation of motion parameters given  $\mathbf{Q}$  ([5]). Let the *singular value decomposition* of  $\mathbf{Q}$  ( or  $\mathbf{Q}'$  ) be given by

$$\mathbf{Q} = \mathbf{U} \Delta \mathbf{V}^t ; \quad (3.5)$$

where  $\mathbf{U}$  and  $\mathbf{V}$  are orthogonal matrices and  $\Delta$  is a diagonal matrix containing the singular values of  $\mathbf{Q}$ . Then, given  $\mathbf{Q}$ , there are two solutions for the rotation matrix,  $\mathbf{R}$  and  $\mathbf{R}'$ , given by

$$\mathbf{R} = \mathbf{U} \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & \omega \end{bmatrix} \mathbf{V}^t \quad (3.6a)$$

and

$$\mathbf{R}' = \mathbf{U} \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & \omega \end{bmatrix} \mathbf{V}^t . \quad (3.6b)$$

In eqns.(3.6a) and (3.6b),  $\omega = \det(\mathbf{U})/\det(\mathbf{V})$ . A solution for the translational vector  $\mathbf{T}$  (up to a scale factor) is given by

$$\mathbf{T} = \lambda \begin{bmatrix} (-\|\mathbf{l}_1\|^2 + \|\mathbf{l}_2\|^2 + \|\mathbf{l}_3\|^2)^{1/2} \\ (\|\mathbf{l}_1\|^2 - \|\mathbf{l}_2\|^2 + \|\mathbf{l}_3\|^2)^{1/2} \\ (\|\mathbf{l}_1\|^2 - \|\mathbf{l}_2\|^2 - \|\mathbf{l}_3\|^2)^{1/2} \end{bmatrix} ; \quad (3.6c)$$

where  $\lambda$  is a non-recoverable scale factor and  $\|\cdot\|^2$  indicates the norm operation. The proofs for eqns.(3.6a), (3.6b), and (3.6c) are given in Appendix C.

### 3.1.2. The Second Method

As shown by Zhuang *et al.* ([6,7]) :

$$\mathbf{T} \times \mathbf{r}' = \mathbf{T} \times \mathbf{R} \mathbf{r} ; \quad (3.7a)$$

or

$$\mathbf{z}' / z \mathbf{T} \times \mathbf{V}' = \mathbf{T} \times \mathbf{R} \mathbf{V} ; \quad (3.7b)$$

where 'x' denotes the cross-product. We obtain the basic motion equation from eqns.(3.7a) and (3.7b) as

$$\mathbf{V}' \cdot (\mathbf{T} \times \mathbf{R} \mathbf{V}) = 0 . \quad (3.7c)$$

For any 3x1 vector or matrix column  $\chi$

$$\mathbf{T} \times \chi = \Gamma \chi ; \quad (3.8a)$$

where  $\mathbf{T}$  is given by (3.6c) and  $\Gamma$  is a skew-symmetric matrix determined from

$$\mathbf{Q} = \Gamma \mathbf{R} \quad (3.8b)$$

and

$$\Gamma = \begin{bmatrix} 0 & t_z & -t_y \\ -t_z & 0 & t_x \\ t_y & -t_x & 0 \end{bmatrix} . \quad (3.8c)$$

If

$$\mathbf{R} = [ \mathbf{c}_1 \ \mathbf{c}_2 \ \mathbf{c}_3 ] ;$$

where  $\mathbf{c}_\alpha$  ( $\alpha = 1,2,3$ ) is the  $\alpha$ 'th column of  $\mathbf{R}$ , then

$$\Gamma \mathbf{R} = \Gamma [ \mathbf{c}_1 \ \mathbf{c}_2 \ \mathbf{c}_3 ] = [ \Gamma \mathbf{c}_1 \ \Gamma \mathbf{c}_2 \ \Gamma \mathbf{c}_3 ] = [ \mathbf{T} \times \mathbf{c}_1 \ \mathbf{T} \times \mathbf{c}_2 \ \mathbf{T} \times \mathbf{c}_3 ] \quad (3.9)$$

$$= \mathbf{T} \times [ \mathbf{c}_1 \ \mathbf{c}_2 \ \mathbf{c}_3 ] = \mathbf{T} \times \mathbf{R} .$$



The algorithm for the computation of motion parameters is implemented in the following steps :

**Step I : Compute**

$$a = (-\|l_1\|^2 + \|l_2\|^2 + \|l_3\|^2) / 2 ; \quad (3.10a)$$

$$b = (+\|l_1\|^2 - \|l_2\|^2 + \|l_3\|^2) / 2 \quad (3.10b)$$

and

$$c = (+\|l_1\|^2 + \|l_2\|^2 - \|l_3\|^2) / 2 . \quad (3.10c)$$

**Step II : If ( $|a| \geq |b|, |c|$ ), define**

$$T' = \begin{bmatrix} a^{1/2} \\ -(l_1, l_2) / a^{1/2} \\ -(l_1, l_3) / a^{1/2} \end{bmatrix} ; \quad (3.11)$$

$$p_1 = [ (l_2 \times l_3) \times l_1 + a^{1/2} (l_2 \times l_3) ] / (a^{1/2} \|T'\|^2) ; \quad (3.12a)$$

$$p_1' = [ (l_2 \times l_3) \times l_1 - a^{1/2} (l_2 \times l_3) ] / (-a^{1/2} \|T'\|^2) ; \quad (3.12b)$$

$$p_2 = (l_3 + b^{1/2} p_1) / a^{1/2} = (p_1 \times l_2) / a^{1/2} ; \quad (3.12c)$$

$$p_2' = (l_3 - b^{1/2} p_1') / (-a^{1/2}) = -(p_1' \times l_2) / a^{1/2} ; \quad (3.12d)$$

$$p_3 = (-l_2 + c^{1/2} p_1) / a^{1/2} = (p_1 \times l_3) / a^{1/2} ; \quad (3.12e)$$

$$p_3' = (-l_2 - c^{1/2} p_1') / (-a^{1/2}) = -(p_1' \times l_3) / a^{1/2} \quad (3.12f)$$

**GO TO Step V.**

**Step III : If ( $|b| \geq |c|, |a|$ ), define**

$$\mathbf{T}' = \begin{bmatrix} -(l_2, l_1) / b^{1/2} \\ b^{1/2} \\ -(l_2, l_3) / b^{1/2} \end{bmatrix}; \quad (3.13)$$

$$\mathbf{p}_2 = [ (l_3 \times l_1) \times l_2 + b^{1/2} (l_3 \times l_1) ] / (b^{1/2} \|\mathbf{T}'\|^2); \quad (3.14a)$$

$$\mathbf{p}_2' = [ (l_3 \times l_1) \times l_2 - b^{1/2} (l_3 \times l_1) ] / (-b^{1/2} \|\mathbf{T}'\|^2); \quad (3.14b)$$

$$\mathbf{p}_3 = (l_1 + c^{1/2} \mathbf{p}_2) / b^{1/2} = (\mathbf{p}_2 \times l_3) / b^{1/2}; \quad (3.14c)$$

$$\mathbf{p}_3' = (l_1 - c^{1/2} \mathbf{p}_2') / (-b^{1/2}) = -(\mathbf{p}_2' \times l_3) / b^{1/2}; \quad (3.14d)$$

$$\mathbf{p}_1 = (-l_3 + a^{1/2} \mathbf{p}_2) / b^{1/2} = (\mathbf{p}_2 \times l_1) / b^{1/2}; \quad (3.14e)$$

$$\mathbf{p}_1' = (-l_3 - c^{1/2} \mathbf{p}_2') / (-b^{1/2}) = -(\mathbf{p}_2' \times l_1) / b^{1/2}; \quad (3.14f)$$

**GO TO Step V.**

**Step IV :** If  $(|c| \geq |a|, |b|)$ , define

$$\mathbf{T}' = \begin{bmatrix} -(l_3, l_1) / c^{1/2} \\ -(l_3, l_2) / c^{1/2} \\ c^{1/2} \end{bmatrix}; \quad (3.15)$$

$$\mathbf{p}_3 = [ (l_1 \times l_2) \times l_3 + c^{1/2} (l_1 \times l_2) ] / (c^{1/2} \|\mathbf{T}'\|^2); \quad (3.16a)$$

$$\mathbf{p}_3' = [ (l_1 \times l_2) \times l_3 - c^{1/2} (l_1 \times l_2) ] / (-c^{1/2} \|\mathbf{T}'\|^2); \quad (3.16b)$$

$$\mathbf{p}_1 = (l_2 + a^{1/2} \mathbf{p}_3) / c^{1/2} = (\mathbf{p}_3 \times l_1) / c^{1/2}; \quad (3.16c)$$

$$\mathbf{p}_1' = (l_2 - a^{1/2} \mathbf{p}_3') / (-c^{1/2}) = -(\mathbf{p}_3' \times l_1) / c^{1/2}; \quad (3.16d)$$

$$\mathbf{p}_2 = (-l_1 + b^{1/2} \mathbf{p}_3) / c^{1/2} = (\mathbf{p}_3 \times l_2) / c^{1/2}; \quad (3.16e)$$

$$\mathbf{p}_2' = (-l_1 - b^{1/2} \mathbf{p}_3') / (-c^{1/2}) = -(\mathbf{p}_3' \times l_2) / c^{1/2}; \quad (3.16f)$$

**Step V : Let**

$$\mathbf{R} = \text{col.} ( \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3 ) \quad (3.17a)$$

and

$$\mathbf{R}' = \text{col.} ( \mathbf{p}'_1, \mathbf{p}'_2, \mathbf{p}'_3 ) \quad (3.17b)$$

The proofs for all the developments mentioned in this section have been given in Appendix D.

### 3.1.3. The Third Method

Another method for IPC algorithm has been shown by H. C. Longuet-Higgins ([4]).

The algorithm can be implemented by rewriting eqn.(3.2a) as

$$\mathbf{V}^t \mathbf{H} \mathbf{V}' = 0 ; \quad (3.18)$$

where  $\mathbf{H} = \mathbf{Q}^t$ . Thus, the linear eqn.(3.18) is to be solved. We shall introduce the algorithm step by step. For convenience, the length of the vector  $\mathbf{T}$  is adopted as the unit of distance, so that  $\|\mathbf{T}\|^2 = 1$ .

**Step I :** Compute the matrix  $\mathbf{H}^t \mathbf{H}$  where

$$\mathbf{H}^t \mathbf{H} = \mathbf{Q} \mathbf{Q}^t = \Gamma \Gamma^t = -\Gamma^2 = \begin{bmatrix} 1-t_x^2 & -t_x t_y & -t_x t_z \\ -t_y t_x & 1-t_y^2 & -t_y t_z \\ -t_z t_x & -t_z t_y & 1-t_z^2 \end{bmatrix}. \quad (3.19)$$

**Step II :** Normalize the elements of  $\mathbf{H}$  by dividing these by  $(0.5 \times \text{trace of } \mathbf{H}^t \mathbf{H})^{1/2}$ . Choose any arbitrary sign for  $\mathbf{H}$  at this stage.

**Step III :** Define

$$\mathbf{h}_\alpha = \mathbf{T} \times \mathbf{c}_\alpha \quad (3.20a)$$

and

$$\mathbf{W}_\alpha = \mathbf{h}_\alpha \times \mathbf{T} \quad (\alpha = 1,2,3); \quad (3.20b)$$

where

$\mathbf{h}_\alpha$  is the  $\alpha$ 'th row of  $\mathbf{H}$  ;

$\mathbf{c}_\alpha$  is the  $\alpha$ 'th column of  $\mathbf{R}$  and

$\mathbf{W}_\alpha$  is a new vector.

Then we can solve for the rows of  $\mathbf{R}^t$  from

$$\mathbf{c}_\alpha = \mathbf{W}_\alpha + \mathbf{W}_\beta \times \mathbf{W}_\gamma. \quad (3.21)$$

The rotation matrix  $\mathbf{R}$  is then determined from the transpose of the matrix  $\mathbf{R}^t$  found from eqn.(3.21). The other solution for the rotation matrix is found by changing the sign of  $\mathbf{H}$  or  $\mathbf{h}_\alpha$  in eqn.(3.20a), or equivalently that of  $\mathbf{T}$ , and then solving eqn.(3.21) as before. The proof for eqn.(3.21) is given in Appendix E.

The three methods discussed in this chapter yield two solutions for the rotation matrix,  $\mathbf{R}$  and  $\mathbf{R}'$ , and a unique solution for the translational vector  $\mathbf{T}$  ( up to a scale factor ). The correct solution for the rotation matrix is found through a test discussed in chapter IV. The flow diagram for the computation of  $\mathbf{T}$ ,  $\mathbf{R}$ , and  $\mathbf{R}'$ , is given in Fig.5. The computation of the motion parameters from the translational vector and the rotation matrix are given in chapter IV and Appendix A.

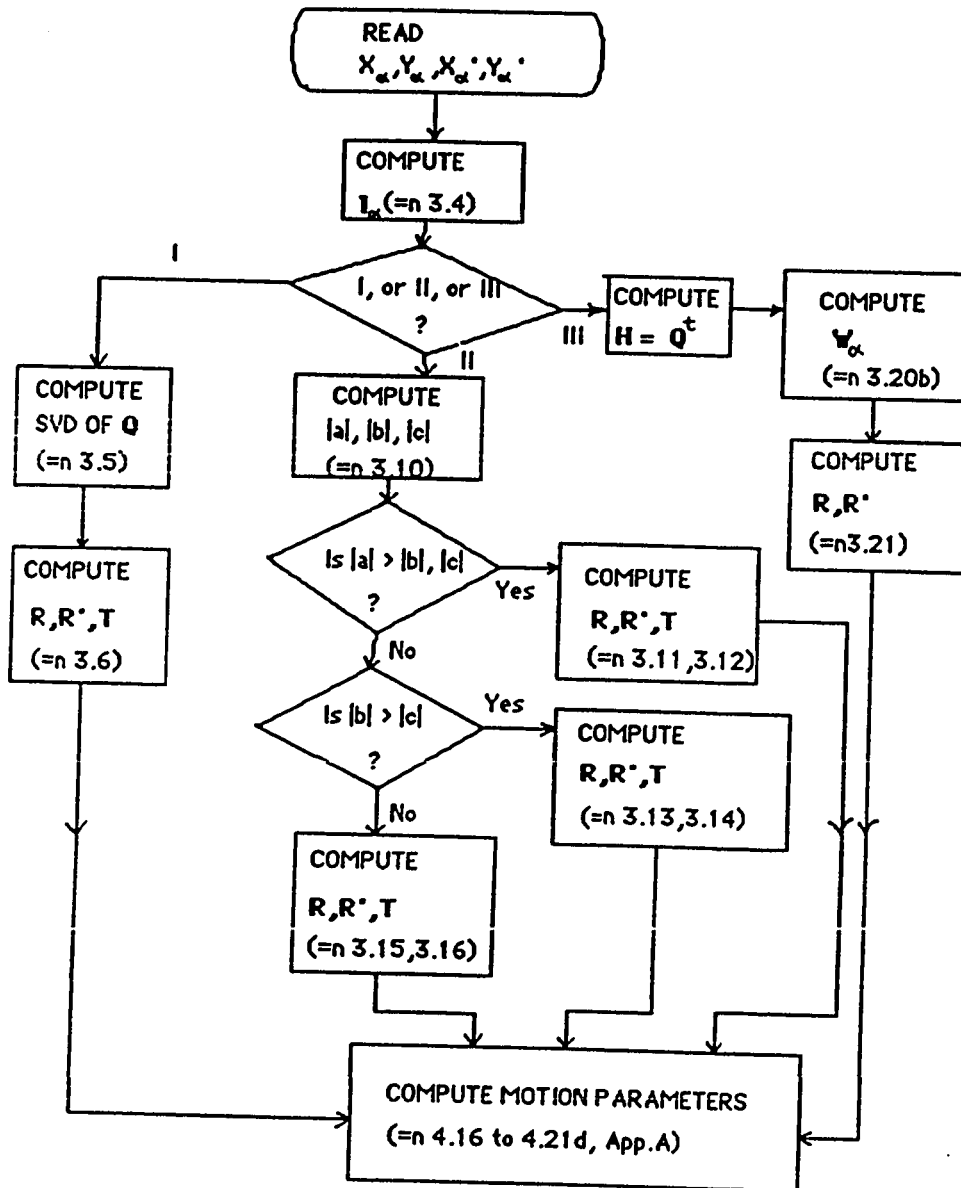


Fig.5: Flow Chart for the Computation of Motion Parameters using the three methods for the IPC Algorithm.

## CHAPTER IV

### THE GENERALIZED IMAGE POINT CORRESPONDENCE ALGORITHM

In a practical situation, when the second image of the object is compared with its first image or the database in the computer memory, the orientation and the location of the camera are seldom the same. The actual situation a robot faces is an important example. In an industry, a robot's job can be to track and pick up machine tools kept on a moving conveyor belt. The space applications of robots include grasping a rotating satellite for repairs, or assembling the Space Station. In many such applications, the robot takes pictures of the changing scenes, at different instants of time, while it is moving. Thus the IPC algorithm, which is applied to the elementary forms of motion analysis, needs an extension for the generalized motion analysis where both the scene and the camera are moving. In this chapter, we consider this general problem. The known correspondences of the images of a set of points on the object obtained by the camera at different instants of time are utilized in this development. The method of solution is called the **Generalized Image Point Correspondence ( GIPC )** algorithm.

Special cases of the above problem, discussed in the previous chapters, are :

- (i) the determination of motion parameters of a moving object by a stationary camera ( two-view motion analysis );
- (ii) the determination of position and orientation of a fixed object from pictures obtained by two cameras ( stereoscopic vision );
- (iii) the determination of position and orientation from pictures from a single moving camera taken at different instants of time ( stereo motion ).

#### 4.1. THE ALGORITHM

The general case of motion analysis is illustrated in Fig.6. For simplicity in presentation, we consider in detail the equations that track a single point P on a moving

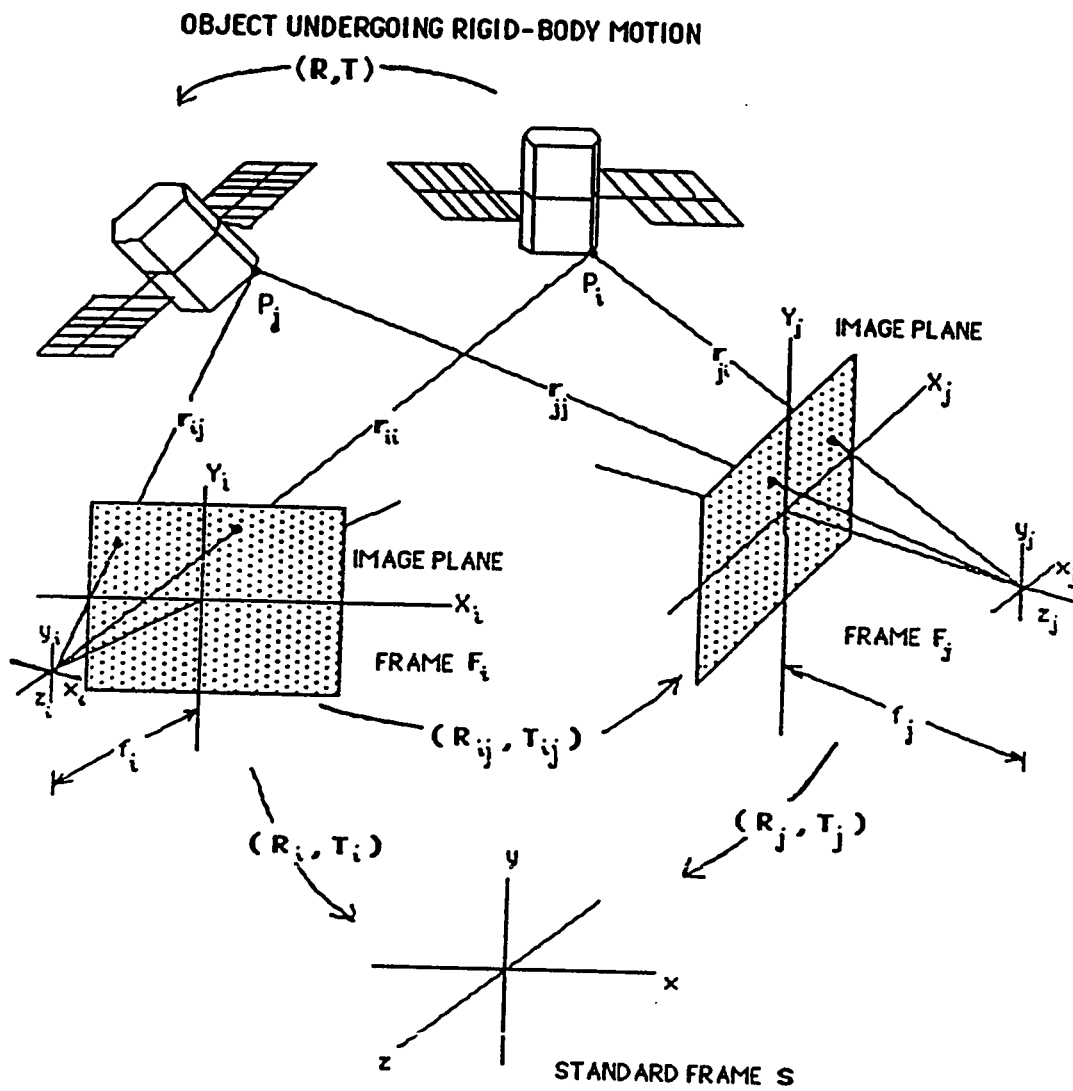


Fig.6: Geometry illustrating the GIPC Algorithm.

object by a moving camera.  $F_i$  and  $F_j$  are the two frames with which the camera coordinate system coincides at two different instants of time  $t_i$  and  $t_j$  ( $t_j > t_i$ ) respectively. The point  $P$  moves from the position  $P_i$  to another position  $P_j$  due to the rigid-body motion of the object. We assume  $(R_i, T_i)$  and  $(R_j, T_j)$  to be the transformation parameters (rotation and translation) that link the frames  $F_i$  and  $F_j$  respectively with the standard frame  $S$ . Also, let  $(R_{ij}, T_{ij})$  be the transformation parameters that link the frame  $F_i$  with the frame  $F_j$ . The object moves with the unknown motion parameters  $(R, T)$ . The various transformation parameters for 'n' frames have been shown in Fig.7. Next, as before, let

$r = (x, y, z)^t$  be the co-ordinates of  $P_i$  relative to the standard frame  $S$ ;

$r' = (x', y', z')^t$  be the co-ordinates of  $P_j$  relative to the standard frame  $S$ ;

$r_{ii} = (x_{ii}, y_{ii}, z_{ii})^t$  be the co-ordinates of  $P_i$  with respect to the frame  $F_i$ ;

$r_{ij} = (x_{ij}, y_{ij}, z_{ij})^t$  be the co-ordinates of  $P_j$  with respect to the frame  $F_i$ ;

$r_{ji} = (x_{ji}, y_{ji}, z_{ji})^t$  be the co-ordinates of  $P_i$  with respect to the frame  $F_j$  and

$r_{jj} = (x_{jj}, y_{jj}, z_{jj})^t$  be the co-ordinates of  $P_j$  with respect to the frame  $F_j$ .

To carry out a detailed analysis of the problem, the relationship between the three sets of frames, in terms of the co-ordinates of the point  $P$ , needs to be established. For this purpose, the problem is split into two steps - one corresponding to the initial position  $P_i$  and the other to the final position  $P_j$ .

### ***Initial Position of the Object :***

The transformation from the frame  $F_i$  to the frame  $F_j$ , as given by eqn.(2.3), is

$$\Phi_i r_{ii} = \Phi_i R_{ij} r_{ji} + T_{ij}; \quad (4.1)$$

where  $\Phi_i$  is the matrix of the basis vectors of the frame  $F_i$ .



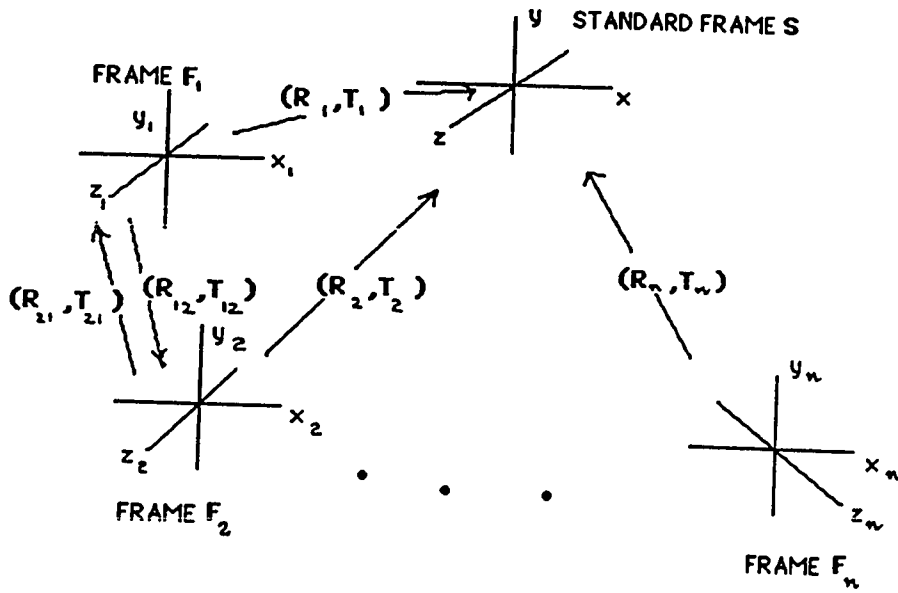


Fig.7: Geometry illustrating the use of 'n' frames for the GIPC Algorithm and their relationship with each other.

The transformations from  $F_i$  to  $S$ , and  $F_j$  to  $S$  are special cases of transformation shown in eqn.(4.1), and are expressed respectively as :

$$r_{ii} = R_i r + T_i \quad (4.2a)$$

and

$$r_{jj} = R_j r + T_j . \quad (4.2b)$$

***Final Position of the Object :***

The motion of the object relates  $r$  and  $r'$  as follows :

$$r' = R r + T ; \quad (4.3)$$

As in eqn.(4.1), the transformation from  $F_i$  to  $F_j$  is expressed as

$$\Phi_i r_{ij} = \Phi_i R_{ij} r_{jj} + T_{ij} ; \quad (4.4)$$

The transformations from  $F_i$  to  $S$ , and  $F_j$  to  $S$  are special cases of eqn.(4.4) and are expressed as :

$$r_{ij} = R_i r' + T_i . \quad (4.5a)$$

and

$$r_{jj} = R_j r' + T_j . \quad (4.5b)$$

This results in two sets of equations (4.1) and (4.3,4.4), corresponding to two positions of the object. From either set of equations, we find the relationship between  $( R_{ij} , T_{ij} )$ ,  $( R_i , T_i )$  and  $( R_j , T_j )$  as (Fig.6)

$$R_{ij} = R_i R_j^{-1} \quad (4.6a)$$

and

$$T_{ij} = \Phi_i ( T_i - R_i R_j^{-1} T_j ) , \quad (4.6b)$$

The pictures of the moving object are taken by a moving camera at discrete instants of time. For this general motion problem, we have two different cases. In the first case, the initial picture of the moving object is taken by the camera at a location. Then the camera is moved to another location so that the object is in the field of view when the second picture is taken. In the second case, the camera can be moved first to another location in order to take the second picture of the object. These two cases are found to be identical in terms of the motion equation. Furthermore, the three types of motion discussed in chapter III will be shown as special cases of the general motion problem.

#### 4.1.1. The First Case

The motion of the object before the motion of the camera has been assumed in this case. The relationship between the initial and final positions of the point P is expressed by the following set of motion equations :

$$\mathbf{r}' = \mathbf{R} \mathbf{r} + \mathbf{T}$$

for the motion of the object from eqn.(4.3), and

$$\Phi_i \mathbf{r}_{ij} = \Phi_i \mathbf{R}_{ij} \mathbf{r}_{ji} + \mathbf{T}_{ij}$$

for the motion of the camera from eqn.(4.4).

The desired relationship between  $\mathbf{r}_{jj}$  and  $\mathbf{r}_{ii}$ , the co-ordinates of the initial and the final positions of the point P recorded by the camera, with respect to the frames  $\mathbf{F}_j$  and  $\mathbf{F}_i$  respectively, is given in the following equations (Appendix F) :

$$\mathbf{r}_{jj} = \mathbf{R}_{ij}' \mathbf{r}_{ii} + \mathbf{T}_{ij}' ; \quad (4.7a)$$

where

$$\mathbf{R}_{ij}' = \mathbf{R}_{ij}^t \mathbf{R}_i \mathbf{R} \mathbf{R}_i^t = \mathbf{R}_j \mathbf{R} \mathbf{R}_i^t ; \quad (4.7b)$$

$$\mathbf{T}_{ij}' = -\mathbf{R}_j \mathbf{R} \mathbf{R}_i^{-1} \mathbf{T}_i + \mathbf{R}_j \mathbf{T} + \mathbf{T}_j . \quad (4.7c)$$

Eqn.(4.7a) gives the expression for the *generalized version of the motion equation*, when the object moves before the camera moves. We shall derive (Appendix F) special cases of motion analysis from the eqns.(4.7a,4.7b,4.7c).

**Special Cases** : If  $F_i$  coincides with  $S$ , the motion equation can be written as

$$\mathbf{r}_{jj} = \mathbf{R}_{ij}' \mathbf{r}_{ii} + \mathbf{T}_{ij}' ; \quad (4.8a)$$

where

$$\mathbf{R}_{ij}' = \mathbf{R}_{ij}^{-1} \mathbf{R} = \mathbf{R}_j \mathbf{R} \quad (4.8b)$$

and

$$\mathbf{T}_{ij}' = \mathbf{R}_j \mathbf{T} + \mathbf{T}_j . \quad (4.8c)$$

The IPC algorithm can be used to estimate the motion parameters (  $\mathbf{R}_{ij}$  ,  $\mathbf{T}_{ij}$  ), and hence (  $\mathbf{R}$  ,  $\mathbf{T}$  ), of the moving object, assuming  $\mathbf{R}_{ij}$  and  $\mathbf{T}_{ij}$  are known.

(i) *Monocular vision case* : Eqn.(4.8a) reduces to the case of the two-view motion-equation when the location of the camera, taking the pictures of the moving object, is fixed. in that case,

$$\mathbf{R}_i = \mathbf{R}_j = \mathbf{R}_{ij} = \mathbf{I} \text{ and } \mathbf{T}_{ij} = \mathbf{T}_j = \mathbf{O} . \quad (4.9)$$

Therefore, the generalized motion-equation reduces to

$$\mathbf{r}_{jj} = \mathbf{R} \mathbf{r}_{ii} + \mathbf{T} . \quad (4.10a)$$

or, to the more familiar two-view motion equation

$$\mathbf{r}' = \mathbf{R} \mathbf{r} + \mathbf{T} \quad (4.10b)$$

as the frames  $\bar{F}_i$  and  $F_j$  coincide with the frame  $S$ , so that  $\mathbf{r}_{ii} = \mathbf{r}$  and  $\mathbf{r}_{jj} = \mathbf{r}'$ .

(ii) *Stereo vision case* : For a stereo vision case, the object is assumed to be stationary. In that case,

$$\mathbf{R} = \mathbf{I} ; \mathbf{R}_{ij} = \mathbf{R}_j \text{ and } \mathbf{T} = \mathbf{O} \quad (4.11)$$

and the generalized motion equation reduces to

$$\mathbf{r}_{jj} = \mathbf{R}_j \mathbf{r}_{ii} + \mathbf{T}_j \quad (4.12)$$

as should be the case.

In another special case of stereo vision, the object is moving and the cameras are rigidly fixed to each other in a manner that their optical axes are parallel, as shown in Fig.4. In this case the frame  $F_i$  coincides with the standard frame  $S$  so that

$$\mathbf{R}_{ij} = \mathbf{R}_j = \mathbf{I} ; \mathbf{T}_{ij} = \mathbf{T}_j = \text{col. } (-d, 0, 0) . \quad (4.13)$$

#### 4.1.2. The Second Case

In this case, the motion of the camera before that of the object has been assumed. The relationship that exists between  $\mathbf{r}_{jj}$  and  $\mathbf{r}_{ii}$  is found to be (Appendix F) :

$$\mathbf{r}_{jj} = \mathbf{R}_{ij}' \mathbf{r}_{ii} + \mathbf{T}_{ij}' ; \quad (4.14a)$$

where

$$\mathbf{R}_{ij}' = \mathbf{R}_j \mathbf{R} \mathbf{R}_i^t \quad (4.14b)$$

and

$$\mathbf{T}_{ij}' = -\mathbf{R}_j \mathbf{R} \mathbf{R}_i^t \mathbf{T}_i + \mathbf{R}_j \mathbf{T} + \mathbf{T}_j . \quad (4.14c)$$

As is clear from eqns.(4.8a, 4.8b, 4.8c) and (4.14a,4.14b,4.14c), it does not matter whether the object or the camera is moved first.

## 4.2. THE DETERMINATION OF THE MOTION PARAMETERS

Some of the definitions used in the following sections are presented by H. C. Longuet-Higgins ([4]) and Zhuang *et al.* ([5,6,7]). The motion parameters to be determined are

### 4.2.1. The Object Surface Shape

The shape of the object surface is found by a map of relative depths of 3D object surface points by either of the two methods discussed in what follows :

*Method I* : From the generalized motion equation, for any two consecutive frames  $F_i$  and  $F_{i+1}$ , one finds that ( [4] )

$$X_{i+1} = x_{i+1} / z_{i+1} = \frac{\mathbf{p}_1 \cdot (\mathbf{r}_i - \mathbf{T}')}{\mathbf{p}_3 \cdot (\mathbf{r}_i - \mathbf{T}')} = \frac{\mathbf{p}_1 \cdot (\mathbf{V}_i - \mathbf{T}' / z_i)}{\mathbf{p}_3 \cdot (\mathbf{V}_i - \mathbf{T}' / z_i)} ; \quad (4.15)$$

where  $\mathbf{p}_\alpha$  ( $\alpha = 1,2,3$ ) is the  $\alpha$ 'th row of the rotation matrix  $\mathbf{R}_{i,i+1}'$ . Thus, the 'z' coordinates (or the depth) of the features, from their 2D image points, can be found from the following equation :

$$z_{\lambda,i} = \frac{(\mathbf{p}_1 - X_{i+1} \mathbf{p}_3) \cdot \mathbf{T}'}{(\mathbf{p}_1 - X_{i+1} \mathbf{p}_3) \cdot \mathbf{V}_i} = \lambda z_i . \quad (4.16a)$$

Therefore,

$$x_{\lambda,i} = z_{\lambda,i} X_i = \lambda x_i \quad \text{and} \quad y_{\lambda,i} = z_{\lambda,i} Y_i = \lambda y_i . \quad (4.16b)$$

The eqns.(4.16a,4.16b) give the object surface shape at instant  $t_i$ , with respect to the frame  $F_i$ . The shape of the object surface is the set of 3D object-space coordinates of the surface points relative to a frame. Thus, the shape of an object before motion can be determined, but not its size, due to the non-measurable scale factor  $\lambda$  involved. In order to determine the shape of the object surface after the object has moved ( i.e. with respect to the frame  $F_{i+1}$ , one has to compute ' $x_{i+1}$ ' and ' $y_{i+1}$ ' from

$$x_{\lambda,i+1} = z_{\lambda,i+1} X_{i+1} \quad \text{and} \quad y_{\lambda,i+1} = z_{\lambda,i+1} Y_{i+1} . \quad (4.16c)$$

*Method II* : The relative depth from the two consecutive frames is determined by ([6,7])

$$z_{i+1} / z_i = \| \mathbf{T}' \times \mathbf{R} \mathbf{V}_i \| / \| \mathbf{T}' \times \mathbf{V}_{i+1} \| \quad (4.17a)$$

when  $\mathbf{T}' \times \mathbf{V}_{i+1} \neq 0$ , which, in turn, means that if ' $z_i$ ' is known we could find ' $z_{i+1}$ ' or the surface shape of the object after motion. This information may then be used for the identification of the object in motion with any other object whose shape information is stored in the computer.

When the translation  $\mathbf{T}$  is not zero and is known, the absolute depths ' $z_i$ ' and ' $z_{i+1}$ ' could be determined from the following equations :

$$z_{i+1} = - \| \mathbf{T} \times \mathbf{R} \mathbf{V}_i \| / \| \mathbf{V}_{i+1} \times \mathbf{R} \mathbf{V}_i \| \quad (4.17b)$$

and

$$z_i = - \| \mathbf{T} \times \mathbf{V}_{i+1} \| / \| \mathbf{V}_{i+1} \times \mathbf{R} \mathbf{V}_i \| ; \quad (4.17c)$$

As before, the 'x' and 'y' coordinates of the features, whose 2D image coordinates were available, can be found when their 'z' coordinates are computed.

For finding the shape of the object surface one needs to know the correct solution for the rotation matrix  $\mathbf{R}$ . If the coordinates ' $z$ ' and ' $z$ ' of any point are both of the same sign, the corresponding rotation matrix ( $\mathbf{R}$ ) is considered. When the signs are opposite, the other rotation matrix ( $\mathbf{R}'$ ) is considered as the final result ([4,5]).

The two solutions for the rotation matrix are found since the matrix  $\mathbf{Q}$  has two and only two decompositions ([7])

$$\mathbf{Q} = \mathbf{T}' \times \mathbf{R} = (-\mathbf{T}') \times \mathbf{R}' ; \quad (4.18)$$

where  $\mathbf{T}'$  is any real vector or equals  $\lambda \mathbf{T}$ .

One of the rotation matrices corresponds to the motion of the object on the same side of the camera ( either front or back ) and the other one to the motion from either front to the back or back to the front. In our analysis, we reject the second solution for the rotation matrix corresponding to a situation not encountered in practice.

#### 4.2.2. Range/Interframe Range Rate

The *range* of the available features, which determines the depth of object surface (up to a scale factor at instant  $t_i$  ), is defined as

$$\mathfrak{R}_i = z_{\lambda_i} \quad (4.19a)$$

as conventionally, the object is viewed along the z-axis of the camera coordinate system. The *interframe range rate* ( up to a scale factor ) between any two consecutive frames  $F_i$  and  $F_{i+1}$  at instants  $t_i$  and  $t_{i+1}$ , is given by

$$\dot{\mathfrak{R}}_{i,j+1} = \frac{\mathfrak{R}_{i+1} - \mathfrak{R}_i}{\Delta t} = \frac{z_{\lambda_{i+1}} - z_{\lambda_i}}{\Delta t} \quad (\Delta t = t_{i+1} - t_i) . \quad (4.19b)$$

#### 4.2.3. Interframe Attitude/Attitude Rates

The rotation matrix  $\mathbf{R}$  can be estimated between the two consecutive frames  $F_i$  and  $F_{i+1}$  from the following equations :

$$\mathbf{R} = \mathbf{R}_i^{-1} \mathbf{R}_{i,i+1} \mathbf{R}_{i,i+1}' \mathbf{R}_i = \mathbf{R}_{i+1}^{-1} \mathbf{R}_{i,j+1}' \mathbf{R}_i \quad (4.20)$$

assuming the relative transformation parameters between these frames is known.

Once the rotation matrix  $\mathbf{R}$  is estimated, the *attitude* parameters of the object are computed. The details are given in Appendix A.

The *interframe attitude rates* between the two frames are given as

$$\text{the angular velocity of rotation} , \omega_{\theta} = (\theta_{i+1} - \theta_i) / \Delta t \quad (4.21a)$$

if the first representation of  $\mathbf{R}$  is used ( Appendix A ). If the second representation of  $\mathbf{R}$  is used, then



$$\text{roll rate} , \omega_{\theta} = ( \theta_{i+1} - \theta_i ) / \Delta t ; \quad (4.21b)$$

$$\text{yaw rate} , \omega_{\phi} = ( \phi_{i+1} - \phi_i ) / \Delta t \text{ and} \quad (4.21c)$$

$$\text{pitch rate} , \omega_{\psi} = ( \psi_{i+1} - \psi_i ) / \Delta t . \quad (4.21d)$$

No sudden changes in the interframe attitude parameters are assumed.

#### 4.2.4. Object Tracking

The process of tracking can be divided into two tasks : *acquisition* and *tracking proper* ( see Gennery [8]). In acquisition portion, the object is located in the scene and its attitude/attitude rate parameters are determined to help in the process of recognition later. This portion is divided into three major portions : feature tracking, the stereo solution, and matching to the object model. The features are detected and tracked over several frames, corresponding to different instants of time. These features are matched between consecutive frames, and the motion parameters are estimated by the stereo solution using the GIPC algorithm. From the 'z' co-ordinate of any feature, distance  $r_i$  with respect to the reference frame is determined to a scale factor. The scale factor is removed when the feature is projected onto the image plane of the camera. Thus, knowing the motion parameters between the two consecutive frames, it is possible to estimate the position of the feature, on the image plane, in the next frame. Each time this information about the motion parameters is fed to the tracking proper phase. This completes the tracking portion of the problem.

The process of recognition is carried out after the object surface is reconstructed, and hence identified, from a knowledge of 3D object space coordinates of the features that reside on the surface of the object. Since we find the 3D coordinates up to a scale factor, the object can be matched, and recognized, with a reference object in the main computer memory to a scale factor. For example, primitives like a cube can be identified

to be a cube stored in the computer memory. It is the shape that matters for the process of recognition, not its size.

## CHAPTER V

### ERROR ANALYSIS

#### 5.1. PRELIMINARY REMARKS

##### 5.1.1. Sources of Error

Since the IPC/GIPC algorithm can be implemented using a sequence of images of an object, any error in the input data and sensor parameters becomes a source of inaccuracy in the output data. The input data is the set of feature co-ordinates of the object, and the output data are the 3-D motion parameters. The perturbation errors, as shown by Ray *et al.* ( [28] ), are due to :

- (i) spurious noise in the sensor ( camera ) and its display ( monitor ),
- (ii) defocusing,
- (iii) limited resolution of the camera,
- (iv) motion blur,
- (v) thermal instability of light sources,
- (vi) optical system aberrations,
- (vii) diffraction effects,
- (viii) stray light, and
- (ix) optical properties of the medium.

In this chapter, we shall analyze the effect of changes in the input parameters on the output parameters. The three methods for the IPC/GIPC algorithm, as presented in chapter III, have been presented separately. Since various steps are involved in these methods, the propagation of the error will be studied, and the error bounds found, at each stage of the algorithm ( see the flow chart in Fig.5 ). Before the sensitivity of the IPC/GIPC algorithm to error in the input parameters can be analyzed, we shall define

*error, relative error, and error bounds.*

### 5.1.2. Definitions

The following definitions are taken from Hildebrand ( [29] ).

#### 5.1.2.1. Error, Relative error

If  $N$  is true value of any parameter,  $\bar{N}$  its approximation, then

$$\text{Error, } E = \text{True Value} - \text{Approx.} = N - \bar{N} \quad (5.1a)$$

and

$$\text{Relative Error, } RE = \frac{\text{True Value} - \text{Approx}}{\text{True Value}} = \frac{N - \bar{N}}{N} \quad (5.1b)$$

#### 5.1.2.2. Error bounds

When any function  $f(N)$ , with a continuous derivative, is evaluated with  $N$  replaced by an approximation  $\bar{N}$ , the relation

$$f(N) - f(\bar{N}) = f'(\eta) (N - \bar{N}) ; \bar{N} \leq \eta \leq N ;$$

gives us

$$| E (f(\bar{N})) | \leq | f'(\eta) |_{\max} | E(N) | \quad (5.2a)$$

$$| RE (f(N)) | \leq \frac{| f'(\eta) |_{\max}}{| f(N) |} | E(N) | \quad (5.2b)$$

The maximum error in the product  $P = N_1 N_2$  is found to be

$$RE(P) = \frac{N_1 N_2 - \bar{N}_1 \bar{N}_2}{N_1 N_2} = 1 - (1 - RE_1)(1 - RE_2) \quad (5.3a)$$

i.e.,

$$RE(P) = RE_1 + RE_2 - RE_1 RE_2 \quad (5.3b)$$

where  $RE(P)$  refers to  $P$ ,  $RE_1$  to  $N_1$  and  $RE_2$  to  $N_2$ .  $| RE(P) |$  is largest when

$RE_1$  and  $RE_2$  are negative. Thus, generally if  $P = N_1, N_2, \dots, N_m$ ,

$$RE(P) = 1 - [(1-RE_1)(1-RE_2)\dots(1-RE_m)] \quad (5.3c)$$

### 5.1.2.3. Errors in simultaneous equations

In order to investigate errors, we suppose that the set actually solved is

$$\begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix} \quad (5.4a)$$

whereas the *true* values of the coefficients are  $a_{\alpha\beta} + \delta a_{\alpha\beta}$ , and the *true* values of the right-hand members are  $c_\alpha + \delta c_\alpha$ . If the *true* values of the unknowns are denoted by  $x_\alpha + \delta x_\alpha$ , we see that

$$\begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} \delta x_1 \\ \vdots \\ \delta x_n \end{bmatrix} = \begin{bmatrix} \delta c_1 \\ \vdots \\ \delta c_n \end{bmatrix} - \begin{bmatrix} \delta a_{11} & \dots & \delta a_{1n} \\ \vdots & \ddots & \vdots \\ \delta a_{n1} & \dots & \delta a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \quad (5.4b)$$

assuming the products of errors, of the form  $(\delta a_{\alpha\gamma})(\delta x_\gamma)$ , to be relatively negligible. Thus, if the errors  $\delta a_{\alpha\beta}$  and  $\delta c_\alpha$  were known, the solution errors  $\delta x_\alpha$  would be obtained by solving a set of equations which differs from the set actually solved only in that the right-hand member  $c_\alpha$  is to be replaced by  $\eta_\alpha$  where

$$\eta_\alpha = \delta c_\alpha - (x_\alpha \delta a_{\alpha 1} + \dots + x_n \delta a_{\alpha n}). \quad (5.5)$$

In practice, the errors  $\delta a_{\alpha\beta}$  and  $\delta c_\alpha$  do not exceed a certain positive no., say  $\epsilon$ , in magnitude, so that

$$-\epsilon \leq \delta a_{\alpha\beta}, \delta c_\alpha \leq \epsilon.$$

In such cases, we are certain only that

$$|\eta_\alpha| \leq E, \quad (5.6a)$$

where

$$E = [ 1 + |x_1| + |x_2| + \dots + |x_n| ] \epsilon. \quad (5.6b)$$

Therefore,

$$\delta x_\gamma = \tilde{A}_{\gamma 1} \eta_1 + \tilde{A}_{\gamma 2} \eta_2 + \dots + \tilde{A}_{\gamma n} \eta_n \quad (\gamma = 1, 2, \dots, n) \quad (5.7a)$$

and

$$|\delta x_\gamma| \leq \left[ |\tilde{A}_{\gamma 1}| + |\tilde{A}_{\gamma 2}| + \dots + |\tilde{A}_{\gamma n}| \right] E \quad (5.7b)$$

where  $\tilde{A}_{\gamma\alpha}$  ( $\alpha = 1, 2, \dots, n$ ) are the elements of the  $\gamma$ 'th row of the inverse of the coefficient matrix. Thus, if the inverse matrix is calculated, approximate upper bounds on the effects of input errors are obtained from the above equation. They are not *strictly* upper bounds. However, they may be accepted as close approximations to true upper bounds.

## 5.2. ERROR PROPAGATION IN THE PRESENT ALGORITHMS

In this section, the error propagation through various stages in the present algorithms is investigated. The sensitivity of the *essential elements* to the error will be studied before the effects of error on the motion parameters, found by using the three methods for the IPC algorithm, are presented ( Fig.5 ).

### 5.2.1. Error in matrix Q due to error in feature co-ordinates

For convenience, the matrices **Q** and **R** have been redefined as

$$\mathbf{Q} = \begin{bmatrix} q_1 & q_2 & q_3 \\ q_4 & q_5 & q_6 \\ q_7 & q_8 & q_9 \end{bmatrix} = \begin{bmatrix} q_1 & q_2 & q_3 \\ q_4 & q_5 & q_6 \\ q_7 & q_8 & q_9 \end{bmatrix} = \begin{bmatrix} q_{11} & q_{12} & q_{13} \\ q_{21} & q_{22} & q_{23} \\ q_{31} & q_{32} & q_{33} \end{bmatrix} \quad (5.8a)$$

and

$$\mathbf{R} = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix} = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}. \quad (5.8b)$$

In our case, the equation actually solved is ( eqn.(3.3a) )

$$\mathbf{N} \mathbf{Q}' = \mathbf{G} \quad (5.9a)$$

and the true equation to be solved is

$$\tilde{\mathbf{N}} \tilde{\mathbf{Q}}' = \mathbf{G} ; \quad (5.9b)$$

where  $\tilde{\mathbf{N}} = \mathbf{N} + \delta\mathbf{N}$  and  $\tilde{\mathbf{Q}}' = \mathbf{Q}' + \delta\mathbf{Q}'$  . Therefore,

$$\mathbf{N} \mathbf{Q}' + \delta\mathbf{N} \mathbf{Q}' + \mathbf{N} \delta\mathbf{Q}' = \mathbf{G} \quad (5.9c)$$

where the products of errors are neglected, and  $\mathbf{G}$ ,  $\mathbf{Q}'$ ,  $\mathbf{N}$  and  $\mathbf{N}_\alpha$  are given by eqns.(3.3b,3.3c,3.3d,3.3e). Let

$$\delta\mathbf{N}_\alpha = ( \delta a_{\alpha 1}, \delta a_{\alpha 2}, \delta a_{\alpha 3}, \dots, \delta a_{\alpha 8} ) ; \quad (5.10a)$$

where

$$\delta a_{\alpha 1} = \delta X_{\alpha'} X_\alpha + \delta X_\alpha X_{\alpha'} ; \quad (5.10b)$$

$$\delta a_{\alpha 2} = \delta X_{\alpha'} Y_\alpha + \delta Y_\alpha X_{\alpha'} ; \quad (5.10c)$$

$$\delta a_{\alpha 3} = \delta X_{\alpha'} ; \quad (5.10d)$$

$$\delta a_{\alpha 4} = \delta Y_{\alpha'} X_\alpha + \delta X_\alpha Y_{\alpha'} ; \quad (5.10e)$$

$$\delta a_{\alpha 5} = \delta Y_{\alpha'} Y_\alpha + \delta Y_\alpha Y_{\alpha'} ; \quad (5.10f)$$

$$\delta a_{\alpha 6} = \delta Y_{\alpha'} ; \quad (5.10g)$$

$$\delta a_{\alpha 7} = \delta X_\alpha ; \quad (5.10h)$$

and

$$\delta a_{\alpha 8} = \delta Y_\alpha . \quad (5.10i)$$

Therefore, using eqn.(5.4b), it follows that

$$\mathbf{N} \delta \mathbf{Q}' = -\delta \mathbf{N} \mathbf{Q}' = - \begin{bmatrix} \eta_1 \\ \eta_2 \\ \vdots \\ \vdots \end{bmatrix} = - \begin{bmatrix} \delta a_{11} & \dots & \dots \\ \vdots & \ddots & \vdots \\ \vdots & \dots & \delta a_{nn} \end{bmatrix} \mathbf{Q}' . \quad (5.11a)$$

In eqn.(5.11a), if  $\mathbf{N}$  is square and non-singular,

$$\eta_\alpha = \delta a_{\alpha 1} q_1 + \delta a_{\alpha 2} q_2 + \dots + \delta a_{\alpha 8} q_8 \quad \text{and} \quad |\eta_\alpha| \leq E ; \quad (5.11b)$$

$$E = |\delta a_{\alpha 1}| |q_1| + |\delta a_{\alpha 2}| |q_2| + \dots + |\delta a_{\alpha 8}| |q_8| = (|q_1| + |q_2| + \dots + |q_8|) \varepsilon \quad (5.11c)$$

for  $(-\varepsilon \leq \delta a_{\alpha\beta} \leq \varepsilon)$ . Therefore,

$$\delta q_\gamma = \tilde{A}_{\gamma 1} \eta_1 + \tilde{A}_{\gamma 2} \eta_2 + \dots + \tilde{A}_{\gamma 8} \eta_8 \quad (\gamma = 1, 2, \dots, 8) \quad (5.12a)$$

and

$$|\delta q_\gamma| \leq \left[ |\tilde{A}_{\gamma 1}| + |\tilde{A}_{\gamma 2}| + \dots + |\tilde{A}_{\gamma 8}| \right] E ; \quad (5.12b)$$

where  $\tilde{A}_{\gamma\alpha}$  ( $\alpha = 1, 2, \dots, 8$ ) are the elements of the inverse, or the pseudo-inverse (eqns.(3.4a,3.4b)), of  $\mathbf{N}$ .

If  $\mathbf{N}$  is not a square matrix, we define a residual matrix by

$$\delta \mathbf{N}^+ = \tilde{\mathbf{N}}^+ - \mathbf{N}^+ ; \quad (5.12c)$$

where  $\mathbf{N}^+$  and  $\tilde{\mathbf{N}}^+$  are the pseudo-inverses of  $\mathbf{N}$  and  $\tilde{\mathbf{N}}$  respectively. Then it is shown that, as shown by Lawson and Hanson ([30]),

$$\|\delta \mathbf{N}^+\| \leq \|\delta \mathbf{N}^+\| + \|\delta \mathbf{N}^+\| + \|\delta \mathbf{N}^+\| ; \quad (5.12d)$$

where

$$\|\delta \mathbf{N}^+\| \leq \|\delta \mathbf{N}\| \cdot \|\mathbf{N}^+\| \cdot \|\tilde{\mathbf{N}}^+\| , \quad (5.12e)$$

$$\|\delta \mathbf{N}^+\| \leq \|\delta \mathbf{N}\| \cdot \|\tilde{\mathbf{N}}^+\|^2 \quad (5.12f)$$

and



$$\|\delta \mathbf{N}_f\| \leq \|\delta \mathbf{N}\| \cdot \|\mathbf{N}^+\|^2 . \quad (5.12g)$$

### 5.2.2. Error in rotation matrix R due to error in matrix Q

The three methods for the computation of the rotation matrix  $\mathbf{R}$  from  $\mathbf{Q}$  have already been presented in chapter III. Here, we shall treat these methods separately in order to investigate the propagation of error due to error in *essential elements*.

#### 5.2.2.1. Error analysis using the first method

The propagation of error in the rotational elements, when the first method for IPC algorithm is used, is investigated in this section.

If the SVD of  $\tilde{\mathbf{Q}}$  (eqn.(3.5)), redefined as

$$\tilde{\mathbf{Q}} = \mathbf{Q} + \delta \mathbf{Q} = \mathbf{U}' \Delta' \mathbf{V}'^t = (\mathbf{U} + \delta \mathbf{U}) (\Delta + \delta \Delta) (\mathbf{V} + \delta \mathbf{V})^t \quad (5.13a)$$

$$= (\mathbf{u}_1', \mathbf{u}_2', \mathbf{u}_3') \begin{bmatrix} \sigma_{11}' & 0 & 0 \\ 0 & \sigma_{22}' & 0 \\ 0 & 0 & \sigma_{33}' \end{bmatrix} \begin{bmatrix} \mathbf{v}_1'^t \\ \mathbf{v}_2'^t \\ \mathbf{v}_3'^t \end{bmatrix}$$

is an approximation to the true value

$$\mathbf{Q} = \mathbf{U} \Delta \mathbf{V}^t = (\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3) \begin{bmatrix} \sigma_{11} & 0 & 0 \\ 0 & \sigma_{22} & 0 \\ 0 & 0 & \sigma_{33} \end{bmatrix} \begin{bmatrix} \mathbf{v}_1^t \\ \mathbf{v}_2^t \\ \mathbf{v}_3^t \end{bmatrix} \quad (5.13b)$$

where

$$\mathbf{U} = \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ u_{21} & u_{22} & u_{23} \\ u_{31} & u_{32} & u_{33} \end{bmatrix}, \quad \mathbf{V} = \begin{bmatrix} v_{11} & v_{12} & v_{13} \\ v_{21} & v_{22} & v_{23} \\ v_{31} & v_{32} & v_{33} \end{bmatrix}, \quad (5.13c)$$

and  $\delta \mathbf{Q}$ ,  $\delta \mathbf{U}$ ,  $\delta \Delta$ ,  $\delta \mathbf{V}$  being the error matrices, then the  $\alpha$ 'th singular vector  $\mathbf{u}_\alpha'$  of the perturbed matrix  $\tilde{\mathbf{Q}}$  can be approximated by the first order Taylor Series expansion and is given by, as shown by Vaccaro *et al.* ([31]),

$$\mathbf{u}'_{\alpha} = \mathbf{u}_{\alpha} + \frac{\partial \mathbf{u}'_{\alpha}}{\partial q_{\beta}} \Big|_{q_{\beta}'=q_{\beta}} (q_{\beta}' - q_{\beta}), \quad (\alpha = 1, 2, 3), \quad (5.14a)$$

or

$$\delta \mathbf{u}_{\alpha} = \frac{\partial \mathbf{u}'_{\alpha}}{\partial q_{\beta}} \Big|_{q_{\beta}'=q_{\beta}} \delta q_{\beta}, \quad (5.14b)$$

or

$$|\delta \mathbf{u}_{\alpha}| \leq \left| \frac{\partial \mathbf{u}'_{\alpha}}{\partial q_{\beta}} \Big|_{q_{\beta}'=q_{\beta}} \right| |\delta q_{\beta}|; \quad (5.14c)$$

where  $q_{\beta}'$  ( $\beta = 1, 2, \dots, 9$ ) are the entries of the perturbed matrix, and the derivatives of the singular vectors are

$$\frac{\partial \mathbf{u}'_{\alpha}}{\partial q_{\beta}} \Big|_{q_{\beta}'=q_{\beta}} = \frac{1}{\sigma_{\alpha\alpha}} \mathbf{B}_{\beta} \mathbf{v}_{\alpha} - \frac{1}{\sigma_{\alpha\alpha}} (\mathbf{u}_{\alpha}^{\dagger} \mathbf{B}_{\beta} \mathbf{v}_{\alpha}) \mathbf{u}_{\alpha} \quad (5.14d)$$

$$+ \sum_{\substack{\gamma \neq \alpha \\ \gamma=1}}^3 \left[ \frac{\sigma_{\gamma\gamma}}{\sigma_{\alpha\alpha}^2 - \sigma_{\gamma\gamma}^2} (\mathbf{u}_{\alpha}^{\dagger} \mathbf{B}_{\beta} \mathbf{v}_{\gamma}) + \frac{\sigma_{\gamma\gamma}^2}{\sigma_{\alpha\alpha} (\sigma_{\alpha\alpha}^2 - \sigma_{\gamma\gamma}^2)} (\mathbf{u}_{\gamma}^{\dagger} \mathbf{B}_{\beta} \mathbf{v}_{\alpha}) \right] \mathbf{u}_{\gamma}; \quad (5.14e)$$

$$[\mathbf{B}_{\beta}]_{\gamma\lambda} = \begin{cases} 1, & \text{if } \gamma + \lambda - 1 = \beta \\ 0, & \text{otherwise} \end{cases} \quad (5.14f)$$

and  $1 \leq \gamma \leq 3$ ,  $1 \leq \lambda \leq 3$ .

Based on the above results, we can calculate the errors in the right singular vectors from those in the left singular vectors by using the fact that the two matrices  $\mathbf{U}'$  and  $\mathbf{V}'$ , by definition, should be orthogonal. Therefore, we get

$$\delta \mathbf{V} = -\mathbf{U}^* \delta \mathbf{U}^{\dagger} \mathbf{V}; \quad (5.15)$$

where  $\mathbf{U}^* = (\mathbf{U}^{\dagger})^{-1}$ .

We shall now present the analysis of error propagation in rotational elements due to error in  $\mathbf{U}$  and  $\mathbf{V}$ . Since the actual rotation matrix is either ( eqns.(3.6a,3.6b))

$$\mathbf{R} = \mathbf{U} \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & \omega \end{bmatrix} \mathbf{V}^t \quad (5.16a)$$

or

$$\mathbf{R}' = \mathbf{U} \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & \omega \end{bmatrix} \mathbf{V}^t, \quad (5.16b)$$

where  $\omega = \det(\mathbf{U})/\det(\mathbf{V})$ , the actual rotation matrix is either

$$\mathbf{R} + \delta\mathbf{R} = (\mathbf{U} + \delta\mathbf{U}) \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & \omega' \end{bmatrix} (\mathbf{V} + \delta\mathbf{V})^t, \quad (5.16c)$$

or

$$\mathbf{R}' + \delta\mathbf{R}' = (\mathbf{U} + \delta\mathbf{U}) \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & \omega' \end{bmatrix} (\mathbf{V} + \delta\mathbf{V})^t; \quad (5.16d)$$

where  $\omega' = \det(\mathbf{U}')/\det(\mathbf{V}') = \pm \omega$ , and might introduce a sign change so that  $|\omega'| = |\omega|$ .

Therefore,

$$|\delta r_{\alpha\beta}| \leq E_r \quad (5.17a)$$

where

$$E_r = (|v_{\beta 1}| + |u_{\alpha 2}| + |v_{\beta 2}| + |u_{\alpha 1}| + |\omega| |v_{\beta 3}| + |\omega| |u_{\alpha 3}|) \epsilon_1; \quad (5.17b)$$

where  $\epsilon_1 = \epsilon_u = \epsilon_v = \epsilon_\omega$  ( for  $-\epsilon_u \leq \delta u_{\alpha\beta} \leq \epsilon_u$ ;  $-\epsilon_\omega \leq \delta\omega \leq \epsilon_\omega$  and  $-\epsilon_v \leq \delta v_{\alpha\beta} \leq \epsilon_v$  ) has been assumed.

Similarly, for other solution of the rotation matrix, we see that

$$|\delta r_{\alpha\beta}'| \leq E_r. \quad (5.17e)$$

### 5.2.2.2. Error analysis using the second method

The error propagation in the rotational elements, for the second method, is studied in this section.

From the definitions expressed in chapter III, we have

$$\begin{aligned} \mathbf{l}_2 \times \mathbf{l}_3 = & (q_{22} q_{33} - q_{23} q_{32}) \hat{\mathbf{i}} + (q_{23} q_{31} - q_{21} q_{33}) \hat{\mathbf{j}} \\ & + (q_{21} q_{32} - q_{22} q_{31}) \hat{\mathbf{k}} \end{aligned} \quad (5.18a)$$

and

$$\begin{aligned} (\mathbf{l}_2 \times \mathbf{l}_3) \times \mathbf{l}_1 = & [q_{13} (q_{23} q_{31} - q_{21} q_{33}) - q_{12} (q_{21} q_{32} - q_{22} q_{31})] \hat{\mathbf{i}} \\ & + [q_{11} (q_{21} q_{32} - q_{22} q_{31}) - q_{13} (q_{22} q_{33} - q_{23} q_{32})] \hat{\mathbf{j}} \\ & + [q_{12} (q_{22} q_{33} - q_{23} q_{32}) - q_{11} (q_{23} q_{31} - q_{21} q_{33})] \hat{\mathbf{k}} . \end{aligned} \quad (5.18b)$$

where  $\hat{\mathbf{i}}$ ,  $\hat{\mathbf{j}}$ ,  $\hat{\mathbf{k}}$  are the unit vectors along x-, y-, and z-axes respectively. Also,

$$\|\mathbf{T}\|^2 = \sum_{\alpha=1}^3 \sum_{\beta=1}^3 q_{\alpha\beta}^2 = \|\mathbf{Q}\|^2 \text{ and } a = t_x^2, b = t_y^2, c = t_z^2;$$

where  $\|\mathbf{Q}\|^2$  is the *Frobenius*-norm of  $\mathbf{Q}$ . Therefore, for the rows of the rotation matrix  $\mathbf{R}$ , we have

$$\mathbf{p}_1 = r_{11} \hat{\mathbf{i}} + r_{12} \hat{\mathbf{j}} + r_{13} \hat{\mathbf{k}} ; \quad (5.19a)$$

where

$$\begin{aligned} r_{11} = & \frac{[(q_{23} q_{31} - q_{21} q_{33}) q_{13} - (q_{21} q_{32} - q_{22} q_{31}) q_{12} + t_x (q_{22} q_{33} - q_{23} q_{32})]}{t_x \sum_{\alpha=1}^3 \sum_{\beta=1}^3 q_{\alpha\beta}^2} \\ & = f_{r_{11}}(q_{\alpha\beta}) \end{aligned} \quad (5.19b)$$

and so on. Similarly,

$$\mathbf{p}_2 = r_{21} \hat{\mathbf{i}} + r_{22} \hat{\mathbf{j}} + r_{23} \hat{\mathbf{k}} \quad (5.19c)$$

and

$$\mathbf{p}_3 = r_{31} \hat{\mathbf{i}} + r_{32} \hat{\mathbf{j}} + r_{33} \hat{\mathbf{k}} \quad (5.19d)$$

where

$$r_{21} = \frac{[q_{31} + t_y (q_{13} (q_{23} q_{31} - q_{21} q_{33}) - q_{12} (q_{21} q_{32} - q_{22} q_{31}))]}{t_x} \quad (5.19e)$$

$$= f_{r_{21}}(q_{\alpha\beta})$$

and so on, and

$$r_{31} = \frac{[-q_{21} + t_z (q_{13} (q_{23} q_{31} - q_{21} q_{33}) - q_{12} (q_{21} q_{32} - q_{22} q_{31}))]}{t_x} \quad (5.19f)$$

$$= f_{r_{31}}(q_{\alpha\beta})$$

and so on. In general,

$$r_{\alpha\beta} = f_{r_{\alpha\beta}}(q_{\alpha\beta}) \quad \text{for } \alpha, \beta = 1, 2, 3 \quad (5.19g)$$

Therefore, as before

$$|\delta r_{\alpha\beta}| \leq \varepsilon_q \sum_{\alpha=1}^3 \sum_{\beta=1}^3 \left| \frac{\partial f_{r_{\alpha\beta}}}{\partial q_{\alpha\beta}} \right|; \quad -\varepsilon_q \leq \delta q_{\alpha\beta} \leq \varepsilon_q. \quad (5.20)$$

### 5.2.2.3. Error analysis using the third method

In this section, the error propagation, using the third method for the computation of  $\mathbf{R}$  from  $\mathbf{Q}$ , is presented.

By definition,  $\mathbf{H} = \mathbf{Q}^t$  and

$$\mathbf{W}_\alpha = \mathbf{h}_\alpha \times \mathbf{T} = \mathbf{W}_{\alpha 1} \hat{\mathbf{i}} + \mathbf{W}_{\alpha 2} \hat{\mathbf{j}} + \mathbf{W}_{\alpha 3} \hat{\mathbf{k}}; \quad (5.21a)$$

where  $\mathbf{h}_\alpha$  is defined in eqn.(3.20a), and

$$\mathbf{W}_{\alpha 1} = [q_{2\alpha} t_z - q_{3\alpha} t_y]; \quad (5.21b)$$

$$\mathbf{W}_{\alpha 2} = [q_{3\alpha} t_x - q_{1\alpha} t_z] \quad (5.21c)$$

and

$$\mathbf{W}_{\alpha 3} = [q_{1\alpha} t_y - q_{2\alpha} t_x] . \quad (5.21d)$$

Thus, columns of  $\mathbf{R}$  are given by

$$\mathbf{c}_\alpha = \mathbf{W}_\alpha + \mathbf{W}_\beta \times \mathbf{W}_\gamma = r_{1\alpha} \hat{\mathbf{i}} + r_{2\alpha} \hat{\mathbf{j}} + r_{3\alpha} \hat{\mathbf{k}} \quad (\alpha, \beta, \gamma = 1, 2, 3; \alpha \neq \beta \neq \gamma) ; \quad (5.22a)$$

where

$$\begin{aligned} r_{1\alpha} &= ( [q_{3\beta} t_x - q_{1\beta} t_z] [q_{1\gamma} t_y - q_{2\gamma} t_x] - [q_{1\beta} t_y - q_{2\beta} t_x] [q_{3\gamma} t_x - q_{1\gamma} t_z] ) + (q_{2\alpha} t_z - q_{3\alpha} t_y) \\ &= f_{r_{1\alpha}} [q_{\beta\gamma}] \end{aligned} \quad (5.22b)$$

$$\begin{aligned} r_{2\alpha} &= ( [q_{1\beta} t_y - q_{2\beta} t_x] [q_{2\gamma} t_z - q_{3\gamma} t_y] - [q_{2\beta} t_z - q_{3\beta} t_y] [q_{1\gamma} t_y - q_{2\gamma} t_x] ) + (q_{3\alpha} t_x - q_{1\alpha} t_z) \\ &= f_{r_{2\alpha}} [q_{\beta\gamma}] \end{aligned} \quad (5.22c)$$

$$\begin{aligned} r_{3\alpha} &= ( [q_{2\beta} t_z - q_{3\beta} t_y] [q_{3\gamma} t_x - q_{1\gamma} t_z] - [q_{3\beta} t_x - q_{1\beta} t_z] [q_{2\gamma} t_z - q_{3\gamma} t_y] ) + (q_{1\alpha} t_y - q_{2\alpha} t_x) \\ &= f_{r_{3\alpha}} [q_{\beta\gamma}] \end{aligned} \quad (5.22d)$$

In the above discussion, the combination  $\alpha = 1, \beta = 2, \gamma = 3$  gives us first column, and so on. Hence, the bounds are defined as

$$|\delta r_{1\alpha}| \leq E_{r_{1\alpha}} ; |\delta r_{2\alpha}| \leq E_{r_{2\alpha}} ; |\delta r_{3\alpha}| \leq E_{r_{3\alpha}} ; \quad (5.23a)$$

where

$$E_{r_{1\alpha}} = \sum_{\beta=1}^3 \sum_{\gamma=1}^3 \left| \frac{\partial f_{r_{1\alpha}}}{\partial q_{\beta\gamma}} \right| |\delta q_{\beta\gamma}| = \epsilon_q \sum_{\beta=1}^3 \sum_{\gamma=1}^3 \left| \frac{\partial f_{r_{1\alpha}}}{\partial q_{\beta\gamma}} \right| \quad (5.23b)$$

$$E_{r_{2\alpha}} = \epsilon_q \sum_{\beta=1}^3 \sum_{\gamma=1}^3 \left| \frac{\partial f_{r_{2\alpha}}}{\partial q_{\beta\gamma}} \right| \quad (5.23c)$$

and

$$E_{r_{3\alpha}} = \epsilon_q \sum_{\beta=1}^3 \sum_{\gamma=1}^3 \left| \frac{\partial f_{r_{3\alpha}}}{\partial q_{\beta\gamma}} \right| \quad (5.23d)$$

where  $-\varepsilon_q \leq \delta q_{\beta\gamma} \leq \varepsilon_q$ .

### 5.2.3. Error in the motion parameters due to error in rotation matrix R

In this section, the errors in the motion parameters are studied separately for the two representations of the rotation matrix R ( Appendix A ).

#### 5.2.3.1. Using the first representation of R

From the definitions of the directional cosines of an arbitrary axis, and the angle of rotation around this axis, we see that

$$v_1 = f_{v_1}(r_2, r_3, r_4, r_6, r_7, r_8) = \pm (r_8 - r_6)/d ; \quad (5.24a)$$

$$v_2 = f_{v_2}(r_2, r_3, r_4, r_6, r_7, r_8) = \pm (r_3 - r_7)/d ; \quad (5.24b)$$

$$v_3 = f_{v_3}(v_1, v_2) = (1 - (v_1^2 + v_2^2))^{1/2} \quad (5.24c)$$

and

$$\theta = \sin^{-1}(\pm d/2) ; \quad (5.24d)$$

where

$$d^2 = (r_8 - r_6)^2 + (r_3 - r_7)^2 + (r_4 - r_2)^2 . \quad (5.24e)$$

As before,

$$|\delta v_\alpha| \leq \sum_{\beta=2}^8 \left| \frac{\partial f_{v_\alpha}}{\partial r_\beta} \right| |\delta r_\beta| = \varepsilon_r \sum_{\beta=2}^8 \left| \frac{\partial f_{v_\alpha}}{\partial r_\beta} \right| ; \quad (5.25)$$

where  $-\varepsilon_r \leq r_\beta \leq \varepsilon_r$ , and  $\alpha = 1,2,3$  and  $\beta = 1,2,\dots,9$ . For  $v_1$ , we have

$$\frac{\partial f_{v_1}}{\partial r_2} = \pm \frac{(r_8 - r_6)(r_4 - r_2)}{d^3} ; \quad (5.26a)$$

$$\frac{\partial f_{v_1}}{\partial r_3} = \mp \frac{(r_8 - r_6)(r_3 - r_7)}{d^3} ; \quad (5.26b)$$

$$\frac{\partial f_{v_1}}{\partial r_4} = \mp \frac{(r_8 - r_6)(r_4 - r_2)}{d^3}; \quad (5.26c)$$

$$\frac{\partial f_{v_1}}{\partial r_6} = \mp \frac{(r_3 - r_7)^2 + (r_4 - r_2)^2}{d^3}; \quad (5.26d)$$

$$\frac{\partial f_{v_1}}{\partial r_7} = \pm \frac{(r_8 - r_6)(r_3 - r_7)}{d^3} \quad (5.26e)$$

and

$$\frac{\partial f_{v_1}}{\partial r_8} = \pm \frac{(r_3 - r_7)^2 + (r_4 - r_2)^2}{d^3}. \quad (5.26f)$$

Similarly for  $v_2$ , we have

$$\frac{\partial f_{v_2}}{\partial r_2} = \pm \frac{(r_3 - r_7)(r_4 - r_2)}{d^3}; \quad (5.27a)$$

$$\frac{\partial f_{v_2}}{\partial r_3} = \pm \frac{(r_3 - r_7)^2 + (r_4 - r_2)^2}{d^3}; \quad (5.27b)$$

$$\frac{\partial f_{v_2}}{\partial r_4} = \mp \frac{(r_3 - r_7)(r_4 - r_2)}{d^3}; \quad (5.27c)$$

$$\frac{\partial f_{v_2}}{\partial r_6} = \pm \frac{(r_3 - r_7)(r_8 - r_6)}{d^3}; \quad (5.27d)$$

$$\frac{\partial f_{v_2}}{\partial r_7} = \mp \frac{(r_3 - r_7)^2 + (r_8 - r_6)^2}{d^3} \quad (5.27e)$$

and

$$\frac{\partial f_{v_2}}{\partial r_8} = \mp \frac{(r_3 - r_7)(r_8 - r_6)}{d^3}. \quad (5.27f)$$

Also for  $v_3$ , we have

$$|\delta v_3| \leq \left| \frac{\partial f_{v_3}}{\partial v_1} \right| |\delta v_1| + \left| \frac{\partial f_{v_3}}{\partial v_2} \right| |\delta v_2|; \quad (5.28a)$$



where

$$\frac{\partial f_{v_3}}{\partial v_1} = -\frac{v_1}{[1 - (v_1^2 + v_2^2)]^{1/2}} \quad \text{and} \quad \frac{\partial f_{v_3}}{\partial v_2} = -\frac{v_2}{[1 - (v_1^2 + v_2^2)]^{1/2}}, \quad (5.28b)$$

For  $\theta$ , we have

$$|\delta\theta| \leq \sum_{\beta=2}^8 \left| \frac{\partial f_{\theta}}{\partial r_{\beta}} \right| |\delta r_{\beta}| \quad (\alpha = 1, 2, 3); \quad (5.29a)$$

where

$$\frac{\partial f_{\theta}}{\partial r_2} = \mp \frac{(r_4 - r_2)}{d'}; \quad (5.29b)$$

$$\frac{\partial f_{\theta}}{\partial r_3} = \pm \frac{(r_3 - r_7)}{d'}; \quad (5.29c)$$

$$\frac{\partial f_{\theta}}{\partial r_4} = \pm \frac{(r_4 - r_2)}{d'}; \quad (5.29d)$$

$$\frac{\partial f_{\theta}}{\partial r_6} = \mp \frac{(r_8 - r_6)}{d'}; \quad (5.29e)$$

$$\frac{\partial f_{\theta}}{\partial r_7} = \mp \frac{(r_3 - r_7)}{d'}; \quad (5.29f)$$

$$\frac{\partial f_{\theta}}{\partial r_8} = \pm \frac{(r_8 - r_6)}{d'}; \quad (5.29g)$$

$$d' = d (1 - d^2/4)^{1/2}. \quad (5.29h)$$

Therefore,

$$|\delta\theta| \leq E_{\theta}; \quad (5.30a)$$

where

$$E_{\theta} = \frac{2 \varepsilon_r}{|d'|} \sum_{\beta=2}^8 |r_{\beta}|. \quad (5.30b)$$

### 5.2.3.2. Using the second representation of R

The second representation of the rotation matrix  $\mathbf{R}$  is used in this case. By definitions, we know that

$$\text{Roll} = \theta = f_{\theta}(r_6) = \sin^{-1} r_6 ; \quad (5.31a)$$

$$\text{Yaw} = \phi = f_{\phi}(r_3, r_6) = \sin^{-1} \left[ -\frac{r_3}{(1-r_6^2)^{1/2}} \right] \quad (5.31b)$$

and

$$\text{Pitch} = \psi = f_{\psi}(r_4, r_6) = \sin^{-1} \left[ -\frac{r_4}{(1-r_6^2)^{1/2}} \right]. \quad (5.31c)$$

Therefore,

$$|\delta\theta| \leq |f'_{\theta}(r_6)| |\delta r_6| ; \quad (5.32a)$$

where

$$f'_{\theta}(r_6) = \frac{1}{(1-r_6^2)^{1/2}} \quad (5.32b)$$

Similarly,

$$|\delta\phi| \leq \left| \frac{\partial f_{\phi}}{\partial r_3} \right| |\delta r_3| + \left| \frac{\partial f_{\phi}}{\partial r_6} \right| |\delta r_6| . \quad (5.33a)$$

and

$$|\delta\psi| \leq \left| \frac{\partial f_{\psi}}{\partial r_4} \right| |\delta r_4| + \left| \frac{\partial f_{\psi}}{\partial r_6} \right| |\delta r_6| . \quad (5.33b)$$

In eqns.(5.35a,5.35b), using the properties of rotational elements from Appendix A, we have

$$\frac{\partial f_{\phi}}{\partial r_3} = -\frac{1}{r_9} ; \quad (5.34a)$$

$$\frac{\partial f_{\phi}}{\partial r_6} = -\frac{r_3 r_6}{r_9 (1 - r_6^2)} ; \quad (5.34b)$$

$$\frac{\partial f_{\psi}}{\partial r_4} = -\frac{1}{r_5} \quad (5.34c)$$

and

$$\frac{\partial f_{\psi}}{\partial r_6} = -\frac{r_4 r_6}{r_5 (1 - r_6^2)} . \quad (5.34d)$$

## CHAPTER VI

### EXPERIMENTAL RESULTS

In this chapter, we present some experimental results for the IPC and the GIPC algorithms. The algorithms have been tested successfully on simulated and real data. A brief description for these experiments will be given in the following two sections. In the third section, the various error plots will be discussed. These plots indicate the relationship between the errors in the input set of data ( co-ordinates of the features ) and the errors in the output data ( motion parameters ).

For simulation purposes, the data, kept in a file called *test\_coord*, appear in the form of a set of 3-D co-ordinates of feature points (Fig.8). This set of points is rotated, and then translated, with some reference parameters to another set of points. These two different sets of points act as the inputs to the IPC and the GIPC algorithms. A brief description of the implementations of these algorithms in the form of computer programs is presented in Appendix G. For experimental results with real data, the data is kept in a separate file called *nasa\_data* (Figs. 13,14), which appear in the form of either 2-D or 3D co-ordinates of the features of the object, corresponding to before and after the motion of the object whose parameters are to be determined. The last two rows in this file indicate the motion ( in terms of roll, yaw, and pitch in degrees ) of camera frame  $F_i$  to another frame  $F_j$ , and the orientation of  $F_i$  with respect to the standard frame  $S$ .

The details of the experiments with both sets of data follow :

#### 6.1. Using Simulated Data

The first three experiments, as indicated in Fig.9 through Fig.11 , are run with simulated data. All these experiments use the first method of the IPC algorithm.

### *Experiment 1 :*

In the first experiment (Fig.9), the first representation of the rotation matrix  $\mathbf{R}$  is used. In this case, the reference motion parameters are :

$v_1 = 0.1$ ,  $v_2 = 0.2$ ,  $\theta = 12^\circ$ ,  $t_x = 1$ ,  $t_y = 1$ ,  $t_z = 1$ , and the number of points used for the analysis equals 8. As described in chapter III, we get two different solutions for the rotation matrix, and hence for the parameters. The test for choosing the correct solution of the rotation matrix has been applied, and the surface shape has been computed. The translational vector is estimated up to a scale factor. The same procedure applies to other two experiments also.

### *Experiment 2 :*

In the second experiment (Fig.10), the second representation of the rotation matrix has been used. In this case, the reference motion parameters are

Roll =  $11^\circ$ , Yaw =  $12^\circ$ , Pitch =  $13^\circ$ ,  $t_x = 1$ ,  $t_y = 2$ ,  $t_z = 3$ , and the number of points equals 8.

### *Experiment 3 :*

This experiment (Fig.11) is somewhat similar to the first experiment, but the number of points used in this case is 16.

## **6.2. Using Real Data**

The experiments using the real data are shown in Fig.12 through Fig.20. Separate experiments, corresponding to 2-D images of the two positions of an *octbox*, were conducted. The octbox has two parallel octagonal faces opposite to each other, in addition to eight rectangular faces. In the first case (Fig.12b), the camera taking the picture was kept at the same height as the object in such a way that the center of geometry of the object coincided with the origin of the image plane of the camera. The octbox was rotated around the x-axis by  $15^\circ$ . In the second case (Fig.12c), the octbox was rotated

around the x-axis by  $105^\circ$ . In both cases, rotation around the x-axis means that the angle of rotation, by definition, is *roll*, or equivalently, the directional cosines of the arbitrary axis around which the octbox rotates, are given by

$v_1 = 1.0$ ;  $v_2 = 0.0$ ;  $v_3 = 0.0$  ( Appendix A ). The translation along the three axes is 1 unit each. The data for these two cases of octbox rotations are shown in Fig.13 and Fig.14 respectively, where 3-D co-ordinates of the vertices of the octbox before the motion and their 2-D co-ordinates after the motion are given.

#### *Experiment 4 :*

This experiment (Fig.15) shows how the IPC algorithm computes the parameters for the first case of octbox rotation, using real data. As expected, the angle of rotation found is  $-15^\circ$  ( there is an ambiguity in the sign of motion parameters, which is explained in Appendix A ).

#### *Experiment 5 :*

The fifth experiment (Fig.16), similar to the fourth experiment, computes the parameters for the data using the second case of the octbox rotation. In the fourth and the fifth experiments, the first representation of the rotation matrix has been used.

#### *Experiment 6,7 :*

In the sixth and the seventh experiments (Fig.17,18), the second representation of the rotation matrix is used for the same set of data. In the sixth experiment, the angles come out to be right. But in the seventh experiment, addition or subtraction of angles by  $180^\circ$  takes place, because  $\sin \theta = + \sin ( 180 \pm \theta )$ , and  $\cos \theta = - \cos ( 180 \pm \theta )$ .

#### *Experiment 8,9 :*

The GIPC algorithm has been applied to the fourth and fifth experiments, and the results are shown in Fig.19 and Fig.20. In both the experiments, the camera was rotated through  $10^\circ$  around its x-axis. With the same set data for these experiments for

the octbox rotation (Figs.13,14), the directional cosines are found to be same. The angles of rotation are  $-5^\circ$  and  $95^\circ$  respectively, which means that the angles of rotation of the octbox are subtracted by the amount of rotation by the camera, and that indeed should be the case. These two experiments show the success of GIPC algorithm with real data.

### 6.3. Experimental Error Estimates

Because of the difficulties in determining the constants in the various inequalities appearing in the error bounds of the preceding chapter, we obtained error estimates experimentally. This was done by perturbing the data and measuring the corresponding errors in the parameters calculated by our algorithms. These errors are sample errors rather than averaged ones.

The error plots for previous experiments are shown in Fig.21 through Fig.24. In these plots, the x-axis represents the magnitude of the percent relative errors in the input set of co-ordinates, and the y-axis indicates the corresponding magnitude of percent relative errors in the motion parameters. A brief discussion of these plots follows :

*Plots 1,2 ( Figs.21,22 ) :*

These are the plots drawn for the first experiment (fig.9). Here, the first representation of  $\mathbf{R}$  is used. In the plots of Fig.21,  $a_1$ ,  $b_1$ ,  $th_1$  represent the error curves for the motion parameters ( directional cosines, and the angle of rotation ) found by using the first method of the IPC algorithm. Similarly,  $a_{II}$ ,  $b_{II}$ ,  $th_{II}$  represent the error curves for the motion parameters found by using the second method. Another plot is drawn in Fig.22 in a similar manner, where the error curves for the motion parameters found by using third method (  $a_{III}$ ,  $b_{III}$ ,  $th_{III}$  ) are compared with those found by the second method. As can be observed from these two plots, the second method gives better results than the first method.

*Plots 3,4 ( Figs.23,24 ) :*

These plots are similarly drawn for the second experiment ( Fig.9 ), but the second representation of  $\mathbf{R}$  is used. In these plots,  $r_I$ ,  $y_I$ ,  $p_I$ , and  $r_{II}$ ,  $y_{II}$ ,  $p_{II}$ , and  $r_{III}$ ,  $y_{III}$ ,  $p_{III}$  represent the error curves for the motion parameters ( roll, yaw, pitch ) using the first, second, and third methods respectively.

From these four plots, and various others, we have found that the first method of the IPC algorithm gives poorer results than the other two methods. A telescopic increase in the output errors is due to the reason that the IPC algorithm assumes rigid-body motion of an object, and an error in the input set of co-ordinates amounts to the fact that the object has deformed.



x	y	z
4	4	8
12	1.2	4.1
13	1.1	2.3
14	1	12
8	0.12	0.2
9	12	11
12	0.23	21
400	0.4	40
53	21.1	3.3
12	22.9	3
12	2.5	-41
10	20	100
111	1121	6
11	132	3
111	121	1
12	910	-1
176	829	-3
12	324	-6
12	23	10
100	1000	100

Fig.8 : 3-D data kept in *test\_coord* for simulation purposes.

**THE DEMONSTRATION OF THE IPC ALGORITHM  
USING THE FIRST METHOD**

The estimated Translational vector (up to a scale factor) is

16.599274    16.599274    16.599274

Estimated Rotation Matrix is either R given by

-0.217034    0.736785    0.640347  
0.558248    -0.444447    0.700590  
0.800784    0.509524    -0.314849

or R' given by

0.978366    -0.202210    0.043712  
0.203084    0.979022    -0.016531  
-0.039452    0.025051    0.998907

The directional cosines of the axis and the angle of rotation about the axis, ( corresponding to R and R' )

are respectively

0.622784    0.522951    0.581947  
188.823823  
0.100000    0.200000    0.974679  
12.000000

**Conclusion :** Choose R' and its associated parameters as the final solution. The estimated coordinates before and after motion are

x	y	z	x'	y'	z'
66.397096	66.397096	132.794192	73.938506	92.892411	148.292158
199.191287	19.919130	68.057022	210.428356	75.428082	77.222349
215.790561	18.259202	38.178329	225.698113	77.667964	46.679838
232.389835	16.599274	199.191287	249.312173	76.752148	206.820405
132.794192	1.991913	3.319855	146.262946	45.462933	14.726347
33.198548	199.191287	182.592013	16.782605	215.335477	202.671958
663.970958	3.817833	348.584753	680.671384	149.416513	338.703529
6639.709575	6.639710	663.970958	6540.347246	1360.544109	418.058419

Fig.9 : Details of Experiment 1.

**THE DEMONSTRATION OF THE IPC ALGORITHM  
USING THE FIRST METHOD**

The estimated Translational vector (up to a scale factor) is

1.669470 3.338940 5.008410

Estimated Rotation Matrix is either  $R$  given by

-0.767551 -0.006352 0.640957  
0.574024 -0.451791 0.682921  
0.285241 0.892101 0.350419

or  $R'$  given by

0.944154 0.258690 -0.204092  
-0.220818 0.956468 0.190809  
0.244568 -0.135086 0.960176

The directional cosines of the axis and the angle of rotation about the axis, ( corresponding to  $R$  and  $R'$  )  
are respectively

43.072326 298.665956 -128.204875  
11.000000 12.000000 13.000000

**Conclusion :** Choose  $R'$  and its associated parameters as the final solution. The estimated coordinates before and after motion are

x	y	z	x'	y'	z'
6.677880	6.677880	13.355759	6.976114	10.799921	18.563399
20.033639	2.003364	6.844826	19.705580	2.137359	16.209603
21.703109	1.836417	3.839781	21.851933	1.035640	13.755080
23.372579	1.669470	20.033639	20.079949	3.597245	29.734889
13.355759	0.200336	0.333894	14.263038	0.645072	6.568331
3.338940	20.033639	18.364169	6.256468	25.267227	20.751584
66.778796	0.383978	35.058868	57.663016	-4.350215	54.951166
667.787960	0.667788	66.778796	618.707633	-130.739996	232.356948

Fig.10 : Details of Experiment 2.

**THE DEMONSTRATION OF THE IPC ALGORITHM  
USING THE FIRST METHOD**

The estimated Translational vector (up to a scale factor) is

16.599260 16.599259 16.599259

Estimated Rotation Matrix is either R given by

0.978366 -0.202210 0.043712  
0.203084 0.979022 -0.016531  
-0.039452 0.025051 0.998907

or R' given by

-0.217034 0.736785 0.640347  
0.558248 -0.444447 0.700590  
0.800784 0.509524 -0.314849

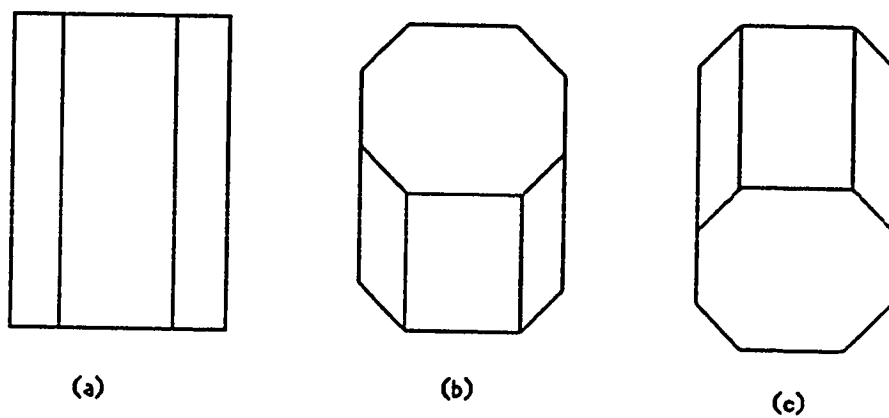
The directional cosines of the axis and the angle of rotation about the axis, ( corresponding to R and R' ) are respectively

0.100000 0.200000 0.974679  
11.999999  
0.622784 0.522951 0.581947  
188.823824

**Conclusion :** Choose R and its associated parameters as the final solution.  
The estimated coordinates before and after motion are

x	y	z	x'	y'	z'
66.397060	66.397060	132.794120	73.938464	92.892358	148.292074
199.191071	19.919108	68.056948	210.428132	75.428002	77.222267
215.790339	18.259183	38.178290	225.697885	77.667885	46.679791
232.389208	16.599229	199.190750	249.311534	76.751952	206.819876
132.794063	1.991911	3.319852	146.262807	45.462890	14.726333
33.198531	199.191185	182.591919	16.782597	215.335364	202.671852
663.969597	3.817825	348.584038	680.670016	149.416213	338.702848
6639.676700	6.639677	663.967670	6540.314956	1360.537392	418.056355
879.760729	350.244372	54.777554	808.898828	537.256147	45.382231
199.191197	380.123194	49.797799	136.793172	426.377501	68.006540
199.191037	41.498133	-680.569376	173.340452	108.930128	-670.045514
165.992907	331.985813	1659.929066	184.429435	347.890280	1676.482407
1842.562200	18608.218256	99.597957	-1939.117916	18606.998055	509.549479
182.592356	2191.108274	49.797915	-245.646196	2198.000353	114.028526
1842.521231	2008.514135	16.599290	1413.843197	2356.890936	10.803764
199.195253	15105.640012	-16.599604	-2843.754732	14846.077343	370.570590

Fig.11 : Details of Experiment 3.



**Fig.12 : Experiments with real data.**

**(a) octbox in its initial position.**

**(b) the first case where the octbox is rotated through 15 deg. around x-axis.**

**(c) the second case where the octbox is rotated through 105 deg. around x-axis.**

x	y	z	X'	Y'
0	-1	-1	3.414214	7.595754
-1.5	-1	-0.5	-3.058409	10.654163
-2	1	1	-0.585786	-0.131652
-1.5	1	-0.5	-0.238625	0.584223
-1.5	1	2.5	-0.379110	-1.268983
0	1	-1	0.449490	0.767327
1.5	1	-0.5	1.193126	0.584223
2	1	1	1.757359	-0.131652
0	0	0		
0	0	0		

Fig.13 : Real data kept in *nasa\_data* for the first case of octball rotation.

x	y	z	X'	Y'
0	-1	-1	-4.44949	-1.303225
-1.5	-1	-0.5	-1.936348	0.633123
-2	1	1	-0.449490	0.767327
-1.5	1	-0.5	-0.644449	2.700675
-1.5	1	2.5	-0.136105	0.359012
0	1	-1	3.414214	7.595754
1.5	1	-0.5	3.222247	2.700675
2	1	1	1.348469	0.767327
0	0	0		
0	0	0		

Fig.14 : Real data kept in *nasa\_data* for the second case of octball rotation.

**THE DEMONSTRATION OF THE IPC ALGORITHM  
USING THE FIRST METHOD**

The estimated Translational vector (up to a scale factor) is

3.863706 3.863707 3.863717

Estimated Rotation Matrix is either  $R$  given by

-0.333334 0.816496 0.471405  
0.666666 -0.149429 0.730224  
0.666667 0.557678 -0.494521

or  $R'$  given by

1.000000 0.000000 -0.000000  
-0.000000 0.965926 -0.258819  
0.000000 0.258819 0.965926

The directional cosines of the axis and the angle of rotation about the axis, ( corresponding to  $R$  and  $R'$  )  
are respectively

0.574043 0.649617 0.498469  
188.643723  
1.000000 0.000002 0.000000  
344.999983

**Conclusion :** Choose  $R'$  and its associated parameters as the final solution.

Fig.15 : Details of Experiment 4.



**THE DEMONSTRATION OF THE IPC ALGORITHM  
USING THE FIRST METHOD**

The estimated Translational vector (up to a scale factor) is

1.035277    1.035273    1.035283

Estimated Rotation Matrix is either R given by

1.000000    -0.000002    0.000003  
0.000002    -0.258819    -0.965926  
0.000003    0.965926    -0.258819

or R' given by

-0.333332    0.471405    -0.816496  
0.666667    0.730223    0.149431  
0.666667    -0.494521    -0.557677

The directional cosines of the axis and the angle of rotation about the axis, ( corresponding to R and R' )  
are respectively

1.000000    -0.000000    0.000002  
105.000000  
0.395384    0.910658    -0.119890  
234.521795

**Conclusion :** Choose R and its associated parameters as the final solution.

Fig.16 : Details of Experiment 5.

**THE DEMONSTRATION OF THE IPC ALGORITHM  
USING THE FIRST METHOD**

The estimated Translational vector (up to a scale factor) is

3.863706   3.863706   3.863707

Estimated Rotation Matrix is either R given by

-0.333333   0.816496   0.471405  
0.666667   -0.149429   0.730224  
0.666667   0.557678   -0.494521

or R' given by

1.000000   0.000000   -0.000000  
-0.000000   0.965926   -0.258819  
0.000000   0.258819   0.965926

The directional cosines of the axis and the angle of rotation about the axis, ( corresponding to R and R' )  
are respectively

46.905138   223.629053   -102.633685  
-14.999999   0.000008   0.000004

**Conclusion :** Choose R' and its associated parameters as the final solution.

Fig.17 : Details of Experiment 6.

**THE DEMONSTRATION OF THE IPC ALGORITHM  
USING THE FIRST METHOD**

The estimated Translational vector (up to a scale factor) is

1.035277    1.035273    1.035281

Estimated Rotation Matrix is either R given by

1.000000    -0.000002    0.000003  
0.000002    -0.258819    -0.965926  
0.000003    0.965926    -0.258819

or R' given by

-0.333332    0.471405    -0.816497  
0.666667    0.730223    0.149431  
0.666667    -0.494522    -0.557677

The directional cosines of the axis and the angle of rotation about the axis, ( corresponding to R and R' )  
are respectively

-75.000021    180.000604    -179.999529  
8.593925    124.333547    -42.394949

**Conclusion :** Choose R and its associated parameters as the final solution.

Fig.18 : Details of Experiment 7.

**THE DEMONSTRATION OF THE GIPC ALGORITHM  
USING THE FIRST METHOD**

The estimated Translational vector (up to a scale factor) is

3.863706    3.863707    3.863717

Estimated Rotation Matrix is either  $R$  given by

-0.333334    0.816496    0.471405  
0.772304    -0.050319    0.633257  
0.540773    0.575154    -0.613810

or  $R'$  given by

1.000000    0.000000    -0.000000  
-0.000000    0.996195    -0.087156  
0.000000    0.087156    0.996195

The directional cosines of the axis and the angle of rotation about the axis, ( corresponding to  $R$  and  $R'$  )  
are respectively

0.576984    0.688845    0.438843  
182.886128  
1.000000    0.000006    0.000001  
354.999983

**Conclusion :** Choose  $R'$  and its associated parameters as the final solution.

Fig.19 : Details of Experiment 8.

**THE DEMONSTRATION OF THE GIPC ALGORITHM  
USING THE FIRST METHOD**

The estimated Translational vector (up to a scale factor) is

1.035277    1.035273    1.035283

Estimated Rotation Matrix is either  $R$  given by

1.000000   -0.000002   0.000003   0.000003   -0.087156   -0.996195   0.000002   0.996195   -0.087156

or  $R'$  given by

-0.333332   0.471405   -0.816496  
0.772304   0.633257   0.050321  
0.540773   -0.613810   -0.575153

The directional cosines of the axis and the angle of rotation about the axis, ( corresponding to  $R$  and  $R'$  )  
are respectively

1.000000    0.000000    0.000002  
95.000000  
0.431055    0.880937    -0.195299  
230.385837

**Conclusion :** Choose  $R$  and its associated parameters as the final solution.

Fig.20 : Details of Experiment 9.

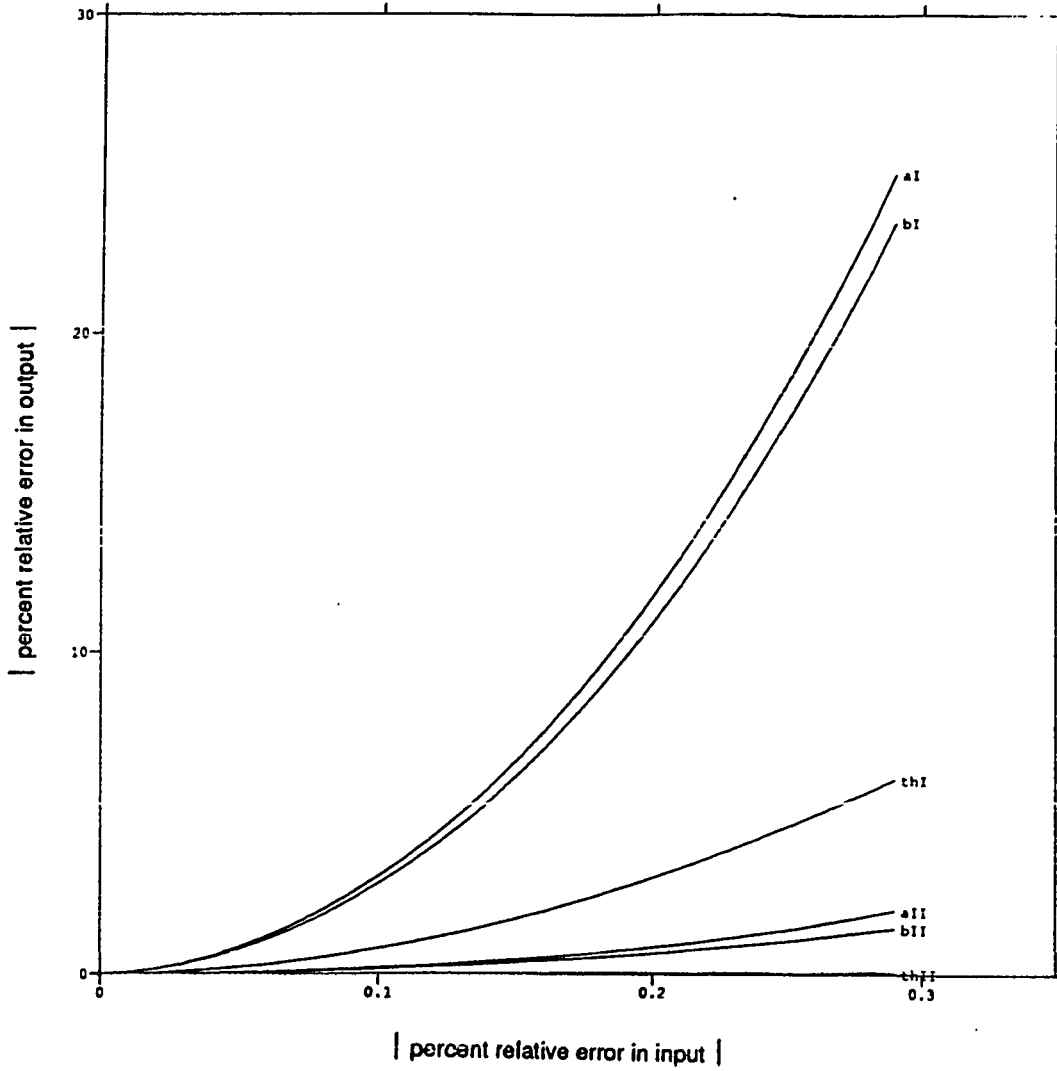


Fig.21 : Error Plots for Experiment 1 using First and Second Methods.

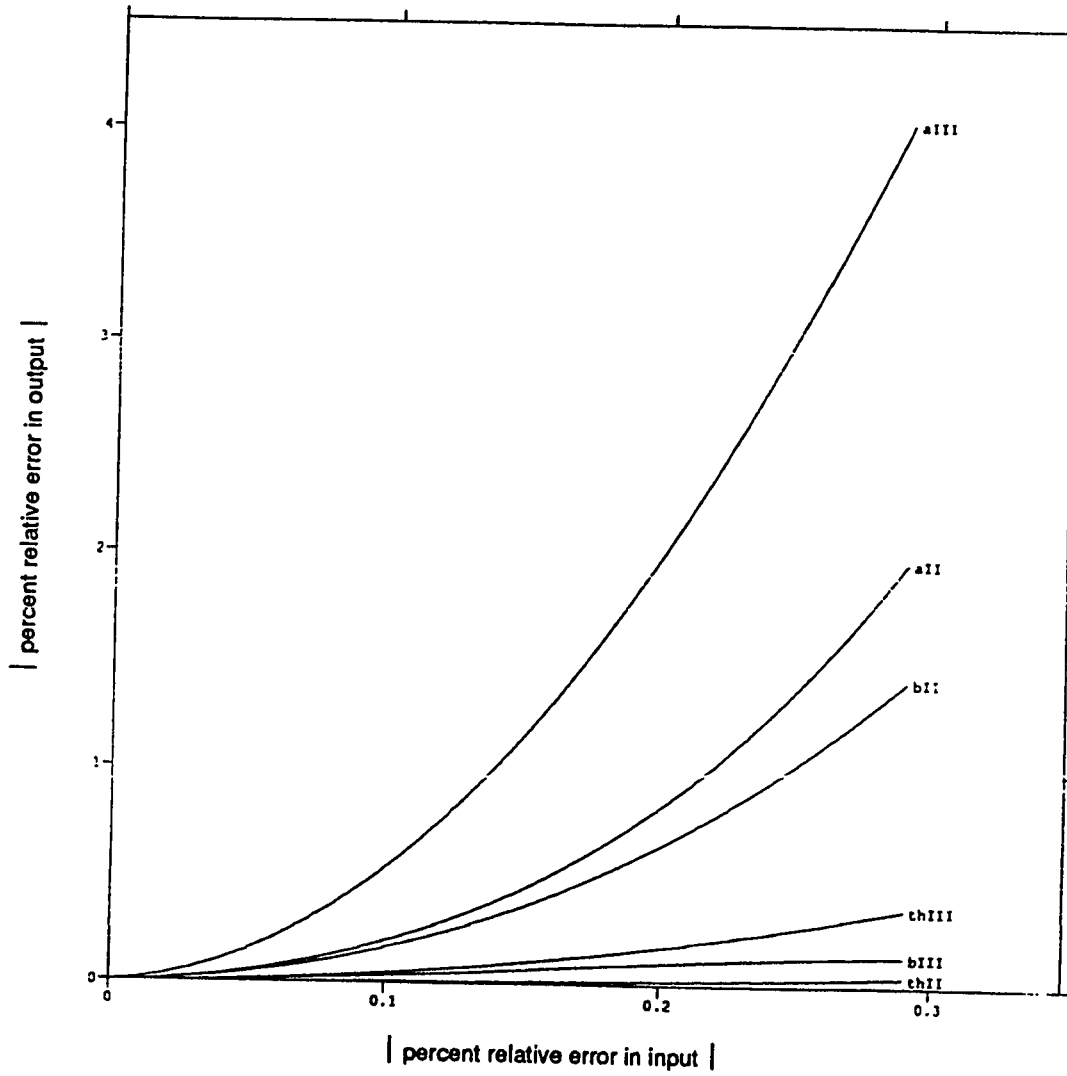


Fig.22 : Error Plots for Experiment 1 using Second and Third Methods.

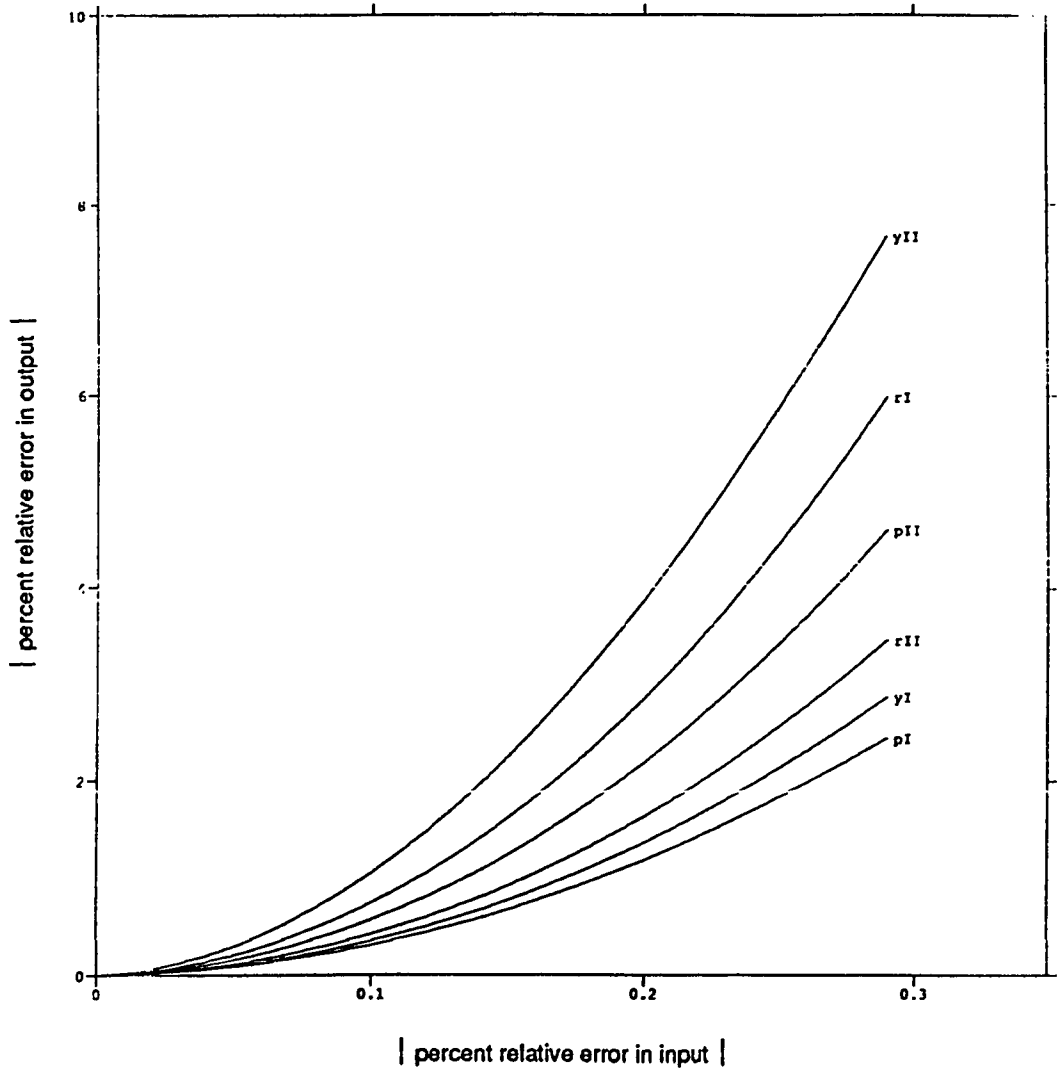


Fig.23 : Error Plots for Experiment 2 using First and Second Methods.



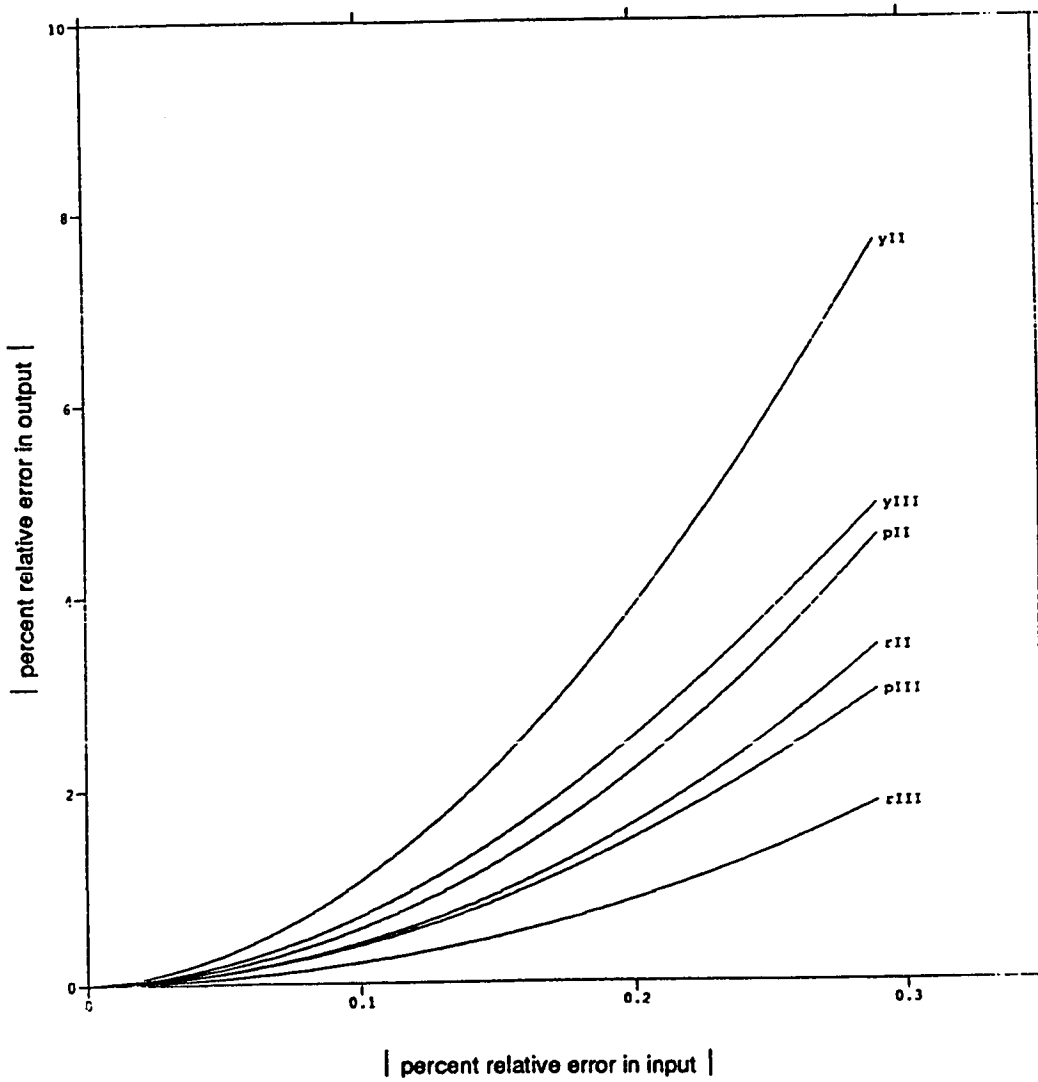


Fig.24 : Error Plots for Experiment 2 using Second and Third Methods.

## **CHAPTER VII**

### **CONCLUSIONS**

#### **Summary**

In this research, we worked on the following problem :

Given an image-sequence of a moving object, obtain its 3-D motion parameters. These parameters give us information regarding the attitude, attitude rate, surface shape, and track of the object. In addition to this, if there are multiple objects in the scene moving with different velocities, we can segment the scene so that this information is used for the pattern analysis of the scene later. From the knowledge of the distribution of the surface points, the objects in the scene can be matched to some models stored in the computer so as to complete the recognition part of the problem.

We used the Image Point Correspondence ( IPC ) algorithm to solve this problem. This algorithm assumes the correspondence of the 2-D image co-ordinates of the features of the object to be known over consecutive frames. The problem of feature extraction from an image-sequence has not been dealt with in this research. Three different methods were employed for this purpose, and implemented in the form of computer programs, in an effort to perform a thorough survey of all the existing techniques based on image point correspondences, because the IPC algorithm applies to the objects with curved surfaces, with a few exceptions. The perspective projections of the object have been considered. Simple linear algebra techniques, instead of iterative procedures, are employed. The application of this algorithm to three types of motion analysis ( monocular vision, stereo vision, and stereo motion ) was presented. Their equivalence in terms of motion-analysis equation was also shown.

In this research, the new developments are :

(i) The extension in the IPC algorithm, which we called the Generalized Image Point Correspondence ( GIPC ) algorithm. A generalized expression for motion-analysis equation was derived, and the other three cases of motion-analysis were found to be special cases of this case. The applications of such a situation of motion analysis are explained in chapter I.

(ii) The study of error flow through the three methods for the IPC algorithm. Expressions for error bounds were derived for various stages of the algorithm. Experimentally, the perturbed set of co-ordinates were fed as the input to the IPC/GIPC algorithms, and the errors in the output set of motion parameters were examined in terms of various plots of percentage of modulus of relative errors in input set of co-ordinates *versus* percentage of modulus of relative errors in the motion parameters.

### **Future Work**

The research done could be extended by developing the following algorithms for future work :

(i) The first algorithm should enable one to extract features of an object from its image-sequence at a sufficiently fast speed, with less computer time.

(ii) A second algorithm is needed to match these features over consecutive frames, using some information about the object and its surface.

(iii) A third algorithm should be developed which would enable one to track these features over several frames.

(iv) A fourth algorithm should use less than eight points as the input to the IPC algorithm by finding some sort of relationship between the surface points, or by using *a priori* information about the shape of the surface of the object.

## APPENDIX A

### THE ROTATION MATRIX

In this appendix, the description and the properties of the rotation matrix  $\mathbf{R}$ , and the computation of the rotation parameters from it, as shown by Ganapathy ([27]) and Huang *et al.* ([5]), has been presented.

Let

$$\mathbf{R} = \begin{bmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{c}_1 & \mathbf{c}_2 & \mathbf{c}_3 \end{bmatrix} = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix} \quad (\text{A.1})$$

where  $\mathbf{p}_\alpha$  ( $\alpha = 1,2,3$ ) is the  $\alpha$ 'th row of  $\mathbf{R}$ ,  $\mathbf{c}_\alpha$  ( $\alpha = 1,2,3$ ) its  $\alpha$ 'th column, and  $r_1, r_2, \dots, r_9$  are its elements.

#### A.1. THE PROPERTIES OF THE ROTATION MATRIX $\mathbf{R}$

The rotation matrix  $\mathbf{R}$  is 3X3 orthonormal matrix (i.e.,  $\mathbf{R}^{-1} = \mathbf{R}^t$ ) of the first kind (i.e.,  $\det(\mathbf{R}) = +1$ ) and not of the second kind (i.e. its determinant is not -1). Its property that the determinant must be unity implies the following constraints :

$$\|\mathbf{p}_1\|^2 = \|\mathbf{p}_2\|^2 = \|\mathbf{p}_3\|^2 = 1; \quad (\text{A.2a})$$

$$\mathbf{p}_1 = \mathbf{p}_2 \times \mathbf{p}_3; \quad \mathbf{p}_2 = \mathbf{p}_3 \times \mathbf{p}_1; \quad \mathbf{p}_3 = \mathbf{p}_1 \times \mathbf{p}_2 \quad (\text{A.2b})$$

and

$$\mathbf{p}_1 \cdot \mathbf{p}_2 = \mathbf{p}_2 \cdot \mathbf{p}_3 = \mathbf{p}_3 \cdot \mathbf{p}_1 = 0; \quad (\text{A.2c})$$

where the operations  $\|\cdot\|^2$ ,  $\times$  and  $\cdot$  represent the norm, cross-product, and dot-product respectively. Also,

$$\|\mathbf{c}_1\|^2 = \|\mathbf{c}_2\|^2 = \|\mathbf{c}_3\|^2 = 1; \quad (\text{A.2d})$$

$$\mathbf{c}_1 = \mathbf{c}_2 \times \mathbf{c}_3; \quad \mathbf{c}_2 = \mathbf{c}_3 \times \mathbf{c}_1; \quad \mathbf{c}_3 = \mathbf{c}_1 \times \mathbf{c}_2 \quad (\text{A.2e})$$

and

$$\mathbf{c}_1 \cdot \mathbf{c}_2 = \mathbf{c}_2 \cdot \mathbf{c}_3 = \mathbf{c}_3 \cdot \mathbf{c}_1 = 0 ; \quad (\text{A.2f})$$

The equations (A.2a) through (A.2f) can also be written as

$$r_1^2 + r_2^2 + r_3^2 = r_4^2 + r_5^2 + r_6^2 = r_7^2 + r_8^2 + r_9^2 = 1 ; \quad (\text{A.3a})$$

$$r_1^2 + r_4^2 + r_7^2 = r_2^2 + r_5^2 + r_8^2 = r_3^2 + r_6^2 + r_9^2 = 1 ; \quad (\text{A.3b})$$

$$r_1 = r_5 r_9 - r_6 r_8 ; r_2 = r_6 r_7 - r_4 r_9 ; r_3 = r_4 r_8 - r_5 r_7 ; \quad (\text{A.3c})$$

$$r_4 = r_8 r_3 - r_2 r_9 ; r_5 = r_1 r_9 - r_3 r_7 ; r_6 = r_2 r_7 - r_1 r_8 ; \quad (\text{A.3d})$$

$$r_7 = r_2 r_6 - r_3 r_5 ; r_8 = r_3 r_4 - r_1 r_6 ; r_9 = r_1 r_5 - r_2 r_4 \quad (\text{A.3e})$$

$$r_1 r_4 + r_2 r_5 + r_3 r_6 = r_4 r_7 + r_5 r_8 + r_6 r_9 = r_7 r_1 + r_8 r_2 + r_9 r_3 = 0 . \quad (\text{A.3f})$$

and

$$r_1 r_2 + r_4 r_5 + r_7 r_8 = r_2 r_3 + r_5 r_6 + r_8 r_9 = r_3 r_1 + r_6 r_4 + r_9 r_7 = 0 . \quad (\text{A.3g})$$

All of these are not independent. Their interdependence can be recognized from the following identity :

$$\begin{aligned} (r_1 r_4 + r_2 r_5 + r_3 r_6)^2 &= (r_1^2 + r_2^2 + r_3^2)(r_4^2 + r_5^2 + r_6^2) - (r_1 r_5 - r_2 r_4)^2 \\ &\quad - (r_2 r_6 - r_3 r_5)^2 - (r_3 r_4 - r_1 r_6)^2 . \end{aligned} \quad (\text{A.3h})$$

## A.2. THE TWO REPRESENTATIONS OF THE ROTATION MATRIX $\mathbf{R}$

The rotation matrix  $\mathbf{R}$ , used extensively in the motion analysis, can be expressed in either of the two representations explained in what follows :

### A.2.1. The First Representation

The rotation of an object by an angle  $\theta$  around an axis, passing through the origin of the frame with which the camera co-ordinate system aligns, with directional cosines  $v_1, v_2, v_3$ , is expressed as a rotation matrix  $R$  by Huang *et al.* ([5]), where

$$R = \begin{bmatrix} v_1^2 + (1 - v_1^2) \cos \theta & v_1 v_2 (1 - \cos \theta) - v_3 \sin \theta & v_1 v_3 (1 - \cos \theta) + v_2 \sin \theta \\ v_1 v_2 (1 - \cos \theta) + v_3 \sin \theta & v_2^2 + (1 - v_2^2) \cos \theta & v_2 v_3 (1 - \cos \theta) - v_1 \sin \theta \\ v_1 v_3 (1 - \cos \theta) - v_2 \sin \theta & v_2 v_3 (1 - \cos \theta) + v_1 \sin \theta & v_3^2 + (1 - v_3^2) \cos \theta \end{bmatrix}$$

with the additional constraint : (A.4)

$$v_1^2 + v_2^2 + v_3^2 = 1. \quad (A.5)$$

#### A.2.1.1. Computation of the rotation parameters

Any matrix can be decomposed uniquely into a sum of a symmetric and a skew-symmetric matrix. Thus

$$R = S + K ; \quad (A.6a)$$

where  $S$  is symmetric and  $K$  is skew-symmetric. Since

$$R^t = S - K ; \quad (A.6b)$$

we get

$$S = 1/2 (R + R^t) ; \quad K = 1/2 (R - R^t) ; \quad (A.7)$$

where

$$S = \begin{bmatrix} v_1^2 + (1 - v_1^2) \cos \theta & v_1 v_2 (1 - \cos \theta) & v_1 v_3 (1 - \cos \theta) \\ v_1 v_2 (1 - \cos \theta) & v_2^2 + (1 - v_2^2) \cos \theta & v_2 v_3 (1 - \cos \theta) \\ v_1 v_3 (1 - \cos \theta) & v_2 v_3 (1 - \cos \theta) & v_3^2 + (1 - v_3^2) \cos \theta \end{bmatrix} \quad (A.8a)$$

and

$$\mathbf{K} = \sin \theta \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix}. \quad (\text{A.8b})$$

As is clear from eqn.(A.4), there are exactly two sets of rotation parameters corresponding to a particular rotation matrix  $\mathbf{R}$ , with one set the negative of the other. The relationship between  $\mathbf{R}$  and its parameters may be regarded as one to one because they can be computed very easily, once  $\mathbf{R}$  is determined.

From eqn.(A.4), we see that

$$\mathbf{K} = 1/2 \begin{bmatrix} 0 & r_2-r_4 & r_3-r_7 \\ r_4-r_2 & 0 & r_6-r_8 \\ r_7-r_3 & r_8-r_6 & 0 \end{bmatrix}; \quad (\text{A.9})$$

or

$$v_1 \cdot \sin \theta = (r_8 - r_6)/2; \quad v_2 \cdot \sin \theta = (r_3 - r_7)/2; \quad v_3 \cdot \sin \theta = (r_4 - r_2)/2; \quad (\text{A.10})$$

Solving the above equation for the rotation parameters gives

$$\sin \theta = \pm (d/2); \quad v_1 = \pm (r_8 - r_6)/d; \quad v_2 = \pm (r_3 - r_7)/d; \quad v_3 = \pm (r_4 - r_2)/d;$$

where

$$d^2 = (r_8 - r_6)^2 + (r_3 - r_7)^2 + (r_4 - r_2)^2. \quad (\text{A.11b})$$

In order to fix  $\theta$ , we need to define its cosine as

$$\cos \theta = \frac{r_1 - v_1^2}{1 - v_1^2} = \frac{d^2 r_1 - (r_8 - r_6)^2}{d^2 - (r_8 - r_6)^2}. \quad (\text{A.11c})$$

This representation is generally used in connection with the rigid-body motion of an object.

### A.2.2. The Second Representation

The rotation is also expressed in terms of the Euler angles :  $\theta$  ( *roll* or *tilt* ),  $\phi$  ( *yaw* or *swing* ), and  $\psi$  ( *pitch* or *pan* ). Here, the rotation matrix  $\mathbf{R}$  is given by, as shown

by S. Ganapathy ([27]),

$$\mathbf{R} = \mathbf{R}_\phi \mathbf{R}_\theta \mathbf{R}_\psi \quad (\text{A.12})$$

where  $\mathbf{R}_\psi$  represents a rotation of angle  $\psi$  around the z-axis in the camera co-ordinate frame,  $\mathbf{R}_\theta$  represents a rotation of angle  $\theta$  around the new x-axis, and  $\mathbf{R}_\phi$  represents the rotation of angle  $\phi$  around the new y-axis. This is one of the conventions used in this research - the other conventions being the product of the three matrices in any manner. These conventions do not necessarily give the same rotation matrix  $\mathbf{R}$  as given by the first convention used in this research because the product of the matrices, in general, are not commutative. The three rotations are individually given as

$$\mathbf{R}_\psi = \begin{bmatrix} \cos \psi & \sin \psi & 0 \\ -\sin \psi & \cos \psi & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad -\pi \leq \psi < +\pi ; \quad (\text{A.13a})$$

$$\mathbf{R}_\theta = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta & \sin \theta \\ 0 & -\sin \theta & \cos \theta \end{bmatrix}, \quad -\pi/2 \leq \theta < +\pi/2 \quad (\text{A.13b})$$

and

$$\bar{\mathbf{R}}_\phi = \begin{bmatrix} \cos \phi & 0 & \sin \phi \\ 0 & 1 & 0 \\ -\sin \phi & 0 & \cos \phi \end{bmatrix}, \quad 0 \leq \phi < 2\pi . \quad (\text{A.13c})$$

Here, the rotation parameters are the three Euler angles, and the elements  $r_1, r_2, \dots, r_9$  in the rotation matrix, expressed in terms of these angles are :

$$r_1 = \cos \phi \cos \psi - \sin \psi \sin \phi \sin \theta ; \quad (\text{A.14a})$$

$$r_2 = \cos \phi \sin \psi + \cos \psi \sin \phi \sin \theta ; \quad (\text{A.14b})$$

$$r_3 = -\cos \theta \sin \phi ; \quad (\text{A.14c})$$

$$r_4 = -\sin \psi \cos \theta ; \quad (\text{A.14d})$$



$$r_5 = \cos \psi \cos \theta ; \quad (\text{A.14e})$$

$$r_6 = \sin \theta ; \quad (\text{A.14f})$$

$$r_7 = \cos \psi \sin \phi + \sin \psi \sin \theta \cos \phi ; \quad (\text{A.14g})$$

$$r_8 = \sin \psi \sin \phi - \cos \psi \sin \theta \cos \phi ; \quad (\text{A.14h})$$

$$r_9 = \cos \phi \cos \theta . \quad (\text{A.14i})$$

### A.2.2.1. Computation of the rotation parameters

Solving for  $\theta$  from eqn.(A.14d) through eqn.(A.14f), we get the cosine and the sine of the angle in order to fix  $\theta$  as  $\sin \theta$  alone does not determine  $\theta$  uniquely. Same procedure will be applied to other angles. Therefore,

$$\sin \theta = r_6 ; \quad \cos \theta = (r_4^2 + r_5^2)^{1/2} . \quad (\text{A.15})$$

Similarly, solving for  $\psi$  from the same set of equations, we get

$$\sin \psi = -\frac{r_4}{\cos \theta} = -\frac{r_4}{(1-r_6^2)^{1/2}} ; \quad \cos \psi = \frac{r_5}{\cos \theta} = \frac{r_5}{(1-r_6^2)^{1/2}} ; \quad (\text{A.16})$$

Solving for  $\phi$  from eqn.(A.14c) and eqn.(A.14i), we get

$$\sin \phi = -\frac{r_3}{\cos \theta} = -\frac{r_3}{(1-r_6^2)^{1/2}} ; \quad \cos \phi = \frac{r_9}{\cos \theta} = \frac{r_9}{(1-r_6^2)^{1/2}} ; \quad (\text{A.17})$$

**Special Cases** : The similarity for the two approaches used so far can be found if the arbitrary axis in the first case aligns with any of the three axis. The rotation matrix  $R$  reduces to one of the rotation matrices for the respective axis. For example, if the arbitrary axis aligns with the x-axis, we have

$$v_1 = 1 ; \quad v_2 = 0 ; \quad v_3 = 0$$

and

$$\mathbf{R} |_{v_1=1} = \mathbf{R}_\theta . \quad (\text{A.18a})$$

Other results are obvious for alignments of the arbitrary axis with y- and z-axes. Thus,

$$\mathbf{R} |_{v_2=1} = \mathbf{R}_\phi \quad (\text{A.18b})$$

and

$$\mathbf{R} |_{v_3=1} = \mathbf{R}_\psi \quad (\text{A.18c})$$

## APPENDIX B

### ON THE SPATIAL DISTRIBUTION OF THE SURFACE POINTS

In this Appendix, we shall investigate the number of conjugate pairs that are needed for  $\Gamma$  and  $R$  to be unique factors of  $Q$  in chapter III ( eqn.(3.8b) ), and state the restrictions on the spatial distribution of the points on the surface of the object undergoing rigid-body motion in order for the IPC algorithm to work satisfactorily. The various developments presented in this Appendix are shown by Huang *et al.* ( [5] ).

Let  $r$  be transformed to  $r'$  with some reference rotation matrix  $R_r$  and reference translational vector  $T_r$ , where

$$T_r = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}. \quad (B.1)$$

Thus,

$$r' = R_r r + T_r. \quad (B.2)$$

Let

$$Q_r = \Gamma_r R_r \quad (B.3a)$$

where

$$\Gamma_r = \begin{bmatrix} 0 & t_z & -t_y \\ -t_z & 0 & t_x \\ t_y & -t_x & 0 \end{bmatrix}. \quad (B.3b)$$

Therefore, eqn.(3.2a) becomes

$$(r'^t R_r^t + T_r^t) Q_r r = \begin{bmatrix} x & y & z & 1 \end{bmatrix} C \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = 0; \quad (B.4a)$$

where

$$\mathbf{C} = \begin{bmatrix} \mathbf{R}_r^t \mathbf{Q} & \mathbf{0} \\ \mathbf{T}_r^t \mathbf{Q} & \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{R}_r^t \mathbf{\Gamma} \mathbf{R} & \mathbf{0} \\ \mathbf{T}_r^t \mathbf{\Gamma} \mathbf{R} & \mathbf{0} \end{bmatrix} \quad (\text{B.4b})$$

Note that if  $\mathbf{C}$  is skew-symmetric, eq.(B.4a) is always satisfied, irrespective of what  $x$ ,  $y$ ,  $z$ , or  $X$ ,  $Y$  are.

*Lemma* : The necessary and sufficient conditions for  $\mathbf{C}$  to be skew-symmetric is that

$$\mathbf{R} = \mathbf{R}_r ; \quad (\text{B.5a})$$

or

$$\mathbf{R} = \mathbf{U}^t \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{U} \mathbf{R}_r ; \quad (\text{B.5b})$$

where  $\mathbf{U}$  is a 3 x 3 orthonormal matrix such that

$$\mathbf{\Gamma} = \mathbf{U}^t \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{U} ; \mathbf{Q} = \alpha \mathbf{Q}_r \quad \text{and} \quad \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} = \alpha \begin{bmatrix} t_{x_r} \\ t_{y_r} \\ t_{z_r} \end{bmatrix} ; \quad (\text{B.6})$$

where  $\alpha$  is some constant. The proof for this lemma is given in [5].

We will now present a theorem to show that  $\mathbf{C}$  has to be skew-symmetric if some conditions on the surface points are satisfied.

*Theorem* : If

$$\begin{bmatrix} X' & Y' & 1 \end{bmatrix} \mathbf{Q} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \quad (\text{B.7})$$

is satisfied by the image point correspondence of a group of object points satisfying the following conditions :

(i) the group of points do not reside on two planes with one plane containing the

origin and

(ii) the points are not traversed by a cone containing the origin,

then  $\mathbf{C}$  in eqn.(B.4a) has to be skew-symmetric. The proof for this theorem is given in [5].

We note that five or fewer points in space can always be traversed by two planes with one plane containing the origin and that six or fewer points in space can always be found to reside on the surface of a cone containing the origin. Thus a minimum of seven points is needed to violate these conditions and ensure a unique solution for the motion parameters, which follows from the theorem and the lemma.

The criterion for the existence of a cone containing the origin that passes through 'n' points is whether the following  $n \times 7$  matrix has full column rank or not :

$$\begin{bmatrix} x_1^2 & x_1 \cdot y_1 & y_1^2 & x_1 & y_1 & z_1 \cdot x_1 & z_1 \cdot y_1 \\ x_2^2 & x_2 \cdot y_2 & y_2^2 & x_2 & y_2 & z_2 \cdot x_2 & z_2 \cdot y_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n^2 & x_n \cdot y_n & y_n^2 & x_n & y_n & z_n \cdot x_n & z_n \cdot y_n \end{bmatrix}. \quad (\text{B.8})$$

However, we know the image coordinates instead of the 3D coordinates. The only useful criterion available is whether the  $8 \times 8$  matrix  $\mathbf{N}$  in eqn.(3.3c) is non-singular or not, assuming the correspondence of exactly eight points is known.

## APPENDIX C

### PROOF FOR ' THE FIRST METHOD '

The proofs for eqns.(3.6a) and (3.6b) in chapter III will be presented here. We will verify eqn.(3.6c) first by writing

$$\mathbf{Q} \mathbf{Q}^t = \mathbf{\Gamma} \mathbf{R} \mathbf{R}^t \mathbf{\Gamma}^t = -\mathbf{\Gamma}^2 = \begin{bmatrix} t_z^2 + t_y^2 & -t_x t_y & -t_x t_z \\ -t_x t_y & t_z^2 + t_x^2 & -t_y t_z \\ -t_x t_z & -t_x t_z & t_x^2 + t_y^2 \end{bmatrix} \quad (\text{C.1a})$$

$$= \begin{bmatrix} \|l_1\|^2 & (l_1, l_2) & (l_1, l_3) \\ (l_2, l_1) & \|l_2\|^2 & (l_2, l_3) \\ (l_3, l_1) & (l_3, l_2) & \|l_3\|^2 \end{bmatrix}. \quad (\text{C.1b})$$

Therefore,

$$t_z^2 + t_y^2 = \|l_1\|^2; \quad (\text{C.2a})$$

$$t_z^2 + t_x^2 = \|l_2\|^2; \quad (\text{C.2b})$$

$$t_x^2 + t_y^2 = \|l_3\|^2; \quad (\text{C.2c})$$

$$(l_1, l_2) = -t_x t_y = (l_2, l_1); \quad (\text{C.2d})$$

$$(l_3, l_1) = -t_x t_z = (l_1, l_3); \quad (\text{C.2e})$$

$$(l_2, l_3) = -t_y t_z = (l_3, l_2); \quad (\text{C.2f})$$

Solving the above set of equations for  $t_x$ ,  $t_y$ , and  $t_z$  we get the required solution for the translational vector as shown in eqn.(3.6c) or eqn.(3.11) or eqn.(3.13) or eqn.(3.15). This vector is estimated up to a scale factor because of the ratios of the *essential parameters*.

We shall verify eqns.(3.6a,3.6b) now, as shown by Huang *et al.* ( see [5] ).

Any skew-symmetric matrix  $\Gamma$  can be expressed in the following normal form

$$\Gamma = \Omega^t \begin{bmatrix} 0 & \phi & 0 \\ -\phi & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \Omega . \quad (\text{C.3})$$

If  $\Theta = i.Q$ , where  $i = \sqrt{-1}$ , we have

$$\Theta = \Psi R ; \quad (\text{C.4})$$

where  $\Psi = i.\Gamma$  is hermitian. Eqn.(C.4) gives the polar decomposition of  $\Theta$ . We can see that  $\Gamma$  and  $R$  would be unique if the polar decomposition of the matrix  $\Theta$  with distinct singular values is unique but in that case  $\Theta$  has to be nonsingular. As it is seen,  $\Gamma$  is singular and hence  $\Theta$  is singular, and

$$\Theta \Theta^* = \Psi^2 = -\Gamma^2 \quad (* \text{ denotes the conjugate operation } )$$

$$= \Omega^t \begin{bmatrix} \phi^2 & 0 & 0 \\ 0 & \phi^2 & 0 \\ 0 & 0 & 0 \end{bmatrix} \Omega .$$

Therefore, the singular values of  $\Theta$  are  $\phi$ ,  $\phi$  and 0 (repeated). However, because of a special structure of  $\Gamma$ , once  $Q$  is given,  $\Gamma$  and  $R$  are unique and is proved as below :

From previous equations, we see that

$$Q = U \Delta V^t = \Gamma R = \Omega^t \begin{bmatrix} 0 & \phi & 0 \\ -\phi & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \Omega R .$$

Since

$$Q Q^t = \Gamma \Gamma^t = \Omega^t \begin{bmatrix} \phi^2 & 0 & 0 \\ 0 & \phi^2 & 0 \\ 0 & 0 & 0 \end{bmatrix} \Omega ,$$

therefore,  $\Omega^t$  is one of the singular vector matrices of  $Q$ , namely,  $U$ . Therefore,

$$\mathbf{Q} = \mathbf{U} \Delta \mathbf{V}^t = \mathbf{U} \begin{bmatrix} 0 & \phi & 0 \\ -\phi & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{U}^t \mathbf{R} . \quad (\text{C.5a})$$

Premultiplying by  $\mathbf{U}^t$  gives

$$\begin{bmatrix} \pm 1 & 0 & 0 \\ 0 & \pm 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{V}^t = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{U}^t \mathbf{R}$$

after canceling  $\phi$  throughout. Again premultiplying by

$$\begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

we obtain

$$\begin{bmatrix} 0 & \mp 1 & 0 \\ \pm 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{V}^t = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{U}^t \mathbf{R} . \quad (\text{C.5b})$$

Eqn.(C.5b) shows that the first and second rows of the orthonormal matrix  $\mathbf{U}^t \mathbf{R}$  are the same as the second and first rows of  $\mathbf{V}$  respectively, except for a possible sign change. This implies that the third row of  $\mathbf{U}^t \mathbf{R}$  must equal the third row of  $\mathbf{V}$  up to a possible sign difference. Therefore,

$$\begin{bmatrix} 0 & \mp 1 & 0 \\ \pm 1 & 0 & 0 \\ 0 & 0 & \omega \end{bmatrix} \mathbf{V}^t = \mathbf{U}^t \mathbf{R} ; \quad (\text{C.6})$$

where  $\omega = \pm 1$ .

Taking determinants of both sides of eqn.(C.6), we have

$$\omega = \det(\mathbf{U}) / \det(\mathbf{V}) . \quad (\text{C.7})$$

Thus eqn.(C.6) gives

$$\mathbf{R} = \mathbf{U} \begin{bmatrix} 0 & \mp 1 & 0 \\ \pm 1 & 0 & 0 \\ 0 & 0 & \omega \end{bmatrix} \mathbf{V}^t . \quad (\text{C.8})$$



which proves eqn.(3.6a) and eqn.(3.6b). Although  $\mathbf{U}$  and  $\mathbf{V}$  are not unique, eqn.(C.8) gives us only two possible solutions for  $\mathbf{R}$ , and is proved below :

Let  $\mathbf{U}_1$  and  $\mathbf{V}_1$  be some singular vector matrices of  $\mathbf{Q}$ . Also, let

$$\mathbf{R}_1 = \mathbf{U}_1 \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & \omega \end{bmatrix} \mathbf{V}_1^t \text{ and}$$

$$\mathbf{R}_2 = \mathbf{U}_1 \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & \omega \end{bmatrix} \mathbf{V}_1^t .$$

Suppose  $\mathbf{U}_2$  and  $\mathbf{V}_2$  be any other singular vector matrices of  $\mathbf{Q}$  and let

$$\mathbf{R}_3 = \mathbf{U}_2 \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & \omega \end{bmatrix} \mathbf{V}_2^t \text{ and}$$

$$\mathbf{R}_4 = \mathbf{U}_2 \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & \omega \end{bmatrix} \mathbf{V}_2^t .$$

We observe that eqn.(C.5a) must be maintained whether  $\mathbf{U}$  equals  $\mathbf{U}_1$  and  $\mathbf{U}_2$ , irrespective of  $\mathbf{R}$ . If  $\mathbf{R}_3$  and  $\mathbf{R}_4$  are different from  $\mathbf{R}_1$  and  $\mathbf{R}_2$ , contradiction arises if we replaced  $\mathbf{R}$  by  $\mathbf{R}_3$  ( or  $\mathbf{R}_4$  ) and  $\mathbf{U}$  by  $\mathbf{U}_1$  since eq.(C.8) says  $\mathbf{R}$  must be equal to  $\mathbf{R}_1$  ( or  $\mathbf{R}_2$  ) which is different from  $\mathbf{R}_3$  ( or  $\mathbf{R}_4$  ). Hence, we conclude that there are only two possible solutions for  $\mathbf{R}$  no matter what  $\mathbf{U}$  and  $\mathbf{V}$  are. The reason for two possible solutions for  $\mathbf{R}$  has been explained earlier.

**APPENDIX D**  
**PROOF FOR ' THE SECOND METHOD '**

The proof for the translational vector has already been established in Appendix C. In this appendix, the proof for eqn.(3.12) in chapter III is presented. The proofs for eqn.(3.14) and eqn.(3.16) are similar and will not be derived here.

We start with the definition of  $\mathbf{Q}$  in eqn.(3.2d). It is seen that

$$\begin{aligned} (\mathbf{l}_2 \times \mathbf{l}_3) = & (r_1 t_x^2 + r_4 t_x t_y + r_7 t_x t_z) \hat{\mathbf{i}} + (r_2 t_x^2 + r_5 t_x t_y + r_8 t_x t_z) \hat{\mathbf{j}} \\ & + (r_3 t_x^2 + r_6 t_x t_y + r_9 t_x t_z) \hat{\mathbf{k}} ; \end{aligned} \quad (\text{D.1})$$

where  $\hat{\mathbf{i}}$ ,  $\hat{\mathbf{j}}$  and  $\hat{\mathbf{k}}$  are the unit vectors along x-, y- and z- axes respectively. The properties of the rotation matrix  $\mathbf{R}$  from the Appendix A have been extensively used. Similarly

$$\begin{aligned} (\mathbf{l}_2 \times \mathbf{l}_3) \times \mathbf{l}_1 = & [\mp r_1 t_x (t_y^2 + t_z^2) \pm r_4 t_x^2 t_y \pm r_7 t_x^2 t_z] \hat{\mathbf{i}} \\ & + [\mp r_2 t_x (t_y^2 + t_z^2) \pm r_5 t_x^2 t_y \pm r_8 t_x^2 t_z] \hat{\mathbf{j}} \\ & + [\mp r_3 t_x (t_y^2 + t_z^2) \pm r_6 t_x^2 t_y \pm r_9 t_x^2 t_z] \hat{\mathbf{k}} . \end{aligned} \quad (\text{D.2})$$

Therefore, it can be shown that

$$(\mathbf{l}_2 \times \mathbf{l}_3) \times \mathbf{l}_1 \mp t_x (\mathbf{l}_2 \times \mathbf{l}_3) = \mp [r_1 \ r_2 \ r_3] t_x (t_x^2 + t_y^2 + t_z^2)$$

or

$$\mathbf{p}_1 = \frac{(\mathbf{l}_2 \times \mathbf{l}_3) \times \mathbf{l}_1 \mp t_x (\mathbf{l}_2 \times \mathbf{l}_3)}{(\mp t_x \|\mathbf{T}\|^2)}$$

where  $\|\cdot\|^2$  denotes the norm operation. Thus, we complete the proof for eqn.(3.12a) and eqn.(3.12b). The proofs for eqn.(3.12c) through eqn.(3.12f) are trivial and will not be discussed here.

**APPENDIX E**  
**PROOF FOR ' THE THIRD METHOD '**

In this appendix, the proof for eqn.(3.21) in chapter III, as shown by H. C. Longuet-Higgins, is presented as follows :

From eqn.(3.20a), we observe that  $\mathbf{c}_\alpha$  is orthogonal to  $\mathbf{h}_\alpha$  and, therefore, can be expressed as a linear combination of  $\mathbf{T}$  and  $\mathbf{h}_\alpha \times \mathbf{T}$ . Writing

$$\mathbf{c}_\alpha = \tau_\alpha \mathbf{T} + \eta_\alpha \mathbf{W}_\alpha \quad (\text{E.1})$$

and substituting in eqn.(3.20a), we get

$$\mathbf{h}_\alpha = \mathbf{T} \times (\tau_\alpha \mathbf{T} + \eta_\alpha \mathbf{W}_\alpha) = \eta_\alpha (\mathbf{T} \times \mathbf{W}_\alpha) = \mathbf{h}_\alpha ,$$

since  $\|\mathbf{T}\|^2$  is unity. Thus,

$$\eta_\alpha = 1 . \quad (\text{E.2})$$

From the following property of  $\mathbf{R}$  ( Appendix A )

$$\mathbf{c}_\alpha = \mathbf{c}_\beta \times \mathbf{c}_\gamma ,$$

we get

$$\tau_\alpha \mathbf{T} + \mathbf{W}_\alpha = (\tau_\beta \mathbf{T} + \mathbf{W}_\beta) \times (\tau_\gamma \mathbf{T} + \mathbf{W}_\gamma) = \tau_\beta \mathbf{h}_\gamma - \tau_\gamma \mathbf{h}_\beta + \mathbf{W}_\beta \times \mathbf{W}_\gamma . \quad (\text{E.3})$$

In eqn.(E.3), the vectors  $\mathbf{W}_\alpha$ ,  $\mathbf{h}_\beta$  and  $\mathbf{h}_\gamma$  are all orthogonal to  $\mathbf{T}$ , whereas  $\mathbf{W}_\beta \times \mathbf{W}_\gamma$  is, by eqn.(3.20b), a multiple of  $\mathbf{T}$ . Thus,

$$\tau_\alpha \mathbf{T} = \mathbf{W}_\beta \times \mathbf{W}_\gamma \quad (\text{E.4})$$

from which we deduce eqn.(3.21) by substituting eqn.(E.2) and eqn.(E.4) in eqn.(E.1).

## APPENDIX F

In this appendix, the proofs for developments given in chapter IV are presented.

We shall first derive eqn.(4.2).

For transformation of  $F_i$  to  $S$ , let  $F_j$  coincide with  $S$ . In that case

$$R_{ij} = R_i, \quad T_{ij} = T_i, \quad r_{ji} = r, \quad R_j = I = \Phi_j^{-1}. \quad (F.1)$$

Substituting eqn.(F.1) in eqn.(4.1) gives eqn.(4.2a).

Similarly, let  $F_i$  be  $S$ , where

$$R_{ij} = R_j^{-1}, \quad T_{ij} = -R_j^{-1} T_j, \quad r_{jj} = r, \quad (F.2)$$

$$\Phi_i^{-1} = I = R_i.$$

Substituting of eqn.(F.2) in eqn.(4.1) results in eqn.(4.2b).

Eliminating  $r$  from eqn.(4.2a) and eqn.(4.2b), we have

$$\Phi_i r_{ii} = \Phi_i R_i R_j^{-1} r_{jj} + \Phi_i (T_i - R_i R_j^{-1} T_j). \quad (F.3)$$

Comparison of the eqn.(F.3) with eqn.(4.1) gives eqns.(4.6a) and (4.6b).

The derivation of eqn.(4.7) is as follows :

The 2-D image co-ordinates of  $r_{ij}$  and  $r_{ji}$  are not observed on the image plane, because of the manner in which the pictures are taken. So, eliminating  $r_{ij}$  from eqns.(4.4) and (4.5a) yields

$$\Phi_i (R_i r' + T_i) = \Phi_i R_{ij} r_{jj} + T_{ij}. \quad (F.4a)$$

Elimination of  $r'$  from eqns.(4.3) and (F.4a) gives

$$\Phi_i (R_i R r + R_i T + T_i) = \Phi_i R_{ij} r_{jj} + T_{ij}. \quad (F.4b)$$

Furthermore, elimination of  $r$  from eqns.(4.2a) and (F.4b) yields the desired relationship between  $r_{jj}$  and  $r_{ii}$  as shown in eqn.(4.7).

If  $F_i$  coincides with  $S$ , we have

$$r_{ii} = r, \quad R_i = I = \Phi_i^t, \quad T_i = O, \quad T_{ij} = -R_{ij} T_j = -R_j^t T_j. \quad (F.5)$$

Substituting eqn.(F.5) into eqn.(4.7), we get eqn.(4.8).

We shall now derive eqns.(4.14a,4.14b,4.14c).

The camera moves first in this case, so that

$$\Phi_i r_{ii} = \Phi_i R_{ij} r_{ji} + T_{ij}. \quad (F.6a)$$

Then the object moves, so that

$$r' = R r + T. \quad (F.6b)$$

From eqns.(F.6a,F.6b,4.2b,4.3, and 4.5b), the relationship between  $r_{jj}$  and  $r_{ii}$  can be found to be the one expressed in eqns.(4.14a,4.14b,4.14c).

## APPENDIX G

### SOFTWARE FOR THE IPC AND GIPC ALGORITHMS

In this Appendix, the details of the programs for the three methods for the IPC and the GIPC algorithms are presented. The software for the three methods was developed so as to complete an extensive literature survey on the existing techniques for the IPC algorithms. In addition to that, two different representations of the rotation matrix, as shown in Appendix A, have been used. Different cases have been taken where the number of available features equals, and exceeds eight. The input to the software was either simulated or the real-time data. The software was written for each case, and kept in a different file. A main program accesses each file. There are, therefore, twenty four cases. Another main program for testing the GIPC algorithm has been dealt with similarly. The outlines of these programs are shown in Fig.25 and Fig.26.

File *ipc.c* is the main user-friendly program for estimating 3-D motion parameters using the IPC algorithm. The two pictures, required for the algorithm, are taken by a stationary camera. Its executable file is *ipc*. The notations used for the twenty four cases are given as follows :

**S**: Simulated data (in *test\_coord*); **D**: Real data (in *nasa\_data*);

**I, II, III** : The first, second and third methods respectively for the IPC algorithm;

**A** : First representation of **R**; **B** : Second representation of **R**.

**N** :  $n > 8$ ; **M** :  $n = 8$ ; where 'n' is the number of data-points used.

The options available, with a separate program for each option and kept in a separate file, are given as follows :

*Options available ( File names containing the software ) :*

0 : S I A N (progl.c) ; 9 : S III A M (proglIIa.c); 18 : D II B N (progl\_e.c);  
 1 : S I A M (progl.a.c) ; 10 : S III B N (proglIII\_e.c); 19 : D II B M (proglIIa\_e.c);  
 2 : S I B N (progl\_e.c); 11 : S III B M (proglIIa\_e.c); 20 : D III A N (proglIII.c);  
 3 : S I B M (progl.a\_e.c); 12 : D I A N (progl.c); 21 : D III A M (proglIIa.c);  
 4 : S II A N (progl.c); 13 : D I A M (progl.a.c); 22 : D III B N (proglIII\_e.c);  
 5 : S II A M (proglIIa.c); 14 : D I B N (progl\_e.c); 23 : D III B M (proglIIa\_e.c);  
 6 : S II B N (progl\_e.c); 15 : D I B M (progl.a\_e.c); 24 : quit.  
 7 : S II B M (proglIIa\_e.c); 16 : D II A N (progl.c);  
 8 : S III A N (proglIII.c); 17 : D II A M (proglIIa.c);

The options are kept in a separate file called *ipc\_cases*.

File *gipc.c* is another user-friendly main program for estimating 3-D motion parameters using the GIPC algorithm. The two pictures required for the algorithm are taken by a moving camera. Its executable file is *gipc*.

As before, all different cases are considered. The options available are given as follows :

*Options available ( File names containing the software ) :*

0 : S I A N (gipcl.c) ; 9 : S III A M (gipclIIa.c); 18 : D II B N (gipcl\_e.c);  
 1 : S I A M (gipcla.c) ; 10 : S III B N (gipclIII\_e.c); 19 : D II B M (gipclIIa\_e.c);  
 2 : S I B N (gipcl\_e.c); 11 : S III B M (gipclIIa\_e.c); 20 : D III A N (gipclIII.c);  
 3 : S I B M (gipcla\_e.c); 12 : D I A N (gipcl.c); 21 : D III A M (gipclIIa.c);  
 4 : S II A N (gipcl.c); 13 : D I A M (gipcla.c); 22 : D III B N (gipclIII\_e.c);

5 : S I I A M (gipcIIa.c); 14 : D I B N (gipcl\_e.c); 23 : D I I I B M (gipcIIIIa\_e.c);

6 : S I I B N (gipcII\_e.c); 15 : D I B M (gipcla\_e.c); 24 : quit.

7 : S I I B M (gipcIIa\_e.c); 16 : D I I A N (gipcII.c);

8 : S I I I A N (gipcIIII.c); 17 : D I I A M (gipcIIa.c);

The options are kept in a separate file called *gipc\_cases*. The files containing the software for various options have not been included in this appendix because they are voluminous.



```

/*****
/* IMAGE POINT CORRESPONDENCE ALGORITHM : All the Methods
/*****
/* Author:Sunil Fotedar
/* Co-author : Dr. Rui J. P. de Figueiredo
/* Electrical & Computer Engineering Department
/* Rice University, Houston
/*****
/* This is the main program for estimating 3-D motion parameters
/* using the "Image Point Correspondence" method.
/*****
/* FILE NAME : ipc.c
/*****
#include <stdio.h>
#include <math.h>
#include "sunil.h"

main()
{
    int    i,j,points,data;
    double x(Max),y(Max),z(Max),
           xr[Max],yr[Max],zr[Max],
           R[3][3],
           T[3],
           a,b,c,theta,phi,psi,pi;

    FILE *fp;

    pi=4.0*atan(1.0);

    printf("\n\n\t\t\tTHE DEMONSTRATION OF THE IPC ALGORITHM\n\n");
    printf("\n\nSpecify the type of data ( S or D ), the number of\n");
    printf("method ( I, II or III ) the representation for the rotation\n");
    printf("matrix ( A or B ), and the number of points used ( N or M ) :");

    printf("\n\nS : Simulated data (in test_coord); D : Real-time data (in nasa_data);");
    printf("\n\nI, II, III : The three methods for the IPC algorithm;");
    printf("\n\nA : rotation around an arbitrary axis; B : Euler angles;");
    printf("\n\nN : n > 8; M : n = 8; where 'n' is the number of data-points used.");

    printf("\n\nOptions available :");

    printf("\n\n0 : S I A N"); printf("\t8 : S III A N");printf("\t16 : D II A N\n");
    printf("\t1 : S I A M"); printf("\t9 : S III A M"); printf("\t17 : D II A M\n");
    printf("\t2 : S I B N"); printf("\t10 : S III B N");printf("\t18 : D II B N\n");
    printf("\t3 : S I B M"); printf("\t11 : S III B M");printf("\t19 : D II B M\n");
    printf("\t4 : S II A N");printf("\t12 : D I A N");printf("\t20 : D III A N\n");
    printf("\t5 : S II A M");printf("\t13 : D I A M");printf("\t21 : D III A M\n");
    printf("\t6 : S II B N");printf("\t14 : D I B N");printf("\t22 : D III B N\n");
    printf("\t7 : S II B M");printf("\t15 : D I B M");printf("\t23 : D III B M\n");
    printf("\t24 : quit\n");

    printf("\n\nEnter option number :\n");

    scanf("%d", &data );

    switch(data) {
#include "ipc_cases"
    }
}
/*****

```

Fig.25 : Computer Program for the IPC Algorithm.

```

/*****
/* GENERALIZED IMAGE POINT CORRESPONDENCE ALGORITHM : All the Methods*/
/*****
/* Author: Sunil Fotedar */
/* Co-author : Dr. Rui J. P. de Figueiredo */
/* Electrical & Computer Engineering Department */
/* Rice University, Houston */
/*****
/* This is the main program for estimating 3-D motion parameters */
/* using the "Generalized Image Point Correspondence" method. */
/*****
/* FILE NAME : gipc.c */
/*****

#include <stdio.h>
#include <math.h>
#include "sunil.h"

main()
{
    int i,j,points,data;
    double x[Max],y[Max],z[Max],
           xr[Max],yr[Max],zr[Max],
           R[3][3],
           T[3],
           theta1,phi1,psil,theta2,phi2,psi2,
           R12[3][3],R1[3][3],R_res[3][3],
           a,b,c,theta,phi,psi,pi;

    FILE *fp;

    pi = 4.0 * atan(1.0);

    printf("\n\n\t\t\t\t\tTHE DEMONSTRATION OF THE GIPC ALGORITHM\n\n");

    printf("\nEnter the values for the Euler angles (theta1,phi1,& psi1\n");
    printf("for R12, and theta2,phi2,& psi2 for R1), then hit 'Return' key.\n");
    printf("R12 is the frame transformation matrix between the frames F1\n");
    printf("and F2. Similarly, R1 is between F1 and the standard frame S.\n");

    scanf("%f %f %f %f %f %f",&theta1,&phi1,&psi1,&theta2,&phi2,&psi2);

    rot_matII(&theta1,&phi1,&psi1,R12);
    rot_matII(&theta2,&phi2,&psi2,R1);

    printf("\n\nSpecify the type of data ( S or D ), the number of\n");
    printf("method ( I, II or III ) the representation for the rotation\n");
    printf("matrix ( A or B ), and the number of points used ( N or M ) :");

    printf("\n\nS : Simulated data (in test_coord); D : Real-time data (in nasa_data):");
    printf("\nI, II, III : The three methods for the IPC algorithm:");
    printf("\nA : rotation around an arbitrary axis; B : Euler angles:");
    printf("\nN : n > 8; M : n = 8; where 'n' is the number of data-points used.");

    printf("\n\nOptions available :");

    printf("\n\n0 : S I A N"); printf("\t8 : S III A N");printf("\t16 : D II A N\n");
    printf("\t1 : S I A M"); printf("\t9 : S III A M"); printf("\t17 : D II A M\n");
    printf("\t2 : S I B N"); printf("\t10 : S III B N");printf("\t18 : D II B N\n");
    printf("\t3 : S I B M"); printf("\t11 : S III B M");printf("\t19 : D II B M\n");
    printf("\t4 : S II A N");printf("\t12 : D I A N");printf("\t20 : D III A N\n");
    printf("\t5 : S II A M");printf("\t13 : D I A M");printf("\t21 : D III A M\n");
    printf("\t6 : S II B N");printf("\t14 : D I B N");printf("\t22 : D III B N\n");
    printf("\t7 : S II B M");printf("\t15 : D I B M");printf("\t23 : D III B M\n");
    printf("\t24 : quit\n");

    printf("\nEnter option number :\n");

    scanf("%d", &data );

    switch(data) {
#include "gipc_cases"
    }
}
/*****

```

Fig.26 : Computer Program for the GIPC Algorithm.

## REFERENCES

- [1] H. Shariat, " The Motion Problem : How to use more than Two Frames," Institute for Robotics and Intelligent Systems, School of Engineering, University of Southern California, L. A., California 90089-0273, Oct. 6, 1986. Ph. D. Dissertation.
- [2] K. Krishen, R. J. P. de Figueiredo and Olin Graham, " Robotic Vision/Sensing for Space Applications," Proc. 1987 IEEE Int. Conf. on Robotics and Automation, Raleigh, N. Carolina, Mar. 31 - Apr.3, 1987, pp. 138-150.
- [3] J. -Q. Fang and T. S. Huang," Parameters of a Rigid Body from Two Consecutive Image Frames," IEEE Trans. Pattern Anal. Machine Intell., vol. PAMI-6, No.5, Sept 1984, pp. 545-554.
- [4] H. C. Longuet-Higgins," A Computer Algorithm for Reconstructing a Scene from Two Projections," Nature, vol. 293, pp. 133-135, Sept'81.
- [5] R. Y. Tsai and T. S. Huang," Uniqueness and Estimation of Three-Dimensional Motion Parameters of Rigid Objects with Curved Surfaces," IEEE Trans. Pattern Anal. Machine Intell., vol. PAMI-6, No. 1, January 1984, pp. 13-27.
- [6] X. Zhuang and R. M. Haralick," Two-View Motion Analysis, Stereo Vision and a Moving Camera's Positioning : Their Equivalence and a New Solution Procedure," IEEE Int. Conf. on Robotics and Automation, Mar 25-28, 1985, pp. 6-10.
- [7] X. Zhuang, T. S. Huang and R. M. Haralick," From Two-View Motion Equations to Three-Dimensional Motion Parameters and Surface Structure : A Direct and Stable Algorithm," Proc. IEEE Int. Conf. on Robotics and Automation, San Francisco, California, April 7-10, 1986, pp. 621-626.
- [8] D. B. Gennery, " Stereo Vision for the Acquisition and Tracking of Moving Three-dimensional Objects," Techniques for 3-D Machine Perception, Elsevier Science Publishers B. V. ( North - Holland ), 1986, pp. 53-74.
- [9] B. K. P. Horn, Robot Vision, MIT Press, McGraw Hill, 1986.
- [10] B. K. P. Horn and B. G. Schunck," Determining Optical Flow," Artificial Intelligence 17 (1981), pp. 185-203.
- [11] B. G. Schunck," The Motion Constraint Equation for Optical Flow," Proc. 7th Int. Conf. on Pattern Recognition, vol. 1, July 30 - Aug 2, 1984, pp. 20-22.
- [12] Y. Aloimonos and C. M. Brown," The Relationship Between Optical Flow and Surface Orientation," Proc. 7th int. Conf. on Pattern Recognition, vol. 1, July 30 - Aug 2, 1984, pp. 542-543.

- [13] W. B. Thompson, K. M. Mutch and V. A. Berzins, "Analyzing Object Motion Based on Optical Flow," pp. 791-794. Proc. 7th Int. Conf. on Pattern Recognition, vol. 2, July 30 - Aug 2, 1984, pp. 791-794.
- [14] R. J. P. de Figueiredo, B. A. Bamieh, S. Fotedar, E. Hack, K. Trahan and C. Wu, "Demonstration of a Methodology for Machine Recognition and Attitude Determination of a 3D Object from a Single TV Picture Frame," Technical Report EE8520, Dept. of Electrical and Computer Engineering, Rice University, Houston, TX 77251-1892, Dec. 12, 1985.
- [15] B. A. Bamieh and R. J. P. de Figueiredo, "A Framework and Algorithms for Identification of Space Objects from Camera Data," Technical Report EE8407, Dept. of Electrical and Computer Engineering, Rice University, Houston, TX 77251, Aug. 29, 1984.
- [16] B. A. Bamieh and R. J. P. de Figueiredo, "A General Moment Invariants / Attitude - Graph Method for Three-Dimensional Object Recognition from a Single Image," IEEE Journal of Robotics and Automation, vol. RA-2, No. 1, Mar'86, pp. 31-41.
- [17] M. K. Hu, "Visual Pattern Recognition by Moment Invariants," IRE Trans. Information Theory, IT-8 (Feb'62), pp. 179-187.
- [18] F. L. Alt, "Digital Pattern Recognition by Moments," JACM, vol. 9 (1962), pp. 240-258.
- [19] B. L. Yen and T. S. Huang, "Solving for 3-D Motion Parameters of a Rigid Body by Vector Geometrical Approach : Uniqueness and Numerical Results," Proc. ICASSP, vol. 2, Mar. 19-21, 1984, San Diego, California, pp. 23.5.1-23.5.4.
- [20] B. L. Yen and T. S. Huang, "Determining 3-D Motion/Structure of a Rigid Body over 2 Frames using Correspondences of Straight Lines lying on Parallel Planes," Proc. 7th int. Conf. on Pattern Recognition, vol. 2, July 30 - Aug 2, 1984, pp. 781-783.
- [21] R. Y. Tsai, "A Versatile Camera Calibration Technique for High Accuracy 3D Machine Vision Metrology using Off-the-Shelf TV Cameras and Lenses," Industrial Machine Vision Project, Manufacturing Research, IBM, T. J. Watson Research Center, Yorktown Heights, NY 10598. Unpublished paper.
- [22] Y. Yakimovsky and R. Cunningham, "A System for Extracting Three Dimensional Measurements from a Stereo Pair of TV Cameras," Computer Graphics and Image Processing 7, pp. 195-210.
- [23] E. L. Hall, J. B. K. Tio, C. A. McPherson and F. A. Sadjadi, "Measuring Curved Surfaces for Robot Vision," Computer, vol. 15, No. 12, pp. 42-54, Dec'82.
- [24] R. Y. Tsai and T. S. Huang, "Estimating Three Dimensional Motion Parameters of a Rigid Planar Patch," IEEE Trans. ASSP, vol. ASSP-29, No.6, Dec'81, pp. 1147-1152.

- [25] R. Y. Tsai, T. S. Huang and W. -L. Zhu," Estimating Three - Dimensional Motion Parameters of a Rigid Planar Patch, II : Singular Value Decomposition," IEEE Trans. ASSP, vol. ASSP-30, No.4, Aug'82, pp. 525-534.
- [26] R. Y. Tsai, T. S. Huang and W. -L. Zhu," Estimating Three - Dimensional Motion Parameters of a Rigid Planar Patch, III : Finite Point Correspondences and the Three-View Problem," IEEE Trans. ASSP, vol. ASSP-32, No.2, April'84, pp. 213-220.
- [27] S. Ganapathy," Decomposition of Transformation Matrices for Robot Vision," Proc. Int. Conf. on Robotics, Mar 13-15, 1984.
- [28] R. Ray, J. Birk, and R. B. Kelley," Error Analysis of Surface Normals determined by Radiometry," IEEE Trans. PAMI, vol. 5, No. 6, Nov'83, pp. 631-645.
- [29] F. B. Hildebrand, Introduction to Numerical Analysis, McGraw-Hill Book Company, Inc., 1956.
- [30] C. L. Lawson, and R. J. Hanson, Solving Least Squares Problems, Prentice-Hall, Inc., 1974.
- [31] R. J. Vaccaro, and A. C. Kot," A Perturbation Theory for the Analysis of SVD-Based Algorithms," IEEE Int. Conf. on ASSP, vol.3, April 6-9, 1987, pp. 1613-1616.

### ***Additional Bibliography :***

- [1] R. J. P. de Figueiredo and V. Markandey, " Recovering Shape of Space Objects from the Shadowing Caused by Illumination," Tech. Rep. EE8608, Dept. Of Elect. and Comp. Engg., Rice Univ., Houston, TX 77251, Sept., 1986.
- [2] S. T. Barnard," Interpreting Perspective Images," Artificial Intelligence 21 (1983), pp. 435-462.
- [3] J. A. Webb and J. K. Aggarwal," Structure from Motion of Rigid and Jointed Objects," Artificial Intelligence 19 (1982), pp. 107-130.
- [4] A. N. Netravali and J. Salz," Algorithms for Estimation of Three-Dimensional Motion," AT&T Technical Journal, vol.64, No.2, Feb'85, pp. 335-346.
- [5] W. K. Chow and J. K. Aggarwal," Computer Analysis of Planar Curvilinear Moving Images," IEEE Trans. Computers, Feb'77, pp. 179-185.
- [6] S. R. Searle, Matrix Algebra Useful for Statistics, Wiley Series in Probability and Mathematical Statistics, 1982.

- [7] F. A. Graybill, Introduction to Matrices with Applications in Statistics, Wadsworth Publishing Company, 1969.
- [8] G. Strang, Linear Algebra and its Applications, Academic Press Inc., New York, 1976.
- [9] R. Y. Tsai, "Estimating 3-D Motion Parameters and Object Surface Structures from the Image Motion of Conic Arcs, I : Theoretical Basis," Proc. ICASSP, vol. 1, Apr. 14-16, 1983, Boston, Mass., pp. 122-125.