Integration of 3D Vision Based Structure Estimation and Visual Robot Control

Doctor of Philosophy in Electronic Engineering

By Naser Prljaca

Dublin City University
School of Electronic Engineering
Supervised By Dr. Hugh McCabe

May, 1995

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of PhD in Electronic Engineering is entirely my own and has not been taken from the work of others save and to the extent that such work has been cited and acnowledged within the text of my work.

Signed: Naser Prljaca

Naser Prljaca

Date: May 10 1995

Acknowledgements

I would like to acknowledge the financial support of the School of Electronic Engineering at Dublin City University which funded this research.

I would like to thank my supervisor Dr. Hugh McCabe for his time and support during the course of this work.

My wife Indira and my son Emir have been showing more then love, care and understanding. They have suffered the most. I thank them.

My parents from the war-torn Bosnia and Herzegovina kept encouraging me as they always did, when they needed encouragement the most. I thank them.

Abstract

Integration of 3D Vision Based Structure Estimation and Visual Robot Control

Enabling robot manipulators to manipulate and/or recognise arbitrarily placed 3D objects under sensory control is one of the key issues in robotics. Such robot sensors should be capable of providing 3D information about objects in order to accomplish the above mentioned tasks. Such robot sensors should also provide the means for multisensor or multimeasurement integration. Finally, such 3D information should be efficiently used for performing desired tasks.

This work develops a novel computational frame work for solving some of these problems. A vision (camera) sensor is used in conjunction with a robot manipulator, in the frame-work of active vision to estimate 3D structure (3D geometrical model) of a class of objects. Such information is used for the visual robot control, in the frame-work of model based vision.

One part of this dissertation is devoted to the system calibration. The camera and eye/hand calibration is presented. Several contributions are introduced in this part, intended to improve existing calibration procedures. This results in more efficient and accurate calibrations. Experimental results are presented.

Second part of this work is devoted to the methods of image processing and image representation. Methods for extracting and representing necessary image features comprising vision based measurements are given.

Third part of this dissertation is devoted to the 3D geometrical model reconstruction of a class of objects (polyhedral objects). A new technique for 3D model reconstruction from an image sequence is introduced. This algorithm estimates a 3D model of an object in terms of 3D straight-line segments (wire-frame model) by integrating pertinent information over an image sequence. The image sequence is obtained from a moving camera mounted on a robot arm. Experimental results are presented.

Fourth part of this dissertation is devoted to the robot visual control. A new visual control strategy is introduced. In particular, the necessary homogeneous transformation matrix for the robot gripper in order to grasp an arbitrarily placed 3D object is estimated. This problem is posed as a problem of 3D displacement (motion) estimation between the reference model of an object and the actual model of the object. Further, the basic algorithm is extended to handle multiple object manipulation and recognition. Experimental results are presented.

Table of Contents

Chapter 1: Introduction 1

1.1 Thesis preview 2

Chapter 2: Experimental Set-Up 4

2.1 Robot manipulator(arm) 4

2.2 Camera and frame store 7

2.3 Overall system integration and implementation 7

Chapter 3: Camera and Eye/Hand Calibration 9

3.1 Introduction 10

3.2 Relation to previous work 11

3.3 Camera model 13

3.4 Tsai's calibration method 16

3.5 Minimum variance estimator 18

3.6 Statistical approach to camera calibration 20

3.7 Evaluation of calibration accuracy 25

3.8 Experimental results 28

3.9 Eye/Hand calibration 31

3.10 Accuracy assessment 36

3.11 Experimental results 38

3.12 Conclusions 38

Chapter 4: Image Processing and Representation 40

4.1 Introduction 40

4.2 Edge extraction 41

4.3 Boundary detection 42

4.3.1 Contour following 46

4.3.2 Polygonal approximation 47

4.4 Data structuring 52

4.5 Geometric image rectification 54

4.6 Conclusions 55

Chapter 5: 3D Vision Based Structure Estimation 56

5.1 Introduction 57

5.2 3D structure reconstruction 58

5.2.1 Stereo vision 58

5.2.2 Structure from motion 71

5.3 3D model reconstruction by fusing data from image sequences 74

5.3.1 Relations to previous work 74

5.3.2 3D object model representation 81

5.3.3 Vision-based measurements 82

5.3.4 Prediction phase of tyracking 88

5.3.5 Matching phase of tracking 91

5.3.6 Update phase of tracking 95

5.4 Application of algorithm 97

5.5 Conclusions 113

Chapter 6: Visual Robot Control 116

6.1 Introduction 117

6.2 Relations to previous work 121

6.3 Control strategy 122

6.4 3D vision based reconstruction 125

6.5 3D displacement estimation 126

6.5.1 Rigidity constraints 128

6.5.2 Generating hypotheses 132

6.5.3 3D displacement estimation from point correspondences 133

6.5.4 Verifying hypotheses 137

6.6 Multiple objects manipulation and recognition 140

6.6.1 Multiple identical objects manipulation 140

6.6.2 Multiple different objects manipulation and recognition 141

6.7 Experimental results 142

6.8 Conclusions 160

Chapter 7: Conclusions and Suggestions 161

7.1 Review of research contributions 161

7.2 Further directions of research 164

References 165

Appendix A: Rotation parametrization

Appendix B: 3D displacement estimation from 3D point correspondences

Publications arising from this research

List of Figures

Figure 2.1.1, PUMA Robot Kinematic Structure 6

Figure 2.3.1, Experimental Set-Up 8

Figure 3.3.1, Camera Model 13

Figure 3.6.1, Arrangement for Scale Factor Determination 20

Figure 3.8.1, Calibration Pattern 28

Figure 3.9.1, Eye/Hand Transformation 32

Figure 3.9.2, Relationship Between Different Transformations and Coordinate Systems 32

Figure 4.2.1, Raw Image of Object 44

Figure 4.2.2, Edge Image of Object 45

Figure 4.3.1.1, Gradient Guidance 46

Figure 4.3.2.1, Polygonal Approximation 48

Figure 4.3.2.2, Maximal Separation Between Curve and Straight-Line Segment 49

Figure 4.3.2.3, Example of Polygonal Approximation 52

Figure 4.4.2, Supporting Data Structure 53

Figure 4.4.1, Image Structuring 54

Figure 5.2.1.1, Stereo Configuration 58

Figure 5.2.1.2, Stereo Errors 60

Figure 5.2.1.3, Angle between Object and Camera Axis 63

Figure 5.2.1.4, Relative Depth Uncertainty for Forward vs. Lateral Camera Translation 64

Figure 5.2.1.5, Epipolar Constraint 71
Figure 5.2.2.1, Optical Flow 72
Figure 5.3.1.2, System for 3D Structure Reconstruction 80
Figure 5.3.3.1, 2D Segment Noise Modelling 83
Figure 5.3.3.2, 3D Model Reconstruction 87
Figure 5.3.5.1, Image Search Window 94
Figure 5.4.1, Robot-Camera Kinematic Scheme 98
Figure 5.4.2, Raw Image of Object-Top View 101
Figure 5.4.3, Visually Reconstructed Model after 10 Frames 101
Figure 5.4.4, Length Estimation, Segment No.5 102
Figure 5.4.5, Length Variance, Segment No.5 102
Figure 5.4.6, Raw Image of Object, Top View 105
Figure 5.4.7, Visually Reconstructed Model after 10 Frames 105
Figure 5.4.8, Length Estimation, Segment No.2 106
Figure 5.4.9, Length Variance, Segment No.2 106
Figure 5.4.10, Raw Image of Object, Top View 108
Figure 5.4.11, Visually Reconstructed Model after 10 Frames 108
Figure 5.4.12, Length Estimation, Segment No.8 109
Figure 5.4.13, Length Variance, Segment No.8 109
Figure 5.4.14, Raw Image of Object, Top View 111
Figure 5.4.15, Visually Reconstructed Model after 10 Frames 111
Figure 5.4.16, Length Estimation, Segment No.8 112
Figure 5.4.17, Length Variance, Segment No.8 112
Figure 6.1, Position Based Look and Move Visual Control System 117
Figure 6.2, Image Based Look and Move Visual Control System 118
Figure 6.3, End-Effector Look and Move Visual Control System 120
Figure 6.5.1, Diagram of the Hypothesise and Verify Paradigm 127

Figure 6.3.1, Coordinate Transformations for Visual Robot Control 122

Figure 6.5.1.1, Rigidity Constraints 128

Figure 6.4.1, Eye in Hand Robot Configuration 125

Figure 6.6.1, Raw Image of Object 144

Figure 6.6.2 , 3D Model of Object, Projection on Camera..... 144

Figure 6.6.3 , Raw Image of Displaced Object 145

Figure 6.6.4 , 3D Model of Displaced Object, Projection on Camera..... 145

Figure 6.6.5 , Superposition of Reference and Observed Models,

Projection on Camera 146

Figure 6.6.6 , Superposition of Reference and Observed Models

(After Applying Computed Displacement), Projection on Camera 146

Figure 6.6.7, Raw Image of Object 149

Figure 6.6.8, 3D Model of Object, Projection on Camera..... 149

Figure 6.6.9, Raw Image of Displaced Object 150

Figure 6.6.10, 3D Model of Displaced Object, Projection on Camera..... 150

Figure 6.6.11, Superposition of Reference and Observed Models,

Projection on Camera 151

Figure 6.6.12, Superposition of Reference and Observed Models(After

Applying Computed Displacement), Projection on Camera 151

Figure 6.6.13, Raw Image of Object 154

Figure 6.6.14, 3D Model of Object, Projection on Camera..... 154

Figure 6.6.15, Raw Image of Displaced Object 155

Figure 6.6.16, 3D Model of Displaced Object, Projection on Camera..... 155

Figure 6.6.17, Superposition of Reference and Observed Models,

Projection on Camera 156

Figure 6.6.18, Superposition of Reference and Observed Models
(After Applying Computed Displacement), Projection on Camera 156
Figure 6.6.19, Raw Image of Objects 158
Figure 6.6.20, 3D Model of Objects, Projection on Camera 158
Figure 6.6.21, 3D Model of Objects After Recognising and Extracting
One Object, Projection on Camera 159
Figure 6.6.22, 3D of Objects After Recognising and Extracting Second
Object, Projection on Camera 159
Figure AA.1, Rotation Matrix (Appendix A)
Figure AA.2, Rotation Matrix (Appendix A)

List of Tables

Table 3.8.1, Intrinsic Parameters 30

Table 3.8.2, Fused Intrinsic Parameters 30

Table 3.8.3, Stereo Triangulation Test 31

Table 3.9.1, Eye/Hand Transformation Matrix 38

Table 3.9.2, Errors in Eye/Hand Transformation Matrix 39

Table 5.4.1, System Parameters 99

Table 5.4.2, Estimated and True Lengths of some Segments and Their Variances after First Stereo-Pair 103

Table 5.4.3, Estimated and True Lengths of some Segments and Their Variances after 10 Frames 103

Table 5.4.4 Estimated and True Lengths of some Segments and Their Variances after First Stereo Pair 104

Table 5.4.5 Estimated and True Lengths of some Segments and Their Variances after 10 Frames 104

Table 5.4.6 Estimated and True Lengths of some Segments and Their Variances after First Stereo Pair 107

Table 5.4.7 Estimated and True Lengths of some Segments and Their Variances after 10 Frames 107

Table 6.6.1, Initial Displacement Estimates 143

Table 6.6.2, Final Displacement Estimates 143

Table 6.6.3, Initial Displacement Estimates 147

Table 6.6.4, Final Displacement Estimates 148

Table 6.6.5, Initial Displacement Estimates 152

Table 6.6.6, Final Displacement Estimates 153

Chapter 1

Introduction

To enable robot manipulators to detect, manipulate and/or recognise arbitrarily placed 3D object under sensory control is one of the key issues in successful robotic applications. Such robot sensors should be capable of providing 3D information about objects in order to accomplish above mentioned tasks. Such sensors should also provide the means for multisensor or multimeasurement integration. Finally, such 3D information should be efficiently used for performing desired tasks.

This work develops a novel computational frame work for solving some of these problems. In this work a vision (camera) sensor is used to estimate 3D structure of objects within a class of objects. The objects are assumed to be well modelled as polyhedra. The camera is mounted on a robot manipulator to take advantage of its mobility. The moving camera permits us to resolve traditionally very difficult vision problems more easily. Using image sequences permits us to cope with inherently noisy image measurements, by fusing information over image sequences and consequently minimising impact of the measurement noise. The first goal of this study is to consider the problems of system calibration. The second goal of this study is to consider methods for 3D geometrical model reconstruction of objects based on the moving camera, i.e., the camera fixed on the robot arm. The third goal of this study is to use these 3D visual reconstructed models for the purpose of the robot visual control. The results we have achieved during the course of this work are presented in this thesis and a brief summary is given below.

1.1 Thesis preview

Chapter 2 presents experimental set-up used during the course of this work. It presents devices used and their overall integration.

Chapter 3 presents problem of camera and eye/hand calibration. Several contributions are introduced in this part. First, a modification of an existing camera calibration method is proposed. That permits us to determine the nonlinear camera model by using only closed-form (non iterative) computations. Second, a statistical approach to camera calibration is proposed. This approach permits us to compute a measure of accuracy for the obtained parameter values. This confidence measure allows to asses the camera calibration accuracy and to fuse multiple data in order to achieve better parameter estimates. Next, eye/hand calibration procedure is presented. Experimental results are given and discussed.

Chapter 4 presents methods for image processing and image representation needed for extracting necessary information for the purpose of this work. As a result of this processing stage, an image is represented as a list of straight-line segments. This image representation makes the subsequent stages efficient and feasible.

Chapter 5 introduces a new technique for 3D structure (3D geometrical model) estimation of objects within a class of objects, by using monocular image sequence and known camera motion. The technique is based on tracking line-segments over image sequences. The tracking process consists of prediction, matching and updating stages. These stages are handled in a Kalman filtering framework of covariance based prediction, matching and updating. The prediction stage of our tracking process does not use heuristics about motion in the image plane and applies to arbitrary camera motion. The prediction is based on assumptions about object structure (i.e. a rough knowledge of a

distance between camera and an object is assumed known and the depth extent of the object is small compared with the camera-object distance) for the initialisation phase, the rest of the tracking process is based on estimated object structure. The matching stage is based on the simple nearest-neighbour matching algorithm using the Mahalobonis (statistical) distance as a similarity measure. The updating stage is based on standard Kalman filter estimation algorithm. Experimental results from a camera mounted on a robot arm are presented to show the efficiency and accuracy of the proposed algorithm.

Chapter 6 proposes a novel control scheme for robot manipulator guidance based on visual information. The robot arm positions and orientates its gripper with respect to a 3D object, using vision data, in order to achieve desired grasping relation between the gripper and the object. The proposed control scheme consists of two distinct stages: 1) Teaching stage, in the teaching stage the robot reconstructs a 3D reference model of a presented unknown object within a class of objects, by integrating information from an image sequence obtained from a camera mounted on the robot arm (eye-in-hand configuration). The model is represented and stored as a set of 3D straight-line segments. The robot is also taught desired grasping relation. 2) Execution stage, in the execution stage the robot system reconstructs a 3D observed model of the arbitrarily placed 3D object and determines the necessary motion in order to achieve desired position and orientation of its gripper with respect to the object, by estimating the 3D displacement between reference and observed models. The displacement estimation is based on "hypothesise and verify" paradigm. Further, the basic algorithm is extended to handle multiple object manipulation and recognition The performance of the proposed algorithms has been tested on the real robot system, and the experimental results are presented .

Chapter 7 presents the conclusions of the study and offers suggestions for further study.

Chapter 2

Experimental Set-Up

For the purposes of this work the following equipment, devices and software have been used.

2.1 Robot manipulator (arm)

A robot manipulator consists of a sequence of links connected by independently driven joints. The robot arm has revolute and/or prismatic joints. The assembly of sequential links makes up a kinematic chain.

The primary objective of the robot manipulator is to carry out manipulation tasks in its workspace. The manipulation task is described as a sequence of positions and orientations of its gripper. The robot arm is controlled in the joint space and the task definition is usually given in the world space (Cartesian space). It is a central question to define the relation between the robot joint coordinates and world coordinates of its gripper.

This problem is denoted as robot kinematic modelling [Paul81], [Cry86], [Fu88]. This problem is twofold, it comprises the direct kinematic model and inverse kinematic model. Efficient way to derive the kinematic model is by using homogeneous coordinate transformations. By using Danavit-Hartenberg convention for homogenueous transformation definition, the direct kinematic model of the robot arm can be derived easily.

The direct kinematic model gives the relation between the position and orientation of the robot gripper and the robot joint coordinates, that is

$$\begin{bmatrix} x \\ y \\ z \\ \alpha \\ \beta \\ \gamma \end{bmatrix} = f(\theta_1,\ldots,\theta_n) \qquad \textbf{(2.1)}$$

where x, y, z are coordinates of the robot gripper position vector with respect to the world reference frame, α, β, γ are Euler angles of the gripper rotation (Appendix A) with respect to the world reference frame, $\theta_1,\ldots,\theta_n$ are joint coordinates of the robot and n is the number of joints (degrees of freedom).

The inverse kinematic model is inverse solution of Eq.(2.1), that is

$$\begin{bmatrix} \theta_1 \\ \theta_k \\ \theta_n \end{bmatrix} = f^{-1}(x,y,z,\alpha,\beta,\gamma) \qquad \textbf{(2.2)}$$

The inverse kinematic model is solved either in the closed-form or numerically, depending on the particular robot configuration.

In our research work, the Unimation PUMA robot manipulator is used. The PUMA robot manipulator has six revolute degrees of freedom. This means, that the PUMA robot arm can achieve the arbitrary position and orientation of its hand within its workspace. The PUMA robot manipulator is driven with six DC motors. Its measurement system consists of incremental shaft encoders mounted on each motor. Its overall work is controlled by MARK robot controller. The robot controller supports off-line programming of the robot tasks in the high-level procedural robot language VAL. VAL language offers motion instructions in the joint space and the world space as well. The kinematic structure of the PUMA robot is shown in Fig.2.1.

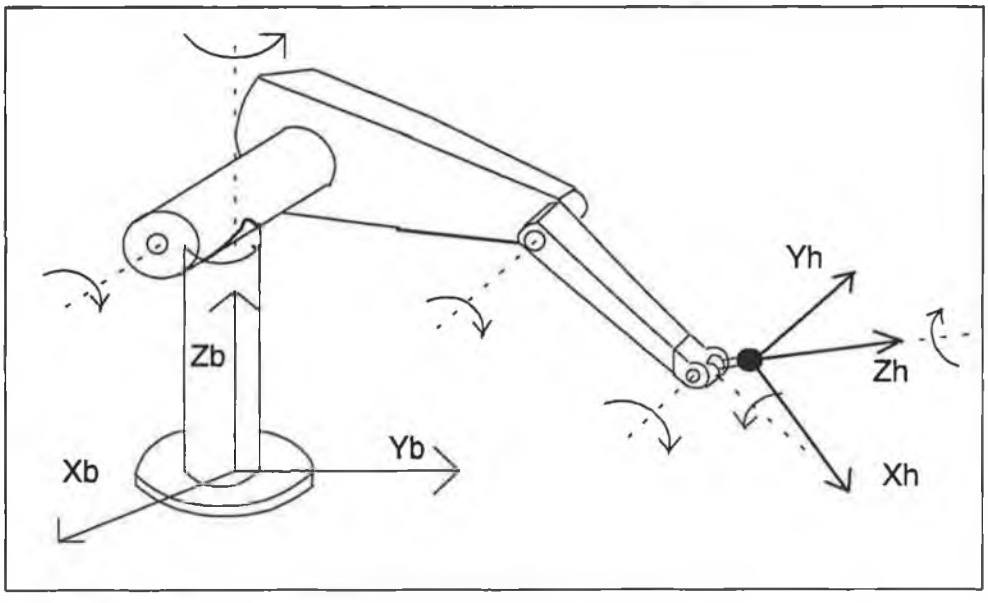

Figure 2.1.1 PUMA Robot Kinematic Structure

As can be seen from Fig.2.1.1 all motion specifications are given with respect to the robot world coordinate system (X_b, Y_b, Z_b) fixed at the robot base, the robot hand coordinate system (X_h, Y_h, Z_h) fixed at the robot hand and the joint space. Transformations between the world and joint space are done according to the inverse and direct robot kinematics. The robot controller provides the world coordinates of the hand and joint coordinates of the robot.

2.2 Camera and frame store

In this work the SONY-77ce CCD camera is used. The camera has 8.8 mm x 6.6 mm sensing area. The sensing cell size is 0.011 mm x 0.011mm. The camera provides CCIR video signal with 14.1875 MHz. We use a 16 mm lens C mounted. The Imaging Technology frame store is used. This frame store generates a digital image with 512 x 512 pixels spatial resolution. The intensity level is quantizied with 8 bits. The frame store sampling frequency is 10 MHz. The frame store is I/O mapped to the PC bus. For the basic access to the frame store the MAVIS software library is used.

2.3 Overall system integration and implementation

The camera is fixed on the robot hand by means of a custom made mechanical fixture. The frame store is fitted to a PC-386 computer. The PC is interfaced to the robot controller via a serial link (9600 bauds). The communication software was developed allowing to control the robot motion and to acquire the robot state information from the PC computer control software. An image of the Set-Up is shown in Fig.2.3.1.

The overall software implementation of the algorithms presented in this disertation was done by the author in C language on the PC computer. The source code is available from the author.

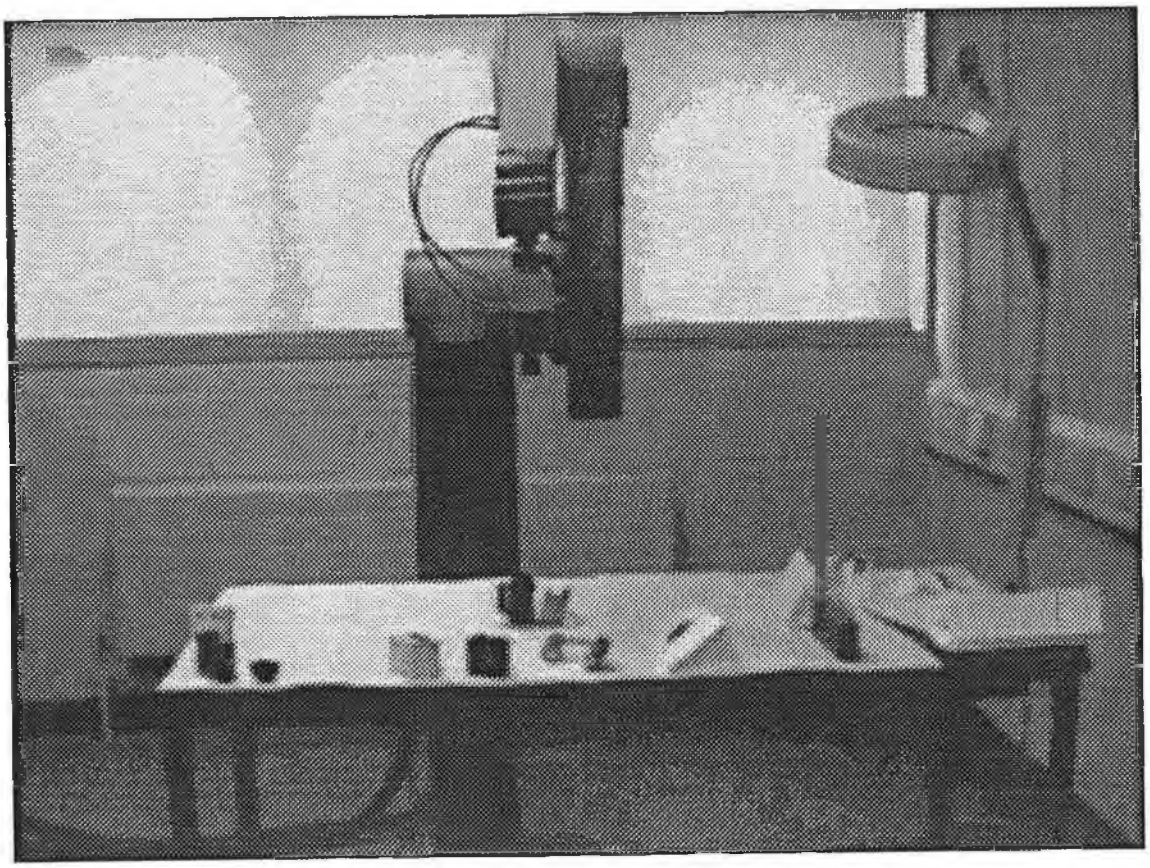

Figure 2.3.1 Experimental Set-Up

Chapter 3

Camera and Eye-Hand Calibration

In this chapter the problem of camera calibration and problem of eye/hand calibration are considered. Camera calibration parameters, intrinsic parameters consisting of the effective focal length, radial distortion coefficient, scale factor and image centre, extrinsic parameters consisting of the rotation matrix and translation vector, are determined and analysed. The main contributions of this work are following. Firstly, a modification of an existing camera calibration method is proposed, that permits determination of the nonlinear camera model by using only closed-form (non iterative) computations. Secondly, by using a statistical error model of calibration points extraction in digital images, we formulate the camera calibration problem in terms of optimal estimation framework. This statistical approach enables us to compute a measure of accuracy of obtained parameters. This confidence measure allows us to asses the camera calibration accuracy and to fuse multiple data in order to achieve better estimates. Thirdly, we introduce a new statistical measure, that can be used to evaluate the performance of the calibrated camera in terms of the 3D measurements. Further, eye/hand calibration is presented. The experimental results are given.

3.1 Introduction

Calibration of cameras is an important issue in the machine vision. Accurate calibration of cameras is especially crucial for measurement applications. The objective of camera calibration is to estimate the intrinsic camera parameters, which describe the image formation process and to estimate the extrinsic camera parameters, which describe the geometrical relation between the camera and the scene. The existing techniques for camera calibration can be classified into following categories:

- Direct nonlinear minimisation: In this group, equations that relate the parameters to be estimated with 3D coordinates of calibration points and their image plane projections are established. The search for unknown parameters consists of an iterative algorithm with an objective to minimise residual errors of some equations. The main advantage of these methods is ability to include different types of distortions. They give accurate solutions provided that the model is good and correct convergence has been achieved. The main disadvantage of these methods is the need for a nonlinear search for several unknowns, consequently, the proper convergence to the global minima can not be guaranteed unless a good initial guess is available, noise level is low and also the search procedure is well designed.

- Closed form solution: In this types of algorithms, parameter values are computed directly through non iterative algorithms. Since no iterations are required, the algorithms are fast and the solution is guaranteed. The main disadvantage is that the nonlinear camera distortions can not be included and the calibration points can not be coplanar.

- Two stage methods: The methods in this group [Tsa87], [Len87], [Wen92] involve a direct non iterative solution for most of the calibration parameters and some iterative solution for the other parameters. The main technique includes one presented by Tsai [Tsa87]. The radial alignment constraint is used to derive a closed-form solution for the most external parameters (the rotation matrix and two components of the

translation vector). Then an iterative search is used to determine the rest of the calibration parameters (the effective focal length, the radial distortion coefficient and the depth component in the translation vector). Lenz and Tsai [Len87] added two additional parameters, namely the image centre coordinates, which were considered to be known in [Tsa87] to the set of iteratively determined parameters. The advantages of their method are as follows:

1) The radial lens distortion is considered

2) The number of parameters to be estimated through iterations is small and a good initial guess is provided

3) The calibration points can be coplanar

3.2 Relation to previous work

In all of those above mentioned techniques either linear or nonlinear unweighted least-squares estimation is done. The unweighted least-squares estimation is optimal in the sense of minimum variance, provided that the equation residual is uncorrelated zero-mean random noise with equal variance. If residual components have different variances or they are correlated the unweighted least-squares will not give an optimal solution in the sense of minimum variance. The least-squares minimization makes sense only when error behaviours are well understood. Moreover, if the optimal solution that minimizes the cost function is found, this does not necessarily mean that the each component of the solution vector is reliable, depending on the noise level and observability of the measurement equations. In case of camera calibration this reflects interaction between the intrinsic and extrinsic camera parameters. In all of those techniques there is no measure of accuracy of final camera parameters. In this chapter, we address the problem of statistical analysis of camera calibration parameters and we give a new method to determine some of the camera parameters. This approach permits us to use the weighted least-squares

estimation of camera parameters and to achieve the minimum variance estimate. More importantly it allows to compute confidence bounds of estimated camera parameters. This is important for the following reasons

- It allows us to asses the quality of the camera calibration
- It detects unreliable and unstable camera calibrations
- It allows us to fuse multiple data in order to improve calibrated parameters

For this purpose we use the two-stage method proposed by Tsai [Tsa87] with the following differences

- In [Tsa87] the image centre coordinates were assumed to be known in advance. Later on in [Len87] an iterative method of determining the image centre coordinates was proposed. We propose a closed-form solution for the image centre coordinates.
- In [Tsa87] the iterative search is employed in order to find some parameters. We propose closed-form solutions for all camera parameters.
- By computing noise properties and using minimum variance estimation, we improve the parameter accuracy.
- The confidence bounds of all parameters are computed.
- The multiple data for intrinsic camera parameters are fused.

3.3 Camera model

This section describes the camera model and defines the calibration parameters. The camera model is basically the same as that used by a lot of calibration techniques mentioned above. The camera model is based on perspective projection and assumes that the lens distortion is radial one. The first model is a pinhole camera model that neglects all optical distortions. The second model takes into account the radial lens distortion as the most dominant one. Fig.3.3.1 illustrates the basic geometry of the camera model.

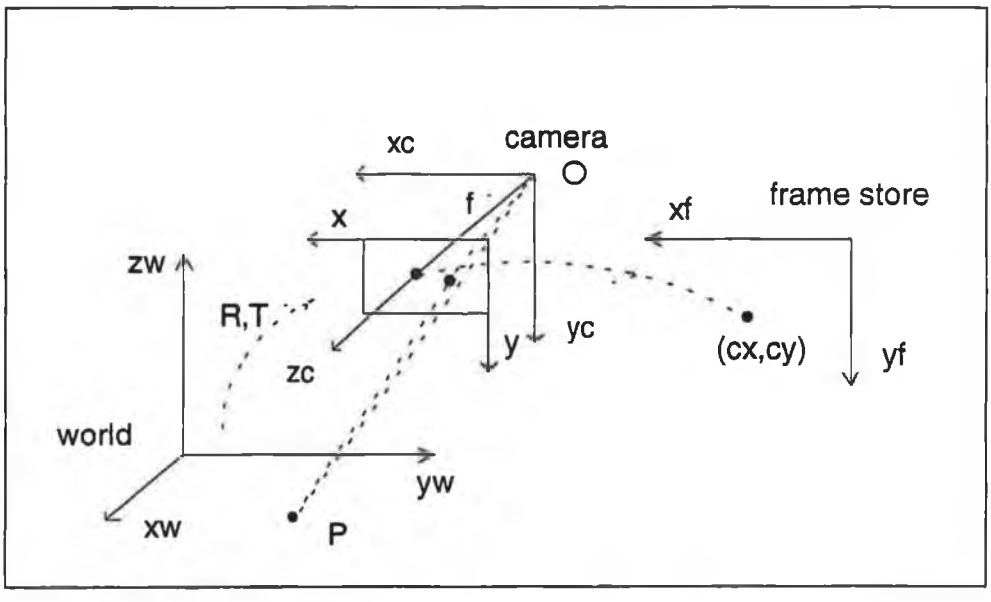

Figure 3.3.1 Camera Model

(x_w, y_w, z_w) are the 3D coordinates of the calibration point P in the 3D world coordinate system. (x_c, y_c, z_c) are the 3D coordinates of the same point in the camera centred coordinate system. The camera coordinate system is centred at point O, the optical centre, with z_c axis the same as the optical axis. (x, y) is the image coordinate system centred at the intersection of the optical axis z_c and the front image plane and parallel to x_c and y_c axes of the camera coordinate system. f is the distance between front image plane and the optical centre. (x_u, y_u) are the image coordinates of P if a perfect pinhole camera model is used. (x_d, y_d) are actual (distorted) image coordinates of P which differ from (x_u, y_u)

due to lens distortions. The image plane is sampled and stored in the computer (frame) memory. (x_f, y_f) are pixel coordinates of the discrete image in terms of rows and columns. Additional parameters must be introduced and calibrated that relate the image coordinates in the front image plane to the computer image coordinate system. Thus, the overall transformation from (x, y, z) to (x_f, y_f) must be established. This overall transformation is given as follows:

1) Rigid body transformation from the object world coordinate system (x_W, y_W, z_W) to the camera coordinate system (x_C, y_C, z_C)

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} = R \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} + T \qquad \textbf{(3.3.1)}$$

where R is a (3x3) rotation matrix defining the camera orientation, and T is a translation vector defining the camera position. Extrinsic parameters to be calibrated are R and T.

2) The transformation from the 3D camera coordinate system (x_C, y_C, z_C) to ideal (undistorted) image coordinates (x_u, y_u) using perspective projection with pinhole camera geometry

$$x_u = f \frac{x_c}{z_c}, \quad y_u = f \frac{y_c}{z_c} \qquad \textbf{(3.3.2)}$$

where f is the effective focal length of the lens. The parameter to be calibrated is f.

3) The transformation between ideal image coordinates and real (distorted) image coordinates is given by

$$x_u = \frac{x_d}{(1+k(x_d^2+y_d^2))}, \quad y_u = \frac{y_d}{(1+k(x_d^2+y_d^2))} \tag{3.3.3}$$

where k is the radial lens distortion coefficient. The parameter to be calibrated is k.

4) The transformation between real image coordinates (x_d, y_d) and computer image (frame buffer) coordinates (x_f, y_f)

$$x_d = s_x dx(x_f - c_x), \quad y_d = s_y dy(y_f - c_y) \tag{3.3.4}$$

where

(x_f, y_f) are row and column numbers of image pixels in computer memory

(c_x, c_y) are row and column numbers of the centre of computer frame memory

dx is centre to centre distance between adjacent sensor elements in x (scan line) direction

dy *is* centre to centre distance between the sensor elements on the image plane along y axis

s_x is scale factor along x direction

s_y is scale factor along y direction

5) To transform between computer image coordinates (in terms of rows and columns in frame buffer) and real image coordinates, obviously distances between the two adjacent sensor elements in both directions (dx, dy) need be used. These distances are available from manufacturers with submicron accuracy for CCD cameras. In case of solid state CCD cameras, there is one to one mapping between rows in the computer image and video lines from cameras. This implies that the vertical scale factor s_y is equal to one. The situation in the x direction is different. A video line is generated by scanning N_c number of sensor elements in the x direction. This line is sampled by the computer into N_f samples. It follows that the horizontal scale factor s_x is roughly given by

$$s_x = \frac{N_c}{N_f} \tag{3.3.5}$$

However, this value is only a rough one due to hardware timing mismatch between image acquisition and camera scanning hardware. It is desirable to calibrate the s_x value. The parameters to be calibrated are the image centre (c_x, c_y) and the horizontal scale factor s_x.

By substituting from Eq.(3.3.1) to Eq.(3.3.5), the complete camera model is given by [Tsa87]

$$\begin{aligned} f\frac{r_1x+r_2y+r_3z+t_x}{r_7x+r_8y+r_9z+t_z} &= \frac{s_x dx(x_f-c_x)}{(1+kD^2)} \\ f\frac{r_4x+r_5y+r_6z+t_y}{r_7x+r_8y+r_8z+t_z} &= \frac{dy(y_f-c_y)}{(1+kD^2)} \\ D^2 &= [sdx(x_f-c_x)]^2+[dy(y_f-c_y)]^2 \end{aligned} \tag{3.3.6}$$

3.4 Tsai's calibration method

The calibration problem is stated in the following terms. Given a sufficient number of calibration points (x_i, y_i, z_i) and their corresponding pixel locations (x_{fi}, y_{fi}), estimate the set of external (R, T) and internal (f, k, c_x, c_y, s_x) parameters from the camera model given by Eq.(3.3.6). The very efficient solution for the camera parameters, provided that the image centre (c_x, c_y) and the horizontal scale factor s_x are known in advance, is given in Tsai's two stage calibration procedure [Tsa87]. The first stage uses the radial alignment constraint. This constraint follows from Eq.(3.3.6) by dividing the first two equations with each other. Calibration points are assumed to be on a common plane. The (x, y, z) coordinate system can be chosen so that $z = 0$ for all points. By doing so we have

$$y_{di}x_i r_1' + y_{di}y_i r_2' + y_{di}t_x' - x_{di}x_i r_4' - x_{di}y_i r_5' = x_{di} \tag{3.4.1}$$

where

$$r_1' = \frac{r_1}{t_y},\ r_2' = \frac{r_2}{t_y},\ r_4' = \frac{r_4}{t_y},\ r_5' = \frac{r_5}{t_y},\ t_x' = \frac{t_x}{t_y} \tag{3.4.2}$$

With the number of calibration points larger than five, an overdetermined system of linear equations can be established and solved for five unknowns r_1', r_2', r_4', r_5', t_x'. By using properties of rotation matrices, we have [Tsa87]

$$t_y^2 = \frac{S_r - [S_r^2 - 4(r_1'r_5' - r_4'r_2')^2]^{0.5}}{2(r_1'r_5' - r_4'r_2')^2},\quad S_r = r_1'^2 + r_2'^2 + r_4'^2 + r_4'^2 \tag{3.4.3}$$

Having determined t_y from Eq.(3.4.3), r_1, r_2, r_4, r_5 and t_x are determined from Eq.(3.4.2). The overall rotation matrix is given by

$$R = \begin{bmatrix} r_1 & r_2 & \sqrt{1-r_1^2-r_2^2} \\ r_4 & r_5 & \sqrt{1-r_4^2-r_5^2} \\ r_7 & r_8 & r_9 \end{bmatrix} \tag{3.4.4}$$

where r_7, r_8, r_9 are determined from the outer product of the first rows using the orthonormal property of the rotation matrix. In the second stage, with R, t_x and t_y determined, the Eq.(3.3.6) becomes a nonlinear equation with f, k and t_z as unknowns. In [Tsa87] the nonlinear least-squares criterion was established and solved by an iterative method, completing the calibration of camera parameters.

3.5 Minimum variance estimator

Let a set of linear measurement equations be given by

$$Z = Hx + \varepsilon, \quad E(\varepsilon) = 0, \quad E(\varepsilon\varepsilon^T) = W \tag{3.5.1}$$

where x is a parameter vector, Z is a measurement vector, ε is a noise vector with zero-mean and covariance matrix W. The minimum variance estimate [Gel74], [Bar88] of the parameter vector x is defined as the minimizer of the cost function given by

$$\min_x \ (Z - Hx)^T W^{-1} (Z - Hx) \tag{3.5.2}$$

Let us consider in general a nonlinear measurement equation in the form

$$f(x,a) = 0, \quad a^* = a + \varepsilon, \quad E(\varepsilon) = 0, \quad E(\varepsilon\varepsilon^T) = R \tag{3.5.3}$$

where a^* is an observation vector corrupted with additive noise ε, x is a parameter vector and R is variance of the measurement noise ε. Suppose that we have a good estimate x^* of the true parameter vector. We can use the idea of linearization and expand $f(x, a)$ in the vicinity (x^*, a^*) in the following manner

$$f(x,a) = 0 \ \approx \ f(x^*,a^*) + \frac{\partial f(x^*,a^*)}{\partial x}(x - x^*) + \frac{\partial f(x^*,a^*)}{\partial a}(a - a^*) \tag{3.5.4}$$

This equation can be rewritten as

$$
\begin{aligned}
z &= hx + m\varepsilon \\
z &= -f(x^*, a^*) + \frac{\partial f(x^*, a^*)}{\partial x} x^* \\
h &= \frac{\partial f(x^*, a^*)}{\partial x} \\
m &= -\frac{\partial f(x^*, a^*)}{\partial a}
\end{aligned}
\tag{3.5.5}
$$

This equation now appears as a linear measurement equations for the parameter vector x with a new additive noise $m\varepsilon$. We can compute second order statistics of the new measurement noise as follows

$$v = m\varepsilon, \quad E(v) = 0, \quad W = E(vv^T) = mRm^T \tag{3.5.6}$$

Having a set of nonlinear measurement equations $f_i(x, a) = 0$, having an approximate solution of the parameter vector x^*, performing above derived linearization and assuming that the measurement noise is uncorrelated, we can define the minimum variance estimator of the parameter vector x as

$$\min_x \quad \sum_{i=1}^{n} (z_i - h_i x)^T w_i^{-1} (z_i - h_i x) \tag{3.5.7}$$

where z_i, h_i and w_i are determined according to Eq.(3.5.5) and Eq.(3.5.6)

The minimum variance estimate is given by

$$\hat{x} = S^{-1}B, \quad Cov(\hat{x}) = S^{-1}, \quad S = \sum_{i=1}^{n} h_i^T w_i^{-1} h_i, \quad B = \sum_{i=1}^{n} h_i^T w_i^{-1} z_i \tag{3.5.8}$$

The minimised residual is given by

$$\hat{J} = \sum_{i=1}^{n} (z_i - h_i \hat{x})^T w_i^{-1} (z_i - h_i \hat{x}) \qquad \textbf{(3.5.9)}$$

The minimised residual has a χ^2 distribution with n-n_x degrees of freedom up to the first order approximation and assuming Gaussian noise. n is the number of equations and n_x is the size of the parameter vector x.

3.6 Statistical approach to camera calibration

The horizontal scale factor s_x can be easily, accurately and independently estimated by using a square calibration pattern in front of the camera, approximately parallel with the image plane and approximately centred at the image centre. This arrangement is shown in Fig.3.6.1.

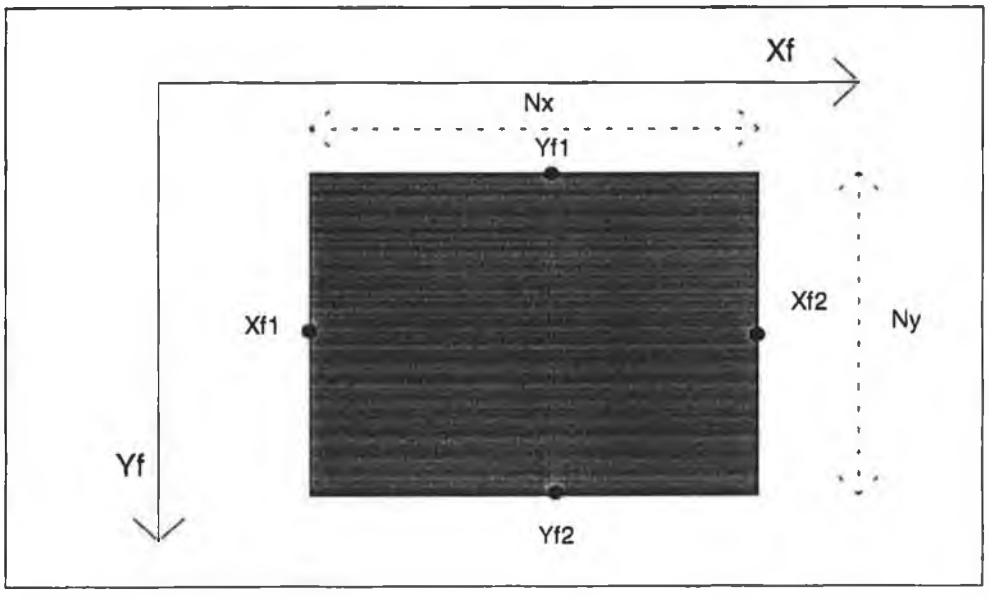

Figure 3.6.1 Arrangement for Scale Factor Determination

From Fig.3.6.1 and Eq.(3.3.4) follows

$$\begin{aligned} x_{d1} &= s_x dx(x_{f1} - c_x), \quad & x_{d2} &= s_x dx(x_{f2} - c_x) \\ y_{d1} &= dy(y_{f1} - c_y), \quad & y_{d2} &= dy(y_{f2} - c_y) \end{aligned} \qquad \textbf{(3.6.1)}$$

by manipulating Eq.(3.6.1) we have

$$\frac{x_{d2}-x_{d1}}{y_{d2}-y_{d1}}=s_x\frac{dx}{dy}\frac{(x_{f2}-x_{f1})}{(y_{f2}-y_{f1})} \tag{3.6.2}$$

taking into account above mentioned geometric assumptions, the left hand side of the Eq.(3.6.2) becomes one, leading to the expression

$$s=\frac{dy}{dx}\frac{N_y}{N_x} \tag{3.6.3}$$

where N_y and N_x are the horizontal and vertical lengths of the square image in pixels. The variance of s_x is computed up to the first-order approximation as

$$\sigma_s^2=\begin{bmatrix}\frac{\partial s}{\partial N_x} & \frac{\partial s}{\partial N_y}\end{bmatrix}\begin{bmatrix}Var(N_x) & 0\\ 0 & Var(N_y)\end{bmatrix}\begin{bmatrix}\frac{\partial s}{\partial N_x}\\ \frac{\partial s}{\partial N_y}\end{bmatrix} \tag{3.6.4}$$

where $Var(N_x)$ and $Var(N_y)$ are variances of N_x and N_y respectively. They are of order of a few pixels. Multiple measurements of s_x can be taken and fused according to the minimum variance estimation (Section 3.5).

We arrange the radial alignment constraint Eq.(3.4.1), in such a way which permits us to determine the image centre coordinates in addition to parameters originally determined from the radial alignment constraint [Tsa87]. This arrangement has the form

$$dyy_{fi}x_i r_1' + dyy_{fi}y_i r_2' + dyy_{fi}t_x' + dxx_{fi}x_i r_4' + dxx_{fi}y_i r_5' - dyx_i p_1 - dyy_i p_2 - dyp_3 = dxx_i \tag{3.6.5}$$

where

$$p_1 = c_y r_1' + \frac{dx}{dy}c_x r_4', \quad p_2 = c_y r_2' + \frac{dx}{dy}c_x r_5', \quad p_3 = c_y t_x' + \frac{dx}{dy}c_x \tag{3.6.6}$$

The equation Eq.(3.6.5) is a linear equation with eight unknowns $(r_1', r_2', r_4', r_5', t_x', p_1, p_2, p_3)$. Minimum of eight calibration points and their corresponding image points are necessary to solve for eight unknowns. Because of noise more points are desirable. An overdetermined system of linear equations can be formed and solved. Having solution for the intermediate parameters (p_1, p_2, p_3) permits us to solve for the image centre coordinates by solving three linear equations with two unknowns Eq.(3.6.6).

We define the minimum variance estimator of the parameters contained in the radial alignment constraint in the following two-step manner

- Solve the overdetermined system of linear equations Eq.(3.6.5), using ordinary least-squares (unweighted least squares) method, for the solution vector $x^* = (r_1', r_2', r_4', r_5', t_x', p_1, p_2, p_3)^*$

- Linearizing measurement equations Eq.(3.6.5) around obtained solution vector x^* and measurement vectors (x_{fi}, y_{fi}) (the measurement vector is the image pixel location with its corresponding extraction noise) and applying the procedure from the section 3.5, we formulate and solve the minimum variance estimation problem (weighted least-squares). That gives the minimum variance estimate and corresponding covariance matrix of the solution vector

$$\hat{x} = S^{-1}B, \quad Cov(\hat{x}) = S^{-1}, \quad S = \sum_{i=1}^{n} h_i^T w_i^{-1} h_i, \quad B = \sum_{i=1}^{n} h_i^T w_i^{-1} z_i$$

- Check if the estimate is consistent, by testing that the minimised residual

$$\hat{J} = \sum_{i=1}^{n} (z_i - h_i \hat{x})^T w_i^{-1} (z_i - h_i \hat{x})$$

is a χ^2 distributed variable with (n-8) degrees of freedom

- Solve for the image centre coordinates (c_x, c_y) from Eq.(3.6.6) by applying the procedure from the section 3.5, that gives the minimum variance estimates and their covariance matrix

- Solve for the elements of the R matrix and t_x and t_y components of the T vector according to Eq.(3.4.2), Eq.(3.4.3) and Eq.(3.4.4). Variances of each element in the matrix R and vector T are computed up to the first-order approximation as

$$\sigma_e^2 = JWJ^T$$

where J is Jacobian of an element with respect to intermediate parameters given by

$$J = \left[\frac{\partial e}{\partial r_1'} \quad \frac{\partial e}{\partial r_2'} \quad \frac{\partial e}{\partial r_4'} \quad \frac{\partial e}{\partial r_5'} \quad \frac{\partial e}{\partial t_x'} \right]$$

and W is the covariance matrix of the intermediate parameters obtained above

By substituting these found values for R, t_x and t_y into Eq.(3.3.6), we have a nonlinear equation relating the rest of camera parameters (f, k, t_z). By introducing intermediate parameters, the nonlinear Eq.(3.3.6) can be rewritten as follows

$$[a,\ b,\ c]X = e, \qquad X = [f,\ t_z,\ kf] \tag{3.6.7}$$

which has a linear form with respect to intermediate parameters X.

We define the minimum variance estimator of the parameters f, k and t_z in the following two-step manner

- Solve the overdetermined system of linear equations Eq.(3.6.7), using ordinary least-squares (unweighted least-squares) method, for the solution vector $x^* = (f,\ t_z,\ kf)$

- Linearizing measurement equations Eq.(3.6.7) around obtained solution vector x^* and measurement vector $(x_{fi},\ y_{fi})$ (the measurement vector is the image pixel locations with its corresponding extraction noise) and applying the procedure from the section 3.5, we formulate and solve the minimum variance estimation problem (weighted least squares). That gives the minimum variance estimate and corresponding covariance matrix of the solution vector

$$\hat{x} = S^{-1}B, \quad Cov(\hat{x}) = S^{-1}, \quad S = \sum_{i=1}^{n} h_i^T w_i^{-1} h_i, \quad B = \sum_{i=1}^{n} h_i^T w_i^{-1} z_i$$

- Check if the estimate is consistent, by testing that the minimized residual

$$\hat{J} = \sum_{i=1}^{n} (z_i - h_i\hat{x})^T w_i^{-1} (z_i - h_i\hat{x})$$

is a χ^2 distributed variable with (n-3) degrees of freedom

This completes the determination of all camera parameters and their variances. Thus, measurement uncertainties introduce calibration inaccuracies whose magnitudes depend on the experimental conditions. Above presented procedure give the relationship between magnitudes of random errors and calibration inaccuracies. Thereby, the validity of calibrations can be assessed, unreliable and unstable calibrations can be detected and data fusion can be performed. Since, the intrinsic parameters are independent of extrinsic parameters, different data with their covariances obtained by moving camera to different position can be fused, giving improved estimate in the minimum variance sense of the intrinsic parameters. At different positions of the camera, we estimate the intrinsic camera

parameters and their covariances. The multiple data are fused according the minimum variance criterion (Section 3.5).

3.7 Evaluation of calibration accuracy

Having camera parameters calibrated and their confidence bounds available, it is still difficult to have insight about the performance of the calibrated camera in the 3D measurement application. Classical criteria to assess the 3D measurement performance of the calibrated camera is the accuracy of 3D coordinate measurements obtained through stereo triangulation using the calibrated camera. One of them is the average norm of 3D positional difference between measured and true values of 3D point coordinates, it is referred to as *RMS* (root mean square) measure. *RMS* criteria is given by

$$RMS = \sqrt{\sum_{i=1}^{n} \frac{(X_{mi} - X_{ti})^2}{n}} \qquad \textbf{(3.7.1)}$$

where X_{mi} and X_{ti} are true and measured 3D point coordinates. This criteria depends on the actual geometry of the stereo set-up and the camera parameters as well, by adjusting the geometry of the stereo set-up a good measure of this criteria can be achieved by having poorly calibrated camera. For that reason, we introduce a statistical criteria for assessing the calibrated parameters, which is insensitive to the actual stereo set-up.
The geometry of the stereo set-up is presented in Fig.3.7.1.

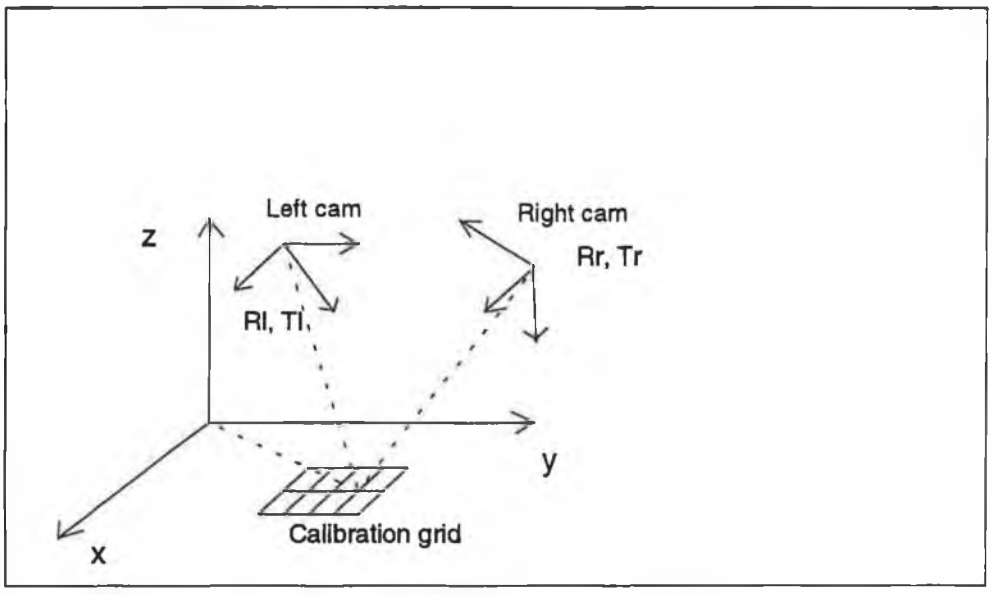

Figure 3.7.1 Geometry of Stereo Set-up

From the figure we have the following equations for computing the 3D coordinates of the points in the world coordinate system, in terms of the left and right calibrated camera

$$x_u^l = f^l \frac{r_1^l x + r_2^l y + r_3^l z + t_x^l}{r_7^l x + r_8^l y + r_9^l z + t_z^l}$$

$$y_u^l = f^l \frac{r_4^l x + r_5^l y + r_6^l z + t_y^l}{r_7^l x + r_8^l y + r_9^l z + t_z^l}$$

$$x_u^r = f^l \frac{r_1^r x + r_2^r y + r_3^r z + t_x^r}{r_7^r x + r_8^r y + r_9^r z + t_z^r}$$

$$y_u^r = f^r \frac{r_4^r x + r_5^r y + r_6^r z + t_y^r}{r_7^r x + r_8^r y + r_9^r z + t_z^r}$$

By locating the point images in the left and right image, and correcting distortion according to the camera model, these equations can be solved for unknown *3D* coordinates of the point. The minimum variance estimator (section 3.5) is used to estimate 3D coordinates. That gives the solution vector and its associated covariance

matrix, which takes into account the image digitisation noise and treats the camera parameters as constants. By doing so, we obtain

$$\hat{X}_i = \begin{bmatrix} \hat{x}_i \\ \hat{y}_i \\ \hat{z}_i \end{bmatrix}, \qquad Cov(\hat{X}_i) = W_i$$

We use the χ^2 statistical test to asses the accuracy of the parameters in the following manner. Let us define

$$q_i = (\hat{X}_i - X_i)^T W_i^{-1} (\hat{X}_i - X_i)$$

q_i is a χ^2 distributed variable with 3 degrees of freedom, X_i is the true position vector of the point. From the *p.d.f* of χ^2 distribution it is easy to determine a confidence interval, such that

$$q_i \leq \chi^2_{(P_r)}$$

is satisfied with desired probability *Pr* (we usually use the confidence interval of 95%, which gives the threshold value of 3.84). From this, it is obvious if the camera parameters are well calibrated, the image noise is well described, regardless of the actual stereo set-up and image resolution, the above test will be satisfied. Based on this we introduce the measure of accuracy as follows

$$Ca = \frac{P_e}{P_a} \qquad \textbf{(3.7.2)}$$

where *Pe* is the probability of the χ^2 distribution equal to *Pr*, and *Pa* is actual probability computed over all test points. It is obvious that this measure is close to one for well calibrated cameras. In ideal case for perfectly calibrated cameras it is equal to one and for well calibrated cameras it is slightly greater than one.

3.8 Experimental results

1) Description of calibration set-up

The set-up used in this calibration experiments consists of a planar calibration pattern. The images of the pattern were taken by a *CCD* camera mounted on a robot arm. The digitizer gives 512 x 512 pixels and 8 bits/pixel. The calibration pattern consists of twenty five squares and was custom made. The calibration pattern is shown in Fig.(3.8.1).

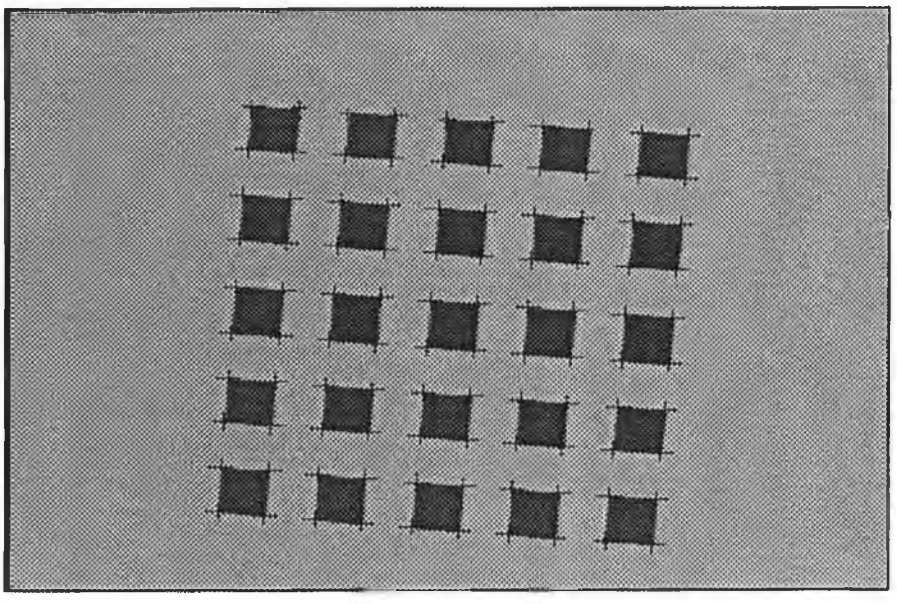

Figure 3.8.1 Camera Calibration Pattern

2) Image processing

The 3D calibration points are chosen to be corners of squares on the calibration plate with z component set to zero. Their spatial positions $(x_{wi}, y_{wi}, z_{wi}=0)$ are reliably established owing to the accuracy of the calibration pattern. The corresponding image-point locations are estimated with a subpixel accuracy using the following procedure. The image of the calibration pattern is thresholded by an automatic threshold selection procedure. The edges of the squares are detected by using a simple binary edge detection

algorithm. Edge points are linked by a contour following algorithm. The straight-lines are fitted to the contour points corresponding to the edges of the squares of the calibration pattern. The corner points are determined by computing intersections of the corresponding computed lines. Fig.3.8.1 shows detected corners. The overall procedure including the correspondence between 2D and 3D corners is fully automatic. One half of the points are used for the calibration and the second half of points are used for the accuracy assessment.

3) Image noise characterisation

The calibration points extraction noise is determined by corresponding digitisation noise. Each pixel is a rectangle with a side-length a in the x direction and b in the y direction. Since the uniform round-off noise with a spacing equal to d has a variance $d^2/12$. The covariance matrix of the image point is given by

$$Cov\begin{bmatrix} x_f \\ y_f \end{bmatrix} = \begin{bmatrix} \frac{a^2}{12} & 0 \\ 0 & \frac{b^2}{12} \end{bmatrix}, \quad a = s_x \frac{dx}{dy}, \quad b = 1$$

From the data-sheet of our camera dx=0.011 mm and dy=0.011 mm. The value of horizontal scale factor s_x=1.4875 is determined independently as described previously. This gives

$$Cov\begin{bmatrix} x_f \\ y_f \end{bmatrix} = \begin{bmatrix} \frac{1.4875^2}{12} & 0 \\ 0 & \frac{1}{12} \end{bmatrix}$$

4) Experiment 1

We used the camera mounted on a robot arm. First, the horizontal scale factor was calibrated. Then, the robot arm was moved to various locations and orientations with

respect to calibration pattern. At each position the camera parameters and their variances were computed. Table 3.8.1 shows the estimated intrinsic parameters of the camera and their standard deviations for five different locations of the camera. Table 3.8.2 shows the fused values of the intrinsic camera parameters and their variances. It is important to note, that each time during the experiments unreliable and unstable calibrations were detected.

n	s_x	*Cx*	*Cy*	*f*	*k*	*Dev* s_x	*Dev Cx*	*Dev Cy*	*Dev f*	*Dev k*
1	*1.487*	*244.1*	*267.7*	*16.13*	*-0.0013*	*0.004*	*18.3*	*14.7*	*0.22*	*0.00040*
2	*1.487*	*240.2*	*262.9*	*16.15*	*-0.0013*	*0.004*	*18.2*	*15.2*	*0.23*	*0.00038*
3	*1.487*	*240.7*	*253.5*	*16.22*	*-0.0014*	*0.004*	*18.0*	*15.5*	*0.24*	*0.00046*
4	*1.487*	*238.4*	*259.5*	*16.14*	*-0.0013*	*0.004*	*19.9*	*15.8*	*0.21*	*0.00030*
5	*1.487*	*247.5*	*280.0*	*16.15*	*-0.0013*	*0.004*	*20.2*	*16.5*	*0.21*	*0.00029*

Table 3.8.1 Intrinsic Parameters

s_x	*Cx*	*Cy*	*f*	*k*	*Dev* s_x	*Dev Cx*	*Dev Cy*	*Dev f*	*Dev k*
1.487	*242.0*	*264.5*	*16.16*	*-0.001*	*0.0017*	*6.9*	*8.4*	*0.101*	*0.00015*

Table 3.8.2 Fused Intrinsic Parameters

5) Experiment 2

The same camera mounted on a robot arm looking at the calibration plate is used. At one position, the intrinsic and extrinsic parameters are estimated by using calibration points. Then the robot arm is moved to another position looking at the calibration plate, and the intrinsic and extrinsic parameters are estimated again by using calibration points. This makes a stereo set-up. After this, intrinsic and extrinsic parameters of the camera at two position and sensed test points in the two images are used to determine 3D coordinates of the test points by using stereo triangulation. Since we know the positions of the test points in the world coordinate system, we can asses the calibration accuracy by comparing true and computed point coordinates. The values of the camera parameters

and results of the triangulation test are given in Table 3.8.3. The rotation matrix is parametrized by Euler angles (Appendix A).

Parameters	Values - Position 1	Values - Position 2
Rotation matrix (roll, pitch, yaw angles)	α=-3.1329 β=-2.412 γ=43.336	α=-1.547 β=7.922 γ=43.799
Translation vector	t_x=-139.215 t_y=-84.65 t_z=289.573	t_x=-107.8 t_y=-77.96 t_z=311.3
Focal length	f=16.20	f=16.12
Image centre	Cx= 243 , Cy=258	Cx=245, Cy=265
Scale factor	s_x=1.487	s_x=1.487
Distortion coeff.	k=-0.0013	k=-0.0014

RMS measure	*RMS*= 0.9 mm
Ca measure	*Ca* = 1.09

Table 3.8.3 Stereo Triangulation Test

The values of the error measures *RMS* (Eq.3.7.1) and *Ca* (Eq.3.7.2) given in Table 3.8.3 show well calibrated camera. Plenty of experiments were performed giving consistent results.

3. 9 Eye/Hand calibration

In order for a robot to use a robot hand mounted camera to estimate the 3D position and orientation of an object, it is necessary to know the relative position and orientation between the camera and the hand (eye/hand). This requires us to determine the relative rotation and translation (homogeneous transformation) between two coordinate frames, one centred at the camera optical centre, and the other at the robot hand. This is shown in Fig.3.9.1.

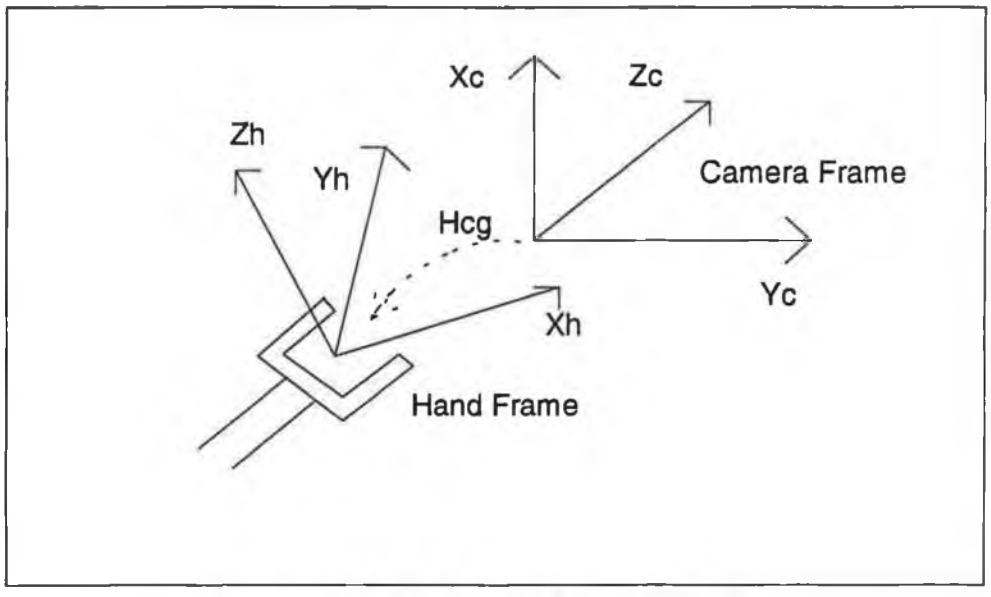

Figure 3.9.1 Eye/Hand Transformation

A few efficient eye/hand calibration algorithms are given in [Shi89], [Tsa89], [Wan92]. The most efficient method for performing the hand/eye calibration with regard to speed and accuracy is method given in [Tsa89]. This approach is presented here and is used in this work. The relations between different coordinate systems and transformations used for calibration are shown in Fig.3.9.2.

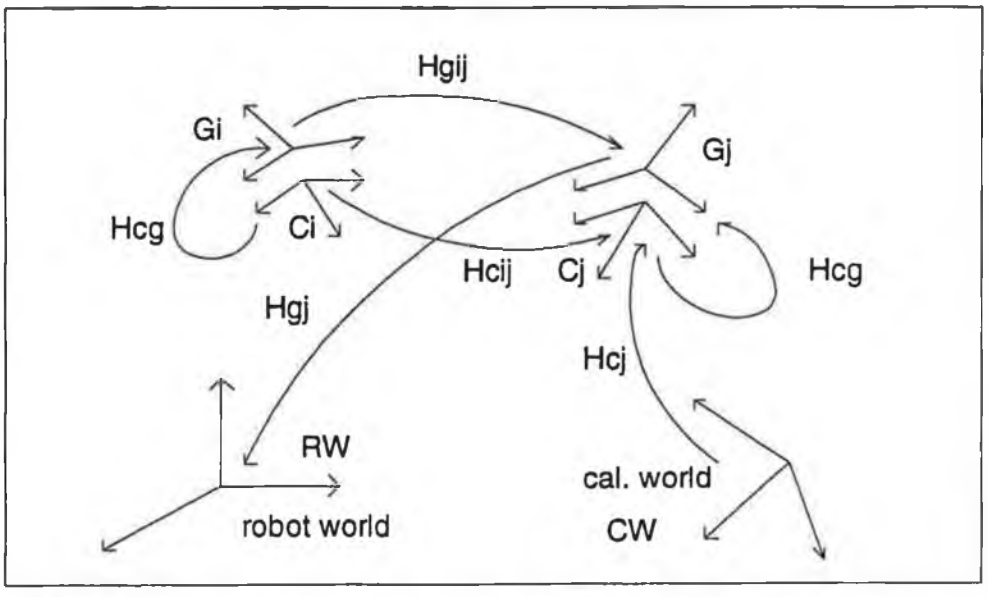

Figure 3.9.2 Relationship Between Different Transformations and Coordinate Systems

The list of definitions for coordinate systems used

- G_i, The hand coordinate frame. The coordinate frame fixed on the robot hand and as the robot moves this frame moves as well.

- C_i, The camera coordinate frame. That is, the coordinate frame fixed on the camera, with the z axis coinciding with the optical axis, and the x, y axes parallel to the image axes.

- *RW*, The robot coordinate frame. It is fixed in the robot work space, and as the robot arm moves around, by using the measurement system the robot controller tells where the hand is relative to *RW*.

- *CW*, The calibration object coordinate frame. This is an arbitrarily selected coordinate frame so that the coordinate of each calibration point is known relative to *CW*.

The list of definitions of homogeneous transformation matrices used

- *Hgi* defines coordinate transformation from G_i to *RW*

$$H_{gi} = \begin{bmatrix} & R_{gi} & & T_{gi} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- *Hci* defines coordinate transformation from *CW* to C_i

$$H_{ci} = \begin{bmatrix} & R_{ci} & & T_{ci} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- *Hgij* defines coordinate transformation from G_i to G_j

$$H_{gij} = \begin{bmatrix} & R_{gij} & & T_{gij} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- *Hcij* defines coordinate transformation from C_i to C_j

$$H_{cij} = \begin{bmatrix} & R_{cij} & & T_{cij} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- *Hcg* defines coordinate transformation from C_i to G_i (eye/hand transformation)

$$H_{cg} = \begin{bmatrix} & R_{cg} & & T_{cg} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Procedure to compute Hcg (Eye/Hand transformation)

Measurements

Measurements are *Hci* and *Hgi* for *i=1,...,N*. *Hci* are obtained from computing camera extrinsic calibration parameters (camera calibration section), using the image gathered at *i-th* position of the robot motion. It defines the relative rotation and translation from *CW to* C_i. In other words it gives *Hci* matrix. *Hgi* are obtained from the robot controller at *i-th* position of the robot.

Elements to be computed

Intermediate elements *Hgij, Hcij*

From Fig.3.9.1 follows

$$H_{gij} = H_{gj}^{-1} H_{gi}, \quad H_{cij} = H_{cj} H_{ci}^{-1} \tag{3.9.1}$$

General homogeneous transformation matrix *H* is given by

$$H = \begin{bmatrix} R & T \\ 0\ 0\ 0 & 1 \end{bmatrix}$$

where R is a rotation matrix and T is a translation vector. Since, the rotation has three degrees of freedom, the matrix representation of the rotation is nonminimal representation. The minimal representation is by three parameters. It is well known that any rotation can be modelled as a rotation by an angle θ around an axis trough the origin defined with a unit vector n. It is obvious that specifying n and θ completely specifies R and vice-versa. The relationship between n, θ and R is given by [Tsa89]

$$n_1 = \frac{R_{32} - R_{23}}{2\sin(\theta)},\ n_2 = \frac{R_{13} - R_{31}}{2\sin(\theta)},\ n_3 = \frac{R_{21} - R_{12}}{2\sin(\theta)},\quad \cos(\theta) = \frac{R_{11} + R_{22} + R_{33}}{2}$$

Another convenient representation is a vector representation of the rotation. In other words it is possible to represent R by n vector scaled by some function of θ. One of such representation is the vector representation of the rotation. In this case, the rotation is presented by the vector [Tsa89]

$$P_r = 2\sin\left(\frac{\theta}{2}\right)\begin{bmatrix} n_1 & n_2 & n_3 \end{bmatrix}^T, 0 \le \theta \le \pi$$

where n is the rotation axis and θ is the rotation angle.

The relation between R and Pr is very simple

$$R = (1 - \frac{|P_r|^2}{2})I + \frac{1}{2}(P_r P_r^T + \alpha Skew(P_r))$$

$$\alpha = \sqrt{4 - |P_r|^2},\quad Skew(V) = \begin{bmatrix} 0 & -V_z & V_y \\ V_z & 0 & -V_x \\ -V_y & V_x & 0 \end{bmatrix} \qquad \textbf{(3.9.2)}$$

According to [Tsa89] the following procedure is used to compute *Hcg*.

Procedure to compute Rcg

Compute the vector *P'cg*. For each pair of different location *i, j* set-up a system of linear equations with *P'cg* as the unknown

$$Skew(P_{gij} + P_{cij})P'_{cg} = P_{cij} - P_{gij}$$

where *Pgij* is the vector representation of the *Rgij, Pcij* is the vector representation of the *Rcij*. Since, *Skew* matrix is always singular, it takes at least two pairs to solve for a unique solution for *P'cg* using linear least squares technique

- Compute *Pcg*

$$P_{cg} = \frac{2P'_{cg}}{\sqrt{1+\left|P'_{cg}\right|^2}}$$

- Compute *Rcg* according to Eq.(3.9.2)

Procedure to compute Tcg

For each pair of locations *i, j,* set up a system of linear equations

$$(R_{gij} - I)T_{cg} = R_{cg}T_{cij} - T_{gij}$$

This system can be solved for T_{cg} by linear least squares. This completes the eye/hand calibration (*Hcg*). The proof of this procedure is given in [Tsa89].

3.10 Accuracy assessment

The accuracy of the eye/hand calibration results is assessed by how accurately, the placement of a camera in 3D world with an arbitrary manipulator movement, can be predicted. Since there is no absolute *Hcg* ground truth to compare with, the accuracy must be assessed as the error between predicted camera pose computed by using the

estimated *Hcg* and measured camera pose for some movements. This is done as follows. For each measurement position, used for collecting data for *Hcg* determination, compute the homogeneous matrix *Hrc* (transformation from robot world frame to calibration world frame *CW*) by

$$H_{rc} = H_{ci}^{-1} H_{cg}^{-1} H_{gi}^{-1}$$

Make an average of *Hrc* for each measurement position. Estimate *Hcg*. Next, move robot to different positions and predict the camera pose relative to the robot world base coordinate *RW* by

$$H_{pc} = H_{cg}^{-1} H_{gk}^{-1}$$

where *Hgk* is obtained from the robot controller. Compare this predicted pose with measured pose

$$H_{mc} = H_{ck} H_{rc}$$

where *Hck* is the measured pose of the camera. In order to compare these transformations we use *RMS* (root mean square) measure. *RMS* measure is defined as

$$RMS = \sqrt{\sum_{i=1}^{n} \frac{(X_{pi} - X_{mi})^2}{n}}$$

where X_{pi} and X_{mi} are predicted and measured vectors. If matrices are to be compared using this measure, than matrices are stacked as vectors.

3.11 Experimental results

1) Description of experimental set-up

A CCD camera is fixed to the last link of a *PUMA-560* robot arm. The robot controller is interfaced to the host computer (PC -386) via a serial link, allowing to control the robot motion and access to the measurement system of the robot. The host computer is fitted with a frame grabber connected to the camera. All system is under full control of the host computer and all processing is fully automatic (Chapter 2).

2) Hcg determination

The robot makes a series of the preplanned motions, at each pause the camera calibration is done and *Hci* matrix is computed, the robot controller supplies *Hgi* matrix. At the end of motion, the Eye/Hand transformation *(Hcg)* and the transformation *(Hrc)* are computed. In Table 3.9.1 the calibrated *Hcg* matrix for 6 pairs of different locations is given.

Rcg			Tcg (mm)
0.98866	0.08650	-0.12174	12.4
-0.07847	0.99123	0.06923	-13.5
0.12718	-0.05779	0.97889	126.4

Table 3.9.1 Eye/Hand Transformation Matrix

3) Hcg accuracy assessment

The robot makes arbitrary motions and predicted and measured camera pose is computed at each pause of the motion At the end of motion *RMS* error in the rotation matrix and translation vector is computed. Table 3.9.2 shows the errors for 6 test robot locations.

Rotation matrix error	Translation vector error (mm)
0.17235	5.5

Table 3.9.2 Errors in Eye/Hand Transformation Matrix

3.12 Conclusions

In this chapter a modification to an existing camera calibration technique has been introduced, that permits us to determine accurately the parameters of the nonlinear camera model using only closed form (non iterative) computations. We have introduced a statistical analysis of the calibrated camera parameters that enable us to asses the accuracy of the estimated parameters, to fuse multiple data and to detect unreilable and unstable calibrations. A statistical measure to evaluate the performance of the calibrated camera in 3D measurement applications has also been introduced. The eye/hand calibration has been presented. The exprimental results indicate high accuracy obtained for both calibration problems.

Chapter 4

Image Processing and Representation

In this chapter we present methods for image processing and image representation needed for extracting necessary information for the purpose of this work. Methods for transforming an image from its digital matrix representation to a very compact and structured representation which is suitable for the algorithms in the following chapters are presented. This representation consists of extracted image contours. Image contour points are extracted by a zero crossing technique. They are then grouped into contours by a gradient guided contour following algorithm. The obtained contours are approximated by straight line segments. Finally, we present an efficient data structure which supports fast data manipulation.

4.1 Introduction

Most image analysis problems are solved in two distinct stages. The first stage is called low-level and consists of extracting visual features (a symbolic representation), the second stage, called analysis, uses this information to perform a particular task. Visual features of an image are objects extracted from the image which contain pertinent information for its analysis. In general such features are either contours or regions of the image. The edges of the image are zones of transition between homogeneous image

regions. They correspond to discontinuities of the image intensity. Physically, intensity discontinuities of the image correspond to discontinuities of parameters of the scene being viewed by the camera. Among them are

- The reflectance of the surface element
- The vector normal to the surface element
- The incident luminance intensity

The type of features used depends of the applications. In this chapter, we describe the low-level processing necessary to construct a symbolic representation of the image which is of use for our system. This representation is based on image contours.

4.2 Edge extraction

There are a lot of techniques for edge detection [Mar82], [Pra91], [Can86], all of them are based on the definition of an edge as a zone of transition between homogeneous regions. Two main approaches for detecting these transitions rely on either the first order derivatives or the second order derivatives of the image. Since, the digitisation and quantisation process introduces noise into a digital image, the computation of first and second order derivatives must be combined with some kind of local smoothing of the image, to improve the signal-to-noise ratio.

Gradient-based techniques usually label a point in an image as the edge point if the gradient magnitude at the point exceeds a given threshold value. The different methods to compute the gradient of the image give different gradient operators, such as Sobel, Robert, Gaussian operators [Pra91].

In this work we use the second order derivative based technique. The zero crossing technique, initially introduced in [Mar82], consists of first filtering an image by a low pass filter, in order to remove the higher spatial frequencies which generally correspond to image acquisition noise. Next, Laplacian of the filtered image is computed. The zero crossings of Laplacian of the filtered image correspond to the maximal points of the gradient image, i.e. to places where the intensity of the image changes the most rapidly. Formally, let *I(x, y)* represent the image intensity function and *G(x, y)* represent the point spread function of the low pass filter. Then filtering is obtained by the convolution product

$$I_f = G * I$$

followed by the Laplacian computation

$$\nabla^2 I_f = \frac{\partial^2 I_f}{\partial^2 x} + \frac{\partial^2 I_f}{\partial^2 y}$$

The filter impulse response is generally Gaussian and is given by

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

where the parameter σ controls the smoothing level of the filter, or the resolution at which zero crossings are detected. The reason for using the Gaussian function for smoothing (filtering) is quite straightforward. The Gaussian function permits us to choose the resolution at which intensity changes are manifested in the image, i.e. to choose the level of detail which is retained in the image. For example, an image which has been smoothed just a little (small σ) will retain a significant amount of detail, while one which has been smoothed a great deal (large σ) will retain only the gross structure.

Since, the two-dimensional Gaussian function is separable into the product of two one dimensional Gaussian function, the convolution of the image with a Gaussian impulse response can be obtained by two one dimensional convolutions, with one dimensional Gaussian function [Ver91]. Specifically

$$G(x,y)*I(x,y) = G(x)*(G(y)*I(x,y))$$

Thus, the image is first convolved with a "vertical" Gaussian and then the resulting image is convolved with a "horizontal" Gaussian. This filtering can be realised by one pass over the image by using buffer size with the length equal to the size of the truncated Gaussian impulse response (Gaussian kernel). Zero crossing points are detected by checking the sign change in a (3x3) window across the image. For the purpose of discrete implementation, the continuous infinite Gaussian impulse response must be discretized and truncated. Normally, one chooses the quantization resolution of the function by deciding on the integer number which will represent the maximum amplitude at the centre point (i.e. 1000), and then one chooses the mask size which includes all non-zero values. Also, the Laplacian operator is replaced by its discrete approximation.

Since, the Laplacian itself is a high-pass filter even after smoothing an image it is possible to extract some zero crossing points which belong to zones of very low contrast and correspond to the texture introduced by image acquisition noise. These points can be suppressed by computing the gradient of the image at the detected zero crossings and retain only zero crossings whose gradient magnitude is above a threshold value. This combines the precision of the edge localisation by zero crossing and robustness to noise of the gradient based edge detectors. This operation is equal to a logical AND between the zero crossing image and the thresholded gradient image. The Sobel (3x3) convolution kernel is used for gradient computation. Zero-crossings of LoG image method for edge detection has the following advantages

- The edge points extraction operations use only simple operations, which can be implemented efficiently, such as convolution and logical operations

- Since zero crossings define the transition from the regions of the different intensity, they form one pixel wide connected contours

- These contours are closed, since by definition they surround regions of the image with constant intensity

These characteristics give a computational advantage in the image processing when compered with the other methods for edge detection, such as gradient based or local maxima techniques.

Fig.4.2.1 shows an image of an industrial part. Fig.4.2.2 shows its edge image obtained as described above.

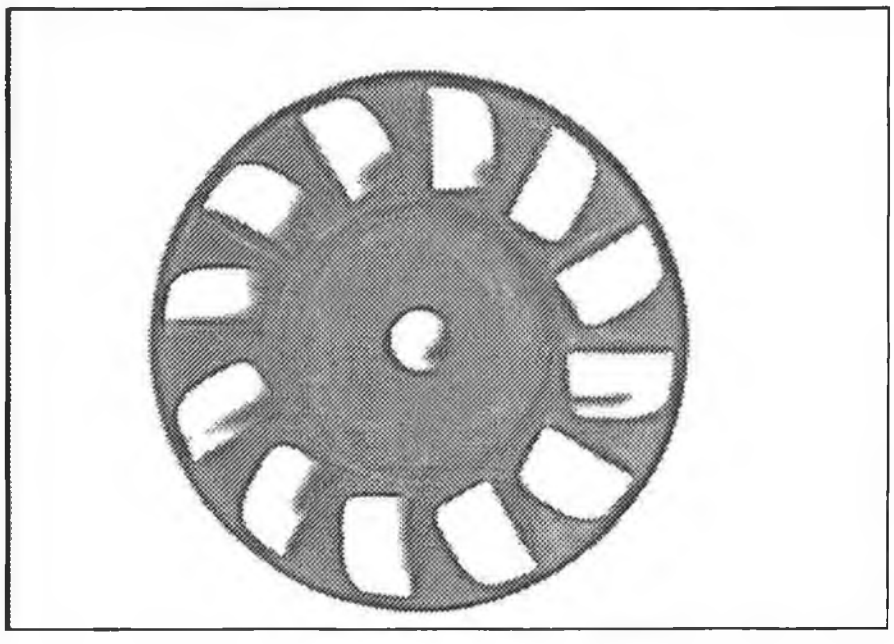

Figure 4.2.1 Raw Image of Object

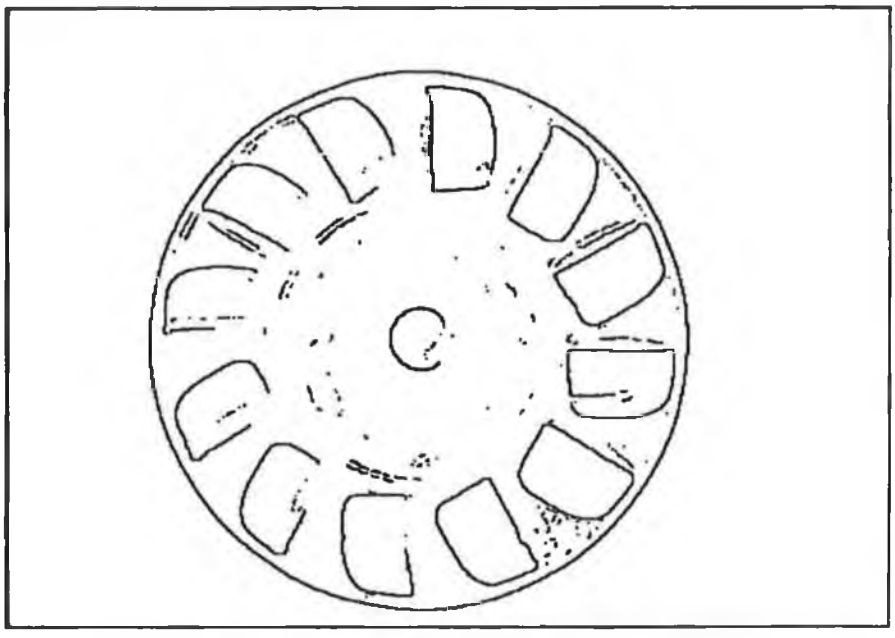

Figure 4.2.2 Edge Image of Object

4.3 Boundary detection

As we discussed previously, edge detection is the first stage of the boundary detection. We need to aggregate these local edge elements, which are a relatively featureless representation, into structures better suited to the process of interpretation. This is usually achieved using processes such as edge linking, gap filling and curve-segments linking in order to generate a distinct, explicit and unambiguous representation of the boundary. There are several techniques for boundary detection and they vary in the amount of domain-dependent information that is used in the grouping process [Bal82], [Ver91]. One of them called contour following is described in detail in the sequel and used in this work.

4.3.1 Contour following

Contour following is a simple and fast approach which uses no domain-dependent information and "follows" the boundary exclusively on the basis of locally derived information. Essentially, all contour following algorithms start with a point which is believed to be on the contour (edge point) and to extend the contour by adding a neighbouring point using local information. This process of extensions is reiterated, starting at this new contour point. We use the following algorithm, described in detail below.

The algorithm scans the zero crossing image starting from the beginning (the upper left corner) in the left to right and top to bottom fashion. As soon as the zero crossing point is encountered a process of building a boundary is started. The point is put in a list and labelled as visited, and the next point to be added to the list from (3x3) neighbourhood is determined on the basis of the edge direction. The edge direction is computed from the gradient direction of the image at this point by using the Sobel (3x3) or (5x5) gradient operator. The edge direction at a point is defined by $\alpha=\beta\pm90$, where β is gradient direction. The edge direction is coarsely quantized into eight direction in the (3x3) neighbourhood. Based on this information the priorities are defined for inclusion of the neighbouring pixel into the boundary. The algorithm attempts to chose the next contour point following directions in the increasing order, provided the direction one is the computed quantisied edge direction, as shown in Fig.4.3.1.1.

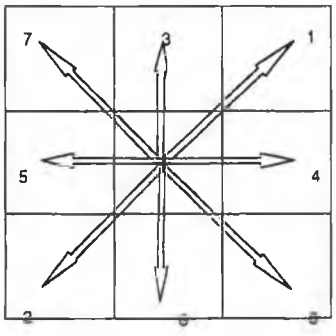

Figure 4.3.1.1 Gradient Guidance

This gradient guided contour following scheme shows a good robustness to the noise. Even though zero crossing contours are supposed to be one pixel wide, because of noise and digitisation it is possible to have thicker contours. These are suppressed by removing all points in the (3x3) neighboured except the chosen one. This allows us to avoid very costly morphological processing such as thinning. The process continues until a gap is discovered, meaning the contour cannot be extended any further. At this stage the algorithm will tray to fill in gaps of predefined length which we chose as admissible breaks in the contour. In our implementation we chose the gap length up to three pixels in any direction. The guiding is based on the edge direction as described before. If the gap is filled in the process continues until the contour can not be followed any further, or the starting point is reached signalling closed contour. If the contour is closed, the contour list consist of all boundary points. If the contour is not closed, the contour following resumes at the starting point in order to group the second part of the open contour. In this case the process terminates when the contour can not be extended any further. In both cases the contour list is formed representing a boundary. After this the image scanning is resumed at the starting point of the previous boundary to group other contours. The process continues until the end of the image is reached. At this stage a list of contours and corresponding contour points have been formed.

4.3.2 Polygonal approximation

After processing an edge image by the contour following algorithm, the image is represented by a set of digital curves. Further structuring of these digital curves is necessary to bring out information pertinent to our intended application (visual reconstruction based on straight-line segments). The new structuring of the image data is obtained by polygonal approximation of the contours. This allows a further reduction of the image data and makes subsequent processing stages very efficient or even possible.

According to [Pav82] each curve can be approximated up to desired accuracy by a polygonal line. This means that a digital curve can be represented instead of by all points by polygonal vertices only. This makes a considerable data reduction depending of the curve type. It is important to note that this approximation is not unique for desired accuracy, i.e. different polygons can achieve the same desired accuracy. One of the procedures for performing the polygonal approximation of the curve is the recursive splitting algorithm. In the splitting scheme, curve segments are continually divided (usually into two parts) as long as they fail some fitting condition.

The recursive splitting algorithm we use is given as follows [Pav82]. Given a curve, the principle is to join two points of the curve by a straight line segment. For every point on the curve, compute its perpendicular distance to the straight line and find the point with the maximal distance ε separating the curve from the line segment. If the distance is less than a threshold, the curve segment is approximated by the straight line segment, otherwise the initial chain is split into two at the point where the distance is ε and replace the starting curve segment with two new segments. Recursively apply the algorithm on both new chains until the predefined threshold is reached on every curve segment. This is shown in Fig.4.3.2.1.

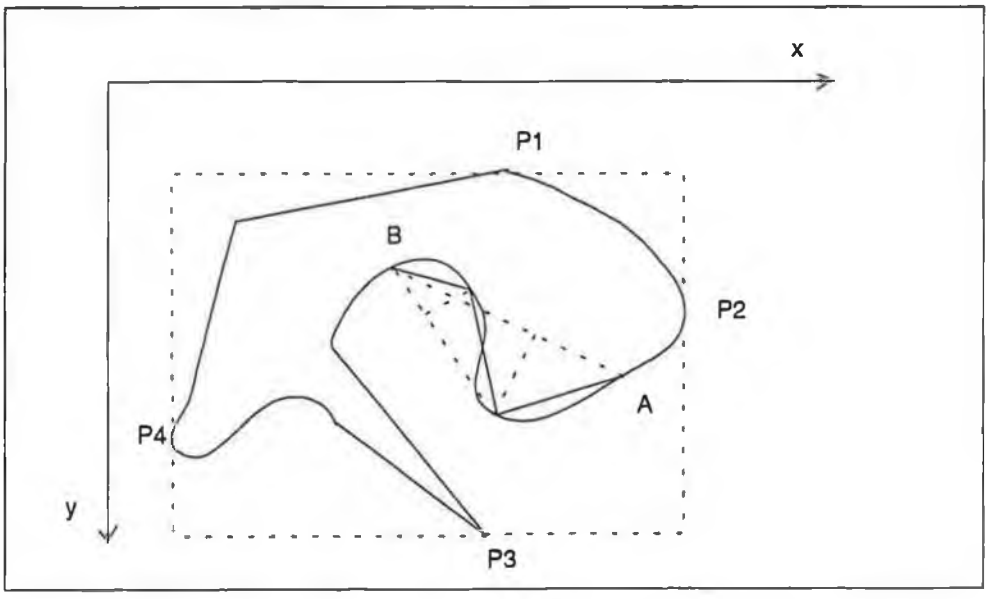

Figure 4.3.2.1 Polygonal Approximation

The main computational demand in this algorithm is to determine a point on a curve segment where the maximal distance ε between the curve segment and a straight-line segment joining the curve segment endpoint occurs. An efficient method to determine this point is by transforming the curve segment to the coordinate system attached to the straight-line segment. In the transformed coordinate system this point is defined by the absolute maximal value of the y' coordinate of the curve segment points. This transformation is given by

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x - x_1 \\ y - y_1 \end{bmatrix}, \quad T = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}, \quad \theta = \operatorname{atan}\left(\frac{y_2 - y_1}{x_2 - x_1}\right), \quad \varepsilon = \max\|y'\|$$

This is shown in Fig.4.3.2.2.

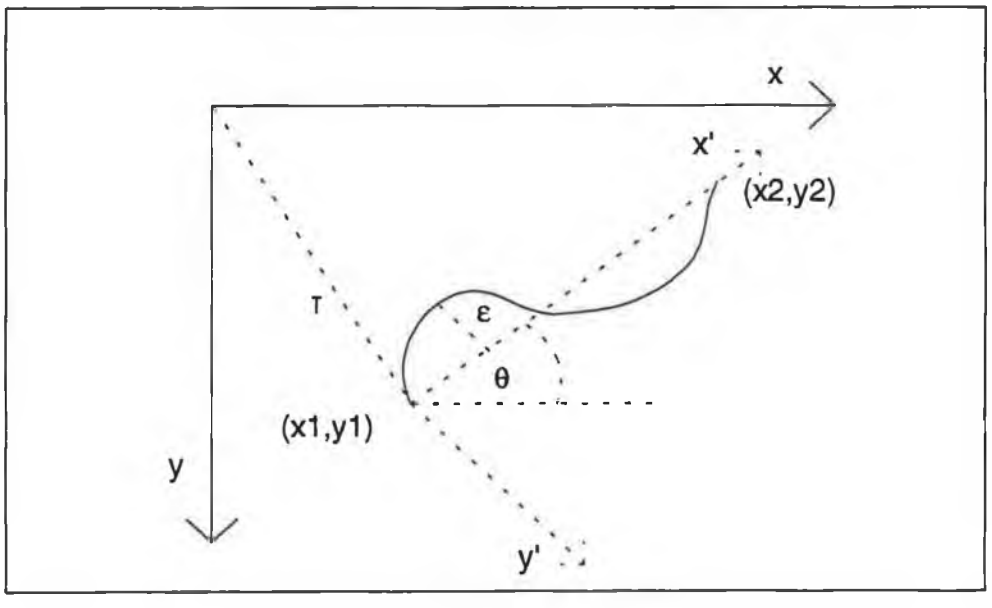

Figure 4.3.2.2 Maximal Separation Between Curve and Straight-Line Segment

From Fig.4.3.2.1 can be seen that the final approximating polygon depends on the curve itself and the starting partitioning points *(A, B)*, even though the desired accuracy of

approximation is independent of the curve and partitioning points. A change in partitioning points may result in a different number of polygon segments and different lengths and orientations of the segments. The simplest way to determine a set of partitioning points is to define a bounding box of a curve. The bounding box of a curve is a rectangle which encloses the curve. As described above by using contour following algorithm the connected points are detected and stored. The extreme points of the contour, that is, those having the minimal and maximal coordinates are stored as well. These points form the bounding box as it is shown in Fig.4.3.2.1 where points (P_1, P_2, P_3, P_4) define the bounding box. These points depend on the curve orientation in the image plane. In the case that the digital curve is close enough to a polygon, the bounding box coordinates will coincide with some vertices of the curve, and the polygonal approximation will be unique and independent of the curve orientation since the vertices of the approximating polygon cluster about high curvature points on the curve. Since the curves we anticipate dealing with consist of curved and straight parts, the partitioning points defined by the bounding box only cannot guarantee a unique approximation, independent of the curve orientation. Here we develop a way to make this process less sensitive to the curve orientation, for curves with distinct features such as sharp turns or corners, by detecting sharp turns, i.e. points on the curve with high curvature. We determine the high curvature points in the following manner. During the contour following, the edge directions of the edge points are stored. The high-curvature points correspond to the sharp changes in the edge directions. Basically, the θ-s curve [Bal82] is constructed, where θ is the angle of a tangent to the curve and s is the arc length of the contour traversed. The horizontal parts of the θ-s curve correspond to the straight-line parts of the original curve (θ is not changing), while jumps and sharp changes in the θ-s curve correspond to corners and high curvature points in the original curve. Thus, by detecting the sharp transitions in the θ-s curve, we in fact perform detection of corners and other high curvature points in the original curve. The problem of detecting sharp changes along the curve is equivalent to the problem of edge detection in one dimension. So, we use the one dimensional zero crossing of the second derivative of the

Gaussian smoothed θ-s curve as points with high curvature. The order of high curvature points are independent of the curve orientation and the starting point on the curve. The bounding box coordinates and high curvature point coordinates are then used to generate partitioning points.

Finally, the algorithm for polygonal fitting can be summarised as follows

Step 1

Build an initial pseudo polygon using the bounding rectangle coordinates and the points with high curvature along the curve. This makes the fitting algorithm work equally well, regardless of the curve orientation and where it starts its operation on the contour.

Step 2

Modify and extend the pseudo polygon. In this step, the initial pseudo polygon is modified and extended to ensure an accurate approximation. Efficient implementation of this algorithm is by a recursive procedure. The family of approximation of a given curve is stored in a binary tree. The leaves of the tree gives the vertices of the polygon approximation sorted by increasing arc length. Any tree traversal can access these vertices. The end and start points for this algorithm are supplied by step 1. After approximation is done for each curve segment , a list of the polygon vertices is built for the entire curve sorted by the increasing arc length. The same procedure is repeated over all contour chains. After this is done a list comprising all straight line segments is formed.

Even though some precautions were taken to robustify the polygonal fit to a digital curve, it is still possible to have broken segments due to noise. By scanning the list of straight-line segments, obtained previously in step 2, all segments with lengths less than a prescribed threshold are removed leaving more reliable segments. This list

represents the image in a very compact form suitable for subsequent processing. An example of polygonal approximation by the above presented algorithm is given in Fig.(4.3.2.3) for the edge image given in Fig.(4.2.2).

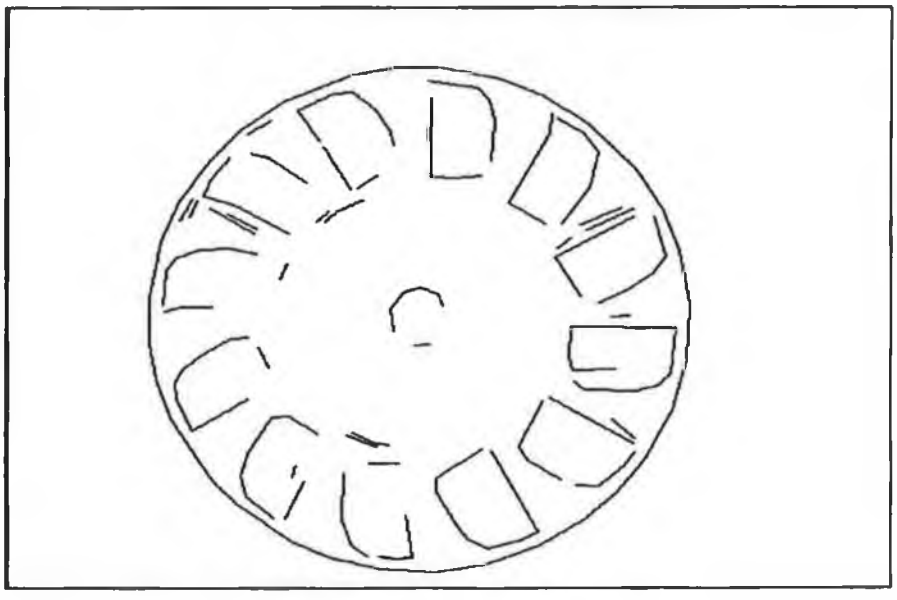

Figure 4.3.2.3 Example of Polygonal Approximation

4.4 Data structuring

The output of the previous processing stage is a list of straight-line segments. The segment is represented by its end points, another convenient representation of a straight-line segment is by its mid point, orientation and length. The last representation is very suitable for the segment matching in two images, as it will be seen in the following chapters. Since the segment matching requires multidimensional range searching operations, in order to avoid the very costly brute-force searching strategy, we developed an indexing method for data structuring based on buckets [Knu75], [Aya91] which permits very efficient range searching. We use rectangular cells to structure the image. These cells form a regular partition of the image, as shown in Fig.(4.4.1).

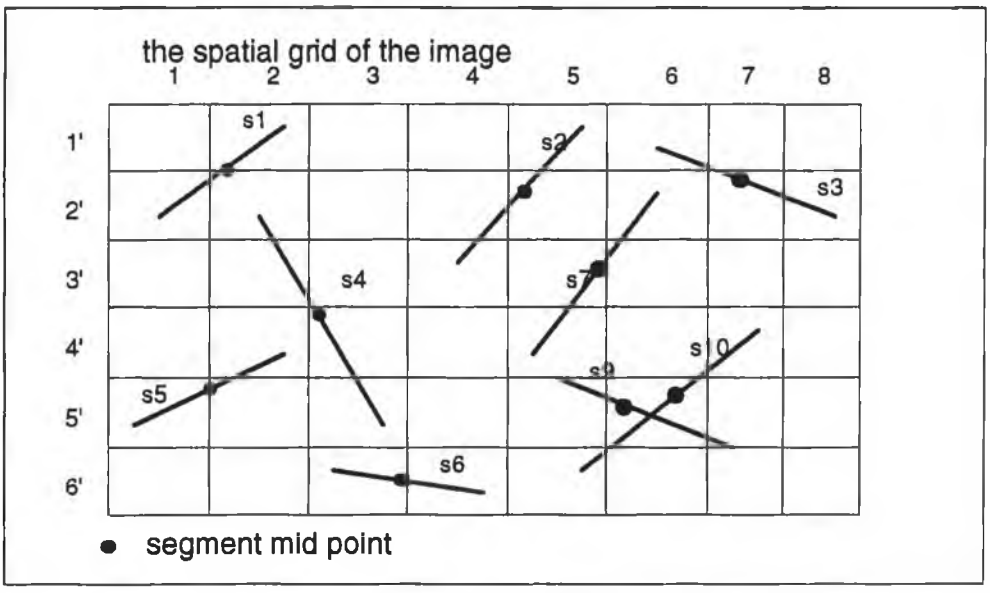

Figure 4.4.1 Image Structuring

After getting the segment list, we construct an indexing table for the image. To each image cell corresponds an entry into the indexing table. The each entry comprises a list of pointers to segments whose mid points belong to that cell (Fig.4.4.1). The algorithm for computing the indexing table has the linear complexity in the number N of segments *(O(N))*. As will be seen in the following chapters, this indexing table drastically reduces the computational complexity of the segment matching algorithms. In fact, these algorithms hypothesise the appearance of a currently observed segment in the next image and verify its hypothesis by analysing the segments of the next image. Since the hypothesised segment appearance involves the position of its mid point, then for verification it is enough to examine segments contained in the image cells overlapped by the hypothesised window, instead of checking all of them (the brute-force algorithm which matches segments in two images has complexity of *O(NN)*, assuming N is number of segments in both images). The complexity of the matching algorithm supported with this data structure lies somewhere between *O(N)* and *O(NM)*, where M is maximum number of segments per cell, depending on the cell size and segments distribution over the image. For the small cell size and equally distributed segments this tends to *kO(N)*,

where k is small compared to N. For the large N this is a significant improvement in speed, having in mind that each comparison requires the matrix calculation. Obviously there is a trade-off between the memory requirements and speed. After computation of the indexing table, the image is represented by the segment list and its indexing table. Fig.4.4.2 shows an example of the data structure for the image given in Fig.4.4.1.

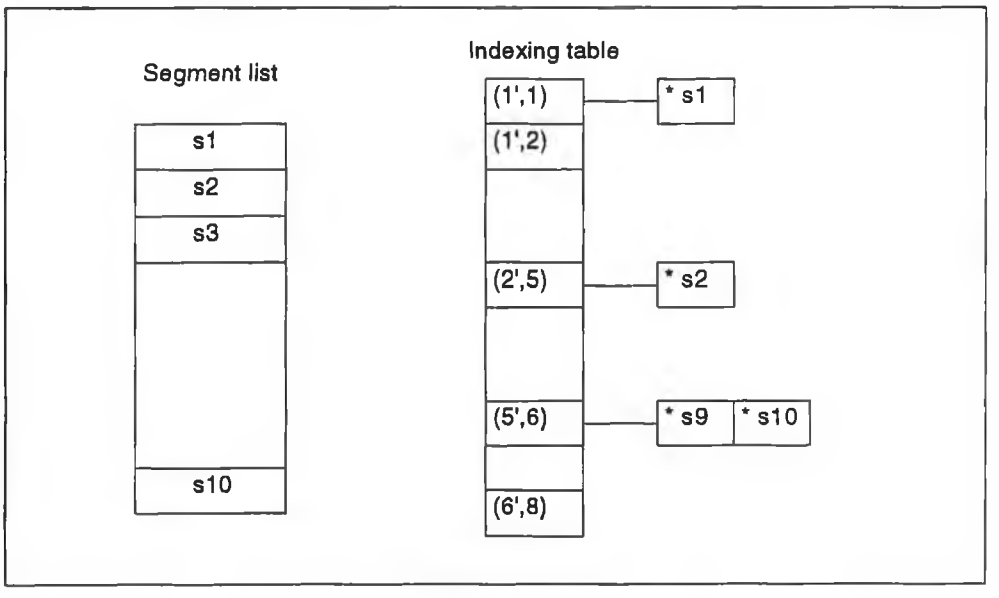

Figure 4.4.2 Supporting Data Structure

For example to find all segments whose mid points are in the spatial range given by the image cell (5',6) just two accesses will retrieve segments (s_9,s_{10}) as opposed to ten accesses without supporting indexing table.

4.5 Image geometric rectification

So far it has been assumed that the image formation is modelled by the ideal pin-hole model and ideal lens. This means that the image of a 3D straight line is a 2D line. Since the lens introduces non-linear distortions, as was mentioned in the previous chapter, image data must be corrected or rectified to account for this distortions. Extracted contour chains are rectified according to the nonlinear lens mode, before

polygonal approximation is done. From the nonlinear lens model and the intrinsic camera calibration parameters (Chapter 3), rectified image coordinates are given by

$$U_u = C_x + \frac{(U_d - C_x)}{(1 + kD^2)}$$
$$V_u = C_y + \frac{(V_d - C_y)}{(1 + kD^2)}$$
$$D^2 = X_d^2 + Y_d^2, \quad X_d = s_x dx(U_d - C_x), \quad Y_d = dy(V_d - C_y)$$

4.6 Conclusion

We have presented methods for transforming an image from its digital matrix representation to a very compact and structured representation which is suitable for the algorithms in the following chapters. This representation consists of extracted image contours. Image contour points are extracted by a zero crossing technique. They are then grouped into contours by the gradient guided contour following algorithm. The obtained contours are approximated by straight line segments. Finally, we have built an efficient data structure which supports fast segment searching.

Chapter 5

3D Vision Based Structure Estimation

The new technique for 3D structure (3D geometrical model) estimation of objects within a class of objects (polyhedra), by using monocular image sequence and known camera motion, is presented in this chapter. The technique is based on tracking line-segments over image sequences. The tracking process consists of prediction, matching and updating stages. These stages are handled in a Kalman filtering framework of covariance based prediction, matching and updating. The prediction stage of our tracking process does not use heuristics about motion in the image plane and applies for arbitrary camera motion. The prediction is based on assumptions about object structure (i.e. a rough knowledge of a distance between camera and an object is assumed known and the depth extent of the object is small compared with the camera-object distance) for the initialization phase, the rest of the tracking process is based on estimated object structure. The matching stage is based on the simple nearest-neighbour matching algorithm using the Mahalobonis (statistical) distance as a similarity measure. The updating stage is based on the standard Kalman filter estimation algorithm. Experimental results from a camera mounted on a robot arm are presented to show the efficiency of the proposed algorithm.

5.1 Introduction

In robotic applications, a robot has to perform a series of tasks such as object manipulation and object recognition. A common requirement in performing any of these tasks is the 3D description of a scene or the objects in the scene. A substantial amount of effort has been invested in the computer vision area, on developing robust methods for 3D structure computation. However, robust computation of 3D structure with respect to image noise is still very desirable. Using known camera motion to estimate 3D structure of an object from image sequences is an important and efficient way to tackle the problem of 3D structure estimation for the following reasons. The image measurements are inaccurate. Errors in image measurements get reflected in the estimation of structure. These errors can be reduced using more images and fusing their pertinent information. A sequence of images allows solution of the problem in an incremental manner. The first few frames can be used to obtain a rough estimate of structure, which can be refined using observations based on successive frames. The correspondence (matching) among image features can be resolved more reliably and easily by tracking image features over a sequence. This chapter describes a new method to integrate 3D structure of an object in terms of a set of 3D straight-line segments, by measuring and fusing their images in terms of 2D image straight-line segments over a sequence of images, using known camera motion [Prl93], [Prl94]. The system is based on fusing multiple stereo views. Stereo views construction and stereo views fusion is realized and highly simplified by tracking image features over an image sequence. This process of tracking can be seen as a cyclic one. The model structure undergoes a cycle of prediction, matching and updating. The process of prediction, matching and updating is based on the Kalman filtering framework. The tracking process permits us to simplify generally very difficult feature-matching problem between successive images by using a simple nearest-neighbour matching algorithm, and to integrate multiple observations in order to improve the overall accuracy of reconstructed 3D structure. Unlike other approaches, our approach requires no constraints on camera motion, and feature tracking is based on generic assumptions

about the object structure and estimated structure itself instead of on traditional feature tracking which is based on image motion heuristics. This approach provides for reliability, accuracy, computational advantages and a very simple tuning of the overall system parameters.

5.2 3D Structure reconstruction

An image represents a mapping from 3D objects or a scene to 2D measurements. Therefore, it is not possible to recover the spatial geometry of an unknown object from a single image. At least two images are necessary to recover 3D information. Two basic techniques aimed at structure recovery are so-called stereo vision and structure-from-motion.

5.2.1 Stereo vision

By having two or more images taken from different viewing angles, it is possible to determine the spatial position of a world point observed at least at two images. This is referred to as stereo vision. The basic stereo configuration is shown in Fig.5.2.1.1.

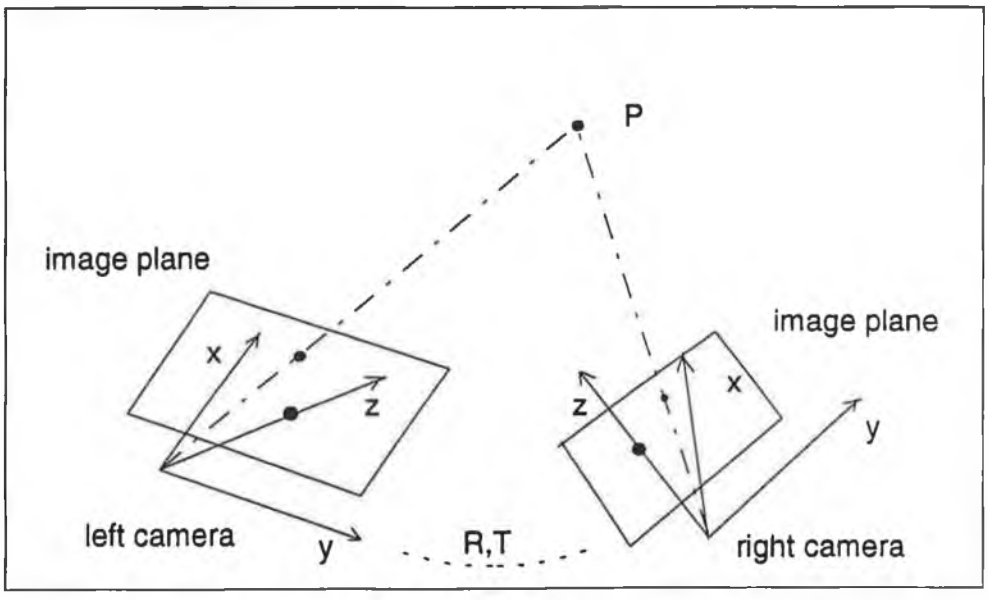

Figure 5.2.1.1 Stereo Configuration

The stereo triangulation is a way to obtain 3D measurements. Let us first define different coordinate systems used. We use one coordinate system centered at the left camera and one coordinate system centered at the right camera. The coordinate system centered at the left camera is used as a coordinate system of the entire stereo configuration, that means 3D computations are done with reference to this system. The orientation and position of the right camera with respect to the left camera is given by a rotation matrix R and translation vector T (assumed known, for example determined trough calibration). The camera model is given in Chapter 3. The relation between image plane projections of a world point P and its 3D coordinates in these frames is given by

$$u_l = s_x \frac{x_l}{z_l} + C_x, \quad v_l = s_y \frac{y_l}{z_l} + C_y$$

$$u_r = s_x \frac{x_r}{z_r} + C_x, \quad v_r = s_y \frac{y_r}{z_r} + C_y \qquad \textbf{(5.2.1.1)}$$

where (u_l, v_l) and (u_r, v_r) are the image plane coordinates in the left and right image, (x_l, y_l, z_l) and (x_r, y_r, z_r) are coordinates of the point P in the right and left coordinate system and (s_x, s_y, C_x, C_y) are camera intrinsic parameters. These coordinates are related by a rotation matrix R and translation vector T between coordinate systems

$$\begin{bmatrix} x_r \\ y_r \\ z_r \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} x_l \\ y_l \\ z_l \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \qquad \textbf{(5.2.1.2)}$$

By substituting Eq.(5.2.1.2) into Eq.(5.2.1.1) we obtain

$$u_l = s_x \frac{x_l}{z_l} + C_x, \quad v_l = s_y \frac{y_l}{x_l} + C_y$$

$$u_r = s_x \frac{r_{11}x_l + r_{12}y_l + r_{13}z_l + t_x}{r_{31}x_l + r_{32}y_l + r_{33}z_l + t_z} + C_x, \quad v_r = s_y \frac{r_{21}x_l + r_{22}y_l + r_{23}z_l + t_y}{r_{31}x_l + r_{32}y_l + r_{33}z_l + t_z} + C_y$$

(5.2.1.3)

Therefore, we have four equations with three unknowns (x_l, y_l, z_l). The process of determining these unknowns according Eq.(5.2.1.3) is called the stereo triangulation.

Stereo error modelling

All images are contaminated by noise and subsequently regardless of the image feature extraction algorithms, the noise will propagate and contemn the image features (in this case a location of image projections of a world point). From this fact it is very important to analyse, take into account and minimise the impact of the noise on the corresponding 3D reconstruction. This effect is shown in Fig.5.2.1.2.

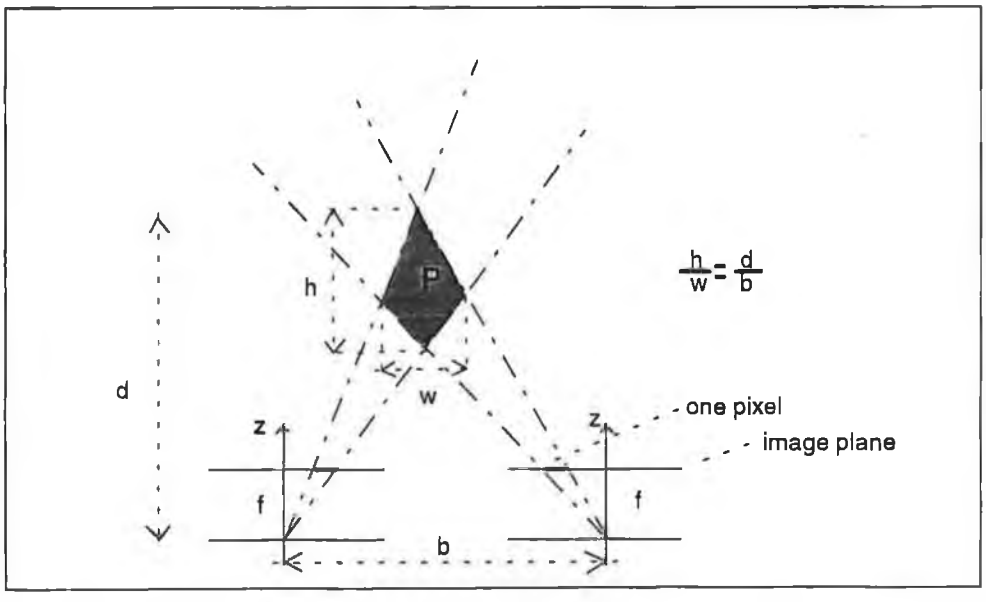

Figure 5.2.1.2 Stereo Errors

From this simple stereo configuration and taking into account only spatial digitisation noise, it is obvious that the true world point can lie anywhere inside the shaded region. Also it is clear that different components of a three-dimensional point determined by stereo triangulation have quite different uncertainties, i.e. the depth component is much less reliable than the lateral components (since the distance to the object is usually larger than the distance between cameras). It is of essential importance to take into account the image noise during the 3D reconstruction and to have a measure of uncertainties in the obtained results.

Given specific descriptions of real cameras and scenes, we can obtain bounds on the estimation accuracy of the 3D stereo computation, using perturbation or covariance analysis techniques based on first-order Taylor expansions [Mat87], [Aya87]. In the sequel the covariance analysis is presented in detail. Here, the perturbation analysis is used in order to give some insight about the relative accuracy of depth estimates obtained from different stereo configurations (we construct the stereo configurations by moving a camera). It is well known that the rotational displacements between images provides no depth information. Furthermore, for the translational displacements the accuracy of depth estimates depends on the type of translations between images forming the stereo system.

A short derivation that demonstrates the relative accuracy obtainable from forward and lateral image displacements forming the stereo system will be given. For clarity, we consider only stereo images displaced by translation along X and Z axes. For the lateral displacement along X axis, i.e. for $t_y = t_z = 0$ and $R = I$ (identity matrix), from Eq.(5.2.1.3) the inverse depth is given by

$$\frac{1}{z} = \frac{u_r - u_l}{s_x t_x} \qquad \textbf{(5.2.1.4.A)}$$

whereas forward displacement, i.e. $t_y = t_x = 0$ and $R = I$, gives the depth

$$\frac{1}{z} = \frac{-(u_r - u_l)}{(u_r - C_x)t_z} \qquad \textbf{(5.2.1.4.B)}$$

For the sake of simplicity in derivations, we will deal with the inverse depth (disparity). The perturbation in the inverse depth arising from the perturbation in image measurements (differences of image coordinates) can be expressed for the lateral displacement by

$$d_l = \frac{1}{z}, \quad \delta d_l = \frac{\partial d_l}{\partial(u_r - u_l)}\delta(u_r - u_l), \quad \delta d_l = \frac{\delta(u_r - u_l)}{s_x t_x} \qquad \textbf{(5.2.1.5)}$$

and the forward displacement by

$$d_f = \frac{1}{z}, \quad \delta d_f = \frac{\partial d_f}{\partial(u_r - u_r)}\delta(u_r - u_l), \quad \delta d_f = -\frac{\delta(u_r - u_l)}{(u_r - C_x)t_z} \qquad \textbf{(5.2.1.6)}$$

These equations give the error in the inverse depth as a function of the error in the measured difference of the point locations in the stereo images, the amount of displacement between images, and the position of the feature in the field of view.

Since we are interested in comparing forward and lateral displacements, a good way to visualise these equations is to plot the relative depth uncertainty. Assuming that the measurement perturbations are the same in both models, the relative uncertainty is given by

$$\frac{|\delta d_f|}{|\delta d_l|} = \frac{|s_x t_x|}{|(u_r - C_x)t_z|} = \frac{|f t_x|}{|dx(u_r - C_x)t_z|}, \quad s_x = \frac{f}{dx} \qquad \textbf{(5.2.1.7)}$$

where f and dx are camera intrinsic parameters (Chapter 3). The image coordinate u_r indicates where the object appears in the field of view. Fig.5.2.1.3 shows that the above formula for the relative uncertainty is thus

$$\frac{\delta d_f}{\delta d_t} = \frac{t_x}{\tan(\theta) t_z} \qquad \textbf{(5.2.1.8)}$$

where θ is the angle between the object and the camera (optical) axis.

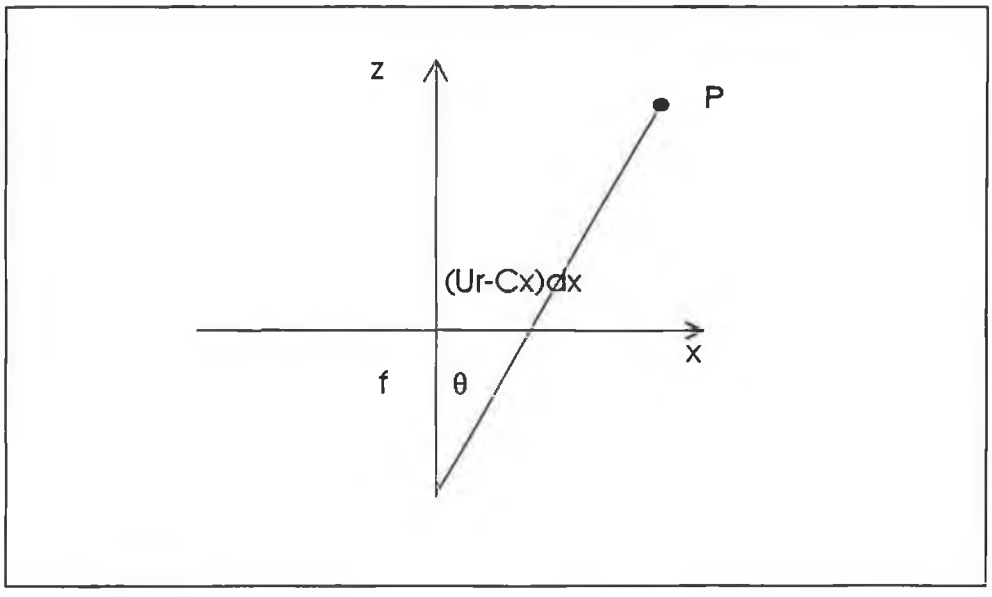

Figure 5.2.1.3 Angle between Object and Camera Axis

This relationship is plotted in Fig.5.2.1.4 for $t_x = t_z$. At 45° from the optical axis, depth uncertainty is equal for lateral and forward displacements. As this angle approaches zero, the ratio of uncertainty grows, first slowly then increasingly rapidly. In general, for practical fields of view, the accuracy of depth extracted from the forward translation will be effectively unusable for a large part of the image. This fact suggests that the "best" interframe camera displacements are lateral ones, while forward displacements are the "worst".

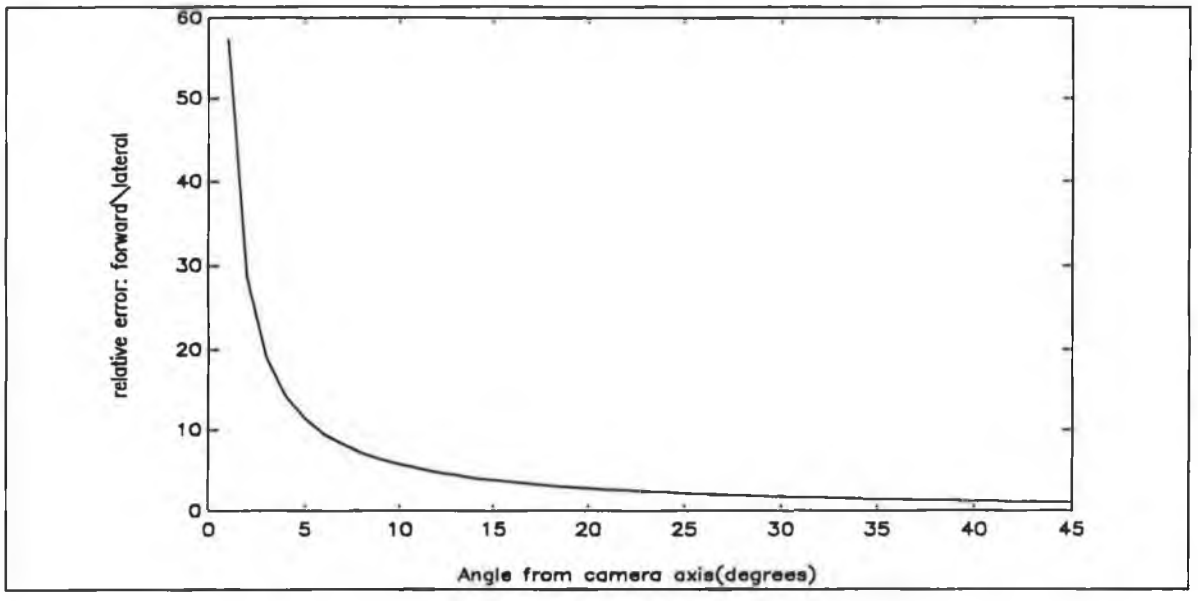

Figure 5.2.1.4 Relative Depth Uncertainty for Forward vs. Lateral Camera Displacement

Estimation criteria

Minimum variance estimator

The problem of solving the stereo triangulation equations Eq.(5.2.1.3) can be tackled in an optimal manner according to the estimation theory concepts. The most commonly used estimation criterion is the minimum variance criterion [Gel74], [Bar88]. One of the main advantages of this criterion is that we do not need to know the exact noise distribution, which is very difficult to obtain in most applications. As we will see the minimum variance estimator does not require knowledge of more than second-order statistics of the noise distribution, which often can be estimated in practice.

Let us suppose that an observation vector y is related to a parameter vector x by the equation

$$y = f(x) + \varepsilon \qquad \textbf{(5.2.1.9)}$$

where ε is a random vector with zero-mean $E(\varepsilon)=0$, and a covariance matrix $E(\varepsilon\ \varepsilon^T)=R$.

The unbiased minimum variance estimator is one that minimises

$$\min_x \ (y-f(x))^T R^{-1}(y-f(x)) \qquad \textbf{(5.2.1.10)}$$

In other words, the optimal parameter vector $\hat{x}$ is the one that minimises the matrix weighted error between the computed observation and the actual observation y. At solution that minimises Eq.(5.2.1.10), the optimal estimate $\hat{x}$ has a covariance matrix given by

$$E((x-\bar{x})(x-\bar{x})^T)=\left(\frac{\partial f(\bar{x})}{\partial x}^T R^{-1}\frac{\partial f(\bar{x})}{\partial x}\right)^{-1} \qquad \textbf{(5.2.1.11)}$$

Returning to the stereo equations Eq.(5.2.1.3) the optimal three-dimensional point $\hat{x}$, according to the minimum variance criterion should minimise

$$\min_x \begin{bmatrix} u_l-u_l(x) \\ v_l-v_l(x) \\ u_r-u_r(x) \\ v_r-v_r(x) \end{bmatrix}^T \begin{bmatrix} R_l & 0 \\ 0 & R_r \end{bmatrix}^{-1} \begin{bmatrix} u_l-u_l(x) \\ v_l-v_l(x) \\ u_r-u_r(x) \\ v_r-v_r(x) \end{bmatrix} \qquad \textbf{(5.2.1.12)}$$

where (u_l, v_l) and (u_r, v_r) are a pair of noisy stereo projections assumed uncorrelated, R_l and R_r are their covariance matrices (determining the covariance matrices will be discussed in the sequel). This is a nonlinear minimisation problem. We first give an approximate solution to this problem. By simple manipulation the stereo equations Eq.(5.2.1.3) can be expressed as a set of linear equations

$$Hx=b \qquad \textbf{(5.2.1.13)}$$

These equations can be solved by the least-squares method

$$\hat{x}=(H^TH)^{-1}H^Tb \qquad \textbf{(5.2.1.14)}$$

The geometrical interpretation of this approximate solution is as follows: Due to noise, the two back projection lines through the observed points in the left and right images, respectively, do not intersect in space. The solution in Eq.(5.2.1.14) is the midpoint of the shortest line segment that connects these two projection lines. From this approximate solution a few (usually one or two) iterations are performed to minimise Eq.(5.2.1.12) (we use the Newton's method). It has been observed that the approximate solution Eq.(5.2.1.14) is very close to the optimal solution Eq.(5.2.1.12), unless the stereo configuration is very unusual (i.e., one camera is far behind the other). Therefore, in most cases the use of the approximate solution (closed-form) will not cause significant performance degradation when compared to the optimal solution (iterative). In our experimental work we use the approximate solution.

The error covariance of the estimated position $\hat{x}$ also needs to be determined. It simply follows from Eq.(5.2.1.11) and Eq.(5.2.1.12)

$$Cov(\hat{x}) = \left(\frac{\partial}{\partial x} \begin{bmatrix} u_l(\hat{x}) \\ v_l(\hat{x}) \\ u_r(\hat{x}) \\ v_r(\hat{x}) \end{bmatrix}^T \begin{bmatrix} R_l & 0 \\ 0 & R_r \end{bmatrix}^{-1} \frac{\partial}{\partial x} \begin{bmatrix} u_l(\hat{x}) \\ v_l(\hat{x}) \\ u_r(\hat{x}) \\ v_r(\hat{x}) \end{bmatrix} \right)^{-1} \quad \textbf{(5.2.1.15)}$$

Kalman filter estimator

Data or sensor fusion is concerned with algorithms that combine multiple measurements and their uncertainties in order to obtain the final (fused) estimate which is better than any particular measurement in some sense. Recent advances in sensor fusion are based on techniques from estimation theory.

One of the very powerful technique is Kalman filter [Kal62], [Gel74], [Bar88]. The Kalman filter is a recursive, optimal in Bayesian sense, estimation technique used to estimate states of linear stochastic dynamic systems being observed with noisy sensors. The optimality in Bayesian sense is closely related to the minimum variance and least-squares optimality. These relations are given in [Gel74], [Bar88]. The filter is based on two separate probabilistic models. The first model, the system model, describes evaluation over time of the current state vector. The transition between states is characterized by the known transition matrix and the addition of Gaussian noise with a known covariance matrix (process noise). The second model is the measurement (sensor) model, that relates the measurement vector to the current state through a measurement matrix and the addition of Gaussian noise with a known covariance matrix (measurement noise). The Kalman filter equations are given by

System model

$$x_{k+1} = Ax_k + Gw_k, \quad w_k \approx (0, Q)$$

Measurement model

$$z_k = Hx_k + v_k, \quad v_k \approx (0, R)$$

Prediction phase

$$\hat{x}_{k+1}^- = A\hat{x}_k, \quad \hat{x}_0 = \hat{x}_p$$

$$P_{k+1}^- = AP_kA^T + GQG^T, \quad P_0 = P_p$$

Update phase

$$K_{k+1} = P_{k+1}^{-} H^T \left[H P_{k+1}^{-} H^T + R \right]^{-1}$$

$$P_{k+1} = P_{k+1}^{-} - K_{k+1} \left[H P_{k+1}^{-} H^T + R \right] K_{k+1}^T$$

$$\hat{x}_{k+1} = \hat{x}_{k+1}^{-} + K_{k+1} \left[z_{k+1} - H \hat{x}_{k+1}^{-} \right]$$

(5.2.1.16)

In order to apply the Kalman filter we need to define the system model and the measurement model, to characterise the process and measurement noise by means of their covariance matrices and to define initial conditions.

For example, let us assume we need to estimate a constant parameter vector from its noisy observations. In this case The Kalman filter estimation is formulated as follows

System model

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \\ x_n(k+1) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_n(k) \end{bmatrix} \qquad \textbf{(5.2.1.17)}$$

where $(x_1, x_2, \ldots, x_n)$ are components of the parameter vector, the state transition matrix is an identity matrix and the process noise is zero, reflecting the fact that the state vector is constant

Measurement model

Measurement model of $(x_1, x_2, ..., x_n)$ components are given by

$$\begin{bmatrix} z_1(k) \\ z_2(k) \\ z_m(k) \end{bmatrix} = H \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_n(k) \end{bmatrix} + \begin{bmatrix} \varepsilon_1(k) \\ \varepsilon_2(k) \\ \varepsilon_n(k) \end{bmatrix}, \quad Cov\left(\begin{bmatrix} \varepsilon_1(k) \\ \varepsilon_2(k) \\ \varepsilon_n(k) \end{bmatrix} \right) = R \qquad \textbf{(5.2.1.18)}$$

where $(z_1, z_2,...,z_m)$ is a measurement vector, H is a measurement matrix and R is the measurement noise covariance matrix. By having the system model and measurement model together with noise description the standard Kalman filter equations Eq.(5.2.1.16) are used to optimise the parameter vector. The Kalman filter algorithm applied to estimate constant parameters is equivalent to the recursive weighted least-squares algorithm [Bar88].

As was mentioned the Kalman filter is the optimal estimator under assumption of Gaussian errors. If we do not make this assumption, then among linear estimators, the Kalman filter produces the unbiased estimate of a state vector with minimum variance [Gel74]. This means that among all the estimators which compute the state vector as a linear combination of the measurements, the Kalman filter minimises the expected value of the error norm, while ensuring the absence of bias.

Stereo correspondence problem

The most difficult part of any stereo vision algorithm is the so called correspondence (matching) problem. The matching problem comprises determining the image points in two images corresponding to the same world points. The matching algorithms are either region (pixels) or contour based. The region based matchers match image regions (intensity levels) based on correlation techniques, while contour matchers match contours between images based on their geometrical properties. The region based matchers are generally less accurate and computationally much more demanding.

In this work we use straight line segments as image features. The reasons why we chose to use contours instead of regions are as follows

- Reduce the complexity by reducing the number of matches to be carried out, there are always less segments than points

- Explicitly take into account the continuity of contours, when two points correspond, often their neighbours correspond as well

- Geometric attributes measured on contour segments are richer and hence more discriminant than measurements of points. They can provide stronger matching constraints

- The position and orientation of a segment are generally measured with more precision than the position of an isolated point, therefore three dimensional reconstruction is more accurate

- The corresponding 3D structure obtained from 2D image segments reconstruction consists of a set of 3D segments, which is compact and structured enough to be efficiently used and manipulated

Regardless of the feature types, all stereo matching algorithms use geometric and heuristic constraints to reduce complexity and improve accuracy and reliability of finding matches. The fundamental geometric constraint is the so-called epipolar constraint [Hor86]. This constraint states that given a point in the left image its corresponding (match) point in the right image lies on the straight line (epipolar line) completely defined by the coordinates of the left point and the stereo configuration parameters (R, T). The problem is symmetric if the right point is considered. The epipolar line coefficients simply

follow by manipulating the stereo equations Eq.(5.2.1.3). In case the stereo cameras are parallel, the epipolar lines are horizontal (raster) lines in the images making the searching very simple. If the maximal and minimal depth of the scene is known in advance, than the epipolar line can be constrained much further, since in that case a segment of the epipolar line contains the match point. The end points of the segment are defined by the maximal and minimal depth. These geometric constraints are shown in Fig.5.2.1.4.

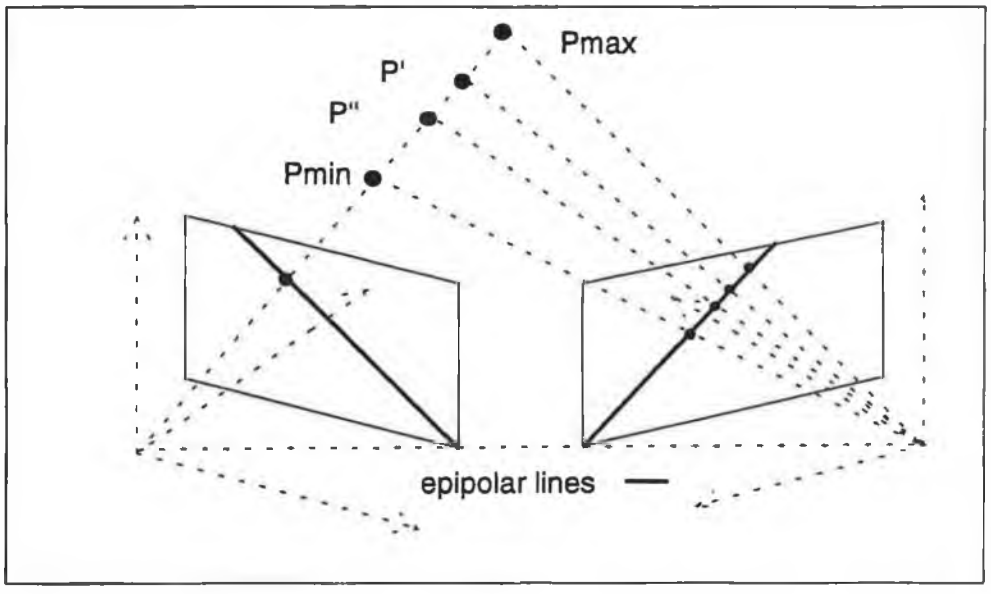

Figure 5.2.1.4 Epipolar Constraint

In general small displacements between stereo images minimise the matching problem but sacrifice the accuracy of the 3D reconstruction. This problem can be traded off by fusing and integrating information over the image sequence.

5.2.2 Structure-from-motion

A relative motion between a camera and an object can be used to obtain 3D information of the object. The process of computing 3D structure using image displacements resulting from motion is also called structure-from-motion. The approaches to structure-from-motion are categorised as optical flow based or feature based

approaches, the instantaneous positional velocity fields in conjunction with additional information or assumptions are used to compute the 3D structure. Feature based approaches consists of tracking the 2D features (edges) and interpreting the 3D structure in terms of the displacements of features in an image sequence.

The motion in the image plane caused by the camera motion in the static scene is given by the well-known optical flow equations [Hor86]. Let a camera move with an angular velocity ω and a translational velocity V (Fig.5.2.2.1).

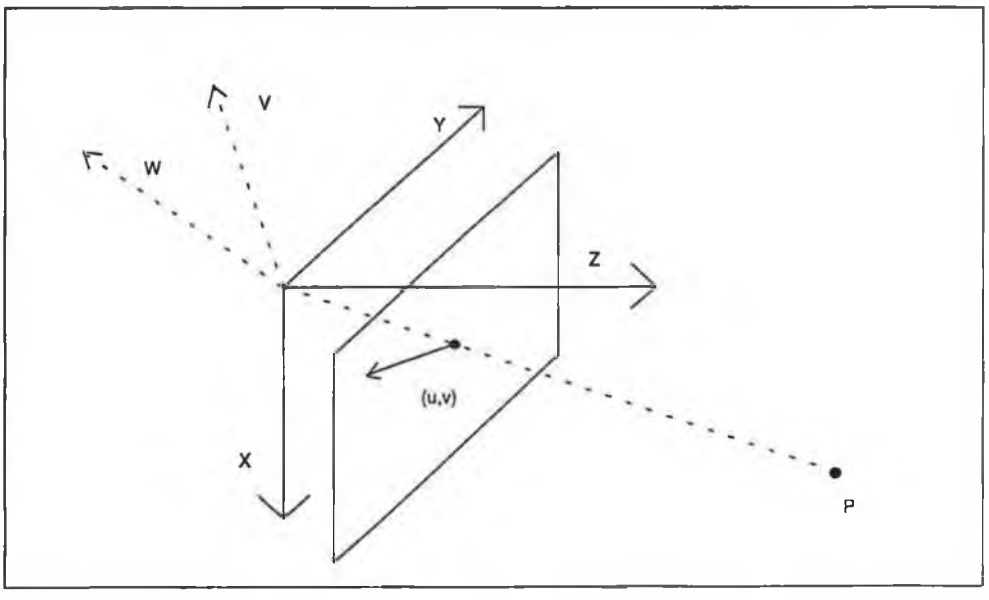

Figure 5.2.2.1 Optical Flow

The motion of a 3D point P in the camera coordinate frame is described by the equation

$$\frac{dP}{dt} = -V - \omega \times P \qquad \textbf{(5.2.2.1)}$$

Expanding this into components yields

$$\begin{aligned}\frac{dx}{dt} &= -V_x - \omega_y z + \omega_z y \\ \frac{dy}{dt} &= -V_y - \omega_z x + \omega_x z \\ \frac{dz}{dt} &= -V_z - \omega_x y + \omega_y x\end{aligned} \qquad \textbf{(5.2.2.2)}$$

Now, projecting *(x, y, z)* onto an ideal, unit focal length image (for the sake of simplicity)

$$u = \frac{x}{z} + C_x, \quad v = \frac{y}{z} + C_y \qquad \textbf{(5.2.2.3)}$$

takeing the derivatives of *(u, v)* with respect to time, substituting in from Eq.(5.2.2.2), discretizing with a sampling time ΔT and assuming slow changing camera velocity leads to the familiar equations of optical flow

$$\begin{bmatrix} u_{k+1} \\ v_{k+1} \end{bmatrix} - \begin{bmatrix} u_k \\ v_k \end{bmatrix} = \frac{\Delta T}{z} \begin{bmatrix} -1 & 0 & u_k \\ 0 & -1 & v_k \end{bmatrix} \begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix} + \Delta T \begin{bmatrix} u_k v_k & -(1+u_k^{\,2}) & v_k \\ (1+v_k^{\,2}) & -u_k v_k & u_k \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}$$

(5.2.2.4)

These nonlinear equations relate the depth *z* of the world point and the camera motion to the induced image displacement or optical flow. By measuring optical flow and assuming known motion the 3D structure can be recovered from the above equation.

However, no method exists for reliable computation of optical flow [Alo92], [Barr94], and all methods are highly demanding with respect to computational resources.

Feature-based approaches provide better estimates of the structure if a set of correctly matched features is available. In the feature based approaches the more critical problem is the detection of robust features and the solution of the correspondence (matching) problem. The feature based approaches will be more discussed in the sequel. As is the case with stereo vision, small interframe displacements (which is the fundamental assumption for structure-from-motion, since in that case Eq.(5.2.2.4) is a valid approximation of Eq.(5.2.2.3)) ease the correspondence problem (small displacements constrain possible image displacements Eq.(5.2.2.4)) but sacrifice the depth accuracy. Again, the overall accuracy can be increased by processing an image sequence. As a conclusion of this section, if we consider only two frames, the stereo vision accuracy is superior to the structure from motion accuracy thanks to its allowable larger displacement between images at the expense of much more complex matching problem.

5.3 3D Model reconstruction by fusing data from image sequences

The effective way to increase accuracy of the reconstructed 3D structure from noisy image data consists of fusing (combining) multiple observations and their uncertainties in order to reduce adverse impact of the image noise.

5.3.1 Relations to previous work

Review of previous work

One important approach in this direction is to fuse multiple 3D stereo frames (3D visual maps) obtained from stereo image sequences. The 3D visual maps usually consist of basic geometric primitives such as points, lines and planar patches. The previous work [Aya89], [Zha90], [Wen92] to cite a few, used estimation theory concepts to fuse multiple observations after resolving 3D primitive correspondences. Both Ayache and Faugeras [Aya89] and Zhang and Faugeras [Zha90] tracked 3D line segments over a

sequence of stereo image frames and used extended Kalman filter algorithm to fuse multiple 3D visual maps (stereo views) consisting of 3D line segments. To do the fusion, the displacement between successive stereo-pair coordinate frame is either assumed known or determined. Weng et al. [Wen92] used the batch-recursive least-squares algorithm to fuse multiple 3D maps in order to improve convergence rate.

Another important approach in this direction is based on structure from motion techniques. Much of the work in structure from motion techniques has concentrated on deriving depths of pixels, points and lines. The main idea of these approaches is based on the fact that the small interframe (camera) displacement implies the small image displacement. This fact permits to set tight search windows for correspondences, which in turn improves reliability and reduces computational load of finding matches. In order to get depth estimates, this stage is followed by applications of the structure-from-motion techniques to resulting correspondences. As was mentioned earlier, robust matching for simple features such as pixels and points is difficult for real scenes. Higher level image features such as lines can privide stronger constraints for matching and improve the robustness of the computations. They also can allow the larger interframe displacement while preserving efficient computation. These techniques are based on tracking image features over the image sequence.

Tracking of image features over image sequences have been mostly based on heuristics about motion of these features in the image plane. Since the scene structure and camera motion are involved in the features motion, the constant velocity motion models are not very reliable. The previous work [Matt89], [Cro92], [Der90], [Saw93], [Kum92], [She92] has identified Kalman filtering framework as an efficient way to tackle this problem. Mathhies and Kanade [Mat89] showed recovery of depth from lateral camera motion in a dense image sequence by tracking grey level values. Crowley et al. [Cro92] showed recovery of 3D structure of a scene by tracking 2D line segments. They employed constant velocity model to track line segments in the image plane and extended Kalman

filter to fuse data. Deriche and Faugeras [Der90] also employed the motion heuristics to track line segments. Sawhney and Hanson [Saw93] described a system which is based on constraining the camera motion and 3D structure, which permits modelling the image plane motion by an affine transformation. This transformation is used to track line segments in the image plane. Kumar and Hanson [Kum92] presented a technique to integrate 3D model, assuming an initial 3D model is known, by using an optic flow based line tracking algorithm. The main benefit arising from the tracking image features is that the tracking drastically simplifies matching problem. In fact if the tracking is accurate enough the matching can be accurately and reliably based on a simple nearest-neighbour matching algorithm. This is the point where Kalman filter gives a big advantage. If the tracking of a target by the Kalman filter is consistent [Bar88], then the prediction phase of the Kalman filter will give the truthful probability distribution of the next state, this also gives the probability distribution of the next measurement vector according to the measurement equations. These probability distributions can be used to associate the multiple observations to the multiple targets [Bar88] in the multitarget tracking environment, which resembles the image features tracking.

Main difficulties associated with previous work

The main difficulties associated with data fusion arising from stereo image sequences is the correspondence problem. The correspondence problem in this case is twofold. The first correspondence solution is related to matching 2D image features in order to reconstruct 3D structure or 3D visual map. The second correspondence problem is related to matching 3D features between 3D maps in a stereo sequence in order to improve 3D structure accuracy. These problems are difficult thanks to their combinatorial nature and computationally are very demanding.

The main difficulty associated with data fusion arising from structure-from-motion techniques is the feature tracking. As was mentioned above, the tracking of image

features over image sequences is based on assumptions about image motion heuristics such as constant velocity or acceleration motion. In order to give an insight into this modelling we will make a simple derivation with regard to a point motion modelling. Let us rewrite the optical flow equations derived previously (Eq. 5.2.2.4) as follows

$$\begin{bmatrix} u_{k+1} \\ v_{k+1} \end{bmatrix} = \begin{bmatrix} u_k \\ v_k \end{bmatrix} + \frac{\Delta T}{z} \begin{bmatrix} -1 & 0 & u_k \\ 0 & -1 & v_k \end{bmatrix} \begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix} + \Delta T \begin{bmatrix} u_k v_k & -(1+u_k^2) & v_k \\ (1+v_k^2) & -u_k v_k & u_k \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}$$

(5.3.1.1)

These nonlinear equations relate the depth z of the world point and the camera motion to the induced image displacement or optical flow. The tracking approaches based on the image motion heuristics attempt to approximate the complex motion equations Eq.(5.3.1.1) by a simple motion model such as constant velocity motion in the image plane.

For example such a model is given by

(5.3.1.2)

The measurement model is given

$$\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} u_k \\ \dot{u}_k \\ v_k \\ \dot{v}_k \end{bmatrix} + \begin{bmatrix} \varepsilon_5 \\ \varepsilon_6 \end{bmatrix} \qquad \textbf{(5.3.1.3)}$$

The state of the system is estimated by the Kalman filter based on the above models. The attempts are made to cover the complex nonlinear behaviour by a liner model and a random input. This model could be used to track points between frames. The similar models are derived for tracking parameters of higher level image features such as line segments. For example, in [Cro92], [Der90], [Saw93] each parameter a of a straight-line segment is modeled as

$$\begin{bmatrix} a_{k+1} \\ \dot{a}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & \Delta T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} a_k \\ \dot{a}_k \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \end{bmatrix} \qquad \textbf{(5.3.1.4)}$$

and the measurement model is given by

$$z_1 = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} a_k \\ \dot{a}_k \end{bmatrix} + \varepsilon_3 \qquad \textbf{(5.3.1.5)}$$

The essential benefit of tracking based on Kalman filter is that the predicted point position together with its predicted covariance matrix lies at the heart of matching algorithm. It is clear that it is very difficult to chose the proper level of the process noise covariance matrix even for a given scene and camera motion. On the one hand if the process covariance matrix is too small a lot of matches will be undiscovered. On the other hand by choosing the covariance matrix to large all advantages arising from tracking to allow fast, simple and reliable matching are lost. Another difficulty associated with this model is that it is not clear how to chose the initial conditions for Kalman filter. From this discussion it is obvious that a lot of filter parameters must be tuned "by hand". These difficulties associated with both techniques led us to find a method which would use good properties of both techniques namely, high accuracy of stereo vision and ease of matching supported by tracking.

New approach for 3D structure estimation

In this chapter we present a new approach for recursive 3D structure estimation from image sequences using known camera motion and generic assumptions about object structure. A rough distance estimate between a camera and an object is assumed known. The depth extent of an object is assumed to be small compared to the camera-object distance.

The novelty of this technique lies in the fact that this technique incorporates advantages of both previously described techniques (stereo image sequence and feature tracking) while minimizing their disadvantages. In fact a stereo image sequence (3D visual maps) is fused to achieve high accuracy while matching problem is highly simplified by tracking. The 3D (structure) matching problem is completely eliminated and the 2D (image) problem is simplified to the nearest-neighbour checking. The tracking is based on some assumptions about structure and the estimated structure, eliminating the need for motion models in the image plane and subsequently permitting arbitrary camera motion and allows larger interframe displacements. This leads to the considerable reduction in a number of images to be processed in order to achieve high accuracy.

The approach developed here is first to build an initial 3D model by using motion stereo which only requires a rough estimate of the distance between camera and objects, and then refine and extend it by viewing it over a sequence of images. Both the modelled and unmodelled features are tracked over the sequence based on the known camera motion, currently estimated 3D model for modelled features and the estimated average distance of the object for unmodelled features. Correspondences among features in the sequence are reliably and easily established by tracking. After resolving correspondences, the motion stereo triangulation process is used to obtain new observations for modelled features and for creation of new features. The Kalman filter is used to fuse new

observations with previous ones. This algorithm can be seen as a cyclic process, in which the 3D model structure undergoes a cycle of prediction, matching and updating

- In the prediction phase, the current state of the model is used to predict the state of the external world at the time that the next observation is taken

- In the matching phase, the observation is brought into correspondence with the prediction. Such matching requires the predictions and observations be transformed to the same coordinate system

- In the update phase, the observed information are integrated into existing model

The approach described here accounts for arbitrary camera motion and only needs a rough estimate of the distance between the camera and scene for the initialisation phase. The block diagram of the overall system is illustrated in Fig.5.3.1.2.

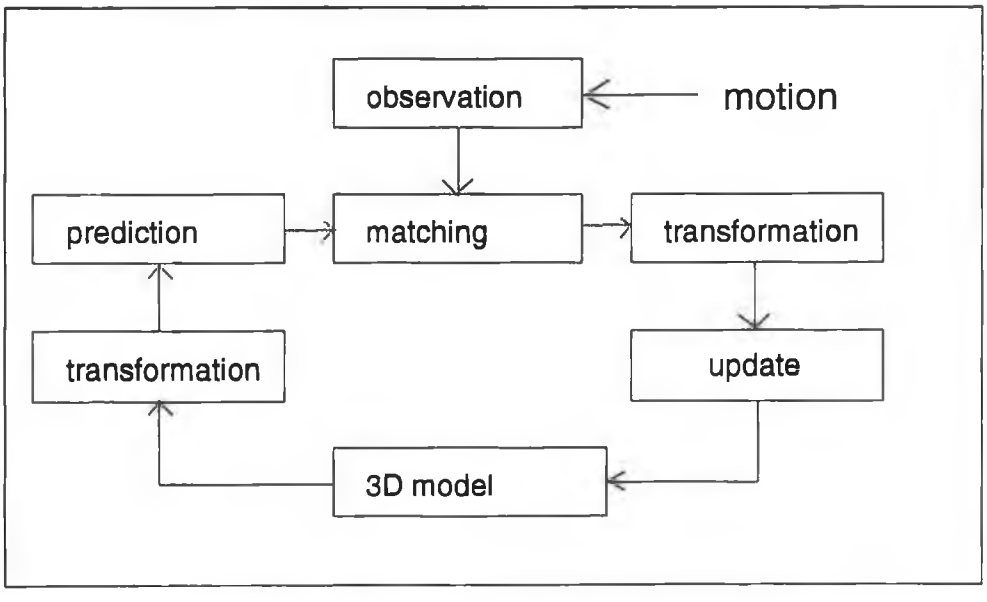

Figure 5.3.1.2 System for 3D Structure Reconstruction

5.3.2 3D Object model representation

A dynamic object model we deal with in this work is a list of geometric primitives given in a reference frame, which describes the object model at an instant in time

$$M(t) = \{P_1(t), P_2(t), \ldots, P_n(t)\} \qquad \textbf{(5.3.2.1)}$$

We assume that the actual objects can be accurately represented by these geometric primitives, and these primitives or their parameters can be measured by using vision sensors. Each geometric primitive describes a local part of the object as a conjunction of estimated geometric properties and their uncertainties. In addition, each primitive is assigned a unique identification label ID. The identification label ID acts as a name by which the primitive can be referred to

$$P(t) = \{X(t),\ W(t),\ ID\} \qquad \textbf{(5.3.2.2)}$$

The actual state of the object *M(t)* is observed trough a measurement process corrupted by the measurement noise. In order to apply an estimation algorithm such as the Kalman filter it is of crucial importance to have proper estimates of uncertainty in both system model and measurement model. The uncertainty estimates provide two crucial roles in the system, first they provide tolerance bounds for matching observations and predictions, second they provide the relative strengths of predictions and observations when calculating new estimates according the Kalman filter algorithm.

In this work the particular 3D geometric primitives used are 3D straight-line segments (polyhedral objects). The minimal representation of a 3D segment consists of six parameters. Here we use the representation given by segment endpoints and their covariance matrices, that is, the 3D segment is represented by

$$S_w = [p_1 \quad p_2 \quad W_1 \quad W_2] \qquad \textbf{(5.3.2.3)}$$

where $p_1(x_1,y_1,z_1)$ and $p_2(x_2,y_2,z_2)$ are endpoint coordinates in a reference frame, and W_1 and W_2 are their covariance matrices.

5.3.3 Vision-based measurements

It is assumed that the object model we want to reconstruct consists of a set of 3D straight-line segments as described previously. Taking into account a pin-hole camera model a 3D straight-line segment will be projected as a 2D straight-line segment. Hence, the basic measurement to be done is the measurement of 2D line segments in the image.

The minimal representation of a 2D straight-line segment requires four parameters. The basic representation comprises the coordinates of two endpoints in the image plane

$$S_i = [u_1 \quad v_1 \quad u_2 \quad v_2] \qquad \textbf{(5.3.3.1)}$$

Endpoints of a 2D image line segments (u, v) are related to endpoints of a 3D world line segments (x,y,z) through the camera model

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} s_x \frac{x}{z} + c_x \\ s_y \frac{y}{z} + c_y \end{bmatrix} + \begin{bmatrix} \varepsilon_u \\ \varepsilon_v \end{bmatrix} \qquad \textbf{(5.3.3.2)}$$

where $(\varepsilon_u, \varepsilon_v)$ are image measurement noise terms.

The image noise must be characterised. It is well known that the effect of image noise on the process of extracting straight-line segments is more pronounced along the direction of the edge then in the direction perpendicular to the edge. This is due to random effects which break edge lines into smaller segments. The uncertainties in the endpoints of a segment can be modelled with standard deviations σ_p along the edge and σ_v perpendicular to the edge in a coordinate system aligned with the line segment (Fig. 5.3.3.1).

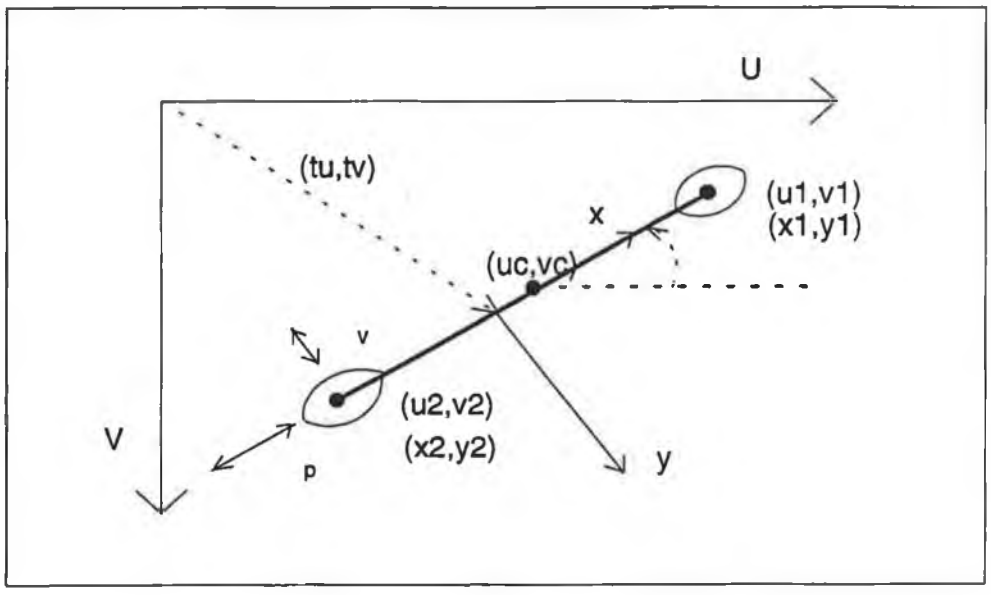

Figure 5.3.3.1 2D Segment Noise Modelling

In the aligned coordinate system *(x, y)* a covariance matrix of any endpoint is given by

$$Cov\left(\begin{bmatrix} x \\ y \end{bmatrix}\right) = \begin{bmatrix} \sigma_p^2 & 0 \\ 0 & \sigma_v^2 \end{bmatrix} \qquad \textbf{(5.3.3.3)}$$

using a reasonable assumption that the perpendicular and parallel noise terms are uncorrelated. Since, we need covariance matrices expressed in the original image coordinate frame *(u, v)*, they can be computed as follows. From Fig.5.3.3.1 the relation between *(u, v)* and *(x, y)* coordinates is given by

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_u \\ t_v \end{bmatrix} \qquad \textbf{(5.3.3.4)}$$

where θ is a rotation angle between *(x, y)* and *(u, v)* coordinate systems and (t_u, t_v) is an arbitrary constant translation vector between the coordinate systems. Then, the covariance matrix is given by

$$Cov\left(\begin{bmatrix} u \\ v \end{bmatrix}\right) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} \sigma_p^2 & 0 \\ 0 & \sigma_v^2 \end{bmatrix} \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}^T \qquad \textbf{(5.3.3.5)}$$

which leads to

$$W_p = Cov\left(\begin{bmatrix} u \\ v \end{bmatrix}\right) = \begin{bmatrix} \sigma_p^2 \cos^2(\theta) + \sigma_v^2 \sin^2(\theta) & (\sigma_p^2 - \sigma_v^2)\cos(\theta)\sin(\theta) \\ (\sigma_p^2 - \sigma_v^2)\cos(\theta)\sin(\theta) & \sigma_v^2 \cos^2(\theta) + \sigma_p^2 \sin^2(\theta) \end{bmatrix} \qquad \textbf{(5.3.3.6)}$$

Hence, in order to determine the uncertainties in image 2D segment measurements, we have to estimate the parallel σ_p and perpendicular σ_v deviations as given by Eq.(5.3.3.3). If we consider the effects of image noise on the process of extracting straight-line segments we can make following observations

- The perpendicular position of a line segment can be measured with precision. The accuracy is a function of the edge detection algorithm and is usually on the order of a pixel. We use the value for σ^2_v of one or two pixels

- Along the edge, the position of a line segment and consequently the position of its endpoints is much less precise, thanks to the random effects which make the polygonal approximation algorithm produce different polygons (shortened or extended

segments). It is difficult to find an analytical expression for these effects. By experimentation we found the proper value for σ^2_p to be between four and six pixels.

It is important to note that this noise model (Gaussian error distribution) cannot handle rough errors (outliers), for example such as breaking a large segment in two or more segments of the similar length, that often exist in reality. A way to cope with outliers will be discussed in the sequel.

We also use another line segment representation given in [Cro92], [Der90]. The 2D line segment is described by four parameters

$$S_2 = [u_c \quad v_c \quad l \quad \theta] \tag{5.3.3.7}$$

where (u_C, v_C) are coordinates of the segment midpoint, l is its length and θ is its orientation as given in Fig.5.3.3.1. This representation has a computational advantage for segment matching, since these parameters are less correlated when compared to other representations. Having endpoints of a segment and their covariances, the segment parameters given by Eq.(5.3.3.7) and their covariances can be computed as follows

$$u_c = \frac{u_1 + u_2}{2}, \quad v_c = \frac{v_1 + v_2}{2}, \quad \theta = \tan^{-1}\left(\frac{v_2 - v_1}{u_2 - u_1}\right), \quad l = \sqrt{(u_1 - u_2)^2 + (v_1 - v_2)^2}$$

(5.3.3.8)

The covariance matrix is computed as

$$W_1 = Cov\left(\begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \end{bmatrix}\right) = \begin{bmatrix} W_p & 0 & 0 & \\ 0 & 0 & & W_p \\ 0 & 0 & & \end{bmatrix}, \quad W_2 = Cov\left(\begin{bmatrix} u_c \\ v_c \\ l \\ \theta \end{bmatrix}\right) = JW_1J^T = \begin{bmatrix} \frac{W_p}{2} & 0 & 0 & \\ 0 & 0 & & 0 \\ 0 & 0 & 2\sigma_p^2 & \\ 0 & 0 & 0 & \frac{2\sigma_v^2}{l^2} \end{bmatrix}$$

(5.3.3.9)

where J is Jacobian matrix of Eq.(5.3.3.8) with respect to endpoint coordinates given by

$$J = \begin{bmatrix} \frac{\partial u_c}{\partial u_1} & \frac{\partial u_c}{\partial v_1} & \frac{\partial u_c}{\partial u_2} & \frac{\partial u_c}{\partial v_2} \\ \frac{\partial v_c}{\partial u_1} & \frac{\partial v_c}{\partial v_1} & \frac{\partial v_c}{\partial u_2} & \frac{\partial v_c}{\partial v_2} \\ \frac{\partial l}{\partial u_1} & \frac{\partial l}{\partial v_1} & \frac{\partial l}{\partial u_2} & \frac{\partial l}{\partial v_2} \\ \frac{\partial \theta}{\partial u_1} & \frac{\partial \theta}{\partial v_1} & \frac{\partial \theta}{\partial u_2} & \frac{\partial \theta}{\partial v_2} \end{bmatrix} \qquad \textbf{(5.3.3.10)}$$

and W_p is given by Eq.(5.3.3.6).

So far we have been concerned only with geometric attributes of segments. We also use one non-geometric parameter which may provide a strong constraint for matching depending on the scene. It is the average luminescence intensity level a_l along segment and its variance W_l. Finally the segment is represented by a list of two sets of redundant geometric parameters, its average luminescence intensity, their covariance matrices, a unique identification label *ID* used to identify the segment during the processing and C is a counter that keeps track of how many times the segment has been observed sequentially since its first appearance.

$$S = [u_1 \quad v_1 \quad u_2 \quad v_2 \quad u_c \quad v_c \quad l \quad \theta \quad a_l \quad W_1 \quad W_2 \quad W_l \quad ID \quad C] \qquad \textbf{(5.3.3.11)}$$

The *ID* is a unique index which makes it possible to identify a segment in different images. A segment in the current image inherits *ID* from its corresponding segment in the previous image after finding its match. Segments with no match found in the previous image get unique *ID* and are regarded as first observations of the new world segments. From the two corresponding segments we calculate a 3D segment. The 3D segment conserves this *ID*, which makes it possible to directly associate a reconstructed 3D

segment with a segment in the 3D model for fusion purposes without need for 3D segment matching.

The C is a counter which counts a number of times a segment has been observed sequentially since its first appearance. This number is used to make the system robust with respect to the rough image noise. A new 2D segment does not construct a new 3D segment unless it is observed sequentially a prescribed number of times (we usually use two, three or four in the experiments). This rule copes successfully with rough errors (outliers) arising from the polygonal approximation algorithm (Chapter 4).

In summary, each image in the sequence, after image processing stage (Chapter 4), is represented by a list of segments given by Eq.(5.3.3.12) and each segment is described by parameters given by Eq.(5.3.3.11).

$$I(t) = \{S_1(t), S_2(t), \ldots, S_n(t)\} \qquad \textbf{(5.3.3.12)}$$

Fig.5.3.3.2 shows the notation used in the subsequent derivations.

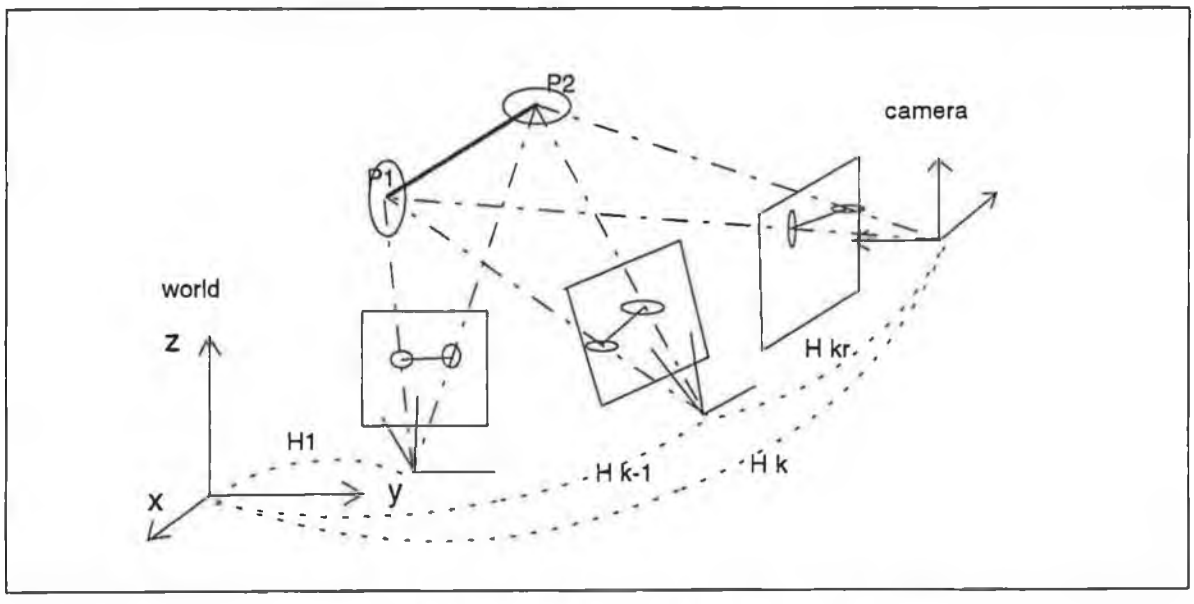

Figure 5.3.3.2 3D Model Reconstruction

The 3D model is computed in the world reference frame. Matrices $H_1,...,H_k$ denote the homogenous transformation matrices among the camera coordinate system and the world system. Matrices $H_{2r},...,H_{kr}$ denote relative transformations between successive frames. Either absolute or relative transformations are assumed known in this work (known camera motion), possibly their noisy values.

5.3.4 Prediction phase of tracking

Relations among coordinates of a 3D world point, their image projections and camera motion parameters, in two successive frames in an image sequence (motion stereo Fig.(5.2.1)) are given by

$$u_k = s_x \frac{x_k}{z_k} + C_x, \quad v_k = s_y \frac{y_k}{z_k} + C_y$$

$$u_{k+1} = s_x \frac{x_{k+1}}{z_{k+1}} + C_x, \quad v_{k+1} = s_y \frac{y_{k+1}}{z_{k+1}} + C_y$$

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ z_{k+1} \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} x_k \\ y_k \\ z_k \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

(5.3.4.1)

By manipulating above equations, we have

$$u_{k+1} = s_x \frac{s_y r_{11}(u_k - C_x) + s_x r_{12}(v_k - C_y) + s_x s_y r_{13} + s_x s_y \frac{t_x}{z_k}}{s_y r_{31}(v_k - C_x) + s_x r_{32}(v_k - C_y) + s_x s_y r_{33} + s_x s_y \frac{t_z}{z_k}} + C_x$$

$$v_{k+1} = s_y \frac{s_y r_{21}(u_k - C_x) + s_x r_{22}(v_k - C_y) + s_x s_y r_{23} + s_x s_y \frac{t_y}{z_k}}{s_y r_{31}(v_k - C_x) + s_x r_{32}(v_k - C_y) + s_x s_y r_{33} + s_x s_y \frac{t_z}{z_k}} + C_y$$

(5.3.4.2)

These equations relate the depth of the physical point z_k to its image projections $(u_k, v_k, u_{k+1}, v_{k+1})$ and camera motion (R, T) between successive frames.

These equations express so-called the epipolar geometric constraint for the stereo cameras. The epipolar constraint says, given a point in the first stereo image its corresponding (match) point in the second image necessarily belongs to a straight line (Eq.(5.3.4.2)) of the second image determined completely by the first point. This line is called the epipolar line associated with the point in the first image. This epipolar line model can be used to predict the image location of the point between current and next frame. By having an estimate of the point depth, its image coordinates in the current image and the camera motion, the next image location is predicted by Eq.(5.3.4.2).

Its covariance matrix up to the first-order approximation is predicted by

$$Cov\left(\begin{bmatrix} u_{k+1} \\ v_{k+1} \end{bmatrix}\right) = JWJ^T \qquad \textbf{(5.3.4.3)}$$

where J is Jacobian matrix of Eq.(5.3.4.2) with respect to (u_k, v_k, z_k) given by

$$J = \begin{bmatrix} \frac{\partial u_{k+1}}{\partial u_k} & \frac{\partial u_{k+1}}{\partial v_k} & \frac{\partial u_{k+1}}{\partial z_k} \\ \frac{\partial v_{k+1}}{\partial u_k} & \frac{\partial v_{k+1}}{\partial v_k} & \frac{\partial v_{k+1}}{\partial z_k} \end{bmatrix} \qquad \textbf{(5.3.4.4)}$$

and W is a covariance matrix of the (u_k, v_k, z_k) given by

$$W = \begin{bmatrix} W_p & 0 \\ 0 & Var(z_k) \end{bmatrix} \qquad \textbf{(5.3.4.5)}$$

where W_p is given by Eq.(5.3.3.6) and $Var(z_k)$ is the variance of the current depth estimate. In the same manner the motion uncertainties can be taken into account.

The line segments are predicted by predicting their endpoints as explained above and computing necessary representations and their uncertainties. In order to use prediction equations Eq.(5.3.4.2) it is necessary to have the depth estimate. This is especially important at the beginning of the tracking process (first stereo-pair), since for the rest of the tracking process currently available depth estimates are used. Here we use the task constraint of our system to determine the initial depth estimates. Since our robot-camera system is intended to reconstruct 3D structure of an object to be manipulated and detected a rough distance between the initial robot-camera position and the object is always known. So, in order to initiate our algorithm we only need to supply the rough estimate of the distance between the camera and the object and its variance (which takes into account the error in the estimate and the depth extent of the object).

$$z_0 = z(0), \quad \sigma^2_{z(0)} = \mathrm{Var}(z_0) \qquad \textbf{(5.3.4.5)}$$

By experimentations it was found that the error of up to 20% for our set-up can be handled. For this initialization to work properly it is also assumed that the depth extent of the object is small when compared to the distance from the camera, which is satisfied for most objects we deal with. By experimentations it was found that the ratio of the distance to the depth extent of up to 0.2 meets this assumption. The depth estimate Eq.(5.3.4.5) is used for the initial 3D model reconstruction (first stereo-pair). For the

tracking of new yet unmodelled features entering the field of view during motion, the depth estimate is computed as the average depth between the camera and currently estimated 3D model. The prediction of the already modelled feature is based on the current depth estimate belonging to that feature.

The 3D model is reconstructed in the world frame. The prediction phase deals with the model given in the camera frames. The currently estimated world model must be transformed to the camera frames for the purpose of the prediction. This is done using known camera motion. From the Fig.5.3.3.2 these transformations are given by

$$p_{ck} = H_k^{-1} p_w, \quad Cov(p_{ck}) = H_k^{-1} Cov(p_w) H_k^{-1^T} \qquad \textbf{(5.3.6.1)}$$

where p_{ck} is a 3D position vector of the segment endpoint in the current camera frame, $Cov(p_{ck})$ is its covariance matrix, H_k is the current camera transformation matrix, p_w is a position vector of the segment endpoint in the world frame and $Cov(p_w)$ is its covariance matrix. Having all segments transformed to the current camera frame their appearances in the next camera position are predicted by Eq.(5.3.4.2).

5.3.5 Matching phase of tracking

The matching phase determines the most likely association of observed and predicted primitives based on the similarity between the predicted and observed properties. The mathematical measure for such similarity is to determine the difference between these properties. The threshold value must be defined on the difference to decide whether these properties are similar or not. Since the geometric properties are subject to noise, which can be estimated as it was described previously, it is essential to take into account the noise estimates of geometric properties when choosing threshold values.

The most reliable way to take into account noise effects in comparing properties is the so-called Maholobonis distance [Bar88] which permits us to define a dynamic threshold for comparison depending on the noise level. So as a match measure we use the Maholobonis distance between predicted segment parameters from the previous frame and observed segment parameters from the current frame

$$d = (S_p - S_o)^T (W_p + W_0)^{-1} (S_p - S_o) \qquad \textbf{(5.3.5.1)}$$

where S_p, S_O are geometric segment parameters (midpoint, length and orientation) given by Eq.(5.3.3.7) and possibly luminescence intensity incorporated, W_p W_O are their covariance matrices computed as described previously. This measure should be computed for each predicted segment against observed segments. The smallest Maholobonis value Eq.(5.3.5.1) below prescribed threshold is chosen as the match (nearest-neighbour). After this all observed segments are labelled either as new observations of the already modelled world (matched segments) or as observations of the new yet unmodelled world entities (unmatched segments). If an observed segment is matched to a predicted segment, then it inherits the ID (identification label) from its matched predicted segment. Unmatched segments are assigned unique ID.

The threshold value depends on the errors in the measurement and prediction phases. If all errors are assumed Gaussian, then the Maholobonis distance is a chi-square distributed variable. By looking up the χ^2 distribution table, a threshold on this variable can be chosen giving the desired confidence level in accepting a match. Although in this case errors are not exactly Gaussian, we use this way to set up the threshold value in our implementation. The match measure Eq.(5.3.5.1) is a chi-square variable with four degrees of freedom. We usually use the threshold value of 9.49 for the desired confidence of 95%.

A naive matching algorithm yields a *O(NN)* complexity to match *N* segments, assuming there are *N* segments in booth images. However, the matching process may be very slow, especially when there is a large number of segments. This is because the computation of the Mahalanobis distance (Eq.(5.3.5.1)) involves a matrix inversion (4x4 matrix, the luminescence attribute if used is uncorrelated with geometric attributes) and is relatively expensive. Since the computational complexity of the matching is a big part of the whole algorithm complexity, we use the indexing scheme for structuring image segments as presented in Chapter 4 (Section 4.5). This indexing scheme together with computed uncertainties of predicted and observed segments give the complexity of matching equal to *kO(N)*, where *N* is the number of segments and *k* is a small number compered to *N* depending on the segments distribution over an image. The computed uncertainty of a predicted segment midpoint is used to define an image search window in which the midpoint of its corresponding observed segment should lie with a desired confidence. By defining the Maholobonis distance for the segment midpoint coordinates we have

$$\begin{pmatrix} x_p^m - x_o^m \\ y_p^m - y_p^m \end{pmatrix}^T (W_p^m + W_o^m)^{-1} \begin{pmatrix} x_p^m - x_o^m \\ y_p^m - y_p^m \end{pmatrix} \leq d_m \qquad \textbf{(5.3.5.2)}$$

where (x_p^m, y_p^m) and (x_o^m, y_o^m) are the midpoint coordinates of predicted and observed segments and W_p^m and W_o^m are their covariance matrices. The Eq.(5.3.5.2) presents the uncertainty ellipse within which a match to a predicted segment should lie with the desired confidence. This is shown in Fig.5.3.5.1. In order to find the predicted segment match it is enough to test only observed segments whose midpoints belong to the uncertainty ellipse, since the rest of segments cannot be matches. The image search windows to be searched is defined by the size of the uncertainty ellipse. In order to compute the parameters of the ellipse Eq.(5.3.5.2) for each predicted segment, the value of $W_O{}^m$ covariance matrix is necessary. Since this matrix in general is different for each observed segment, it is

approximated by a fixed worst-case matrix. The worst-case matrix is simply defined by the expresion for $W_o{}^m$ matrix given by Eq.(5.3.3.9).

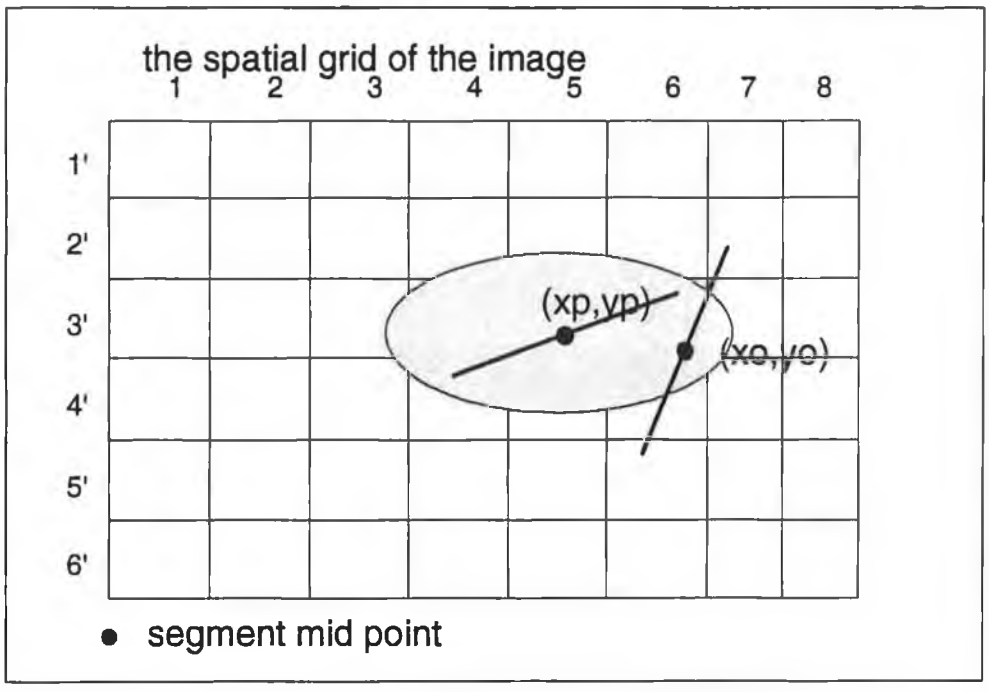

Figure 5.3.5.1 Image Search Window

Moreover, in our experimentations we found no significant difference in performance between usage of the full covariance matrices and their diagonal terms only (setting off-diagonal elements to zero) in computing Eq.(5.3.5.1) at the benefit of rather big computational savings, since the matrix inversion is fully avoided. This approximation is justified, by weak correlations among segment parameters by using second segment representation (midpoint, length and orientation). These two techniques make matching process extremely fast.

5.3.6 Updating phase of tracking

Having resolved correspondences among segments in two successive frames, 3D model refinement and expansion are performed.

Model refinement

The Model updating is carried out by the Kalman filter algorithm. Since 3D measurements and their uncertainties are defined in the current camera coordinate system, they must be transformed to the world reference system. From Fig.5.3.3.2, this is computed according to

$$p_{wk} = H_k p_{ck}, \quad Cov(p_{wk}) = H_k Cov(p_{ck}) H_k^T \qquad \textbf{(5.3.6.1)}$$

where p_{ck} is a 3D measurement in the camera frame, $Cov(p_{ck})$ is its covariance matrix, H_k is the current camera transformation matrix, p_{wk} is a transformed measurement and $Cov(p_{wk})$ is its covariance matrix.

Having all measurements transformed to the common reference system, the Kalman filter algorithm is applied to the following estimation model

The state transition model is given by

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ z_{k+1} \end{bmatrix} = I \begin{bmatrix} x_k \\ y_k \\ z_k \end{bmatrix} \qquad \textbf{(5.3.6.2)}$$

where (x_k, y_k, z_k) are coordinates of endpoints of 3D segments in the world frame, I is identity matrix, because we are dealing with constant parameters (stationary object in the world reference frame).

The measurement model

The measurement model of (x_k, y_k, z_k) coordinates is given by

$$m_k = I\begin{bmatrix} x_k \\ y_k \\ z_k \end{bmatrix} + \begin{bmatrix} \varepsilon_x \\ \varepsilon_y \\ \varepsilon_z \end{bmatrix}, \quad Cov\left(\begin{bmatrix} \varepsilon_x \\ \varepsilon_y \\ \varepsilon_z \end{bmatrix}\right) = R_k \qquad \textbf{(5.3.6.3)}$$

where m_k is the measurement vector given by Eq.(5.3.6.1), I is identity matrix, $(\varepsilon_x, \varepsilon_y, \varepsilon_z)^T$ is the measurement noise vector and R_k is its covariance matrix given by Eq.(5.3.6.1).

The system model and measurement model, together with their covariances are used in the the standard Kalman filter equations to update positions of 3D segments and their covariances.

Convergence rate analysis

For the particular system and measurement model Eq.(5.3.6.2) and Eq.(5.3.6.3) the Kalman filter is equivalent to the recursive weighted least-squares estimation algorithm. This gives a possibility to assess its convergence rate as a function of the number of processed frames.

The recursive Kalman filter optimization of the above system is equivalent to the following weighted least-squares batch optimization problem

$$\min_p \sum_{k=1}^{n} (m_k - I_k p)^T R_k^{-1} (m_k - I_k p)$$

The solution vector and its covariance matrix is given by

$$p = \left[\sum_{k=1}^{n} R_k^{-1}\right]^{-1}\left[\sum_{k=1}^{n} R_k^{-1} m_k\right],\ Cov(p) = \left[\sum_{k=1}^{n} R_k^{-1}\right]^{-1}$$

Assuming that the covariance of successive stereo measurements are constant, resulting covariance is given by

$$Cov(p) = \frac{R_k}{n}$$

Thus, the covariance decreases inversely to the number of processed frames, this has been confirmed experimentally.

Model expansion

The newly observed 2D segments which have been tracked successfully over a prescribed number of frames are added to the 3D model by applying stereo motion equations Eq.(5.3.6.1) to the image segments between current and previous observations. Also, the 3D segments within a model which have not been updated for a prescribed number of times are removed from the model. This controls the model size and removes unreliable segments.

5.4 Application of algorithm

Consider a camera mounted on a robot arm as shown in Fig.5.4.1. The camera is rigidly mounted on the robot end-effector. The camera pose with respect to the end-effector is described by the transformation matrix H (Chapter 3). The end-effector pose with respect to the robot base is described by the transformation matrix T (Chapter 2). The matrix H is determined by the eye/hand calibration procedure (Chapter 3). The

matrix T is provided by the robot controller. The camera position with respect to the robot base is given by

$$H_c = TH \tag{5.4.1}$$

This matrix defines the camera motion as a result of the robot arm motion. Our goal is to estimate 3D structure of an object by using above described approach.

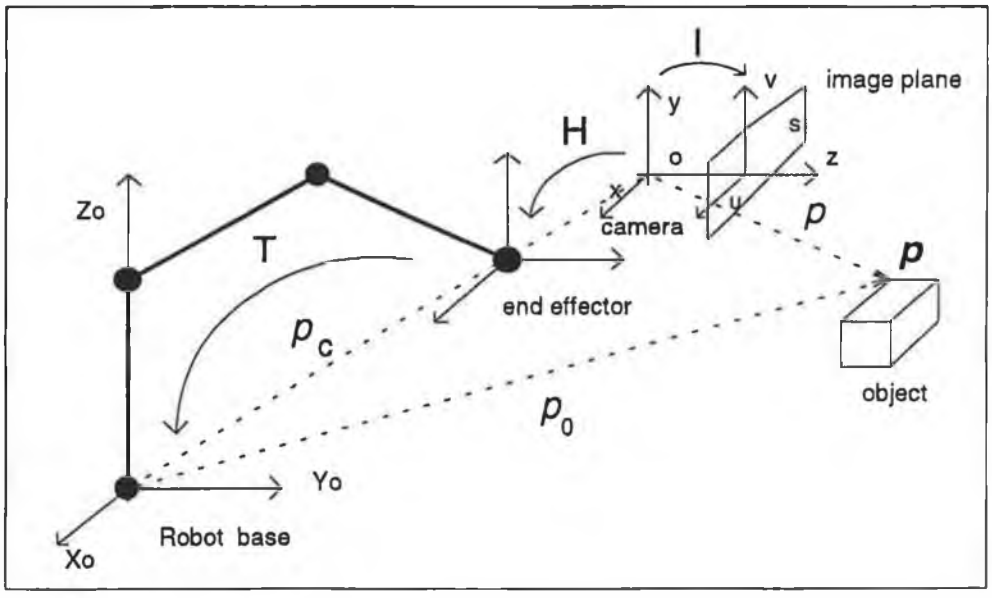

Figure 5.4.1 Robot-Camera Kinematic Scheme

Experimental results

Our experimental set-up consists of a CCD camera with 16mm lens mounted on a PUMA-560 robot arm. A frame-grabber is fitted to the PC-386 which has 512x512 pixels spatial resolution. The PC is interfaced to the robot controller by a RS-232 serial link. This allows us to control the motion of the robot and to acquire the robot state information (Chapter 2). The necessary image processing software such as edge detection, edge linking and edge segmenting was realised according to the material presented in Chapter 4. The Eye/Hand and camera calibration software was realised according the material presented in Chapter 3.

The objects used for our experiments are located at a distance of about 40 cm from the camera. The camera follows prescribed trajectories. Camera displacements between consecutive frames are between 3-5 cm and camera rotations are between 0-5 degrees. The robot moves and stops for processing. On PC-386 it takes a few seconds to execute the algorithm per image for a few tens of segments excluding low-level image processing (filtering and edge detection).

For the system to be initiated, a set of initial parameter values must be supplied to it, as previously described in details.

In Table 5.4.1 we give a summary of these parameters and their particular values used in the experiments presented here

σ resolution of Gaussian filter	ε tolerance for polygonal approx.	σ^2_D parallel segment uncertainty	σ^2_V normal segment uncertainty	Z_0 object-camera distance estimate	σ^2_Z object-camera distance uncertainty
1	2 pixels	6 pixels	2 pixels	400 mm	3600

Table 5.4.1 System Parameters

Here experimental results for four test objects are given. First one is an artificial cubic like object with known ground truth in terms of 3D segment lengths with "good" geometric and reflectance properties. The rest of the objects are industrial parts.

First test object

Fig.5.4.2 shows a raw image of the cube. Fig.5.4.3 shows the reconstructed 3D segments of the cube after ten processed images with their labels superimposed. Fig.5.4.4 and Fig.5.4.5 show the length estimation of the segment labelled by number 5 and its variance over 10 frames. Table 5.4.2 shows true lengths of some segments, their estimated lengths and their variances after first stereo-pair is processed. Table 5.4.3 shows true lengths of some segments, their estimated lengths and their variances after ten processed images. From the presented results it can be seen that the 3D structure which is recovered very quickly converges to precision on the order of a millimetre. All 3D segments have the similar convergence rate and it usually takes five or six images to achieve steady state.

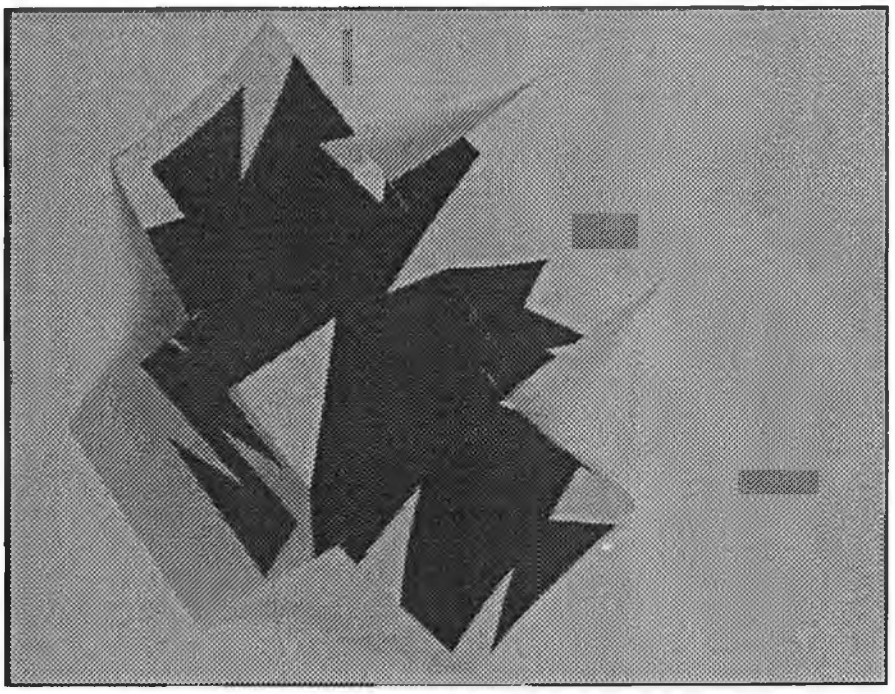

Figure 5.4.2 Raw Image of Object, Top View

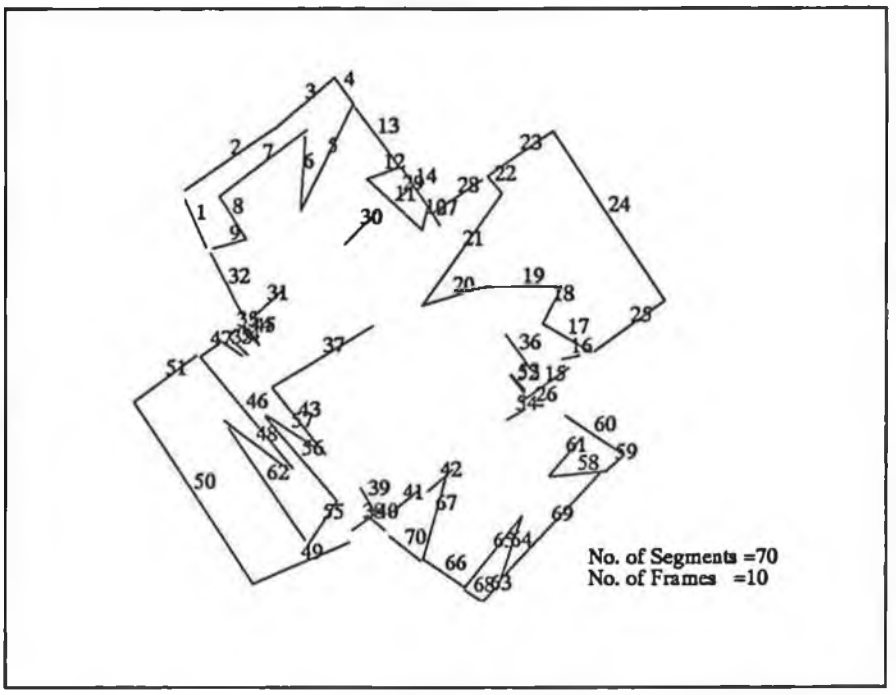

Figure 5.4.3 Visually Reconstructed Model after 10 Frames,

Projection on Camera

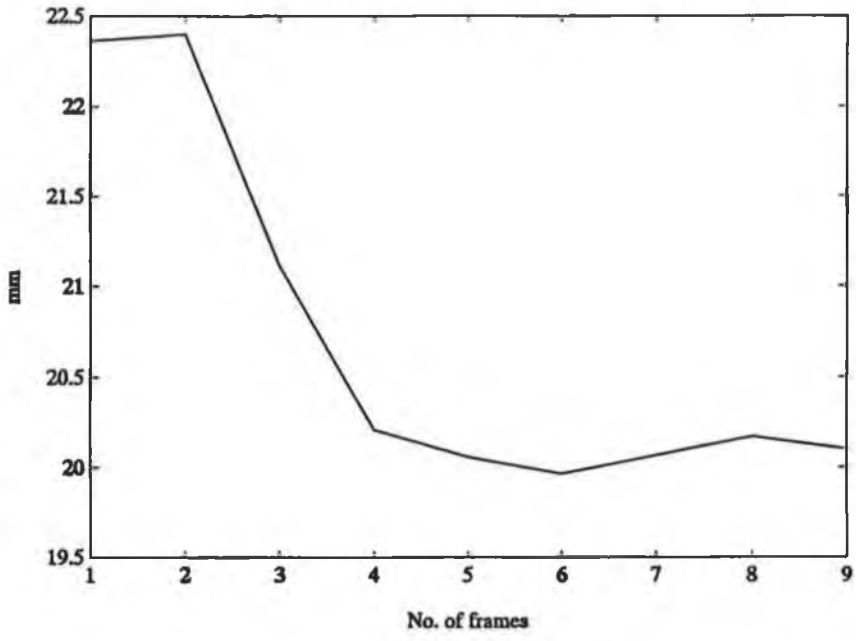

Figure 5.4.4 Length Estimation, Segment No.5

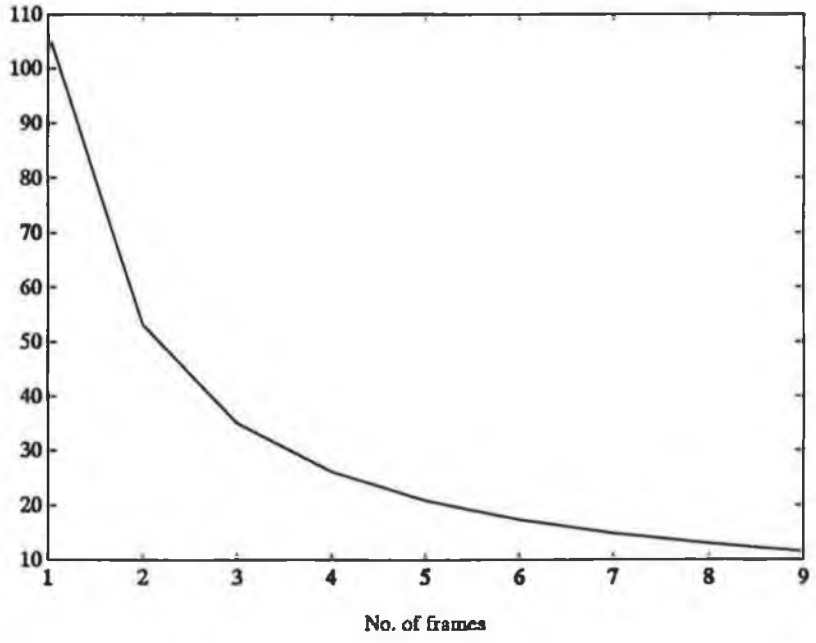

Figure 5.4.5 Length Variance, Segment No.5

Segment No.	l(mm)	ltrue(mm)	Cov(l)
1	14	10	105
2	21	25	108
3	18	15	100
4	13	12	110
5	22	20	105
6	11	15	98
7	21	23	104
8	14	12	114
9	9	5	108
10	12	9	101

Table 5.4.2

Estimated and True Lengths of some Segments and Their Variances after First Stereo-Pair

Segment No.	l(mm)	ltrue(mm)	Cov(l)
1	9	10	10
2	24	25	11
3	16	15	12
4	11	12	13
5	20	20	12
6	14	15	13
7	23	23	11
8	12	12	14
9	7	5	10
10	8	9	13

Table 5.4.3

Estimated and True Lengths of some Segments and Their Variances after 10 Frames

Second test object

Fig.5.4.6 shows a raw image of a connector. Fig.5.4.7 shows the reconstructed connector model after the tenth image with their labels. Fig.5.4.8 and Fig.5.4.9 show the length estimation of the segment labelled by number 2 and its variance over 10 frames. Table 5.4.4 shows true lengths of some segments, their estimated lengths and their variances after first stereo-pair is processed. Table 5.4.5 shows true lengths of some segments, their estimated lengths and their variances after ten processed images.

Segment No.	l(mm)	ltrue(mm)	Cov(l)
37	13	18	180
36	19	16	185
35	29	30	188
1	26	30	175
2	24	16	180
9	9	16	182

Table 5.4.4

Estimated and True Lengths of some Segments and Their Variances after First Stereo-Pair

Segment No.	l(mm)	ltrue(mm)	Cov(l)
37	18	18	22
36	17	16	18
35	30	30	25
1	29	30	20
2	15	16	20
9	11	16	22

Table 5.4.5

Estimated and True Lengths of some Segments and Their Variances after 10 Frames

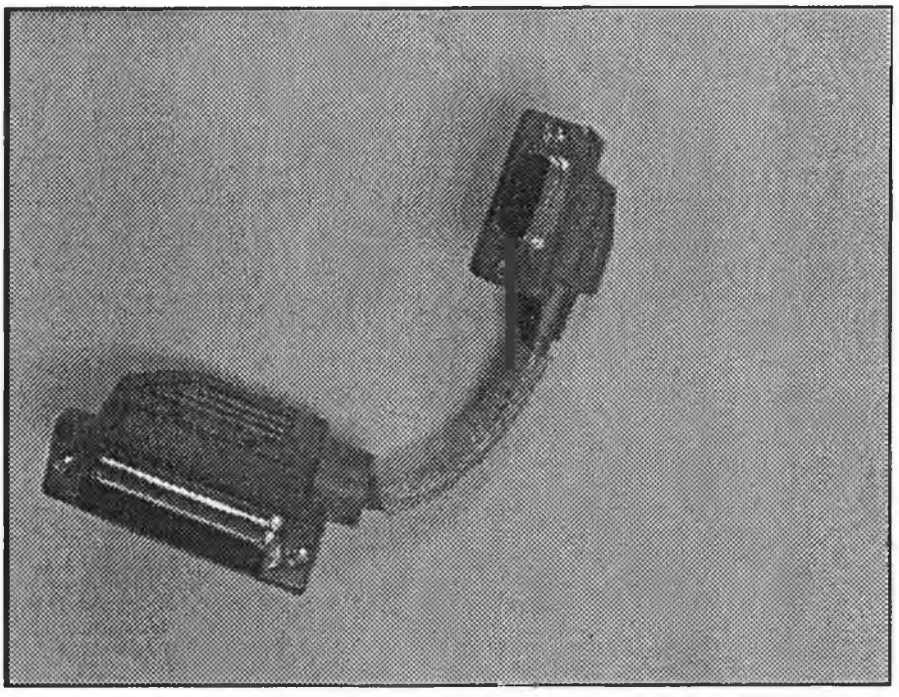

Figure 5.4.6 Raw Image of Object, Top View

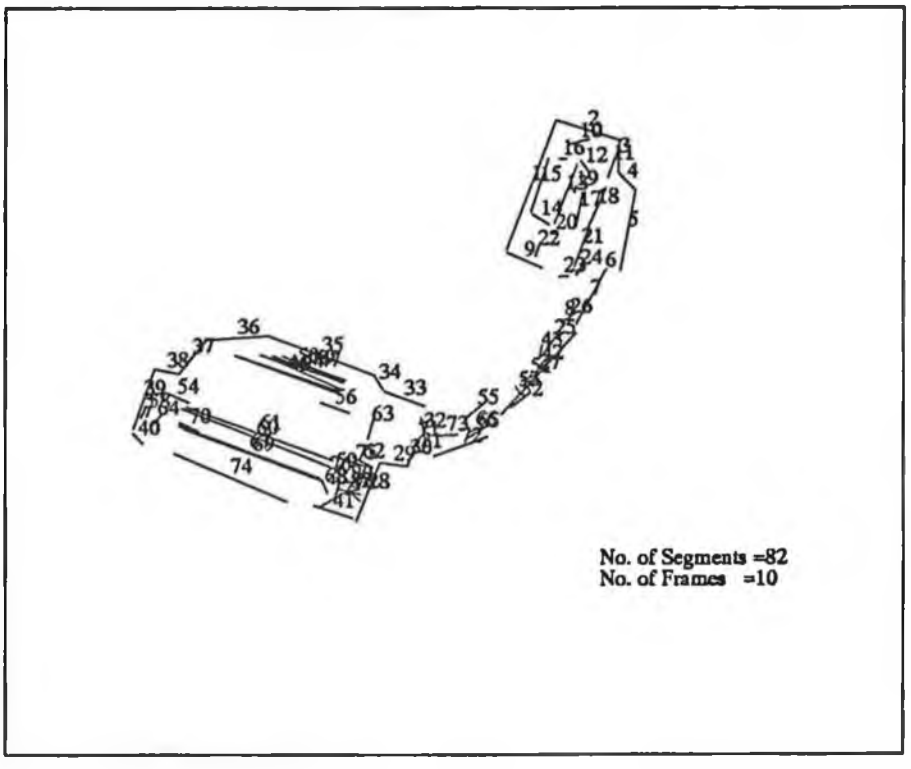

Figure 5.4.7 Visually Reconstructed Model after 10 Frames,

Projection on Camera

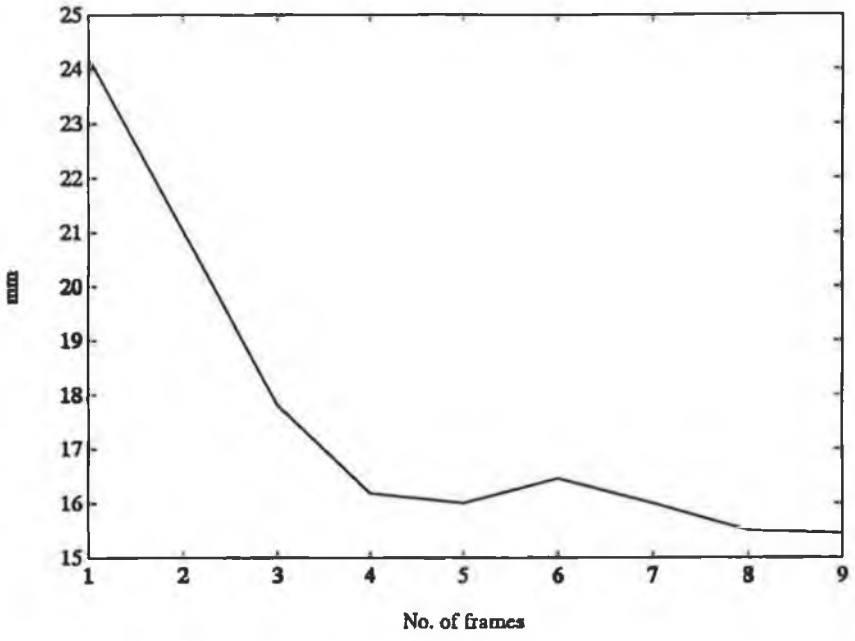

Figure 5.4.8 Length Estimation, Segment No.2

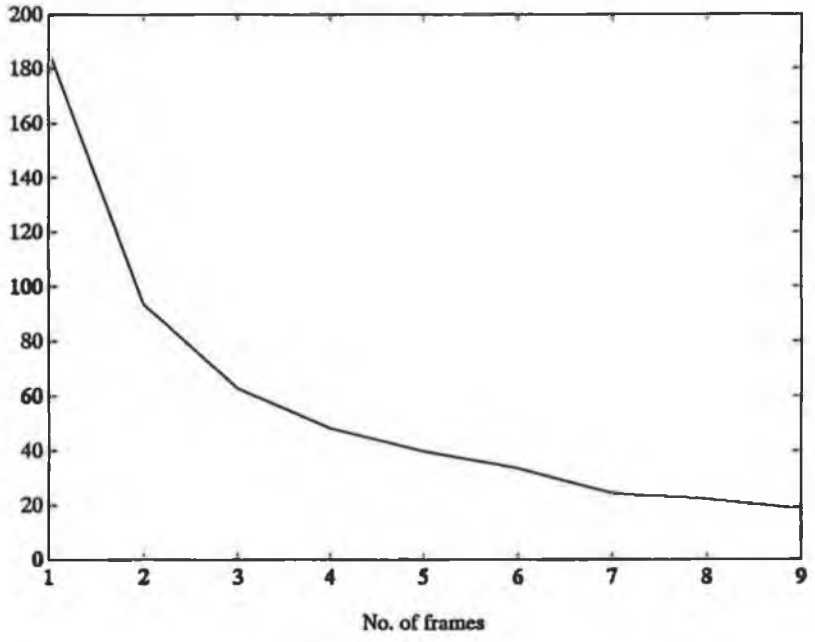

Figure 5.4.9 Length Variance, Segment No.2

Third test object

Fig.5.4.10 shows a raw image of a switch box. Fig.5.4.11 shows the reconstructed switch box model after the tenth image with their labels. Fig.5.4.12 and Fig.5.4.13 show the length estimation of the segment labeled by number 3 and its variance over 10 frames. Table 5.4.6 shows true lengths of some segments, their estimated lengths and their variances after first stereo-pair is processed. Table 5.4.7 shows true lengths of some segments, their estimated lengths and their variances after ten processed images.

Segment No.	l(mm)	ltrue(mm)	Cov(l)
7	13	11	100
8	32	30	90
9	46	45	108
10	10	11	110
32	15	18	90
15	14	18	95

Table 5.4.6

Estimated and True Lengths of some Segments and Their Variances after First Stereo-Pair

Segment No.	l(mm)	ltrue(mm)	Cov(l)
7	11	11	11
8	30	30	10
9	45	45	12
10	11	11	12
32	16	18	10
15	17	18	11

Table 5.4.7

Estimated and True Lengths of some Segments and Their Variances after 10 Frames

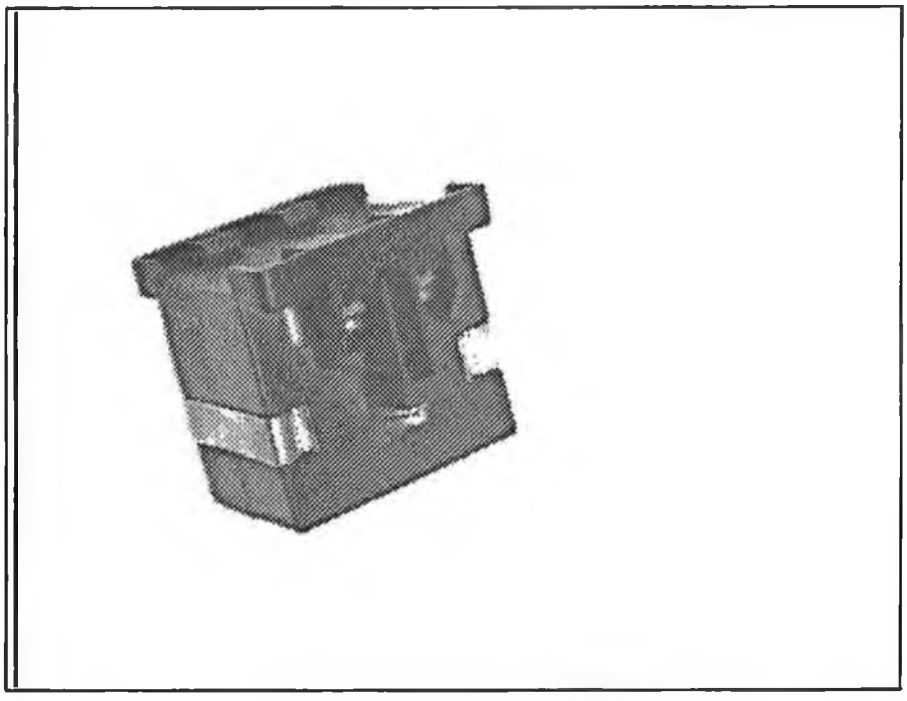

Figure 5.4.10 Raw Image of Object, Top View

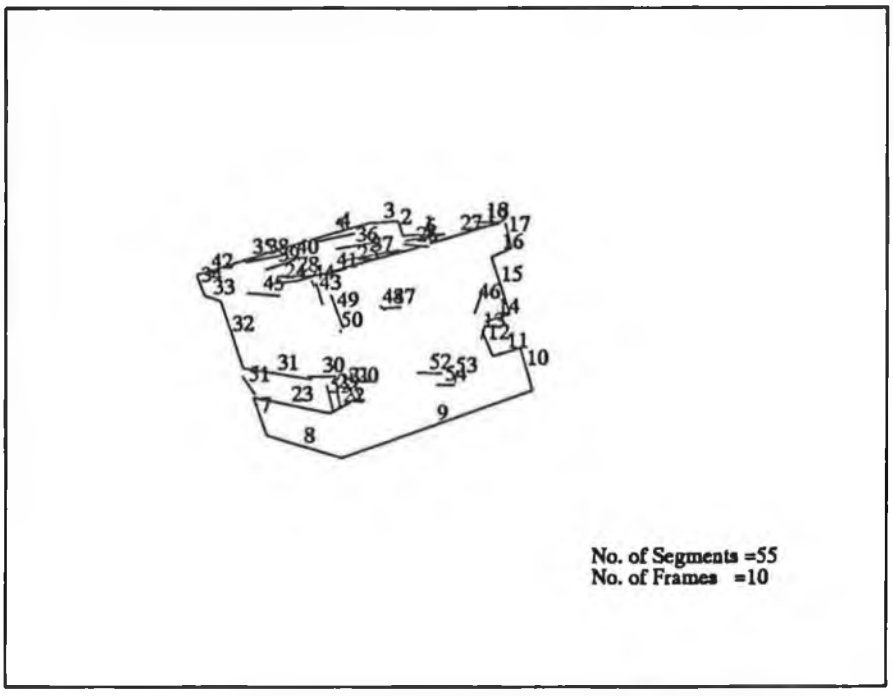

Figure 5.4.11 Visually Reconstructed Model after 10 Frames, Projection on Camera

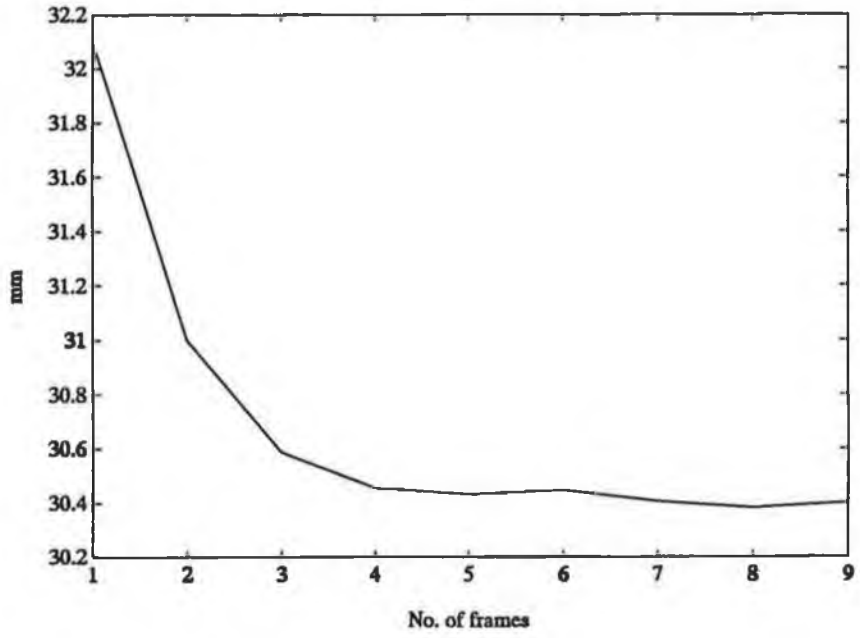

Figure 5.4.12 Length Estimation, Segment No.8

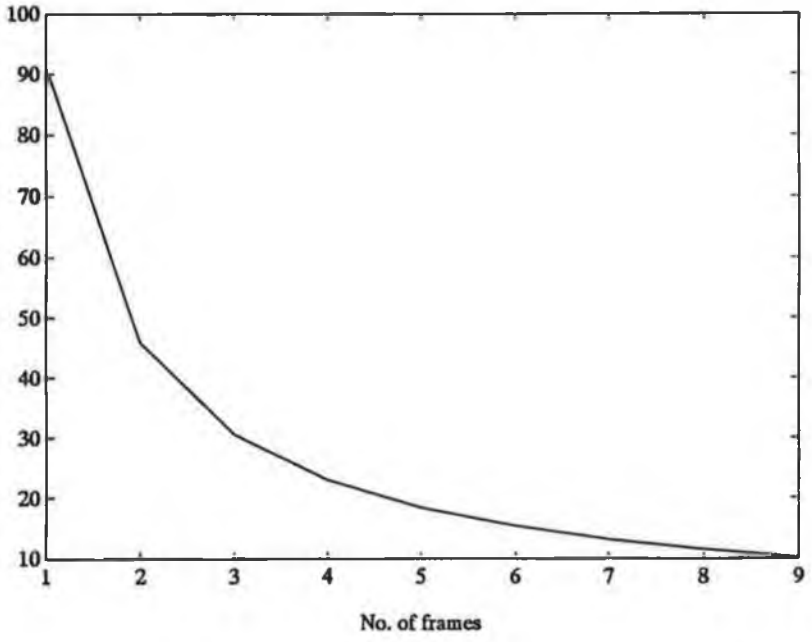

Figure 5.4.13 Length Variance, Segment No.8

Fourth test object

Fig.5.4.14 shows a raw image of two switch boxes. Fig.5.4.15 shows the reconstructed 3D model after the tenth image with their labels. Fig.5.4.16 and Fig.5.4.17 show the length estimation of the segment labeled by number 3 and its variance over 10 frames. Table 5.4.8 shows true lengths of some segments, their estimated lengths and their variances after first stereo-pair is processed. Table 5.4.9 shows true lengths of some segments, their estimated lengths and their variances after ten processed images.

Segment No.	l(mm)	ltrue(mm)	Cov(l)
30	13	18	70
29	25	30	70
6	34	36	55
7	14	11	55
51	20	25	65
52	38	35	182

Table 5.4.8

Estimated and True Lengths of some Segments and Their Variances after First Stereo-Pair

Segment No.	l(mm)	ltrue(mm)	Cov(l)
30	15	18	8
29	28	30	10
6	36	36	6
7	11	11	6
51	24	25	8
52	36	35	10

Table 5.4.9

Estimated and True Lengths of some Segments and Their Variances after 10 Frames

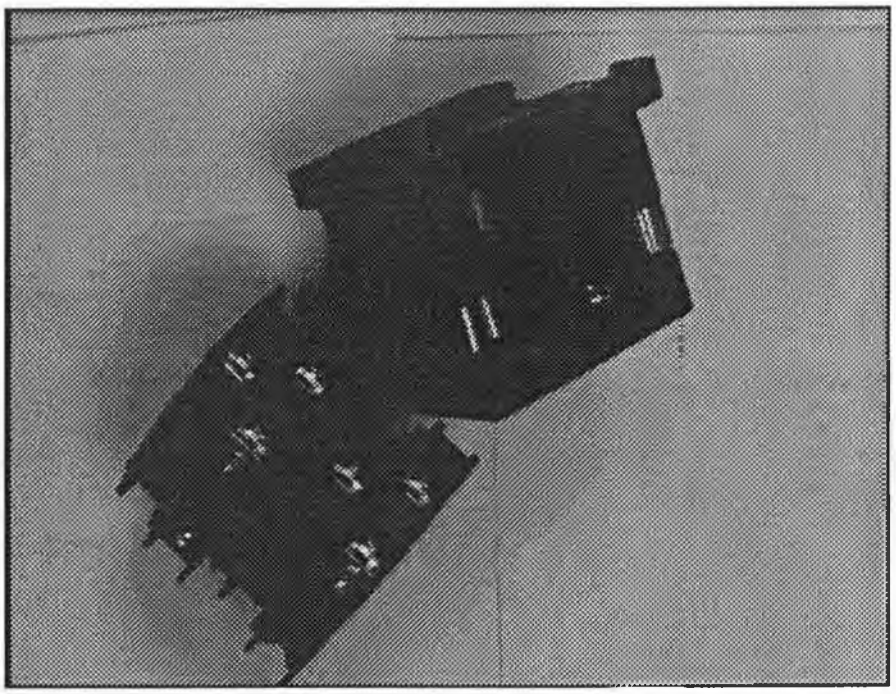

Figure 5.4.14 Raw Image of Object, Top View

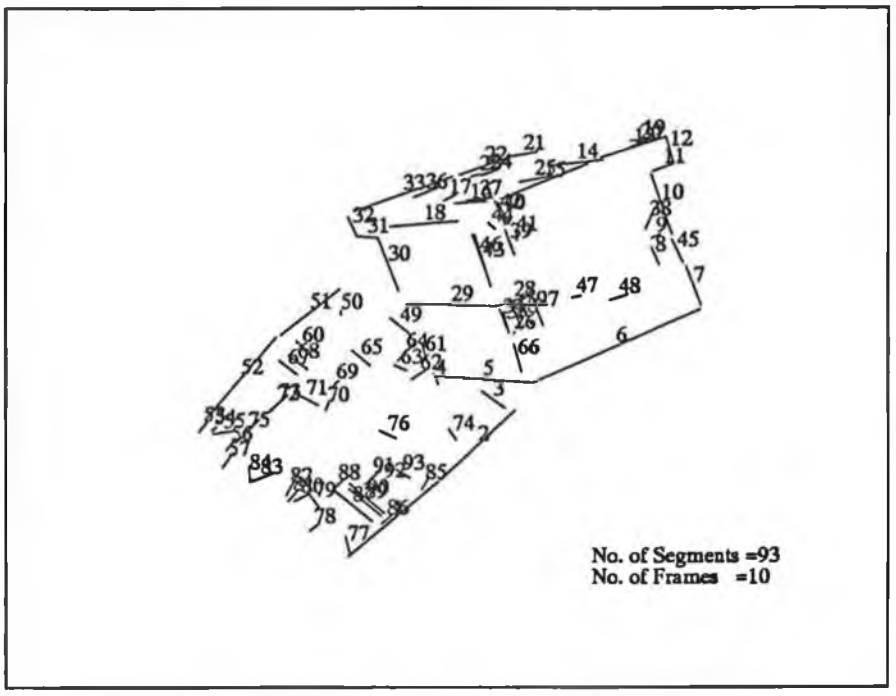

Figure 5.4.15 Visually Reconstructed Model after 10 Frames, Projection on Camera

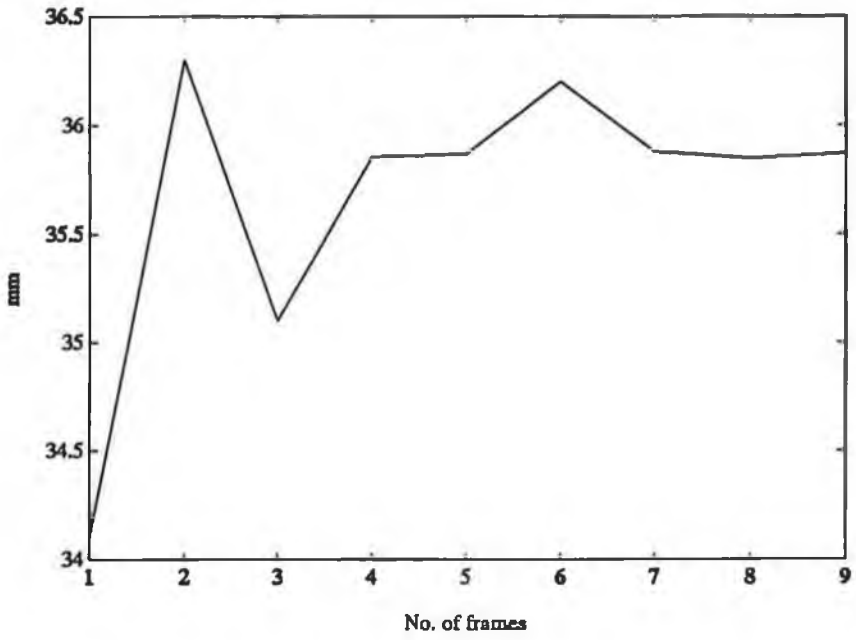

Figure 5.4.16 Length Estimation, Segment No.6

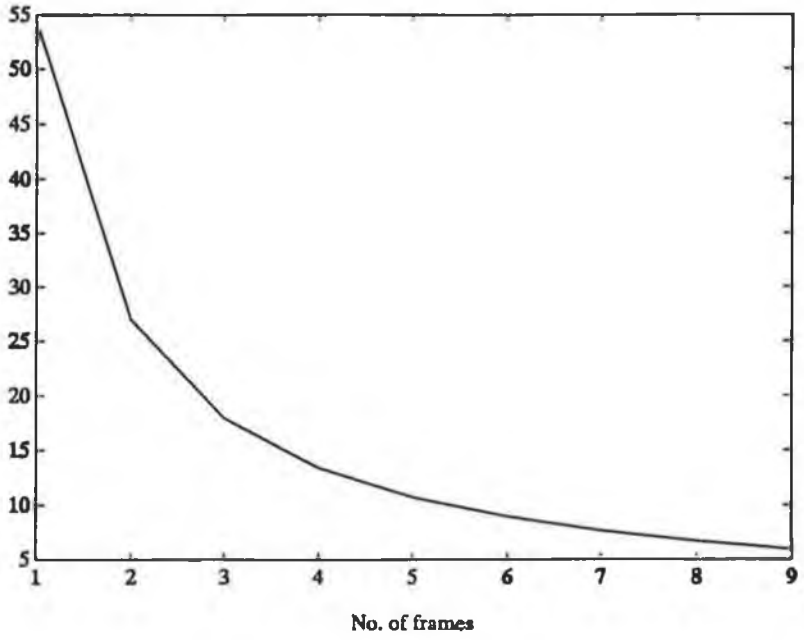

Figure 5.4.17 Length Variance, Segment No.6

Results discussion

We have performed a number of experiments with these and other objects, which satisfy the assumption that objects can be well modelled by a set of straight-line segments. In all cases the algorithm was able to reconstruct 3D models of objects accurately, reliably and quickly. Overall repeatability is very high and false 3D reconstructed segments are very scarce. The algorithm has proven to be very robust to variations in the system parameters Table 5.4.1 and the camera motion. We have performed a number of experiments by changing the nominal system parameters Table 5.4.1, using different camera trajectories and different interframe displacements. The system has performed well as long as reasonable perturbations have been made.

5.5 Conclusions

The new technique for 3D structure estimation from monocular image sequences, using known camera motion, based on tracking line-segments over image sequences has been presented in this chapter. The tracking process consists of prediction, matching and updating stages. These stages are handled in a Kalman filtering framework of covariance based prediction, matching and updating. The prediction stage of our tracking process does not use heuristics about motion in the image plane and applies for arbitrary camera motion. The prediction is based on assumptions about object structure (i.e. a rough knowledge of a distance between camera and an object is assumed known and the depth extent of the object is small compared with the camera-object distance) for the initialization phase, the rest of the tracking process is based on the estimated object structure. The matching stage is based on the simple nearest-neighbour matching algorithm using the Mahalobonis (statistical) distance as a similarity measure. The updating stage is based on standard Kalman filter estimation algorithm.

The reliable and accurate function of the system depends on three major factors. The first factor influencing the reliability and accuracy of the system is calibration of the

camera, the position of the camera on the robot arm and the robot arm itself. When the system is not properly calibrated, the result is an increased error in the prediction from 3D to 2D space and overall 3D reconstruction. The second factor is the satisfaction of the assumptions about object structure. If the initial distance estimate of the object from the camera is severely missed (known with a big error) than the matching will fail and no 3D structure will be reconstructed. It is important to note that the influence of the accuracy of the initial distance estimate can be controlled by controlling the size of the inter-frame displacements. The smaller displacements the less sensitivity to the accuracy of the initial distance estimate. This is a trade-off between the accuracy of the initial distance estimate and the number of processed frames required. If the depth extent of the object is too large, missed and false matches are more likely to result. It is important to note that for the robotic pick and place operations these assumptions are almost always met. The third factor influencing the reliability and accuracy is the uncertainties description. If the uncertainties associated with the different stages of the tracking process are too small than the matching will fail and the reconstructed geometrical model will contain much less segments and poor overall reconstruction results. On the other hand if these uncertainties are too large, the matching process based on the nearest neighbour matching algorithm may generate false matches because of much more competing matches. False matches reconstruct false 3D segments and again poor overall reconstruction results. It is of essential importance to have these uncertainties properly defined.

The main advantage of our system is that determination of uncertainties and other system parameters is very simple and has a firm analytical basis. No parameter is tuned or guessed by numerous trials and errors. The uncertainties due to feature extraction errors are handled on analytical basis. Specifically, it is shown that for our assumption about object structure, prediction of object image motion can be based on 3D structure constraint and not on heuristics about the image motion of features.

The technique has been implemented to provide 3D information for a robot manipulator. The experimental results show reliability and accuracy of the proposed technique. The 3D structure which is recovered very quickly converges to precision on the order of a millimetre provided the system is well calibrated and system parameters are tuned properly.

Chapter 6

Robot Visual Control

In this chapter a new control scheme for a robot manipulator control based on visual information is proposed. The control system determines the position and orientation of the robot gripper in order to achieve desired grasping relation between the gripper and a 3D object. The proposed control scheme consists of two distinct stages: 1) Teaching stage, in the teaching stage the robot reconstructs a 3D geometrical model of a presented unknown object within a class of objects (polyhedra), by integrating information from an image sequence obtained from a camera mounted on a robot arm (eye-in-hand configuration). The model is represented by a set of 3D straight-line segments and denoted as a reference model. The robot is also taught desired grasping relation by manual guidance. 2) Execution stage, in the execution stage the robot system reconstructs a 3D model of the arbitrarily placed 3D object. This model is denoted as an observed model. Then, the necessary position and orientation of its gripper is determined in order to achieve desired position and orientation of its gripper with respect to the object. This is done by estimating 3D displacement between the reference and observed models. The displacement estimation is based on "hypothesise and verify" paradigm. Further, the basic algorithm is extended to handle multiple objects manipulation and recognition The performance of the proposed algorithms has been tested on the real robot system, and the experimental results are presented .

6.1 Introduction

The robotic systems that include vision and other sensors have been receiving a lot of attention. Such systems can solve many problems which limit robot applications, by making them to adapt to environment changes and execute their tasks. In order to present the various approaches which researchers have chosen to adopt in solving the problems of vision-guided robotics, it is useful to consider a taxonomy of visual control strategies for robots due to [San82] and [Wij93]:

1) Static and dynamic position based "Look and Move" control scheme

The block diagram of this strategy is shown in Fig.6.1.

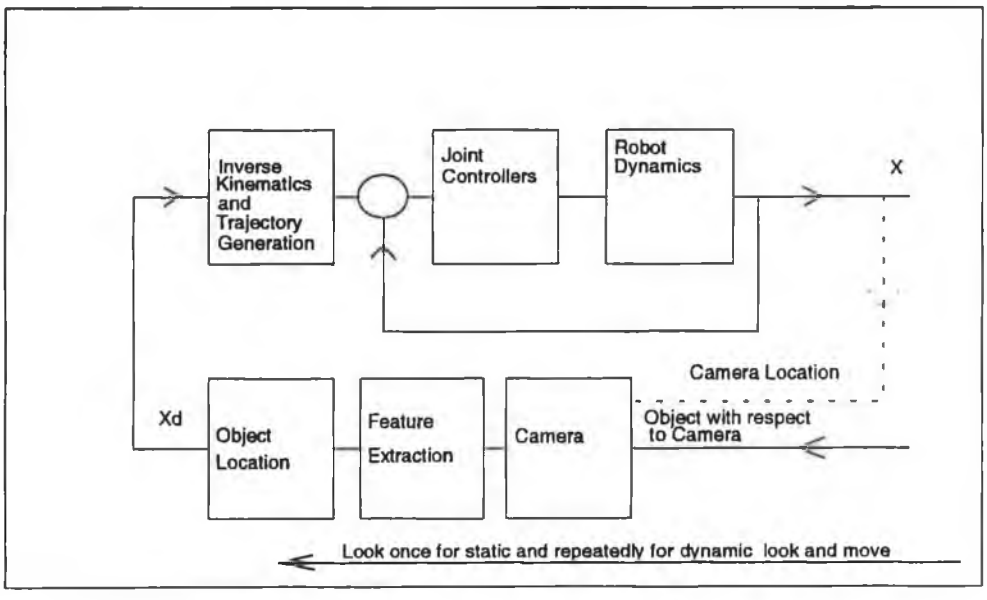

Figure 6.1 Position Based "Look and Move" Visual Control System

In the static and dynamic "look and move" approach a vision system provides an outer feedback loop defining a target position and orientation in world space, while the

dynamic robot arm control relies on an inner control loop based upon feedback derived from joint-mounted sensors. Static look and move is characterised by a sequence of stop-look-move steps, while dynamic look and move is characterised by a sequence of look-while-moving iterations. The distinction between static and dynamic "look and move" strategies is basically dependent on timing. These methods require accurate system calibrations. Examples of static "look and move" systems are given in [Sar82] and [Dri84]. They deal with 2D objects in a plane parallel to a camera plane, effectively determining three degrees of freedom *(x, y)* together with *x-y* plane orientation of an object. An example of dynamic "look and move" which must be used if a moving object is to be picked is given in [All93]. In [All93] stereo cameras are used to determine the centroid coordinates *(x, y, z)* of a moving object being tracked. No object orientation is determined.

2) Static and dynamic image based "Look and Move" control scheme

The block diagram of this strategy is shown in Fig.6.2.

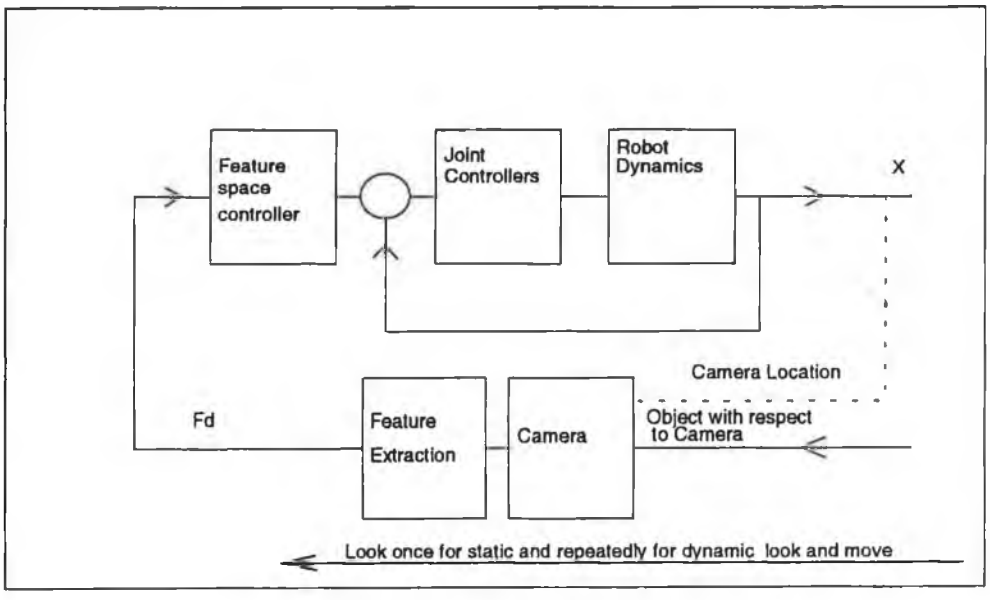

Figure 6.2 Image Based "Look and Move" Visual Control System

In static and dynamic image based "look and move" approaches a vision system provides a feedback loop in terms of an intermediate feature space. This requires a mapping between geometric features in an image of the perceived object and the position and orientation of the object relative to camera. If image processing delays can be overcome, a controller operating directly on the relationship between image features can be designed. If image processing delays are significant to prevent direct dynamic control using vision derived data, then an intermediate approach can be used in which a vision feedback provides an outer corrective loop to the inner joint-based control loop. The major difficulty with this method is connected with selection of image features for control and specifying demands in terms of these features, which is far from being trivial. These methods require accurate system calibrations. Examples of this approach are given in [Fed89] and [Pap92]. In [Fed89] a simple straight line feature is used. They deal with picking a 2D moving object in a plane parallel to a camera plane, effectively determining three degrees of freedom *(x, y)* position and *x-y* plane orientation of the object. In [Pap92] a point or two point features of a 2D object are used to track an object constrained in a plane parallel to a camera plane.

3) End-Effector "Look and Move" control scheme

The block diagram of this strategy is shown in Fig.6.3. These methods use direct feedback of the robot end-effector positions as a means of reducing the problems of robot inaccuracy which result from poor system calibrations. The basic idea in this method is to use visual feedback to match the robot end-effector and object positions in the image. Exact spatial coordinates are not required, and a well-chosen feedback can correct for inaccuracies in calibrated parameters [Wij93]. The control of the joints can be based upon closed-loop joint control with the end-effector tracking scheme providing only an outer corrective loop. The major difficulty with this approach is connected with selection of object features, end-effector features and corresponding end-effector tracking algorithms. An example of this method is given in [Hol94]. There, the stereo

cameras are used to track the robot end-effector (a planar patch associated with it) and the object (a planar patch associated with it) by the use of affine active contour models derived from the planar patch boundaries. Basically, the system determines the position coordinates *(x, y, z)* of the centroids of both planar patches and their orientations (normals onto patches). The control demand is to bring these two patches into desired relative position and orientation.

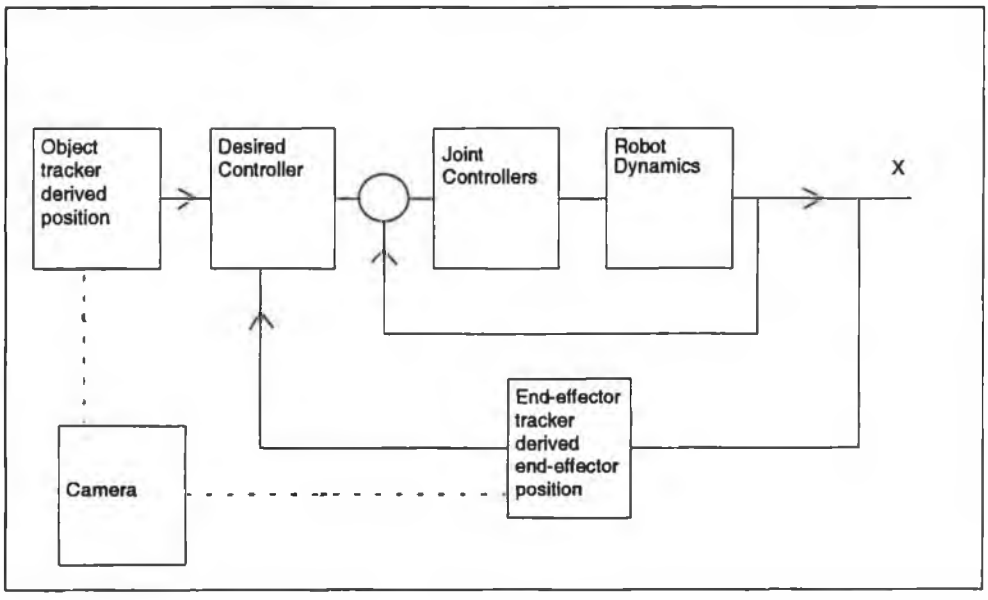

Figure 6.3 End-Effector "Look and Move" Visual Control System

In this chapter a new approach to vision based control is proposed. The task considered in this chapter is to move the robot manipulator gripper to a position and orientation where grasping of a 3D object should be performed. The novelty of this scheme lies in the fact that 3D unknown objects within a class of objects (polyhedra) are considered and the complete procedure is fully automated. The control scheme is based on 3D vision based model reconstruction and 3D displacement estimation. According to the previous classification, this control strategy can be classified as Static Position Based Look and Move Visual Control.

6.2 Relation to previous work

The crucial point in this visual robot control scheme is the problem of 3D displacement estimation of an object, based on 3D observations of the object. Such a problem is related with motion analysis in computer vision. In particular, this problem is related to the problem of motion estimation from two or more stereo frames. Basically, this problem consists of two parts. The first part is to establish correspondences between 3D features in two stereo frames. The second part is to estimate 3D displacement using matched features. Computationally efficient solutions to these problems based on "hypothesise and verify" paradigm were reported in the literature [Fau86], [Aya86], [Aya89], [Zha90], [Zha92]. Unlike other approaches this approach attempts to find locally consistent matches by using rigidity assumptions about objects. Then, the global matching is checked by using displacement estimates obtained from locally consistent matches. In [Aya89] the initial displacement (motion) estimate between two stereo frames is assumed to be known, and two stereo frames are matched and the displacement estimate refined by using the extended Kalman filter algorithm. In [Zha90], [Zha92] no assumption about displacement is used and a set of rigidity constraints is used to discover potential matches, followed by an application of the extended Kalman filter algorithm to estimate 3D displacement between two stereo frames.

Based on these ideas, we develop a technique for 3D displacement estimation and robot control, which brings following modifications and improvements

- We formulate a set of rigidity constraints for geometric primitives which we use and incorporate the uncertainty of measurements in their formulation. These rigidity constraints are used to discover potential matches between two 3D models.

- We use a closed-form algorithm to estimate 3D displacement. This algorithm thanks to its non iterative nature outperforms the efficiency of nonlinear algorithms.

- We formulate the usage of this information in order to accomplish the robot control task.

6.3 Control strategy

In robotics, the task of pick and place is the most fundamental one. In order to pick a workpiece the desired position and orientation of the robot gripper with respect to the workpiece must be achieved. In this chapter the problem of determining the desired gripper position and orientation for the arbitrarily placed 3D workpiece based on visual information is discussed. The relations between different coordinate systems and homogeneous transformation matrices used for robot visual control are shown in Fig.6.3.1.

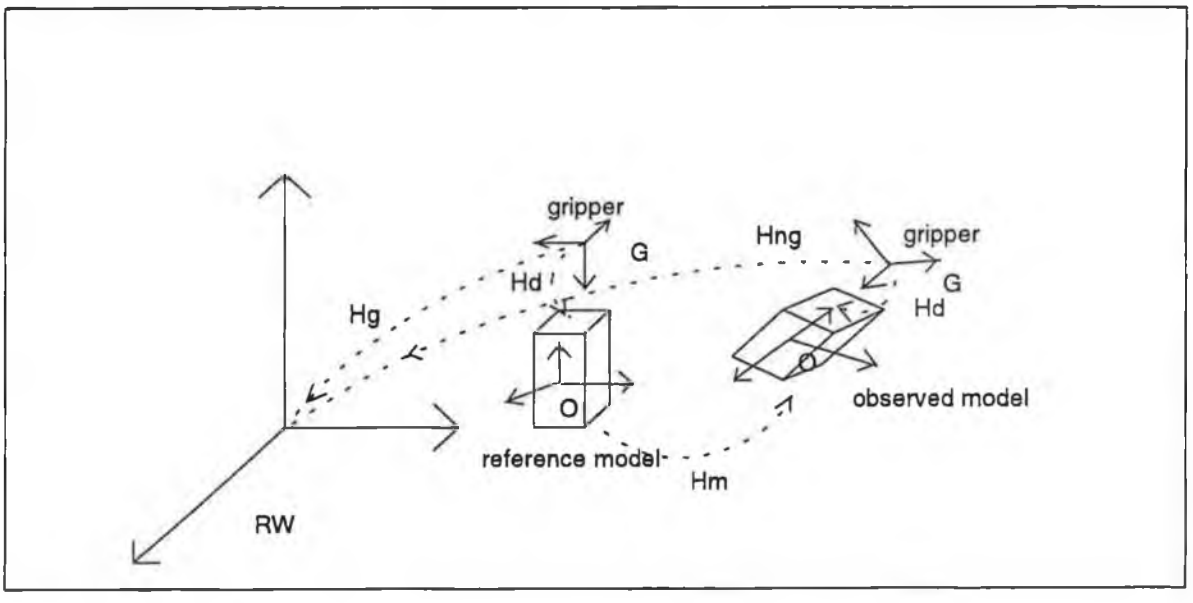

Figure 6.3.1 Coordinate Transformations for Visual Robot Control

The list of definitions for coordinate systems used

- G, The gripper coordinate frame. That is, the coordinate system fixed on the robot gripper.

- O, The coordinate system fixed on the object

- RW, The robot coordinate system. The position and orientation of the gripper coordinate system G is controlled with respect to the RW

The list of definitions of homogeneous transformation matrices used

- H_g, H_{ng} define coordinate transformations from G to RW

$$H_g = \begin{bmatrix} R_g & T_g \\ 0 & 1 \end{bmatrix}$$

- H_m defines the coordinate transformation between the reference and observed object

$$H_m = \begin{bmatrix} R_m & T_m \\ 0 & 1 \end{bmatrix}$$

- H_d is the desired coordinate transformation between the gripper frame G and the object frame O

$$H_m = \begin{bmatrix} R_m & T_m \\ 0 & 1 \end{bmatrix}$$

As was mentioned before, our control scheme consists of two stages. The first stage is the teaching stage. During this stage an unknown object within a class of objects is presented to the robot-camera system. The system reconstructs a 3D model of the object using the algorithm presented in Chapter 5. We call this model a reference model. Next, the robot arm is manually guided to the desired grasping position and orientation

for the object. This position and orientation defines Hg transformation. This matrix definition and the reference model construction makes the teaching stage of the control scheme. Second stage is the execution stage. This is a working stage of the robot arm, in which the robot arm is required to grasp the arbitrarily placed object keeping the desired relative orientation and position defined by H_d matrix. During this stage the robot-camera system reconstructs a 3D model of the object. We call this model the observed model. Having the reference model and the observed model we can estimate the 3D displacement (motion) between them. This displacement is given by H_m coordinate transformation. Having determined H_m, the new gripper position and orientation is given by the homogeneous matrix H_{ng}. This matrix simply follows from Fig.6.3.1 as

$$H_{ng} = H_g H_m^{-1} \qquad \textbf{(6.3.1)}$$

The fundamental problem in this approach is to construct an accurate and reliable 3D representation of an object by using visual data and to estimate the relative displacement (motion) of the object by using these 3D representations. In order to estimate 3D displacement between the reference and actual pose of the object, the matching (correspondence) problem must be resolved between the reference and observed object models. We solve this problem by using so-called "hypothesise and verify" paradigm for tackling the matching problem. Having resolved correspondences, we estimate the 3D displacement using a closed-form algorithm. In the sequel approaches for solving these two difficult and important problems will be presented.

6.4 3D Vision based model reconstruction

Our eye-in-hand robot configuration is shown in Fig.6.4.1.

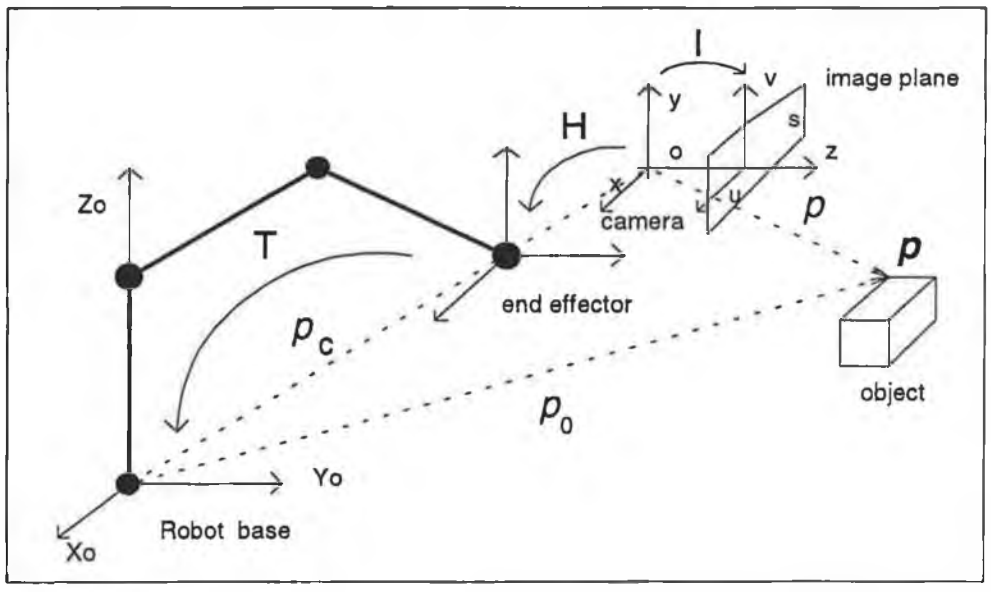

Figure 6.4.1 Eye in Hand Robot Configuration

A camera is rigidly mounted on the robot gripper. The camera pose with respect to the gripper is described by the transformation matrix H. The gripper pose is described by its transformation matrix with respect to the robot base. The matrix H is determined by the eye/hand calibration procedure (Chapter 3). The matrix T is provided by the robot controller (Chapter 2) . The camera position with respect to the robot base is then given by

$$H_c = TH \tag{6.4.1}$$

The algorithm for 3D model reconstruction given in Chapter 5 is used. The algorithm integrates 3D structure of a scene from the image sequence using known camera motion. The algorithm integrates 3D model of a scene in the form of 3D straight-line segments by tracking and fusing 2D line-segments measurements over the image sequence. A 3D line-segment is described by coordinates of its two end-points and their covariance matrices

$$S = [p_1, p_2, W_1, W_2] \tag{6.4.2}$$

The process of tracking, matching and updating is based on Kalman filtering framework. This approach provides for reliability, accuracy and computational advantages. In order to construct 3D representation of an object, the object is placed in the field of view of the camera, the robot makes a sequence of movement (stop and go motion) at each pause an image is taken, processed and the 3D model is updated. The 3D model is defined in the robot base coordinate frame.

6.5 3D Displacement estimation

The problem of estimating 3D displacement (motion) of an object from its two 3D observations is usually solved in two steps. The first step is to establish feature correspondences between two observations. The correspondence (matching) problem is regarded as a very difficult one. The rigidity assumption about the object provides a strong constraint which simplifies the analysis and is used in almost all matching algorithms [Bol82], [Hora84], [Gri84], [Pol87], [Gri87], [Che88], [Zha92]. The form of rigidity constraints depend on the geometrical primitives used to represent 3D object, among them are 3D points, lines and planar patches.

One approach for solving the correspondence problem, originally developed for the recognition purposes, is so-called the interpretation tree search. This approach operates by examining all hypotheses about pairings between two 3D models. By using geometric constraints the search tree can be considerably constrained making computations more feasible [Bol82], [Hora84], [Gri84], [Gri87]. This approach suffers from exponential combinatorics in the case of spurious data and occluded objects.

Another approach for solving this problem is so-called "hypothesise and verify" paradigm [Fau86], [Aya86], [Aya89], [Zha92]. The distinctive feature of this approach is that the displacement parameters are initially determined using a locally consistent set of

the geometrical primitives determined by rigidity assumptions ("hypothesise stage"). To ensure global consistency, verification is done for overall primitives ("verify stage"). After the verification stage is done the best hypothesis is chosen. This is shown in Fig.6.5.1.

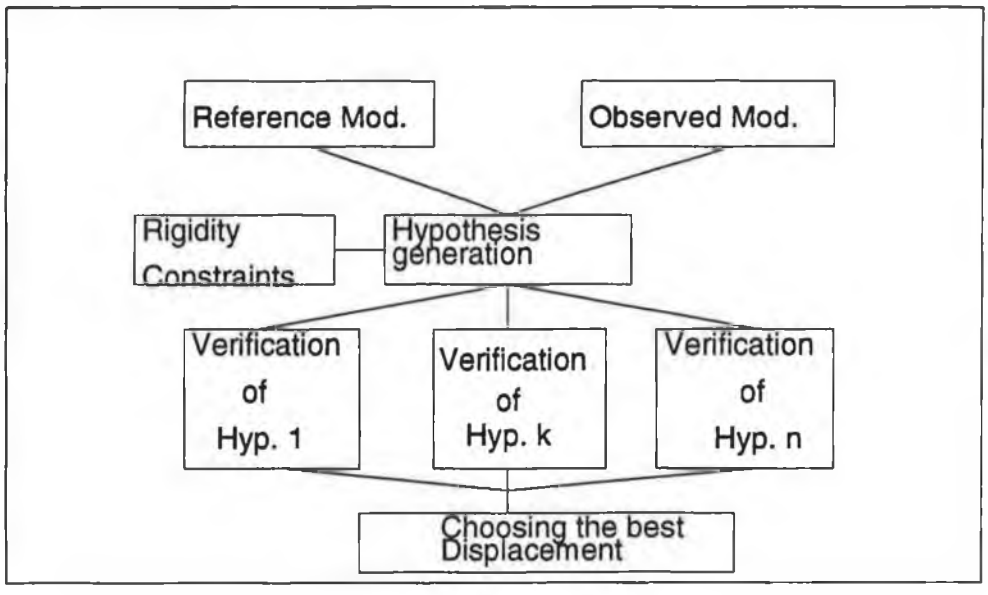

Figure 6.5.1 Diagram of the Hypothesise and Verify Paradigm

Posing the solution in this way cuts down the computational complexity and improves the robustness of the method in case of partially observed object. In this work we use this paradigm to solve for 3D displacement.

Once correspondences are established the second step is to estimate the 3D displacement between two observations. Many closed and iterative solutions exist in the literature to the problem of displacement estimation from known correspondences [Aru87], [Fau83], [Hua86]. Due to the errors in measurements these methods find solutions by minimising some criteria (usually least squares).

6.5.1 Rigidity constraints

The 3D rigid body motion can be uniquely represented by a rotation around the origin of the coordinate system followed by a translation. Let P and P' be a position vector of the same point before and after motion, then we have

$$P' = RP + T \qquad \textbf{(6.5.1.1)}$$

where R is the rotation matrix and T is the translation vector. Under rigid motion, the geometry of a rigid body remains constant. In other words the geometry of the object does not change during displacement. Since we deal with objects represented by 3D line segments, a set of rigidity constraints for line-segments is given below. If two line segments S_1 and S_2 in the reference model are matched to the S_1' and S_2' segments in the observed model, as shown in Fig.6.5.1.1, then:

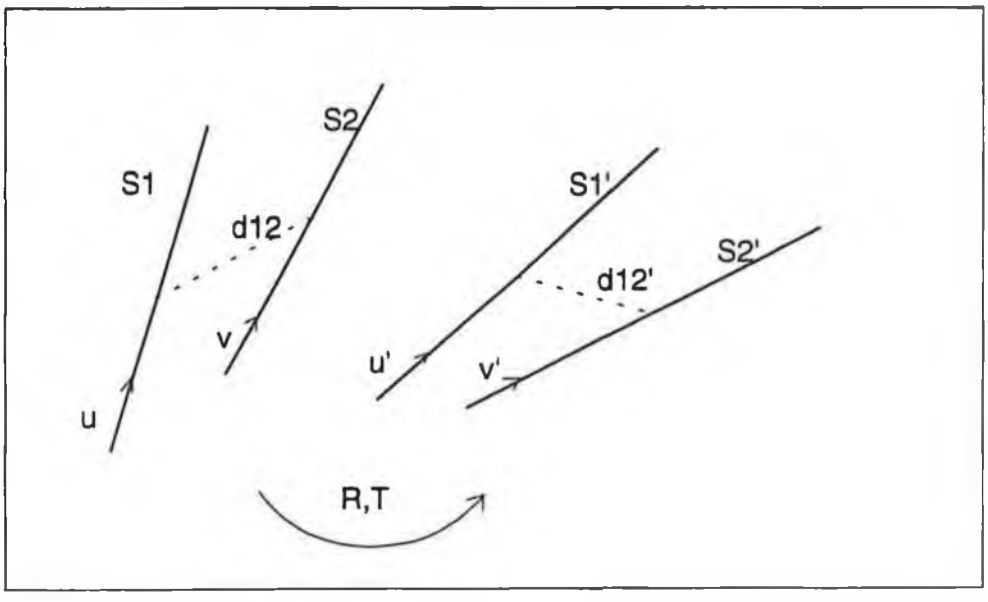

Figure 6.5.1.1 Rigidity Constraints

The following rigidity constraints hold

- Length constraint

$$l_1 = l_1', \quad l_2 = l_2'$$

- Distance constraint

$$d_{12} = d_{12}'$$

- Angular constraint

$$uv = u'v' \qquad \textbf{(6.5.1.2)}$$

where l_i and l_i' are segment lengths, d_{12} and $d_{12'}$ are distances between segment midpoints, u and v are segment unit vectors and uv is a dot-product. The above equalities can be easily verified using the properties of rigid displacement Eq.(6.5.1.1).

Above mentioned constraints hold in case of noise-free measurements. Since the 3D vision based measurements are always corrupted by noise due to image processing, system calibration errors and stereo reconstruction, the constraints are not satisfied. Here the rigidity constraints are reformulated by explicitly taking into account the uncertainty of measurements. One way is to chose fixed threshold values for constraints. However, the errors of measurements given by a stereo system have different distributions in different directions, so such phenomena cannot be handled properly with prefixed threshold values. The idea is to dynamically compute a threshold for each constraint. Since the module for 3D visual reconstruction (Chapter 5) supplies the 3D information and their uncertainties in the form of covariance matrices Eq.(6.4.2), the rigidity constraints should be reformulated to take into account measurement uncertainties explicitly.

Let us consider a nonlinear equation of the form

$$y = f(x), \ \bar{x} = x_0, \ Cov(x) = W \qquad \textbf{(6.5.1.3)}$$

where x is a normally distributed random vector with the mean x_0 and $Cov(x) = W$, then up to the first-order approximation y is the normally distributed variable with the mean and variance given by

$$\bar{y} = f(x_0),\ Var(y) = Jac(x_0) W Jac(x_0)^T \qquad \textbf{(6.5.1.4)}$$

where *Jac* is the gradient vector of the *f(x)*.

Under this assumption

$$d = (y - \bar{y})^2 / Var(y) \qquad \textbf{(6.5.1.5)}$$

d is a $\chi 2$ distributed variable with one degree of freedom. The threshold k on this variable can be set such that the desired probability of d falls in the interval $[0, k]$ is satisfied, by looking at a $\chi 2$ distribution table.

$$d \leq k \qquad \textbf{(6.5.1.6)}$$

For example, one can use $k = 3.84$ for a probability of 95%.

By applying this formalism, the rigidity constraints Eq.(6.5.1.2) are reformulated as follows

Length constraint

The length constraint says that the difference between the norms of two vectors should be zero. For convenience, we use the squared norm of a vector instead its norm. Taking into account the uncertainty of measurements, we formalise the constraint as follows:

$$d = l^T l - l'^T l'$$

Given the covariance matrix W_l of l and the covariance matrix $W_{l'}$ of l', the variance W_d of d can be computed according to Eq.(6.5.1.4).

$$W_d = 4(l^T W_l l + l'^T W_{l'} l')$$

Therefore, the length constraint can be expressed as

$$\frac{d^2}{W_d} < k \qquad \textbf{(6.5.1.7)}$$

According to Eq.(6.5.1.6), a threshold k on this variable can be set such that the desired probability of d^2/W_d falls in the interval $[0, k]$ is satisfied. We usually use k=3.84 for a probability of 95%.

Distance constraint

Exactly the same derivation applies to the distance constraint.

Angular constraint

The dot-product constraint says that the difference between the cosines of angles between two vector should be zero.

$$d = u^T v - u'^T v'$$

Now the variance of d can be computed. Under the first-order approximation, we have

$$W_d = v^T W_u v + u^T W_v u + v'^T W_{u'} v' + u'^T W_{v'} u'$$

Therefore, the dot-product constraint is expressed as

$$\frac{d^2}{W_d} < k \qquad \textbf{(6.5.1.8)}$$

According to Eq.(6.5.1.6), a threshold k on this variable can be set such that the desired probability of d^2/W_d falls in the interval $[0, k]$ is satisfied. One can use k=3.84 for a probability of 95%.

6.5.2 Generating hypotheses

In this stage the rigidity constraints are used to generate hypotheses about matches between the reference and observed model. The generation of hypotheses is implemented as follows. For a segment S_1 in the observed model, a segment S_1' is found in the reference model such that its length is compatible with that of S_1. For the pair (S_1, S_1'), we then find pairs (S_k, S_k') such that two pairs satisfy the rigidity constraints. When this is done, we go to the next segment S_2 of the observed model. Since, we do not want to recover all matches in this stage, but to recover potential displacements between them, different heuristics are used to reduce complexity of the process.

Firstly, all segments in the reference and observed model are sorted in decreasing length order so that we can easily find, by binary search, the segments in the reference model that are compatible in length with segments in the observed model.

Secondly, instead of finding all possible pairs compatible with a given pair, we find a sufficient number of them (usually three) and stop. The number of hypotheses to be generated is predefined. We usually generate twenty hypotheses with three matched pairs for objects having a few tenths of segments.

Thirdly, the number of segments is reduced by considering only longest segments in the observed and reference model (for instance, one half of segments).

6.5.3 3D Displacement estimation from 3D point correspondences

In this section commonly used techniques for 3D displacement estimation from 3D point correspondences are presented.

Let $y_1,...,y_n$ be N points in 3D space. Let R be a rotation matrix and T be a translation vector. Let $x_1,...,x_n$ be the points in 3D space which match $y_1,...,y_n$. Each x_i is the same rigid body motion of y_i. Hence, each y_i is given by

$$y_i = Rx_i + T \qquad \textbf{(6.5.3.1)}$$

3D displacement (motion) estimation problem is to infer R and T from $y_1,...,y_n$ and $x_1,...,x_n$. In theory three noncollinear point correspondences are enough to uniquely estimate rigid body displacement. In the presence of noise, we need to take into account the uncertainties in the points that are constructed by stereo triangulation. Using the estimated 3D positions and their estimated covariance matrices Eq.(6.5.3.1) becomes

$$\hat{y}_i = y_i + \varepsilon_{yi}, \quad Cov(\varepsilon_{yi}) = W_{yi}, \quad \hat{x}_i = x_i + \varepsilon_{xi}, \quad Cov(\varepsilon_{xi}) = W_{xi}$$

$$\hat{y}_i = R\hat{x}_i + T + \varepsilon_i, \quad \varepsilon_i = R\varepsilon_{xi} - \varepsilon_{yi}, \quad Cov(\varepsilon_i) = W_i = W_{yi} + RW_{xi}R^T \qquad \textbf{(6.5.3.2)}$$

According to the minimum variance estimation (Chapter 3, Section 3.5), the displacement parameters should minimise

$$\sum_{i=1}^{N} (\hat{y}_i - R\hat{x}_i - T)^T W_i^{-1} (\hat{y}_i - R\hat{x}_i - T) \qquad \textbf{(6.5.3.3)}$$

where N is the number of point correspondences and Wi is given by Eq.(6.5.3.2). It is important to note that Wi depends on the unknown matrix R. Since the matrix R is a rotation matrix, there are only three degrees of freedom in R. Letting (α, β, γ) denote

Euler angles parametrization of R (Appendix A), Eq.(6.5.3.3) is a nonlinear function of a six dimensional parameter vector $m^T=(\alpha, \beta, \gamma, tx, ty, tz)$. Thus, the objective is to determine the vector m which minimizes Eq.(6.5.3.3).

Closed-form estimation

A closed-form solution that minimizes a special form of Eq.(6.5.3.3)

$$\min_{(R,T)} \sum_{i=1}^{N} \gamma_i \left\| y_i - Rx_i - T \right\|^2, \quad \text{subject to } RR^T = I \qquad \textbf{(6.5.3.4)}$$

where γ_i are weighting factors, is given in [Aru87], [Free89]. The solution to this problem is based on singular value decomposition (*SVD*) of a (3x3) matrix (Appendix B). This objective function is in fact a scalar weighted least squares criterion. However, since the depth component of a point is significantly less reliable than the lateral components, and the errors in the three components have considerable correlations, the scalar weighted objective function does not properly treat those uncertainties. A closed form solution to a matrix weighted objective function given by

$$\sum_{i=1}^{n} (\hat{y}_i - R\hat{x}_i - T)^T W_i^{-1} (\hat{y}_i - R\hat{x}_i - T) \qquad \textbf{(6.5.3.5)}$$

where *Wi* are weighting matrices, is given in [Wen92]. Estimates of *Wi* follow from Eq.(6.5.3.2) by taking $R=I$. This is a valid approximation for small rotations only.

Extended Kalman filter estimation

The standard Kalman filter is a powerful recursive tool to deal with state estimation problem in a linear noisy system. The extended Kalman filter [Bar88], [Gel74], [Jaz76] applies the standard Kalman filter to nonliner systems with additive Gaussian

noise by continually updating a linearization around the previous state estimate, starting with an initial guess. The use of the extended Kalman filter for displacement estimation were reported in [Aya86], [Aya89], [Zha90], [Zha92].

Linearizing the measurement equation Eq.(6.5.3.2) around the current displacement estimate $m=(\alpha, \beta, \gamma, tx, ty, tz)$ and the current observation vector (y, x) (Chapter 3, Section 3.5) we get a linearized measurement equation

$$z = Hm + \varepsilon, \quad Cov(\varepsilon) = W$$

Now, the standard Kalman filter (Chapter 5) can be applied to update the displacement estimate m. This process is performed sequentially over all point correspondences. The distinctive advantage of the *EKF* based displacement estimation is that all possible types of differentiable measurement equations (not only the 3D point primitive) can be incorporated in a uniform manner.

Direct iterative estimation

The last method we consider is the direct iterative optimization of the objective function Eq.(6.5.3.3). The modified Newton's method [Den83] is used to optimise the cost function.

The accuracy of the presented methods was compared. A number of simulations with the simulated and real data to test the performance of the methods with regard to the accuracy, reliability and computational complexity was performed. The direct iterative optimisation found accurate solutions in all tested cases with a few iterations, while having the biggest computational demand. The well known problems with *EKF* were confirmed. Unless a very good initial guess is provided and noise level is low, the initial divergence occurs for a small number of processed measurements for systems with severe

nonlinearities (the use of an iterated *EKF* at the begining is usually a must). It may take a lot of measurements (point correspondences) for the *EKF* to pull back from the initial divergence. Once the good convergence is achieved, *EKF* performs well. The computational demand of *EKF* is smaller than for the direct iterative optimisation thanks to its sequential nature. Closed-form solutions are computationally most efficient, even though they are the most sensitive to the measurement noise. For the small number of point correspondences and high level of the measurement noise they may be unreliable. By performing numerous tests with real data obtained from our system, we found closed-form solutions nearly as good as nonlinear solutions with regard to accuracy. Finally, we settled on the following algorithm for displacement estimation

- We find the closed-form solution which optimises Eq.(6.5.3.4). This step is performed in the "hypothesise" stage, when the number of point correspondences is very small (usually three). Since a relatively large number of the hypothesised displacements must be computed, it is of a great interest to have a fast and accurate algorithm in this stage. Having determined displacement estimate m, its covariance matrix is determined by (Chapter 3, Section 3.5)

$$Cov(\hat{m}) = \left(\sum_{i=1}^{N} \frac{\partial \hat{y}(\hat{m})_i^T}{\partial m} W_i^{-1} \frac{\partial \hat{y}(\hat{m})_i}{\partial m} \right)^{-1}$$

- When new point correspondences become available in the "verify" stage, the final displacement can be determined by recomputing the closed-form solution for all point correspondences. This solution can be improved further by performing the iterative optimisation, using the closed-form solution as a starting point. It usually takes just one iteration for modified Newton's method to converge. After performing a number of experiments we do not find this necessary, since the closed-form solution itself is of high accuracy.

In our case segments are given by their endpoints Eq.(6.4.2), and we do not establish correspondence between endpoints but between segments, the midpoints of matched segments are used to estimate displacement. A midpoint of a segment given by Eq.(6.4.2) and its covariance matrix is given by

$$p_m = \frac{p_1 + p_2}{2}, \qquad Cov(p_m) = \frac{W_1 + W_2}{4} \tag{6.5.3.6}$$

It is obvious, if two segments are matched their midpoints are matched unambiguously. Therefore, at least three matched segments are needed to determine displacement. In our experiments, hypotheses consisting of three segment pairs are usually used. So, for each generated hypotheses we estimate corresponding displacement as presented above.

6.5.4 Verifying hypotheses

At this stage, from each hypothesis an initial estimate of the rotation and translation is computed using the method from the previous section. Each initial displacement estimate is applied to the reference model. Then, we have to match the transformed reference model and the observed model. If a transformed segment is near enough to some segment in the observed model, then this pair is considered to be matched. After discovering matches, the final displacement using all matched segments is computed. The same procedure is done for each hypothesis. Next step is to chose the best hypothesis. A simple criteria is used, the best hypothesis is one with maximal number of matches.

In order to verify the quality of an initial displacement estimate and possibly to improve it, all segments in the reference model are transformed by using the initial displacement estimate. Transformed segments should be matched to the observed

segments. Again, the statistical measure of closeness between segments is used. Segments are given by

$$S = [p_1, p_2, W_1, W_2] \tag{6.5.4.1}$$

Let H_i be the initial displacement estimate

$$H_i = \begin{bmatrix} R_i & T_i \\ 0 & 1 \end{bmatrix}$$

We transform the reference segments and their covariance matrices with the initial transformation

$$p_1^t = H_i p_1^o, \quad p_2^t = H_i p_2^o, \quad Cov(p_1^t) = H_i W_1^o H_i^T, \quad Cov(p_2^t) = H_i W_2^o H_i^T \tag{6.5.4.2}$$

We want to know whether transformed reference segments can be matched to the observed segments. The first step is to examine if two segments are compatible in length. According to Eq.(6.5.1.7) the Mahalanobis distance between the segment lengths for matched segments must satisfy

$$\frac{d_o^2}{W_o} < k \tag{6.5.4.3}$$

the threshold value for k of 3.84 for the probability of 95% is used. If this constraint is satisfied, the similarity in orientation between segments is examined next. According to Eq.(6.5.1.8), the Mahalanobis distance between the segment orientations for matched segments must satisfy

$$\frac{d_o^2}{W_o} < k \qquad \textbf{(6.5.4.4)}$$

the threshold value for k of 3.84 for the probability of 95% is used. If this constraint is satisfied as well, the distance between the midpoints of two segments is examined. The Mahalanobis distance between the midpoints is given by

$$d_m^2 = (m-m')^T (W+W')^{-1} (m-m') \qquad \textbf{(6.5.4.5)}$$

where m, m' are segment midpoints and W, W' are their covariance matrices. If d_m^2 is less than the prescribed threshold (we use the threshold value of 7.81 for the probability of 95%), then two segments are considered matched. If more then two segments satisfy this test, we choose as a match two segments with the smallest value of the sum of values Eq.(6.5.4.3), Eq.(6.5.4.4) and Eq.(6.5.4.5).

It can be seen that the complexity of the verifying stage is in the worst case is $O(MN)$ (where M and N are numbers of observed and reference segments) for each hypothesis. The speed of the algorithm depends essentially on the ability to quickly access the segments in the observed model. It is possible to use several techniques to structure data to achieve this. Here the simplest one is used. All segments are sorted by length, binary search is used to discard segments from comparison in the observed model that are not compatible in length with the transformed reference segments. Depending on the model structure, this might lead to very efficient matching .

6.6 Multiple objects manipulation and recognition

6.6.1 Multiple identical objects manipulation

The algorithms presented above are easily applied for multiple objects segmentation and manipulation. Let us consider a case where the robot arm is supposed to manipulate multiple identical objects in its field of view. The teaching stage of the visual robot control is the same as described previously. The reference model of the object is reconstructed and the desired grasping position is thought. The execution stage of the control scheme reconstructs the observed model consisting of a number of identical objects. The objects must be segmented (labelled) out for the purpose of manipulation. We propose the following algorithm for object segmentation.

- The same algorithm as in case of a single object is applied. If the sufficient number of the reference model segments are matched to the segments in the observed model (more than 1/2 of the total number of segments in the reference model), it is considered that an instance of the object is found. The matched segments in the observed model are removed from the observed model. The matched segments in the observed and the reference model are used to manipulate the object instance.

- The same algorithm is applied repeatedly between the reference and the observed model (after removal of the found object) and the next instances of the object are found.

- The algorithm stops when all objects are labelled. This is defined when sufficient number of matches cannot be found (1/2 of the total number of segments in the reference model).

6.6.2 Multiple different objects manipulation and recognition

The algorithms presented above are easily applied for multiple objects segmentation and manipulation. Let us consider a case where the robot arm is supposed to manipulate multiple different objects in its field of view. The teaching stage of the visual robot control is the same as described previously. The reference model of each object is reconstructed and the desired grasping position is thought. The execution stage of the control scheme reconstructs the observed model consisting of a number of objects. The objects must be segmented (recognised) for the purpose of manipulation. We propose the following algorithm for object segmentation.

- The same algorithm as in case of multiple identical objects is applied. We start by taking one reference model. If the sufficient number of the reference model segments are matched to the segments in the observed model (more than 1/2 of the total number of segments in the reference model), it is considered that an instance of the object is found. The matched segments in the observed model are removed from the observed model. The matched segments in the observed and the reference model are used to manipulate the object instance.

- The same algorithm is applied repeatedly between the reference and the observed model (after removal of the found object) and the next instances of the objects are found.

- The procedure stops when all objects of one type are labelled. This is defined when sufficient number of matches cannot be found (1/2 of the total number of segments in the reference model).

- Then, the next reference model is chosen and the procedure is repeated.

- The algorithm stops when all reference models are applied.

6.7 Experimental results

The approach presented in this chapter has been tested with a lot of various objects. The algorithm has succeeded in correctly matching, computing 3D displacement and controlling the robot arm in almost all tested cases. In this section a few experimental examples are provided to demonstrate the performance of the matching process and accuracy of the displacement estimation.

First test object

Fig.6.6.1 shows an image of an artificial cubic like object. Fig.6.6.2 shows a 3D reconstructed model of the object. A sequence of six images was used for 3D model reconstruction. It forms the reference model of the object. The robot is manually guided to the desired grasping pose, and the gripper pose is stored. Fig.6.6.3 shows an image of the displaced object, there is a translation and rotation in the displacement. Fig.6.6.4 shows a 3D reconstructed model of the displaced object. A sequence of six images was used for reconstruction. In both cases the robot arm follows the same motion sequence (stop, process and go) during the model reconstruction phase. Fig.6.6.5 shows the reference and observed 3D models superimposed. The reference model consists of 44 segments, while the observed model consists of 43 segments. For the convenience of presentation all results are given with respect to the camera coordinate frame where the first image is taken.

Applying the displacement estimation technique to these two models, twenty hypotheses are generated consisting of three matched segment pairs. For all hypotheses initial displacement estimates are computed. The initial displacement estimates based on five correct hypotheses are given in Table 6.6.1. The rotation matrix is parametrized by Euler angles (Appendix A).

Hyp. No.	α (roll) deg.	β (pitch) deg.	γ (yaw) deg.	*tx* (mm)	*ty* (mm)	*tz* (mm)
1	2.5	4.5	50.6	36.5	39.8	1.1
2	1.3	2.9	53.7	35.1	38.8	0.9
3	3.5	3.3	50.1	37.9	41.4	1.9
4	3.3	3.4	53.7	35.7	40.5	1.3
5	2.6	4.1	51.7	36.1	41.3	1.6

Table 6.6.1 Initial Displacement Estimates

All initial estimates are applied to the reference model in order to do their verification and to improve estimates. Table 6.6.2 shows the number of matched segments between the transformed reference model and observed model and the final displacement estimates for the five correct hypotheses. As can be seen all of them yield the correct estimate of displacement.

Hyp.No	α (roll) deg.	β (pitch) deg.	γ (yaw) deg.	*tx* (mm)	*ty* (mm)	*tz* (mm)	No. of Matches
1	1.2	2.3	50.9	37.9	39.4	0.3	36
2	1.2	2.3	50.9	37.9	39.4	0.3	36
3	1.2	2.3	50.9	37.9	39.4	0.3	36
4	1.2	2.3	50.9	37.9	39.4	0.3	36
5	1.2	2.3	50.9	37.9	39.4	0.3	36

Table 6.6.2 Final Displacement Estimates

To determine how good the estimates are, the best computed estimate is applied to the reference model and superimpose it on the observed model. Fig.6.6.6 shows the superposition of the models by using hypothesis No.1. The estimate of displacement is very good. This is also confirmed by controlling the robot arm to grasp the object using this displacement estimate.

Ten successive experiments were performed with this object placed at different locations. In all cases displacement was correctly estimated and the robot was controlled successfully. The robot arm takes the straight-line path to approach the object from above.

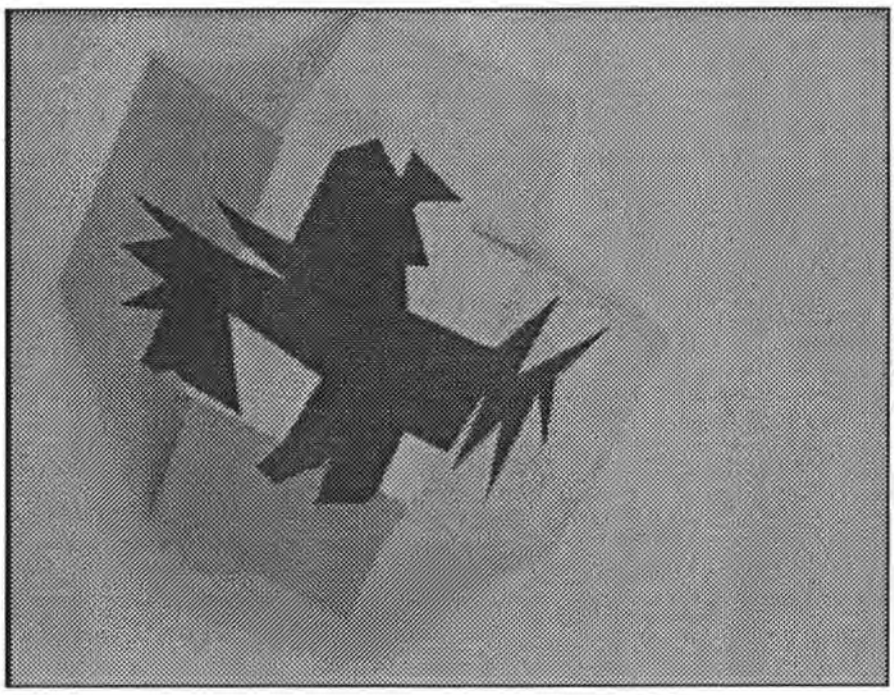

Figure 6.6.1 Raw Image of Object

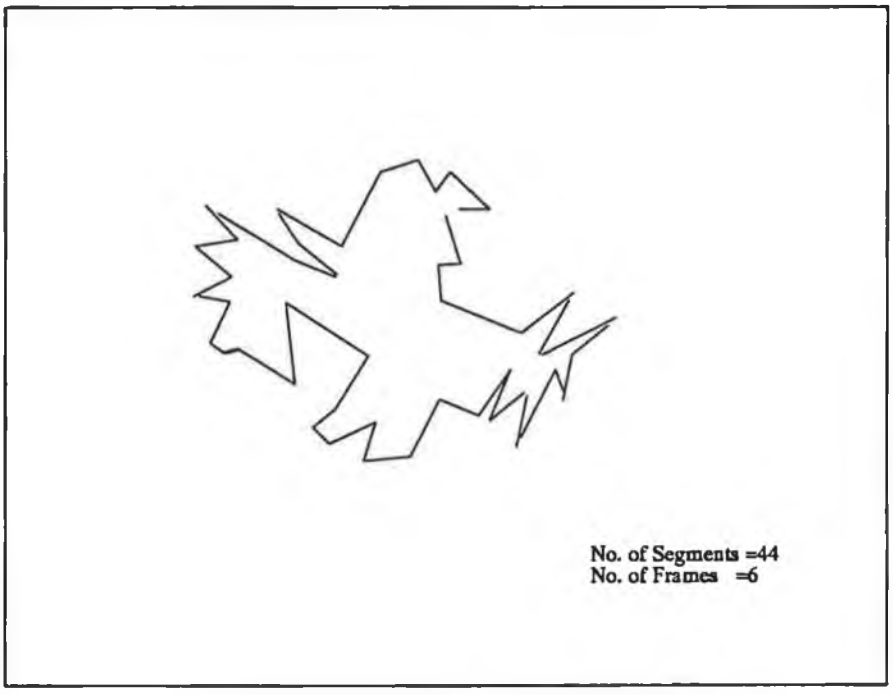

Figure 6.6.2 3D Model of Object, Projection on Camera

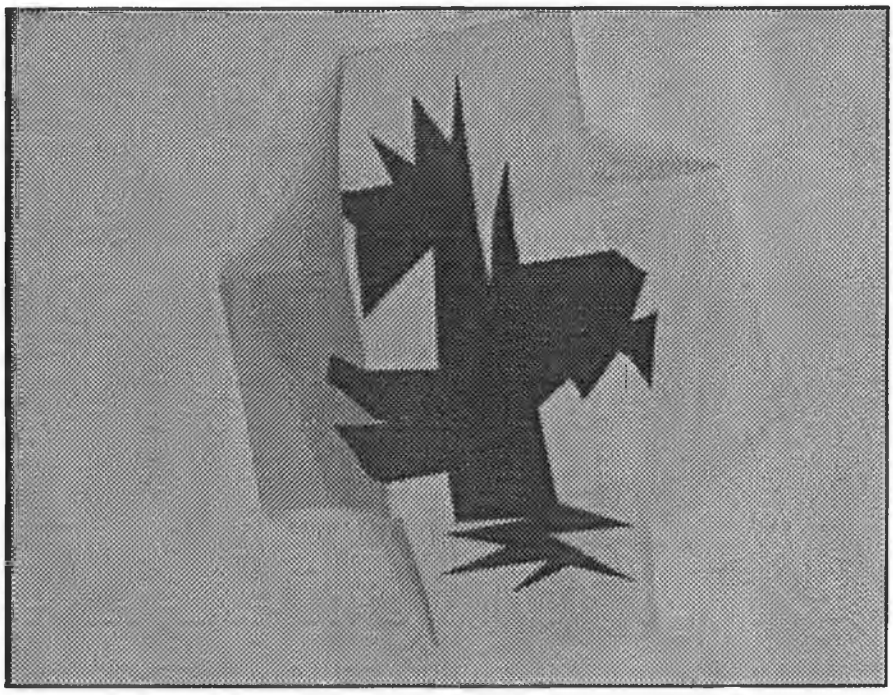

Figure 6.6.3 Raw Image of Displaced Object

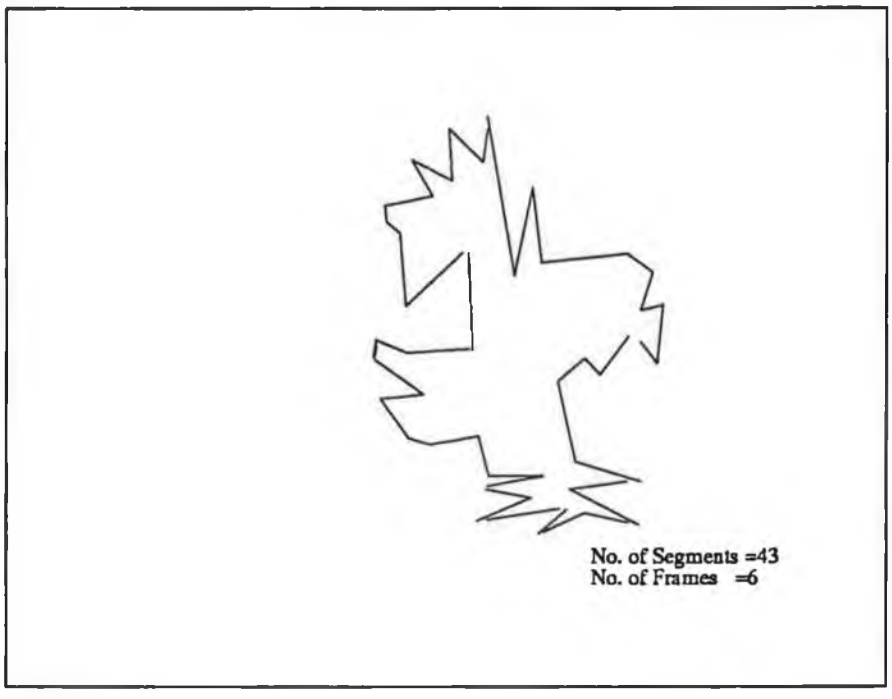

Figure 6.6.4 3D Model of Displaced Object, Projection on Camera

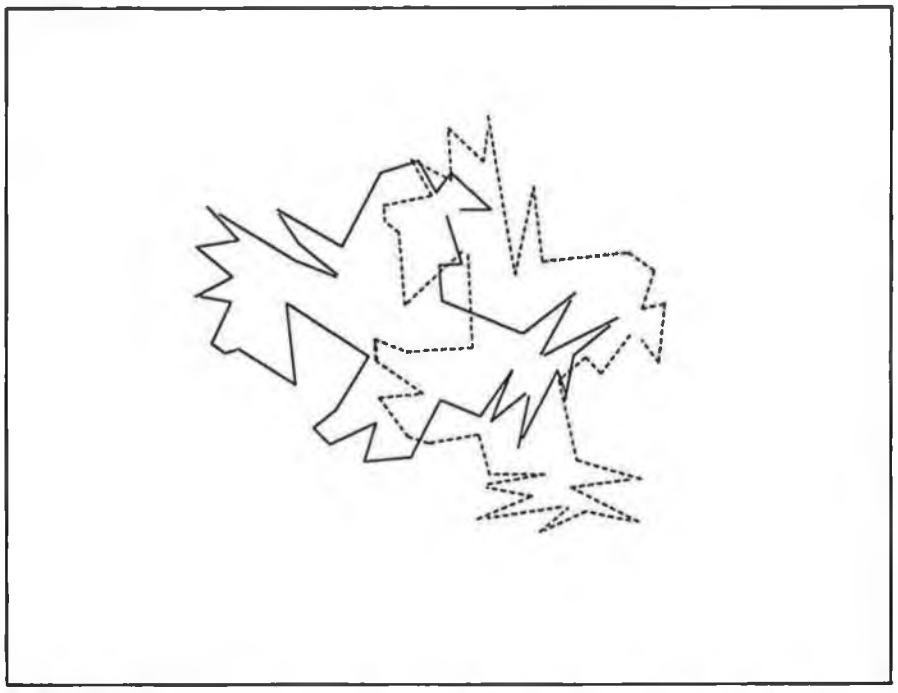

Figure 6.6.5 Superposition of Reference (solid) and Observed (dashed) Models, Projection on Camera

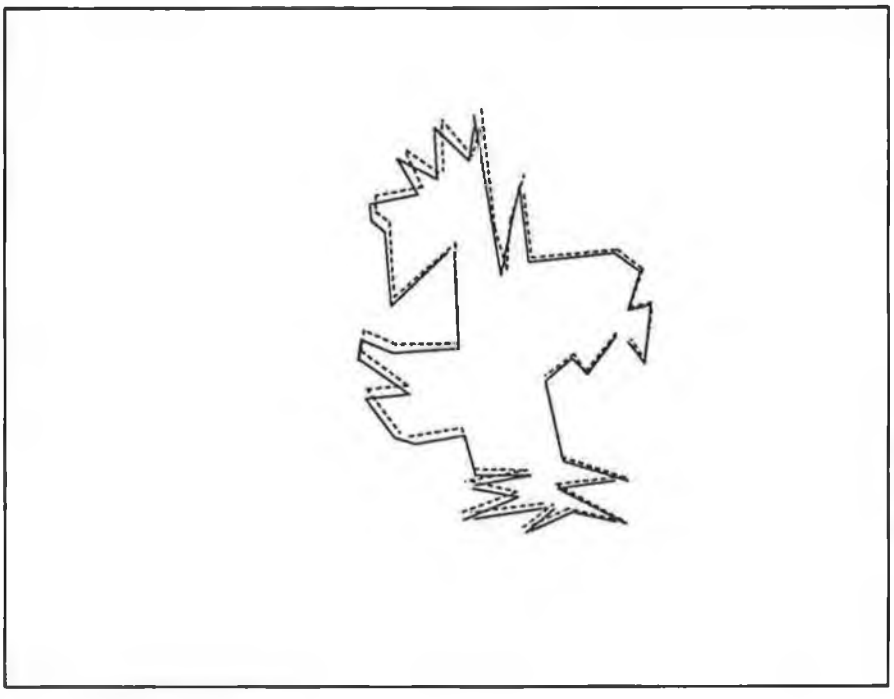

Figure 6.6.6 Superposition of Reference(solid) and Observed(dashed) Models (After Applying Computed Displacement), Projection on Camera

Second test object

Fig.6.6.7 shows an image of a connector. Fig.6.6.8 shows a 3D reconstructed model of the object. A sequence of six images was used for 3D model reconstruction. It forms the reference model of the object. The robot is manually guided to the desired grasping pose, and the gripper pose is stored. Fig.6.6.9 shows an image of the displaced object, there is a translation and rotation in the displacement. Fig.6.6.10 shows a 3D reconstructed model of the displaced object. A sequence of six images was used for reconstruction. In both cases the robot arm follows the same motion sequence (stop and go). Fig.6.6.11 shows the reference and observed 3D models superimposed. The reference model consists of 48 segments, while the observed model consists of 64 segments.

Applying the displacement estimation technique to these two models, we generate twenty hypotheses consisting of three matched segment pairs. For all hypotheses initial displacement estimates are computed. Their values for five hypotheses are given in Table 6.6.3. All initial estimates are applied to the reference model in order to do their verification and to improve estimates. Table 6.6.4 shows number of matched segments between the transformed reference model and observed model and the final displacement estimates for the five hypotheses. As can be seen all of them yield the correct estimates of the displacement.

Hyp. No.	α (roll) deg.	β (pitch) deg.	γ (yaw) deg.	tx (mm)	ty (mm)	tz (mm)
1	3.5	1.5	35.6	16.5	9.8	10.1
2	1.3	3.9	33.7	15.1	8.8	9.9
3	4.5	4.3	35.1	17.9	9.4	10.9
4	3.3	2.4	37.7	15.7	10.5	11.3
5	2.6	5.1	32.7	16.1	11.3	10.6

Table 6.6.3 Initial Displacement Estimates

Hyp.No.	α (roll) deg.	β (pitch) deg.	γ (yaw) deg.	*tx* (mm)	*ty* (mm)	*tz* (mm)	No. of Matches
1	1.2	1.5	35.9	15.9	10.2	10.3	26
2	1.2	1.5	35.9	15.9	10.2	10.3	26
3	1.2	1.5	35.9	15.9	10.2	10.3	26
4	1.2	1.5	35.9	15.9	10.2	10.3	26
5	1.2	1.5	35.9	15.9	10.2	10.3	26

Table 6.6.4 Final Displacement Estimates

To determine how good the estimates are, we apply the best computed estimate to the reference model and superimpose this transformed model on the observed model. Fig.6.6.12 shows the superposition of the models by using hypothesis No.1. The estimate of displacement is very good. This is also confirmed by controlling the robot arm to grasp the object using this displacement estimate.

Ten successive experiments were performed with this object placed at different locations. In all cases the displacement was correctly estimated and the robot arm was controlled correctly.

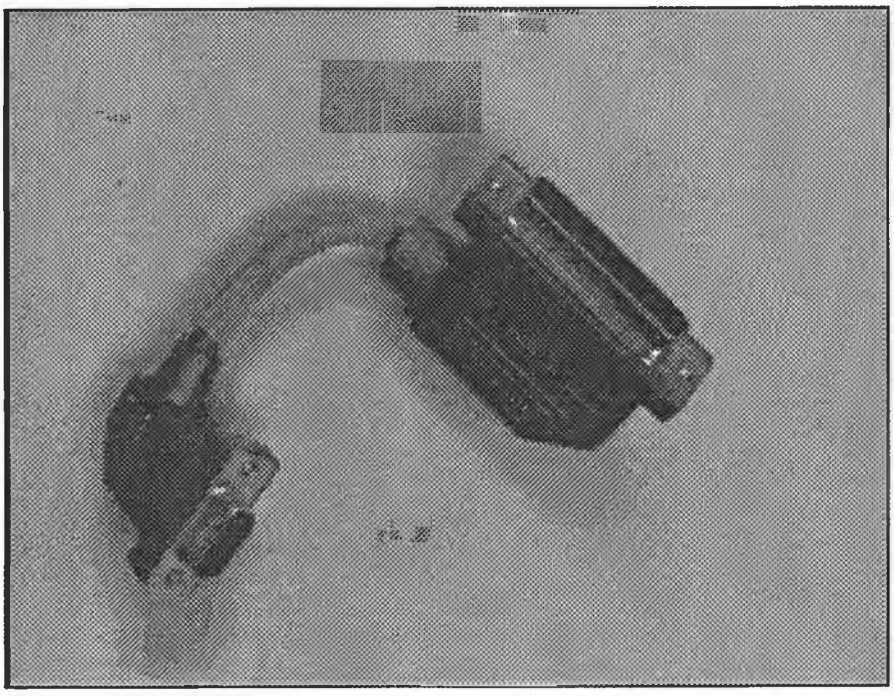

Figure 6.6.7 Raw Image of Object

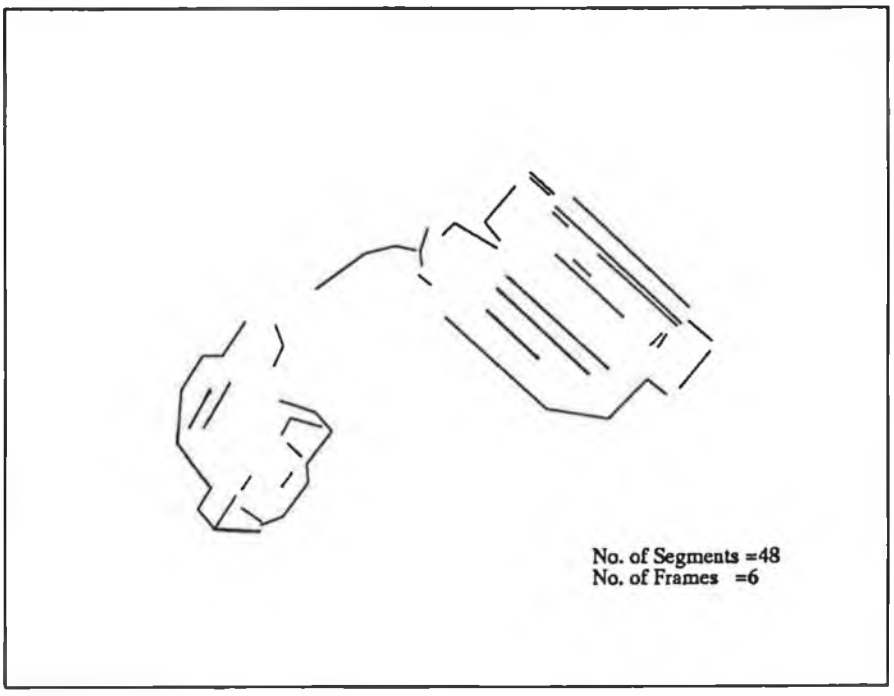

Figure 6.6.8 3D Model of Object, Projection on Camera

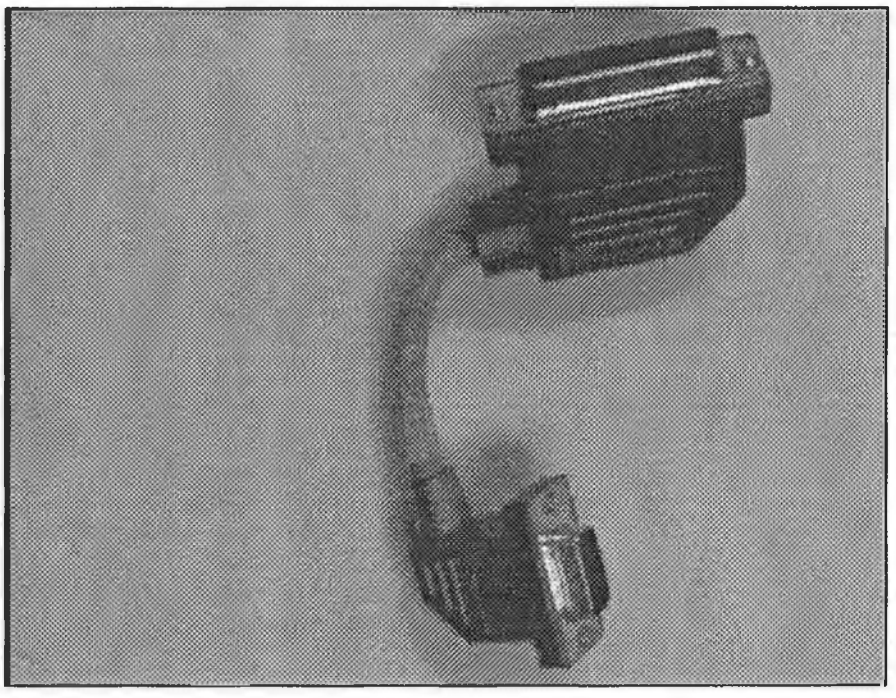

Figure 6.6.9 Raw Image of Displaced Object

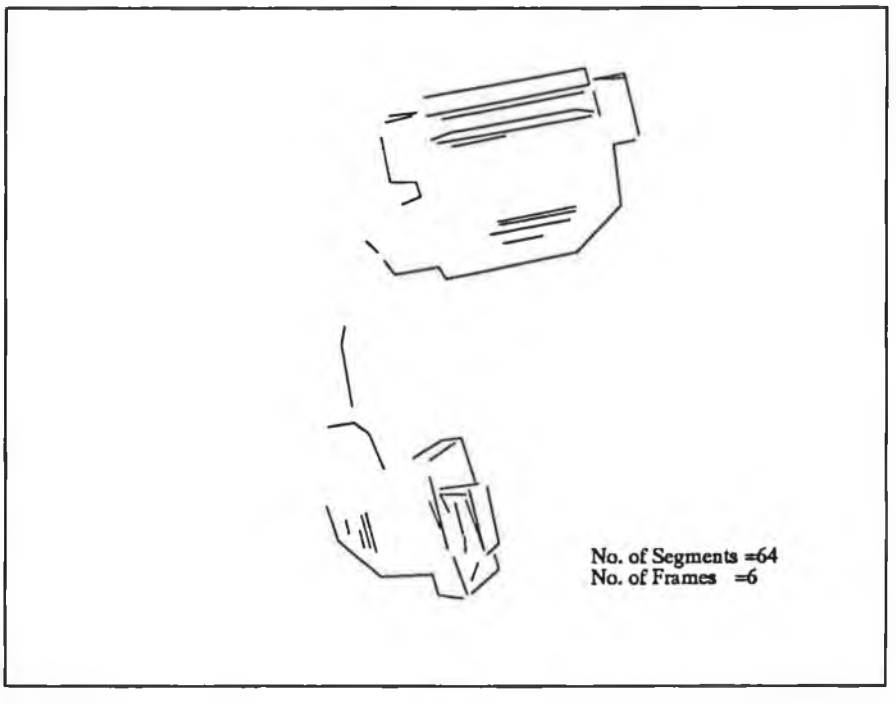

Figure 6.6.10 3D Model of Displaced Object, Projection on Camera

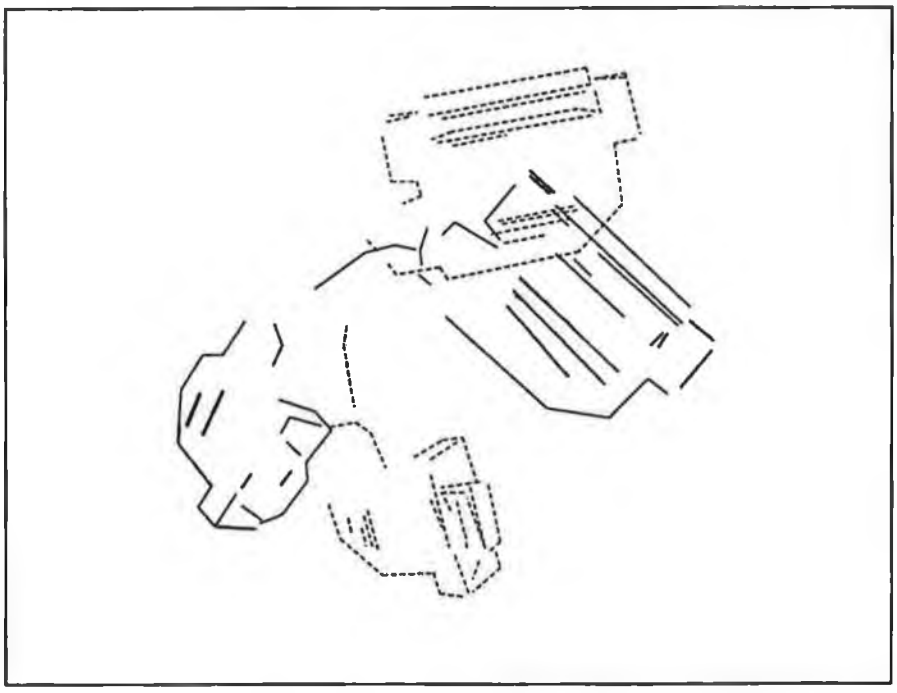

Figure 6.6.11 Superposition of Reference (solid) and Observed (dashed) Models, Projection on Camera

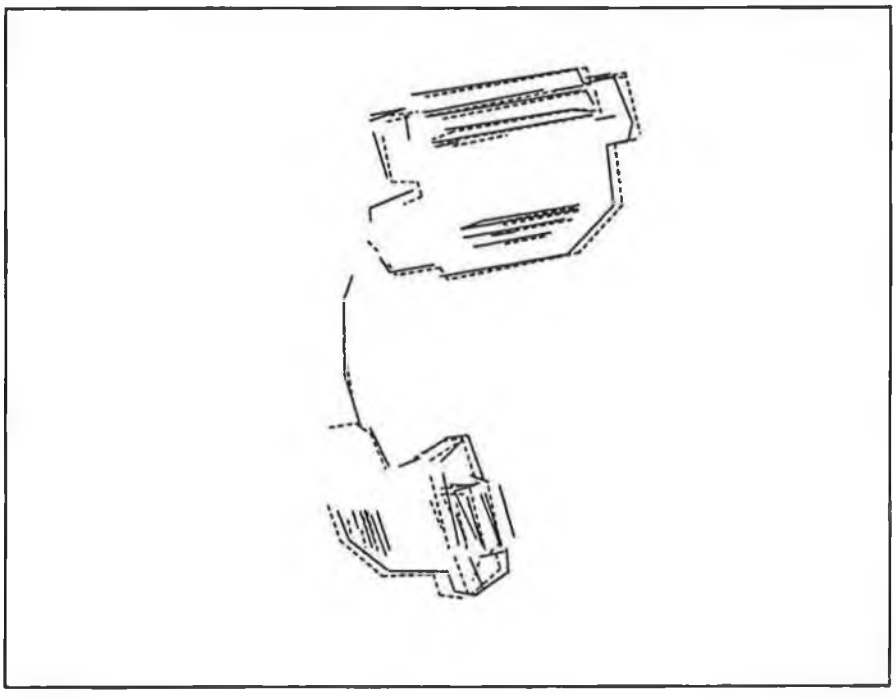

Figure 6.6.12 Superposition of Reference(solid) and Observed(dashed) Models (After Applying Computed Displacement), Projection on Camera

Third test object

Fig.6.6.13 shows an image of a switch box. Fig.6.6.14 shows a 3D reconstructed model of the box. A sequence of six images was used for 3D model reconstruction. It forms the reference model of the object. The robot is manually guided to the desired grasping pose, and the gripper pose is stored. Fig.6.6.15 shows an image of the displaced object, there is a translation and rotation in the displacement. Fig.6.6.16 shows a 3D reconstructed model of the displaced object. A sequence of six images was used for reconstruction. In both cases the robot arm follows the same motion sequence (stop and go). Fig.6.6.17 shows the reference and observed 3D models superimposed. The reference model consists of 43 segments, while the observed model consists of 47 segments.

Applying the displacement estimation technique to these two models, twenty hypotheses are generated consisting of three matched segment pairs. For all hypotheses initial displacement estimates are computed. Their values based on five correct hypotheses are given in Table 6.6.5.

Hyp. No.	α (roll) deg.	β (pitch) deg.	γ (yaw) deg.	*tx* (mm)	*ty* (mm)	*tz* (mm)
1	2.5	4.5	-30.6	16.5	9.8	0.5
2	1.3	2.9	-33.7	15.1	8.8	0.4
3	3.5	3.3	-30.1	17.9	8.4	0.6
4	3.3	3.4	-33.7	15.7	9.5	0.4
5	2.6	4.1	-35.7	16.1	9.3	1.2

Table 6.6.5 Initial Displacement Estimates

All initial estimates are applied to the reference model in order to do their verification and to improve estimates. Table 6.6.6 shows number of matched segments between the transformed reference model and observed model and the final displacement estimates for five hypotheses. As can be seen all of them yield the correct estimate of displacement.

Hyp.No.	α (roll) deg.	β (pitch) deg.	γ (yaw) deg.	*tx* (mm)	*ty* (mm)	*tz* (mm)	No. of Matches
1	1.3	1.2	-30.9	15.3	9.8	0.9	34
2	1.3	1.2	-30.9	15.3	9.8	0.9	34
3	1.3	1.2	-30.9	15.3	9.8	0.9	34
4	1.3	1.2	-30.9	15.3	9.8	0.9	34
5	1.3	1.2	-30.9	15.3	8.8	0.9	34

Table 6.6.6 Final Displacement Estimates

To determine how good the estimates are, the best computed estimate is applied to the reference model and superimpose on the observed model. Fig.6.6.18 shows the superposition of the models by using hypothesis No.1. The estimate of displacement is very good. This is also confirmed by controlling the robot arm to grasp the object using this displacement estimate.

Ten successive experiments were performed with this object placed at different locations. In all cases displacement was correctly estimated and the robot arm was controlled correctly.

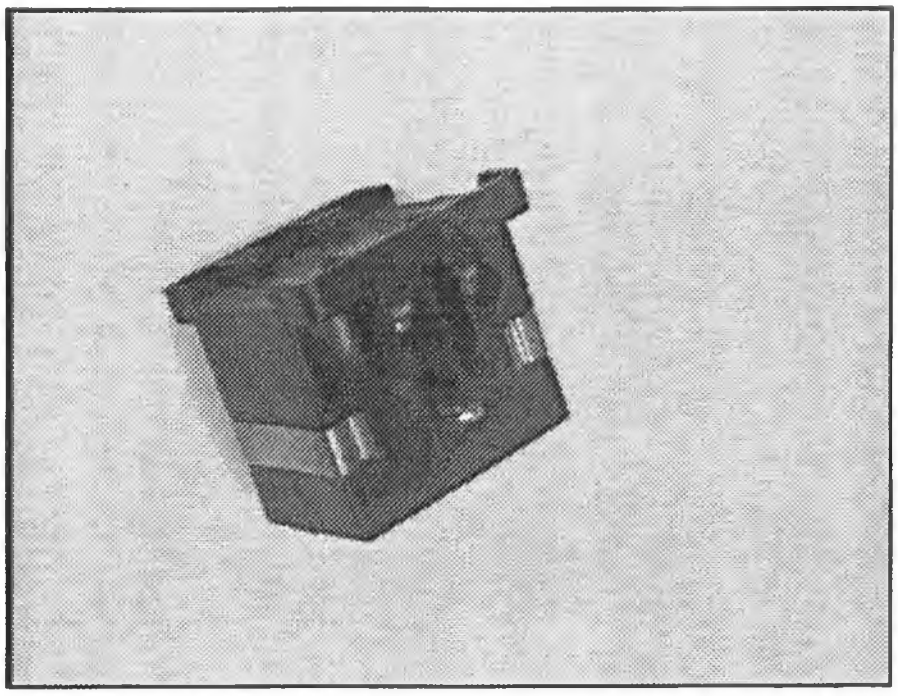

Figure 6.6.13 Raw Image of Object

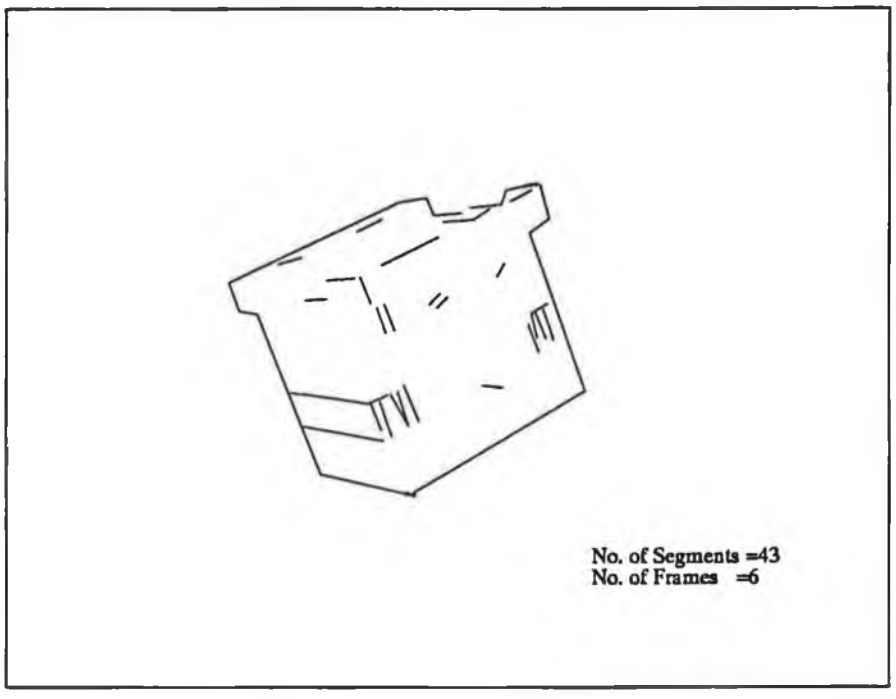

Figure 6.6.14 3D Model of Object, Projection on Camera

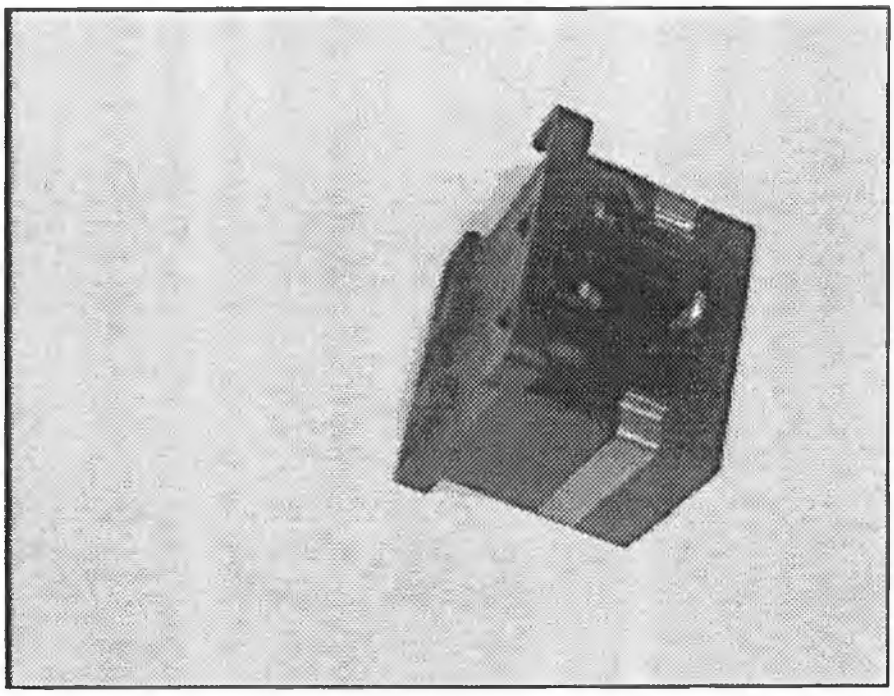

Figure 6.6.15 Raw Image of Displaced Object

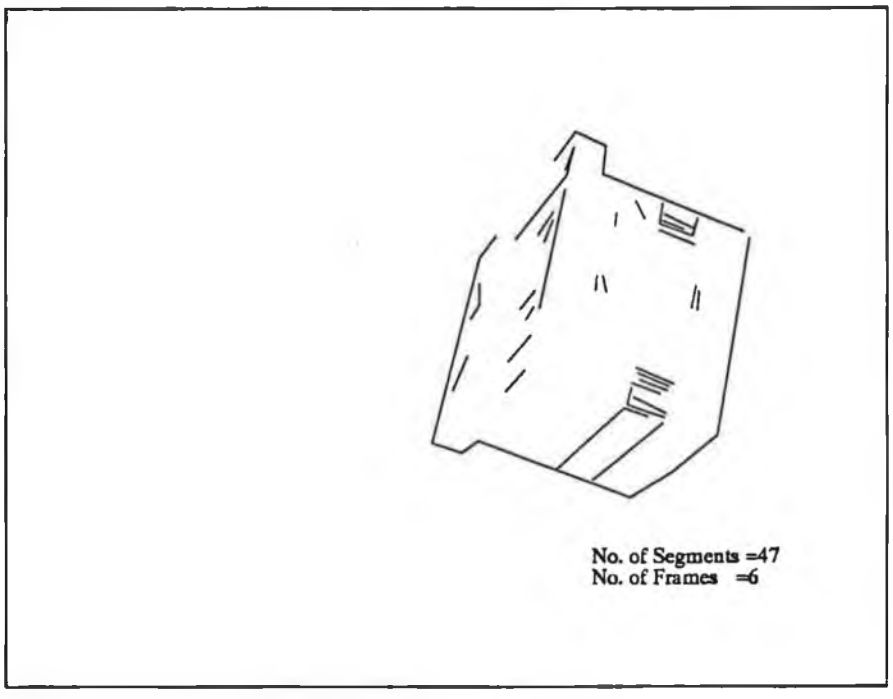

Figure 6.6.16 3D Model of Displaced Object, Projection on Camera

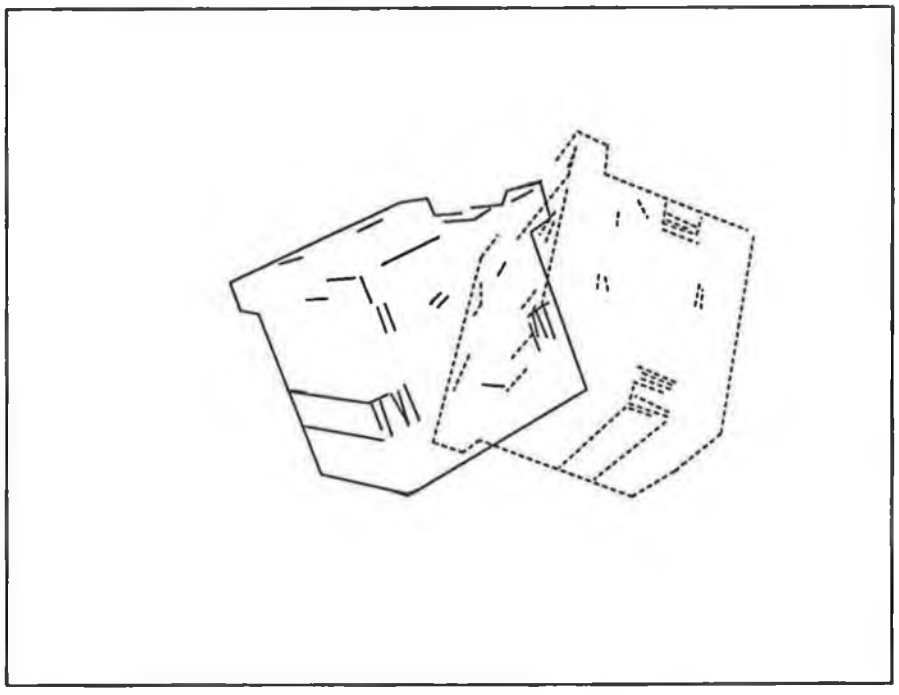

Figure 6.6.17 Superposition of Reference (solid) and Observed (dashed) Models, Projection on Camera

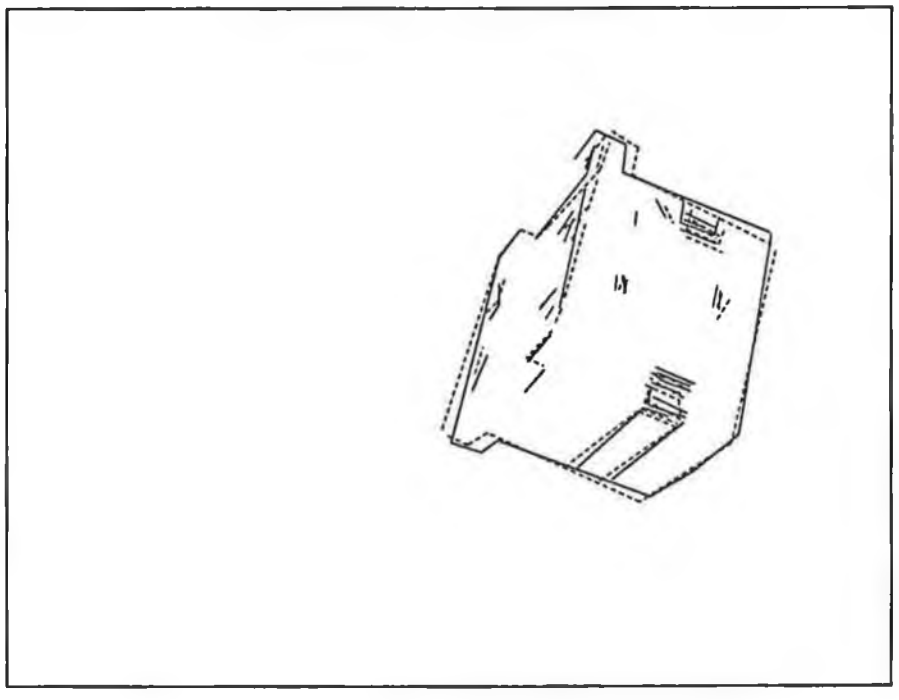

Figure 6.6.18 Superposition of Reference(solid) and Observed(dashed) Models (After Applying Computed Displacement), Projection on Camera

Multiple objects test

Fig.6.6.19 shows an image of the connector and the switch box. Fig.6.6.20 shows a 3D reconstructed model of the objects. The sequences of six images were used for 3D models reconstruction. Applying the above described algorithm the objects were recognized and manipulated. Fig.6.6.21 shows the observed model after recognising and extracting the connector. Fig.6.6.22 shows the observed model after recognising and extracting the switch box.

Ten successive experiments were performed with these objects placed at different locations. In all cases objects were recognised and displacements were correctly estimated and the robot arm was controlled correctly.

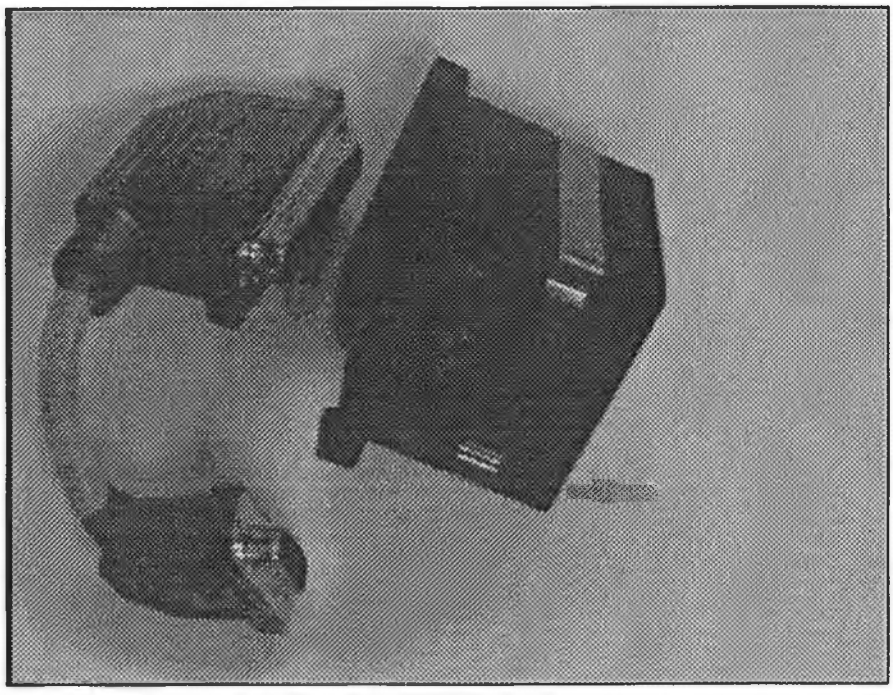

Figure 6.6.19 Raw Image of Objects

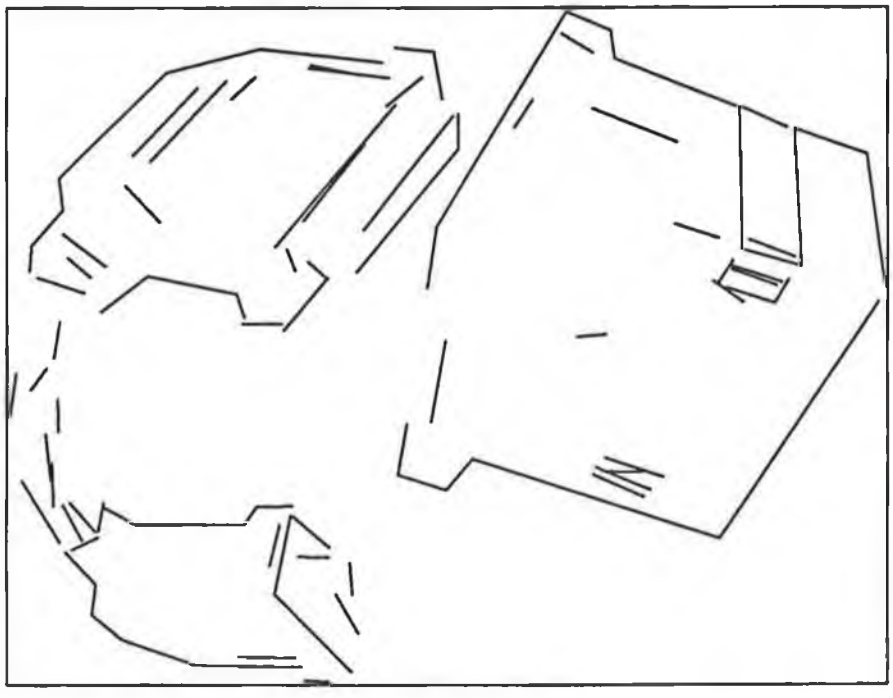

Figure 6.6.20 3D Model of Objects, Projection on Camera

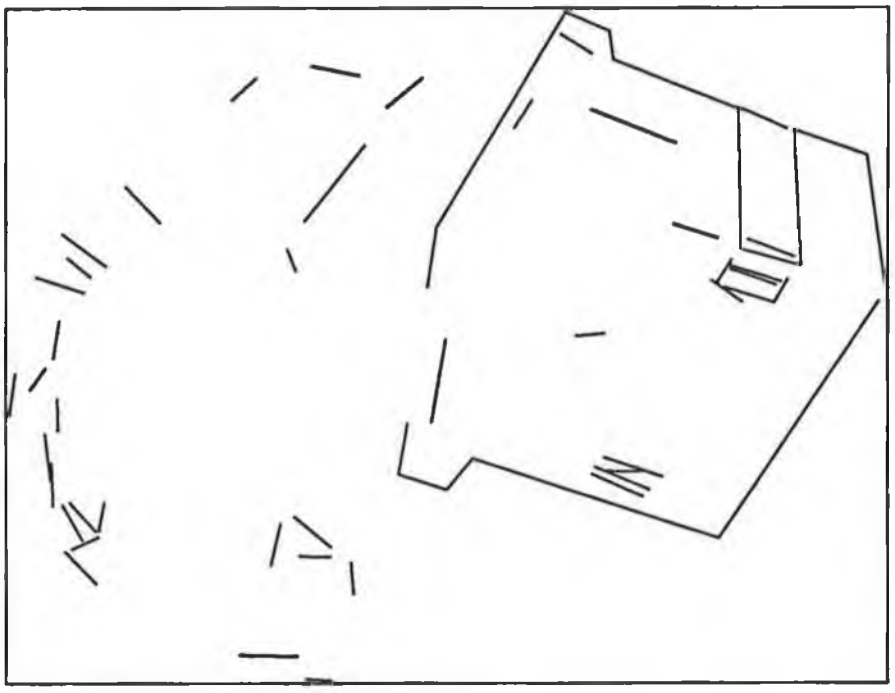

Figure 6.6.21 3D Model of Objects After Recognising and Extracting One Object, Projection on Camera

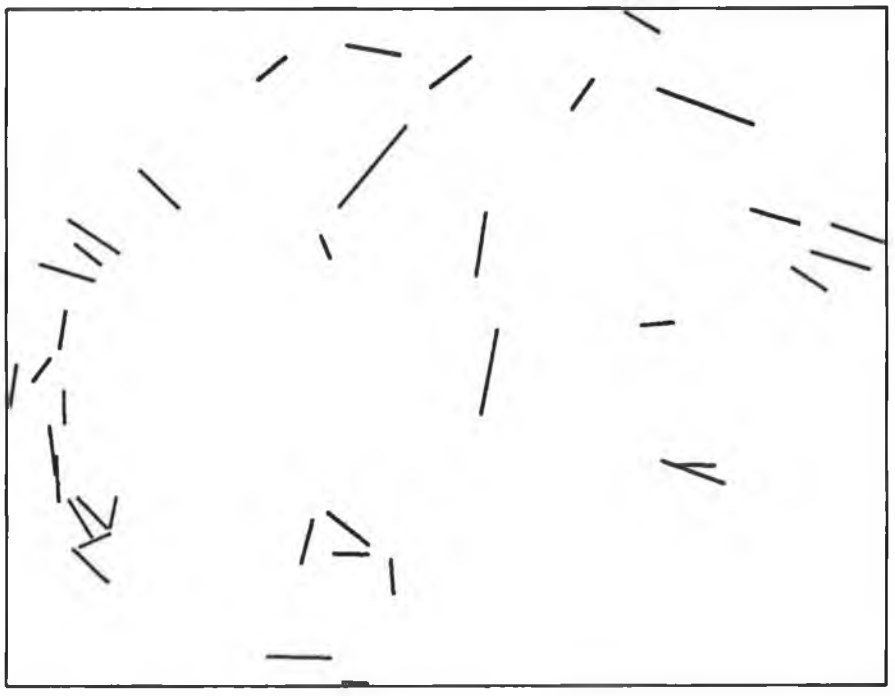

Figure 6.6.22 3D Model of Objects After Recognising and Extracting Second Object, Projection on Camera

6.8 Conclusions

An approach for robot arm visual control has been developed. The control task considered is to determine necessary position and orientation of the robot gripper in order to grasp the arbitrarily placed 3D object. Basically, the control scheme is based on the 3D displacement estimation between the reference and observed 3D models of the object.

In order to determine displacement, we have presented an approach based on "hypothesise and verify" paradigm for matching two 3D models and computing the 3D displacement between them. The rigidity constraints are used to generate hypotheses of segment correspondences between two models. It has been shown that a unique displacement can be computed from minimum three pairings of 3D segments. The uncertainty of measurements has been integrated into the formalism of the rigidity constraints. An initial estimate of the displacement can then be computed for each hypothesis. In order to compute the displacement a closed-form algorithm is used. This initial estimate is applied to the whole reference model in an attempt to verify and improve it. Matching is performed between the transformed reference model and the observed model. Finally, the best hypothesis is retained, as one which has the maximal number of matched segments, and the final estimate of the displacement is computed from all matched segments. This displacement is used to control the robot arm. The algorithm was successfully tested with various objects and displacements. The algorithm has been successfully extended to handle multiple objects manipulation and recognition.

Chapter 7

Conclusions and Suggestions

To enable robot manipulators to manipulate arbitrarily placed 3D object under sensory control is one of the key issues in successful robotic applications. Such robot sensors should be capable of providing 3D information about objects in order to accomplish above mentioned tasks. Such sensors should also provide the means for multisensor or multimeasurement integration in order to minimize the impact of measurement noise. Finally, such 3D information should be efficiently used for performing desired tasks. The work outlined in this thesis has attempted to solve some of these problems.

The purpose of this chapter is to summarize and evaluate the contributions made by this work. Further directions for research are also suggested.

7.1 Review of research contributions

A novel computational frame work for solving some of above mentioned problems has been developed in this thesis. In this work a vision (camera) sensor in conjunction with a robot manipulator is used to estimate 3D structure of objects within a class of objects. The objects are assumed to be well modelled as polyhedra.

The camera is mounted on the robot manipulator to take advantage of its mobility. The moving camera permits us to resolve traditionally very difficult vision problems more easily. Using image sequences permits us to cope with inherently noisy image measurements, by fusing information over image sequences and consequently minimising the impact of adverse measurement noise. The 3D visually derived object structure is used for the robot control.

In order to attain these goals the following issues have been considered. First, the issues of system calibration. Second, the issues of image processing and representation. Third, the issues of 3D vision based structure estimation. Fourth, the issues of visual robot control and object recognition. The results we have achieved during the course of this work are summarised below.

In Chapter 3 a modification to an existing camera calibration technique, that permits us to determine accurately parameters of the nonlinear camera model using only closed form (noniterative) computations, has been introduced. A new statistical analysis of the calibrated camera parameters that permits us to asses the accuracy of the obtained camera parameters, to fuse multiple data and to detect unreliable and unstable calibrations, has been introduced. A new statistical measure to evaluate the performance of the calibrated camera in 3D measurement applications has been proposed. The usefulness of the proposed methods have been proven.

In Chapter 4 image processing methods for straight line feature extraction have been presented. Minor contributions such as improving robustness and efficiency of polygonal approximation and efficient data structuring have been introduced.

In Chapter 5 a new technique for 3D structure estimation from monocular image sequences, using known camera motion, based on tracking line segments over image sequences has been introduced. The tracking process consists of prediction, matching and updating stages. These stages are handled in the Kalman filtering

framework of covariance based prediction, matching and updating. The prediction stage of the tracking process does not use heuristics about motion in the image plane and applies for the arbitrary camera motion. The prediction is based on assumptions about object structure (i.e. a rough knowledge of a distance between camera and an object is assumed known and the depth extent of the object is small compared with the camera-object distance) for the initialisation phase, the rest of the tracking process is based on the estimated object structure. The matching stage is based on the simple nearest-neighbour matching algorithm using the Mahalobonis (statistical) distance as a similarity measure. The updating stage is based on standard Kalman filter estimation algorithm.

The main advantages of the new method are as follows: It has been shown that for some assumptions about the object structure, the prediction of object image motion can be based on the 3D structure constraints and not on heuristics about the motion of image features. This fact has implied that successful tracking can be realised in the case of arbitrary camera motion and large interframe displacements. It has been shown that the determination of uncertainties and other system parameters is very simple having a firm analytical basis. No parameter is tuned or guessed by numerous trials and errors. The technique has been implemented to provide 3D information for a robot manipulator. The experimental results show the reliability and accuracy of the proposed technique. The 3D structure which is recovered very quickly (a few images) converges to precision on the order of a millimetre provided the system is well calibrated and system parameters are tuned properly.

In Chapter 6 a new approach for robot arm visual control has been introduced. The control task considered was to determine necessary position and orientation of the robot gripper in order to grasp the arbitrarily placed 3D object. Basically, the control scheme is based on the 3D displacement estimation between the reference and observed 3D models of the object. The "hypothesise and verify" approach has been used to estimate 3D displacement. The main characteristics of the new control

approach are as follows: No geometrical knowledge about 3D object to be manipulated is required. The system automatically builds the geometrical representation of the object. The specification of grasping demands is very simple, and is done by manual robot guidance. The proposed algorithm was quite successfully tested for a class of objects (polyhedra). The system has been successfully extended for multiple object manipulation and object recognition.

7.2 Further directions of research

The methods presented in this thesis have proven to be quite robust and efficient, in spite of the fact that they do not exploit a great deal of available information. We believe that these methods can be readily extended to incorporate other information such as proximity and connectivity between line segments. These constraints would make the 3D reconstruction, 3D displacement estimation and object recognition more robust. Another direction for the further research may be development of techniques to include other primitives in geometrical modelling, such as 3D curves and 3D surface patches, in order to model more complex industrial parts. Further, issues of sensor (camera motion) planing could be investigated.

References

[And85] R.L. Anderson, "Real Time Intelligent Visual Control of a Robot", *in Proc. of IEEE Workshop on Intelligent Control*, pp.89-94, 1985

[Aya86] N. Ayache and O. Faugeras, "HYPER: A new approach for the recognition and positioning of two dimensional objects", *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-8, pp.44-54, Jan.1986

[Aya89] N. Ayache and O. Faugers, "Maintaining Representations of the Environment of a Mobile Robots", *IEEE Transaction on Robotics and Automation*, Vol.5, No.6, Decembar 1989

[Aya91] N. Ayache, *Artificial Vision for Mobile Robots*, MIT Press Cambridge, 1991

[Aru87] K. Arun, T. Huang and S.Blostein, "Least -squares fitting of two 3D point sets", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-9, pp.698-700, Septembar 1987

[Bal82] D. Ballard, C. Brown, *Computer Vision*, Prentice Hall, 1982

[Bar78] Y. Bar-Shalom, "Tracking methods in a multitarget evironment", *IEEE Transaction on Automatic Control,* Vol. 23 , pp. 618-626

[Bar88] Y. Bar-Shalom and T.E. Fortmann, *Tracking and Data Association*, Academic Press: San Diego, CA: 1989

[Barr94] J.L.Barron, D.J.Fleet and S.S. Beauchemin, " Performance of Optical Flow Techniques", *International Journal of Computer Vision*, pp.43-77, 1994.

[Bol82] R. Bolles and R. Cain, " Recognizing and locating partially visible objects: The Local-feature-focus method", *International Journal Robotics Research*, Vol.1, No.3, pp.57-82, 1982

[Bow87] M. Bowman and A. Forrest, "Robot Model Optimization", *Int. Journal of Robotic Research*, 1987

[Che88] H. Chen and T. Huang, "Maximal matching of 3D points for multiple object motion estimation", *Pattern Recognition*, vol.21, no.2, pp.75-90, 1988

[Che87] H. Chen and T. Huang, "An algorithm for matching 3D line segments with application to multiple object motion estimation", *in Proc. IEEE Workshop Computer Vision*, Nov. 30- Dec.2, 1987, pp.151-156

[Cro88] J. L. Crowley, P. Stelmaszyk and C. Discours, "Measuring image flow by tracking edge lines", *in Proc. 2nd International Conference on Computer Vision,* Tampa, FL, pp. 658-664, 1988

[Cro92] J. L. Crowley, P. Stelmaszyk, T. Skordas and P. Puget, "Measurement and Integration of 3D Structures by Tracking Edge Lines", *International Journal of Computer Vision*, Vol.8, pp. 29-52, 1992.

[Cra86] J. J. Craig, *Introduction to Robotics: Mechanics and Control*, Addison-Weslley, Reading, Mass.

[Der90] R. Deriche and O. Faugeras, "Tracking line segments", *Proc 1st Europ.Conf. Comput. Vis.*, pp.259-268, France, 1990.

[Fau86] O. D. Faugeras and and G. Toscani, "Calibration problem for stereo", *in Proc. Int. Conf. Computer Vision and Pattern Recognition*, pp. 15-20, June 1986.

[Fau83] O. Faugeras and M. Hebert, " A 3D recognition and positioning algorithm using using geometrical matching between primitive surfaces", *in Proc. Int. Joint Conf. Artificial Intelligence*, Karlshrue, West Germany, Aug.1983, pp.996-1002

[Fau86] O. Faugeras and M. Hebert, "The representation, recognition and locating of 3D shapes from range data", *International Journal on Robotics Research*, vol.5, No.3, pp.27-52, 1986

[Fed89] J. T. Feddema and O.R. Mitchell, " Vision Guided Servoing with Feature-Based Trajectory Generation", *IEEE Tran. on Robotics and Automation,* Vol.5, No.5, pp.691-700, Oct. 1989

[Fed89] J. T. Feddema, " Automatic Selection of Image Features for Visual Servoing of a Robotic Manipulator", *in Proc. of IEEE Int. Conf. on Robotics and Automation*, pp.831-837, 1989

[Fre89] H. Freeman (Editor), *Machine Vision for Inspection and Measurement*, Academic Press Inc., 1989

[Fu87] K.S. Fu, R.C. Gonzalez and C.S.G. Lee, *Robotics, Control, Sensing, Vision and Inelligence*, McGraw-Hill , 1987

[Gel86] A. Gelb, *Applied Optimal Estimation*, MIT Press, Cambridge, MA 1986

[Gri84], W. Grimson and T. Lozano-Perez, "Model-based recognition and localization from sparse range or tactile data", *International Journal Robotics Research*, Vol.5, pp. 3-34, 1984.

[Grim87] W. Grimson and T. Lozano-Perez, "Localizing overlapping parts by searching the interpretation tree", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-7, no.4, pp. 469-482, December 1987

[Has92] H. Hashimoto at al, "Self-organizing visual servo system based on neural networks", *IEEE Control System*, pp.31-36, April 1992

[Hol94] N. Hollinghurst and R. Cipolla, "Uncalibrated stereo hand-eye coordination", *Image and Vision Computing*, Vol.12, no.3, pp.187-192, April 1994

[Hora84] P. Horaud and R. Bolles, " 3DPO's strategy for matching three-dimensional objects in range data", *in Proc. Int. Conf. Robotics and Automation*, pp.78-85, March 1984

[Hor86] B.K.P. Horn, *Robot Vision*, MIT Press, Cambridge, 1986

[Hua86] T.S. Huang, S.D. Blostein and E.A. Margerum, "Least squares estimation of motion parameters from 3D point correspondences", *in Proc. IEEE Conference Computer Vision and Pattern Recognition*, Miami Beach, FL, June 24-26, 1986

[Iza87] A. Izaguirre, J. Summers and P. Pu, "A new Development in Camera Calibration", *Int. Journal of Robotic Research*, 1987

[Jaz70] J.E. Jazwinski, *Stochastic Processes and Filtering Theory*, Academic Press Press: New York, 1970

[Kal60] R. E. Kalman, " A new approach to linear filtering and prediction problems", *Basic Eng.* pp. 35-45, 1960

[Kan92] K. Kanatani, " Statistical analisys of focal lenght calibration using vanishing points", *IEEE Tran. on Robotics and Automation*, vol. 8, No. 6, pp 767-775, Dec. 1992.

[Knu75] D. Knuth, *Sorting and Searching in the Art of Computer Programming*, Addison Welsey, Reading, 1975, Volume 3

[Kri89] D. J. Kriegman, E. Triendi and T.O. Binford, "Stereo Vision and Navigation for Mobile Robots", *IEEE Transactions on Robotics and Automation"*, Vol.5, No.6, pp. 792-802, December 1989

[Kum92] R. Kumar and A. Hanson, "Model Extension and Refinement Using Pose Recovery Techniques", *Journal of Robotic Systems*, Vol 9, pp. 753-771,1992

[Len87] R. K. Lenz and R. Y. Tsai , "Techniques for calibration of the scale factor and image center for high accuracy 3D machine vision metrology", *in Proc. IEEE Int. Conf. Robotics and Automation*, Mar. 1987, pp. 68-75.

[Marr82] D. Marr, *Vision*, W.H. Freeman and Co.,1982

[Matt89] L. Matthies and T. Kanade, " Kalman Filter-based Algorithm for Estimating Depth from Image Sequences", *International Journal of Computer Vision*, Vol.3, pp.209-236, 1989

[Matt87] L. Matthies and S.A. Shafer, "Error modeling in stereo navigation", *IEEE Journal of Robotics and Automation*, pp.239-248, 1987

[Pap93] N. P. Papanikolopoulos, P. K. Khosla and T. Kanade, "Visual Tracking of a Moving Target by a Camera Mounted on a Robot: A Combination of Control and Vision", *IEEE Transactions on Robotics and Automation,* Vol.9, No. 1, pp. 14-34, February 1993

[Pav82] T. Pavlidis, *Agorithms For Graphics and Image Processing*, Computer Science Press, 1982

[Paul81] *Robot Manipulator: Mathematics, Programming and Control*, MIT Press, Cambridge, 1981

[Pol87] S. Pollard, J. Porrill, J. Mayhew and J. Frisby, " Matching geometrical descriptions in three space", *Image Vision Computing*, Vol.5, pp.73-78, May 1987

[Prl93] N. Prljaca and H. McCabe, "3D Geometrical Model Building for Visual Robot Control ", *In SPIE Proc. ,* Vol.2056, pp.296-306, Boston, September 1993

[Prl94] N. Prljaca and H. McCabe "3D Structure Estimation for Vision Based Robotics", *In SPIE Proc.,* Vol.2354, pp.66-78, Boston, Novembar 1994

[Pus87] G. Puskorius and I. Feldkamp , "Calibration of Robot Vision", *in Proc. IEEE Int. Conf. on Robotics and Automation*, 1987.

[Pug83] A. Pugh, *Robot Vision*, IFS/Springer-Verlag, 1982

[Pug85] A. Pugh, "Robot Sensors - A personal view", *in Proc. of the 5th International Conference on Robot Vision and Sensory Controls* (Ed. N.J. Zimmerman), 35-46

[Pra91] W. Pratt, *Digital Image Processing*, John Wiley & Sons, 1991

[San82] A.C. Sanderson and L.E. Weiss, "Adaptive Visual Servo Control of Robots", Robot Vision, Ed. A. Pugh, pp.107-116, IFS, UK, 1982.

[Sar82] P. Saraga and B.M. Jones, "Simple Assembly Under Visual Control", Robot Vision, Ed. A. Pugh, pp.107-116, IFS, UK, 1982.

[Shi89] Y. Shiu and S. Ahmad, "Calibration of Wrist-Mounted Robotic Sensors by Solving Homogeneous Transform Equations of the Form AX=XB", *IEEE Transactions on Robotics and Automation*, Vol.5, No.1, February 1989.

[Saw93] H. Sawhney and A. Hanson, "Tracability as a Cue for Potential Obstacle Identification and 3D description", *International Journal of Computer Vision*, pp. 237-265, 1993.

[She92] C. Shekhar and R. Chellapa, "Passive Ranging Using Moving Camera", *Journal of Robotic Systems*, Vol.9, pp.729-752, 1992

[Tsa87] R. Y. Tsai, "A versatile camera calibration technique for high accuracy 3D machine vision metrology using of-the-shelf TV cameras and lenses", *IEEE J. Robotics Automation* , vol. RA-3, no. 4, pp 323-344, Aug. 1987.

[Tsa89] R. Y. Tsai and R. Lenz, "A New Technique for Fully Autonomous and Efficient 3D Robotics Hand/Eye Calibration", *IEEE Transactions on Robotics and Automation,* Vol.5, No. 3, June 1989.

[Tsa84] R. Tsai, T.Huang, "Uniqueness and Estimation of Three Dimensional Motion Parameters of Rigid Objects with Curved Surfaces", *IEEE Trans. on PAMI*, Vol.6, 1984

[Ull79] S. Ullman, *The Interpretation of Visual Motion,* MIT Press, Cambridge, 1979

[Ver91] D. Vernon, *Machine Vision*, Prentice Hall, 1991

[Wan92] C. Wang, "Extrinsic Calibration of a Vision Sensor Mounted on a Robot", *IEEE Transactions on Robotics and Automation*, Vol.8, No. 2, April 1992.

[Wen92] J. Weng, P. Cohen and M. Herniou, "Camera calibration with distortion models and accuracy evaluation", *IEEE Tran. on Pattern Analysis and Mach. Intell.,* vol. 14, No. 10, pp 965-980, Oct. 1992.

[Wen92] J. Weng, P. Cohen and N. Rebibo, "Motion and Structure Estimation from Stereo Image Sequences", *IEEE Transaction on Robotics and Automation* , Vol. 8, No. 3, June 1992

[Wes87] L.E. Wess, A.C. Sanderson and C.P. Neuman, "Dynamic Sensor Based Control of Robots with Visual Feedback", *IEEE Journal of Robotics and Automation*, Vol.RA-3, No.5, pp.404-417, 1987

[Wij93] W.S. Wijesoma, D. F. Wolf e and R.J. Richards, "Eye-to-Hand coordination for vision-guided robot control applications" , *Int. Journal of Robotics research"*, Vol. 12 , No. 1, pp.65-78, MIT Press, Cambridge, USA, 1993.

[Zha90] Z. Zhang and O. Faugeras, "Determining motion from 3D line segments: A comparative study", *in Proc. Brit. Machine Vision Conf.* , pp.85-90, September, 1990

[Zha92] Z. Zhang and O. Faugeras, "Estimation of Displacement from Two 3D Frames Obtained from Stereo", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.14, No.12, pp. 1141-1155, December 1992

Appendix A

Rotation Parametrizations

Any rotation can be represented by an orthogonal 3x3 matrix R with determinant equal to one, that is

$$RR^T = I, \quad \det(R) = 1$$

Euler first showed that this matrix has the three-dimensional parametrization. Here we present the common rotation parametrizations.

Parametrization by Euler angles

The most common way to represent the rotation R is to decompose it into the product of three rotations of a predefined fixed coordinate system about axes x, y and z. Denoting these angles of rotation by (α, β, γ), let

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

]The angles (α, β, γ) are generally called roll, pitch and yaw [Pau82], [Cra86], and are shown in Fig.(AA.1).

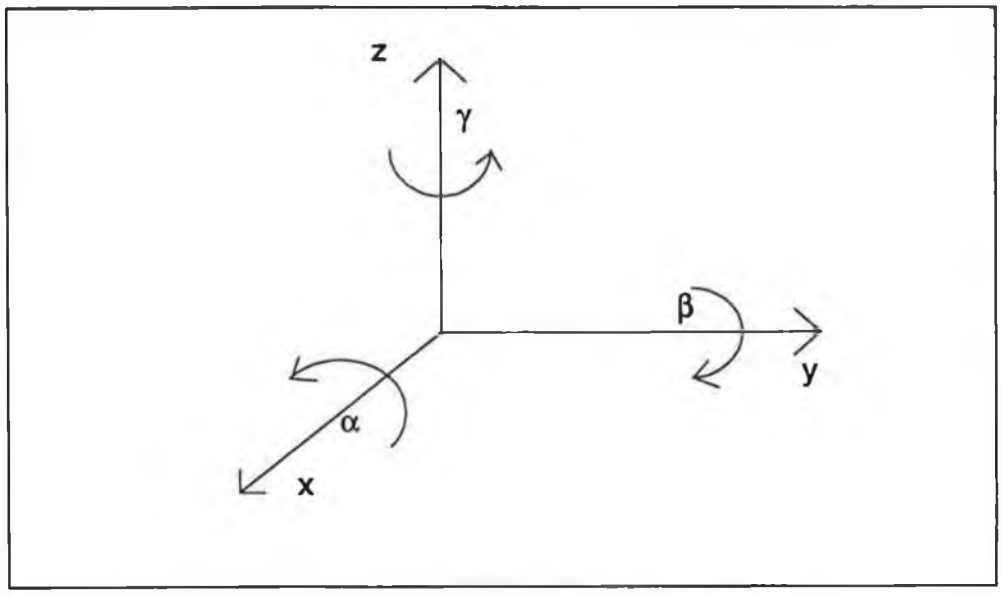

Figure AA.1 Rotation $R(\alpha, \beta, \gamma)$

The rotation $R(\alpha, \beta, \gamma)$ is thus written

$$R(\alpha,\beta,\gamma) = \begin{bmatrix} \cos(\alpha)\cos(\beta) & \cos(\alpha)\sin(\beta)\sin(\gamma) - \sin(\alpha)\cos(\gamma) & \cos(\alpha)\sin(\beta)\cos(\gamma) + \sin(\alpha)\sin(\gamma) \\ \sin(\alpha)\cos(\beta) & \sin(\alpha)\sin(\beta)\sin(\gamma) + \cos(\alpha)\cos(\gamma) & \sin(\alpha)\sin(\beta)\cos(\gamma) - \cos(\alpha)\sin(\gamma) \\ -\sin(\beta) & \cos(\beta)\sin(\gamma) & \cos(\beta)\cos(\gamma) \end{bmatrix}$$

To determine the Euler angles of the rotation R (for which we denote elements by R_{ij}), let

$$\eta = (R_{11}^2 + R_{21}^2)$$

we have the relations

$$\cos(\alpha) = \frac{R_{11}}{\eta} \qquad \sin(\alpha) = \frac{R_{21}}{\eta}$$
$$\cos(\beta) = \eta \qquad \sin(\beta) = R_{31}$$
$$\cos(\gamma) = \frac{R_{33}}{\eta} \qquad \sin(\gamma) = \frac{R_{32}}{\eta}$$

which determine the angles (α, β, γ).

Parametrization by rotation vector

Another common way to represent rotation R is by a rotation vector. This representation is based on the fact that any rotation R has an invariant axis with normal direction vector n, say. Vectors collinear to n are invariant by R, while orthogonal vectors undergo a rotation of angle θ in the plane orthogonal to n. We denote the rotation by the angle θ about the n axis by $R(n, \theta)$. It is shown in Fig.(AA.2).

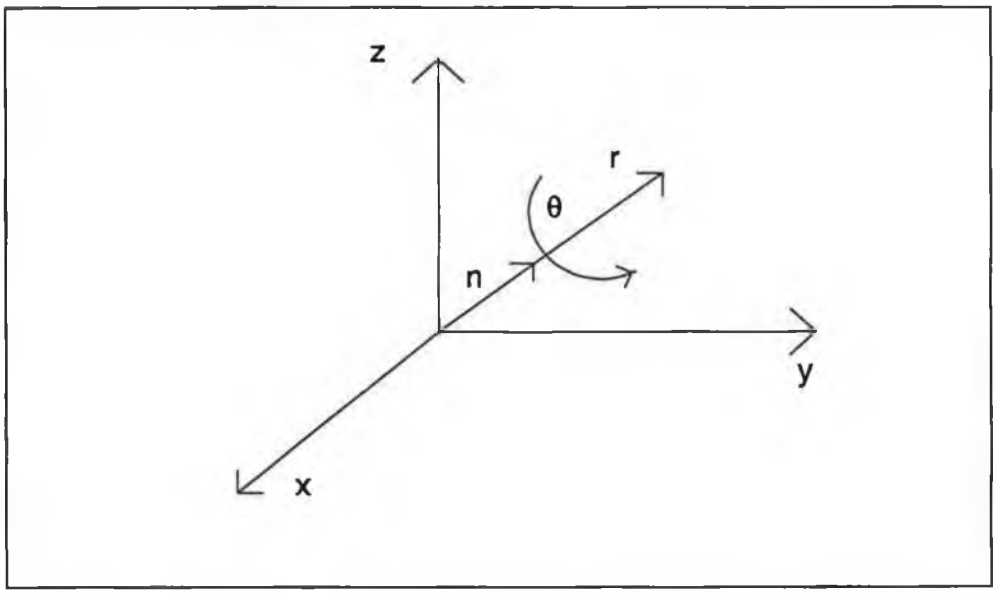

Figure AA.2 Rotation $R(n, \theta)$

Therefore, for any vector r, we can write

$$\theta = \|r\|, \quad \theta \le 2\pi, \quad n = \frac{r}{\|r\|} = (n_x, n_y, n_z)^T$$

The vector r is mapped into the rotation R by the means of Rodrigues' formula

$$R = \begin{bmatrix} \cos(\theta)+r_x^2 g(\theta) & r_x r_y g(\theta)-r_z f(\theta) & r_x r_z g(\theta)+r_y f(\theta) \\ r_x r_y g(\theta)+r_z f(\theta) & \cos(\theta)+r_y^2 g(\theta) & r_y r_z g(\theta)-r_x f(\theta) \\ r_x r_z g(\theta)-r_y f(\theta) & r_y r_z g(\theta)+r_x f(\theta) & \cos(\theta)+r_z^2 g(\theta) \end{bmatrix}$$

where

$$f(\theta) = \frac{\sin(\theta)}{\theta}, \qquad g(\theta) = \frac{1-\cos(\theta)}{\theta^2}$$

The rotation matrix R is mapped into the vector r by

$$\cos(\theta) = \frac{R_{11}+R_{22}+R_{33}-1}{2}, \quad r_x = \frac{R_{32}-R_{23}}{2f(\theta)}, \quad r_y = \frac{R_{13}-R_{31}}{2f(\theta)}, \quad r_z = \frac{R_{21}-R_{12}}{2f(\theta)}$$

Appendix B

3D Displacement Estimation from 3D Point Correspondences

Let $y_1,...,y_n$ be N points in Euclidean 3-space. Let R be a rotation matrix and T be a translation vector. Let $x_1,...,x_n$ be the points in Euclidean 3-space which match $y_1,...,y_n$. Each x is the same rigid body motion of y. Hence each y is obtained as a rotation of x plus a translation plus noise.

$$y_n = Rx_n + T + \varepsilon$$

The problem of displacement estimation is to infer R and T from $y_1,...,y_n$ and $x_1,...,x_n$. To determine R and T we can set up a constrained least squares problem. We will minimise the criterion

$$\min_{(R,T)} \sum_{n=1}^{N} \gamma_n \left\| y_n - Rx_n - T \right\|^2, \text{ subject to } RR^T = I \qquad \textbf{(AB.1)}$$

the constraint $RR^T = I$, is a consequence that R is a rotation matrix, and γ_n are weighting factors. To be able to express these constraints using Lagrangian multipliers we let

$$R = \begin{bmatrix} r_1^T \\ r_2^T \\ r_3^T \end{bmatrix}, \quad \text{where each } r_i \text{ is a 3x1 vector}$$

The constraint $RR^T = I$, then amounts to six constraint equations

$$r_1^T r_1 = 1,\ r_2^T r_2 = 1,\ r_3^T r_3 = 1,\ r_1^T r_2 = 0,\ r_1^T r_3 = 0,\ r_2^T r_3 = 0$$

The least squares problem with constraints given by Eq.(AB.1) can be written as minimising

$$\sum_{n=1}^{N}\sum_{k=1}^{3}\gamma_n(y_{nk}-r_k^T x_n-t_k)^2+\sum_{k=1}^{3}\lambda_k(r_k^T r_k-1)+2\lambda_4 r_1^T r_2+2\lambda_5 r_1^T r_3+2\lambda_6 r_2^T r_3 \qquad \textbf{(AB.2)}$$

where

$$x_n=\begin{bmatrix}x_{n1}\\x_{n2}\\x_{n3}\end{bmatrix},\quad y_n=\begin{bmatrix}y_{n1}\\y_{n2}\\y_{n3}\end{bmatrix},\quad \mathrm{T}=\begin{bmatrix}t_1\\t_2\\t_3\end{bmatrix}$$

Taking partial derivative of Eq.(AB.2) with respect to T, and setting these partials to zero results in

$$\sum_{n=1}^{N}\gamma_n(y_n-Rx_n-T)=0$$

By rearranging we obtain

$$T=\bar{y}-R\bar{x},\quad \text{where } \bar{x}=\frac{\sum_{n=1}^{N}\gamma_n x_n}{\sum_{n=1}^{N}\gamma_n} \text{ and } \bar{y}=\frac{\sum_{n=1}^{N}\gamma_n y_n}{\sum_{n=1}^{N}\gamma_n} \qquad \textbf{(AB.3)}$$

Thus once R is known, T is quickly determined from Eq.(AB.3). Substituting the value for T Eq.(AB.3) into Eq.(AB.2) we obtain

$$\sum_{n=1}^{N}\gamma_n\sum_{k=1}^{3}(y_{nk}-\bar{y}_n-r_k^T(x_n-\bar{x}))^2+\sum_{k=1}^{3}\lambda_k(r_k^T r_k-1)+2\lambda_4 r_1^T r_2+\lambda_5 r_1^T r_3+\lambda_6 r_2^T r_3$$

(AB.4)

Now we take partial derivatives of Eq.(AB.4) with respect to the components of each r_n. Setting these partial derivatives to zero we obtain

$$\sum_{n=1}^{N}\gamma_n(x_n-\bar{x}(x_n-\bar{x})^T r_1+\lambda_1 r_1+\lambda_4 r_2+\lambda_5 r_3=\sum_{n=1}^{N}\gamma_n(y_{n1}-\bar{y}_1)(x_n-\bar{x})$$

$$\sum_{n=1}^{N}\gamma_n(x_n-\bar{x})(x_n-\bar{x})^T r_2+\lambda_4 r_1+\lambda_2 r_2+\lambda_6 r_3=\sum_{n=1}^{N}\gamma_n(y_{n2}-\bar{y}_2)(x_n-\bar{x})$$

$$\sum_{n=1}^{N}\gamma_n(x_n-\bar{x})(x_n-\bar{x})^T r_3+\lambda_5 r_1+\lambda_6 r_2+\lambda_3 r_3=\sum_{n=1}^{N}\gamma_n(y_{n3}-\bar{y}_3)(x_n-\bar{x})$$

(AB.5)

Let

$$A=\sum_{n=1}^{N}(x_n-\bar{x})(x_n-\bar{x})^T,\quad \Lambda=\begin{bmatrix}\lambda_1 & \lambda_4 & \lambda_5\\ \lambda_4 & \lambda_2 & \lambda_6\\ \lambda_5 & \lambda_6 & \lambda_3\end{bmatrix},\quad B=\begin{bmatrix}b_1 & b_2 & b_3\end{bmatrix} \text{ where } b_k=\sum_{n=1}^{N}\gamma_n(y_{nk}-\bar{y}_k)(x_n-\bar{x})$$

Then Eq.(AB.5) can be simply rewritten as

$$AR^T+R^T\Lambda=B$$

Multiplying both sides of this equation on the left by R we have

$$RAR^T+\Lambda=RB$$

Since $A=A^T$, $(RAR^T)^T=RAR^T$, Since both RAR^T and Λ are symmetric, the left hand side must be symmetric. Hence, the right hand side is also symmetric. This means,

$$RB=(RB)^T$$

The solution for R comes quickly. Let the singular value decomposition of B be

$$B=UDV$$

where U and V are orthonormal and D is diagonal. Then

$$RUDV=(UDV)^T R^T=V^T DU^T R^T$$

By observation, a solution for R is immediately obtained as

$$R=V^T U^T$$

This solution is given in [Aru87], [Fre89].

Publications arising from this research

The following papers have been published as a result of this research:

Naser Prljaca and Hugh McCabe, "3D Geometrical Model Building for Visual Robot Control ", *In SPIE Proc.*, Vol.2056, pp.296-306, Boston, September 1993

Naser Prljaca and Hugh McCabe, "3D Structure Estimation for Vision Based Robotics", *In SPIE Proc.*, Vol.2354, pp.66-78, Boston, Novembar 1994

The following papers have been submitted for publications:

Naser Prljaca and Hugh McCabe, "Novel Approach to Robot Visual Control"

Naser Prljaca and Hugh McCabe, "Statistical approach to camera calibration"