# 3D Object Recognition and Retrieval Using Navigating Robots with Active Vision

**"MROLR"**
**A Mobile Robotic Object Locator and Retriever**

A thesis submitted for the degree of

Doctor of Philosophy


**Michael Wingate**

August 2003

School of Electrical Engineering

Faculty of Science, Engineering and Technology

Victoria University

P.O. Box 14428, MCMC, Melbourne

Victoria 8001, Australia

# Declaration

This thesis contains the work of my doctoral studies under the supervision of Assoc. Prof. Mike Sek (Victoria University of Technology), Assoc. Prof. Patrick Leung (Victoria University of Technology), and Dr Hao Shi (Swinburne University of Technology). The contents of this thesis and related work have been referenced in the following papers that have been published at conferences, journals or are under review for publication.

Publications

1. M. Wingate and A. J. Davison, "Object Retrieval using an Active Stereo Head and a Robot Arm," *4th Int. Conf. on Modelling and Simulation,* pp. 417-422, Australia, Nov 2002.

2. M. Wingate, "3D Modeling for the Recognition of Remote Objects," *4th Int. Conf. on Modelling and Simulation,* pp. 405-410, Australia, Nov 2002.

3. M. Zulli and M. Wingate, "A Micro Mobile Surveillance System with Wireless Video", *4th Int. Conf. on Modelling and Simulation,* pp. 452-457, Australia, Nov 2002.

4. N. Burley and M. Wingate, "An Active Camera Target Surveillance Tracker," *4th Int. Conf. on Modelling and Simulation,* pp. 440-445, Australia, Nov 2002.

5. H. Moyssidis and M. Wingate, "A Speech Guided Robot Controller with 3D Graphical Positioning," *4th Int. Conf. on Modelling and Simulation,* pp. 428-433, Australia, Nov 2002.

6. *M.* Wingate, " "MROLR" A Mobile Robotic Object Locator and Retriever," *Post Graduate Seminar Presentation,* School of Built Environment, Victoria University, Sept 2002.

7. M. Wingate and A. J. Davison, "3D Recognition of Remote Objects Utlizing Stereo Vision on a Roving Platform," *ACCV2002, Proc. 5th Asian Conf. on Computer Vision*, pp. 682 - 688, Australia, January 2002.

8. *M.* Wingate, "Structure Recovery of Objects Using Multiple Camera Views", *Post Graduate Seminar Presentation,* School of Electrical Engineering, Victoria University, 1999.

9. *M.* Wingate, "3D Structure Recovery of Rigid Objects Using Pan-Tilt-Vergence Stereo" *Proc. Joint DICTA'97 & IVCNCZ'97 Conf.* Auckland, N.Z. pp. 83-88, Dec 1997.

*10.* C. Jones, and M. Wingate, "Robotic Harvesting in a Simulated Environment" *3rd Int. Conf. on Modelling and Simulation*, Melbourne, pp. 363 - 367, Oct. 1997.

*11.* M. Wingate, "3D Structure Recovery of Objects Using Line Features in Multiple Camera Views" 3rd *Int. Conf. on Modelling and Simulation,* Melbourne, pp. 386 - 391, Oct. 1997.

12. M. Wingate, and A. Stoica, "An Application of Fuzzy Neurons to Visual Learning", *3rd Int. Conf. on Modelling and Simulation,* Melbourne, Oct., pp. 45- 50, 1997.

13. M. Wingate, and A. Stoica, "A Step Towards Robot Apprentices - Visual Learning", *4th IASTED Conf. Robobtics & Manufacturing*, Hawaii. pp.120 - 123 Aug. 1996.

14. P.T. Im, W.S. Lee, and M. Wingate, "A progressive fuzzy clustering approach for the detection of linear boundary and corners", *Australian Journal of Intelligent Information Processing Systems*, Vol.3, No3, pp. 42-58, Australia, 1996.

15. P.T. Im, M. Wingate, and B. Qiu, "A cluster prototype centring by membership algorithm for fuzzy clustering", *2nd Asian Conference on Computer Vision*, Singapore, vol. 2, pp. 67-70, 1995.

16. A.Y.S Chong and M.Wingate, "A Colour Feature Based Stereo Vision System for Robot Assembly", *ICA/ACME Int. Conf. on Assembly combined with the Australian Conf. on Manufacturing Engineering,* Nov. 1993.

## Statement of Originality

This thesis contains no material that has been accepted for the award of any other degree or diploma in any university and that, to the best of the author's knowledge and belief, the thesis contains no material previously published or written by other persons, except where reference is made in the text of this thesis.

---

Michael WINGATE

School of Electrical Engineering,

Faculty of Science, Engineering and Technology,

Victoria University of Technology

P.O. Box 14428, MCMC

Melbourne, Victoria 8001

AUSTRALIA

# Acknowledgements

I am indebted to each of my supervisors for their guidance, suggestions, feedback and wise counselling. Since commencing this research the following members of staff have taken on the burden of my supervision. Dr. Mike Sek, Dr. Patrick Leung, Dr. Hao Shi, and Dr. Len Herron. The latter two unfortunately having left Victoria University prior to my thesis submission.

In 1999 I had the good fortune of spending three months at the University of Sheffield (U.K). There my work gained major impetus under the direction of Dr. Peter Rockett, and Dr. John Porril, and PG Students Simon Crossley, Bala Amavasai and Fabio Caparrelli (each having gained a Ph.D.). In particular I wish to thank them for their valuable discussions and for providing me with access to, and help with, the University's hardware and software resources. It is at this University that "TINA" was born and is now being continued at the University of Manchester by Dr. Neil Thacker and others (formally of the University of Sheffield). I am also indebted to Dr. Neil Thacker for his valuable assistance in debugging library modules of TINA. The TINA environment is a significant software component of the work in this thesis.

Dr. Andrew Davison (of the Robotics and Research Group, University of Oxford, UK) is another eminent scholar I am deeply indebted to. He has generously made his software (SCENE) available to all, software that is also a significant component of the work in this thesis. Andrew guided the adoption and modification to "SCENE" for my navigation needs. He has also co-authored several papers relating to this research.

I adopted code from Gurvinder Pal Singh's "Virtual Robotics Lab" to implement a "3D simulation environment for MROLR. Gurvinder has generously made his software available.

I owe thanks to academic staff within my discipline area, namely Dr. Wee Sit Lee and Dr. Fu Chun Zheng, whom I frequently turned to for assistance in obtaining analytical solutions to difficult problems.

I owe special gratitude to a large number of the Technical Officers within the School of Electrical Engineering. In particular to Foster Hayward for providing excellent technical advice and contribution in establishing the local area network required for MROLR. Both Foster and Taky Chan also played a key role in the construction of vital hardware and in keeping it functioning. Hahn Vuong is owed gratitude for maintaining the various Linux, Windows and DOS platform software. Ralph Phillips (now retired) also gave significant valuable technical assistance and advice.

Several PG students contributed either directly or indirectly to this work in the form of the knowledge gained during my role as supervisor and joint author of papers published. They were Adrian Stoica and Paul Im (both having now completed their Ph.D. before their supervisor), and Chris Jones.

Dr. Greg Martin bravely volunteered to proof read this thesis, I am truly grateful to him for this and his useful suggestions.

Finally I wish to thank the Senior Administrators of the Faculty of Science, Engineering and Technology, the School of Electrical Engineering, and former Department of Electrical and Electronic Engineering of this University for allowing me the time and facilities to pursue this research.

# Abstract

*This work describes the development of a new type of "service robot" called "MROLR" capable of searching for, locating (pose determining), recognizing, and retrieving 3D objects in an indoor environment. Key related contributions include the development of 3D object-model creation and editing facilities, and the application of a vision enhanced navigation algorithm.*

*MROLR comprises a mobile robot transporting a four degrees of freedom (DOF) stereo head, equipped with two calibrated charge-coupled device (CCD) video cameras, and a docking robot arm. The principal navigation mode utilises Simultaneous Localisation and Map-building (SLAM) to update and maintain a map of the robot's location and multiple navigational-feature positions as it steers towards specified coordinates. The algorithm is Kalman Filter based. Navigation-features comprise either of known (previously stored) or newly acquired, visible landmarks. During navigation, automatic navigation-feature selection and measurement is performed and the results of measurements used to substantiate or correct odometry readings.*

*Objects to be retrieved from a scene have an associated (previously created and stored) object-model. Object-models are created by the integration of a sequence of 3D models constructed from stereo image pairs, each representative of the object in varying angular positions. The model of an object sought is retrieved from a database and matched against scene-models until recognition and pose is established. Scene-models are formed during the search while panning the scene. Both object-models and scene-models comprise 3D straight line and conic segments. Recognition is based on verifying the existence of mutual groups of 3D line and or conic edge features in both the scene and model object. Where the object has sufficient distinctive features, recognition is view independent and tolerant to both scale variations and occlusions. On finding the object, a six DOF robot arm, attached to a caster platform, is manually docked with the mobile robot. Using the object's pose transform the arm is able to grasp and place the object on the mobile base. The arm is manually de-coupled from the mobile robot and the object transported back to the home position.*

*While it would be possible to mount the head and arm on one mobile base, the intention here is to ultimately have a light weight fast moving "scout robot" to seek and find using vision, and a slower heavy-duty transport robot to lift and carry.*

*Finally a "3D Virtual Environment" for simulating MROLR, that could be useful for evaluating alternative map-building strategies, object grasping points, as well as for demonstration and educational purposes, has been implemented, although is not yet complete.*

Key Words: 3D modeling, recognition, stereo vision, mobile robot navigation.

# Contents

# Table of Figures

# List of Notation

Note the symbols listed here are in a logical rather than an alphabetical sequence

## Camera, Head and Associated Symbols

$\nabla$ (del) operator examples $\nabla(y)_x = \frac{\partial y}{\partial x}$, $\nabla(\mathbf{f_v})_u = \begin{bmatrix} \nabla z(t+\Delta t)_{v_1} & \nabla z(t+\Delta t)_{v_2} \\ \nabla x(t+\Delta t)_{v_1} & \nabla x(t+\Delta t)_{v_2} \\ \nabla \varphi(t+\Delta t)_{v_1} & \nabla \varphi(t+\Delta t)_{v_2} \end{bmatrix}$

| | |
|---|---|
| $\mathbf{P}$ | 3D vector point comprising coordinate components $P_x, P_y, P_z$ |
| $\mathbf{P}^W$ | 3D vector point in frame W (world ref. frame) |
| $\mathbf{p}^F$ | 3D image vector point in camera frame F ( coordinates $(x_c, y_c, f)^t$ ) |
| $\mathbf{m}^F = \mathbf{P}^F$ | 3D vector point in camera frame F |
| u,v | 2D image coordinates in pixels |
| $u_o, v_o$ | 2D coordinates of image centre in pixels |
| $(x_c, y_c, f)$ | 3D coordinates of image point in camera frame |
| f | camera effective focal length |
| $k_u$ , $k_v$ | number of horizontal and vertical pixels/m respectively |
| $\mathbf{C}$ | camera matrix |
| $\mathbf{C}^{-1}$ | inverse camera of matrix |
| $\alpha$ | head pan angle |
| $\gamma_L$ , $\gamma_R$ | head left & right vergence angle |
| e | head tilt angle |
| $C_0$, C0 | head coordinate frame |
| L, R | left & right frames respectively, also used for camera & head vergence frames |
| $C_2$ | head tilt frame |

| | |
|---|---|
| $C_1$ | head pan frame |
| $\mathbf{m}_R$ , $\mathbf{m}_L$ | 3D scene point in right and left camera frames respectively |
| $\mathbf{m}_G^{CO}$ , $\mathbf{m}_G$ | 3D scene point in head frame |
| $\mathbf{c}_R$, $\mathbf{c}_L$ | offset vector along respective vergence axis, between the intersections with the tilt axis and camera optic axis |
| $\mathbf{n}_R$ $\mathbf{n}_L$ | offset vector along respective optic axis between camera optic centre and intersection with vergence axis |
| $\mathbf{p}_L$ $\mathbf{p}_R$ | horizontal offset vector (along tilt axis) between pan and tilt axes intersection, to respective vergence axis |
| $\mathbf{M}_h$ | vertical offset vector (along pan axis) between mobile base centre and head frame origin |
| I | horizontal scalar distance between camera optic centres (i.e., inter-ocular distance) |
| p | scalar horizontal offset distance (along tilt axis) between pan and tilt axes intersection, to respective vergence axis |
| c | scalar offset distance along respective vergence axis, between the intersections with the tilt axis and camera optic axis |
| n | scalar offset distance along respective optic axis between camera optic centre and intersection with vergence axis |
| $M_h$ | scalar vertical offset distance (along pan axis) between mobile base centre and head frame origin |
| $\beta$ | angle difference between $\mathbf{m}$ and $\mathbf{m}_{orig}$ |
| $\mathbf{R}^{C10}$ | transform rotation matrix between frames C1 and C0 |
| $\mathbf{R}^{C21}$ | transform rotation matrix between frames C2 and C1 |
| $\mathbf{R}^{LC2}$ | transform rotation matrix between frames L and C2 |
| $\mathbf{R}^{RC2}$ | transform rotation matrix between frames R and C2 |
| $\mathbf{R}^{C2R}$ | inverse ($\mathbf{R}^{RC2}$) |
| $\mathbf{r}$,$\mathbf{r}_1$,$\mathbf{r}_2$ | ray vector ($\mathbf{r}_1 = \mathbf{a}+\lambda\mathbf{b}$, $\mathbf{r}_2 = \mathbf{c}+\lambda\mathbf{d}$ ) |

| | |
|---|---|
| **a,c** | vector point on ray |
| **b,d** | vectors in direction of ray |
| $\lambda,\mu$ | scale factor |
| $\hat{\mathbf{u}}$ | unit vector |
| $\{\mathbf{u}\}$ | column vector |
| $[\tilde{\mathbf{u}}]$ | skew-symmetric matrix of $\hat{\mathbf{u}}$ |

**Mobil Base, Landmark Features and Associated Symbols**

| | |
|---|---|
| v | centre vehicle velocity |
| R | arc radius of movement |
| $v_1$ , $v_2$ | linear driven wheel velocities |
| $\theta$ | angular rotation (arc angle) |
| D, D1, $D_2$ | axial wheel offset distances from centre of base |
| x, x(t) | mobile base x coordinate in world reference frame |
| z, z(t) | mobile base z coordinate in world reference frame |
| $\phi,\phi(t)$ | mobile base orientation relative to world reference frame (x axis) |
| $\mathbf{x}_v$ | $\mathbf{x}_v = \begin{pmatrix} z(t) \\ x(t) \\ \phi(t) \end{pmatrix}$ |
| $z(t+\Delta t)$ | robot's estimated new z coordinate position following a constant time step increment $\Delta t$ |
| **Z** operator | Shi and Tomasi operator, $\mathbf{Z} = \sum_{patch} \begin{bmatrix} g_x^2 & g_x g_y \\ g_y g_x & g_y^2 \end{bmatrix}$ |
| $g_x$ , $g_y$ | horizontal, vertical gradients of image pixel intensities |
| $\lambda_n$ | $n^{th}$ eigenvalue |

| | |
|---|---|
| N | normalised sum-of-squared-difference measure. Used for feature matching. $N = \frac{1}{n}\left( \sum_{patch} \left[ \frac{g_1 - \bar{g}_1}{\sigma_1} - \frac{g_0 - \bar{g}_0}{\sigma_0} \right]^2 \right)$. N is zero for a perfect match. |
| $\bar{g}_0, \bar{g}_1, \sigma_0$ $\sigma_1$,n | means and standard deviations of the intensity across the patches 0 and 1 respectively. n is the number of pixels in a patch |
| $\varepsilon$ | angular fixation error |
| $\mathbf{T}_{Object}^{C_0}$ | standard homogeneous 4x4 transform relating the rotation and translation of an object frame wrt $C_0$ (the head) frame |
| **[n o a p]** | the column vector elements of the homogeneous transform **T** = **[n o a p]**: **n** is the normal vector, **o** is the orientation vector a the approach vector **p** the translation vector. |

## Extended Kalman Filter and Associated Symbols

| | |
|---|---|
| **f** | state transition function: The state transition function is a function of $\hat{\mathbf{x}}$ and **u** |
| $\mathbf{f}_v$ | vehicle estimated position vector ( function of $\mathbf{x}_v$ and **u** ) |
| $\Delta t$ | step time interval |
| **z** | measurement vector, **z** is used to designate an actual measurement (i.e., using the active stereo head ) |
| $\mathbf{m}_{Gi}^{C0}$, | $\mathbf{m}_{Gi}$ measurement vector of a scene-feature in head frame |
| $m_{Gi}$ | scalar length of vector $\mathbf{m}_{Gi}$ |
| $\mathbf{m}_i$ | measurement vector of $i^{th}$ scene-feature in world frame, transformed into angular coordinates |
| $\mathbf{m}_i(\mathbf{x}_v,\mathbf{y}_i)$ | emphasizing measurement vector $\mathbf{m}_i$ is a function of $\mathbf{x}_v$ and $\mathbf{y}_i$ |
| $m_{Gi\rho}$ | projection of $\mathbf{m}_{Gi}$ onto xz plane $m_{Gi\rho} = \sqrt{(m_{Gix}^2 + m_{Giz}^2}$ |
| **m**, **m**(**x**) | prediction of the measurement **z**, **m**(**x**) = $\mathbf{m}_i(\mathbf{x}_v,\mathbf{y}_i)$ |
| **x** | system state vector |

| | |
|---|---|
| $\hat{\mathbf{x}}$ | estimate of system state vector |
| $\hat{\mathbf{x}}(k+1/k)$ | current estimation of system state vector (estimate of **x** at a time step k + 1 based on an estimate at time, step k and an observation (measurement) made at time step k + 1. At any given time, estimates of mobile-robot and scene-feature locations (in the world reference coordinate frame) are stored in the system state vector $\hat{\mathbf{x}}$ |
| $\mathbf{P}$ | 3(n+1)x3(n+1) covariance symmetric matrix, representing the uncertainty in $\hat{\mathbf{x}}$. n = number of known features |
| $\mathbf{P}_{xx}$ | 3x3 covariance matrix of the estimated mobile robot state $\hat{\mathbf{x}}_v$ and itself. This matrix is a partition of **P** |
| $\mathbf{P}_{xy_i}$ | 3x3 covariance matrix between the estimated mobile robot state $\hat{\mathbf{x}}_v$ and feature $\hat{\mathbf{y}}_i$. This matrix is a partition of **P** |
| $\mathbf{P}_{y_iy_j}$ | 3x3 covariance matrix between the estimated feature state $\hat{\mathbf{y}}_i$ and feature $\hat{\mathbf{y}}_j$. This matrix is a partition of **P** |
| $\mathbf{P}_{y_iy_i}$ | 3x3 covariance matrix between the estimated feature state $\hat{\mathbf{y}}_i$ and itself. This matrix is a partition of **P** |
| $\mathbf{Q}$ | noise, provides a means of allowing for random or unaccounted effects in the dynamic model. |
| $\boldsymbol{\nu}$ | innovation, $\boldsymbol{\nu}$ is the difference between the actual measurement **z** and the predicted measurement **m** |
| $\mathbf{R}$ | covariance matrix of the noise This matrix represents the covariance of the noise in the measurement |
| R | scalar noise variance measurement ($\Delta\alpha^2$, $\Delta e^2$ or $\Delta\gamma^2$) |
| $\mathbf{R}_L$ | covariance matrix of the noise transformed into cartesian measurement space |
| $\mathbf{S}$ | innovation covariance, represents the uncertainty in a measurement (i.e., the amount by which an actual measurement differs from its predicted value). |
| $\mathbf{S}_{mGi}$ | innovation covariance of measurement associated with $\mathbf{m}_{Gi}$ |

| | |
|---|---|
| $\mathbf{W}$ | Kalman gain |
| $\mathbf{x}_v$ | mobile-robots position |
| $\hat{\mathbf{x}}_v$ | mobile-robots position estimate |
| $\mathbf{y}_i$ | 3D location of the *i*th feature |
| $\hat{\mathbf{y}}_i$ | estimated 3D location of the $i^{th}$ feature |
| $\mathbf{Q}$ | covariance matrix , expressing the uncertainty in $\mathbf{f}_v$ |
| $\mathbf{U}$ | covariance matrix of control vector **u,** and $\sigma_{v_1}$, $\sigma_{v_2}$ are the |
| $\sigma_{v_1}$, $\sigma_{v_2}$ | standard deviation estimates of the errors in the velocity control inputs $v_1$ and $v_2$ |
| $\mathbf{U}_L$ | covariance matrix of image vector $\mathbf{u}_L = \begin{pmatrix} u_L \\ v_L \end{pmatrix}$ |
| Vs | scalar space volume measure |

## Object Modelling, Recognition and Associated Symbols

| | |
|---|---|
| $(\mathbf{l}_i, \mathbf{l}_{i+1}, \mathbf{e}_i, \mathbf{m}_i)$ | quadruple representing 3D line segment i, that is the two end points $\mathbf{l}_i$ and $\mathbf{l}_{i+1}$, the direction vector between them $\mathbf{e}_i$ ,and centroid of the line, its midpoint $\mathbf{m}_i$ |
| $\alpha$ | orientation difference between two line segments |
| $\mathbf{h}$ | vector distance between two line segments |
| h | scalar distance between two line segments $\mathbf{h} = h\mathbf{d}$ |
| $\mathbf{d}$ | the unit vector normal between two line segments |
| $s_i$ | scalar distance from the point of minimum distance between two line segments (i, i+1), to the start of the segment i, measured parallel to segment i. |
| $t_i$ | scalar distance from the point of minimum distance between two line segments (i, i+1), to the end of the segment i, measured parallel to segment i. |

| | |
|---|---|
| $q_i$ | scalar distance from the point of minimum distance between two line segments (i, i+1), to the centre of the segment i, measured parallel to segment i. |
| $\beta_i$ | permissible scalar error values |
| $\mathbf{m}_{i+1}'$ | $\mathbf{m}_{i+1}$ vector modified by the addition of $-\mathbf{h}$ to make it coplanar with $\mathbf{m}_i$ |
| $\theta_i$ | permissible angular orientation error values |

# Chapter 1 About This Thesis

## 1.0 Introduction

The work presented in this thesis comprises the development and analysis of a mobile robotic system capable of visually locating specific 3D objects in an indoor environment and transporting them to a given location. Its classification would fit into the category of "service robots" and it has been named "MROLR", A **M**obile **R**obotic **O**bject **L**ocator and **R**etriever.

## 1.1 Thesis Objectives

The main objectives of this thesis are to:

- gain a theoretical understanding of the problems relating to visually locating, identifying, and physically retrieving specified 3D objects in a known indoor environment, utilising mobile robotic platforms
- develop feasible solutions for the implementation of a working system
- evaluate implemented solutions

Consider the requirements of a robotic system that can be given instructions to fetch a specified object in the current environment, at present confined to be indoors. If the object is visible this is of course a relatively simple task for a human, but still a difficult problem for a robot. Practically the problems involved can be identified as:

- describing the target object to the robot
- autonomously navigating in the environment (including avoiding obstacles)
- visually being able to recognise the object
- determining the object's pose (that is its position and orientation)
- grasping the object
- remaining both within financial and time constraints

The robotics/machine vision discipline is now relatively mature and a significant number of authors have developed systems that can perform several of the above tasks, i.e. autonomous navigation, object recognition and pose determination, and visually guided object grasping by a robot arm. Many of the authors and systems are reviewed or mentioned in the following chapters.

Putting together the desired system by building on the work of others, on the surface at least, appears quite straightforward, merely a matter of integrating and supplementing successfully proven concepts. Choosing from available learned publications nevertheless poses dilemmas. Authors invariably claim improvements in one or more aspects of their work over previous attempts by others, claims that often are difficult to substantiate. Furthermore, interpreting their work, implementation, and finally reaching a satisfactory degree of operation, can prove difficult. By and large the above steps constitute a substantial part of this work.

Acknowledgement of the contributions made by others to this work is given in the various chapters where they occur.

The incentive and motivation for developing "MROLR" and the contributions made by this work together with an outline of the thesis structure is given in sections 1.3,1.4 and 1.5 respectively. A brief survey of service robots is given below.

## 1.2 Service Robots

Historically robots were designed predominantly for use in factories for purposes such as manufacturing and transportation, but advances in technology have widened their applications, enabling them to automate many tasks in non-manufacturing sectors such as agriculture, construction, health care, retailing and other services. Broadly these systems, including the one being developed in this work, may be classified as "service robots", and fall into categories such as:

Cleaning, Lawn mowing, and Housekeeping Robots
Entertaining Robots
Humanoid and Apprentice Robots
Rehabilitation Robots
Humanitarian Mine Disarming Robots
Medical Robots
Agricultural and Harvesting Robots
Sheep Shearing Robots
Surveillance Robots
Inspection Robots
Construction Robots
Automatic Refilling Robots
Fire Fighting and Rescue Robots
Robots in Food Industry
Tour Guides and Office Robots
Flying Robots
Space Exploration and Terrain Mapping Robots

The list of service robots, with ever improving abilities, is growing at an impressive rate.

## 1.3 Incentive and Motivation for Developing "MROLR"

As implied above, the catalogue of service robots is growing rapidly and it is anticipated that in the near future autonomous systems, providing safe, reliable navigation, search, and object recognition and retrieval abilities will join the list. The development of such a system was an incentive for commencing this work and has led to the creation of MROLR. MROLR is a navigating mobile robotic system capable of searching for, locating (pose determining), recognising, and retrieving 3D objects in an indoor environment. The system comprises a mobile robot transporting a 4 DOF stereo head, equipped with 2 calibrated charge couple device (CCD)

video cameras, and a docking robot arm. The principal navigation mode incorporates Simultaneous Localisation and Map-building (SLAM) that uses an Extended Kalman Filter (EKF) to update and maintain a map of the robot's location and multiple navigational-feature positions as it steers towards specified coordinates. Navigation-features comprise either of known or newly acquired visible features. Known features are a priori stored while newly obtained features are acquired while tracking. During navigation, automatic navigation-feature selection and measurement is performed. The results of these measurements in combination with odometry readings are used to obtain good robot and feature position estimates. Limited obstacle detection and circumnavigation (avoidance) is incorporated.

Surveying of the scene for objects to be recognised occurs at specified locations termed waypoint stations. At waypoints associated with object locations (i.e., tables laden with objects), the head's gaze is lowered and it sweeps through a pan of –20 to +20 degrees in search of the desired object. At present (while panning for scene-objects) the tilt and vergence angles remain fixed during this sweep. This ensures the cameras' motion (and consequently the structure-from-motion) is consistent with their calibration. It is proposed to vary the tilt and vergence angles in a sequence of movements in the future for a variety of different calibrated camera motions.

The recognition system utilises a database of object-models from which sought objects are selected and subsequently matched against scene-models. Database stored object-models are individually constructed at a prior time. Object-model creation comprises the integration of a sequence of 3D models constructed from stereo image pairs, each representative of the object in varying angular positions. The object-model of an object of interest to be located in a scene is retrieved from a database and matched against scene-models until recognition and pose is established. Scene-models, unlike object-models are formed during searching, while panning the scene. Both object-models and scene-models comprise 3D straight line and conic segments. Recognition is based on verifying the existence of mutual groups of 3D line and or conic edge features in both the scene and model object. Where the object has sufficient distinctive features, recognition is view independent and tolerant to both scale variations and occlusions. On finding the object, a 6 DOF robot arm attached to a caster platform is manually docked with the mobile robot. Using the object's pose transform the arm is able to grasp and place the object on the mobile base. The arm is manually de-coupled from the mobile robot and the object transported back to the home position. In a future version of the MROLR, the arm will be mounted on a mobile transport robot, that will be signalled via a wireless modem, and on receiving this command navigate to and dock with the active vision base. The construction of this "transport robot" (at the time of this writing) is near completion (Figs 6.7.2). While it would be possible to mount the head and arm on one mobile base, the intention here is to ultimately have a light weight fast moving "scout robot", and a slower heavy-duty "transport robot". The scout would map and model the environment, and seek and find target objects using vision. When a target object was located, then its positional information and path directions would be relayed to the transport robot which would proceed with the task of object retrieval and transport.

Motivation for this work also stems from the desire to ultimately develop a system capable of rapid and reliable retrieval of objects in both domestic and industrial environments. It is believed

that potential applications for such an improved system range from simple domestic and industrial "robotic aids", to "assistants" for the severely physically disabled or visually impaired.

A prolific collection of literature exists on each of the fields of 3D object recognition, pose determination, mobile robot navigation, and robot arm kinematics relating to object grasping. Each of these comprises significant areas of research and development for the machine vision and robotics fraternity. The merging of both related and unrelated scientific endeavors provide a wealth of design building blocks and opportunities for creative mechatronic engineering applications. The emergence of service robots is such an example.

## *1.4 Contributions of this thesis:*

The major contribution made by this thesis is in the design and implementation of "MROLR", a new type of "service robot", comprising a navigating mobile robotic platform system with active vision and retrieval abilities.

Related contributions stem from:

- Development of a 3D object-model creating facility (design and construction of a motor driven turntable, and related software).
- Development of a model editor to facilitate the manual removal and/or addition of 3D segments in formed models.
- Extension and modification of existing recognition algorithms to facilitate automated recognition while searching and scanning the scene.
- Extension and modification of existing active vision based navigation software to work with the designed mobile base. This included the writing of an obstacle detection and avoidance module.
- Development of kinematic solutions from object pose information, to enable the robot arm to grasp and retrieve the desired object.
- Preliminary design and construction of a "transport" robot base with mounted robot arm, and related software.
- Preliminary implementation of a "3D Virtual Environment" for simulating MROLR, that could be useful for evaluating alternative map-building strategies, object grasping points, as well as for demonstration and educational purposes.

As is common with many engineering projects, a substantial amount of ancillary software and electronic hardware is required and inevitably must be tailor designed and built to meet special requirements. An example is the software to enable handshaking and the exchange of both video and numerical data between the various interconnected PC platforms, and hardware devices, comprising MROLR. These include the mobile base, active 4 DOF head, robot-arm, and frame-grabber, and facilitated by the establishment of a local area network. Hardware examples include electronic boards required to drive rotating or moving items such as the computer controlled model creating facility, and "transport" mobile base.

Much of the work carried out in this project was greatly assisted by the generous co-operation and advice given by colleagues acknowledged on page v, and throughout this thesis.

## 1.5 The Structure of this Thesis

Chapter 2 commences with a review of historical developments and work related to this research. It is primarily a synoptic review. The areas covered include service robots, 3D object-recognition, pose determination, and mobile robots with metric maps.

Chapter 3 describes the system equipment comprising MROLR. Both hardware and software requirements are examined. Specific details such as hardware circuits designed and built, and software driver modules written, are given in Appendix B.

Chapter 4 provides detailed background information on mathematical notation and concepts used throughout this thesis. A variety of models that form the preliminary analysis to following chapters are introduced. Models included are for the camera, stereo head, and mobile base. Algorithms and tools used for object and scene-modeling are also introduced.

Chapter 5 develops algorithms for autonomous navigation from an initial coordinate frame. Navigational features facilitated through active stereo vision and odometry include simultaneous map building and localisation, and limited obstacle avoidance. The algorithms are based on and build upon those of Davison [Dav98], [Dav01] and use an Extended Kalman Filter with maintenance of full covariance knowledge between sets of map-features.

Chapter 6 comprises of four sections, the first outlines the 3D object-model producing facility developed to provide an object-model data base of objects that MROLR could be requested to locate and retrieve in the future. These objects would be searched for at designated stopover points (waypoints) along navigational routes. The facility also includes a 3D object-model editor.

The second section outlines the development of the object recognition module that is responsible for (a) matching requested target objects with objects in a scene. (b) Obtaining appropriate transforms to graphically transport the object-model into the scene, after successful matches.

The third section establishes the transforms necessary to guide the robot arm to grasp and retrieve the located object. Grasping locations are contained within the data provided for each created object-model (i.e., handle of a cup, etc.).

The Final section outlines the implementation of a "3D Virtual Environment" for simulating MROLR.

Chapter 7
Specific tests performed and results obtained, are described in this chapter. It is considered that these verify the methodology of this research. Tests include navigating to a variety of specified waypoints and recognising and retrieving requested target objects. During navigation, self-localisation map building and acquisition and storing of navigational features (to become "known-feature" aids for future use) are also performed.

Chapter 8, this final chapter concludes by summarizing the main contributions of this work and outlines envisaged future directions and extensions.

Appendix A describes some early approaches and methods evaluated, but ultimately discarded by the author in the process of obtaining an effective recognition module for MROLR.

Appendix B
A substantial amount of ancillary software and electronic hardware was designed and developed for MROLR. Appendix B contains additional specific details relating to these designs. This is included for clarification and extension of information provided in various chapters.

Appendix C continues with the algorithmic development for continuous tracking of multiple features using active vision commenced in Chapter 5. It concludes with the "$V_s$" criterion that provides direction as to which navigational landmark feature is optimal to next track.

Finally a comprehensive Bibliography is provided. This is a collection of references maintained and used by the author.

## *1.6 Conclusion*

This introductory chapter has detailed the objectives, incentive and motivation, and contributions of this work. It concludes with a summary description of the content of each chapter that follows.

# Chapter 2 Historical Developments and Related Work - A Synoptic Review

## 2.0 Introduction

Modern robot designs and applications extend well beyond the requirements of the manufacturing sector. Service robots are being utilised as aids and for tasks hither too unimaginable by most. MROLR is classified as a "service robot", its object recognition and pose determining modules are "model based", while navigation, a combination of landmark mapping (for route planning and localisation) and odometry based. Significant progress and advances in these fields have taken place over the past three decades, spurred undoubtedly by the quest for machines with greater autonomy. A vast quantity of literature relating to these developments has also emerged, with much of it now readily accessible via the Internet. A synoptic review of associated historical developments and related work is given in the following sections of this chapter.

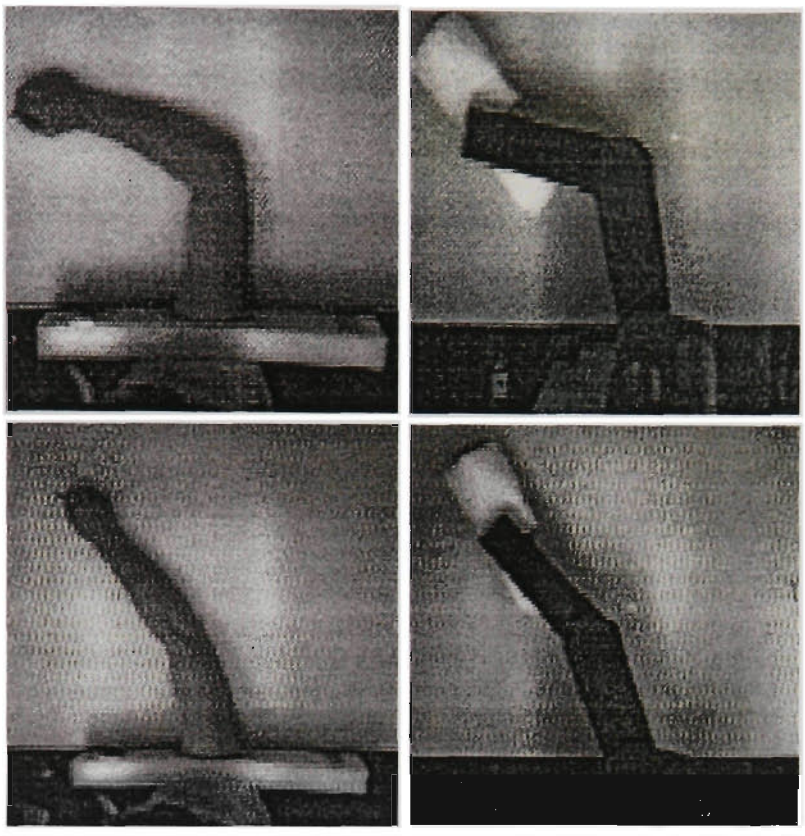
Section reviews are organised as follows:

- **2.1** Service robots and applications
- **2.2** 3D object recognition and pose determination
- **2.3** Robots utilising metric maps (several of these are service robots)
- **2.4** Mobile robots using vision sensors to aid navigation

## 2.1 Service Robots

### 2.1.1 Apprentice Robots

Moderate success has been achieved in getting robot apprentices to learn 3D motions and tasks by visual observations (see for example Bentivegna et al [BA02], Atkeson and Schaal [AS97]). Wingate and Stoica [WS96], [WS97] developed a fuzzy-neural robot arm controller, based on the composition of triangular norms and co-norms (S-T norms). The system utilised 2 cameras to enable the robot to view (a) the motion of the Master's arm (human arm, or robot teacher's arm), and (b) the motion of its own arm (Apprentice's arm). Two learning modes were developed. In the first the robot apprentice looked at the Master's arm and its own alternately, and attempted on each viewing to minimise position error variables. In the second mode, only the Master's arm is watched. The robot being free to issue arm joint commands more or less at random, such that its arm took up a variety of postures. For each position the Master attempts to place his arm in a corresponding posture to that of the robot. When arm positions are sufficiently similar, the robot is advised, and a process of validation takes place. Association between images of the Master's arm, and joint motor commands that the apprentice robot gives to position its own arm (to arrive at similar postures) are learnt, and subsequently used as part of the apprentice robot's command set.



Apprentice robot learning (a) from human Master's arm and (b) A Master (pre-programmed) robot.

**Fig 2.1.1. 1 Apprentice robot**

As discussed in Chapter 8, under "Future Directions and Work", it is intended to revisit this work with a view to adding gesture recognition capabilities to MROLR.

### 2.1.2 Template-Based Recognition of Pose and Motion Gestures On a Mobile Robot

Waldherr et al [WTRM98] produced a vision-based interface that has been designed to instruct a mobile robot through both pose and motion gestures. An adaptive dual-colour algorithm enables the robot to track and, if required, follow a person around at speeds of up to one foot per second while avoiding collisions with obstacles. This tracking algorithm quickly adapts to different lighting conditions. Gestures are recognised by a real-time template-matching algorithm. This algorithm works in two phases, one that recognises static arm poses, and another that recognises gestures (pose and motion). In the first phase, the algorithm computes the correlation of the image with a pre-recorded set of example poses (called: *pose templates*). In the second phase, the results of the first phase are temporally matched to previously recorded examples of gestures (called: *gesture templates*), using the Viterbi algorithm Rabiner & Juang [RJ86]. The gesture template matcher can recognise both pose and motion gestures. The result is a stream of probability distribution over the set of all gestures, which is then thresholded and passed on to the robot's high-level controller. This approach has been integrated into their existing robot navigation and control software, where it enables human operators to:

- provide direct motion commands (e.g., stopping)

- guide the robot to places which it can memorise
- point to objects (e.g., rubbish on the floor)
- initiate clean-up tasks, where the robot searches for rubbish, picks it up, and delivers it to the nearest wastebasket.

### 2.1.3 LOLA

LOLA: Visual Object Detection and Retrieval, is a project being undertaken at the Centre for Robotics and Intelligent Machines NC State University, USA, and commenced in 1995 [LOLA95]. The project is aimed at the producing an indoor mobile robot with a high degree of autonomy and the ability to locate and retrieve objects. Features include multi-visual integration and data fusion, navigation, and real-time computer architectures.

The recognition and detection algorithm utilises object features of colour and shape obtained from images generated from an onboard main camera. Processing is as follows: once correct colour is detected, a second camera is used to determine the shape of the object by analysing the distortion of a projected grid emitted by an infrared laser. The robot is equipped with a four degree of freedom arm and gripper, and if the colour and shape match required parameters, the robot will pick up the object and deliver it to a predetermined location.
The hardware platform consists of a Nomad 200 robot-base and turret, from Nomadic Inc., and an Intel 80486 processor.

The objectives of the LOLA project are in many respects similar to those aspired to by the work of this thesis. Significant differences however exist in the navigation and recognition paradigms used.

### 2.1.4 Jumbo Jet Washing Robots-SKYWASH

In a joint venture with AEG and Dornier, the Fraunhofer Institute IPA, the Putzmeister AG in Aichtal, Germany have developed an aircraft cleaning manipulator titled "SKYWASH". Two SKYWASH robots working cooperatively clean Boeing 747-400 jumbo jets, in approximately 3.5 hours, instead of 9 hours for normal manual washing. Huge cleaning brushes travel a distance of approximately 3.8 kilometres and a surface of around 2,400 $m^2$, about 85% of the entire plane's surface area, including the exterior of its engines. The main vehicle is a reinforced chassis made by Mercedes Benz and is equipped with a 380 horsepower diesel engine. Four supporting legs and a manipulator arm of 33 meters in length and 22 tons in weight are mounted on the chassis. The arm has five degrees of freedom (DOF) excluding the attached revolving brush. All the subsystems required for its operation are transported on board including the main computer which is designed as an interactive man-machine interface for effective communications with the operator, sensors, and washing fluid controls. A compact disc read-only memory (CD-R), contains aircraft-specific geometrical data. A 3D camera accurately positions the mobile robot as it travels around the aircraft. The purchase price is around 5 billion German Marks. It is claimed this is amortised within a few years.

### 2.1.5 NOMAD

The planet rover "NOMAD" (a NASA project of 1997) was developed by CMU, its function to explore the landscape to test new ways of communication and control technologies. Weighing 550 kg, it was supplied with a 4-wheel drive enabling it to turn on the spot, and fitted with a chassis that could alter its tracks and wheel base to adjust to any kind of terrain. Visual transfer was enabled with an innovative "panospheric camera" supplying high quality pictures at an extremely wide angle. Distorted images are corrected with specially developed software modules. Exact locating and positioning was carried out using a DGPS system (Differential Global Positioning System) which determined the position of the Nomad as accurately as 20 cm from its actual position in space. Sensed obstacles were recorded and plotted on a digital map. Travelling speed was limited to 0.2 meters per second. The rover completed its first mission in 1997 in Chile.

### 2.1.6 RHINO the tour guide robot

The Artificial Intelligence group at the Institute for Information Technology III, at Bonn University developed Rhino, an autonomously navigating mobile robot capable of interacting with people and performing duties. RHINO is based on the mobile platform B21, (US manufacturer Real World Interface in Jaffrey, NH). It is equipped with 56 infrared sensors reacting to touch. Two 2-D laser scanners and 24 ultrasound sensors, and a stereo colour camera system, provides measurements to generate maps of the surroundings. It is able to explore and map its surroundings. Path planning is based on acquired maps. In 1997, Rhino guided some 2,000 visitors around an exhibition held in the German Museum of Bonn.

### 2.1.7 HELPMATE

A landmark service robot is "HELPMATE". While HELPMATE has been deployed at numerous hospitals throughout the world (King and Weinman [KW90]), it does not interact with people other than by avoiding them. HELPMATE finds its way by using a map of the facility that is stored in its memory. The map is created using AutoCad and contains information about halls, elevators, doors, and stations and is usually limited, in definition, to the areas where the robot will travel. The robot uses this map to determine the exact route that is required to navigate from station to station.

A number of ultrasonic transducers and a camera based vision system help the robot to detect obstacles. Using a spread spectrum radio link, HELPMATE controls elevators and door openers, but it is also possible to use this link for a remote control of the robot.

Typical applications for the robot are:

- pickup and delivery of interoffice mail, medical records and x-ray images
- lab sample retrieval
- pharmaceutical delivery
- delivery of food plates and sterile supplies

### 2.1.8 JIJO-2 Mobile Robot for Office Services

JIJO-2 is an office robot currently being developed by at the Electro-technical Laboratory Tsukba, Japan (Matsui et al [MAFMAKHO99]). Its purpose is to provide office services, such as answering queries about people's location, route guidance, and delivery tasks. Particular objectives of the project include learning abilities, face recognition, and natural spoken conversation with the office dwellers.

www.etl.go.jp/~7440/

**Fig 2.1.8.1. 1 JIJO-2 mobile robot for office services**

## *2.2 3D Object Recognition and Pose Determination*

Numerous authors have carried out detailed reviews of the host of different recognition techniques that exist (e.g. Binford [Bin82]. Besl and Jain [BJ85], Grimson, [Gri90a ], Jain and Flynn [JF93 ], Reiss [Rei93], Weiss [Wei93], Andersson et al [ANE93], Arman and Aggarwal [AA93], Koschan [Kos93], Pope [Pop94], Rothwell [Rot96], Ullman [Ull96], Forsyth and Ponce [FP02]). In light of these, only a selection of model-based recognition systems are reviewed in the section below. In the subsequent sections, a variety of existing methods of object modeling, recognition, pose determination, and hypothesis verification are described. Some of the descriptions below are summaries of sections of the references mentioned above.

### 2.2.1 3D Recognition Systems.

Perhaps the first model based recognition system was produced by Roberts [Rob66]. The system used line segments and corners to recognise polyhedral objects and was based on a prediction and verification strategy. A follow up system was later produced, Guzman [Guz79]. In the 1970's interest focused on feature based techniques as in the prominent work of Waltz [Wal72]. In 1981 Brooks [Bro82] developed the ACRONYM system, which is regarded a landmark in model based vision. Using generalised cylinders as basic primitives for the different parts of the object models, a variety of objects (including articulated objects) could be modeled and

identified. Invariant and semi invariants from object models were calculated and matched to edge structures, ribbons and ellipses, in 2D images. Geometric reasoning and constraint manipulation were used to identify and position objects. Examples of objects recognised were aeroplanes (in an airfield) and engine assemblies. Limitations of ACRONYM were that time overheads were large and also that recognition of objects, such as aeroplanes, approximated a two-dimensional task.

SCERPO constructed by Lowe [Low87] used 2D line segments for the recognition of 3D objects. Perceptual organization was used to initially group lines according to proximity, parallelism and colinearity. Ultimately these grouped lines were matched to similar structures. The implementation used 3D models and 2D images. In the 3DPO system by Bolles et al. [BHH84] range data was used, from which edges are extracted. These can be used to extract ellipses for example. The object models used were created using an extended version of a non-commercial Computer Aided Design (CAD) system. Conveniently making it possible to use the system in connection with different available CAD modelers.

An edge-based system, which uses sparse visual data, titled TINA, was developed at Sheffield Univ. [PPPMF88]. Primitives used for recognition are 3D line segments and ellipses. A feature of this system is the possibility to match several 3D descriptions (views) of one scene to each other. In principle, this facility could be used to produce 3D models by combining multiple view matches. Faugeras and his co-workers [FH86], [Fau92], [MF92], [ZF92], [Fau93] also demonstrated similar work.

Range data systems using surface information, have also been developed, for example IMAGINE I [Fis89]. For this, 3D surfaces and boundary information are used. Objects recognised include robot arm assemblies, rubbish cans etc. A feature of IMAGINE I is that both articulated and occluded objects can be identified. A significant drawback however was that segmentation of surfaces was carried out by hand.

An object recognition system not requiring the use of a stored model was developed by Fan et al [FMN88]. This system relies on a number of views of an object to be first matched, then at a later time, an object may be recognised by matching it to these views.

Researchers have also considered CAD based computer vision. A system developed by Flynn and Jain [FJ91] also based on range images and surfaces, and obtains model information extracted from CAD models. This system unlike the 3DPO system utilises a commercial CAD modeler.

A novel system was developed by Dickinson [Dic91] who describes objects using geons (Biedman [Bie85]). By looking at the geons from different directions aspects are produced. Unlike most systems described, direct metric information is not used for recognition, instead statistics and graph structures consisting of 2D closed contours, obtained from the image aspect geons objects, are utilised.

Another approach relying on view-direction, was developed Caparrelli [Cap99]. The system builds 3D object models from 2D views are collected from a single camera. During training,

images are pre-processed and object views represented in the system's memory using pairwise geometric histograms (PGHs). Image boundaries are first polygonized enabling the geometrical relationship between each pair of line segments to form a frequency-histogram that constitutes a feature vector during matching. PGH registers the geometric relationship between a line segment and each of the surrounding line segments, which lie within a circular region centred on the reference line. It is claimed the characteristic makes the PGH technique a local shape descriptor that works robustly in occluded and cluttered scenes.

## 2.2.2 Modeling

Most machine vision recognition systems rely on model representations of one form or another. In this section various aspects of object modeling are examined; initially different types of geometrical models are introduced, specific descriptions and details follow this.

### 2.2.2.1 Geometric Modeling

With the introduction of CAD came the development of a significant number of geometric modelers. Systematic introductions to CAD modeling can be found in Requicha [Req80] and Requicha and Voelcker [RV81], [RV83], who numerate factors such as: *Domain, Completeness, Uniqueness, Conciseness, Ease of creation, Efficacy for applications*, as important considerations in deciding on a particular modeling methodology. Unfortunately the principal needs of CAD systems are not commensurate with those of machine vision recognition systems, which essentially require that the models consist of primitives that can be identified from an image and that such primitives are easy to extract from the model. Never-the-less partial CAD model representations have been used effectively in various systems (see section 2.2.2.9). Variations on different ways in which objects have been modeled for computer vision applications are briefly presented below.

### 2.2.2.2 Wireframe

The Wire frame model is one of the simplest representations and is commonly used where the primitives extracted from the images are edges or line segments. Early systems developed by Roberts [Rob75] and Pollard et al [PPPMF88] used Wire frame representation.

### 2.2.2.3 Polygon Approximations

Many object surface boundaries can be described or approximated by polyhedrals, and for this reason polyhedral model representation has been of significant interest in computer vision e.g. Lowe [Low91] and Dhome et al [DRLR89].

### 2.2.2.4 Generalized Cylinders

Generalized cylinders [Bin87] comprise of (2D) surfaces swept along a spine (axis) according to a set of rules. Cubes, cones, cylinders, wedges etc. are some examples. Good representations of many objects can be obtained by combining "generalised cylinders". For example, legs of a human being can be modeled with two cylinders attached at the knee etc. Extensive use has been

made of generalised cylinders. Perhaps the best known application was in the ACRONYM system [Bro81].

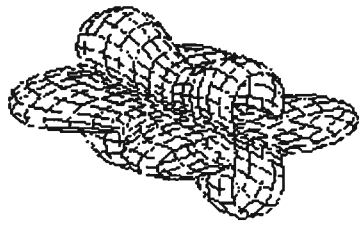
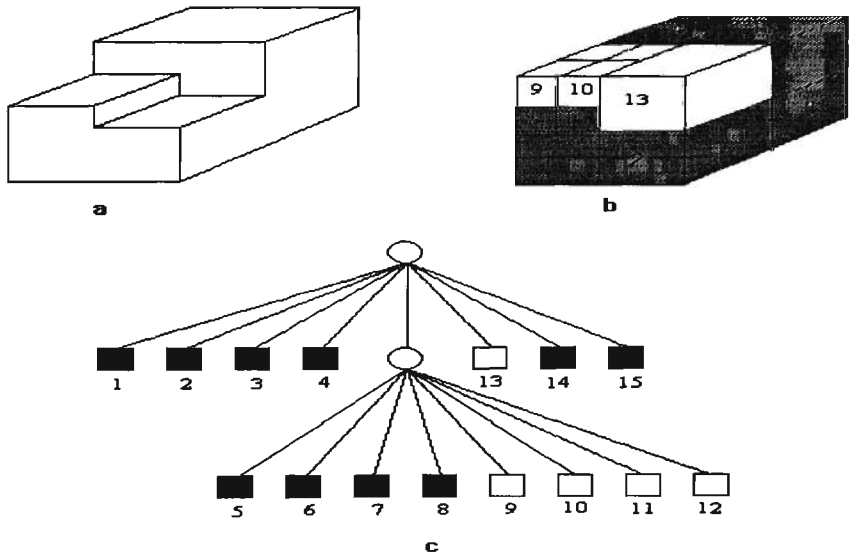


**Fig 2.2.2. 1 Generalized cylinders [Bin87]**

### 2.2.2.5 Octrees

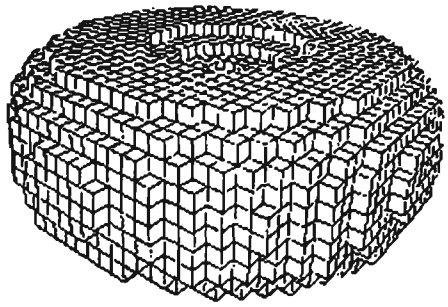
Like Quadtrees, which are areas decomposed into groups of four rectangular primitives, Octrees recursively divide volumes into eight smaller cubes known as cells. Sub-division stops if any one of the resulting cells is homogeneous, that is if the cell lies entirely inside or outside the object. If on the other hand, the cell is heterogeneous, that is intersected by one or more of the object's bounding surfaces, the cell is sub-divided further into eight sub-cells. The sub-division process halts when all the leaf cells are homogeneous to some degree of accuracy (Carlbom et al [CCV85]).




http://graphics.lcs.mit.edu/classes/6.837/F98/TALecture/

**Fig 2.2.2. 2 Octrees**

One drawback in using Octrees to model 3D objects or scenes is that only an approximation can be made due to the use of blocks. See example below.



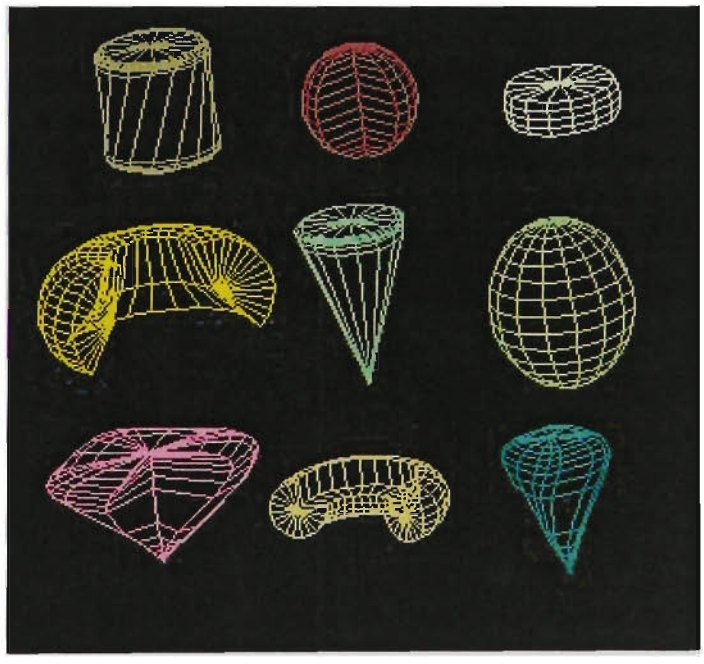
http://graphics.lcs.mit.edu/classes/6.837/F98/TALecture/

**Fig 2.2.2. 3 Octrees block appearance**

### 2.2.2.6 Superquadrics

Superquadrics, see e.g. Solina and Bjelogrlic [SB91] are generalizations of quadrics (Quadratics take the form $Ax^2 + By^2 + Cz^2 + Dxy + Eyz + Fzx + Gx + Hy + Jz + K = 0$, and are the 3D analogue of conics. Conics are 2 dimensional curves described by the general equation $Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0$).

The general form of a superquadric is: $(Ax)^n + (By)^n + (Cz)^n = K$. By varying the choice of the five parameters (A,B,C,K,n), a wide variety of object surfaces can be modeled. One potential drawback of superquadrics representation is that sharp corners, planar surfaces etc, cannot be represented exactly but will be slightly rounded. Superquadrics have been used by Pentland [Pen87] ,Solina and Bjelogrlic [SB91] and Dickinson t [Dic91] and others.



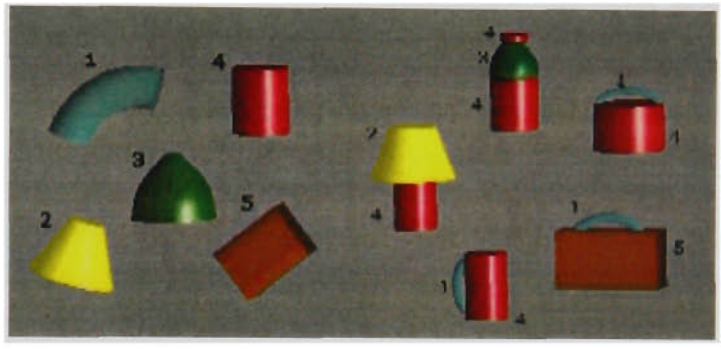
http://www.okino.com/slidshow/super3wi.htm

**Fig 2.2.2. 4 Models produced from Superquads**

#### 2.2.2.7 Geons

The modeling schemes above have all been of a quantitative nature. Biederman [Bie85] proposed a qualitative description called geons (geometrical ions) that allows all objects to be segmented into a maximum of 36 elements. Each element consists of a unique combination of the four features edge (straight or curved), symmetry (rotational/reflective, reflective or asymmetric), size variation (constant, expanding or expanding/contracting) and axis (straight or curved).

To achieve recognition, the scheme proposes a hierarchical set of 4 processing layers. In layers 1 and 2 data is decomposed into edges, component axes, oriented blobs, and vertices. In layer 3, 3D geon primitives, i.e. cones, cylinders and boxes are labelled. In layer 4 the structure is extracted that specifies how the geon components inter-connect; for example, for a human figure, the arm cylinder is attached near the top of the torso cylinder. (Hummel and Biederman [HB92]. Examples of geons and components formed by them are shown below:



(From: Pourang Irani & Colin Ware (2000))

**Fig 2.2.2. 5 Example of Geons**

#### 2.2.2.8 Multiview Representations

The appearance of any 3D object is view direction dependent and can thus be represented by a number of images taken from different directions. One recognition approach is to construct an aspect graph, comprising nodes that represent a view in which identical features are visible. Graph matching methods can then be used to match features in the image. A drawback of this approach is that complex shaped objects may have very large and complicated aspect graphs. A variation on the above method, especially for complex objects, is the use of a view sphere comprising tesselated faces. Each face in the tesselation is used to represent an aspect, which is then used for matching Flynn and Jain [FJ91b]. Aspects are useful both for matching when parts of an object may be occluded and thus not visible, with applications to matching both to 2D images and to 3D range data.

#### 2.2.2.9 CAD

Considerable developmental efforts have gone into producing powerful CAD modeler packages for the manufacturing and building industries. Many manufactured objects as a consequence already have existing CAD produced models. Unfortunately, in general, CAD models are not produced for computer vision recognition tasks and thus obtaining suitable data from the CAD system directly is often quite difficult. From the literature there appears to be two different approaches to using CAD information in computer vision. In the first, the output from the CAD system is used to extract and calculate the information needed, while in the second modification to the CAD system itself is performed to produce the required output information.

The work produced in Flynn and Jain [FJ91b] and Arman and Aggarwal [AA91] are examples of the first approach, while that in Bolles et al. [BHH84] and Park and Mitchell [PM88] are examples of the second. A current drawback in using CAD systems for computer vision is the numerous formats that exist. One attempt to standardize has been in the use of the IGES (Initial Graphics Exchange Specification) format, which many CAD modelers can produce. This format was used both in [FJ91b] and [AA91].

A complication in using IGES, is that output can be in several different forms for the same object. For example GEOMOD, the modeler used in [FJ91b] can produce IGES outputs both in "analytic form" and in NURBS (NonUniform Rational BSplines). Object features such as line segments, circular arcs, parametric spline curves, composite curves, planes and surfaces of revolution are often more easily matched using the "analytic form" output.
In more recently image understanding packages such as [VXL01], and TargetJunior [TJ] have been created which incorporate CAD like formats.

### 2.2.3 3D Object Recognition

The number of recognition techniques developed and described in the literature is quite vast, as already alluded to above. The following is therefore by necessity limited to approaches that are most relevant to the applications in this work.
For a robotic vision system involving remote object retrieval, recognition must be accompanied by pose determination. To facilitate recognition, knowledge of how the objects may appear, plus images of the scene possibly containing those objects will be required. Knowledge of objects appearance is most often provided by way of "object models". Irrespective of the recognition approach used, it must be able to cope with extraneous, spurious, missing and inaccurate data, caused perhaps by poor segmentation, occlusion, image discretization, and inaccuracies in camera modeling. By and large the process of recognition involves "hypothesis generation and verification testing"

#### 2.2.3.1 Hypothesis Generation and Verification Testing

The generation of initial hypothesis involves identifying which features on a certain model, matches that extracted scene features, or alternatively, to which object one or several extracted scene features belong to. If the model database is large, an exhaustive search approach is prohibitive, constrained searches are therefore required. Features having invariant properties under perspective transformations can be used to constrain the search process. Nielsen [Nie85] used a certain ratio of the areas of triangles that were unique for each object. Colors have also been widely exploited (Koschan [ Kos93b]).

#### 2.2.3.2 Use of Invariants

Features that are invariant under transformations, such as rigid motions, perspective projection, and scaling have been widely used in recognition systems for some time. Systems utilizing both 3D object-models and 3D scene data can exploit invariants such as lengths, angles, surface areas, and volumes to aid the matching process. Invariant features or nearly invariant features were used in the ACRONYM system [Bro81] and also by Lowe [Low87] who used groupings of lines based on colinearity, parallelism and proximity. Projective invariants have also attracted interest particularly in work utilizing uncalibrated cameras. Rothwell et al. [RZFM91] use 4 invariants, 2 calculated from 5 coplanar lines, another from a conic and 2 lines lying in the same plane, and a final invariant obtained from 2 conics. An elegance that stems from this work is that models can be acquired from a single view. For recognition, hypotheses as to which object is in the scene are formed by extracting lines and conics from the image, and these are used to establish groups of invariants in an indexed hash table. In circumstances where groups of lines and conics correspond to the same object, these are merged to form final hypotheses. A verification step is then used to narrow down the list to the most plausible object.

#### 2.2.3.3 Geometric Hashing

Geometric hashing was first introduced by Lamdan and Wolfson [LW88] and has similarities with the Hough Transform. Grimson [Grim90a] claims that the method performs well with small scale recognition problems (i.e., models with small number of features and scenes with limited occlusion and clutter), but it tends to suffer from false positives when the problem size is increased. Fundamental to geometric hashing as with most recognition methods, is the assumption that the object can be represented by a number of features. Initially, it is required that several extracted features be used to define object based local coordinate systems, invariant to the transformations applied to the object. To explain further, consider as an example the image a 2D object. A two dimensional coordinate system is established as follows: Two convenient points on the object are chosen to define the x-axis, and the distance between these points is set to 1. The y-axis is taken as a perpendicular line through the centre of the two original basis points chosen. The remaining points on the object are now referenced to this coordinate frame. Essentially this procedure is repeated for each pair of points, i.e. all remaining point pairs on the object are chosen as basis points, and for each such pair the coordinates of the points on the object in each coordinate system used to build a hash table. This procedure is a preprocessing step and need be carried out only once. For the recognition phase, pairs of points in the image are chosen to define local coordinate systems and if a pair is chosen correctly, the coordinates of

the points will, when hashed, point to an entry in the hash table. These points are then used to vote for a certain model. A good match is evident when, a lot of points in the object vote for the same object. A detailed example may be found in Dijck et al [DKH98]

#### 2.2.3.4 Constrained Search

Assuming a database of objects represented by features, in theory it would be possible to try to match every scene extracted feature to every model feature in every possible combination. The number of match combinations can be expressed as an interpretation tree, but in general, if more than a few features are to be matched, the resulting number of combinations will be prohibitively large. The constrained search approach is to prune the interpretation tree significantly and therefore the resulting combinations, by using relations among the matched features, or pairs of matched features.

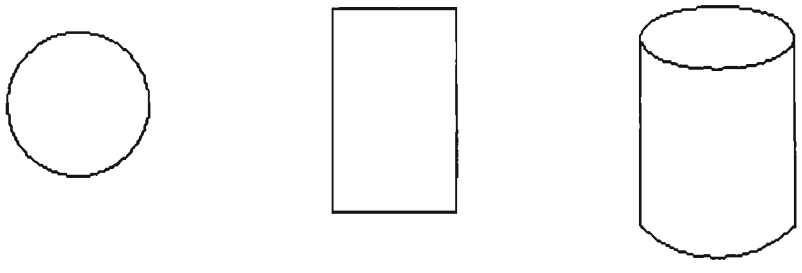
Typical constraints used :

- The length (area) of a data fragment must be smaller or equal to the length (area) of a corresponding model fragment, up to some bounded measurement error.
- The angle between the normals to a pair of data fragments must differ from the angle between the normals of the corresponding model fragments by no more that a bounded measurement error.
- The range of distances between two data fragments must lie within the range of distances of the corresponding model fragments, where the model range has been expanded to account for measurement errors.

Grimson [Gri90b] has shown that if all data features are known to belong to the same object, the degree of complexity of the constrained search is quadratic, whereas for the case that some features do not belong to the model, the complexity may increase to exponential. Pollard et al [PPPMF88] constrained the matching process searches by exploiting several pair-wise relationships between 3D line segments obtained from stereo. While noting that from any pair of matches, all six pose parameters could be calculated by restricting matching to only several parameter related quantities, considerable pruning of the search time could be obtained.

#### 2.2.3.5 Aspect graphs

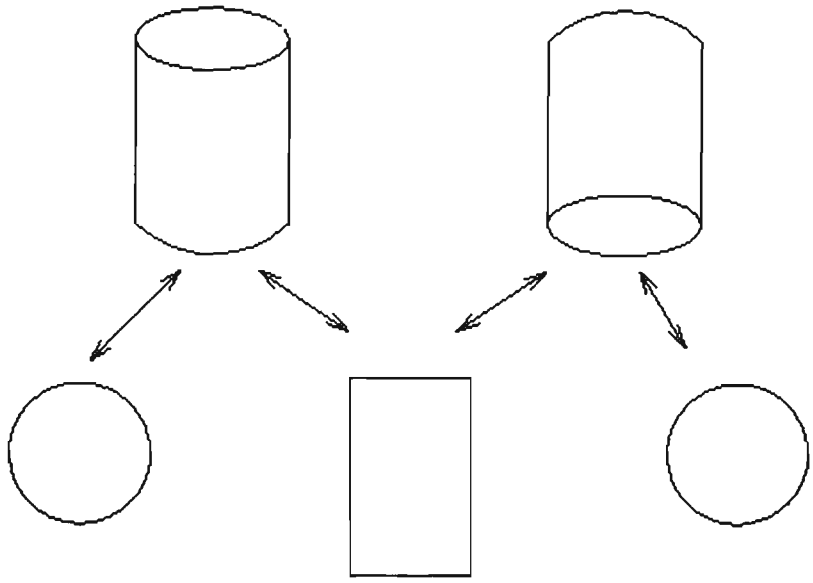
Traditional features used in aspect graphs are lines and curves or surfaces. A cylinder for example has three characteristic views or *aspects*, Fig 2.2.3.1.



http://www.dai.ed.ac.uk/CVonline/LOCAL_COPIES/OWENS/LECT13/node4.html

**Fig 2.2.3. 1 Three characteristic views of a cylinder (Aspects)**

These characteristic views divide 3-space into a finite class of volumes, and each volume represents a different aspect of the object. In essence these views define the *aspect graph*. The aspect graph also comprises of nodes and arcs. Nodes are the volumes in space while the associated arcs correspond to whether volumes are neighbours or not. The aspect graph of a cylinder is given in figure Fig 2.2.3. 2.



http://www.dai.ed.ac.uk/CVonline/LOCAL_COPIES/OWENS/LECT13/node4.html

**Fig 2.2.3. 2 Aspect graph of a cylinder**

Moving from one aspect to another is called an *event*. This corresponds to moving along an arc in the aspect graph. A drawback of the method of aspect graphs is that it is unwieldy for complicated objects. For example, for an object such for Michaelangelo's "David" it has been estimated that there are approximately $10^4$ different aspects (Owens [Owe97]).

Various attempts have been made to reduce the search space (Eggert,et.al [EBDCG92]). An interesting approach has been used by Lowe [Low91] in which a simple hidden line algorithm eliminates most lines that can not be seen from certain viewing directions. For such cases aspects are generated on line as opposed to pre-computation, which is the more common method and quite time consuming.

### 2.2.4 Pose Determination

Determining the pose of an object for recognition based vision systems is important for a number of reasons. If physical retrieval of the object by a robot arm is desired then pose information is essential to direct grasping of the object. If pose can be calculated during the matching phase (say from 2, 3D line matches), this can be used to constrain the search process considerably.

Methods used to calculate pose vary depending upon factors such as: type of recognition system (i.e., matching 2D objects from 2D data, 3D objects from 2D data or 3D objects from 3D data), the features used (i.e., points, lines, surfaces etc), and the imaging (i.e., range images, projective, orthographic etc). For some recognition systems pose can be determined during the matching

stage. One example of this is mentioned above, another is the case of aspect graph matching, since each unique aspect corresponds to a unique view direction, position and scale is apparent. More commonly however pose is calculated either analytically or iteratively from a number of matched features. Solutions to pose determination have been given by authors Huttenlocher and Ullman [HU88], Dhôme et al. [DRLR88], Lowe [Low85], [Low91], Faugeras and M. Hebert [FH86]. A number of these are be briefly discussed below.

For recognition systems using the matching of 2D features to 3D models, Huttenlocher and Ullman [HU88] demonstrated that for 3 point to point matches, under a weak perspective projection (orthographic projection + scale) a unique analytic solution exists. Dhôme et al [DRLR88] also presented a method using 3 line correspondences under perspective projection. The solution, while analytical, requires that the roots of an 8th degree polynomial be determined. An alternative iterative solution, using more than 3 lines and based on Newton's method suggested by Lowe [Low85] is also given. Lowe [Low91] developed another iterative solution using a Levenberg-Marquart technique and a stabilizing function, making it possible to solve under constrained conditions. The method is able to handle articulated objects as well as solving for object parameters, ie. size. In the case of recognition systems using 3D feature to 3D model matching, if line-features are used, a unique solution is possible from 2 matches, while for point features, 3 matches are required.

### 2.2.5 Verification of Object Hypotheses

To complete the recognition process, once a hypothesis of a matched object has been established it is necessary is to verify that the hypothesis is correct. Frequently the pose of the object will make it possible to predict the location of unmatched model features in the image(s). These can then be searched for and if matched verification accepted. Fisher [Fis89] argues that all features (surfaces) in a model should be accounted for, that is, either they should be found in the image, or predictably occluded behind some object. Other methods use an empirically determined threshold on the fraction of model features that must be matched to data features.

### 2.2.6 Recognition Strategies

Prior to implementing a recognition system, decisions need to be made relating to strategies in applying primitives and their use as features, as well as the order of matching. Often these choices are governed by the specifics of the applications involved. Intuition, refined by trial-and-error, also play a role in the decision making process. In more recent years, systems that automatically produce recognition strategies for a given model, have been considered e.g. the classic works by Goad [Goa83] and Hansen and Henderson [HH89] in which automatic recognition strategies were produced from CAD models of objects. A variation by Ikeuchi and Kanade [IK88] demonstrated that a recognition strategy could be achieved from an object model and, importantly, a good model of the sensor. Draper [Dra93] in his "Schema Learning System" (SLS), incorporates learning knowledge-directed recognition strategies from training images (with solutions) and a library of generic visual procedures. In order to represent strategies, recognition is modeled in SLS as a sequence of small verification tasks interleaved with representational transformations. At each level of representation, features of a representational instance, called a hypothesis, are measured in order to verify or reject the hypothesis. Hypotheses

that are verified are then transformed to a more abstract level of representation, where features of the new representation are measured and the process repeats itself. The recognition graphs learned by SLS are executable recognition graphs capable of recognizing the 3D locations and orientations of objects in scenes.

### 2.2.7 Choice of Recognition System for MROLR

From the preceding review of the literature, and from a knowledge that target objects to be located and retrieved by MROLR are man-made, a conclusion reached is that matching techniques utilising features such as lines, curved segments, and/or corners, in a constrained search, is the most appropriate for this work. The system based on the work of Pollard et al [PPPMF88] was chosen to form the foundation building blocks for the recognition module of this research. More specific details of the benefits of using 3D edge-based models are enumerated in Section 6.1.

## *2.3 Robots with Metric Maps*

In recent years considerable advances have occurred in the area of automated mobile vehicles. The ultimate objective of much of the research is to build mobile robots capable of autonomous navigation without any human intervention. Duckett [Duc00, page 37] points out: Most roboticists are more interested in building robots that work than in producing realistic cognitive models with knowledge acquisition capabilities. It is common practice in mobile robotics to incorporate in the design the necessary behaviours, feature detectors, etc., required for a particular application. If the robot is then transferred to a new environment the preinstalled competencies may fail. By contrast a learning robot need not be given all of the details of its environment by the system designer, and its sensors and actuators need not be finely tuned (Dorigo & Columbetti [DC98]).

The robots described below (in some instances summaries of Duckett [Duc00] Surveys), were selected for their particular relevance to the problem in hand of concurrent map building and self-localisation. Each employ models of the environment in which an explicit Cartesian reference frame is used for referencing, and mapping is either feature-based or grid-based. No attempt is made to separate the systems on the grounds of the sensor types used for perception. Ideally any navigational system should be able to accommodate and integrate new sensor types if they prove to be of benefit.

### 2.3.1 Feature-Based Maps

#### 2.3.1.1 AGVs

Leonard & Durrant-Whyte [LD92] using various Robosoft automatic guided vehicles (AGVs) were able to attain precise metric maps from sonar sensors. The maps were built from a set of pre-defined geometric features including planes, cylinders, corners and edges. These features were made up of primitives known as "regions of constant depth" (RCDs), and consisted of groups of sonar returns of similar range. While the robot was in motion, self-localisation was

achieved by tracking stationary features in the environment and applying an extended Kalman filter to combine the inferred position estimates. An important contribution of this work was the development of an improved sonar sensor model for robot navigation.

A multiple hypothesis tracking technique to deal with the problems of map building in dynamic environments was proposed by Cox & Leonard [CL94]. Specifically the approach endeavoured to account for different possible interpretations of the robot's sensor data by maintaining multiple environment models at each time step. Models were associated with a probability, reflecting likelihood of it being the "correct" model. Testing was verified in selected environments.

A map building mobile robot named ARNE equipped with a single rotating sonar sensor, was implemented by David Lee [Lee95]. Navigation utilised a feature-based metric map and self-localisation obtained via an extended Kalman filter. An additional grid-based map was used for the purpose of path planning, and this was derived from the feature-based map. Evaluations of various exploration strategies, used by the robot to build its maps were performed (Lee & Recce [LR97]). The best results were achieved by a hybrid model-based reactive exploration strategy, consisting of wall following with some map-based interventions according to a predefined set of heuristics. Testing was carried out in a static, laboratory environment.

Davison [Dav98] implemented a real-time feature-tracking autonomously navigating mobile robot system capable of exploration in unknown environments, using the OXFORD GTI mobile robot platform and Yorick active binocular vision system. Map building was carried out using information from measurements of arbitrary features and robot odometry. Map features were updated (i.e., added or deleted), in an automated systematic maintenance procedure. An emphasis of the work was in producing maps useful over long periods of time, permitting features to be re-detected/re-measured, in areas previously visited, even if the original trajectory was not followed, and in addition to be able to recover accurate position estimation after periods of drift. A Kalman Filter-based method utilizing full-covariance was used.

Testing in a real environment was carried out with ground truth comparisons performed using an accurate grid. Greater details of Davison's algorithms and approach are given in Chapter 6 of this thesis.

**Fig 2.3.1. 1 Oxford's GTI vehicle (from [Dav98])**

### 2.3.2 Grid-Based Maps

Grip-based maps utilise Cartesian occupancy grids, where each cell contains a measure of certainty that any object occupies the corresponding location in the robot's environment. The probabilities are obtained from pre-defined sensor models that project the robot's range-finder readings onto the corresponding grid cells. A Bayesian update rule is used to combine multiple readings over the same cell when necessary. Self-localisation is achieved through correlation of a grid constructed from the current sensor readings with a stored map, and then finding the displacement and rotation which produces the best match between the two grids. Early successfully implementations using grid-maps were carried out by Moravec and Elfes [ME85], [ Elf 87].

An advantage of the grip-base map approach is that it avoids the correspondence problem since there is no need to identify the source of the robot's sensor returns. In addition, occupancy grids provide a natural representation for combining different sensor modalities, provided that a good model is available for each different type of sensor. For example, Thrun et al [TFB98a] used an occupancy grid to fuse range information from stereo vision and sonar. The disadvantages of the approach are that it takes up a large amount of memory, requires precise position information for map building and depends on accurate range finder sensing. For example, the specular effects associated with sonar sensors often result in geometric errors in the map.

#### 2.3.2.1 ARIEL

Yamauchi et al [YSA98] developed ARIEL an autonomous mobile robot built on a Nomad 200 platform. ARIEL was equipped with a planar laser range finder, sonar and infrared sensors. During exploration of an unknown environment it built its own map using an integrated map building strategy known as frontier-based exploration (Yamauchi [Yam97]) which included a continuous localisation technique for correcting the robot's odometry (Schultz & Adams [SA98]). Regions between open and unexplored space in a global grid model known as "frontiers" were detected using a process analogous to edge detection and region extraction in computer vision. An interesting feature of the system was a sensor scanning technique in which specular reflections affecting the robot sonar sensors were corrected by the laser range finder.

The exploration strategy encouraged the robot to navigate to the nearest frontier. If this was reached, a new sensor scan was performed and the map updated, with any new frontiers detected, added to the list of unexplored locations. A short-term local occupancy grid, constructed from the robot's recent perceptions, was matched to the long-term global map to perform self-localisation. Using this matching process the possible translational and rotational errors were restricted to a small space in the global grid which centred round the current position estimate, produced by dead reckoning. Duckett [Duc00] points out that this dependence on prior position knowledge for self-localisation means that the system would be unable to recover from becoming lost.

http://www.aic.nrl.navy.mil/~schultz/research/ariel/

**Fig 2.3.2. 1 ARIEL**

### 2.3.2.2 RHINO

RHINO (also earlier referred in Sect 2.1.6) uses both vision and sonar sensors to build accurate metric maps. Occupancy probabilities for the grid-based map cells are obtained from a neural network fed from sonar readings which are trained to convert proximity readings into circular, metric maps. Increasing geometric accuracy of the maps is obtained by combining information from neighbouring sensors to reduce specular effects. Using vision sensors in addition to sonar sensors permits depth information of objects to be recovered from camera image edges, which may not be detectable using sonar alone.

RHINO also produces a topological map that is used for path planning. This map is derived from the grid-based map by using critical points detected in a Voronoi diagram to partition the unoccupied space in the grid into a set of discrete locations (Thrun [Thr98b]). The benefit of using both metric and topological maps is that it allows the robot to exploit the "orthogonal strengths" of each. Localisation is estimated from a position probability density function by repeatedly matching the sensor readings with a model of the environment.

**Fig 2.3.2. 2 RHINO (From Univ. Bonn, Germany)**

### 2.3.3 Robots with Topological Maps

Topolopogical Maps utilise a graph of connected places to represent the environment; self-localisation becomes the task of place recognition (Kortenkamp & Weymouth [KW94]). Pioneering work on navigation using topological maps carried out by Kuipers & Byun [KB91]. Systems incorporating topological maps have the advantage that the robot does not need to know its exact position for map building. Lee [Lee96] implemented Kuipers' "Spatial Semantic Hierarchy" on a real robot, but the system was only tested in a small-scale laboratory environment consisting of three cardboard corridors.

#### 2.3.3.1 TOTO

TOTO, a wall-following mobile robot developed by Mataric [Mat91], used a topological map and explored its environment. Landmarks were classified according to a designer-determined set of categories and identified using sonar sensors and an electronic compass. The topological map comprised of nodes representing each distinctive landmark which were linked to adjacent landmarks on route during the exploration phase. While navigating, a path to a goal location was found by spreading activation from the destination node.

www-robotics.usc.edu

**Fig 2.3.3. 1 TOTO**

### 2.3.4 Self-Organising Robots

#### 2.3.4.1 ALDER and CAIRNGORM

ALDER and CAIRNGORM, the name given to two Fischertechnik robots developed by Nehmzow [Neh92], utilise a reactive controller, and wall-following and robot-determined landmarks to identify places. A Kohonen self-organising feature map is used for map building, allowing the robot to construct its own internal representation of the environment by clustering together similar groups of sensor readings. All world models used in this system were acquired autonomously by the robots, and sensor-motor competences were learned by a simple feedforward neural network. Others (e.g Owen & Nehmzow [ON97], Duckett & Nehmzow [DN98]) validated the efficacy of self-organising feature maps for route-learning, wall following, and relocalisation after becoming lost, utilising a Nomad 200 robot operating in untreated, middle-scale environments. One drawback of this approach is that it is that the robot is restricted

to following fixed paths because location recognition depends on visiting locations in the same sequence as used for map building.

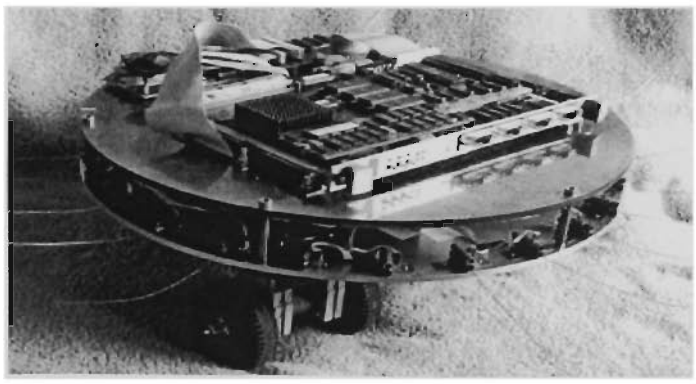
#### 2.3.4.2 ALEF

ALEF, a topological mapping system, developed by Kurz [Kur96] using a modified RWI-B12 robot, utilises classification algorithms such as self-organising feature maps to group together similar sets of sonar readings obtained from sonar sensors. From resulting feature categorisations, the robot's environment is partitioned into contiguous regions known as "situation areas". A graph-based representation of the environment containing topological relations between the situation areas was constructed during map building.

Using the so called A* algorithm (Nilsson [Nil80]), for path planning, the robot was also able to navigate to arbitrary locations in the map. Self-localisation was achieved using the average of odometry dead reckoning, and the coordinates obtained of observed situation areas via a Kalman filter. The approach used has the disadvantage that dead reckoning is soley relied upon when exploring unknown areas.

#### 2.3.4.3 ALICE

Zimmer [Zim95a] developed ALICE, a concurrent map building and self-localisation mobile robot, built deliberately using only low resolution, low reliability sensors, to permit the investigation of poor quality sensor information when navigating in an unknown environment. It was equipped with passive light sensors and touch-sensitive whiskers for location recognition, and a basic dead reckoning mechanism affected by drift errors of up to 25%. ALICE, in contrast to many other robots, had no separate phases for map learning and navigation. Instead it was able to continuously adapt its internal representations through a process of lifelong learning. An interesting extension of the Growing Neural Gas network (Fritzke [Fri95]), consisting of a set of stored sensor prototypes augmented with Cartesian coordinates and the topological connections between them, was used for map building. Exploration was carried out using a reactive controller, which was subject to top-down influence from various "instincts", such as trying to reach areas of unexplored territory, or trying to improve localisation quality by moving through areas of previously charted territory. One of the most important contributions of this research was the handling of the concurrent updates to the robot's environment and location models.



http://transit-port.net/Uwe.Zimmer/Projects/ALICE/ALICE.html

**Fig 2.3.4. 1 ALICE**

## 2.3.5 Hidden Markov Models

### 2.3.5.1 DERVISH

Nourbakhsh [Nou98] using a customised Nomad 100 robot, implemented a navigation system named DERVISH, in which predefined feature detectors were used to identify landmarks such as doors or junctions. It utilised a high-level self-localisation algorithm known as state set progression. The robot's location model consisted of a "state set", containing a subset of the possible locations in a pre-installed topological map. Features detected in the current sensor data, as determined by the landmarks associated with the locations in the map, were used to initialise the set. Updating of a state consisted of removing it from the set and replacing it with all of the possible subsequent states, derived from the new sensor data and the robot's direction of travel. Each state was assigned a probability value using a predefined "certainty matrix", which represented the likelihood of obtaining the observed features from the actual landmarks in the environment. During the set update, these values were propagated in Bayesian fashion. While Dervish used the most likely state to plan a path to a goal location, it would stop and re-plan if values suggested it was no longer on the correct path. A shortcoming of the approach was the necessity to use a preinstalled topological map, feature models and an uncertainty matrix.

### 2.3.5.2 XAVIER

XAVIER, a mobile robot system implemented on a RWI-B24 by Koenig & Simmons [KS96], [KS98], was specifically designed for performing delivery tasks in an office environment.While the methodology used for feature detection and self-localisation was similar to that used by DERVISH, the environment and location models were obtained from a Partially Observable Markov Decision Process (POMDP). A pre-installed topological map was also required, but the system had some ability to adapt its environment model through on-line learning, the user-defined map being augmented with distance information from the robot's sensors and actuators. A specially written compiler translated pre-installed models into the POMDP representation. During navigation an extended version of the Baum-Welch algorithm (Rabiner [Rab89]) was applied to improve the compiled distance, sensor and actuator models by adjusting the corresponding probabilities in the POMDP model. The Baum-Welch algorithm is an expectation maximisation (EM) method for acquiring Hidden Markov Models (HMMs) and POMDPs from data. As with the previous systems, the main disadvantage of this approach is its reliance on pre-installed knowledge.

Carnegie Mellon University, Robot Learning Lab

**Fig 2.3.5. 1 XAVIER**

#### 2.3.5.2 RAMONA

Shatkay & Kaelbling [SK97], using a modified RWI-B21 robot named RAMONA, applied an extended the Baum-Welch algorithm to perform off-line map building from pre-recorded sensor data. An extended HMM was used to incorporate odometric information, and an expectation maximisation algorithm maintained geometric consistency in the model. A clustering algorithm provided the initial model, which was based on local odometric relations extracted from the recorded sensor data. The final version of the algorithm produced better models from less data and fewer training iterations, and was capable of learning models for environments containing loops. A drawback in the use of HMM and POMDP models is that the robot's environment model has to be quantized into a set of discrete states.

http://www.cs.brown.edu/research/robotics/robots/ramona.html

**Fig 2.3.5. 2 RAMONA**

### 2.3.6 Robots with Hybrid Maps

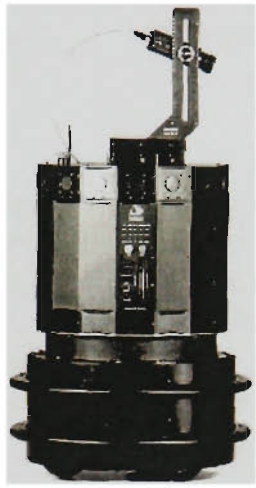
Integrated systems utilising both topological and metric representations have been investigated by researchers such as Edlinger & Weiss [EW95]; Yamauchi & Beer [YB96], Simhon & Dudek [SD98], Gasos & Saffotti [GS99]. In these systems, a graph of connected regions is used to represent the environment globally, and each region by a small-scale local metric map. Path planning and middle-scale navigation, rely on topological representation, while local metric maps are used for small-scale navigation. The advantage of this approach is that a globally consistent metric map is not required.

#### 2.3.6.1 ELDEN

A navigation system, known as ELDEN was developed for a Nomad 200 robot, by Yamauchi & Beer [YB1996]. A reactive controller was used for exploration that included an arbitration mechanism to combine predefined behaviours such as obstacle avoidance and wandering. a topological map was constructed from odometry calculations, with new places augmented whenever the distance to the nearest stored place exceeded a preset value. Dead reckoning drift errors were corrected by periodically returning the robot to its starting location and activating a special recalibration routine. The recalibration routine used an occupancy grid, constructed from

the current sonar readings, to compare a previously stored grid, and used a hill climbing method to find the transformation producing the best match between the two grids. Finally this transformation was used to update the robot's odometry. Yamauchi & Langley [YL97] subsequently extended ELDEN by incorporating stored occupancy grids, one for each place node in the map. These were used for recalibration thus removing the need for regular homing.

A novel aspect of the system was the use of variable confidence links to represent the uncertainty in the topological relations, and then to use these to adapt the robot's map in dynamic environments. In essence, confidence levels were adjusted using simple rules, to strengthen or weaken them, subsequently enabling the system to use these values to plan alternative routes should obstacles or path blockages occur.



http://www.aic.nrl.navy.mil/~yamauchi/gallery.html

**Fig 2.3.6. 1 ELDEN**

### 2.3.6.2 MOBOT-IV

MOBOT-IV a mobile robot developed by Edlinger & Weiss [EW95], utilised a dynamically acquired internal representation of unexplored environments. The navigation system used a 360 degree laser range-finder to form a set of local metric maps which were linked via stored connections to a global topological map. A temporary metric map was first formed using the laser range finder, and then used for self-localisation and to detect areas of unexplored territory. In practice, a predefined threshold based on the distance of the robot to the nearest stored node, was determined whether the current sensor map was to be added to the global map. Cross-correlation of the current sensor map with the nearest stored scan in the global map was then used to obtain self-localisation. Effectively the approach used was the same as that of Hinkel & Knieriemen [HK88], in that angle histograms were first constructed and convolved to find the most likely rotation of the robot then $x$ and $y$ histograms were matched to determine the most likely translation. A region of open space detected in the current sensor scan with a width greater than that of the robot, was defined as a "passage", ultimately detected passages were added to a stack of goal destinations. Path planning was carried out on the topological map using the A* algorithm (Nilsson [Nil80]). The system was tested in a static indoor corridor that demonstrated

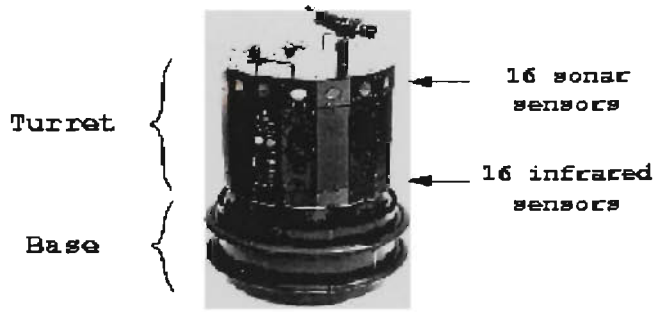
both its reliability and scalability. A weakness of the system is its reliance upon prior positional information for self-localisation.

http://ag-vp-www.informatik.uni-kl.de/Projekte/MOBOT-IV/MOBOT-IV.html

**Fig 2.3.6. 2 MOBOT-IV**

### 2.3.6.3 The Duckett Mobile Robot

Duckett [Duc00], utilising a hybrid deliberative-reactive control architecture developed a mobile robot system in which all of the environment and location models, feature models, and sensor-motor competences required for navigation are acquired independently by the robot. A significant distinction of Duckett's system is the claim that the robot could recover its position even after becoming lost. Learning techniques include self-organisation, where no teaching signal is required, and self-supervised learning, where all of the training examples are generated by the robot. The system is able to build its own maps and navigate in many different, indoor environments. During an exploration phase, the robot builds a graph-like representation of its environment, in which each location is identified by a description of the robot's sensory information known as a landmark. To determine an appropriate landmark recognition mechanism, an experimental procedure was developed which permitted different algorithms to be compared under identical experimental conditions. With this method existing approaches to landmark recognition were evaluated, and a new self-localisation system was developed. To overcome problems such as perceptual aliasing (the fact that landmarks may not be unique to individual places) a self-localisation algorithm was developed which accumulates sensory evidence over time so that the robot can recover its position even after becoming lost. The work was validated by testing in untreated environments, of several hundred metre square.




The Nomad 200 mobile robot *FortyTwo*.

**Fig 2.3.6. 3 Duckett mobile robot [Duc00]**

## 2.4 Mobile Robots using Vision Based Sensors For Navigation

Desouza and Kak [DK02], and Zhang [Zha02a] carried out comprehensive surveys on "Vision for Mobile Robot Navigation". It is evident from these surveys as well as from the robots already discussed in section 2.2 that in the past 20 years very significant developments and progress in vision-based mobile robots for both indoor and outdoor applications has occurred. To a large part these improvements were facilitated by the very substantial improvement in PC speed, and video handling. Desouza and Kak point out that it is now possible to design a vision based navigation system that utilises topological representation of space, and an ensemble of neural networks to guide a robot through interior space. The topological representations helping the robot figure out which neural networks to use, and in which part of the space. Examples of indoor robots are NEURO-NAV [MK93a], [MK93b] and FUZZY- NAV [PPKK95] and FINALE [KK92]. These robots can navigate at an average speed of 17 m/min using an ordinary PC-based architecture (Pentium II 450Mhz, with no special signal processing hardware).

Desouza and Kak also cite the equally impressive progress that has been achieved in computer vision for outdoor robotics. Representative examples are the NAVLAB system (Thorpe et al [TKS87], [THK88], [Pom89], Tseng et al [TC92], Pomerleau [Pom94], [PJ96]), the work on vision-guided road-following for "Autobahns" (Dickmanns et al [DZ86], [DZ87], [DM92], [Dic99]), and the Prometheus system (Graefe et al [Gra89], [Gra92a], [Gra92b], [Gra93], [RG94]).

Their findings suggest the following broad base categorisations:

(i) Map-based Navigation. These are systems that depend on user-created geometric models or topological maps of the environment.

(ii) Map building based Navigation. These are systems that use sensors to construct their own geometric or topological models of the environment and then use these models for navigation.

(iii) Map-less Navigation. These are systems that use no explicit representation at all about the space in which navigation is to take place, but rather resort to recognising objects found in the environment or to tracking those objects by generating motions based on visual observations.

They conclude their survey with a somewhat unexpected finding: "... if the goal is to carry out function-driven navigation ... an example being to fetch a fire-extinguisher that is somewhere in a given hallway or to stop at a stop sign under varying illumination and background conditions -- we are still eons away".

To a large extent this last observation puts into context the difficulty of the work undertaken by this thesis.

## 2.5 Conclusion

This chapter has highlighted the rapidly growing applications of robots for the non-industrial and manufacturing sectors these are the so-called "service robots". Brief details of some of these service robots and applications were given.

A synoptic literature review of 3D recognition systems and modeling methods, and robots using autonomous navigation methods was also presented. It is from this review that 3D wire-frame models were decided upon as being the most suitable for the recognition module of this work.
The literature survey concludes with a summary of a recent survey on mobile robots using vision-based sensors for navigation. A noteworthy conclusion reached in this survey is that the "machine vision fraternity" is still a long way off from being able to reliably use mobile robots to find and retrieve objects in "unmodified" settings. That may well be true, but then humans also need artificial aids such as good lighting, and a range of temperatures, to work well in.

In the next Chapter, specifics of the hardware and software required to implement MROLR are discussed.

# Chapter 3 System Equipment

## 3.0 Introduction

To meet the objectives of this thesis in building a robotic system capable of navigating with the aid of active vision, visually locating and physically retrieving real-world objects, requires considerable hardware and software resources, a substantial proportion of which is not readily obtainable "off-the-shelf". Unobtainable hardware was therefore specifically designed and built, and software coded.

In this chapter brief details of the hardware and software modules developed and utilised for this work are described.

## 3.1 Hardware and related software

The major hardware components of MROLR comprise:

(a) A TRC[1] Labmate mobile robot base powered via 2x12V fork lift truck batteries.

(b) A BiSight 4 DOF pan-tilt–vergence stereo head, grayscale CCD cameras fitted with 13mm fixed focal lenses cameras (the head was also purchased from TRC).

(c) A Win32 Imascan framegrabber residing in a win95 platform PC on board the mobile base.

(d) Two computerised revolving horizontal tables (designed by the author and assistant technical staff -see appendix B). These tables facilitated the creation of object-models.

(e) Camera Calibration Tile (Fig 3.1.2).

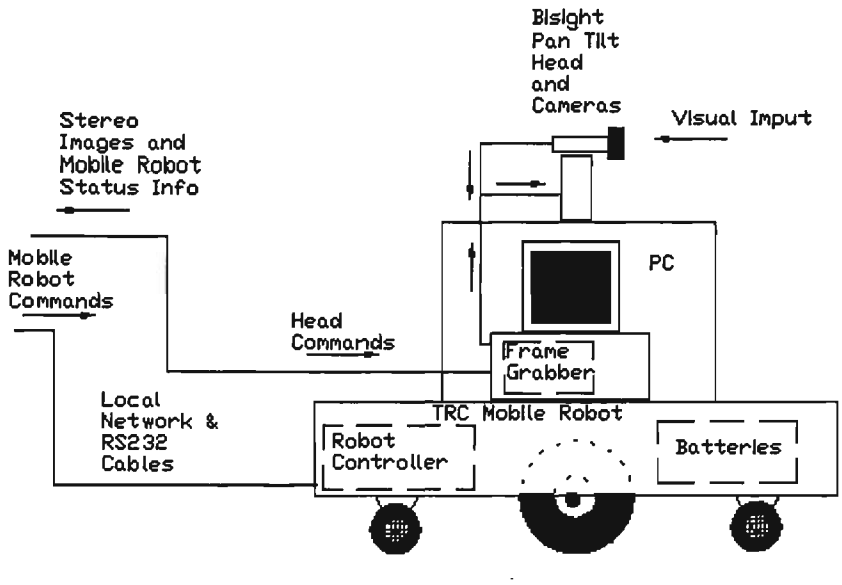
(f) UMI RTX, 6 DOF robot arm, controlled via a PC (docking robot), mounted on either a caster platform or mobile transport robot (refer to (g)).

(g) Mobile Transport Robot: this hardware is in a state of partial completion (designed by the author and assistant technical staff -see appendix B).

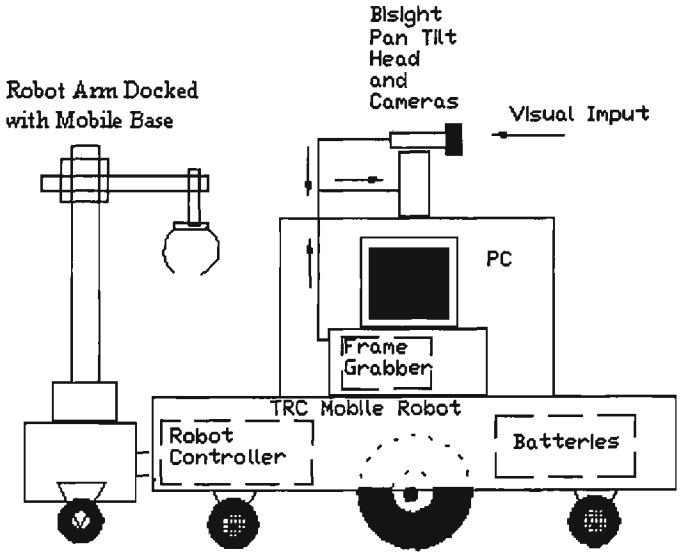
(h) Several PC's interconnected via a local area network.

Figures 3.1.1 (a) and 3.1.1 (b) are sketches of the mobile base and docking arm and show system local network networked data flows. Numerous photo images of the hardware are provided in Chapter 7.

[1] Transition Research Corporation, USA.




(a)




(b)

**(a) TRC Mobile Robot Base, (b) with Docked Robot Arm**

**Fig 3.1. 1 Sketch of MROLR**

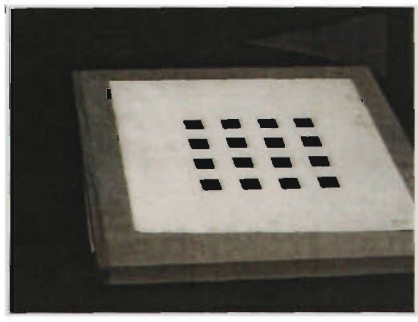


**Fig 3.1. 2 Calibration tile**

**PC Load Distribution**

The computational load is distributed using a number of PC's interconnected via a local TCP/IP network. This additionally accommodates platform specific requirements eliminating the need for the rewriting of third-party proprietary supplied equipment drivers.

Further details are given below:

(1.) Stereo head PC resides on the Mobile Base *(platform: Win95/Dos ).*
Main functions:
Execution of head movements commands.
Capture, and transmission of stereo scene-images.

(2.) TRC Mobile Base PC *(platform: Win3.11/Dos).*
Main functions: Execution of commands relating to mobile base movements.
Monitoring and displaying of odometry positional information.
Control of motorised model-creating tables.

(3.) Principal Navigation and Recognition PC (*platform: RedHat 7.0 Linux*).
Main functions: Processing of stereo image pairs for purposes of navigation, localisation, object recognition and object pose determination.
Coordination of data sharing across the local-network.

(4.) Robot Arm PC *(*platform:Win98 *)*
Main function: Accept object pose information via dynamic data exchange (DDE). DDE provides a convenient mode of data interchange for Win32 applications.
Calculation of the necessary inverse kinematic solution to facilitate the grasping and retrieval of recognised object.

(5.) Transport Mobile Base PC (*platforms:* Motorola HC11 and Win98).
Main function: Respond to request to dock with the TRC mobile base, accept object pose information ... as for (4).

Essential software links and patches were written to facilitate the communication, integration, and co-operation of each of the various platforms and items of equipment. This included image format converters and DDE programs running in the background.

Processing time was not greatly considered as the PC's used were rather old (60-90 MHz, Pentiums with 32 MB RAM). At the time of writing, PC clock speeds over ten times faster were available and consequently would produce a corresponding improvement in performance if obtained and utilised.

## 3.2 Navigation and Recognition Software

The Navigation and Recognition Software has been largely developed around two software resources, namely "TINA" and "SCENE".

TINA is a machine vision research environment, running under Unix (X-windows). It is written in C and includes a substantial collection of integrated image processing functions. The development of TINA commenced at the University of Sheffield and more recently continued at the University of Manchester. TINA has been written to provide a research environment for machine vision programmers. It utilises a mouse driven front end, and an integrated package of vision library routines, running under the UNIX operating system with Xwindows graphics support. The individual modules and display facilities are provided as tools under the control of a parent that also displays diagnostic messages. These sub-tools can be grouped in to two types, graphical infrastructure and research.

The following sub-tools are used in this thesis:
Stereo Tool
Edge Tool
Matcher Tool
Calibration Tool

The Matcher tool was modified and extended. In addition a 3D Model editor tool was added. Further details are given in Chapter 4

SCENE is a flexible open-source C++ library for sequential localisation and map-building (SLAM). It is suited to 2D/3D mobile robot navigation, this being accomplished by the estimation of the states of the moving robot and numerous stationary features. In general measurements are made via one or more sensors mounted on the robot. A feature of SCENE is its modular modeling system, which is used to represent the specifics of a particular system. New model classes can be "plugged in" to the main system with relative ease. Model classes suitable for MROLR were implemented and incorporated into SCENE. In addition a vision based obstacle avoidance function was added. Details are provided in subsequent chapters.

Both TINA and SCENE are significant contributions to the Scientific Community and generously made freely available at the web sites indicated in the Bibliography at the end of this thesis.

## 3.3 Conclusion

A complex system such as MROLR requires considerable hardware and software resources. While a policy of using available resources where obtainable was adopted, tailor designed specific hardware was required to be designed and built, and software written. This chapter outlined the resources required and utilised. Subsequent chapters elaborate on how these resources are used or developed. Chapter 4 details the mathematical notation and models used throughout this work.

# Chapter 4 Background on Models and Notation

## 4.0 Introduction

In this chapter, mathematical notation, concepts, and a variety of preliminary models used and extended in subsequent chapters are outlined and developed.

The chapter is effectively divided into three sections. The first introduces the mathematical notation and conventions adopted throughout this thesis. The second considers modeling of:

- cameras and the Bisight stereo head
- the mobile robot base

The third section considers algorithms and tools used for:

- stereo camera calibration
- object and scene-modeling,
- object recognition

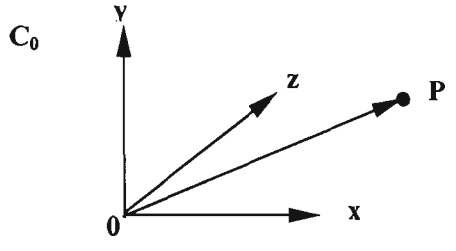
## 4.1 Mathematical notation and conventions adopted

### 4.1.1 Vectors, Matrices, and Coordinate Frames

Vectors and matrices will in general be distinguished by boldface type.
Examples:

vector $\mathbf{P} = P_x\mathbf{i} + P_y\mathbf{j} + P_z\mathbf{k}$ represents a 3D point in space having coordinates $P_x$, $P_y$, $P_z$ respectively.

Coordinate frames are designated by capital letters. Where several frames are related, an additional index integer, or subscript is used (i.e., $C_0$, C1). Vector point **P** in frame $C_0$ is defined by the 1 x 3 matrix given below. (Note C0 and $C_0$ designate the same frame).




$$\mathbf{P}^{C_0} = \begin{pmatrix} P_x^{C_0} \\ P_y^{C_0} \\ P_z^{C_0} \end{pmatrix}$$

**Fig 4.1.1. 1 Vector point P in frame $C_0$**

### 4.1.2 Partial Derivatives

The $\nabla$ (del) operator is used for partial differentiation. $\nabla(y)_x$ is used to represent $\frac{\partial y}{\partial x}$. If either $y$ or $x$ is non-scalar, bold typeface is used. For example the following are equivalent representations:

$$\frac{\partial \mathbf{f}_v}{\partial \mathbf{u}} = \begin{bmatrix} \frac{\partial z(t+\Delta t)}{\partial v_1} & \frac{\partial z(t+\Delta t)}{\partial v_2} \\ \frac{\partial x(t+\Delta t)}{\partial v_1} & \frac{\partial x(t+\Delta t)}{\partial v_2} \\ \frac{\partial \varphi(t+\Delta t)}{\partial v_1} & \frac{\partial \varphi(t+\Delta t)}{\partial v_2} \end{bmatrix} = \nabla(\mathbf{f}_v)_{\mathbf{u}} = \begin{bmatrix} \nabla z(t+\Delta t)_{v_1} & \nabla z(t+\Delta t)_{v_2} \\ \nabla x(t+\Delta t)_{v_1} & \nabla x(t+\Delta t)_{v_2} \\ \nabla \varphi(t+\Delta t)_{v_1} & \nabla \varphi(t+\Delta t)_{v_2} \end{bmatrix}$$

## *4.2 Camera, Bisight Head, and Mobile-Base Models*

In this section notation and preliminary models used for cameras, 4 DOF Bisight head, and mobile-robot base, will be introduced. The modeling is based on the work of Davison [Dav98] which was designed and implemented for Oxford's (Yorick) stereo head, and GTI mobile-robot. Davison has generously made his work publicly available and in light of possible continuing collaboration (see Wingate & Davison [WD02a], [WD02b]) and benefits of standardisation, the following (and later derivations) relating to MROLR's architecture, have wherever possible used notation and equations consistent with those of Davison.

MROLR's stereo-head appears somewhat different to Oxford's Yorick 8-11 (Fig 4.1.2.1), but geometrically the differences are relatively minor, both having 4 degrees of axial movement. Consequently both can be modeled in a similar way.

The TRC mobile-base used by MROLR differs significantly from that of Oxford's GTI mobile-vehicle. The GTI base is 3 wheeled, with the motor driven rear wheel being responsible for steering. In comparison, the TRC (Fig 4.2.6.2) utilises 2 centre positioned motor driven wheels and 4 corner stabilising casters. An advantage of the TRC's configuration is that rotations about its central axis are possible, in some instances simplifying the "shortest route navigation control" to a rotation followed by a translation followed by a final orientation rotation. In modeling the TRC base, the full range of possible movements will be included.

(a) Bi-Sight head

(b) Oxford's Yorick 8-11 head

**Fig 4.1.2. 1 BiSight and Yorick 8-11 stereo heads**

### 4.2.1 Camera Model

Fig 4.2.1.1 shows the basic geometry of the widely accepted pinhole camera model, used throughout this work. Effectively the pinhole model is an ideal camera model in which the rays from world points to image points pass through the camera's focal point, producing an inverted image on the cameras image plane. For ease of modeling, a virtual image plane may be placed in front of the optic centre (as in Fig 4.2.1.1), resulting in projected upright image. Consider a Scene point **P.**

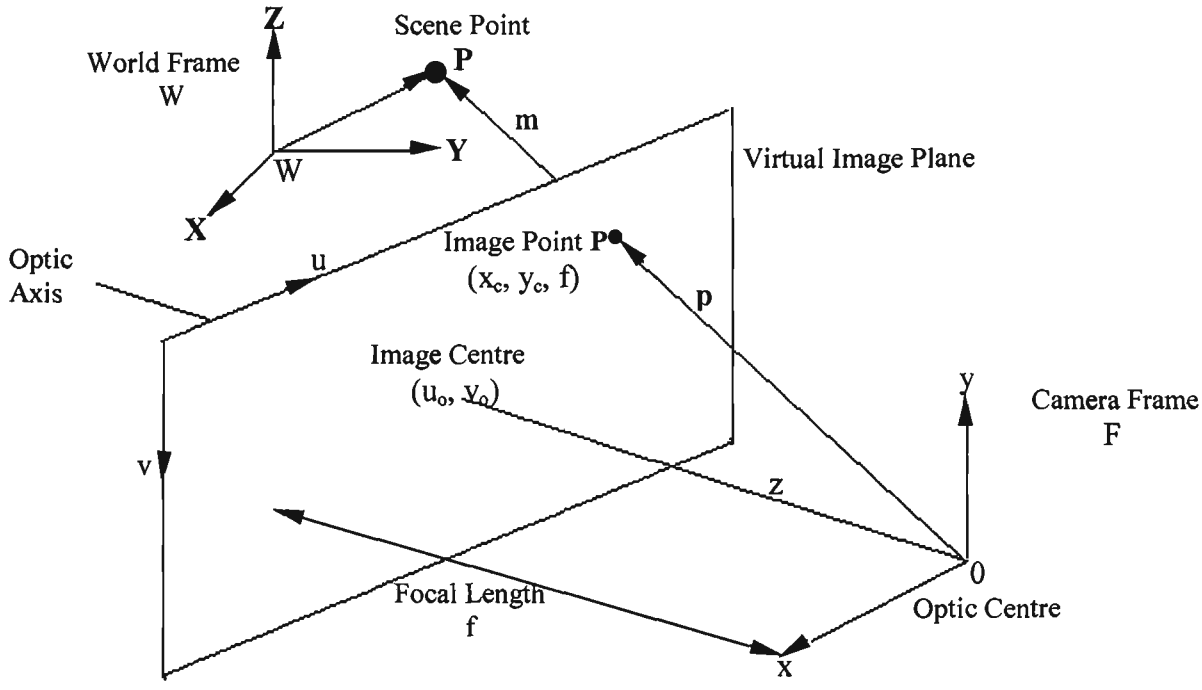



**Fig 4.2.1. 1 Geometry of pinhole camera model**

In world frame W, and camera frame F, this has coordinates

$$\mathbf{P}^W = \begin{pmatrix} P_x^W \\ P_y^W \\ P_z^W \end{pmatrix} \quad \text{and} \quad \mathbf{P}^F = \begin{pmatrix} P_x^F \\ P_y^F \\ P_z^F \end{pmatrix} \quad \text{respectively.}$$

To save case confusion (i.e., $\mathbf{P}^F$ not to be confused with $\mathbf{p}^F$), $\mathbf{m}^F$ will be used in place of $\mathbf{P}^F$, and $\mathbf{p}^F$ to designate the corresponding position vector of the image point ($x_c$, $y_c$, f ) as shown in the above camera model. Thus:

$$\mathbf{P}^F = \begin{pmatrix} P_x^F \\ P_y^F \\ P_z^F \end{pmatrix} = \mathbf{m}^F = \begin{pmatrix} m_x^F \\ m_y^F \\ m_z^F \end{pmatrix}$$

For the pin hole camera model, with the virtual image plane in front of the focal point.

$$\mathbf{p}^F = \begin{pmatrix} x_c \\ y_c \\ f \end{pmatrix} = \frac{f}{m_z^F} \begin{pmatrix} m_x^F \\ m_y^F \\ m_z^F \end{pmatrix} = \frac{f}{m_z^F} \mathbf{m}^F \qquad \textbf{4.1}$$

Light rays impinging on discrete charged couple device (CCD) cells, which make up the camera's image plane, are responsible for the camera's image. This image is mapped to pixels on the computer screen via the computer's frame buffer, and it is from this buffer that the software will normally obtain image information. A point $\mathbf{p} = (x,y,f)^T$ in camera coordinates, projects to computer pixel coordinates of:

$$u = u_0 - k_u x_c \quad , \qquad v = v_0 - k_v y_c \qquad \textbf{4.2}$$

Where horizontal and vertical scaling parameters $k_u$ , $k_v$ accommodate the mapping from the camera's CCD image plane to computer screen via the computer's frame buffer. $k_u$ and $k_v$ represent the "effective" number of horizontal and vertical pixels/metre respectively, and are obtained via the calibration procedure outlined in 4.3.2.1. The pixel position $(u_o,v_o)$ corresponds to where the optic axis passes through the optic centre of the lens. In the ideal case $(u_o,v_o)$ will also correspond to the image centre (i.e., for an image size 600 x 400, $u_o$=300, $v_o$=200 pixels, respectively, and u = 0,v = 0 corresponds to the top left hand corner of the image).

In matrix form equation 4.2 become:

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{bmatrix} -k_u & 0 & u_0/f \\ 0 & -k_v & v_0/f \\ 0 & 0 & 1/f \end{bmatrix} \begin{pmatrix} x_c \\ y_c \\ f \end{pmatrix}$$

or

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{bmatrix} -k_u & 0 & u_0/f \\ 0 & -k_v & v_0/f \\ 0 & 0 & 1/f \end{bmatrix} \mathbf{p}^F \qquad \textbf{4.3}$$

And since vector $\mathbf{m}^F$ is proportional to $\mathbf{p}^F$, it is evident that

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \alpha \begin{bmatrix} -fk_u & 0 & u_0 \\ 0 & -fk_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{m}^F \qquad \textbf{4.4}$$

or

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \alpha \ \mathbf{C}\mathbf{m}^F \qquad \textbf{4.5,} \quad \text{and} \quad \mathbf{m}^F \ \alpha \ \mathbf{C}^{-1} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \qquad \textbf{4.6}$$

where

$$\mathbf{C} = \begin{bmatrix} -fk_u & 0 & u_0 \\ 0 & -fk_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \textbf{4.7} \qquad \mathbf{C}^{-1} = \begin{bmatrix} \frac{-1}{fk_u} & 0 & \frac{u_0}{fk_u} \\ 0 & \frac{-1}{fk_v} & \frac{v_0}{fk_v} \\ 0 & 0 & 1 \end{bmatrix} \qquad \textbf{4.8}$$

Intrinsic camera parameters f, $k_u$, $k_v$, $u_0$ and $v_0$ are normally obtained by a calibration routine involving a tile. The calibration procedure used in this work is described in section 4.3.2. Matrix **C** is referred to as the camera calibration matrix, and contains the camera's intrinsic parameters. Equation 4.5 is the well established perspective projection equation relating a 3D

scene-point, in the camera's reference frame, to the homogeneous vector $\begin{pmatrix} u \\ v \\ 1 \end{pmatrix}$ which describes the pixel location of its image point.

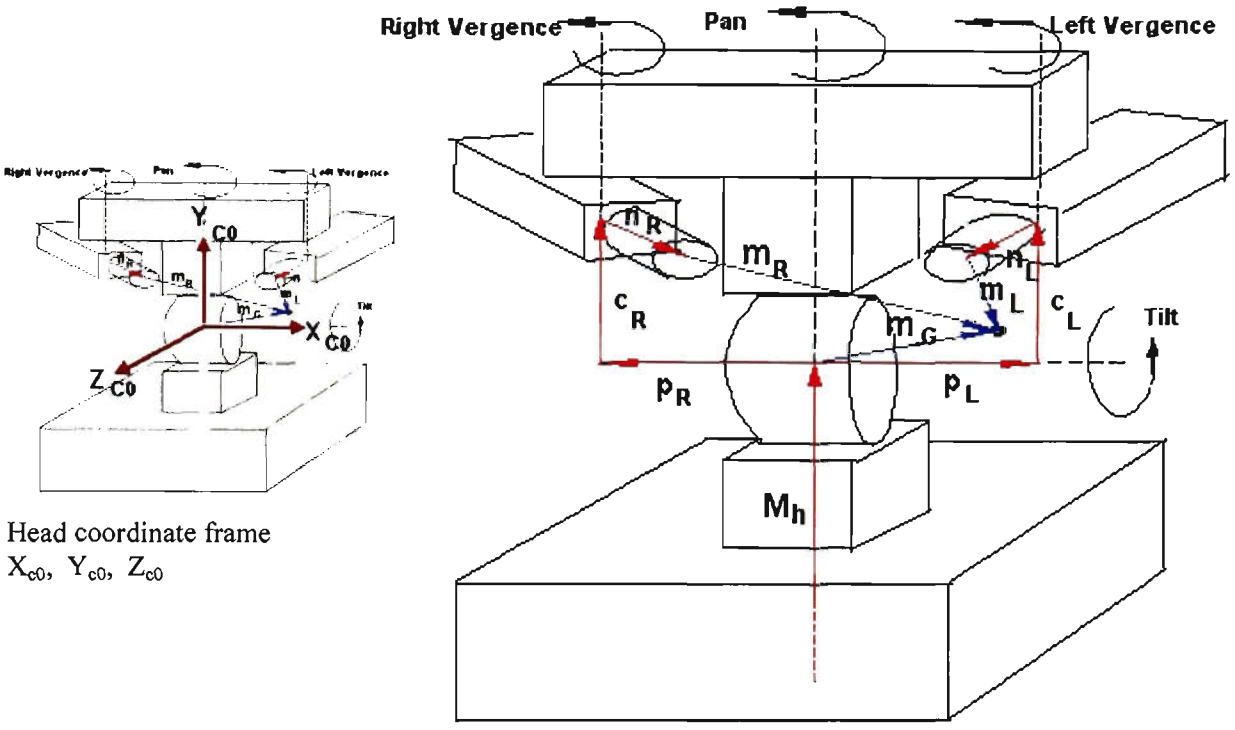
Thus, knowledge of matrix **C** permits the 3D location of point **m** to be determined in the camera's reference frame, up to an unknown scale factor (equation 4.5). That is, it lies somewhere along the light ray that is coincident with vector **m**. To obtain its actual location along the light ray, using a single camera, requires at least two sequential views of the point. Alternatively a pair of cameras may be used in a stereo configuration as formulated in the following section.

### 4.2.2 Stereo Head Model and Frames

The TRC Bisight, four DOF pan-tilt-vergence stereo head, is mounted directly above the central axis of the base. The head is equipped with encoders that provide accurate information on angular movements. A model of this active head is formulated below to facilitate the calculation of 3D Scene points from corresponding, left and right camera image coordinates, and the joint angles necessary for fixation on known 3D Scene feature points.

Each joint motion of the stereo head is initially monitored in an assigned reference frame and defined by an appropriate homogeneous transform. The motion of the active head is obtained from the product of these transforms. The respective reference frames are shown in Fig 4.2.2.2. The head-centre coordinate frame C0 has its origin at the intersection of the pan and the tilt axis. This point remains fixed relative to the mobile base for all head movements. All measurements carried out on the mobile base are referenced with respect to C0, it is the vehicle frame. Its Z-axis lies in the forward direction, with the Y-axis pointing up and the X-axis to the left. This reference frame is also used for the robot arm in obtaining the transforms necessary to guide the gripper for grasping and retrieving of the object. Accurate modeling of the stereo head is thus essential for navigation and object retrieval.

Vectors $\mathbf{M_h}$, $\mathbf{p_L}$, $\mathbf{p_R}$, $\mathbf{c_L}$, $\mathbf{c_R}$, $\mathbf{n_L}$, $\mathbf{n_R}$ represent the head's intrinsic parameters, their scalar magnitudes remaining constant. Subscripts L and R refer to the Left and Right Camera frames respectively. $\mathbf{p_L}$, $\mathbf{c_L}$, $\mathbf{n_L}$ represent the Left Camera offsets from the origin of frame C0, while $\mathbf{p_R}$, $\mathbf{c_R}$, $\mathbf{n_R}$ denote the corresponding offsets for the Right Camera. Vector $\mathbf{m_G}$ represents the location of a Scene feature measured in frame C0. Vectors $\mathbf{m_L}$, and $\mathbf{m_R}$ are vectors from the camera's optical centres. When all head angles are zero, both the right and left camera will be pointing straight ahead and all head axis will be aligned




Vectors $\mathbf{M_h}$, $\mathbf{p_L}$, $\mathbf{p_R}$, $\mathbf{c_L}$, $\mathbf{c_R}$, $\mathbf{n_L}$, $\mathbf{n_R}$, represent the head's intrinsic parameters, their scalar magnitudes remaining constant. Subscripts L and R refer to the left and right camera frames respectively. $\mathbf{m_G}$ a 3D scene vector point.

**Fig 4.2.2. 1 Stereo Head showing Intrinsic Parameters**

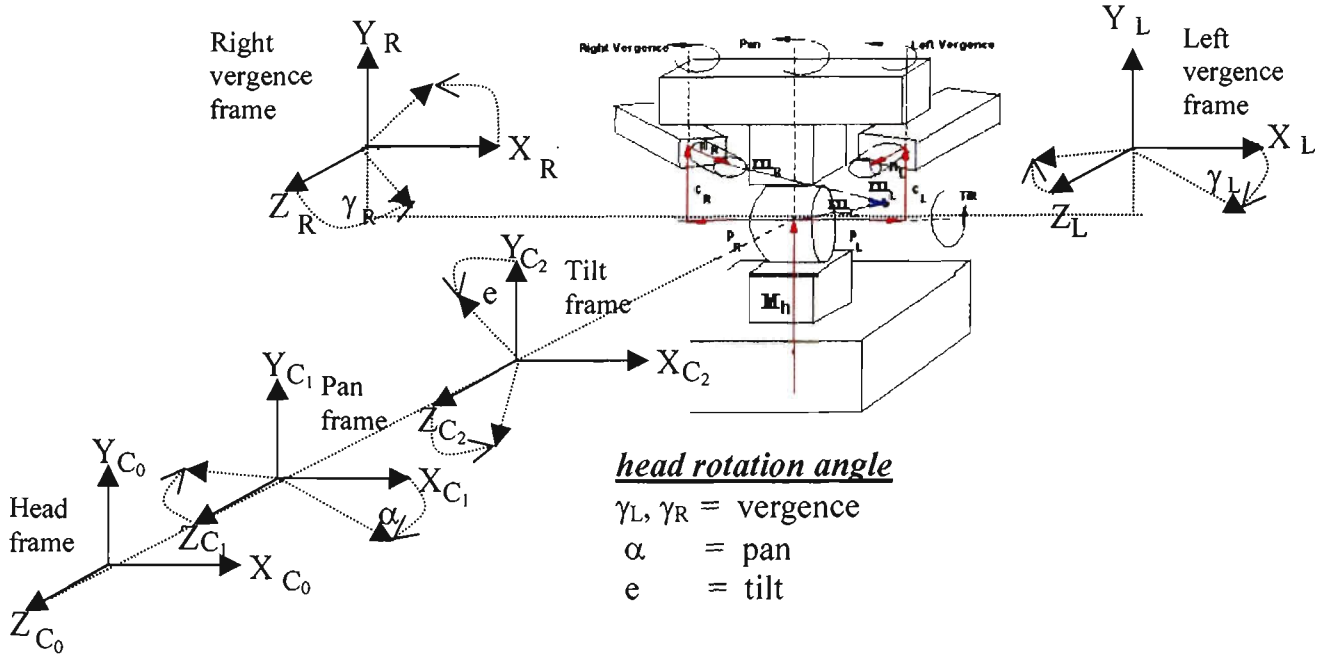



**Fig 4.2.2. 2 Stereo head joint motion reference frames**

#### 4.2.2.1 Mobile Base $M_h$ and Head-frame C0

In this frame

$$\mathbf{M}_h^{C0} = \begin{pmatrix} 0 \\ M_h \\ 0 \end{pmatrix} \quad , \quad \mathbf{m}_G^{C0} = \begin{pmatrix} m_{Gx}^{C0} \\ m_{Gy}^{C0} \\ m_{Gz}^{C0} \end{pmatrix} \qquad \textbf{4.9}$$

$M_h$ is the vertical scalar distance of the head centre from the centre of the mobile base.

#### 4.2.2.2 Pan Frame C1

The pan frame C1 is fixed relative to those parts of the head that move as the head pans around. Its y-axis remains vertical and the corresponding x and z-axes remain in the horizontal plane while the x-axis remains parallel to the head's tilt axis. It is horizontally rotated with respect to C0 by the pan angle $\alpha$.

In this frame:

$$\mathbf{p}_L^{C1} = \begin{pmatrix} I/2 \\ 0 \\ p \end{pmatrix} \quad , \quad \mathbf{p}_R^{C1} = \begin{pmatrix} -I/2 \\ 0 \\ p \end{pmatrix} \qquad \textbf{4.10}$$

Where I is the horizontal distance between the vergence axes and is the inter-ocular distance between the camera optic centres if both cameras are facing forward. $p_L = p_R = p$ are the horizontal scalar offset distances between the intersection of the pan and tilt axes, and respective vergence axis. The respective **p** vectors will rotate with a pan movement.

#### 4.2.2.3 Tilt Frame C2

This frame remains fixed relative to parts of the head that move when the tilt angle changes. Its Y-axis remains parallel to the left and right vergence axes and its X-axis is aligned with the tilt axis. It is vertically rotated with respect to C1 by tilt angle e.

In this frame:

$$\mathbf{c}_L^{C2} = \begin{pmatrix} 0 \\ c \\ 0 \end{pmatrix} \quad , \quad \mathbf{c}_R^{C2} = \begin{pmatrix} 0 \\ c \\ 0 \end{pmatrix} \qquad \textbf{4.11}$$

$c_L = c_R = c$ is the scalar offset along either vergence axis between the intersections with the tilt axis and the camera optic axis

#### 4.2.2.4 Left L and Right R Frames

The left and right camera frames L and R remain fixed relative to the camera's respective optic centres. Their Z-axes are aligned with the cameras optic axes and their X and Y-axes lie parallel to their image planes. These frames are rotated with respect to C2 by vergence angles $\gamma_L$ and $\gamma_R$.

In these frames:

$$\mathbf{n}_L^L = \begin{pmatrix} 0 \\ 0 \\ n \end{pmatrix} \quad , \quad \mathbf{m}_L^L = \begin{pmatrix} m_{Lx}^L \\ m_{Ly}^L \\ m_{Lz}^L \end{pmatrix} \qquad \textbf{4.12}$$

$$\mathbf{n}_R^R = \begin{pmatrix} 0 \\ 0 \\ n \end{pmatrix} \quad , \quad \mathbf{m}_R^R = \begin{pmatrix} m_{Rx}^R \\ m_{Ry}^R \\ m_{Rz}^R \end{pmatrix} \qquad \textbf{4.13}$$

$n_L = n_R = n$ is the scalar offset along either optic axis between the camera optic centre and the intersection with the vergence axis

In summary, vectors $\mathbf{p_L}$, $\mathbf{p_L}$, $\mathbf{c_L}$, and $\mathbf{n_L}$ combine to represent the vector offset from the head centre to the left camera's optic centre. $\mathbf{p_R}$, $\mathbf{p_R}$, $\mathbf{c_R}$, and $\mathbf{n_R}$ represent the same offset for the right camera.

The following rotation matrices permit transformation between the various head-frames.

Note a clockwise rotation is +ve (axis of rotation arrow pointing at observer).

$$\mathbf{R}^{C10} = \begin{bmatrix} \cos\alpha & 0 & -\sin\alpha \\ 0 & 1 & 0 \\ \sin\alpha & 1 & \cos\alpha \end{bmatrix} \qquad \textbf{4.14}$$

$$\mathbf{R}^{C21} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos e & -\sin e \\ 0 & \sin e & \cos e \end{bmatrix} \qquad \textbf{4.15}$$

$$\mathbf{R}^{LC2} = \begin{bmatrix} \cos\gamma_L & 0 & -\sin\gamma_L \\ 0 & 1 & 0 \\ \sin\gamma_L & 1 & \cos\gamma_L \end{bmatrix} \qquad \textbf{4.16}$$

$$\mathbf{R}^{RC2} = \begin{bmatrix} \cos\gamma_R & 0 & -\sin\gamma_R \\ 0 & 1 & 0 \\ \sin\gamma_R & 1 & \cos\gamma_R \end{bmatrix} \qquad \textbf{4.17}$$

### 4.2.3 Calculation of Image Coordinates from a Known 3D Feature Point

With the above framework and reference to Figure 4.2.1.1 it is apparent that:

$$\mathbf{m}_G = \mathbf{p}_L + \mathbf{c}_L + \mathbf{n}_L + \mathbf{m}_L \quad 4.18$$

$$m_G = p_R + c_R + n_R + m_R \quad 4.19$$

Assume for the moment that $\mathbf{m}_G^{C0}$ the 3D position of a point in the vehicle reference frame C0 is known and visible in both cameras.
From equation 4.18 in the left camera frame L:

$$\mathbf{m}_L^L = \mathbf{m}_G^L - \mathbf{p}_L^L - \mathbf{m}_L^L - \mathbf{c}_L^L - \mathbf{n}_L^L \quad 4.20$$

Or in the desired frames:

$$\mathbf{m}_L^L = \mathbf{R}^{LC0}\mathbf{m}_G^{C0} - \mathbf{R}^{LC1}\mathbf{p}_L^L - \mathbf{R}^{LC2}\mathbf{c}_L^L - \mathbf{n}_L^L \quad 4.21$$

Rearranging the transformations into relevant simple components,

$$\mathbf{m}_L^L = \mathbf{R}^{LC2}(\mathbf{R}^{C21}(\mathbf{R}^{C10}\mathbf{m}_G^{C0} - \mathbf{p}_L^{C1}) - \mathbf{c}_L^{C2}) - \mathbf{n}_L^L \quad 4.22$$

Similarly for the right camera:

$$\mathbf{m}_R^R = \mathbf{R}^{RC2}(\mathbf{R}^{C21}(\mathbf{R}^{C10}\mathbf{m}_G^{C0} - \mathbf{p}_R^{C1}) - \mathbf{c}_R^{C2}) - \mathbf{n}_R^R \quad 4.23$$

Equations 4.22 and 4.23 allow the vectors from the left and right optic centres to the scene point in the respective camera coordinate frames to be calculated. Furthermore, equation 4.5 (applied to the left and right camera, see equation 4.24 below) permits the point's image coordinates to be determined. Further details are given in section 4.2.5.

$$\begin{pmatrix} u_L \\ v_L \\ 1 \end{pmatrix} \alpha \quad \mathbf{C}\mathbf{m}_L^L \quad , \begin{pmatrix} u_R \\ v_R \\ 1 \end{pmatrix} \alpha \quad \mathbf{C}\mathbf{m}_R^R \quad 4.24$$

### 4.2.4 Head Angles to Fixate a Known Point

A frequent task required during navigation and map building is to fixate on a known 3D point. This consists of knowing $\mathbf{m}_G^{C0}$ relative to the head-frame, and driving the head so the image of the 3D point passes through $u_0$, $v_0$ the principal point, in both left and right cameras. To simply the task, vergence angles are assigned to be equal and opposite. This form of symmetric fixation is further simplified by turning the pan axis to view the scene point such that $\mathbf{m}_G$ in pan frame C1, lies in vertical Y, Z plane.

In this state:

$$\mathbf{m}_{Gx}^{C1} = 0$$

Furthermore:

$$\mathbf{m}_G^{C1} = \mathbf{R}^{C10}\mathbf{m}_G^{C0} \quad \textbf{4.25}$$

i.e.
$$\begin{bmatrix} cos\,\alpha & 0 & -sin\alpha \\ 0 & 1 & 0 \\ sin\alpha & 1 & cos\alpha \end{bmatrix} \begin{pmatrix} m_{Gx}^{C1} \\ m_{Gy}^{C1} \\ m_{Gz}^{C1} \end{pmatrix} = \begin{pmatrix} m_{Gx}^{C0} \\ m_{Gy}^{C0} \\ m_{Gz}^{C0} \end{pmatrix} \quad \textbf{4.26}$$

Solving the top line of this equation for α gives

$$\alpha = tan^{-1}\frac{m_{Gx}^{C0}}{m_{Gz}^{C0}} \quad \textbf{4.27}$$

To find the remaining frame angles e, $\gamma_L$, $\gamma_R$, use is made of the fact that as the optic axes of both cameras ideally pass through the scene 3D point, therefore $\mathbf{m}_L^L$ and $\mathbf{m}_R^R$ are both non zero in the z coordinate:

$$m_{Lx}^L = m_{Ly}^L = m_{Rx}^R = m_{Ry}^R = 0 \quad \textbf{4.28}$$

Writing equation 4.18 in frame L:

$$\mathbf{m}_L^L = \mathbf{R}^{LC2}(\mathbf{R}^{C21}(\mathbf{R}^{C10}\mathbf{m}_G^{C0} - \mathbf{p}_L^{C1}) - \mathbf{c}_L^{C2}) - \mathbf{n}_L^L \quad \textbf{4.29}$$

Solving for e by setting $m_{Ly}^L = 0$ yields:

$$e = tan^{-1}\frac{m_{Gy}^{C1}}{m_{Gz}^{C1} - p} - sin^{-1}\frac{c}{\sqrt{(m_{Gy}^{C1})^2 + (m_{Gz}^{C1} - p)^2}} \quad \textbf{4.30}$$

Note the components of $\mathbf{m}_G^{C1}$ are determined from known α. The remaining vergence angles γ are found by using e to form $\mathbf{m}_L^{C2}$

$$\mathbf{m}_L^L = \mathbf{R}^{LC2}\mathbf{m}_L^{C2} \quad \textbf{4.31}$$

Setting $m_{Lx}^L = 0$, $\gamma_L$ is solved from:

$$\gamma_L = tan^{-1}\frac{m_{Lx}^{C2}}{m_{Lz}^{C2}} \quad \textbf{4.32}$$

And finally $\gamma_R$ obtained from:

$$\gamma_R = -\gamma_L \quad \textbf{4.33}$$

### 4.2.5 3D Point Location from Image Coordinates $u_L$, $v_L$, $u_R$, $v_R$

Given the left and right image coordinates $u_L$, $v_L$, $u_R$, $v_R$ of a scene point, its 3D location is determined as follows:

Choosing the head-frame C0 from equation 4.18 ,

$$\mathbf{m}_G^{C0} = \mathbf{p}_L^{C0} + \mathbf{c}_L^{C0} + \mathbf{n}_L^{C0} + \mathbf{m}_L^{C0} \qquad 4.34$$

and rearranging this by transforming the vectors into their desired frames,

$$\mathbf{m}_G^{C0} = \mathbf{R}^{C01}\mathbf{p}_L^{C1} + \mathbf{R}^{C02}\mathbf{c}_L^{C2} + \mathbf{R}^{C0L}\mathbf{n}_L^{L} + \mathbf{R}^{C0L}\mathbf{m}_L^{L} \qquad 4.35$$

then regrouping:

$$\mathbf{m}_G^{C0} = \mathbf{R}^{C01}(\mathbf{p}_L^{C1} + \mathbf{R}^{C12}(\mathbf{c}_L^{C2} + \mathbf{R}^{C2L}\mathbf{n}_L^{L})) + \mathbf{R}^{C01}\mathbf{R}^{C12}\mathbf{R}^{C2L}\mathbf{m}_L^{L} \qquad 4.36$$

Similarly for the right Camera:

$$\mathbf{m}_G^{C0} = \mathbf{R}^{C01}(\mathbf{p}_R^{C1} + \mathbf{R}^{C12}(\mathbf{c}_R^{C2} + \mathbf{R}^{C2R}\mathbf{n}_R^{R})) + \mathbf{R}^{C01}\mathbf{R}^{C12}\mathbf{R}^{C2R}\mathbf{m}_R^{R} \qquad 4.37$$

While these give two expressions for $\mathbf{m}_G^{C0}$ , neither is sufficient as $\mathbf{m}_L^{L}$ and $\mathbf{m}_R^{R}$ are calculable only up to an unknown scale factor by inverting equations (4.24) i.e. rewriting equations 4.36 and 4.37 in the form:

$$\mathbf{m}_L^{L} \alpha\, C^{-1}\begin{pmatrix} u_L \\ v_L \\ 1 \end{pmatrix} \quad , \quad \mathbf{m}_R^{R} \alpha\, C^{-1}\begin{pmatrix} u_R \\ v_R \\ 1 \end{pmatrix} \qquad 4.38$$

$$\mathbf{m}_G^{C0} = \mathbf{a}^{C0} + \lambda\mathbf{b}^{C0} \qquad 4.39$$

$$\mathbf{m}_G^{C0} = \mathbf{c}^{C0} + \mu\mathbf{d}^{C0} \qquad 4.40$$

Where

$$\mathbf{a}^{C0} = \mathbf{R}^{C01}(\mathbf{p}_L^{C1} + \mathbf{R}^{C12}(\mathbf{c}_L^{C2} + \mathbf{R}^{C2L}\mathbf{n}_L^{L})) \qquad 4.41$$

$$\mathbf{c}^{C0} = \mathbf{R}^{C01}(\mathbf{p}_R^{C1} + \mathbf{R}^{C12}(\mathbf{c}_R^{C2} + \mathbf{R}^{C2R}\mathbf{n}_R^{R})) \qquad 4.42$$

$$\mathbf{b}^{C0} = \mathbf{R}^{C01}\mathbf{R}^{C12}\mathbf{R}^{C2L}\mathbf{C}^{-1}\begin{pmatrix} u_L \\ v_L \\ 1 \end{pmatrix} \qquad 4.43$$

$$\mathbf{d}^{C0} = \mathbf{R}^{C01}\mathbf{R}^{C12}\mathbf{R}^{C2R}\mathbf{C}^{-1}\begin{pmatrix} u_L \\ v_R \\ 1 \end{pmatrix} \qquad 4.44$$

Where $\lambda$ and $\mu$ are the unknown scale factors. Effectively each camera contains a ray along which the scene point must lie. Ideally the scene point will be at the intersection of these two rays. In practice, because of errors the rays may not intersect and consequently the best estimate

of the scene point will be the mid-point of the rays closest approach to each other. That is, the mid-point of the vector normal to both rays (Figure 4.2.5.1).

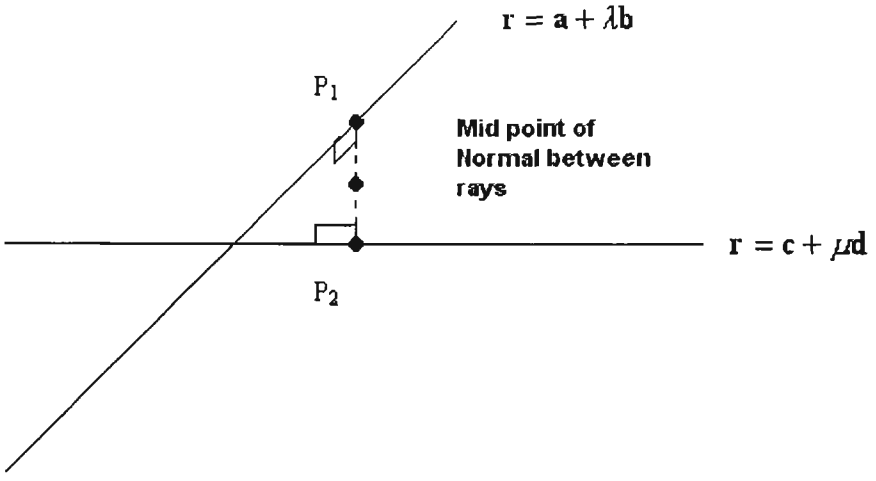



**Fig 4.2.5. 1 Mid-point of normal between 2 rays**

The coordinates of the mid point of the normal between these two camera rays is readily obtainable in terms of vectors **a b c** and **d** by calculating $\lambda$ and $\mu$ for points $P_1$ and $P_2$.. A method for calculating these follows.

**Common perpendicular between two rays and location of mid point**

given $\{\mathbf{a}^{C0}\}$ = known location on ray 1
$\{\hat{\mathbf{u}}_1\}$ = known unit direction of ray 1
$\{\mathbf{c}^{C0}\}$ = known location on ray 2
$\{\hat{\mathbf{u}}_2\}$ = known unit direction of ray 2

Note enclosing {} brackets imply column vectors.

Calculating unit directional vectors:

$$\{\hat{\mathbf{u}}_1\} = \mathbf{b}^{C0} / (_{norm}(\mathbf{b}^{C0})) \qquad 4.45$$

$$\{\hat{\mathbf{u}}_2\} = \mathbf{d}^{C0} / (_{norm}(\mathbf{d}^{C0})) \qquad 4.46$$

$\psi$ = angle between rays $\quad \sin\psi = \text{norm}(\{\hat{\mathbf{u}}_1\} \times \{\hat{\mathbf{u}}_2\}) = \text{norm}([\tilde{\mathbf{u}}_1]\{\hat{\mathbf{u}}_2\})$
$\{\hat{\mathbf{u}}_3\}$ = unit direction along common perpendicular $\{\hat{\mathbf{u}}_3\} = [\tilde{\mathbf{u}}_1]\{\hat{\mathbf{u}}_2\} / \sin\psi$
$d_{12}$ = directed distance (length of common perpendicular) along $\{\hat{\mathbf{u}}_3\}$ from ray 1 to ray 2

$$\{\mathbf{a}^{C0}\} + \lambda\{\hat{\mathbf{u}}_1\} + d\{\hat{\mathbf{u}}_3\} = \{\mathbf{c}^{C0}\} + \mu\{\hat{\mathbf{u}}_2\} \qquad 4.47$$

$$[\{\hat{\mathbf{u}}_1\} \quad \{-\hat{\mathbf{u}}_2\} \quad \{\hat{\mathbf{u}}_3\}] \begin{Bmatrix} \lambda \\ \mu \\ d_{12} \end{Bmatrix} = (\{\mathbf{c}^{C0}\} - \{\mathbf{a}^{C0}\}) \qquad 4.48$$

$$\begin{Bmatrix} \lambda \\ \mu \\ d_{12} \end{Bmatrix} = \left[\{\hat{\mathbf{u}}_1\} \quad \{-\hat{\mathbf{u}}_2\} \quad \{\hat{\mathbf{u}}_3\}\right]^{-1} \quad \left(\{\mathbf{c}^{C0}\} - \{\mathbf{a}^{C0}\}\right) \qquad \textbf{4.49}$$

or

find midpoint of common perpendicular using confluence of 2 rays.

$$\{\, \mathbf{m}_G^{C0} \,\} = \left( [\tilde{\mathbf{u}}_1]^2 + [\tilde{\mathbf{u}}_2]^2 \right)^{-1} \left( [\tilde{\mathbf{u}}_1]^2 \{\mathbf{a}^{C0}\} + [\tilde{\mathbf{u}}_2]^2 \{\mathbf{c}^{C0}\} \right) \qquad \textbf{4.50}$$

**Note**

$[\tilde{\mathbf{u}}]$ = skew-symmetric matrix for $\{\hat{\mathbf{u}}\}$

$$\{\hat{\mathbf{u}}\} = \begin{Bmatrix} u_x \\ u_y \\ u_z \end{Bmatrix} \qquad [\tilde{\mathbf{u}}] = \begin{bmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{bmatrix} \qquad [\tilde{\mathbf{u}}]^T = -[\tilde{\mathbf{u}}]$$

$$[\tilde{\mathbf{u}}]^2 = \{\hat{\mathbf{u}}\}\{\hat{\mathbf{u}}\}^T - [\mathbf{I}_3] = \begin{bmatrix} u_x^2 - 1 & u_x u_y & u_x u_z \\ u_x u_y & u_y^2 - 1 & u_y u_z \\ u_x u_z & u_y u_z & u_z^2 - 1 \end{bmatrix}$$

### 4.2.6 Mobile Base Model

The configuration differences between Oxford's GTI mobile-base and MROLR's base mentioned in Section 4.10, amount to a difference in the control variables. The 3 wheeled GTI base is steered via its economical single motor driven rear wheel. Consequently the control inputs are the velocity and rear wheel relative angle. By contrast MROLR's base is controlled by 2 motor driven wheels as shown in Fig 4.2.6.3. Optical encoders provide feedback for control and odometry information. This has the advantage that by setting $v_1 = -v_2$ the robot is capable of rotating on the spot about its own central axis. In addition, shortest path movements are simple to achieve by an initial on the spot rotation, followed by a linear translation to the desired location and then a final on the spot rotation to end up in the required orientation. The analysis begins by defining the angular velocity of motion $\omega$ in terms of the vehicles rate of change of angular position (see Figs 4.2.6.1/2).

$$\omega = \frac{d\theta}{dt} \qquad \textbf{4. 51}$$

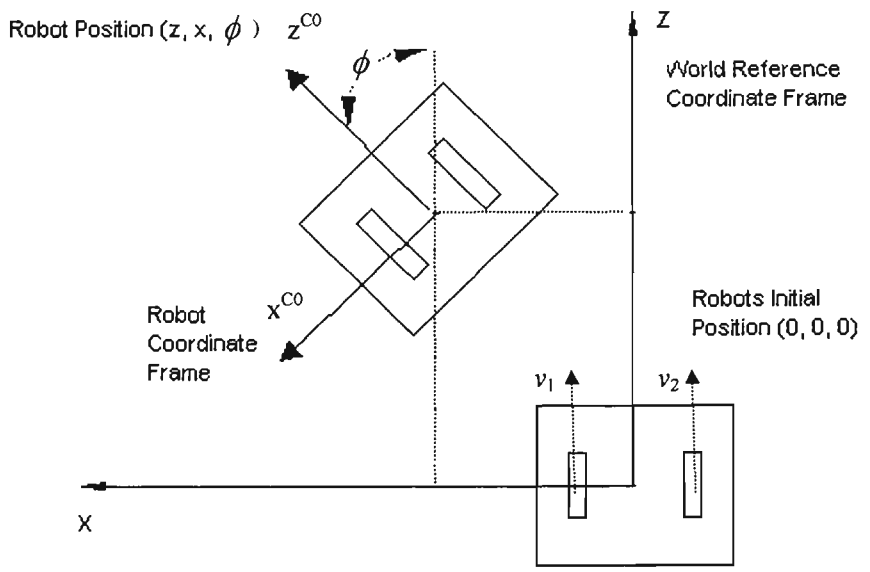



**Fig 4.2.6. 1 MROLR moved to position (z,x ϕ)**

Assuming inner wheel and outer wheel tangential velocities of $v_1$ and $v_2$ metres/sec respectively, the centre vehicle velocity $v$ can be obtained from:

$$v = \frac{v_1 + v_2}{2} = \omega R \qquad \textbf{4. 52}$$

For a given angular velocity ω of the centre of the base, the tangential velocity of each wheel is related via $v_1=\omega(R+D_1)$, and $v_2=\omega(R-D_1)$, the radius of the arc moved through by the base can be derived as follows:

The arc length moved through in time $t$ seconds is $R\theta$, thus

$$R\frac{d\theta}{dt} = (\frac{v_1 + v_2}{2}) \qquad \textbf{4. 53}$$

Solving for R from $v_1=\omega(R+D_1)$/ $v_2=\omega(R-D_1)$ gives:

$$R = \frac{D_1(v_1 + v_2)}{(v_1 - v_2)} \qquad \textbf{4. 54}$$

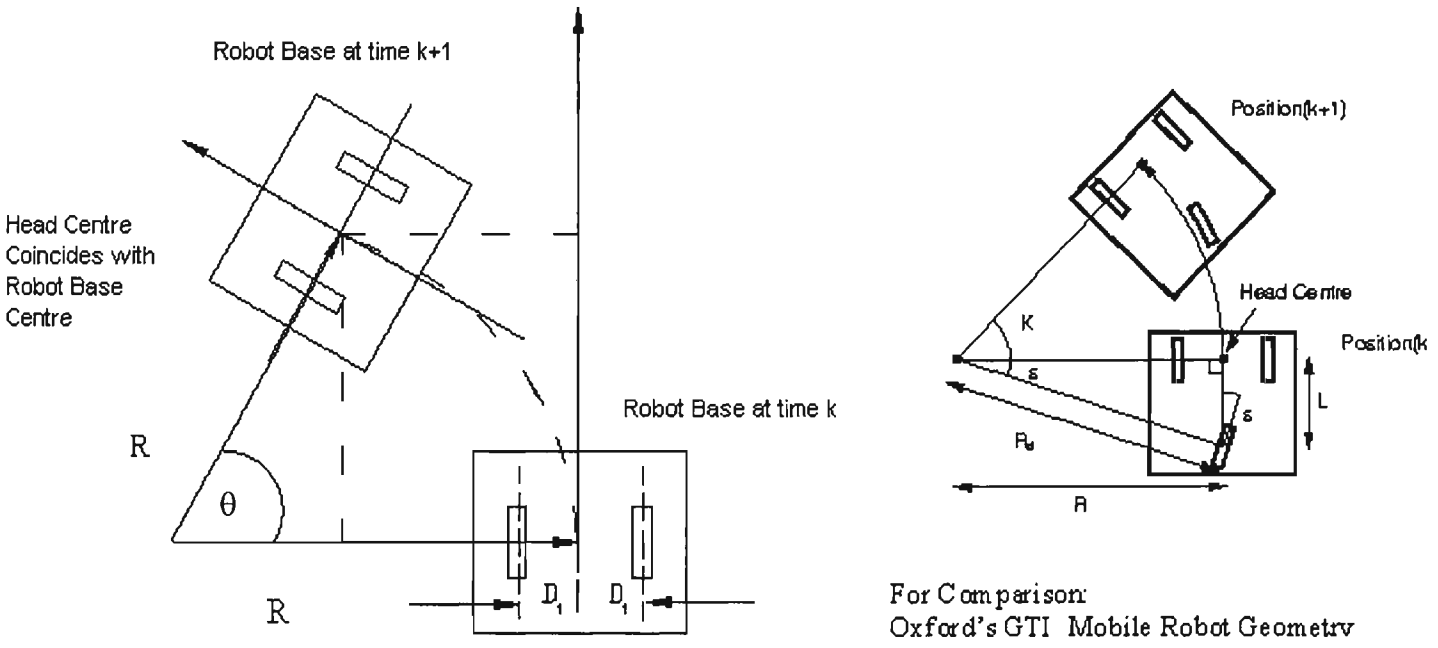



**Fig 4.2.6. 2 Oxford's mobile robot geometry**

Using equation 4.54 to replace R in equation 4.53 and using a time increment of $\Delta t$, leads to

$$\Delta\theta = \frac{v_1 - v_2}{2D_1}\Delta t$$

For a constant time interval Δt the change in $\phi$, will be represented by $\theta$ in place of $\Delta\theta$

i.e., $$\theta = \frac{v_1 - v_2}{2D_1}\Delta t \qquad \textbf{4. 55}$$

Consider a circular arc trajectory increase of θ degrees (Fig 4.2.6.3), the changes in z, x, ϕ are readily seen to be :

$z(t+\Delta t) = z + R( \sin (\phi (t)+\theta ) - \sin(\phi (t)) )$

or rearranging

$$z(t+\Delta t) = z(t) + R(\cos\phi(t)\sin\theta + \sin\phi(t)(\cos\theta-1) ) \quad \textbf{4. 56}$$

$x(t+\Delta t) = x + R(\cos (\phi( (t) - \cos (\phi (t) +\theta) )$

or rearranging gives

$$x(t+\Delta t) = x(t) + R(\sin\phi(t)\sin\theta + \cos\phi(t)(1-\cos\theta)) \quad \textbf{4. 57}$$

and

$$\phi(t)(t+\Delta t) = \phi(t) + \theta \quad \textbf{4. 58}$$

When $v_1 = v_2 = v$ (i.e., straight line case) θ limits towards 0 and R limits towards ∞, the equations may be rewritten in the form:

$$z(t+\Delta t) = z(t) + v\Delta t\cos\phi(t) \quad \textbf{4. 59}$$

$$x(t+\Delta t) = x(t) + v\Delta t\sin\phi(t) \quad \textbf{4. 60}$$

$$\phi(t+\Delta t) = \phi(t) \quad \textbf{4. 61}$$

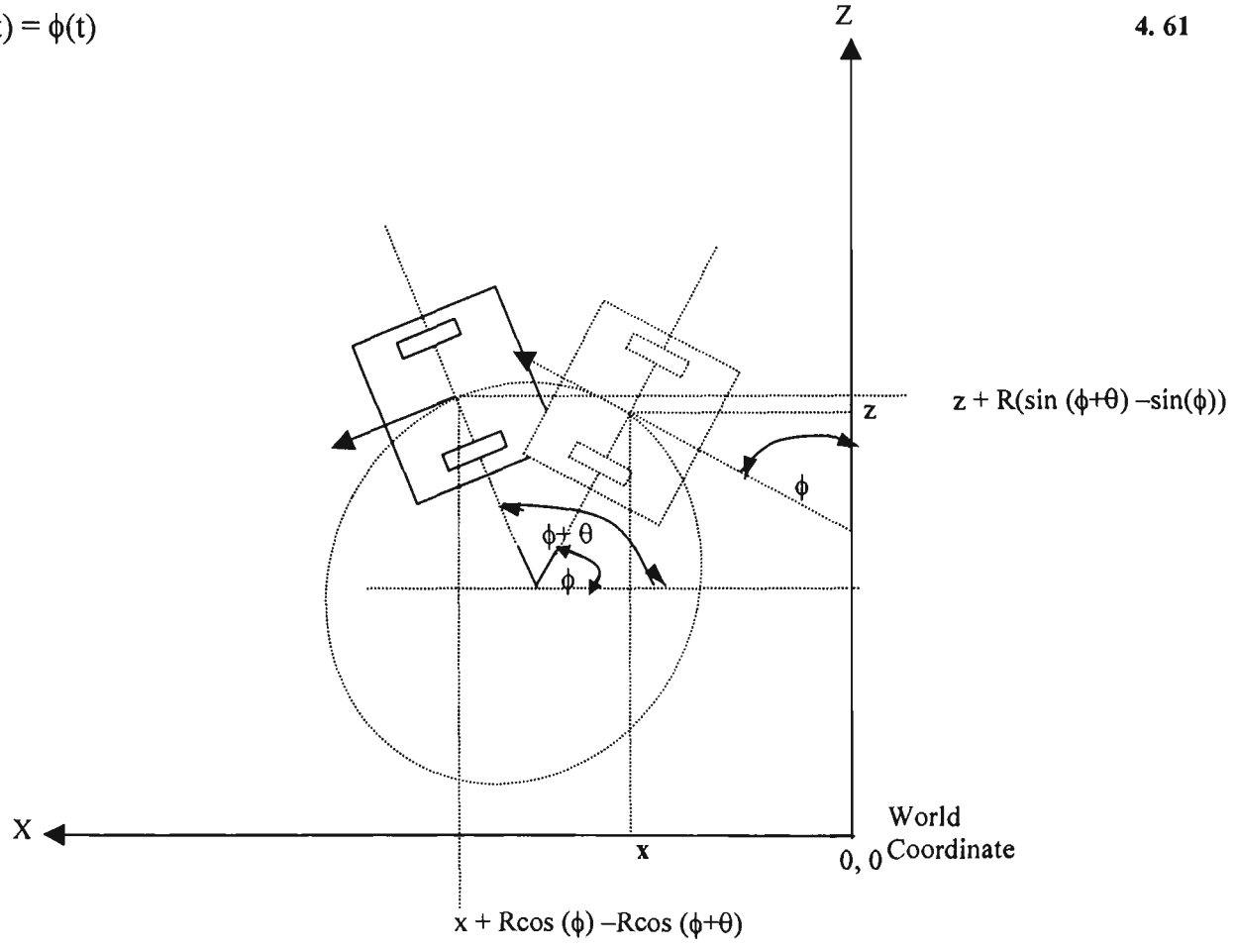



**Fig 4.2.6. 3 Geometry of a circular arc trajectory**

In general the position of the robot base can only be estimated to a degree of uncertainty with the use of the above equations. This is because:
(1) Wheel slippage may occur due to uneven terrain.
(2) The controller will have some calibration error both in terms of distance movement and velocity.
(3) Wheel diameter differences will exist due to tyre pressure variations.
(4) A collision with an obstacle may have occurred.

Additional positional information to that available from wheel odometry is thus essential for reliable robot localisation estimates. In this work additional positional information is obtained with the use of active stereo vision to acquire landmark visual features, which in turn are used for map-building of the local environment, and also self-localisation. The availability of a map is a useful tool when navigating in a previously explored environment, and can considerably reduce the risk of becoming lost.

Landmark visual-feature acquisition, map building and navigation are covered in Chapter 5. Of paramount importance to this work is the ability to identify and accurately locate a target object. Identification is reliant on having accurate model descriptions of the object so that matching can be effected. Two models are used, a 3D object-model created from the integration of multiple views of the object, and a 3D scene-model created from only a single view of the scene (sometimes referred to as a 2.5-D model). In the following tools and algorithms required for building these models are described.

## *4.3 3D Object and Scene Modeling and Recognition*

As outlined in Section 3.2, TINA, Sheffield's Machine Vision Research platform with its comprehensive suite of libraries and tools was adopted as the software environment for developing the model building and recognition modules used by MROLR. Other competing image processing environments such as TargetJr/IUE [TargetJr], Khoros [KHOROS], and Horatio [HORATIO] were also considered as potential platforms, but ultimately were judged not as convenient to use, or in other instances not chosen because of their lateness in arrival on the scene. A view of the TINA main interface module window is displayed in Fig 4.3.1 It will be noted that as typical in Windows type program environments, the software is essentially event driven, with the events consisting of mouse button presses. This is very useful when studying isolated sections of work or specific options, but not conducive to a stand-alone system, requiring essentially only one "initialising" mouse click. A considerable portion of the effort behind this work was in writing and integrating all of the modules (incorporating three different software platforms) to obtain an autonomous system controlled by input events arising from MROLR's physical environment.

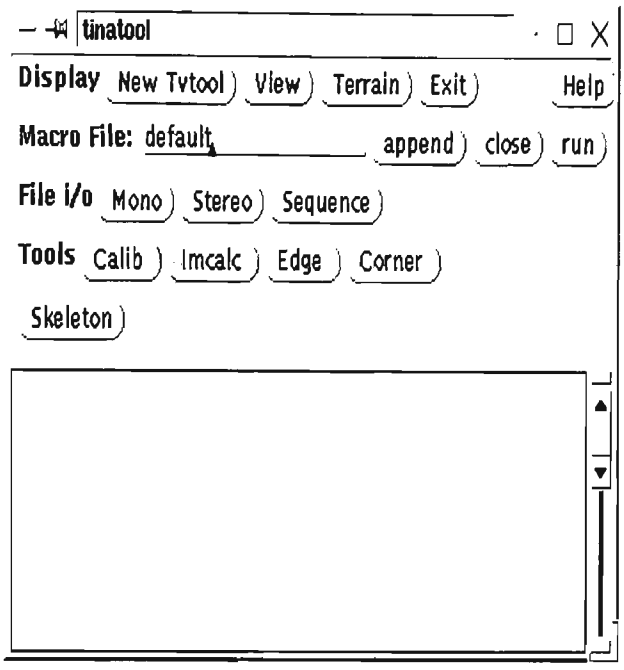
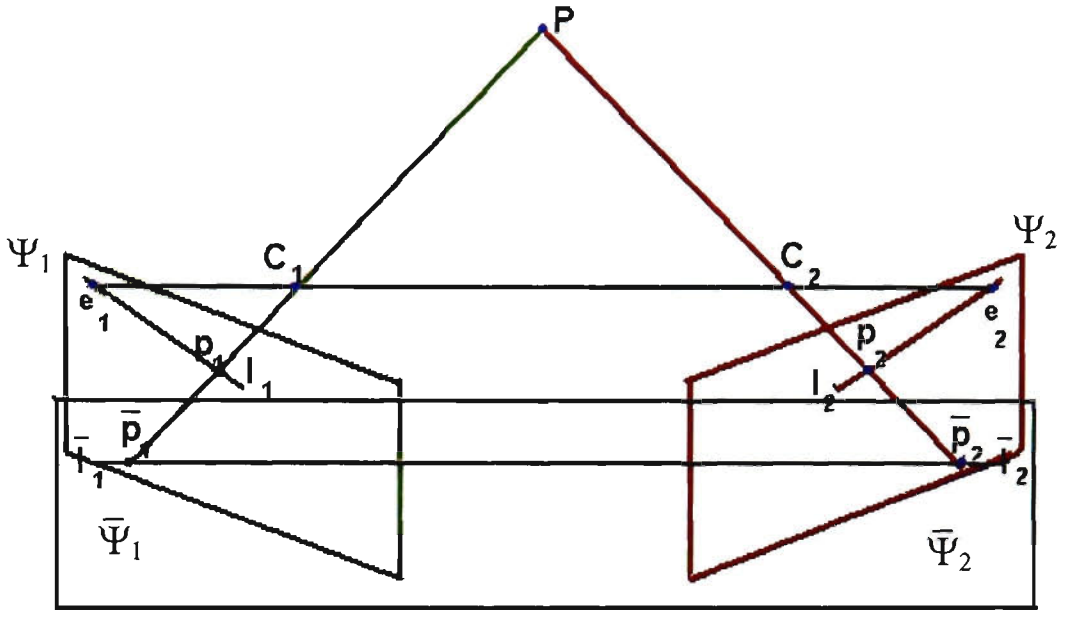



**Fig 4.3. 1 TINA's main interface window**

Prior to outlining the modules/sub-tools utilised, the relevant algorithmic and programming approaches incorporated in TINA will be briefly reviewed. Some of these descriptions are summaries of work in [Tin97a], [Tin97b], [Tin99]. It should be pointed out that much of the following is now well understood and has been incorporated in numerous machine vision applications for some time. This specifically refers to camera modeling and calibration, epipolar geometry, image rectification, edge-extraction, stereo matching, depth recovery, disparity gradient constraints, 3D model-creation and the like. In light of this, the following descriptions are intentionally brief; their inclusion here is purely for the sake of completeness.

### 4.3.1 Parallel Camera Geometry For Non-Parallel Stereo

Using parallel camera geometry (i.e., parallel optical axes, and image planes constrained to be in the same plane) can considerably simplify both stereo correspondences matching problems, and the subsequent 3D reconstruction. This simplification arises as the epipolar lines of images are parallel to the baseline, and disparity analysis can now be restricted to a one-dimensional search along these horizontal lines. If calibrated cameras are used, it is possible to construct an equivalent (virtual) parallel camera geometry by a process of epipolar rectification (refer Fig 4.3.1.1). The rectification process can be implemented by projecting the original images onto the new image plane such that pairs of conjugate epipolar lines become collinear and parallel to one of the image axes (usually the horizontal one). The rectified images can be thought of as acquired by a new stereo rig, obtained by rotating the original cameras. (See for examples, [Aya91], [PPPMF88], [FP02] ).




Stereo rectification: original image planes $\Psi_1$ $\Psi_2$ and associated points $\mathbf{p}_1$ and $\mathbf{p}_2$ re-projected onto a common plane $\bar{\Psi}_1, \bar{\Psi}_2$ parallel to base line $C_1$--$C_2$. Note the epipolar lines $\mathbf{l}_1, \mathbf{l}_2$ are re-projected as the common scanline $\bar{\mathbf{l}}_1, \bar{\mathbf{l}}_2$ and are also parallel to the baseline.

**Fig 4.3.1. 1 Convergent to rectified stereo re-projection**

At the additional cost and associated overheads of a third camera (i.e., trinocular vision)) or even a fourth camera, it is possible to eliminate the uncertainty associated with a one-dimensional search. See for example Ayache and others [Aya91], [Jar94].[FP02] ). Early trials using trinocular vision were evaluated (Wingate [Win99], also Appendix A) but later abandoned in favour of binocular stereo for computational cost reasons. An interesting comparative study on disparity analysis based on convergent (i.e., non-rectified) and rectified stereo was carried out by Schreer et al [SBK00]. They concluded that each method had additional overhead costs that needed to be considered prior to deciding whether the advantages of implementing rectification outweighed these computational costs.

### 4.3.2 Modelling and Recognition Software Environment and Associated Tools

A compelling reason for using the TINA Machine Vision Research Environment is the diverse range of software tools provided. The following modules were found to be of particular value:

(i) Calibration Tool
(ii) Edge Tool
(iii) Stereo Tool
(iv) Matcher Tool

Brief details of these tools are discussed in the following sections, greater details are available in [Tin97a], [Tin99] from which much of this description arises. Details of supplementation to these tools, which is considered an original contribution of this work is outlined in Ch5.

#### 4.3.2.1 The Calibration Tool.

This tool is provided for estimating the intrinsic and extrinsic camera parameters of a stereo vision system, (the widely applied pin hole camera model is used). A calibration tile is utilised and several alternative calibration algorithms are available. These are briefly mentioned below. Arbitrary parameter selection and adjustment, radial distortion parameters, estimation of calibration parameter covariances, for purposes of optimal combination, are features also provided.

**Calibration Methods**
The following calibration procedures are included:
**Tsai**. This widely used routine, first developed by R.Tsai [Tsa87] is initially used on each camera, and while returning satisfactory parameter estimations, improved results may be obtained by additional processing (i.e., following Tsai with IP min etc).

**IP min**. This is a direct image plane minimisation routine that operates on the difference between the observed and predicted locations of selected data. An iterative minimization algorithm relying on initial estimate values is used. Left and right camera calibration is performed separately.

**IS min**. This routine also performs direct image plane minimisation of the difference between the observed and predicted locations of selected data. Left and right cameras are calibrated simultaneously, unlike the routine above.

**EPI min**. This routine performs an off epipolar minimization. The routine returns the left and right camera models that are most consistent with the observed stereo geometry. A limitation of the routine is that only information on the ratio of the two focal lengths and the relative transformation between the camera coordinate systems up to a scale factor is possible. Furthermore, a covariance estimate from a previous calibration must be used to constrain the minimization process. Additional supporting covariance estimation routines are also provided with the tool.

The procedure for calibration of a stereo rig requires a calibration tile to be placed in front of the two cameras, inclined at approximately 30 degrees to the cameras focal axis, and fully visible in both images. These images are then pre-processed with the Canny operator (Edge Tool). Using an initial estimate of the pin hole camera model, the Tsai algorithm is used to obtain an improved estimate. Following this calibration, residual distributions can be examined and re-calibration applied as necessary, using any of the above routines until good results are obtained.

Fig 4.3.2.1 illustrates the process of calibrating the stereo cameras using a planar tile. Accurate location and matching of all corners is essential for good parameter estimation. The "adequacy" of results can be visualised by performing a 3D reconstruction of the tile itself.

#### 4.3.2.2 Stereo Tool

The function of the stereo tool is essentially input/output generic file manipulation and display functions for stereo image data and derived geometric features. It controls 3 display tools, one for each of the left image, right image and resultant 3D stereo matched line/conic (i.e., wire frame) data obtained following processing with the Edge Tool.

#### 4.3.2.3 Edge Tool

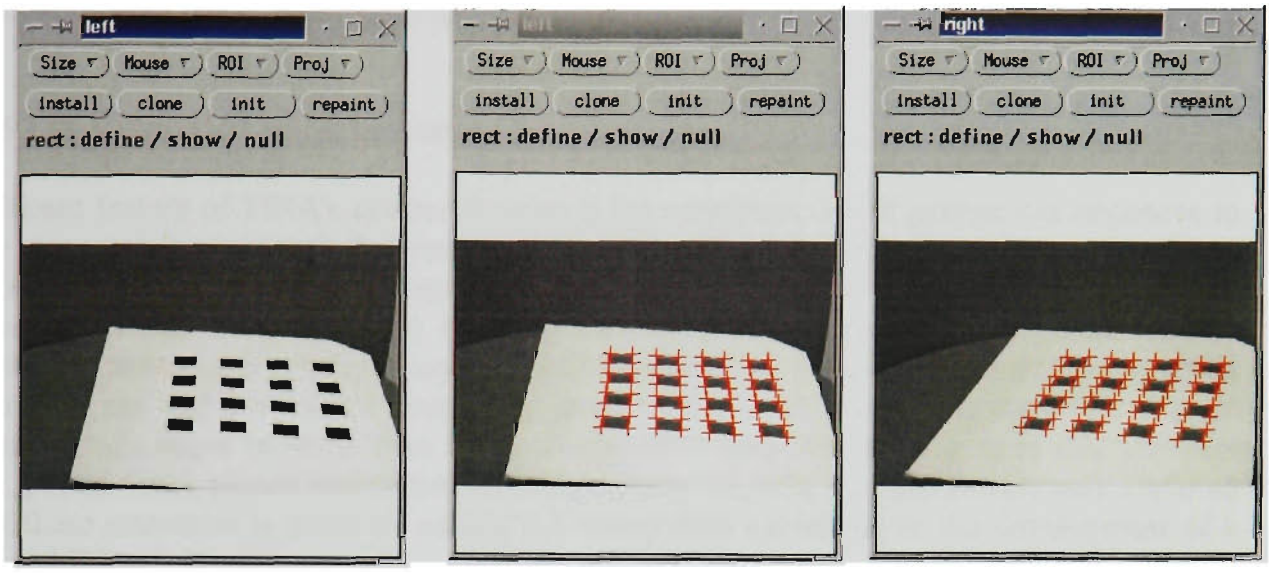
This tool is used primarily as an interface for performing edge detection, and edge based stereo processing. Its main application for this work has been in the recovery of 3D wire frame geometrical scene descriptions used in conjunction with the stereo tool. It utilises the following procedures:

**Canny edge detection:** This edge detection routine comprises of a two-stage process: In the first, edges with intensity gradients above a low threshold are identified, while in the second the edges are linked into strings. (Edge strings that are entirely below an upper threshold are eliminated). Final linked canny edges are obtained, to sub-pixel accuracy.

**Edge rectification:** The position of the edge data, obtained from use of the Canny edge extractor, are recalculated to be consistent with parallel camera geometry. The process is followed with stereo matching, and 3D geometrical recovery.

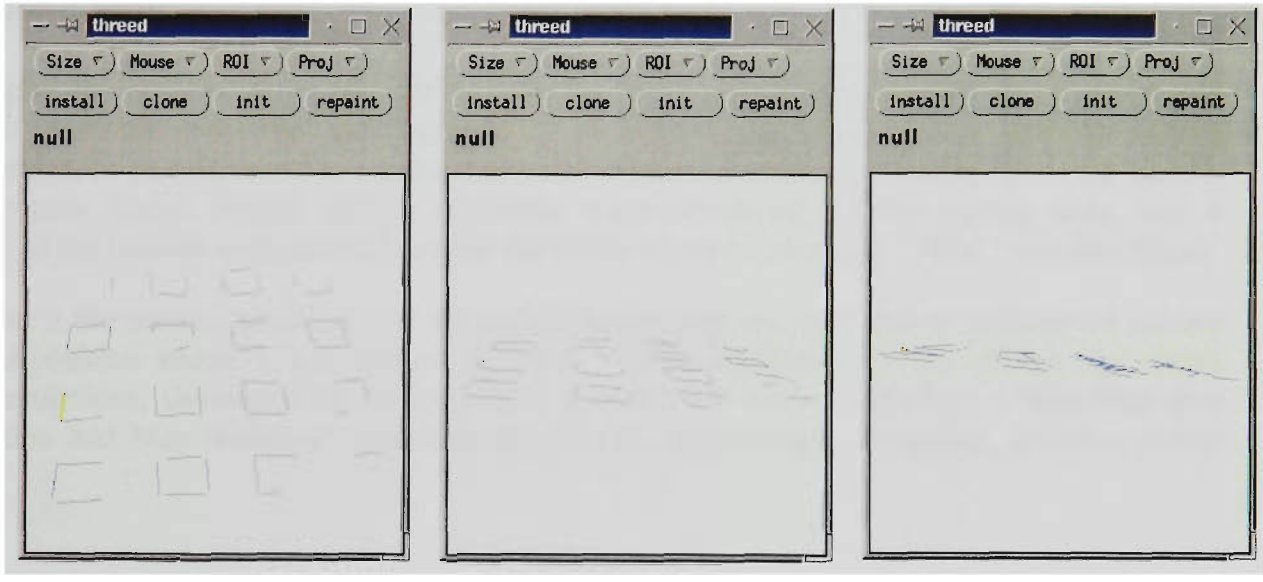
**Stereo:** An effective matching algorithm, derived from the "PMF" stereo algorithm [PMF85] is used, which relies strongly on support provided by local disparity gradient constraints. It utilises the list edge-string structures provided from Canny edge extractor. Refer to section 6.2.1 for details of TINA geometric list structures.

**Geom2 and Geom3:** Standard polygonal algorithms are applied to matched linked-edges to identify and obtain 2D and 3D line and conic structures respectively. The resulting 3D geometry is with respect to the left camera (virtual parallel camera geometry) coordinate frame, the origin of which is at the optical centre.




(a) Left image of calibration tile

(b) Left & right image corners found




(c) 3D reconstruction of calibration tile

(d) 3D views of calibration tile rotated to verify accuracy of planar reconstruction.

**Fig 4.3.2. 1 Calibration of stereo cameras using planar tile**

### 4.3.2.4 Matcher Tool

This tool was designed to provide 3D model creation and matching facilities, but at the time of initial introduction (1999), it was only partially operational. N. Thacker [Tha99] in consultation with M.Wingate, contributed by removing a number of limiting programming bugs, and the tool was subsequently modified and added to by M.Wingate, as part of this work, to incorporate a 3D object-model builder and recognition module. Greater detail of this work is given in chapter 5, The model matcher algorithms provided and modified, are used in the recognition module both

to perform 3D matching of object-model features to scene-model features and to compute the transformation which takes the model into the scene.

#### 4.3.2.5 TINA Generic List Structures

A significant feature of TINA's system libraries is the consistent use of generic list structures to provide connectivity amongst the various data types. These provide the means of building software with good modularity, regularity and robustness. Two basic representative list structures are used. These are (a) the singly connected list structure and (b) the doubly connected list structure. Each type may be applied recursively. Doubly connected lists, indexed by their start and end are termed strings and are a widely used structure to store and process segmented image edges (edgels). Five 3D geometric structures frequently used in this work are vectors, points, lines, planes and conics (curve geometry i.e. arcs, ellipses, circles etc). Detailed code of these structures is given in section 6.2 where their usefulness in the development of a "3D Model Editor", is more explicit.

## *4.4 Conclusion*

This chapter has outlined the mathematical notation and models used throughout this thesis. An important outcome was the development of a stereo head model that enables feature measurements to be referenced to a head frame that remains stationary with respect to the mobile base reference frame. Model motion equations were developed for the mobile base, and a summary of the models and methods used in the vision research platform "TINA" was described.

In Chapter 5 the motion equations for the mobile robot base are extended to include the robot's estimated position vector $\mathbf{f}_v$ and control vector $\mathbf{u}$. These together with the camera and stereo head formulations, developed in sections 4.2.1 to 4.2.5 are used to describe a "Simultaneous Localisation and Map Building" algorithm suitable for autonomous navigation, utilising active vision.

# Chapter 5 Navigation

## 5.0 Introduction

This chapter commences by restating the preliminary mobile robot model equations of motion developed in chapter 4, and then describes a Simultaneous Localisation and Map-building (SLAM) algorithm suitable for autonomous navigation in previously uncharted environments. The algorithm utilises the well known Extended Kalman Filter and is based on the map-building and sequential localisation algorithm developed by Davison [Dav98], [Dav01] for the Oxford GTI Mobile Base, and the collaborative work of Wingate and Davison [WD02a], [WD02b]. For this work, in its current phase, navigation in a totally unknown environment is not required. The system is intended to be capable of navigating to given sets of world coordinates where there is some likelihood of the targeted object being in relative close proximity, (e.g. to a table in a room, with recognition of the table not being a requirement). A limited obstacle avoidance module based on visible feature detection has been implemented to enhance navigational reliability. If an obstacle is detected, information relating its distance from the robot is returned and so movement around the obstacle can be scheduled. A limitation in the current implementation is that many obstacles do not have detectable features. For greater robustness in obstacle detection the future addition of a sonar sensor mounted on a miniature pan/tilt platform is planned. Details of the design and construction of the motorised pant/tilt platform are given in Burley and Wingate [BW02].

In general because of possible wheel slippage, calibration and measurement errors, the position of the mobile-base, calculated from odometry, can only be estimated to a degree of uncertainty. Appropriate modeling can ameliorate location uncertainty due to noise and measurement errors. For instance, wheel-velocities $v_1$ and $v_2$ (subsequently used as control input parameters) may be modelled assuming their errors to have a Gaussian variation with standard deviations proportional to the velocity demands themselves. Values of $\sigma_1=0.14\ v_1$, and $\sigma_2= 0.14\ v_2$ were found by trial and error to give good results. These expressions for uncertainty are incorporated in position estimation modeling in the derivations to follow. For circumstances where severe wheel slippage or obstacle collisions occur, additional positional information to that available from dead reckoning is essential for reliable robot self-localisation estimates.

In this work additional positional information is derived via the use of active stereo vision that acquires landmark visual features, which in turn are used for map-building of the local environment, and self-localisation. A feature map is a powerful tool when navigating in either previously explored or unexplored environments. For the situation of navigating in new environments, provided suitable features can be found from the commencement of motion, these features can be used for improving self-localisation estimates, as well as map building. Once several features are available for tracking they substantially reduce the risk of becoming lost, in the same way as a land-map can assist a human in finding their bearings. Known mapped features (i.e., features mapped during a previous voyage through the terrain) can likewise be helpful in the prevention of the loss of bearings.

## 5.1 Preliminary Mobile-Base Model Equations Revisited

Referring to Figs 4. 2.6.1/2/3 in Chapter 4, the following equations were arrived at and are reformulated here for clarity:

For constant time intervals $\Delta t$ (see equ. 4.55), the change in angular orientation $\theta$ is obtained by:

$$\theta = \frac{v_1 - v_2}{2 D_1} \Delta t \qquad \textbf{5. 1}$$

If the mobile robot's position at time t is: $\mathbf{x}_v = \begin{pmatrix} z(t) \\ x(t) \\ \phi(t) \end{pmatrix}$ **5. 2**

Following a constant time increment $\Delta t$ the robot's estimated new position will be:

$$z(t+\Delta t) = z + R( \sin (\phi(t)+\theta ) - \sin(\phi(t))) \qquad \textbf{5. 3}$$

rearranging to separate out variables $\phi$ (t) and $\theta$

$$z(t+\Delta t) = z(t) + R(\cos\phi(t)\sin\theta + \sin\phi(t)(\cos\theta-1)) \qquad \textbf{5. 4}$$

$$x(t+\Delta t) = x + R(\cos (\phi (t) - \cos (\phi (t) +\theta)) \qquad \textbf{5. 5}$$

rearranging to separate out variables $\phi(t)$ and $\theta$

$$x(t+\Delta t) = x(t) + R(\sin\phi(t)\sin\theta + \cos\phi(t)(1-\cos\theta)) \qquad \textbf{5. 6}$$

and

$$\phi(t+\Delta t) = \phi(t) + \theta \qquad \textbf{5. 7}$$

when $v_1 = v_2 = v$ (i.e., straight line case) $\theta \to 0$ and $R \to \infty$ and these equations can be written in a simplified form

$$z(t+\Delta t) = z(t) + ((v_1+v_2)/2)\Delta t\cos\phi(t) = z(t) + v\Delta t\cos\phi(t) \qquad \textbf{5. 8}$$

$$x(t+\Delta t) = x(t) + ((v_1+v_2)/2)\Delta t\sin\phi(t) = x(t) + v\Delta t\sin\phi(t) \qquad \textbf{5. 9}$$

$$\phi((t+\Delta t) = \phi(t) + ((v_1-v_2)/(2D_1))\ \Delta t = \phi(t) \qquad \textbf{5. 10}$$

Defining the robot's estimated position vector $\mathbf{f}_v$ and the input control vector $\mathbf{u}$ as:

$$\mathbf{f}_v = \begin{pmatrix} z(t+\Delta t) \\ x(t+\Delta t) \\ \phi(t+\Delta t) \end{pmatrix} \quad , \quad \mathbf{u} = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \qquad \textbf{5. 11}$$

$\mathbf{f}_v$ is a function of both $\mathbf{x}_v$ and $\mathbf{u}$ (refer equations 5.8-5.9) and will be normally be designated as $\mathbf{f}_v(\mathbf{x}_v, \mathbf{u})$.

The covariance matrix $\mathbf{Q}$ of $\mathbf{f}_v$ expressing the first order uncertainty in $\mathbf{f}_v$ is:

$$\mathbf{Q} = \nabla(\mathbf{f}_v)_{\mathbf{u}} \mathbf{U} \nabla(\mathbf{f}_v)_{\mathbf{u}}^T \qquad \textbf{5. 12}$$

Where

$$\nabla(\mathbf{f}_v)_{\mathbf{u}} = \begin{bmatrix} \nabla(z(t+\Delta t))_{v_1} & \nabla(z(t+\Delta t))_{v_2} \\ \nabla(x(t+\Delta t))_{v_1} & \nabla(x(t+\Delta t))_{v_2} \\ \nabla(\phi(t+\Delta t))_{v_1} & \nabla(\phi(t+\Delta t))_{v_2} \end{bmatrix} \quad , \quad \mathbf{U} = \begin{bmatrix} \sigma_{v_1}^2 & 0 \\ 0 & \sigma_{v2}^2 \end{bmatrix} \qquad \textbf{5. 13}$$

$\mathbf{U}$ is the covariance matrix of control vector $\mathbf{u}$, and $\sigma_{v_1}$ *and* $\sigma_{v_2}$ are standard deviation estimates of the errors in the velocity control input parameters $v_1$ and $v_2$. $\nabla(\mathbf{f}_v)_{\mathbf{u}}$ is obtained from:

$$\nabla(\mathbf{f}_v)_{\mathbf{u}} = \begin{bmatrix} \nabla(z)_R \nabla(R)_{v_1} + \nabla(z)_\theta \nabla(\theta)_{v_1} & \nabla(z)_R \nabla(R)_{v_2} + \nabla(z)_\theta \nabla(\theta)_{v_2} \\ \nabla(x)_R \nabla(R)_{v_1} + \nabla(x)_\theta \nabla(\theta)_{v_1} & \nabla(x)_R \nabla(R)_{v_2} + \nabla(x)_\theta \nabla(\theta)_{v_2} \\ \nabla(\theta)_{v_1} & \nabla(\theta)_{v_2} \end{bmatrix}$$

Where

$$\nabla(R)_{v_1} = -\frac{2D_1 v_2}{(v_1 - v_2)^2}, \quad \nabla(R)_{v_2} = \frac{2D_1 v_1}{(v_1 - v_2)^2},$$
$$\nabla(\theta)_{v_1} = \frac{1}{2D_1}\Delta t, \quad \nabla(\theta)_{v_2} = -\frac{1}{2D_1}\Delta t \qquad \textbf{5. 14}$$

$$\nabla(z)_\theta = R(\cos\phi\cos\theta - \sin\phi\sin\theta) \quad \nabla(z)_R = (\cos\phi\sin\theta - \sin\phi(\cos\theta - 1)) \qquad \textbf{5. 15}$$

$$\nabla(x)_R = (\sin\phi\sin\theta + \cos\phi(1-\cos\theta)) \quad \nabla(x)_\theta = R(\sin\phi\cos\theta + \cos\phi\sin\theta) \qquad \textbf{5. 16}$$

$$\nabla(\mathbf{f}_v)_{\mathbf{x}_v} = \begin{bmatrix} \nabla(z(t+\Delta t))_z & \nabla(z(t+\Delta t))_x & \nabla(z(t+\Delta t))_\phi \\ \nabla(x(t+\Delta t))_z & \nabla(x(t+\Delta t))_x & \nabla(x(t+\Delta t))_\phi \\ \nabla(\phi(t+\Delta t))_z & \nabla(\phi(t+\Delta t))_x & \nabla(\phi(t+\Delta t))_\phi \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & -R(\sin\phi(t)\sin\theta + \cos\phi(t)(\cos\theta - 1)) \\ 0 & 1 & R(\cos\phi(t)\sin\theta - \sin\phi(t)(1-\cos\theta)) \\ 0 & 0 & 1 \end{bmatrix} \qquad \textbf{5. 17}$$

For straight line travel $v = v_1 = v_2$ the previous associated expressions simplify to:

$$\nabla(z)_{v_1} = \Delta t\cos\phi, \quad \nabla(z)_{v_2} = \Delta t\cos\phi \quad \nabla(x)_{v_1} = \Delta t\sin\phi, \quad \nabla(x)_{v_2} = \Delta t\sin\phi$$

$$\nabla(\theta)_{v_1} = \frac{1}{2D_1}\Delta t, \quad \nabla(\theta)_{v_2} = -\frac{1}{2D_1}\Delta t \text{, while } \nabla(\mathbf{f}_v)_u \text{ and } \nabla(\mathbf{f}_v)_{\mathbf{x}_v} \text{ simplify to:}$$

$$\nabla(\mathbf{f}_v)_u = \begin{bmatrix} \Delta t\cos\phi & \Delta t\cos\phi \\ \Delta t\sin\phi & \Delta t\sin\phi \\ 0 & 0 \end{bmatrix} \qquad \textbf{5. 18}$$

and

$$\nabla(\mathbf{f}_v)_{\mathbf{x}_v} = \begin{bmatrix} 1 & 0 & -v\Delta t\sin\phi(t) \\ 0 & 1 & v\Delta t\cos\phi(t) \\ 0 & 0 & 1 \end{bmatrix} \qquad \textbf{5. 19}$$

respectively.

## 5.2 Landmark Features For Map-Building and Navigation

For a vehicle relying on active vision to navigate autonomously in an uncharted environment, a methodical map-building process to include surrounding stable landmarks is essential. This map can then be used to calculate the location of the vehicle with reference to a convenient reference point.

Essential for any feature-based active vision navigation system is a reliable feature detector and discrete landmark features that are easily detectable from a wide range of locations and angles. In this work, feature detection for map building and localisation is performed automatically using the **Z** operator described by Shi and Tomasi [ST94]. The **Z** operator locates patches in an image that are easy to track due to their large intensity variations across a patch, the associated feature is the centre pixel of the patch.

$$\mathbf{Z} = \sum_{patch} \begin{bmatrix} g_x^2 & g_x g_y \\ g_y g_x & g_y^2 \end{bmatrix} \qquad 5.20$$

where $g_x$ and $g_y$ are the horizontal and vertical gradients of image pixel intensities.

Image patches of size 13 x13 pixels are used. Being larger than required makes them readily distinguishable and thus more reliable, albeit at the cost of greater computational overheads. Patches using the equation for **Z** above are calculated about each pixel point. Eigenvalues $\lambda_1$ and $\lambda_2$ are next calculated and patches classified in ascending orders of interest according to the values of the $\lambda$ pairs. Optimal feature patches are those with the largest small eigenvalues of **Z**. The significance of the smaller eigenvalue being large, is that the patch has high variations in intensity making it more visible and thus making the point feature easier to track.

### 5.2.1 Feature Detection and Matching

To perform a measurement on a feature requires that it first be located in both the right and left image. In the sections that follow, it is shown that it is possible to ascertain a region in the left and right images comprising an ellipse within which a sought feature should lie with a high level of probability. Restricting the search to this ellipse considerably reduces the search overhead. To find a required feature, scanning in raster order of both the left and right image is performed and for each pixel in the ellipse region, a patch is calculated and compared with saved patches that represent target features. The comparison is performed using equation 5.21, which represents a normalised sum-of-squared-difference measure.

$$N = \frac{1}{n}\left( \sum_{patch} \left[ \frac{g_1 - \bar{g}_1}{\sigma_1} - \frac{g_0 - \bar{g}_0}{\sigma_0} \right]^2 \right) \qquad 5.21$$

$g_0$ and $g_1$ are image intensity values corresponding positions in the saved patch and image patch respectively, and $\bar{g}_0, \bar{g}_1, \sigma_0$ and $\sigma_1$ are means and standard deviations of the intensities across the patches. n is the number of pixels in a patch. This expression is 0 for a perfect match, and measures the average difference in standard deviations of values above the patch mean intensity between corresponding pixels in the patches being compared. Davison found a threshold value for N of 0.9 to be a suitable choice (see [Dav98, page 45]. For consistency this manually set threshold value for N was also used (i.e. the lowest value for N below 0.9 is accepted as the best match in the search zone). Note that any patch representation is intrinsically viewpoint variant, as features look different when viewed from new distances or angles. A criterion for expected visibility of the feature based on the differences between the original viewpoint and a new viewpoint, is formulated below

The feature is expected to be visible if the length ratio $\frac{|\mathbf{m}|}{|\mathbf{m}_{orig}|}$ lies in the range 0.7 to 1.4, and the angle difference $\beta = cos^{-1}(\frac{\mathbf{m}.\mathbf{m}_{orig}}{|\mathbf{m}||\mathbf{m}_{orig}|})$ in the range 0 to 45°.

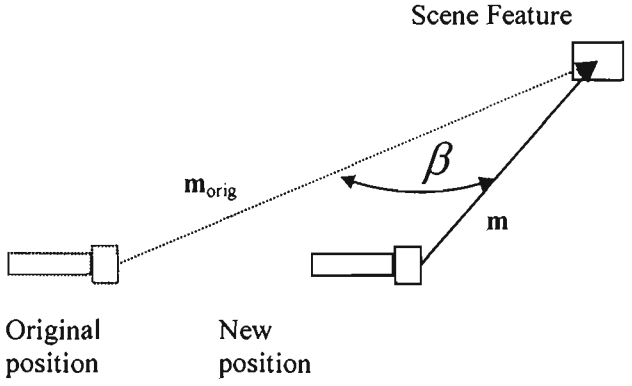



**Fig 5.2. 1 Scene feature as seen from two viewpoints**

### 5.2.2 More elaborate feature types

Rather than representing scene feature points as a two-dimensional image patch, some authors [ZF94], [Ruc97] use planar regions in 3D world coordinates. Many real features would fit this planar description (i.e., wall marks, door panels etc). An added overhead with this type of feature is the need to obtain an accurate transform from world plane to image plane. This transform would ensure the image plane obtained varied in shape and size appropriately with each new view location, giving greater confidence in matched features. With small inter-occular base lines, obtaining accurate transforms is however difficult and is not used here for this reason.
Other possibilities include line features and corner features. Fixating on either of these however is more difficult due to possible line fragmentation.

## *5.3 Feature Acquisition and Fixation*

Fixation is the process whereby the optic axis of the camera passes through the scene feature point. Fixation ensures image processing occurs near the centre of the image where lens distortion is at a minimum, and also minimises the likelihood of part of the feature lying outside the bounds of the image plane. Acquiring a new feature consists of finding the best patch in the left-image and then fixating on the feature by moving the head so the optic axes of both left and right cameras pass through it. The process is as follows: once the left image patch has been located, the corresponding epipolar line is generated in the right image from knowledge of the head angles (camera motion). The epipolar line in the right image, is the image of the light ray on which the feature must lie. A match in the right image is sought in close proximity to this line (typically within ±7 pixels of the line), and if successful the 3D location of the feature is

calculated as outlined in section 4.2.3. From the 3D location, the head angles required for fixation are calculated and the head positioned accordingly. If the feature is located within a small radius (i.e., 2 pixels) of the principal point in both the left and right image, fixation is successful; else the feature position is recalculated and the process repeated. If after several attempts fixation fails, the feature is discarded.

### 5.3.1 Uncertainty of Fixation Measurements

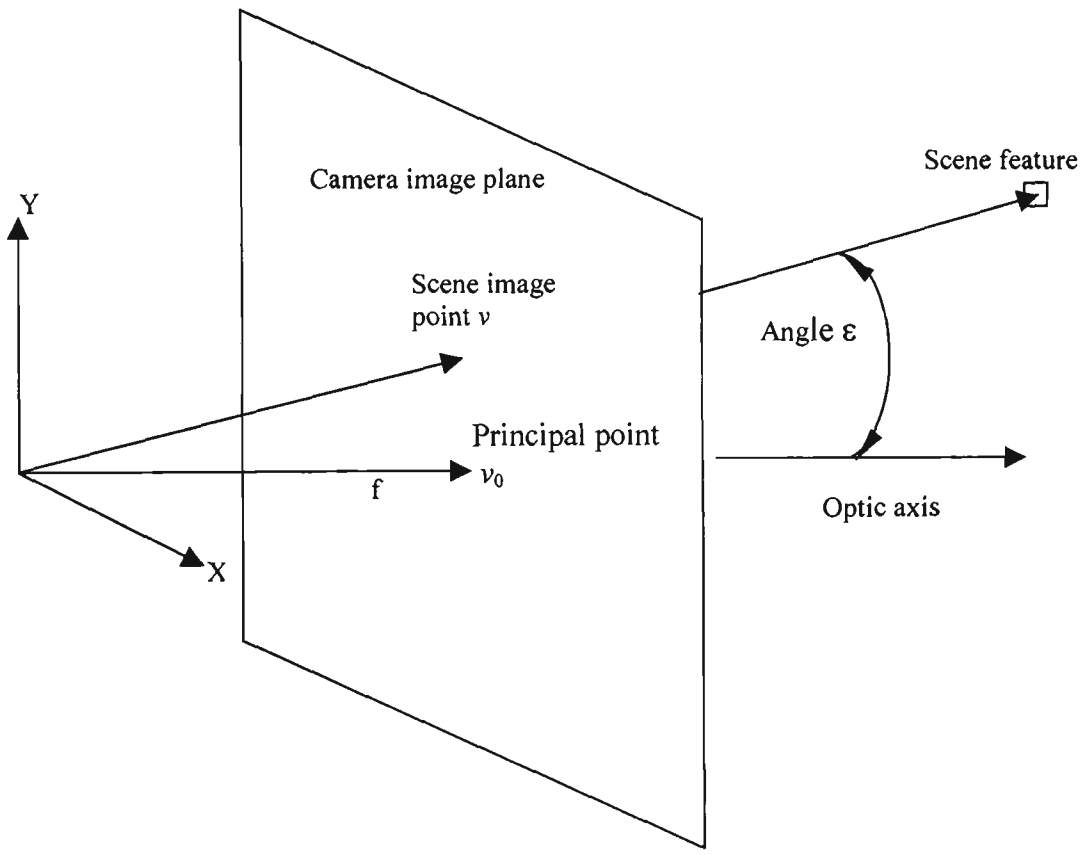



**Fig 5.3.1. 1 Uncertainty of fixation measurements**

Fig 5.3.1.1 shows a feature point with a small error in fixation. An accuracy of ±1 pixel would normally be expect in image measurements of detected features. To determine how this error is translated into angular errors in fixation, the vertical difference between v and $v_0$ in terms of the angle error ε is examine as follows:

$$\tan \varepsilon = \frac{v - v_0}{f k_v}$$

Where $f$ is the focal length and $k_v$ the vertical pixel density in pixels/metre.

The effect of a small change $\delta v$ in $v$ can be ascertained by differentiating the above equation as follows:

$$sec^2 \varepsilon \delta\varepsilon = \frac{\delta v}{f k_v}$$

At fixation ε will be small, approaching zero consequently $sec^2\varepsilon$ tends towards 1 thus:

$$\Rightarrow \delta\varepsilon = \frac{\delta v}{f k_v}.$$

For a fixation with an accuracy of 2 pixel (i.e., $\delta v=2$) ,$fk_v=807$,
$\delta\varepsilon \approx 2/(807) \approx 0.0024$rad $\approx 0.15$ degrees.

Head movements are repeatable to within the order ±0.1 degrees, consequently errors introduced by head movement motion are negligible in comparison to $\delta\varepsilon$.

Essentially the implication of the previous analysis is that an angular uncertainty value of .0024 rad must be added to all head fixation angle measurements. The effect of this uncertainty on 3D scene measurements can be quite significant when depth is large as shown in Fig 5.3.1.3. Error values were obtained using Fig 5.3.1.2 which shows the stereo cameras having an inter-occular spacing I metres, angular vergence uncertainty ε rad, fixation angles $\gamma_{L=}\gamma_R$, angular fixation error ε rad =$\Delta\gamma$ rad, error in Z of $\Delta Z$=Bd, error in X of $\Delta X$=Ad.

Consider triangle ABC, Angle B = $\gamma_L$, Angle C = $\gamma_L$ - ε,
Angle A = π- (B+C) = π - 2$\gamma_L$ + ε.
The error in Z, is $\Delta Z$ = Bd, while the error in X, is $\Delta X$ = Ad.
SideBC = (tan(B) - tan(C))/(I/2),
Using the sine rule SideAB=SideBC*sin(C)/sin(A),
$\Delta Z$= Bd = sideAB*cos(B)
and $\Delta X$= Ad =$\Delta Z$*tan (B)

The significance of these uncertainties with I = 0.25 m (the inter-occular distance for our BiSight head) and vergence uncertainties of 0.005 rad is highlighted in the graphs of Fig 5.3.1.3.

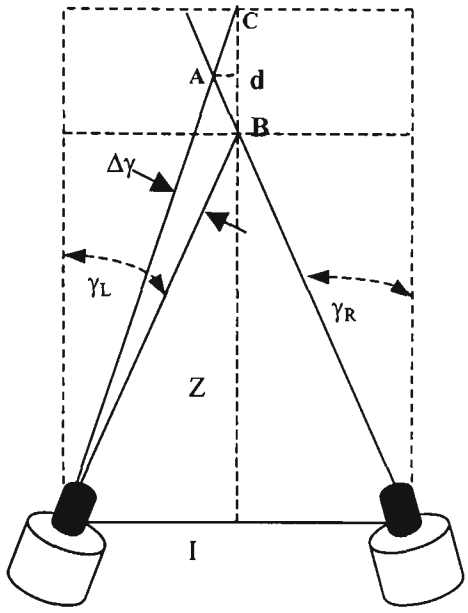
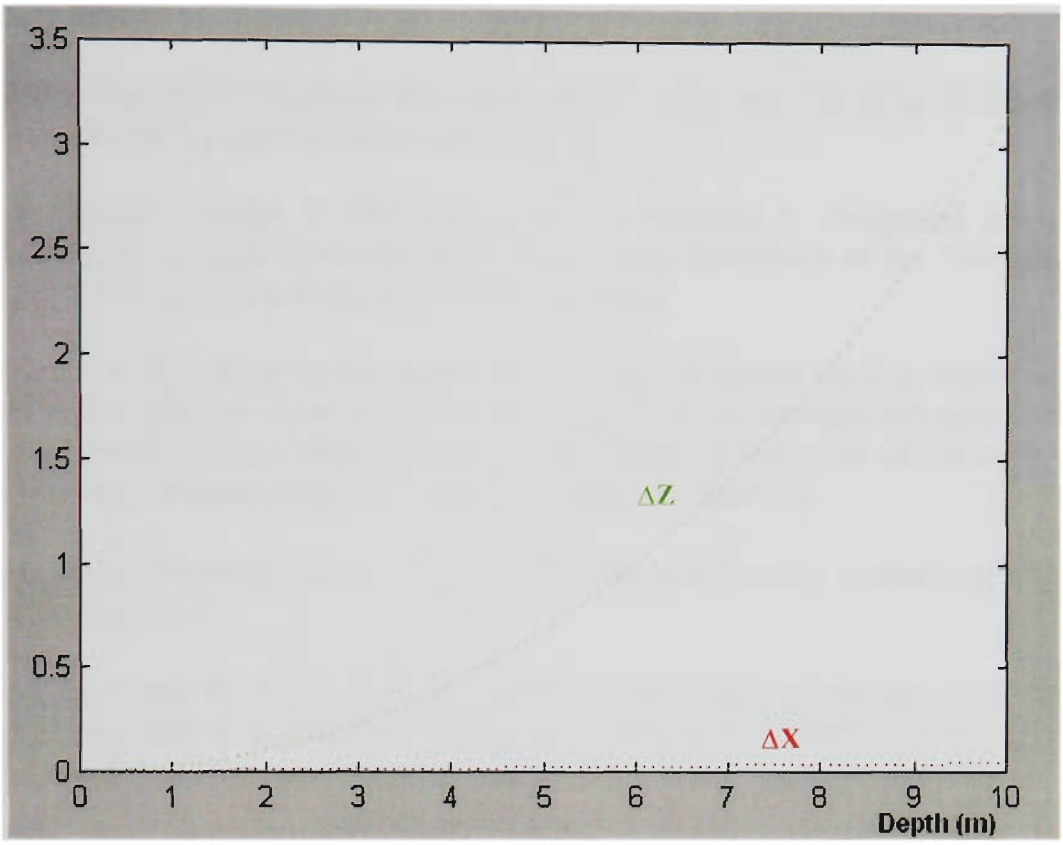



**Fig 5.3.1. 2 Converging stereo cameras showing potential measurement errors in X and Z**

Examination of Fig 5.3.1.3 reveals that transverse errors (ΔX) are quite small, reaching a maximum of 0.04m at a depth of 10m, whereas the corresponding uncertainty in depth estimations (ΔZ), is in the order of 3.1m. This is not unexpected, as for large Z values the camera optical axes are near parallel, and consequently small uncertainties in the vergence angles translate into large depth uncertainties. An ellipse with (ΔZ) as the major axis and (ΔX) as the minor axis, is a useful pictorial way of portraying these uncertainties. Accounting for them during localisation and map building is covered in the next section.




Transverse errors (ΔX) are quite small, reaching a maximum of 0.04m at a depth of 10m, whereas the corresponding uncertainty in depth estimations (ΔZ), is in the order of 3.1m.

**Fig 5.3.1. 3 Fixation error estimations for I=0.25m and vergence uncertainties of .005 rad**

## *5.4 Navigation: Map Building and Localisation*

To be reliable a map-building and localisation process must account for the uncertainties due to noise in any measurements made. An effective tool that incorporates the modeling of noise and provides optimal estimations of location, is the Extended Kalman Filter. As with earlier authors [New99] [Cso97] [CDT00], MROLR's map-building and localisation algorithms use an Extended Kalman Filter.

### 5.4.1 Extended Kalman Filtering

Considerable literature exists on both the Kalman Filter (KF) and Extended Kalman Filter (EKF), see for example [Sor85], [Gel96]. The KF addresses the general problem of trying to estimate the state of a linear discrete-time controlled process in the presence of noise. The EKF is an extension of the KF to permit its application to non-linear systems with state transitions and whose functions contain non-Gaussian noise.

### 5.4.1 Notation

The following briefly outlines the notation used for the Extended Kalman Filter. Further explanations accompany the subsequent formulations.

*State transition function* **f**: The state transition function is designated by **f**. This function incorporates the system dynamics and facilitates the estimation of the "current state" during a time step $\Delta t$ during which no measurements are made.

*State vector* **x**: The state vector models the system, it contains the best estimate of quantities of interest as the system evolves with time. Importantly it incorporates any measurements (**z**) of the quantities made during a time interval $\Delta t$. The expected evolution of the state vector with the passage of time is encapsulated in f, the state transition function.

*Current state estimation vector* $\hat{\mathbf{x}}$ : The current state estimation is stored in this vector and the covariance matrix **P**.

*Covariance matrix* **P**: The covariance matrix represents the uncertainty in $\hat{\mathbf{x}}$ due to noise **Q** in the state transition. **P** is a square symmetric matrix, with a dimension equal to the number of elements in $\hat{\mathbf{x}}$

*Noise* **Q**: This variable provides a means of allowing for random or unaccounted effects in the dynamic model.

*Measurements* **z** and **m**: **z** is used to designate an actual measurement (i.e., using the active stereo head), and **m** for a prediction of the measurement.

$\hat{\mathbf{x}}(k+1/k)$: An estimate $\hat{\mathbf{x}}$ of the state **x** at a time step k + 1, based on the estimate at time step k and an observation (measurement) made at time step k + 1.

*Innovation* $\nu$: The innovation $\nu$ is the difference between the actual measurement **z** and the predicted measurement **m** calculated from the current state.

*Covariance matrix of the noise* **R**: This matrix represents the covariance of the noise in the measurement.

*Gain* **W** and innovation covariance **S**: **W** designates the Kalman gain and **S** the innovation covariance. The innovation covariance represents the uncertainty in $\nu$ (i.e., the amount by which an actual measurement differs from its predicted value).

### 5.4.2 The Extended Kalman Filter

**Prediction**

Initially, in a step of time $\Delta t$ in which no measurements are made, an estimate of the state of the system $\hat{\mathbf{x}}$ may be obtained from the state transition function **f** and the system dynamics as follows:

$$\hat{\mathbf{x}}(k+1/k) = \mathbf{f}(\hat{\mathbf{x}}(k/k), \mathbf{u}(k)) \qquad 5.22$$

Where **u** is the vehicle's control vector specifying the wheel velocities and indirectly the steering angle (see equation5.1).

$$\mathbf{P}(k+1/k) = \nabla(\mathbf{f})_{\mathbf{x}}(k/k)\ \mathbf{P}(k/k)\ \nabla(\mathbf{f})_{\mathbf{x}}^{T}(k/k) + \mathbf{Q}(k) \qquad 5.23$$

During the state transition the covariance matrix changes reflect the increase in uncertainty in the state due to noise **Q. f** and **Q** both depend on **u**, the current mobile-robot control vector which is a function of the wheel demand velocities $v_1$ and $v_2$.

**State Update**

Following a reliable measurement, the state estimate will improve with this new information and the uncertainty represented by **P** will reduce. In the following update equations, the innovation $\nu$ is the difference between the actual measurement **z** and the prediction **m** calculated from the current state. **R** is the covariance matrix of the noise in the measurement, **W** the Kalman gain, and **S** the innovation covariance, which represents the uncertainty in $\nu$ (i.e., the amount by which an actual measurement differs from its predicted value).

$$\hat{\mathbf{x}}(k+1/k+1) = \hat{\mathbf{x}}(k+1/k) + \mathbf{W}(k+1)\nu(k+1) \qquad 5.24$$

$$\mathbf{P}(k+1/k+1) = \mathbf{P}(k+1/k) - \mathbf{W}(k+1)\mathbf{S}(k+1)\mathbf{W}^{T}(k+1) \qquad 5.25$$

Where the innovation $\nu$ (the difference between the actual measurement **z** and the predicted measurement **m**) is obtained from:

$$\nu(k+1) = \mathbf{z}(k+1) - \mathbf{m}(\hat{\mathbf{x}}(k+1/k)) \qquad 5.26$$

The Kalman gain **W** is given by:

$$\mathbf{W}(k+1) = \mathbf{P}(k+1/k)\nabla(\mathbf{m})_{\mathbf{x}}^{T}(k/k)\,\mathbf{S}^{-1}(k+1) \qquad 5.27$$

and the innovation covariance **S** (which represents the uncertainty in $\nu$) from:

$$\mathbf{S}(k+1/k) = \nabla(\mathbf{m})_{\mathbf{x}}(k/k)\,\mathbf{P}(k+1/k)\nabla(\mathbf{m})_{\mathbf{x}}^{T}(k/k) + \mathbf{R}(k+1) \qquad 5.28$$

## 5.5 Localisation and Map-Building

### 5.5.1 Formulation of the State Vector and its Covariance

At any given time, estimates of mobile-robot and scene-feature locations (in the world reference coordinate frame) are stored in the system state vector $\hat{\mathbf{x}}$ and the corresponding uncertainty of these estimates in the covariance matrix $\mathbf{P}$. Equation 5.8 shows these in partitioned form; $\hat{\mathbf{x}}_v$ is the mobile-robots position estimate, and $\hat{\mathbf{y}}_i$ the estimated 3D location of the $i$th feature. The dimensions of both vector $\hat{\mathbf{x}}$ and matrix $\mathbf{P}$ change as scene-features are added or deleted. Note that for brevity frame subscripts have been dropped.

$$\hat{\mathbf{x}} = \begin{pmatrix} \hat{\mathbf{x}}_v \\ \hat{\mathbf{y}}_1 \\ \hat{\mathbf{y}}_2 \\ \vdots \end{pmatrix}, \quad \mathbf{P} = \begin{bmatrix} \mathbf{P}_{xx} & \mathbf{P}_{xy_1} & \mathbf{P}_{xy_2} & \cdots \\ \mathbf{P}_{y_1x} & \mathbf{P}_{y_1y_1} & \mathbf{P}_{y_1y_2} & \cdots \\ \mathbf{P}_{y_2x} & \mathbf{P}_{y_2y_1} & \mathbf{P}_{y_2y_2} & \cdots \\ \vdots & \vdots & \vdots & \cdots \end{bmatrix} \qquad \text{5. 29}$$

Where

$$\hat{\mathbf{x}} = \begin{pmatrix} \hat{z} \\ \hat{x} \\ \hat{\phi} \end{pmatrix}, \quad \hat{\mathbf{y}}_i = \begin{pmatrix} \hat{X}_i \\ \hat{Y}_i \\ \hat{Z}_i \end{pmatrix} \qquad \text{5. 30}$$

Note $\hat{\mathbf{x}}$ comprises 3(n+1) elements, while $\mathbf{P}$ is of size 3(n+1) x 3(n+1), where n is the number of mapped features. The dimensions of both $\hat{\mathbf{x}}$ and $\mathbf{P}$ change as features are added or deleted.

### 5.5.2 Initialization

The filter's initialization occurs when the mobile-robot is in the starting position ($\hat{z} = 0, \hat{x} = 0, \hat{\phi} = 0$) at the origin of the world reference frame and aligned with it, and no known scene-features have yet been acquired. Since the state is known with certainty:

$$\hat{\mathbf{x}} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \qquad \mathbf{P} = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \qquad \text{5. 31}$$

### 5.5.3 State Estimation following a Movement

A new estimate of state $\hat{\mathbf{x}}$ and covariance **P**, following movement of the mobile-robot over a constant time increment $\Delta t$, designated by the label k is:

$$\hat{\mathbf{x}}(k+1/k) = \begin{pmatrix} \mathbf{f}_v(k/k), \mathbf{u}(k) \\ \hat{\mathbf{y}}_1(k/k) \\ \hat{\mathbf{y}}_2(k/k) \\ \vdots \end{pmatrix} \qquad 5.32$$

$$\mathbf{P}(k+1/k) = \begin{bmatrix} \nabla(\mathbf{f}_v)_{x_v} \mathbf{P}_{xx}(k/k)(\mathbf{f}_v)^T_{x_v} + \mathbf{Q}(k) & \nabla(\mathbf{f}_v)_{x_v} \mathbf{P}_{xy_1}(k/k) & (\mathbf{f}_v)_{x_v} \mathbf{P}_{xy_2}(k/k) & \cdots \\ \mathbf{P}_{y_1x}(k/k)(\mathbf{f}_v)^T_{x_v} & \mathbf{P}_{y_1y_1}(k/k) & \mathbf{P}_{y_1y_2}(k/k) & \cdots \\ \mathbf{P}_{y_2x}(k/k)(\mathbf{f}_v)^T_{x_v} & \mathbf{P}_{y_2y_1}(k/k) & \mathbf{P}_{y_2y_2}(k/k) & \cdots \\ \cdot & \cdot & \cdot & \\ \cdot & \cdot & \cdot & \\ \cdot & \cdot & \cdot & \end{bmatrix} \qquad 5.33$$

Where $\mathbf{f}_v$ and **Q(k)** are defined in Equations 5.11 and 5.12.

(Equation 5.33 is obtained from $\mathbf{P}(k+1/k) = \nabla(\mathbf{f})_x \mathbf{P}(k/k) \nabla(\mathbf{f})^T_x + \mathbf{Q}(k)$. Note $\nabla(\mathbf{f})_x$ is the full state transition Jacobian (equation 5.34) ).

$$\nabla(\mathbf{f})_x = \begin{bmatrix} \nabla(\mathbf{f}_v)_{x_v} & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \cdots \\ \vdots & \vdots & & \end{bmatrix} \qquad 5.34$$

### 5.5.4 Measurement and Feature Search

Recall that both the mobile-robot and scene-feature positions are specified in world reference coordinates, however scene-features are physically measured relative to the Head centre. The true *i*th feature position (relative to the Head) is thus given by:

$$\mathbf{m}_{\mathrm{GI}}^{\mathrm{C0}} = \begin{pmatrix} m_{Gix}^{C0} \\ m_{Giy}^{C0} \\ m_{Giz}^{C0} \end{pmatrix} = \begin{pmatrix} (X_i - x)cos\phi - (Z_i - z)sin\phi \\ Y_i - M_h \\ (X_i - x)sin\phi + (Z_i - z)cos\phi \end{pmatrix} \qquad \textbf{5. 35}$$

An estimate of the *i*th feature position at the current mobile base location is obtained by substituting estimated values for the above variables (i.e., $\hat{\mathbf{x}}_v$ and $\hat{\mathbf{y}}_i$).

Following a measurement of this vector, the innovation covariance is:

$$\begin{aligned} \mathbf{S}_{\mathrm{mGI}} &= \nabla(\mathbf{m}_{\mathrm{GI}})_{\mathrm{x}} \mathbf{P} \nabla(\mathbf{m}_{\mathrm{GI}})_{\mathrm{x}}^T + \mathbf{R}_L \\ &= \nabla(\mathbf{m}_{\mathrm{GI}})_{\mathrm{x_v}} \mathbf{P}_{\mathrm{xx}} \nabla(\mathbf{m}_{\mathrm{GI}})_{\mathrm{x_v}}^T + 2\nabla(\mathbf{m}_{\mathrm{Gi}})_{\mathrm{x_v}} \mathbf{P}_{\mathrm{xy_i}} \nabla(\mathbf{m}_{\mathrm{Gi}})_{\mathrm{y_i}}^T \\ &\quad + \nabla(\mathbf{m}_{\mathrm{GI}})_{\mathrm{y_i}} \mathbf{P}_{\mathrm{y_i y_i}} \nabla(\mathbf{m}_{\mathrm{GI}})_{\mathrm{y_i}}^T + \mathbf{R}_L \end{aligned} \qquad \textbf{5. 36}$$

Note in equation 5.36, the measurement noise $\mathbf{R}_L$, is transformed into Cartesian measurement space. For transformation into angular form see section 5.5.5

The process of measuring a scene-feature is as follows: from the above estimated relative scene-feature location, the head angles necessary for fixation are calculated (see Section 5.5.3) and the head moved to them. The vectors $\mathbf{m}_L$ and $\mathbf{m}_R$ and their covariances $\mathbf{P}_{\mathbf{m}_L}$ and $\mathbf{P}_{\mathbf{m}_R}$ are next calculated from transformed $\mathbf{Sm}_{\mathrm{Gi}}$. At fixation both $\mathbf{m}_L$ and $\mathbf{m}_R$ will have zero $x$ and $y$ components as the camera's optic axes pass through scene-feature.

For the left camera:

$$u_L = -fk_u \frac{m_{Lx}^L}{m_{Lz}^L} + u_0 \quad and \quad v_L = -fk_v \frac{m_{Lx}^R}{m_{Lz}^R} + v_0 \qquad \textbf{5. 37}$$

The establishment of the covariance matrix of the image vector $\mathbf{u}_L = \begin{pmatrix} u_L \\ v_L \end{pmatrix}$ is obtained from

$$\mathbf{U}_L = \nabla(\mathbf{u}_L)_{\mathbf{m}_L} \mathbf{P}_{\mathbf{m}_L} \nabla(\mathbf{u}_L)_{m_L}^T \qquad \textbf{5. 38}$$

The Jacobian $\nabla(\mathbf{u}_L)_{\mathbf{m}_L}$ contains mostly zero values since $m^L_{Lx} = m^L_{Ly} = 0$

$$\nabla(\mathbf{u}_L)_{\mathbf{m}_L} = \begin{bmatrix} \frac{-fk_u}{m_{Lz}} & 0 & 0 \\ 0 & \frac{-fk_v}{m_{Lz}} & 0 \end{bmatrix} \qquad \textbf{5. 39}$$

$\mathbf{U}_L$ defines an ellipse in the left image whose size is governed by the declaration of the number of standard deviations. This defined area can be searched to locate the patch representing the scene-feature in question. The procedure is repeated in the right image. Restricting the search areas in this manner is computationally efficient as well as minimizing mismatches. Following a successful feature match in both images the final measurement of $\mathbf{m}^{C0}_{Gi}$ is calculated

### 5.5.5 State Vector Update following a Measurement

There are considerable computational savings if prior to processing a measurement of a scene-feature, it is transformed into angular coordinates (see Davison [Dav98, page 68]). This transformation of $\mathbf{m}^{C0}_{Gi}$, resulting in measurement vector $\mathbf{m}_i$ is given in equation 5.40.

$$\mathbf{m}_i = \begin{pmatrix} \alpha_i \\ e_i \\ \gamma_i \end{pmatrix} = \begin{pmatrix} \tan^{-1} \frac{m_{Gix}}{m_{Giz}} \\ \tan^{-1} \frac{m_{Giy}}{m_{Gi\rho}} \\ \tan^{-1} \frac{I}{2m_{Gi}} \end{pmatrix} \qquad \textbf{5. 40}$$

Where $m_{Gi}$ is the scalar length of vector $\mathbf{m}_{Gi}$ and $m_{Gi\rho} = \sqrt{(m_{Gix}{}^2 + m_{Giz}{}^2}$ is its projection onto the *xz* plane. I is the inter-ocular separation of the head. These angles represent the pan, elevation and vergence angles respectively of an ideal active head positioned at the head centre and fixating the feature (i.e., head offsets =0).

The advantage obtained by the use of angular measurement is that it allows measurement noise to be represented as a constant diagonal matrix. For a successful fixation, lock-on is achieved if the feature is located within a specified radius from the principal point (typically two pixels) in both images. For a specified radius of two pixels his represents an angular uncertainty of around $0.3^\circ$ (see section 5.3.1) and hence errors with this standard deviation (assumed Gaussian) can be assigned to $\alpha_i$, $e_i$, $\gamma_i$. (i.e., $\Delta\alpha_i$ $\Delta e_i$, $\Delta\gamma_i$). The measurement noise covariance matrix $\mathbf{R}$ is given in Equation 5.41, and it is a diagonal matrix in which $\alpha_i$ $e_i$ $\gamma_i$, are independent variables, making it simpler to represent measurement noise as Gaussian, consequently eliminating potential bias during filter update. In addition the measurement vector $\mathbf{m}_i$ can be decoupled, permitting scalar measurements to be used to update the filter. Computationally this has the substantial advantage that in updating the filter, matrix inversion is not required.

$$\mathbf{R}=\begin{bmatrix}\Delta\alpha^2 & 0 & 0\\ 0 & \Delta e^2 & 0\\ 0 & 0 & \Delta\gamma^2\end{bmatrix} \qquad 5.41$$

The Jacobian for each scalar part of the measurement $\mathbf{m}_i$ for the current scene-feature is:

$$\nabla(m_i)_{\mathbf{x}} = \left(\nabla(m_i)_{\mathbf{x}_v} \quad \mathbf{0}^T \quad \cdots \quad \mathbf{0}^T \quad \nabla(m_i)_{\mathbf{y}_i} \quad \mathbf{0}^T \quad \cdots\right),$$

Where scalar component $m_i$ is one of $\alpha_i$ $e_i$ $\gamma_i$. The scalar innovation S is calculated from equation 5.42 in which $\mathbf{P}_{xx}$, $\mathbf{P}_{xy_i}$ and $\mathbf{P}_{y_iy_i}$ are 3 x 3 blocks of the current state covariance matrix **P**, and R is the scalar measurement noise variance ($\Delta\alpha_i^2$, $\Delta e_i^2$ or $\Delta\gamma_i^2$.) of the measurement. The Kalman gain **W** is now calculated by equation 5.43, enabling the filter to be updated (equations 5.44 and 5.45). In equation 5.44, $z_i$ is the actual measurement component obtained from the head and $m_i$ the corresponding prediction component.

$$\begin{aligned}S &= \nabla(m_i)_{\mathbf{x}}\mathbf{P}\nabla(m_i)_{\mathbf{x}}^T + R\\ &= \nabla(m_i)_{\mathbf{x}}\mathbf{P}_{xx}\nabla(m_i)_{\mathbf{x}}^T + 2\nabla(m_i)_{\mathbf{x}_v}\mathbf{P}_{xy_i}\nabla(m_i)_{\mathbf{y}_i}^T + \nabla(m_i)_{\mathbf{y}_i}\mathbf{P}_{y_iy_i}\nabla(m_i)_{\mathbf{y}_i}^T + R\end{aligned} \qquad 5.42$$

$$\mathbf{W} = \mathbf{P}\nabla(m_i)_{\mathbf{x}}S^{-1} = S^{-1}\begin{pmatrix}\mathbf{P}_{xx}\\ \mathbf{P}_{y_1x}\\ \mathbf{P}_{y_2x}\\ \vdots\end{pmatrix}\nabla(m_i)_{\mathbf{x}_v} + S^{-1}\begin{pmatrix}\mathbf{P}_{xy_i}\\ \mathbf{P}_{y_1x_i}\\ \mathbf{P}_{y_2y_i}\\ \vdots\end{pmatrix}\nabla(m_i)_{\mathbf{y}_i} \qquad 5.43$$

Note in equation 5.43 since S is scalar $S^{-1}$ =1/S.

$$\hat{\mathbf{x}}_{new} = \hat{\mathbf{x}}_{old} + \mathbf{W}(z_i - m_i) \qquad 5.44$$

$z_i$ is the measured quantity from the head while $\mathbf{m}_i$ the predicted.

$$\mathbf{P}_{new} = \mathbf{P}_{old} - \mathbf{WSW}^T \qquad 5.45$$

### 5.5.6 New Feature Initialization

For first time observation of a new scene-feature *i*, the vector $\mathbf{m}_{Gi}$ relative to the head centre is obtained and the state of the new feature $\mathbf{y}_i$, initialised as:

$$\mathbf{y}_i = \begin{pmatrix} X_i \\ Y_i \\ Z_i \end{pmatrix} = \begin{pmatrix} x + m_{Gix}\cos\phi + m_{Giz}\sin\phi \\ M_h + m_{Giy} \\ z - m_{Gix}\sin\phi + m_{Giz}\cos\phi \end{pmatrix} \quad \textbf{5.46}$$

The total state vector **x** and covariance **P** are updated to:

$$\mathbf{x}_{new} = \begin{pmatrix} \mathbf{x}_v \\ \mathbf{y}_1 \\ \mathbf{y}_2 \\ \mathbf{y}_i \end{pmatrix} \quad \textbf{5.47}$$

$$\mathbf{P}_{new} = \begin{bmatrix} \mathbf{P}_{xx} & \mathbf{P}_{xy_1} & \mathbf{P}_{xy_2} & \mathbf{P}_{xx}\nabla(\mathbf{y}_i)^T_{\mathbf{x}_v} \\ \mathbf{P}_{y_1x} & \mathbf{P}_{y_1y_1} & \mathbf{P}_{y_1y_1} & \mathbf{P}_{y_3x}\nabla(\mathbf{y}_i)^T_{\mathbf{x}_v} \\ \mathbf{P}_{y_2x} & \mathbf{P}_{y_2y_1} & \mathbf{P}_{y_2y_2} & \mathbf{P}_{y_2x}\nabla(\mathbf{y}_i)^T_{\mathbf{x}_v} \\ \nabla(\mathbf{y}_i)_{\mathbf{x}_v}\mathbf{P}_{xx} & \nabla(\mathbf{y}_i)_{\mathbf{x}_v}\mathbf{P}_{xy_2} & \nabla(\mathbf{y}_i)_{\mathbf{x}_v}\mathbf{P}_{xy_2} & \nabla(\mathbf{y}_i)_{x_v}\mathbf{P}_{xx}\nabla(\mathbf{y}_i)^T_{x_v} + \nabla(\mathbf{y}_i)_{\mathbf{h}_G}\mathbf{R}_L\nabla(\mathbf{y}_i)^T_{\mathbf{h}_G} \end{bmatrix} \quad \textbf{5.48}$$

### 5.5.7 Scene-Feature Deletion

Deleting the *i*th scene-feature simple requires its removal from the state vector, and *i*th row and column from the covariance matrix. The example below applies for $i = 2$.

$$\begin{pmatrix} \mathbf{x}_v \\ \mathbf{y}_1 \\ \mathbf{y}_2 \\ \mathbf{y}_3 \end{pmatrix} \rightarrow \begin{pmatrix} \mathbf{x}_v \\ \mathbf{y}_1 \\ \mathbf{y}_3 \end{pmatrix} \quad \text{and}$$

$$\begin{bmatrix} \mathbf{P}_{xx} & \mathbf{P}_{xy_1} & \mathbf{P}_{xy_2} & \mathbf{P}_{xy_3} \\ \mathbf{P}_{y_1x} & \mathbf{P}_{y_1y_1} & \mathbf{P}_{y_1y_2} & \mathbf{P}_{y_1y_3} \\ \mathbf{P}_{y_2x} & \mathbf{P}_{y_2y_1} & \mathbf{P}_{y_2y_2} & \mathbf{P}_{y_2y_3} \\ \mathbf{P}_{y_3x} & \mathbf{P}_{y_3y_1} & \mathbf{P}_{y_3y_2} & \mathbf{P}_{y_3y_3} \end{bmatrix} \rightarrow \begin{bmatrix} \mathbf{P}_{xx} & \mathbf{P}_{xy_1} & \mathbf{P}_{xy_3} \\ \mathbf{P}_{y_1x} & \mathbf{P}_{y_1y_1} & \mathbf{P}_{y_1y_3} \\ \mathbf{P}_{y_3x} & \mathbf{P}_{y_3y_1} & \mathbf{P}_{y_2y_3} \end{bmatrix} \quad \text{5. 49}$$

### 5.5.8 World-Coordinate Frame Zeroing

For circumstances when it is more convenient to use the current mobile-robot position as the reference coordinate frame, this can be accomplished at any stage by simply zeroing the world coordinate frame and assigning the new state to:

$$\mathbf{x}_{new} = \begin{pmatrix} \mathbf{x}_{v_{new}} \\ \mathbf{y}_{1_{new}} \\ \mathbf{y}_{2_{new}} \\ \vdots \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{m}_{G1} + \mathbf{M}_h \\ \mathbf{m}_{G2} + \mathbf{M}_h \\ \vdots \end{pmatrix} \quad \text{5. 50}$$

$\mathbf{m}_{Gi}$ is the vector to the ith feature from the head centre, and $\mathbf{M}_h$ the constant vertical offset vector from the ground plane to the head centre. The corresponding state covariance and Jacobian matrices are calculated using equations 5.51 and 5.52

$$\nabla(\mathbf{x}_{new})_{\mathbf{x}_{old}} = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots \\ \nabla(\mathbf{m}_{G1})_{\mathbf{x}_v} & \nabla(\mathbf{m}_{G1})_{\mathbf{y}_1} & \mathbf{0} & \cdots \\ \nabla(\mathbf{m}_{G2})_{\mathbf{x}_v} & \mathbf{0} & \nabla(\mathbf{m}_{G2})_{\mathbf{y}_2} & \cdots \\ \vdots & \vdots & \vdots & \end{bmatrix} \quad \text{5. 51}$$

$$\mathbf{P}_{(new)} = \nabla(\mathbf{x}_{new})_{\mathbf{x}_{old}} \mathbf{P}_{old} \nabla(\mathbf{x}_{new})^T_{\mathbf{x}_{old}} \quad \text{5. 52}$$

## 5.6 Map-Building

With the above analysis, the framework for estimating the location of the moving mobile-robot and observed stationary features is now available. The tracking of a scene-feature proceeds as follows:

> Assume that a target feature has been fixated during a measurement phase of the filter. At the commencement of movement, the prediction step of the filter will provide an estimate of the new position of the mobile-robot on reaching the next measurement point, as well as the expected relative position of the target feature. Using this information the stereo head is driven to fixate on the feature in readiness for the next measurement at the end of the current time increment.

The actions required are:

- Using current control inputs, carry out the prediction step and provide an estimate of the robot's new position.
- Select the feature for tracking and calculate the estimated head angles for fixation. Move the head to this position in preparation.
- Move the robot to the new calculated position based on its odometry.
- Capture new images.
- Perform feature matching in both images using correlation as in section 5.2.1
- If a match is found, carry out a measurement and update the filter.

It is sensible to make repeated fixated measurements of a single feature as the robot moves past, the issue is when to search for a new feature, or track an alternative feature. It is not difficult to envisage situations arising where a large number of features with full covariance need to be stored and maintained during navigation. The computational overheads of updating the filter at each step would significantly impinge on the filter's ability to perform real-time tracking. A solution to this situation that does permit continuous tracking without compromising the integrity of the filter is: during motion, update only those parts of the filter that are required for the current tracking task. A full update of the filter can subsequently be carried out when the robot stops moving at the next step, as ample processing time is then available.

The previous expressions lend themselves very efficiently to this type of update, because only those parts of the state vector and covariance matrix which are directly involved in the tracking process at that time need to be updated. These are the estimated states of the sensor and observed feature, $\hat{\mathbf{x}}_v$ *and* $\hat{\mathbf{y}}_i$ and the covariance elements $\mathbf{P}_{xx}$ , $\mathbf{P}_{xy_i}$ , and $\mathbf{P}_{y_i y_i}$ .

The algorithmic development for the continuous tracking of multiple features is continued in Appendix C. There it is explained that by storing a small amount of information at each transition step, the state and covariances can be updated in a generic way at the completion of each tracking motion. A criterion is also provided as to which feature to track during navigation, this is referred to as the "Vs criterion", and is based on known innovation covariance values of the features under consideration.

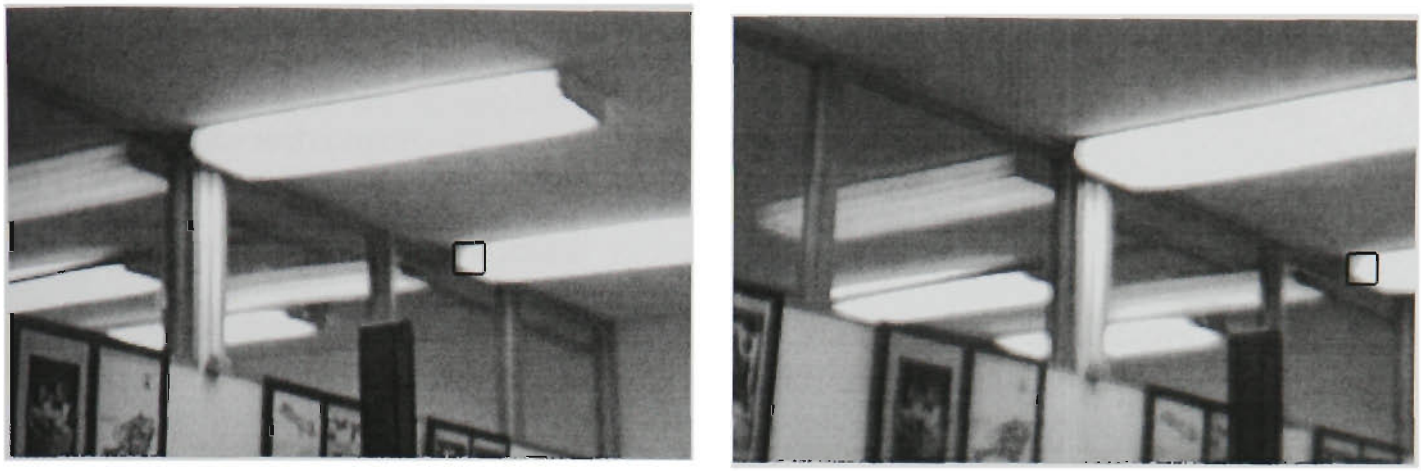
In Section 5.7 further details of the map-building process, and the strategies adopted are outlined.

### 5.6.1 Distance Increments in Place of Time Increments

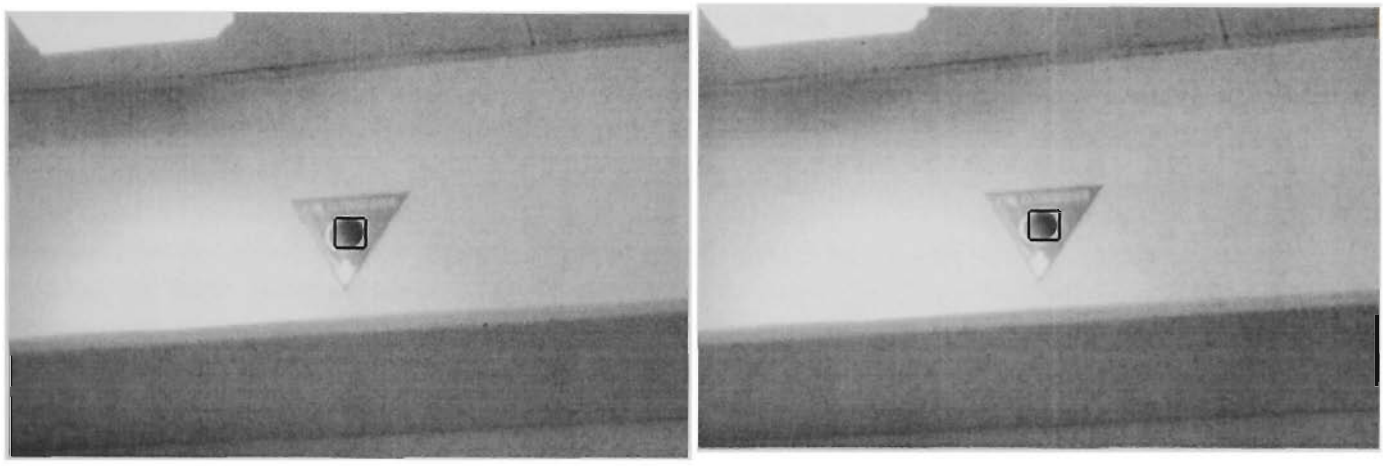
In the preceding analysis, events are governed by constant time increments of $\Delta t$. However, a more useful increment to trigger measurements is vehicle odometry (i.e., the trajectory being divided into steps of fixed wheel encoder counts, for this work the step size nominally used is 20cm). Distance increments are preferable to time increments, as these eliminate accumulation of errors due to velocity fluctuations during a time interval.

### 5.6.2 Advantage of Known Features

Incorporating Known-Features into the map building process has considerable advantages, akin to using recognised landmarks to reinforce one's own position when travelling in relatively unfamiliar territory away from home. The process of acquiring a known-feature consists of manually driving the robot to a suitable position and pointing the head in the direction of a likely suitable feature (i.e., one with high contrast). If the feature is successfully matched and measured, its image patch is saved. Several examples of such feature patches are shown in Fig 5.6.2.1. The saved patch together with its world coordinates and the robot's position are stored in a file of known-features. Known features are initialised into the state vector in the same way as naturally found features (i.e., as feature i, having coordinates $\mathbf{y}_i$). The covariance $\mathbf{P}_{y_i y_i}$ however is set with all elements equal to zero, along with the cross-covariances between the feature-state and that of the vehicle and other features. This is justified as it is assumed that the location of the known feature is 100 percent certain.



(a)



(b)

**Fig 5.6.2. 1 Example of saved known features (a) Light fitting ends and (b) Fire extinguisher sign**

### 5.6.3 Simplified Mobile Base Trajectory Motions

Motion of the mobile base is governed by the control vector $\mathbf{u} = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}$ and while the analysis in the preceding sections places no restriction on the movement of the mobile base, motions comprising pure rotations and linear path trajectories are the simplest and most effective in practice. A pure rotation is accomplished by setting $v_1 = - v_2$ while for translation $v_1 = v_2$. In tests of the mobile base navigating to designated waypoints, the magnitude of the steady state velocity is set to a constant value with angular rotations and number of steps the only trajectory variables required to be calculated by the software controller. Generally, trajectories from one waypoint to the next consist of an initial rotation about the central axis of the base, a translation

directly to the waypoint, followed by a further rotation about the central axis of the base, to align the it with the specified orientation. Occasionally when differences between odometric and position estimates, based on feature measurements exist, the assumption is made that the odometry is in error and corrective movements are made.

## 5.7 Map-Building Strategy Adopted

The final map-building and tracking strategy adopted and the rational for the decision making process, are outlined in this section. These stem from the derivations detailed in Appendix C.

The map-building process commences when the robot is about to move for the first time. In searching for suitable scene-features, in an effort to ensure a wide selection is found, the stereo head is pointing slightly up, and is first directed to the left, then straight ahead and finally to the right. In principal, during this search up to three new features could be found and added to the map. While this is unlikely in view of suitable features being somewhat scarce, the rule that a new feature be added if less than 2 features are visible at any given location has been adopted. This rule ensures a sparse map is created, furthermore in maintaining the map it is important that obsolete features be removed. Features are deleted if out of 10 measurement attempts more than half are unsuccessful.

Which features are subsequently chosen for state vector updating, is determined by assessing the uncertainty of their position relative to the robot. This measure of uncertainty is obtained by comparing the scalar space volume measure Vs, which is related to the innovation covariance values of the feature. The principle adopted is that the 'best' feature to measure (for the long-term integrity of the map as a whole) is the one with the least certainty, as the measurement process will improve its estimated position, thus reducing the overall uncertainty of the state vector. For instance if a feature is known with zero uncertainty, a further measurement of the feature cannot improve on the uncertainty of its position. While tracking a feature however, some cost needs to be attributed to saccading to a new feature as each saccade encroaches on the total navigation time.

The approach adopted is to decide while the robot is moving, whether to switch to a new feature or stay tracking the existing one. The choice is based on the predicted robot and map-state that would exist in a short future interval of time if (a) the saccade and measure of the chosen feature was made or (b) tracking of the existing feature continued. The "short future interval of time" is the estimated time it would take to saccade and measure any of the features.

Formally the decision process is implemented according to:

1. $N_i$, the number of measurements that would be lost while saccading to each of the visible features is calculated. To determine this, an estimate of head movement time to correctly move to and locate each feature is required.

2 Ascertain the largest $N_i$ ($N_{max}$), this represents the largest number of measurements lost during the longest saccade.

3 If an immediate saccade is to be made, estimate for each feature i the state that would exist after $N_{max}$ + 1 steps.

4. Evaluate Vs(max) for each of the above estimated states and saccade to the feature with the lowest Vs(max) unless the Vs(max) of the feature being tracked is the lowest.

## 5.8 Software Development

### 5.8.1 Development and Modification of Navigation Software

As previously stated the development of the navigational software is based on SCENE [Dav01]. The motion models outlined in this Chapter and Appendix C were developed for the TRC mobile base and Bisight head, and these replaced SCENE's existing "default" models where appropriate. Furthermore, because of the trajectory simplifications outlined in Section 5.6.3 the "Control Parameter Section" of the interface GUI was able to be reduced to contain only the Displacement (m), Steering Increment (degrees) and Number of steps. (Velocity, Turret increments, and Time step slide adjustments controls were removed).

In addition to the above and title name changes, the following functions and associated interface buttons were added:
(a) Obstacle Avoidance (Navigate around obstacles),
(b) Recognition

The interface GUIs are shown in Figs 5.8.1.1 to 5.8.1.3.

#### 5.8.1.1 Navigating to Specified Destinations (Waypoints).

The Waypoint Browser (Fig 5.8.1.3) holds the locations of sequential stopping destinations. Navigation to the next waypoint commences on the mouse click of button " Navigate to Next Waypoint" (Fig 5.8.1.2). In this case the route is assumed obstacle free.
Scanning for target objects commences on the mouse click of button "Recognition" (Fig 5.8.1.2)

#### 5.8.1.2 Obstacle Avoidance Function.

If obstacles are likely, then instead of clicking " Navigate to Next Waypoint", the button "Navigate Around Obstacles" is selected. With this selected, prior to the commencement of each step the head tilts down and checks for an obstacle. If one is detected within a depth of 1 metre an additional waypoint is automatically inserted. Fig.5.8.1.4 illustrates this process

Since the search for obstacles occurs at each step, the time taken to navigate between waypoints increases proportionally to the distance travelled.

### 5.8.1.3 Dead Reckoning Option

A command line option has also been provided for use when neither mapping nor obstacle avoidance is required (i.e., SLAM not required). For this "Dead Reckoning Option", the base simply relies on its odometry encoder values for localisation. It is the fastest mode of path navigation, and is quite reliable when the terrain is a level floor. Flow Chart Fig 5.8.1.5 summarises the options.

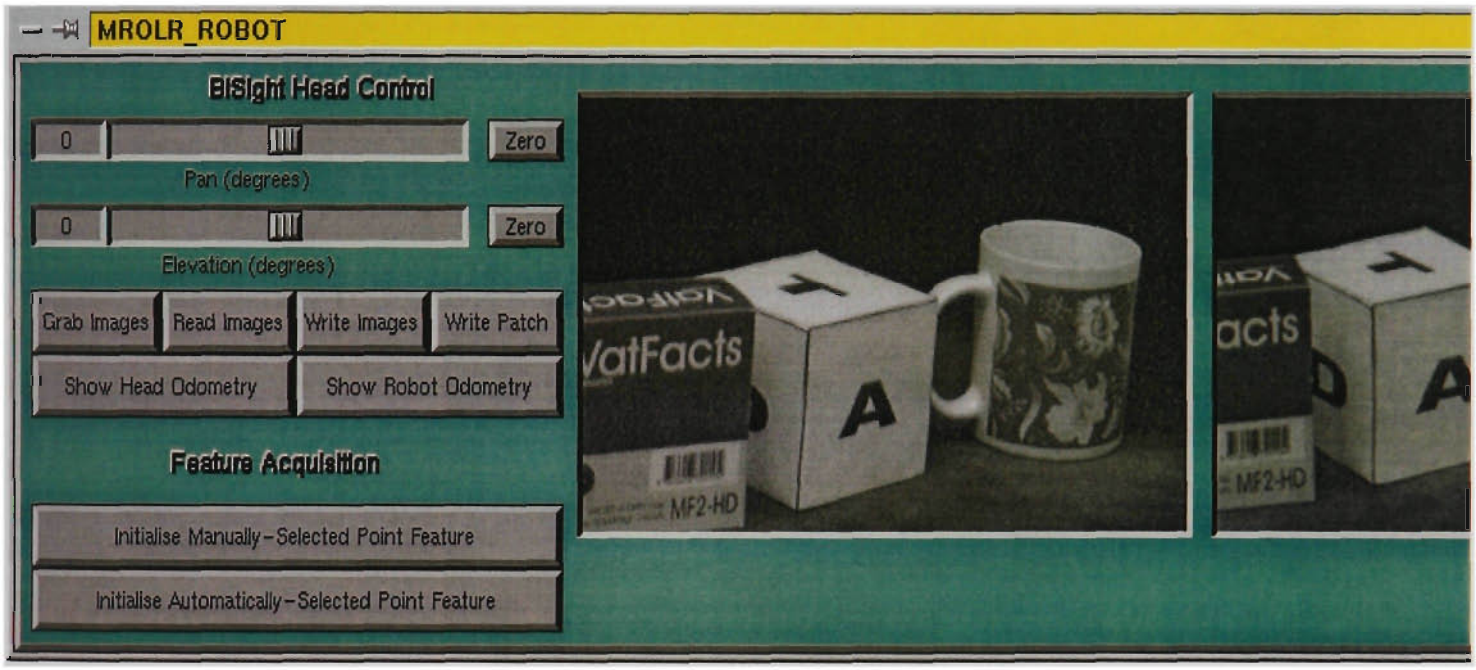



**Fig 5.8.1. 1 Head control GUI**

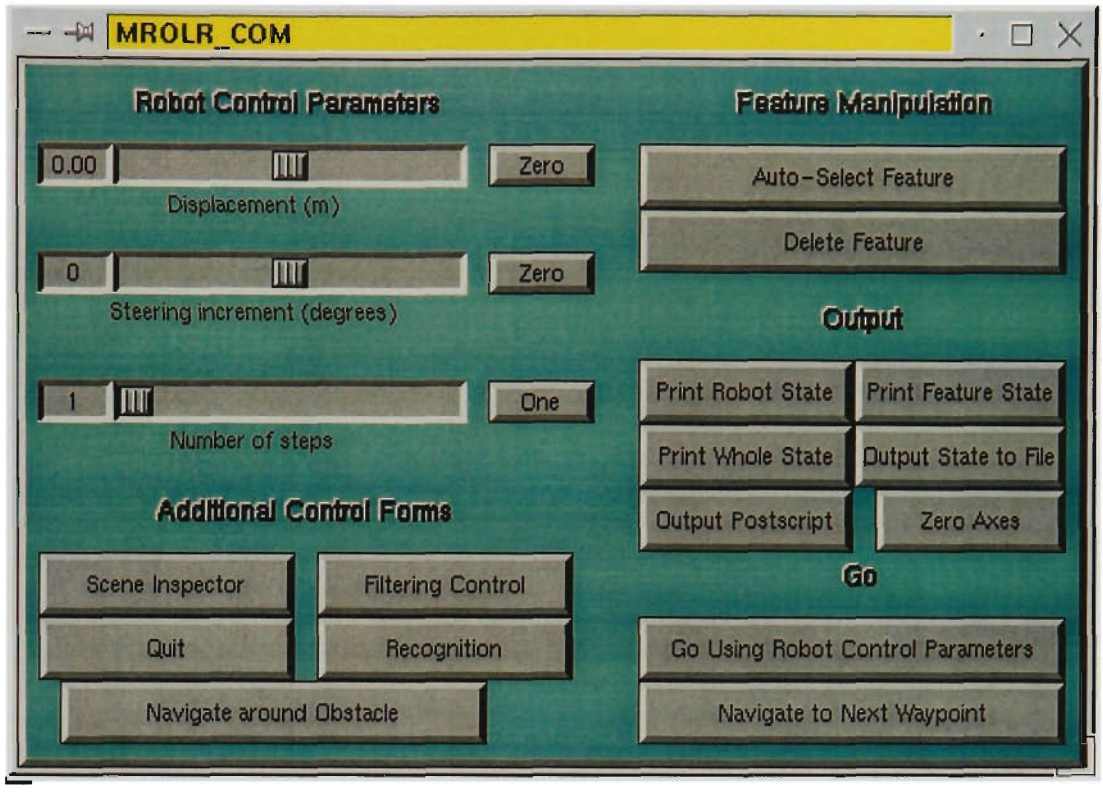



**Fig 5.8.1. 2 Navigation GUI**

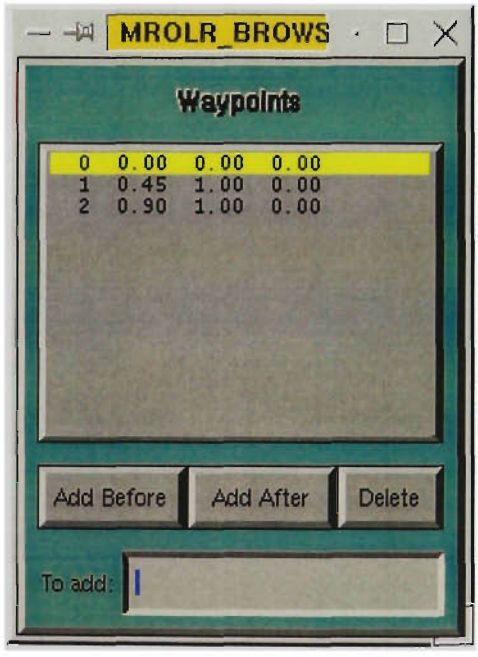
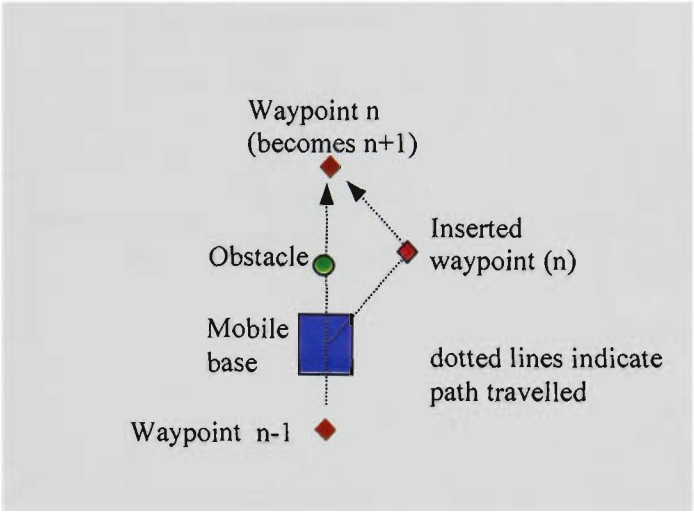



**Fig 5.8.1. 3 Waypoint browser GUI**




If an obstacle is detected, an additional waypoint is inserted to the right or left side of the obstacle, depending on perceived space

**Fig 5.8.1. 4 New waypoint creation for obstacle avoidance**

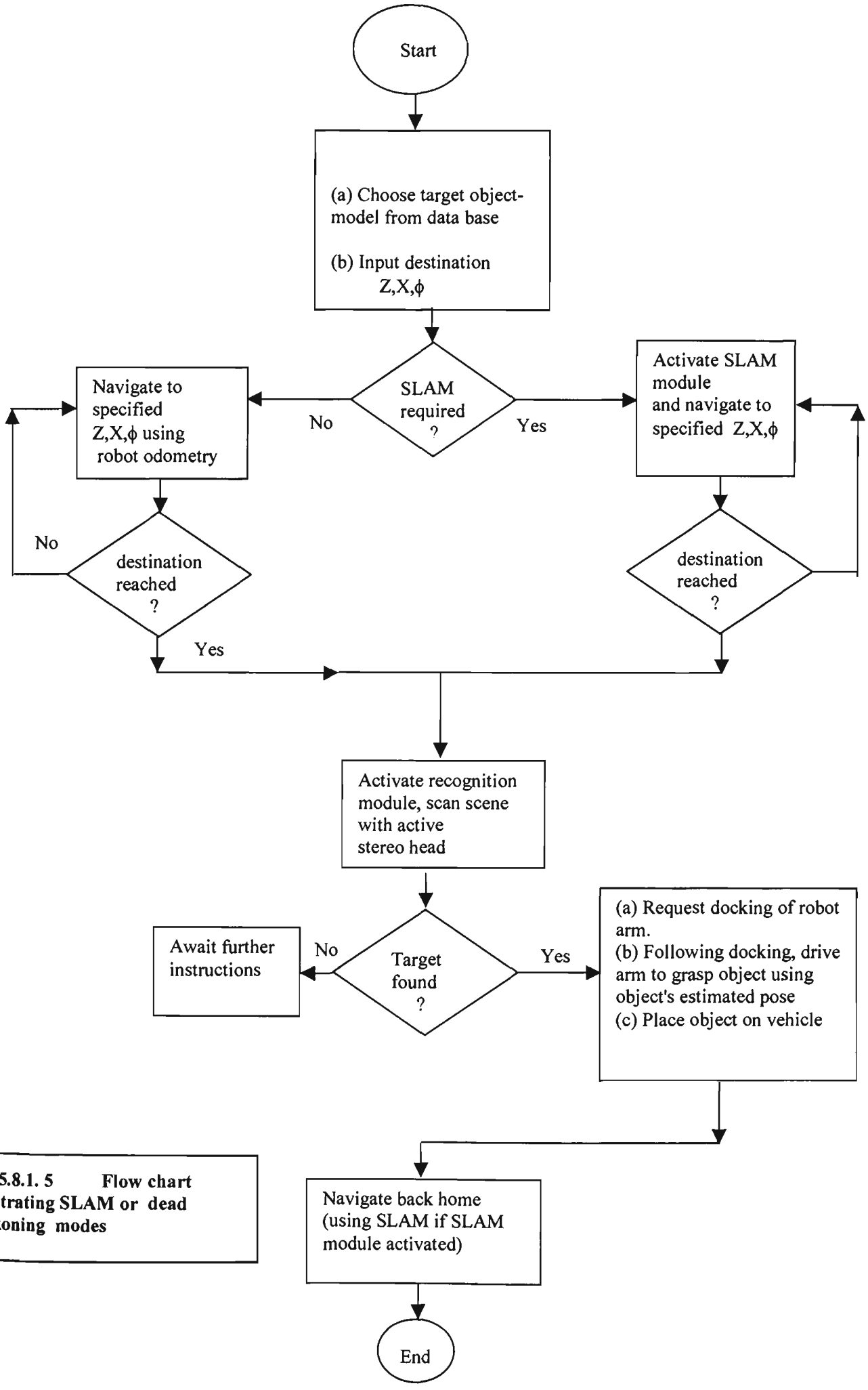



**Fig 5.8.1. 5 Flow chart illustrating SLAM or dead reckoning modes**

## 5.9 Data Measurement Examples

Examples of data measurements, for a map size of 2 features, at step 3 are given below. Note the number of significant figures printed do not reflect the accuracy of the data.

**Step 3**

**No of features 2**

**State vector**

$\hat{x}$ =

0.884382
0.993516
0.0305666
1.04299
1.04177
6.24433
-1.04419
1.04236
4.1569

**Covariance Matrix**

P = [

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0.000472303 | 0.000224005 | -2.74943e-05 | 8.00617e-05 | 5.45943e-05 | 0.000602076 | -0.00023809 | 0.000196054 | 0.000953954 |
| 0.000224005 | 0.000289892 | -2.07999e-05 | 0.000187995 | 0.000195976 | 0.00127189 | -0.000198764 | 0.00020446 | 0.000867573 |
| -2.74943e-05 | -2.07999e-05 | 1.33371e-05 | 3.07865e-05 | 1.31344e-05 | .2439e-05 | 5.32473e-07 | 3.16899e-06 | 6.16399e-06 |
| 8.00617e-05 | 0.000187995 | 3.07865e-05 | 0.00036703 | 0.000285048 | 0.00166124 | -0.000203065 | 0.000229519 | 0.00093282 |
| 5.45943e-05 | 0.000195976 | 1.31344e-05 | 0.000285048 | 0.000701177 | 0.00211657 | -0.000166389 | 0.000184586 | 0.000733035 |
| 0.000602076 | 0.00127189 | 5.2439e-05 | 0.00166124 | 0.00211657 | 0.0124287 | -0.00109314 | 0.00117172 | 0.00475933 |
| -0.00023809 | -0.000198764 | 5.32473e-07 | -0.000203065 | -0.000166389 | -0.00109314 | 0.000313161 | -0.000255231 | -0.00107892 |
| 0.000196054 | 0.00020446 | 3.16899e-06 | 0.000229519 | 0.000184586 | 0.00117172 | -0.000255231 | 0.000393117 | 0.000984229 |
| 0.000953954 | 0.000867573 | 6.16399e-06 | 0.00093282 | 0.000733035 | 0.00475933 | -0.00107892 | 0.000984229 | 0.00420907 |

]

**Eigenvalues associated with each feature measurement**

0.000112516 4.36763e-05 4.23311e-06
0.000110681 4.42953e-05 4.26137e-06

Best score found: 6.0545e-07.
Auto-selected feature with label 1.

**Covariance matrix of noise**

R=[

| | | |
|---|---|---|
| 4e-06 | 0 | 0 |
| 0 | 0.0001 | 0 |
| 0 | 0 | 4e-06 |

]

**Innovation Covariance Matrix**

S = [

| | | |
|---|---|---|
| 4.42963e-05 | -3.15737e-07 | -1.4381e-07 |
| -3.15737e-07 | 0.000110679 | 4.00568e-08 |
| -1.4381e-07 | 4.00568e-08 | 4.2619e-06 |

]

## 5.10 Conclusion

With the navigational formulations in this chapter complete, the remaining algorithms that still require to be developed relate to:

- target object-modeling,
- scene-modeling,
- target object recognition,
- target object retrieval, using the docked robot arm.

Each of these tasks, together with the provision of a "3D Virtual Environment" for simulating MROLR, are considered in the following chapter.

# Chapter 6 Object Modeling, Recognition and Retrieval

## 6.0 Introduction

This chapter considers object and scene modeling, recognition, and retrieval of the located object from the scene.

Section 6.1 outlines an automated 3D object-model producing facility, designed to produce an object-model data base comprising all objects that MROLR is likely to be requested to locate and retrieve on command. Section 6.2 includes the development of a 3D object-model editor. Sections 6.3 to 6.5 outline the development of the object recognition module that is responsible for matching the requested object with objects found in the scene. For a successful match, the appropriate transform (and consequently the object's pose) to transport the object-model into the scene is also provided.

Sections 6.6 and 6.7 deal with the docking of the robot arm, and the establishment of the additional transforms necessary to guide the arm towards the specified object's grasp-point and finally retrieve the located object.

Section 6.8 discusses the development of a "3D Virtual Environment" for simulating MROLR.

## 6.1 Automated Model Creation

The recognition process involves the matching of an a prori specified object-model with a scene-model. Methods of building object-models were reviewed in Chapter 2 where it was concluded that for the current application, a 3D wire-frame (edge-based) model was a suitable choice (section 2.2.7). Reasons for this choice of modeling are:

(a) Matching is carried out in 3D Eucledean space which ensures angular and linear measurement invariance (i.e., relative angles between lines, and arc lengths remain fixed). This is essential, as the object in the scene will in general have an unknown orientation.

(b) Matching in 3D Eucledean space permits positional information to be calculated in terms of a homogeneous transform giving orientation and translation details of the matched scene-object relative to the model-object. This transform is available for transporting the model into the scene allowing visual verification of the quality of the match. It is also a necessary component for obtaining the resulting transform that guides the robot arm towards the object in the scene for retrieval.

(c) It requires only sparse data matching thereby keeping search and computational costs to a minimum. Given the large number of items available in any given scene, a non-sparse data modeling would be prohibitive.

One obvious disadvantage of this modeling approach is the small number of available features for matching (i.e., lines and conics). Additional local characteristics such as object colour, texture and surface properties could be added to the list of features to increase the robustness of the matching process. The addition of these will be considered for future work when faster processors should become available.

The following briefly describes the need for models created from several views and then describes the methodology of the model creating process. If two images of a scene are captured using a pair of calibrated cameras, edge based binocular stereo triangulation can be used to obtain a partial 3D edge outline of the scene as depicted in the left (or right) camera image. This is of limited use for 3D recognition however as local features associated with the object (or scene) are only depicted from one view. Thus if a solid cube object (Fig 6.1.1) has letters 'A' printed on the front and 'B' on the back, the letter 'B' may not be visible from a particular view. If both letters were required for recognition, the block would not be identifiable.

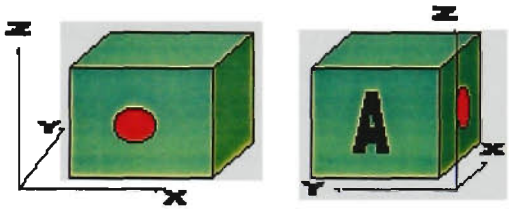



**Fig 6.1. 1 Features remain static w.r.t object's frame**

A 3D outline of the complete cube (i.e., hidden edges included) on the other hand, would ensure all its surface features were visible, and would thus constitute a suitable object-model for recognition.

To facilitate automated object 3D-model creation, a PC controlled rotating horizontal table and two cameras mounted on extended rails, was used to obtain a sequence of stereo images of desired objects (Fig 6.1.2). Two smaller portable PC controlled tables, described below, were also subsequently built and used.

**Fig 6.1. 2 Models created on computerised rotating table**

From each stereo pair of images corresponding to a rotated view, a 3D partial edge outline was derived and then integrated to form a complete model. The coordinate frame for each view was (by default) with respect to the left camera's optical centre. Merging the independent views requires all re-constructions to be with respect to a unique (although arbitrary) coordinate frame attached to the object. As the object rotates with the table, so too does its coordinate frame, thereby ensuring the object's physical dimensions and features remain static with respect to that frame (Fig 6.1.1). These would however change with respect to any stationary reference frame. The world reference was chosen so that its Z-axis coincided with the table's central axis of rotation. The object's arbitrary coordinate frame is aligned with the world coordinate frame for a table rotation angle of zero.

As the final object's reconstruction is required with respect to its own coordinate frame, two transformations of 3D edgels are required. The first from the left camera coordinate frame to the static world reference frame, and then a simple rotational transformation (about the world Z-axis) to transform the edgels to the object's coordinate frame. Using a calibration tile (Fig 3.1.2) whose Z-axis also coincides with the table's centre, and a knowledge of subsequent angles of table rotation, enable the required 4x4 (homogeneous) left camera to world, and world to object transformation matrices to be determined. Each edgel of the reconstruction is subsequently transformed to the desired rotating reference frame and combined to form the 3D models.
The final stage of model creation requires the manual formation of:

(a) Groups (termed cliques) consisting of a distinctive focus feature (such as an arc length or long line) and any number of surrounding features in close proximity.

(b) Grasping location data. Each object-model is given a nominal (X, Y, Z) point together with optional[1] yaw, pitch, roll, guidance angles. This kinematic information in the form of a homogeneous transform is supplied to the robot arm at the time of retrieval.

The clique size and number is essentially determined by the availability of features, moreover, a focus feature in one clique may be used as a surrounding feature of another clique. Cliques are stored and associated with the model, and they provide the features to be matched during the recognition phase. Further details are given in section 6.3. Examples of cliques are given in Fig 7.1.12. Fig 6.1.3 depicts created 3D-models of a cup and an alphabet cube.

The above procedure was coded and added to TINA's Matcher tool source code. Further details of this are provided in appendix B. Initially, three objects, an alphabet cube, a mug, and a teapot were produced for test purposes. These models comprised a simplified database of objects to be recognised in a scene. Fig 6.1.4 shows the stereo images of a scene containing these 3 objects and the subsequent scene-model produced from the images.

[1] For most cases gripper orientation angles of $0^{\circ}$ or $90^{0}$ are specified.

*(a)* *(b)*

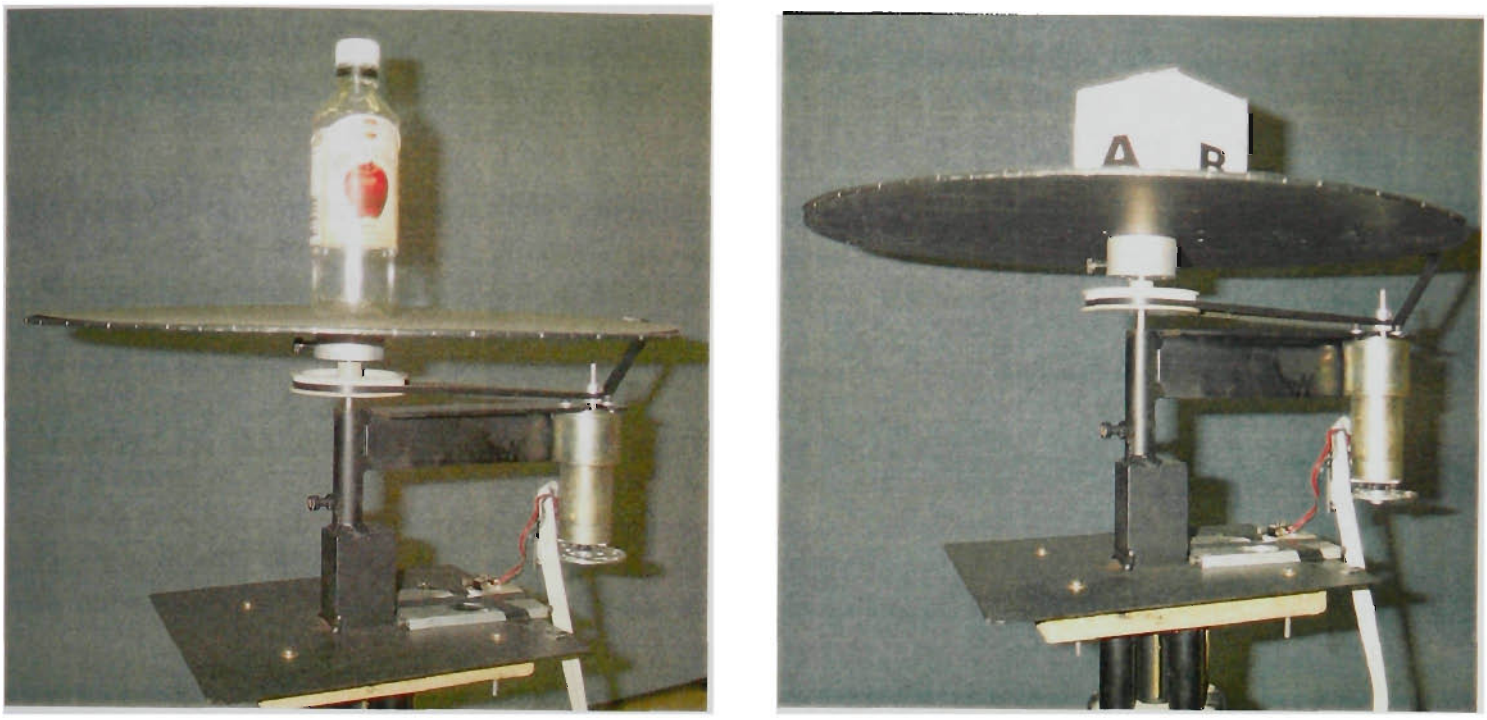
**Fig 6.1. 3 3D created model of (a) cup and (b) cube**


*(a)* *(b)*

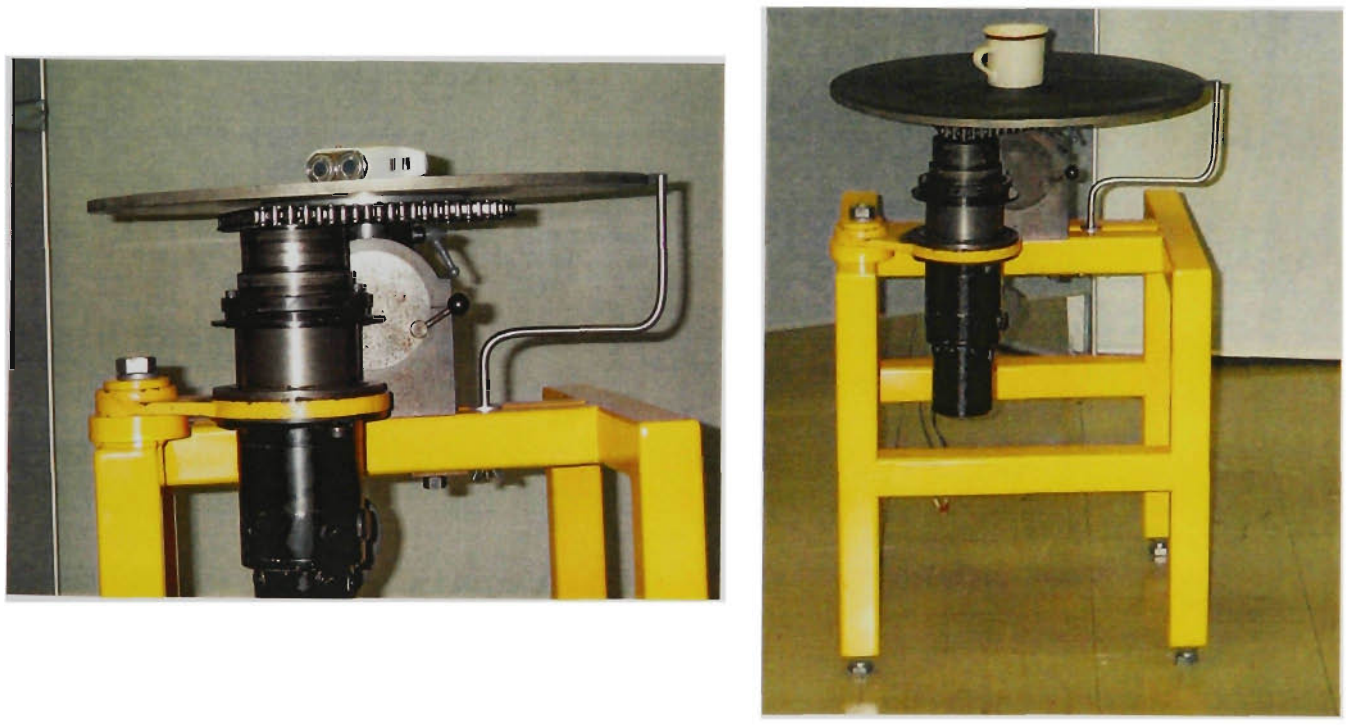
**Fig 6.1. 4 (a) Stereo images of scene (b) 3D scene model**

## Portable Tables for Model Creation

Two portable computer-controlled tables considerably smaller than the one shown in Fig 6.1.2, were later designed and constructed. One being light and easy to move (Fig 6.1.5 a), the other robust, and fitted with level and height adjustment (Fig 6.5.1 b). These permit the cameras mounted on MROLR's stereo head to be used for the automated model creation. The main benefit of these tables is their portability, and the elimination for the need of a duplicate set of cameras. Subsequent models were created using these. Appendix B provides additional details.



(a) two views of the lightweight model creation turntable



(b) Two views of the heavy duty model creation turn table with adjustable height level

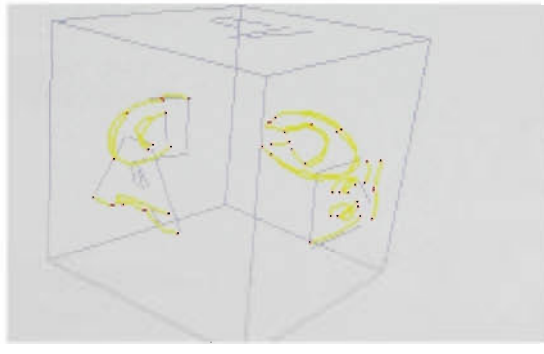
**Fig 6.1. 5 Portable model creation tables**

## 6.2 Model Enhancement – Model Editor

A Model Editor was developed to facilitate the manual removal and addition of 3D segments in formed models. This editor utilises many of the 3D geometric list structures library functions provided with TINA and in particular those associated with TINA's geomstat tool. At the commencement of this work, the geomstat tool was at an incomplete stage having no input/output stream facility. However numerous functions for 3D line manipulation were coded, and a relatively small amount of supplementation made it useable as a model editor. This additional code was added and linked to form a module titled "Model Editor".

Editing of models is desirable on occasion as double lines or conics arise as a result of camera lens distortions and/or calibration inaccuracies, or lines are fragmented or omitted altogether due to occlusions. An example of the editor's application to one of the test cubes is shown in Fig 6.2.1. While such enhancements are aesthetically pleasing they provide an additional processing overhead and in general are found to provide little or no improvement in recognition capability. This stems from the fact that scene-models (produced while panning and therefore not available for editing) frequently contain broken or missing edge segments (Fig 6.1.3).


*(a) Editor (Edited Model, Conics not displayed)*



*(b) Resulting Model*

**Fig 6.2. 1 Application of model editor to a raw 3D cube model**

### 6.2.1 TINA's Geometric List Structures

Five 3D geometric list structures used throughout much of this work, in particular for model creation, editing and object recognition are: vectors, points, lines, planes and conics (i.e., curve geometry such as arcs, ellipses, circles etc). TINA's software infrastructure encourages the use of these list structures. Details (extracted from TINA's User Guide) are given in Fig 6.2.2.

```
typdef struct vec3
{
Ts_id ts_id ;          /* Tina structure identifier */
float el[3] ;                  /* point in 3D coordinates */
} Vec3;

typdef struct lines3
{
Ts_id ts_id ;          /* Tina structure identifier */
unsigned int type;
unsigned int label;
struct vec3 p1,p2;             /* end points */
struct vec3 p;                         /* point on line */
struct vec3 v;                         /* direction vector
*/
float length;                          /* line length */
struct list *props;      /* property list used to associate
                  extra specific information */
} Line3;


typdef struct plane
{
Ts_id ts_id ;          /* Tina structure identifier */
unsigned int type;
unsigned int label;
struct vec3 p;                 /* point on the plane */
struct vec3 n;                 /*   a normal to the plane*/

struct list *props;  /* property list used to associate
                        extra specific information */
} Plane;
```

```
typdef struct conic
{
Ts_id ts_id ;          /* Tina structure identifier */
unsigned int type;
unsigned int label;
int filler1;           /* SUN platform dependance
                        parameter fix*/
double a, b, c, d, e, f;  /* algebraic formulae */
double theta, alpha, beta;
struct vec2 center ;
int filler2;           /* SUN platform dependence
                       parameter fix*/
double t1, t2;         /* conic params of p1 and p2 */
int branch;            /* for hyperbola only */
List *props;           /* property list used to associate
                       extra specific information */
} Conic

typedef struc conic3
{
Ts-id ts_id;                   /* Tina structure identifier */

int type;
struc conic *conic;    /* describes the conic
                                in the plane */
struct vec3 origin;    /* origin point on plane */
struct vec3 ex, ey, ez; /* define x,y,z axes:
                                ez = normal */
} Conic3;
```

The coordinate frame in which the **conic** lies is at origin and has axis **ex**, **ey** and **ez**. Note the 2D conic lies in the **xy** plane.

In conic the arc covered by the curve is between angles **t1** and **t2** in radians. All angles are positive with 0 along the x axis and times Pi/2 along the y axis. The arc goes from **t1** to **t2** in strictly ascending order, **t1** will be less than 2*Pi but **t2** may be larger than 2*Pi if the arc passes back through the x axis.
The implicit algebraic equation that represents its form is given by $ax^2 + 2bxy + cy^2 + 2dx + 2ey + f = 0$;
**Theta** is the angle of the major axis, **alpha** and **beta** are the lengths of the sem-major and semi-minor axis respectively,

The conic **type** can be any one of :
**ELLIPSE**
**HYPERBOLA**
**DEGENERATE**

The field branch is used to differentiate which part of the hyperbola the data lies on.

**Fig 6.2. 2 TINA 3D geometric list structures**

## 6.3 Recognition Algorithm

The recognition algorithm, in TINA's Matcher-tool module, written by Pollard et al [PPMF87]. [PPM91] was extended by the addition of code to both automate the recognition process and to include it as an integrated module of MROLR. This was necessary to facilitate the autonomous operation of the system. Recognition is based on pairwise matching of primitive geometrical features associated with a model and the scene (termed cliques). For object-models cliques are manually chosen following their creation (refer section 6.1), and consist of a distinctive focus feature (such as an arc length or long line segment) and a number of surrounding local features in close proximity. Object models in general require sufficient cliques so that adequate features maybe matched in a scene, irrespective of the object's orientation. Focus features provide a basis for determining maximal consistency of neighbouring elements within a clique. Features comprise 3D line segments and or 3D conic segments.

During matching three pairwise relationships are tabled. These are (a) **orientation**, (b) **minimum distance apart**, and (c) **distance from minimum distance apart**. These geometrical relationships are illustrated for a line feature in a model and a scene in Fig 6.3.1. TINA 3D straight-line segment structure formats, in addition to an Id label identifier, are represented by the quadruple ($l_1$, $l_2$, $e_1$, $m_1$) (see Fig 6.2.2 for code details). These parameters represent the two end vector points $l_1$ and $l_2$, the direction vector between them $e_1$ and the centroid of the line, its midpoint $m_1 = (l_1 + l_2)/2$.

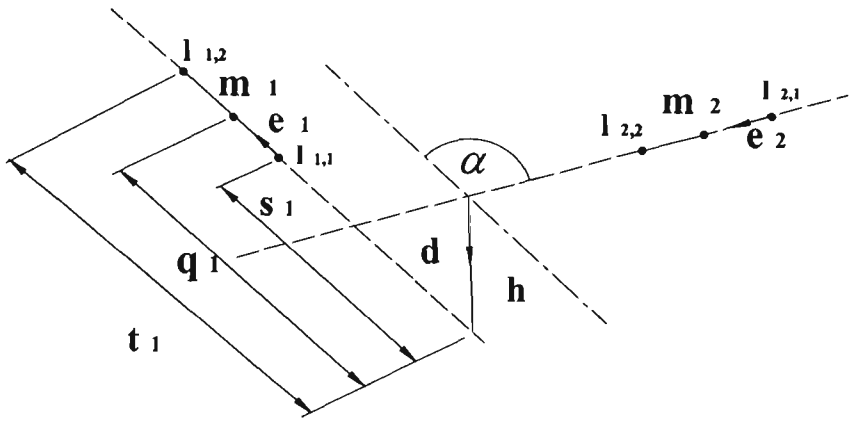



**Fig 6.3. 1 Matching of line segments**

**(a) Orientation difference between two pairs of potential matches** $\alpha$ : This is obtained from the dot product of $\mathbf{e}_1$ and $\mathbf{e}_2$ ie $\alpha = \cos^{-1}(\mathbf{e}_1 \cdot \mathbf{e}_2)$.

**(b) Minimum distance apart between (extended) lines d:**

This is represented by the unit vector (normal to each line) **d,** where:

$\mathbf{d} = (\mathbf{e}_1 \times \mathbf{e}_2)/|\mathbf{e}_1 \times \mathbf{e}_2|$ and the scalar distance between the lines h, where:

$h = (\mathbf{m}_2 - \mathbf{m}_1) \cdot \mathbf{d}$. If the line segments are close to parallel the distance used is the perpendicular distance between the lines:

$h = |(\mathbf{m}_2 - \mathbf{m}_1) - [(\mathbf{m}_2 - \mathbf{m}_1) \cdot \mathbf{e}_1]\mathbf{e}_1|$.

(c) **Distance from minimum distance apart** s, t, q: For non-parallel lines, the distances along the lines to the start and finish of each line, from the point of minimum distance apart is used. From the above, the vector between the points of minimum distance apart is given by: $\mathbf{h} = h\mathbf{d}$. Moving $\mathbf{m}_2$ to $\mathbf{m'}_2$ by adding $-\mathbf{h}$ produces a line $(\mathbf{e}_2, \mathbf{m'}_2)$ such that lines $(\mathbf{e}_1, \mathbf{m}_1)$ and $(\mathbf{e}_2, \mathbf{m'}_2)$ are coplanar and meet at the point of minimum distance apart on $(\mathbf{e}_1, \mathbf{m}_1)$.

$$q_1 = \{(\mathbf{e}_2 \times \mathbf{e}_1) \cdot [\mathbf{e}_2 \times (\mathbf{m'}_2 - \mathbf{m}_1)]\} / |\mathbf{e}_2 \times \mathbf{e}_1|^2$$

is the signed distance to that point from $\mathbf{m}_1$ in the direction $\mathbf{e}_1$. Scalar distances from $\mathbf{l}_{1,1}$ and $\mathbf{l}_{1,2}$ to that point are given by

$s_1 = q_1 + (\mathbf{m}_1 - \mathbf{l}_{1,1}) \cdot \mathbf{e}_1$ and

$t1 = q_1 + (\mathbf{m}_1 - \mathbf{l}_{1,2}) \cdot \mathbf{e}_1$, respectively.

Similarly the distances to the point of minimum distance apart along line2 are

$s_2 = q_2 + (\mathbf{m}_2 - \mathbf{l}_{2,1}) \cdot \mathbf{e}_2$ and

$t_2 = q_2 + (\mathbf{m}_2 - \mathbf{l}_{2,2}) \cdot \mathbf{e}_2$

Prospective matches for each pair of elements $\{(s_i,t_i), (s_{i+1},t_{i+1})\}$ from the object model description can be tested for geometrical correspondence with each pair of elements in the scene model.

### 6.3.1 Stored Table of $\{(s_i,t_i), (s_{i+1},t_{i+1})\}$

An advantage of the use of the pairwise elements $\{(s_i,t_i), (s_{i+1},t_{i+1})\}$ is that they can be calculated separately for each pair of lines and stored in look-up tables. To accommodate errors and scale differences, an overlap range is provided. Errors are accounted for as follows:

Pairs of lines are allowed errors $|\eta_1| < \beta_1$ and $|\eta_2| < \beta_2$ on the location of their centroids ($\mathbf{m}_1$ and $\mathbf{m}_2$) and direction vectors ($\mathbf{e}_1$, $\mathbf{e}_2$). The latter allowable errors are in terms of angles $\theta_1$ and $\theta_2$. On orientation differences, the interval is :

$$[\max(\alpha - \theta_1 - \theta_2, 0), \min(\alpha + \theta_1 + \theta_2, \pi)],$$

while for the minimum distance apart between (extended) lines, the interval is

$$h \pm (\beta_1 + \beta_2 + |q_1| \tan \varphi_1)$$

For $s_1$, $t_1$, and $s_2$, $t_2$ the permissible range is:

$s_1 \pm (\beta_1 + \beta_2 + |q_2| (\tan \theta_2 / \sin \alpha))$

$t_1 \pm (\beta_1 + \beta_2 + |q_2| (\tan \theta_2 / \sin \alpha))$

$s_2 \pm (\beta_1 + \beta_2 + |q_1| (\tan \theta_1 / \sin \alpha))$

$t_2 \pm (\beta_1 + \beta_2 + |q_1| (\tan \theta_1 / \sin \alpha))$

A similar range of overlapping intervals is provided for pairs of conic features (i.e., arc lengths and radii etc).

### 6.3.2 The Matching Process

Object model focus features are selected in sequence, closest matches to the selected focus feature in the scene model are considered as potential matches, and neighbouring features considered in terms of their pairwise geometrical relationships. Found cliques are ranked according to the highest number of neighbourhood matches consistent with the object model group size.

Transformations to place each found clique into its corresponding position in the scene are calculated using the method described by Faugeras et al [FHPP84]. Cliques that produce near identical transforms are considered as belonging to the same object. The mean transform that will transport the model into the scene is returned, facilitating the calculation of the pose of the object in the scene. The transform returned is the commonly used 4 x 4 homogeneous matrix comprising a 3x3 rotational submatrix and a 3x1 column translational vector.

The current sequence of matching a model with a scene is:
1. Build pairwise geometric tables
2. Match cliques
3. Compute the transform of the object model to carry it into the scene.
4. Transform the model into the scene (placing and displaying it over the recognised object. This provides visual verification of the quality of the match).

Associated with the matching process are seven parameters whose values can be adjusted to reflect the confidence in the scene data namely:
(a) Clique size, (b) position error, (c) rotation, (d) length threshold, (e) maximum rotation, (f) maximum translation, (g) length ratio.

## *6.4 Automation of Recognition Process*

To automate the recognition process the "current matching sequence" mentioned above was formulated into several nested iterative loops in which the 7 parameters are systematically modified in a "fuzzified" like, fine-to-course adjustment, to reflect a reduction in data confidence levels. Iteration continues until either recognition occurs, at which stage the process is exited, or the set number of iteration step levels is exceeded, and recognition is deemed to have failed. If recognition failure occurs, the next captured scene model is entered into the loop. This process proceeds as the pan-tilt-vergence head sweeps through a pan arc of –20 to +20 degrees.) tilt and vergence angles being held fixed. This is repeated at each scheduled observation station (waypoint).

Fig 6.4.1 depicts the recognised cup and alphabet cube transformed into the scene, note that pose and scale have been correctly identified.

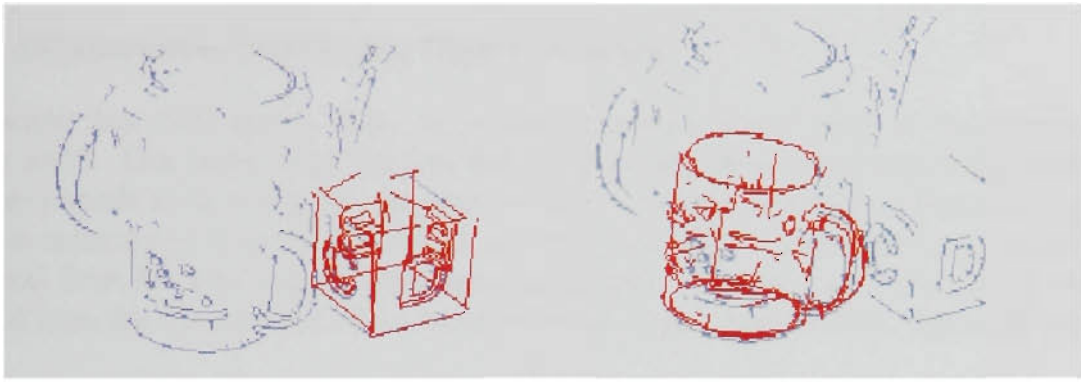


**Fig 6.4. 1 Model Cube and Cup scaled, and placed over objects in the scene**

## 6.5 Model Creation (An Alternative Method)

With the above matcher software it is possible to create a 3D model generating facility without the use of a rotating table. All that this requires is to form a sequence of single view models of arbitrary orientations (by manually turning the object about), and to match each to the first (reference) view. The returned transform for each match would be used to sequentially integrate the matched model with the reference. This method certainly has appeal as it makes the whole scheme more portable (in not needing the extra hardware) and more robust in that the orientation returned by the transform is likely to be quite precise. The drawback of this method is that it requires laborious manual intervention because as the model grows with each sequenced integration of single model views, new focus features and clique groups need to be found and stored, to ensure matched features in the next sequenced view. With the current model creating method, sequenced views are integrated automatically, and clique groups required to be found only for the fully integrated finished model.

## 6.6 Pose for Object Retrieval

Pose information to carry the object-model into the scene (as outlined in section 6.3.2) is returned in the form of a transformation matrix. This matrix provides the necessary rotational and translation information to transport the 3D model into the scene. Observing the object-model correctly aligned with the object in the scene is useful for visual verification of a correct match. To provide a suitable grasping point for the robot arm's gripper, each object-model has a designated X,Y,Z location and angular Yaw, Pitch, Roll, information stored at the time of object-model creation. This positional information is also transformed into the scene along with the model, the numerical value is obtained by multiplication of the grasp point vector by the transform, the resulting vector being $\mathbf{m}_L^L$ (see Fig 6.7.1). It is important to appreciate that the model points transformed into the scene are all initially referenced to the left camera's coordinate frame, which does not remain fixed because of head motion. The head-centre frame $C_0$, having its origin at the intersection of the pan and tilt axis does remain fixed with the heads movement (

Fig 6.7.1). Using the head-model derived in Chapter 4 (and more specifically equation 4.3.6) all transformed model points are referenced to frame $C_0$, $\mathbf{m}_L^L$ thus transforms into $\mathbf{m}_G^{C0}$.

### 6.6.1 An Alternative Method For Object Grasping.

The method described above relies on precise alignment and scaling of the transformed object into the scene. This stems from the fact that the grasping information originates from the object-model and needs to be accurately transported into the scene. A simpler alternative approach is to obtain an estimate of only the translation necessary for object grasping and subsequently pick up all objects from the top, using a wide opening gripper. Translational information can readily be obtained from the 3D centroid of the matched scene-object features (i.e., lines and conics) i.e:

For n matched centre points $\mathbf{m}_i$, $\mathbf{Centroid} = \frac{\sum_{i=1}^{n} \mathbf{m}_i}{n}$.

Unfortunately, wide opening and closing grippers are not readily obtainable.

## *6.7 Retrieval of the Object*

As our robot arm cannot be mounted onboard the mobile base due to space considerations, it is currently mounted on a caster platform and manually docked with the mobile base to facilitate the testing and development of the algorithms. The construction of a second (transport) robot mobile base is in progress (see Fig 6.7.2) and appendix B for design details), and in the next phase of this work, this will be summonsed by a wireless link to dock (automatically) with the main (scout) mobile base following object recognition.

When docked, the base frame of the robot arm remains at a fixed distance relative to the head-frame $C_0$, consequently since the located object's pose is known in this head-frame, it is a simple matter to obtain the pose of the object in the base frame of the arm.

Let $\mathbf{T}_{Object}^{C_0}$, $\mathbf{T}_{C_0}^{Arm}$, $\mathbf{T}_{Object}^{Arm}$ respectively represent the homogeneous transforms of the object with respect to the main head-frame, the main head-frame with respect to the robot arm base, and the object with respect to the robot arm base.

$$\mathbf{T}_{Object}^{C_0} = \begin{bmatrix} n_{x_o} & o_{xo} & a_{x_o} & p_{z_o} \\ n_{yo} & o_{yo} & a_{yo} & p_{yo} \\ n_{zo} & o_{z_o} & a_{zo} & p_{zo} \\ 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{T}_{C_0}^{Arm} = \begin{bmatrix} n_{x_a} & o_{xa} & a_{x_a} & p_{xa} \\ n_{ya} & o_{ya} & a_{ya} & p_{ya} \\ n_{za} & o_{z_a} & a_{za} & p_{za} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Where the 3 x 3 sub matrix comprising of column vectors **n o a** represent the orientation of the frame of interest (i.e., subscripted frame, object) with respect to the reference frame (i.e., superscripted frame, $C_0$) and column vector **p** represent the translation between the two frames.

The object in the base frame of the robot arm is obtainable from:

$$\mathbf{T}_{Object}^{Arm} = \mathbf{T}_{C_0}^{Arm}\mathbf{T}_{Object}^{C_0} \qquad \textbf{6.7.1}$$

If however $\mathbf{T}_{C_0}^{Arm}$ is not easily measured, it can be obtained with the aid of a test object from:

$$\mathbf{T}_{C_0}^{Arm} = \mathbf{T}_{Object}^{Arm}(\mathbf{T}_{Object}^{C_0})^{-1} \qquad \textbf{6.7.2}$$

To find $\mathbf{T}_{C_0}^{Arm}$ it is simply a matter of manually guiding the arm to pick up the test object, noting the arm X, Y, Z coordinates to achieve this, and then using equation 6.7.2. In the docked position the robot arm's reference coordinate frame was aligned parallel to the head-frame so that $X_{C_0} \rightarrow Y_{Arm}$ $Y_{C_0} \rightarrow Z_{Arm}$, $Z_{C_0} \rightarrow X_{Arm}$. The relative orientation of the head-frame with respect to the robot arm is shown in Fig 6.7.1. The associated transform $\mathbf{T}_{C_0}^{Arm}$ was determined to be:

$$\mathbf{T}_{C_0}^{Arm} = \begin{bmatrix} 0.0 & 0.0 & 1.0 & -627.0 \\ 1.0 & 0.0 & 0.0 & -1040.0 \\ 0.0 & 1.0 & 0.0 & 711.0 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{bmatrix}, \text{ distances specified in mm.}$$

The transforms were verified by gasping an object with the following position variables, supplied by the arm's controller, and comparing the results obtained from transform products.

| X | Y | Z | Roll | Pitch | Yaw |
|---|---|---|---|---|---|
| 220.0 | (-570.0) | 270.0 | * | * | * |

* as required for grasping

Choosing a local object coordinate frame (i.e., frame attached to object) with the orientation shown in Fig 6.7.1,

$$\mathbf{T}_{Object}^{Arm} = \begin{bmatrix} -1.0 & 0.0 & 0.0 & 220.0 \\ 0.0 & 1.0 & 0.0 & -570.0 \\ 0.0 & 0.0 & -1.0 & 270.0 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{bmatrix}$$

and

$$\mathbf{T}_{Object}^{C_0} = \begin{bmatrix} 0.0 & 1.0 & 0.0 & 47.0 \\ 0.0 & 0.0 & -1.0 & -441.0 \\ -1.0 & 0.0 & 0.0 & 847.0 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{bmatrix}$$

using $\mathbf{T}_{Object}^{Arm} = \mathbf{T}_{C_0}^{Arm}\mathbf{T}_{Object}^{C_0}$

results in

$$\mathbf{T}_{Object}^{Arm} = \begin{bmatrix} -1.0 & 0.0 & 0.0 & 220.0 \\ 0.0 & 1.0 & 0.0 & -570.0 \\ 0.0 & 0.0 & -1.0 & 270.0 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{bmatrix}$$ as expected.

Several test runs were carried out using this "parallel" arm orientation (see Fig 7.2.12), however the arm's reach restricted the retrieval of scene-object to being in very close proximity.

To obtain greater reach, the robot arm was rotated 45 degrees towards the head.
(i.e., $T_{Arm}^{C_0}$ requiring a 3x3 orientation sub matrix as below, ? values to be determined).

$$\mathbf{T}_{Arm}^{C_o} = \begin{bmatrix} -.707 & 0.707 & 0.0 & ? \\ 0.0 & 0.0 & 1.0 & ? \\ 0.707 & 0.707 & 0.0 & ? \\ 0.0 & 0.0 & 0.0 & 1.0 \end{bmatrix}$$

Once again $\mathbf{T}_{Object}^{Arm}$ was determined from the arm controller:

$$\mathbf{T}_{Object}^{Arm} = \begin{bmatrix} -.707 & 0.0 & 0.707 & 545.0 \\ 0.707 & 0.0 & 0.707 & 0.0 \\ 0.0 & 1.0 & 0.0 & 429.0 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{bmatrix}$$

The object with respect to the head was manually measured to give :

$$\mathbf{T}_{Object}^{C_0} = \begin{bmatrix} 1.0 & 0.0 & 0.0 & 747.0 \\ 0.0 & 1.0 & 0.0 & -246.0 \\ 0.0 & 0.0 & 1.0 & 1003.3 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{bmatrix}$$

(Note this time, an object-frame parallel to head frame $C_0$ has been arbitarily chosen, resulting in a unit 3x3 submatrix).

Using:

$$\mathbf{T}_{Arm}^{C_0} = \mathbf{T}_{Object}^{C_0}\,\mathbf{T}_{Arm}^{Object} = \mathbf{T}_{Object}^{C_0}\,(\mathbf{T}_{Object}^{Arm})^{-1}$$

$$\mathbf{T}_{Arm}^{C_0} = \begin{bmatrix} -.707 & 0.707 & 0.0 & 460.0 \\ 0.0 & 0.0 & 1.0 & -675.0 \\ 0.707 & 0.707 & 0.0 & 618.0 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{bmatrix}$$

Manual measurements confirmed these values to be correct.

This $T_{Arm}^{C_0}$ transform was used in all subsequent scene-object retrieval trials.

Others i.e. Jagerand et al [JFN97] use visual servoing rather than joint control to successfully manipulate objects. This method requires the establishment of a visual-motor Jacobian, obtained by visually observing changes during object movements and relating these to particular controller commands.

In this work the object grasping transform $T_{Object}^{Arm}$ is obtained directly once the object's pose, relative to the main head-frame, is obtained.

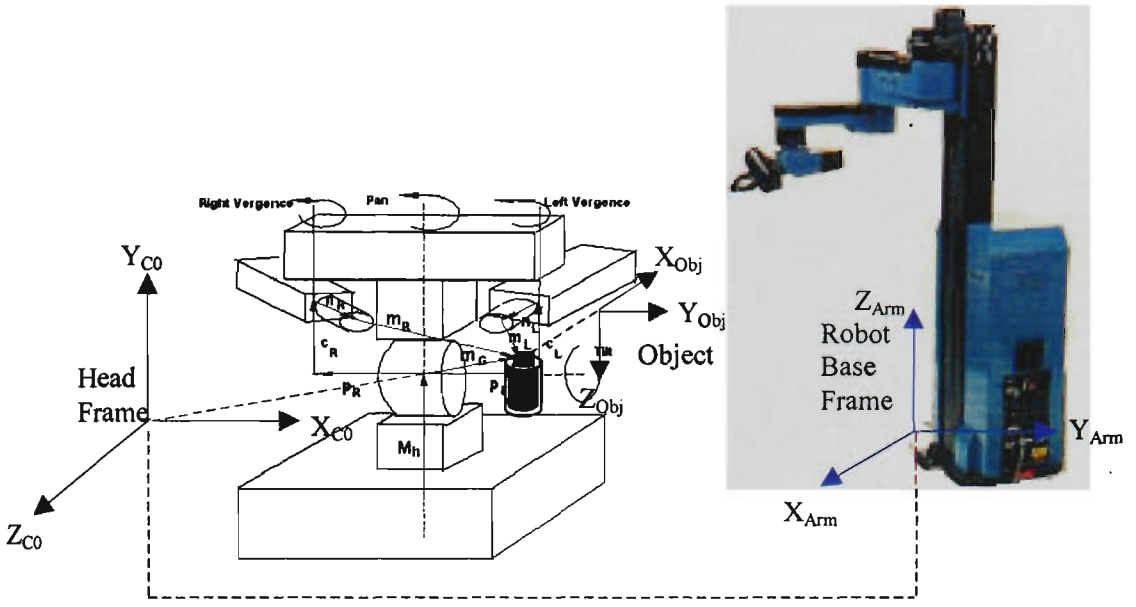



**Fig 6.7. 1** **Relative position and orientation of head and arm coordinate frames remain fixed**

The addition of wireless communication and auto docking capability, a future task

**Fig 6.7. 2 Partially completed transport mobile base**

## 6.8 MROLR System Simulator

The development of a "MROLR System Simulator" was undertaken and is at an early stage of completion. The Simulator will be provided with models representing the mobile base, landmark features, a target-object and a docking robot arm. In its present state, it consists of two non-integrated modules. Each module is based on, and builds upon the work of existing simulators. The first module utilises and builds upon the code associated with a simulator that is part of A. Davison's SCENE package [Dav01]. The second module is based on and builds upon Gurvinder Pal Singh's "Virtual Robotics Lab" [Sin99]. As Gurvinder's software did not include a robot compatible to the UMI RTX, this robot was coded and added. In addition the software extended to provide the necessary additional degrees of freedom (an increase from 5 to 6) required by the RTX. Inverse kinematic solutions were also not included with the simulator; these important algorithms were coded and added to facilitate direct movement to specified "x,y,z" positions with "yaw, pitch, roll" gripper orientations. Details of these additions are given in appendix B. Extra push buttons were also added to the "Control Pad" to facilitate these new movements (see Fig 6.8.4).

Module 1 graphically models the moving vehicle navigating to specified waypoints with the aid of displayed (selectable) landmark features. It utilizes the same EKF and mapping algorithms used by MROLR (with full covariance estimations of vehicle location and landmark features). A random generator is incorporated to simulate wheel slippage, and noise. Other features include the display of true positions in yellow and estimated positions in green. Landmark features can become selected or deselect by mouse clicks. Selected features change to red. Various control options and data outputs are provided with the associated GUI 's that are part of the simulator (see Fig 6.8.2). It is envisaged that this simulator will provide a useful facility for evaluating alternative feature selection and mapping and navigational strategies.

Module 2 is intended (ultimately) for evaluating proposed grasping points of selected 3D target object-models to ensure they can be retrieved. Even with precise modeling of the robot "end-effector" this is quite a difficult study, and one which the author believes has had insufficient attention. For instance, how does one design a generic set of rules for successfully picking up the vast number of assorted shapes and sizes of domestic objects with handles? Standard kinematic modeling incorporating consideration for size, volume, distribution of weight, balance of forces, and surface normals is possibly a start. Such analysis could be incorporated into this simulation unit. An initial attempt at accessing the suitability of a "grasping site" on an object, based on gripper opening size and the surface normals was added to the module. The requirement for a suitable "grasp" is that the object's dimensions be appropriate, and that the surface normals at the points of contact, and the gripper's surface normals, are in close (opposing) alignment (Fig 6.8.1). For situations where the grasping site was deemed unsuitable the gripper "slid" over the surface in an attempt to locate a suitable area. See Fig 6.8.6 for an example of the gripper attempting to pick up a teapot at an unsuitable location, and the gripping slipping "laterally" towards the teapot handle. This approach was later considered unrealistic as it took no account of the surface roughness, texture, or frictional forces involved, and was abandoned. Object physical dimensions only are currently considered in assessing grip ability.

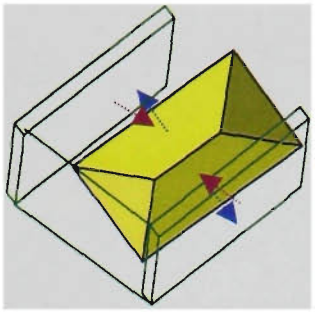
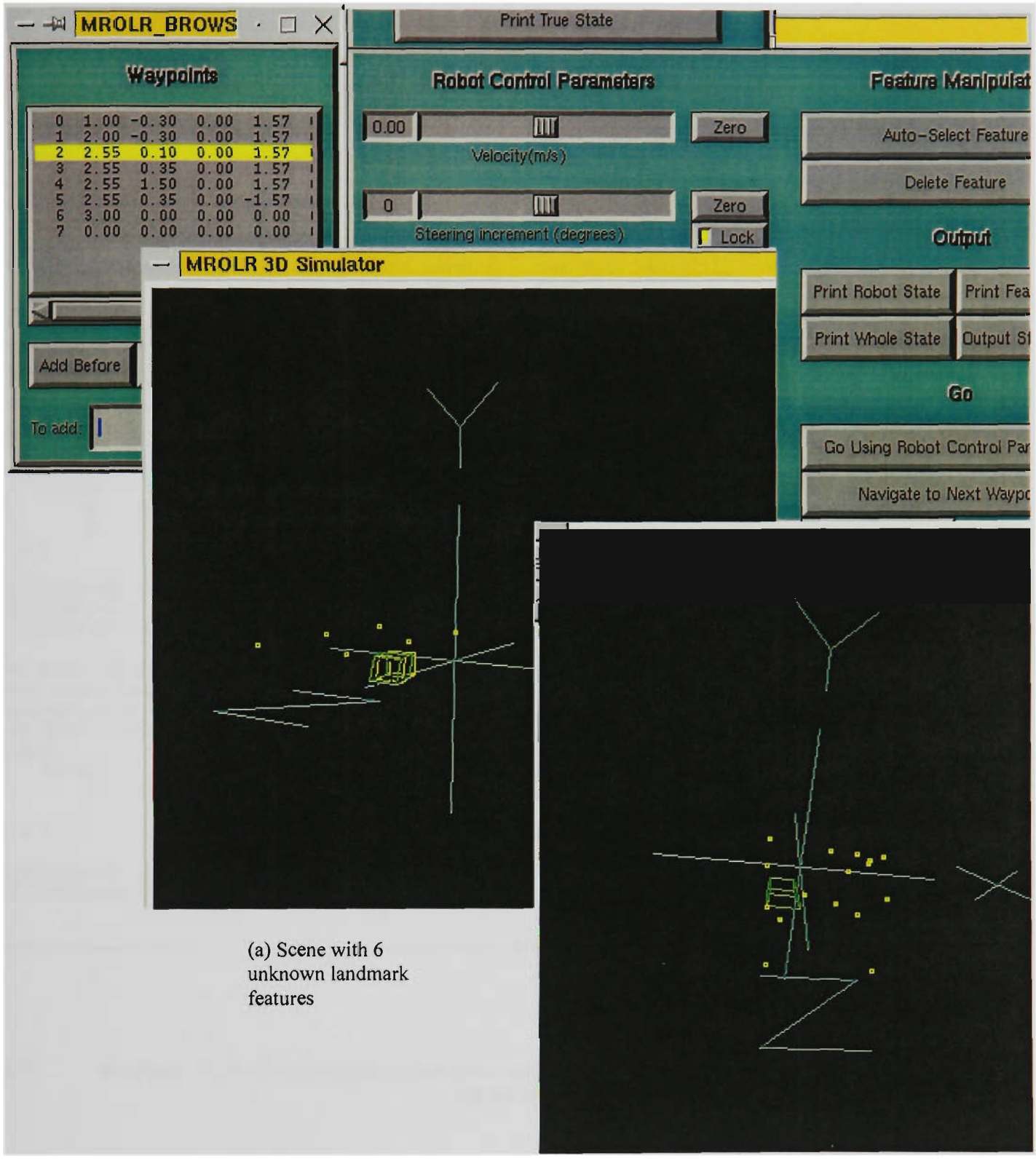


**Fig 6.8. 1 Gripper normals (brown) non aligned with object surface normals (blue).**

Both modules, in addition to uses outlined above will make useful tools for "MROLR System" teaching and demonstration. Where used solely for simulating MROLR, the modules have had their caption title modified to display "MROLR".

Module 1 is shown in Fig 6.8.2. (a) shows the mobile robot navigating to a specified way point with six landmark features. (b) depicts an example with ten known landmark features added. Examples of "Auto Feature Selection" and Covariance Outputs from the Simulator are given in Fig 6.8.3.

Module 2 is displayed in Fig 6.8.4 through Fig 6.8.9, illustrating the "controller" and "arm" retrieving a difficult object to grasp (the small teapot), and moving it to several locations. Fig 6.8.10 displays the arm after having transported an object (the red sphere) some distance from the table. (*Both the teapot and sphere model objects were borrowed from the MS DirectDraw SDK).*




(a) Scene with 6 unknown landmark features

(b) Scene with 16 features (10 known known features )

**Fig 6.8. 2 Several simulated views of the mobile base navigating to waypoints**

```
Added known feature with known feature label 1
Added known feature with known feature label 2
Added known feature with known feature label 3
Added known feature with known feature label 4
Added known feature with known feature label 5
Added known feature with known feature label 6
Added known feature with known feature label 7
Added known feature with known feature label 8
Added known feature with known feature label 9
Added known feature with known feature label 10
Read 8 waypoints.
*** SLOW PREDICTION ***
Best score found: 5.16285e-07.
Auto-selected feature with label 2.
*** SLOW UPDATE ***
R_tot:
    4e-06         0         0
        0    0.0001         0
        0         0     4e-06
S = [

3.79788e-05 -3.25407e-24 4.17903e-23
-3.25407e-24    0.0001 -4.00216e-42
4.17903e-23 -4.00216e-42     4e-06
]
*** SLOW PREDICTION ***
Best score found: 6.23208e-07.
Auto-selected feature with label 9.
*** SLOW UPDATE ***
R_tot:
    4e-06         0         0
        0    0.0001         0
        0         0     4e-06
S = [

5.53167e-05 -2.37655e-06 4.57195e-07
-2.37655e-06 0.000100123 -2.36034e-08
S = [

6.44373e-06 -2.47343e-07 3.11819e-08
-2.47343e-07 0.00010016 -2.0121e-08
3.11819e-08 -2.0121e-08 4.00254e-06
]
*** SLOW PREDICTION ***
Best score found: 2.13885e-07.
Auto-selected feature with label 3.
*** SLOW UPDATE ***
R_tot:
    4e-06         0         0
        0    0.0001         0
        0         0     4e-06
S = [

6.50338e-06 -2.81064e-07 3.54218e-08
-2.81064e-07 0.000100175 -2.20106e-08
3.54218e-08 -2.20106e-08 4.00277e-06
]
*** SLOW PREDICTION ***
Best score found: 2.64448e-06.
Auto-selected feature with label 5.
*** SLOW UPDATE ***
R_tot:
    4e-06         0         0
        0    0.0001         0
        0         0     4e-06
S = [

0.000993588 -8.00296e-07 7.71573e-08
-8.00296e-07 0.000100232 -2.23564e-08
7.71573e-08 -2.23564e-08 4.00216e-06
]
```

**Fig 6.8. 3 Examples of auto-feature selection and covariance outputs from simulator corresponding to Fig 6.8.1(b).**

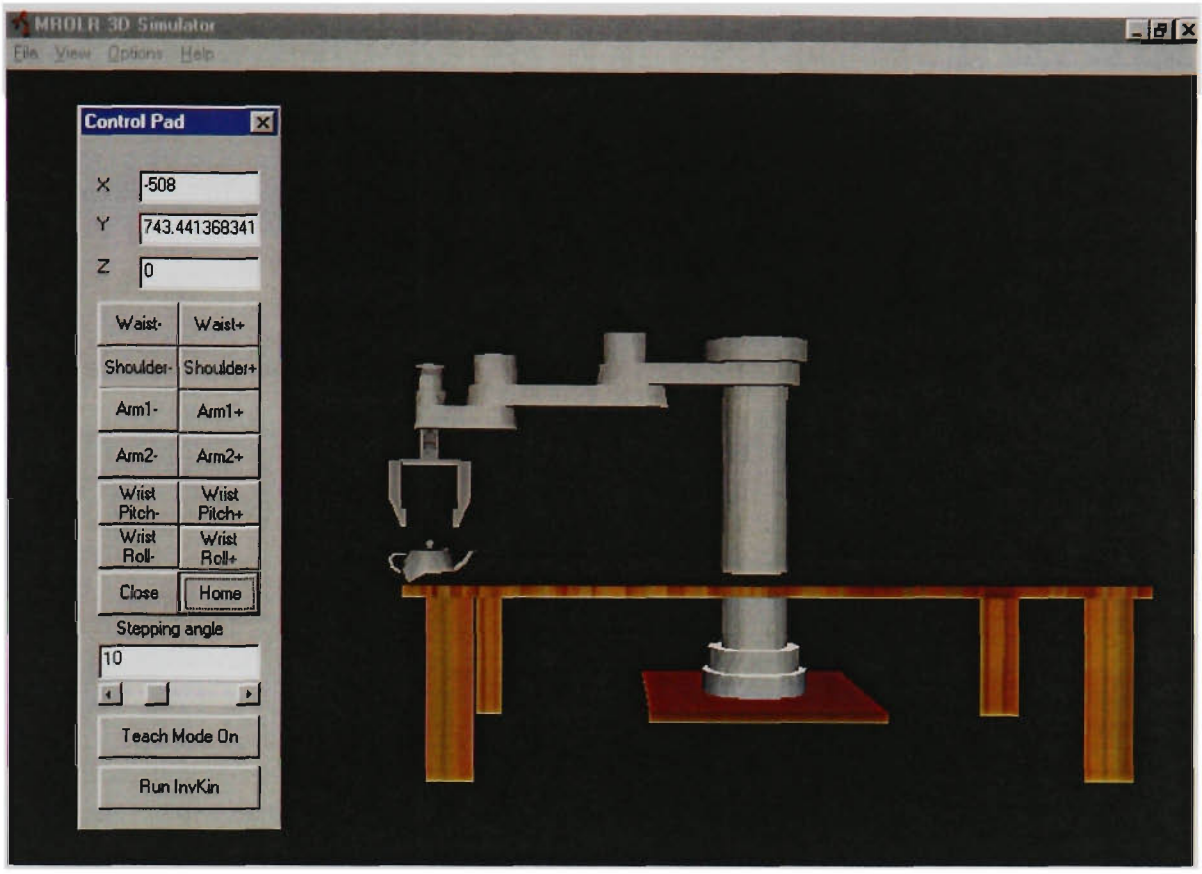



**Fig 6.8. 4 Robot arm at table with target object (small teapot)**

Notice the X, Y, Z positions of the gripper are displayed on the Controller.

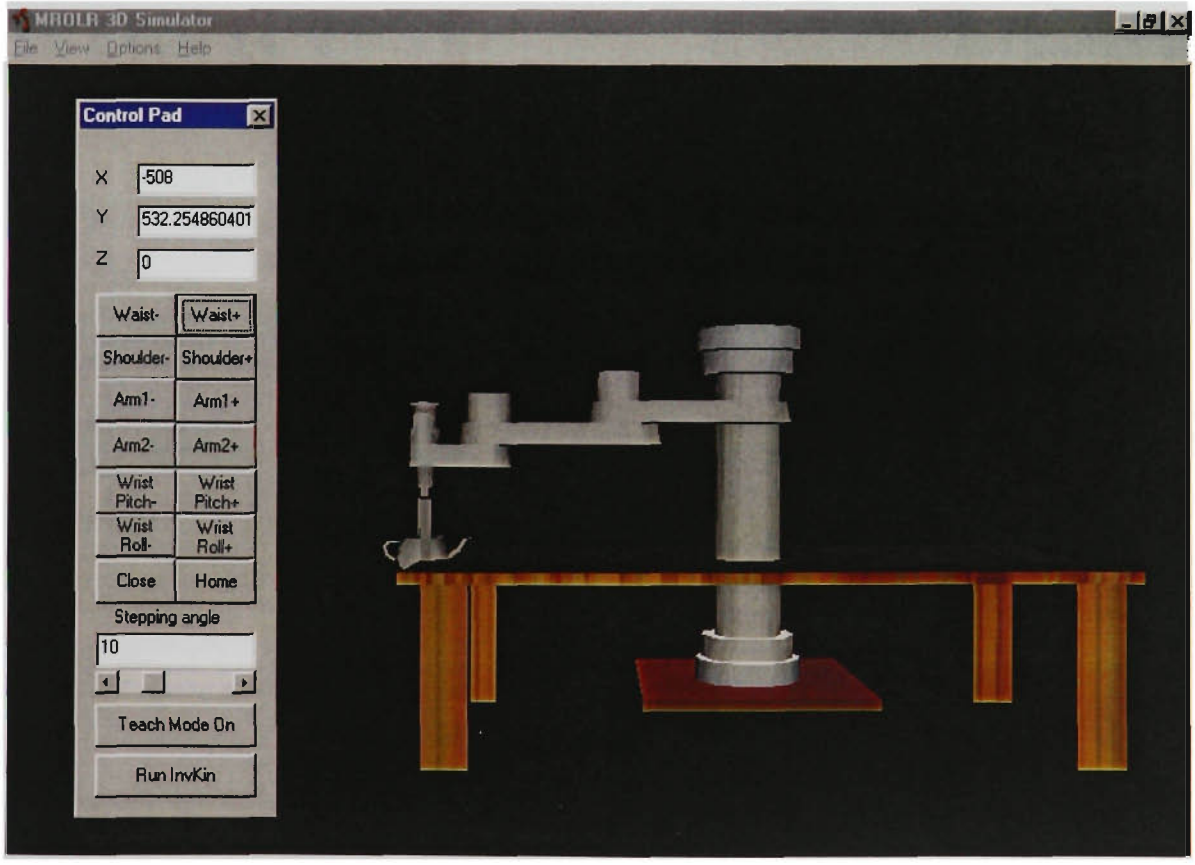



**Fig 6.8. 5 Arm with gripper over target object.**

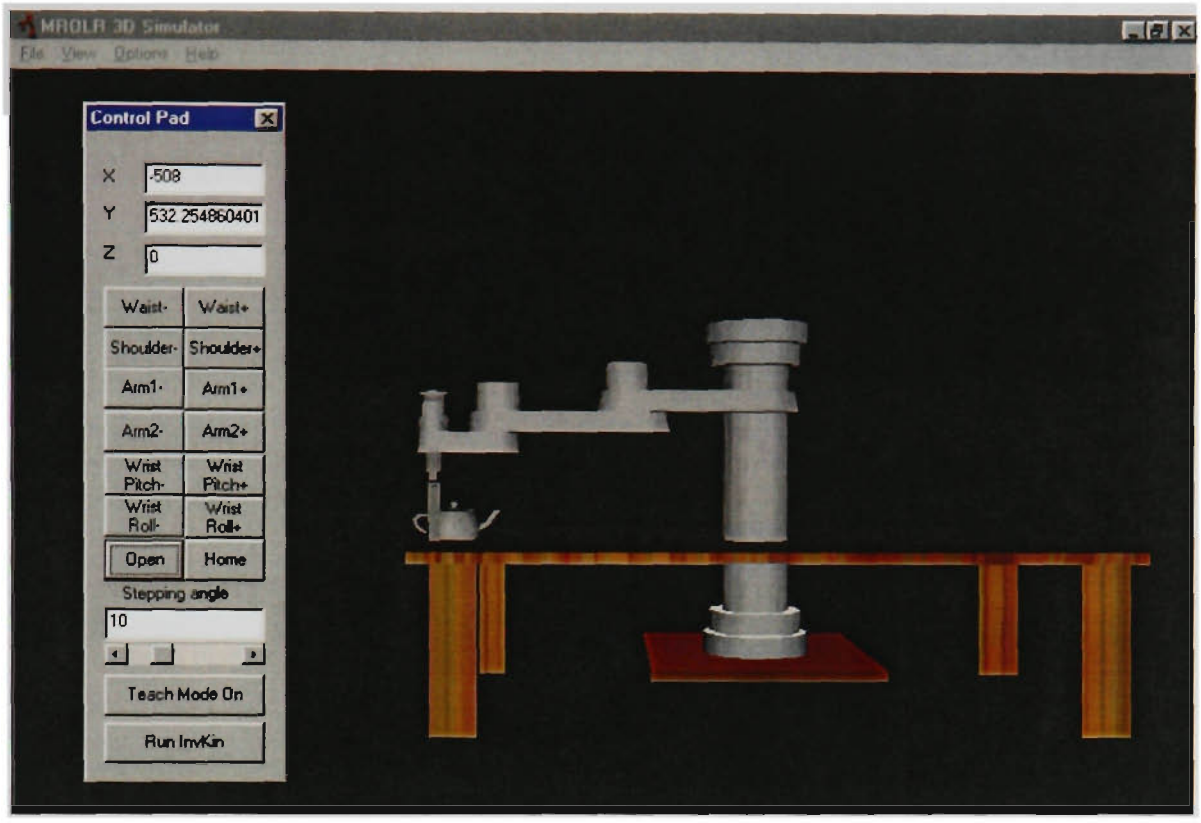



**Fig 6.8. 6** **Gripper closes and slips over surface of object towards handle in search of more appropriate grasping location.**

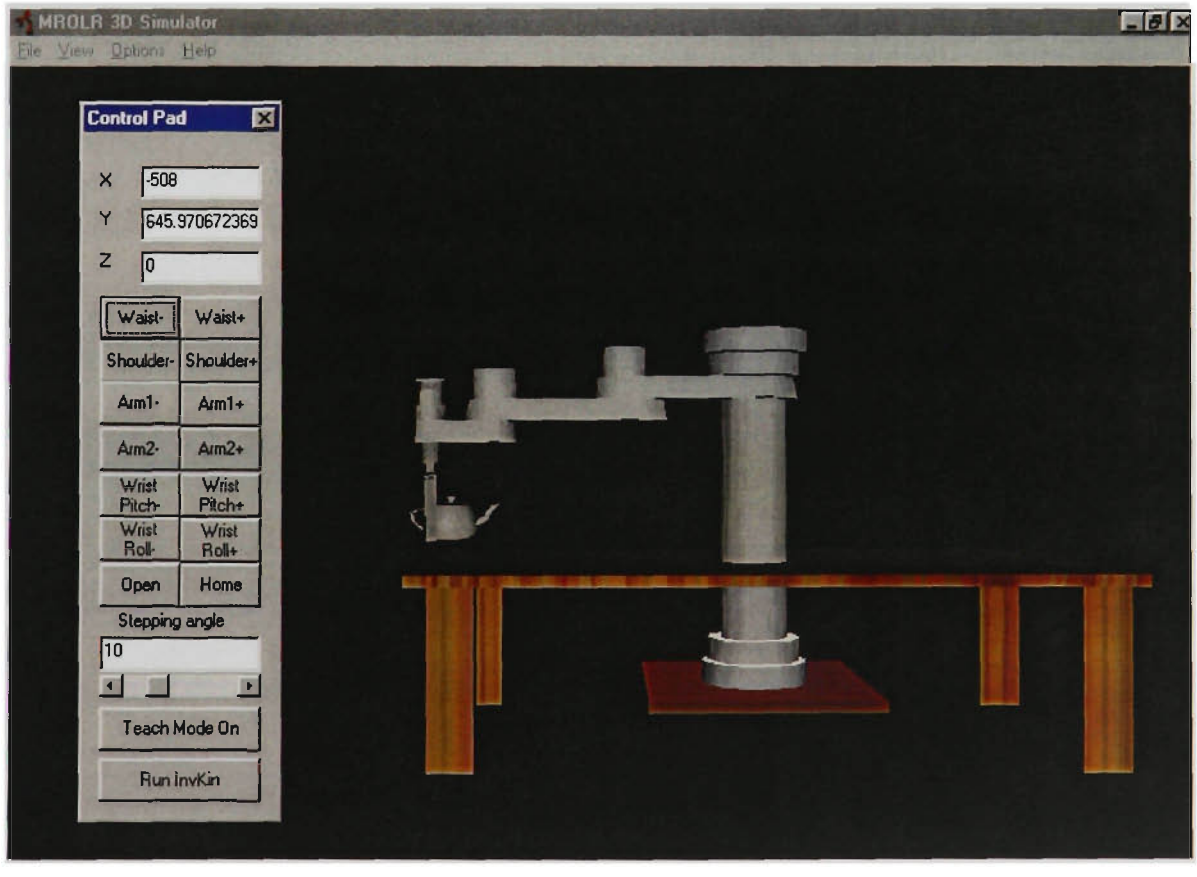



**Fig 6.8. 7** **Target being raised**

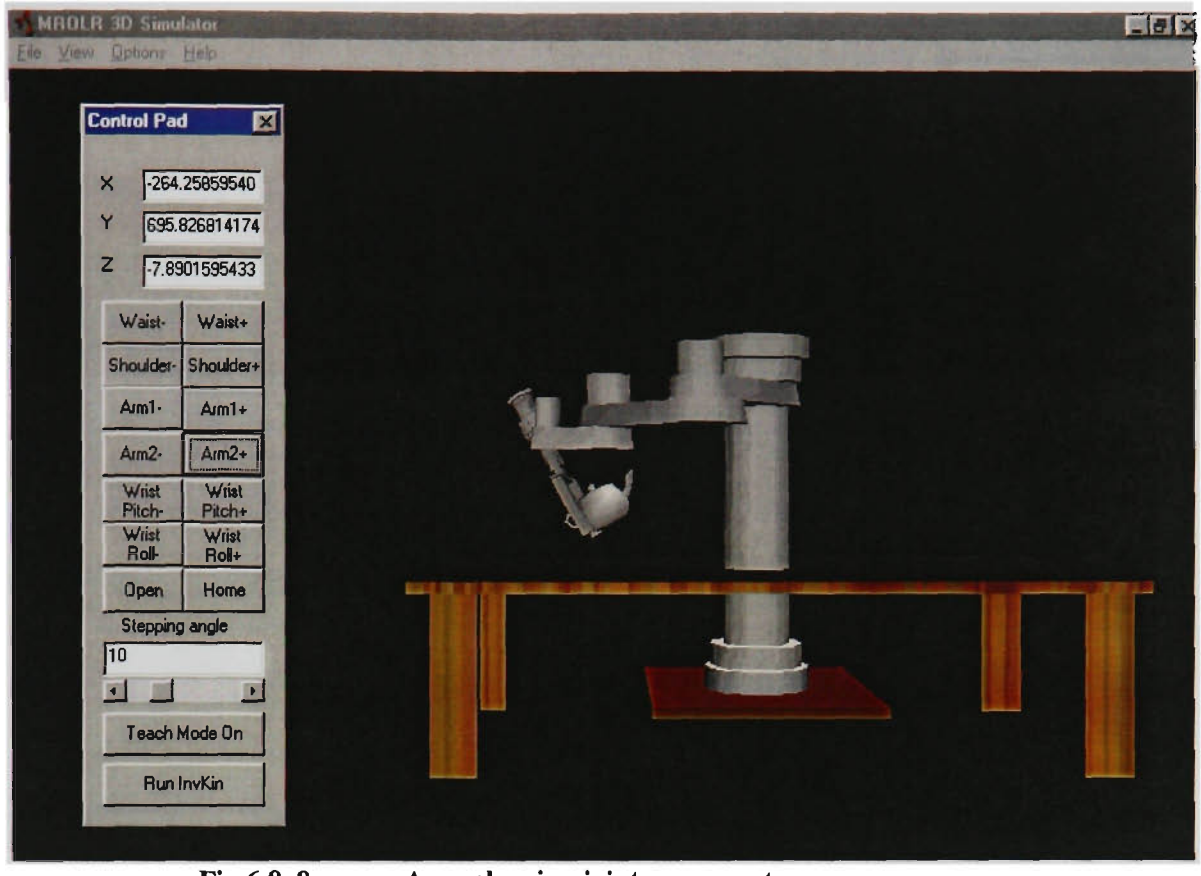



**Fig 6.8. 8 Arm showing joint movements**

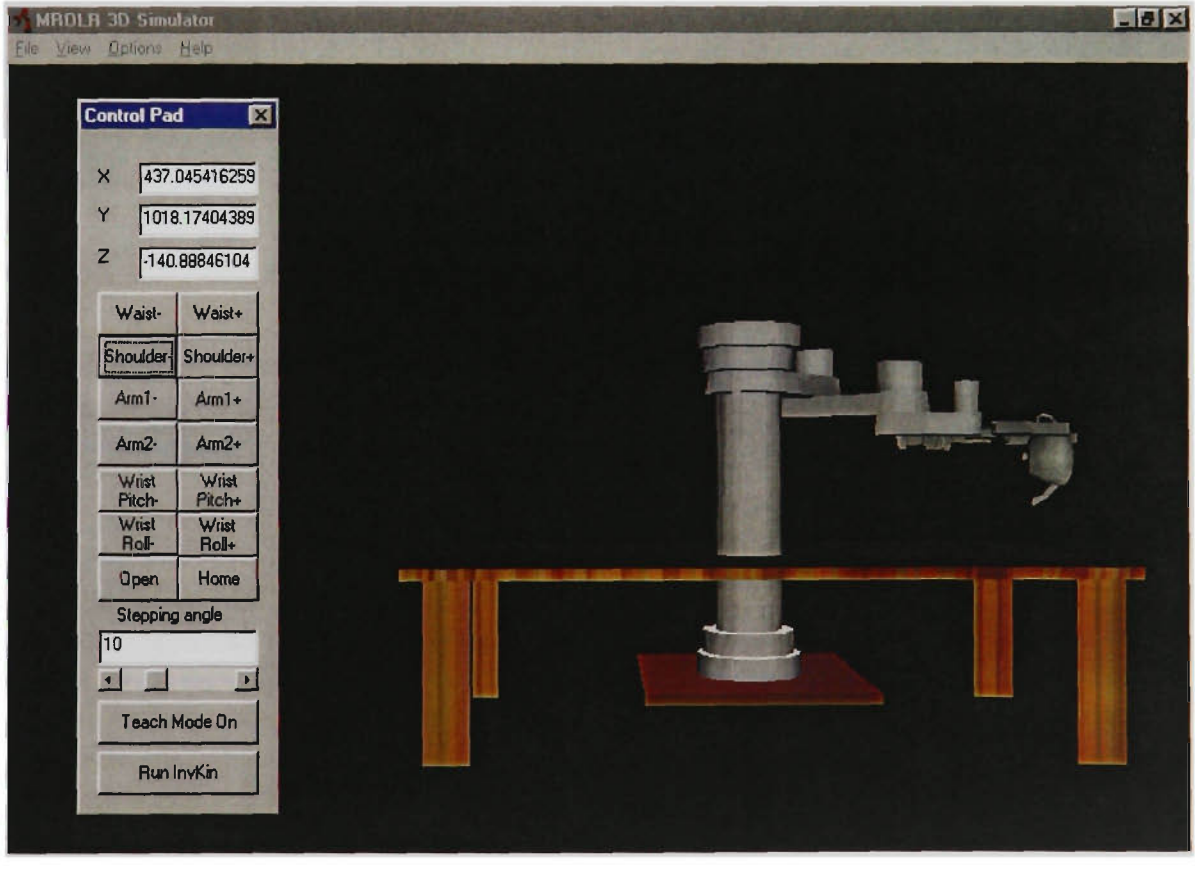



**Fig 6.8. 9 Arm showing joint movements**

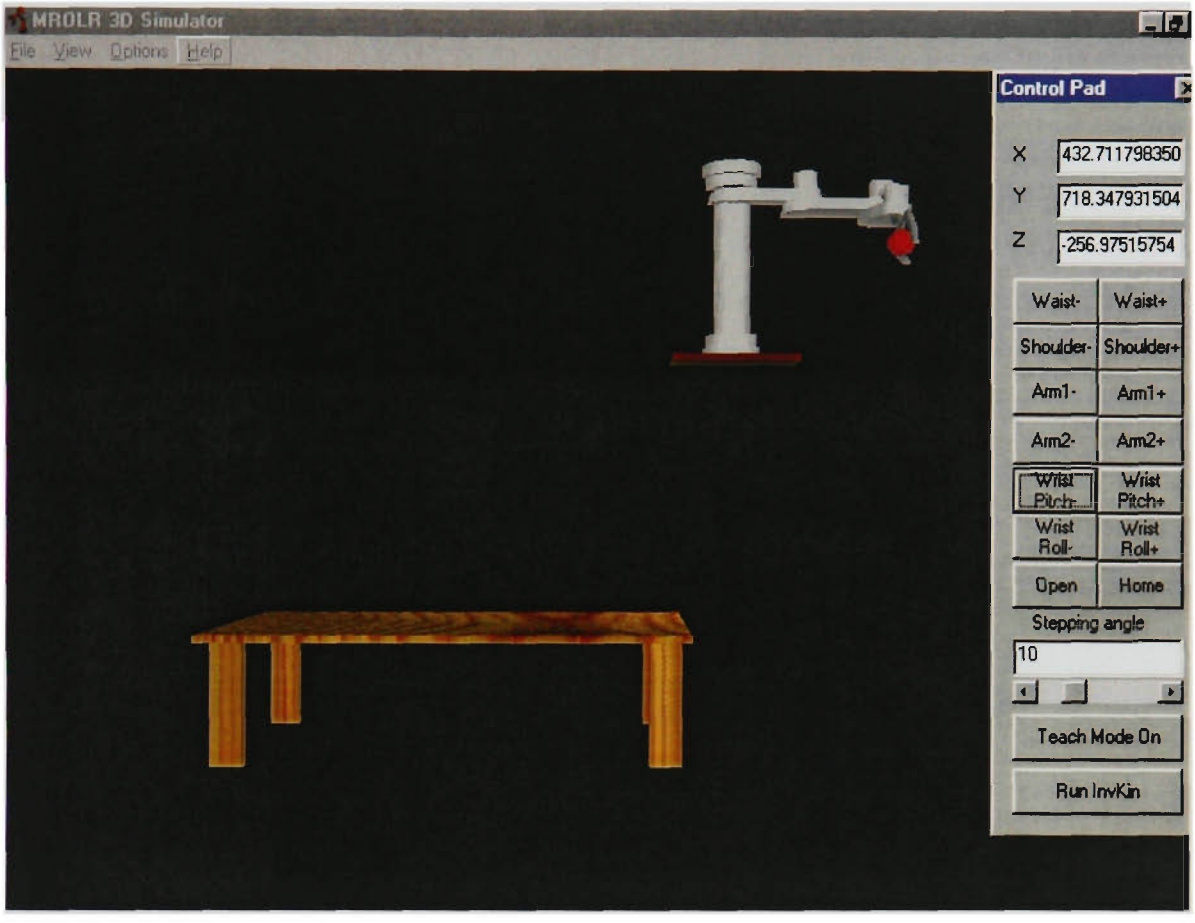



**Fig 6.8. 10 Arm moving away from table with retrieved red sphere**

A useful possible extension to the simulation modules (discussed further in Chapter 8 ) would be for all created objects to be positioned in the simulated scene, mouse selection of the target object (or verbal selection, if voice recognition was added) could then initiate a simulation run.

## *6.9 Conclusion*

This chapter commenced by citing the advantages of using wire frame outline models created from multi-views, and describes a simple and effective method for their production. For poorly formed models an editor is provided that enables both the addition and removal of 3D segments. Paramount to this work is the ability to reliably recognise objects in a scene, whose models have been created and maintained in a database of objects. A recognition algorithm based on the work of Pollard et al was implemented, and extended to automate the recognition process. The recognition module also provides the essential 3D pose parameters, relative to the stereo head, to guide the docked robot arm towards the sought object's grasping point.

With the conclusion of this and previous chapters the framework to perform tests to ensure a fully functioning MROLR have been established. Testing and verification is carried out in Chapter 7.

# Chapter 7 Testing and Verification

## 7.0 Introduction

The algorithmic development and modeling of the previous chapters is now complete and the necessary tools to create and edit 3D object-models that may be requested for future retrieval have been developed. The means for navigating the environment exist. Searching for and mapping landmark features with the aid of active vision at each step, and using the resulting data for self-localisation purposes reduces the likelihood of "loss of bearings" due to wheel slippage or collision. Once in close proximity to a scene containing any of the objects for which a created model exists, a search by panning the stereo head can be performed. If the object is successfully recognised, its 3D position in the scene with respect to a docked robot arm can be ascertained. The resulting homogeneous transform formulated, together with the a priori specified handle location or grasping point, should guide the arm to pick up the object and place it on board MROLR. Limited simulations of MROLR can also be performed using the "virtual environments" of module 1 and 2 described in section 6.8.

The vital task of testing MROLR to vindicate the methodology of this research remains. Tests need to include the creation of a significant number of typical domestic or industrial object models as well as navigational, recognition and retrieving trials. These are described in this chapter.

The results of the creation of ten representative domestic object-models are given in section 7.1. The results of object location and retrieval trials are given in section 7.2, the scope and limitations of tests, their evaluation, and methods of overcoming some of those limitations are discussed in the subsections.

## 7.1 Creation of Database Models

Ten representative domestic object-models were created using the portable (level adjustable), model creating table described in section 6.1. These were:

- large cup,
- vice,
- shaver
- bottle
- Solo can
- cube
- hole punch
- stapler
- drink package
- cup

Their creation is shown in Figs 7.1.1 to 7.1.11. Note that for conservation of space, only one stereo view of each object is shown from Fig 7.1.3 onwards.

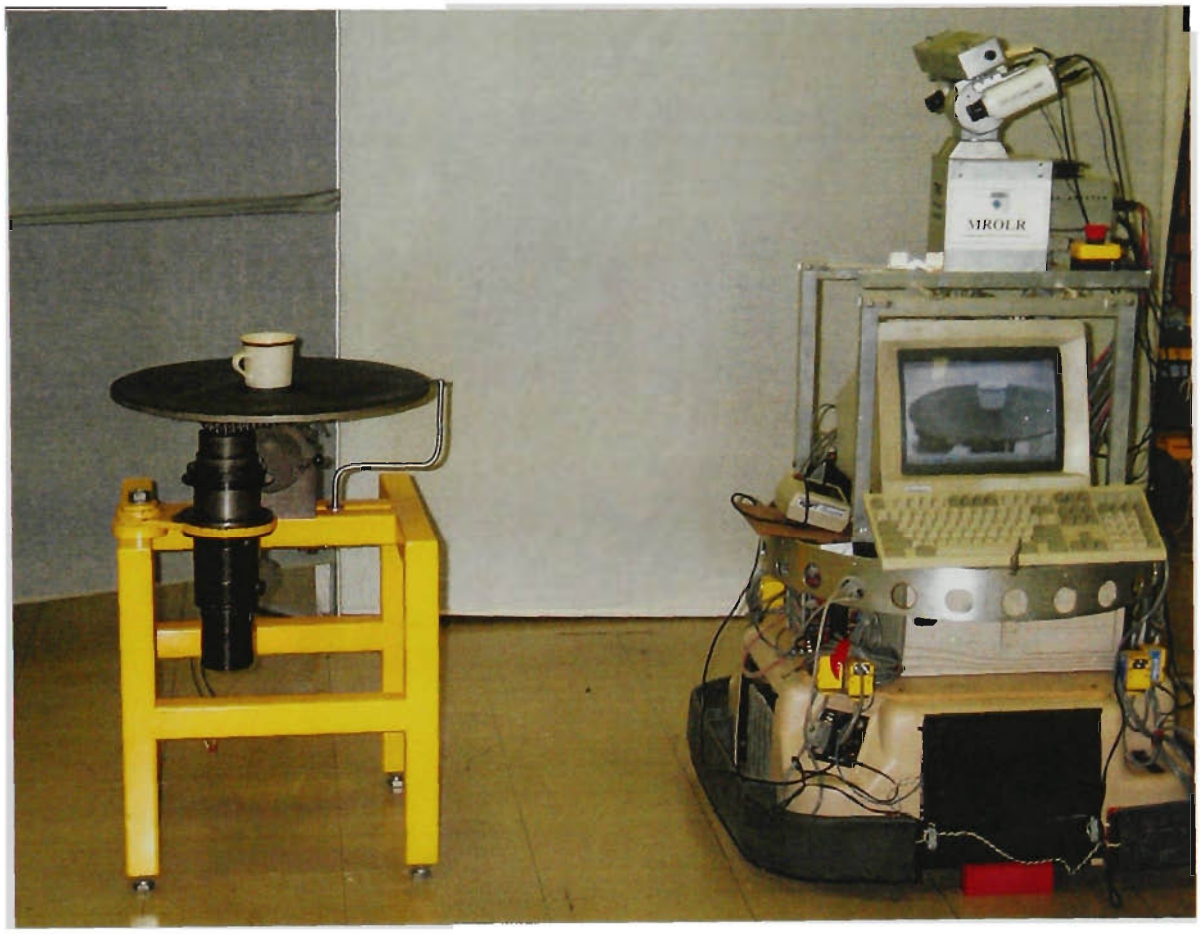
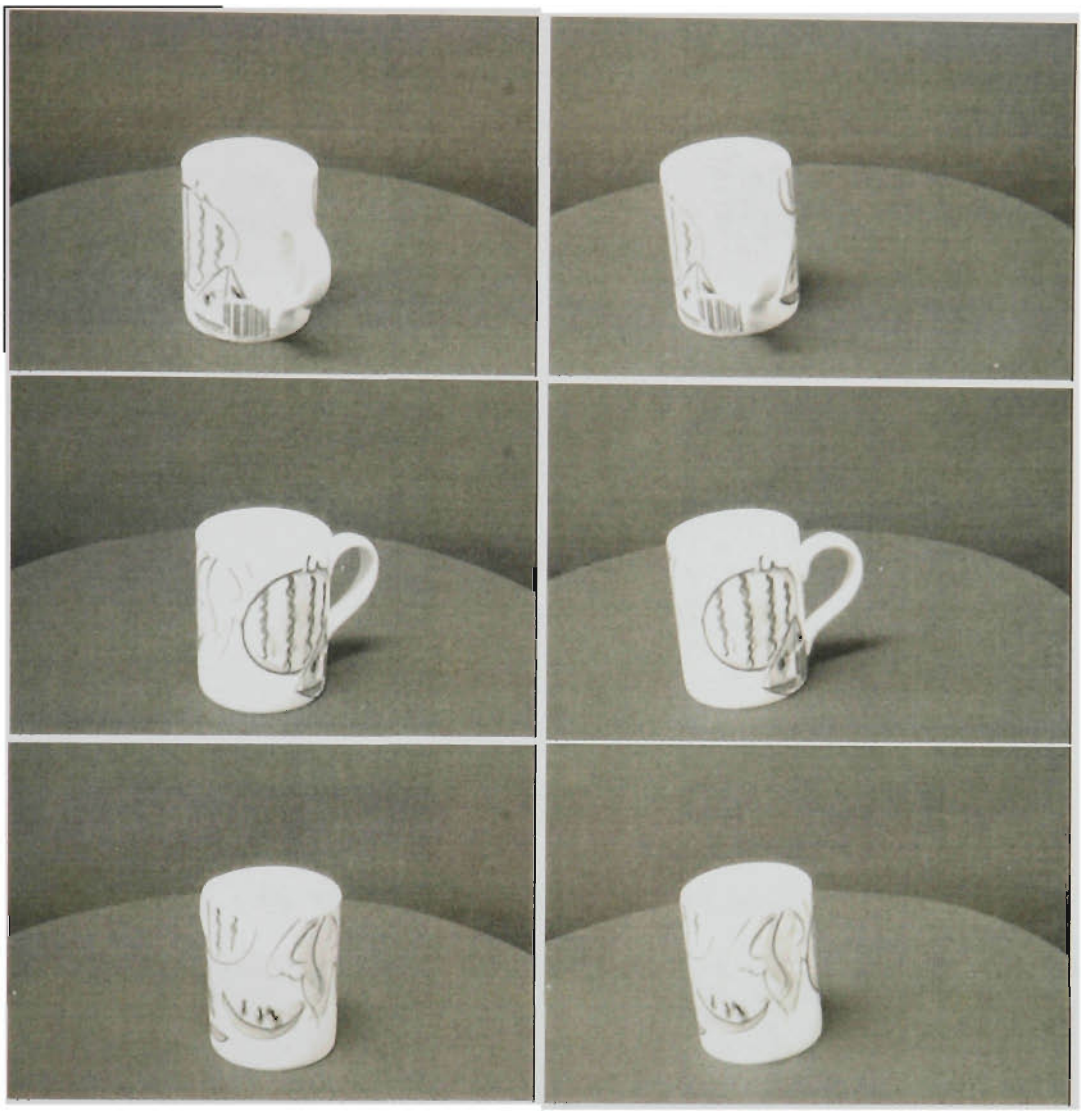
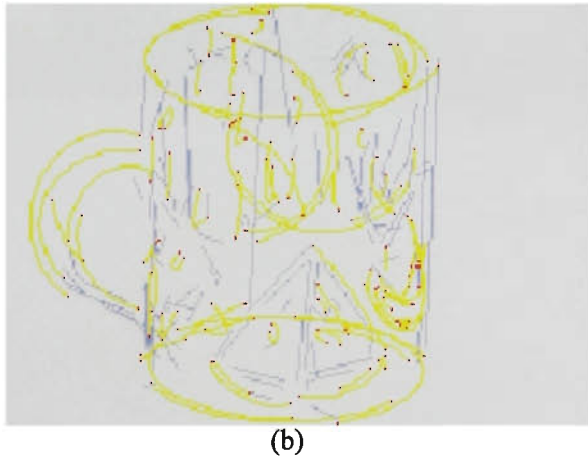



**Fig 7.1. 1 Object model creation**

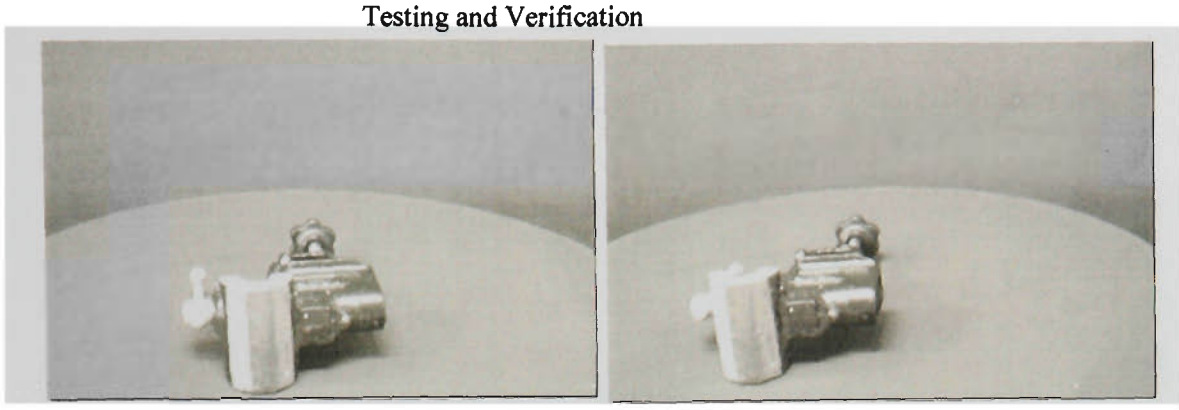


(a)



(b)

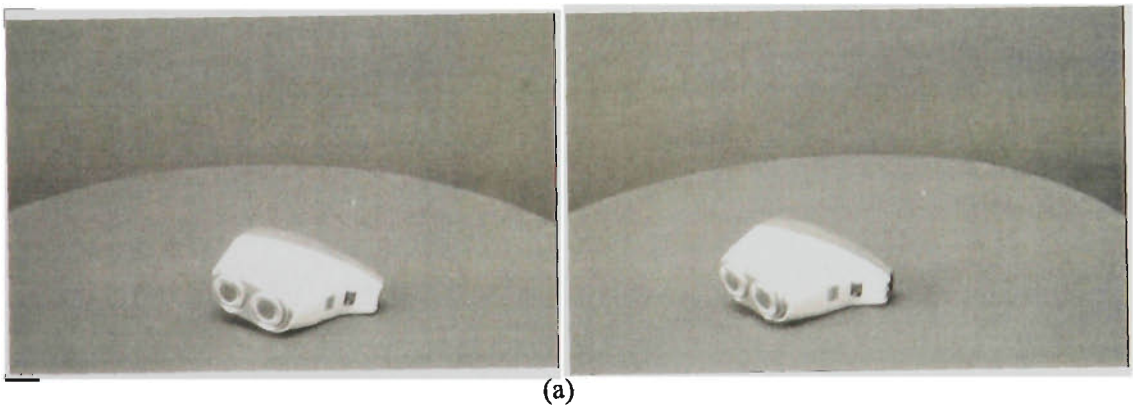
**Fig 7.1. 2 (a) 3 Stereo views of "Large Cup", (b) Model of "Large Cup"**



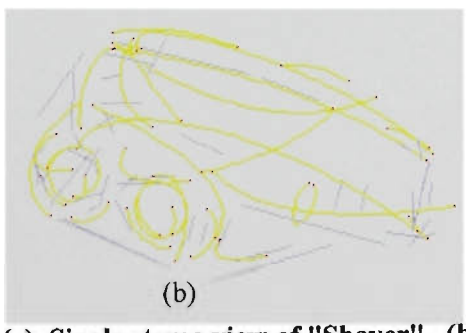
(a)

(b)

**Fig 7.1. 3 (a) Single stereo view of "Vice", (b) Model of "Vice"**

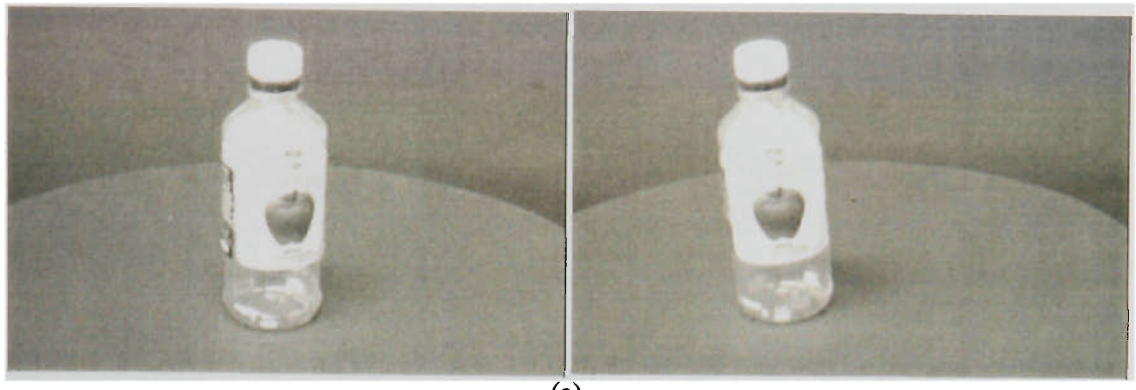


(a)



(b)

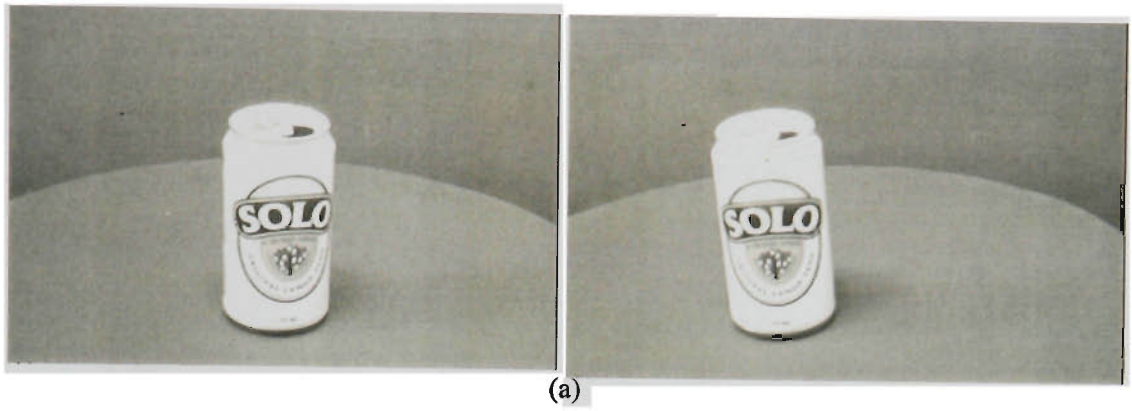
**Fig 7.1. 4 (a) Single stereo view of "Shaver", (b) Model of "Shaver"**



(a)

(b)

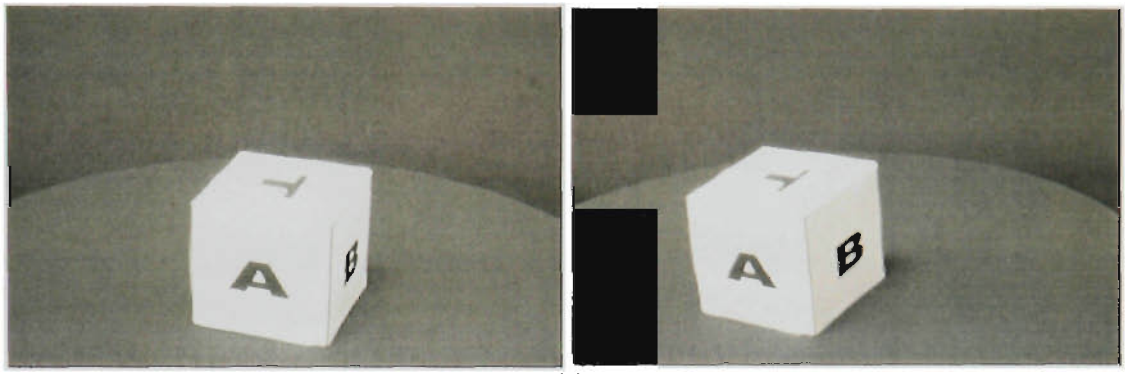
**Fig 7.1. 5 (a) Single stereo view of "Bottle", (b) Model of "Bottle"**



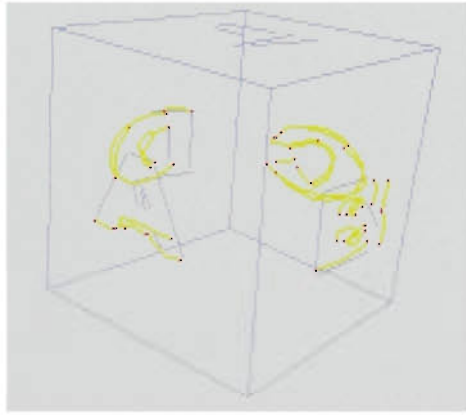

(a)

(b)

**Fig 7.1. 6** **(a) Single stereo view of "Can", (b) Model of "Can"**

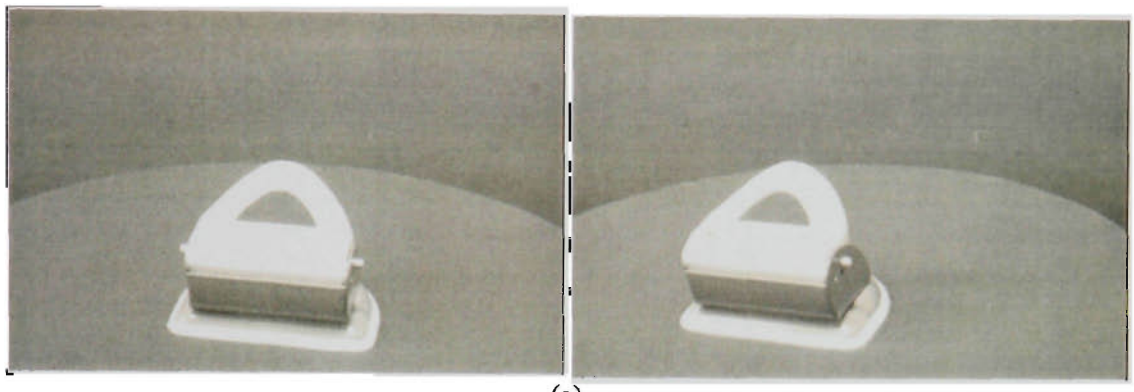



(a)

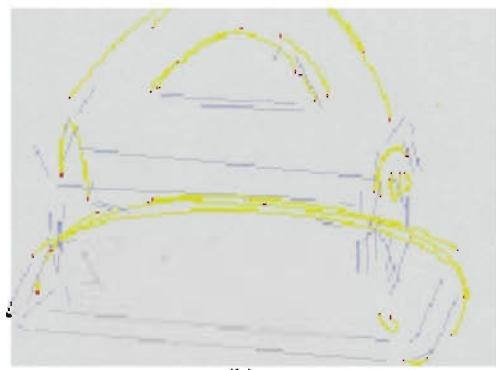


(b)

**Fig 7.1. 7 (a) Single stereo view of "Cube", (b) Model of "Cube"**

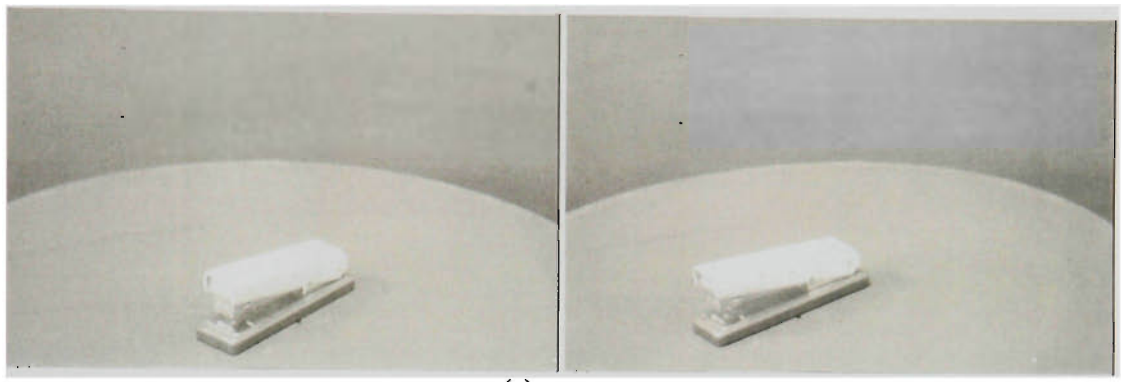


(a)

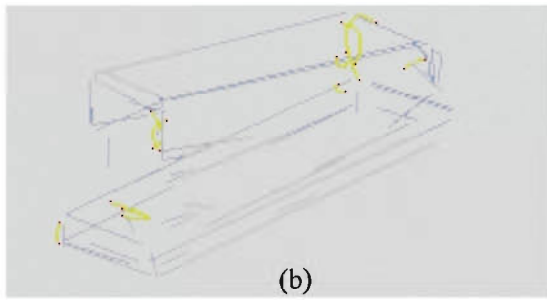


(b)

**Fig 7.1. 8 (a) Single stereo view of "Hole Punch", (b) Model of "Hole Punch"**



(a)



(b)

**Fig 7.1. 9 (a) Single stereo view of "Stapler", (b) Model of "Stapler"**

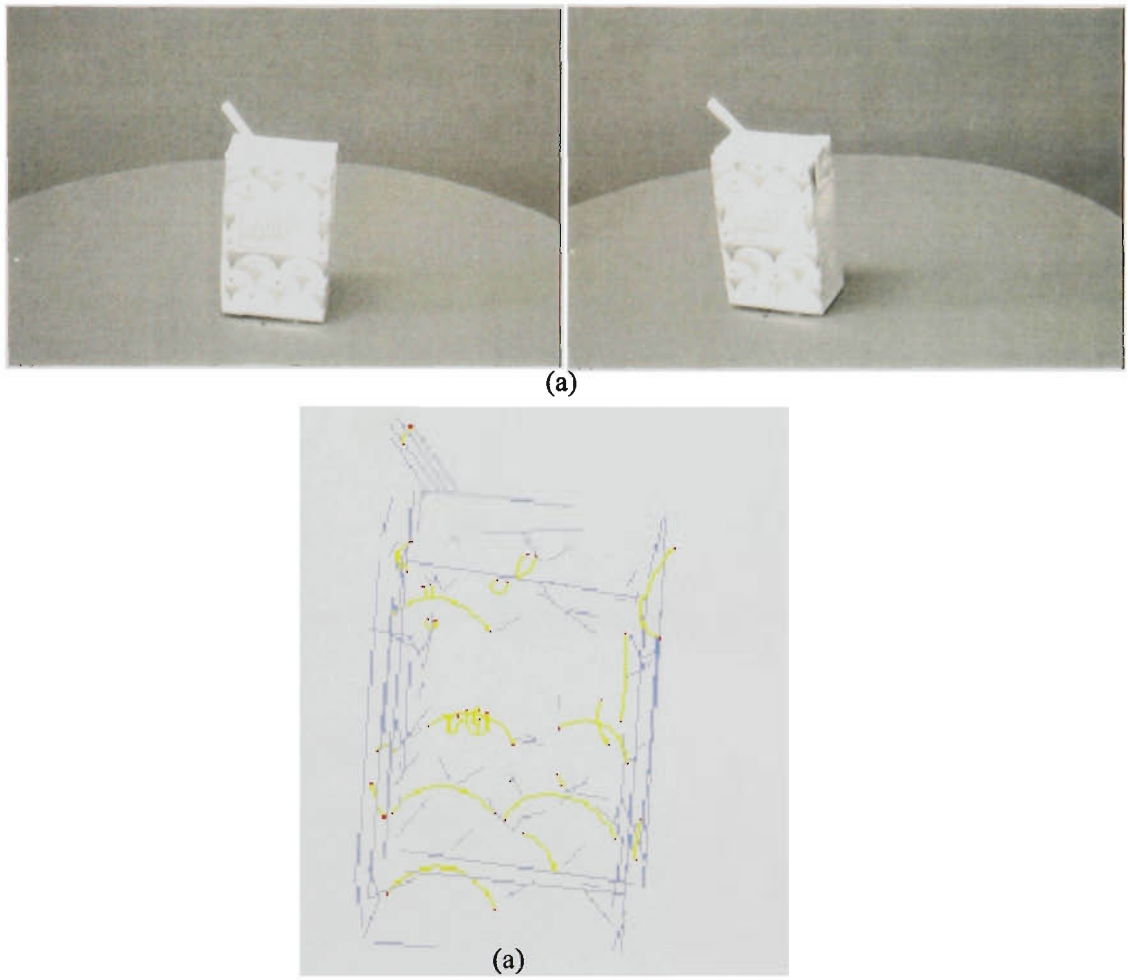
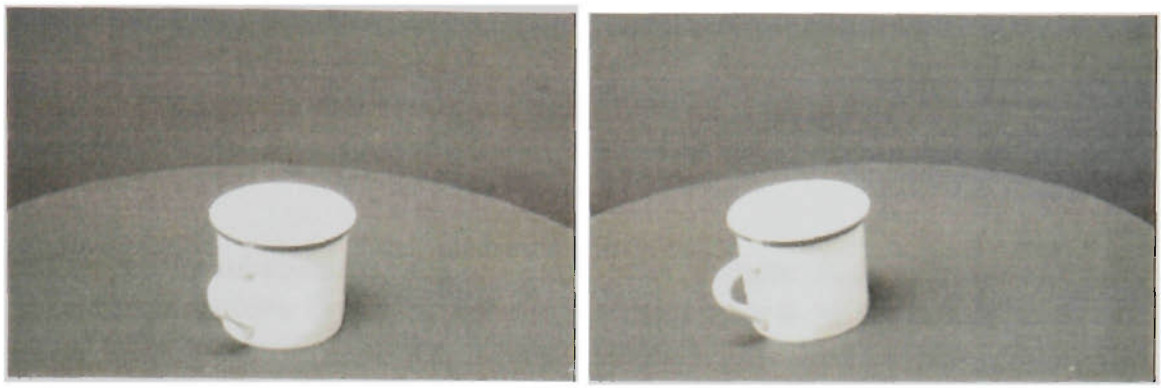
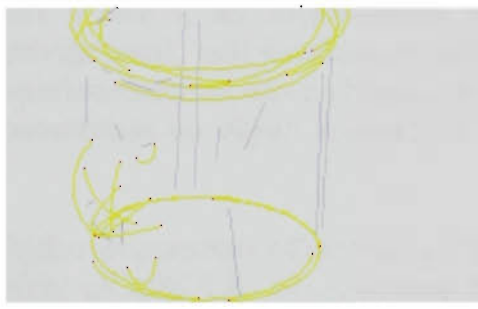



**Fig 7.1. 10 (a) Single stereo view of "Drink Package" , (b) Model of "Drink Package"**



(a)



(b)

**Fig 7.1. 11 (a) Single stereo view of "Cup", (b) Model of "Cup"**

### 7.1.1 Storage of Focus Features and Cliques

Fig 7.1.12 illustrates the stored format of focus features and cliques. The first number (10 in this example of the model cup) is the total number of cliques for the model. Following lines contain (in order) the focus feature identification number (i.e., 560) the number of features in the clique (i.e., 9) followed by the Id number of each feature. The focus feature is also normally listed as the end feature of a clique.

```
10
560 9 1490 556 568 3086 1122 554 550 564 560
564 8 564 550 1098 1122 1096 568 556 1490
554 12 570 568 3086 1112 1098 1096 556 564 1490 3080 1122 554
554 7 1104 564 1132 560 1488 3076 554
1482 7 1482 1132 556 554 1102 1488 546
564 7 1490 556 1102 3080 560 1482 564
1488 5 1488 1132 554 1122 1112
564 5 1490 1482 556 554 564
554 4 554 1488 564 1482
1482 5 1490 1482 556 554 564
```

**Fig 7.1. 12 Table of focus feature and cliques**

Several of these created models will be targeted in the tests that follow in Section 7.2

## 7.2 Tests

Trials were arranged to substantiate MROLR's ability to navigate to specified waypoints, and recognise, locate and retrieve (grasp) a variety of target objects in various orientations and locations on a table at each waypoint. Lengthy tests to demonstrate the validity of the navigational algorithms were not deemed necessary, as others (i.e., [Dav98], [Dav01]), have already done these using the Oxford GTI mobile platform. While our active stereo head, and 2 wheel drive mobile base are different to those of the Oxford project, resulting in the modeling and reformulation outlined in chapters 4 and 5, the methodology and principal algorithmic constructs used are identical. The navigational, self-localisation and mapping-building tests performed here, can be regarded as reinforcement of those obtained by A. Davison. The major testing carried out in this section concentrates on object recognition, pose determination and retrieval.

It is worth mentioning that locating objects in positions other than an "expected region", such as a waypoint, is beyond the present ability of MROLR. To overcome this limitation and enable objects randomly located in a room to be found would require vastly superior cameras and lenses, perhaps with auto focus, zoom and accurate focal point re-calibration. In addition, a path-planning module capable of mapping open spaces that could negotiated would need to be incorporated.

For each test devised, MROLR commenced its journey in the laboratory from a fixed world coordinate (waypoint 0). Following initialisation, the vehicle navigated autonomously using odometry, and visual navigational features, mapping the environment as it travelled. On reaching its destination the head commenced panning for the target object and if this was located, the robot arm was docked with the platform and the desired object grasped and retrieved. If the object could not be found (perhaps because the object was not on the table), MROLR returned empty handed to waypoint 0.

Initialisation data supplied to MROLR consisted of known-features, waypoints to navigate to, and the target object and its associated grasp point (if applicable). As MROLR autonomously navigated towards each waypoint the following behaviour was observed:

(a) head fixation angles for the nearest known-feature were calculated (from stored x, y, z positional information), and the head cameras driven to gaze in the direction of the feature. A search for a corresponding "patch match" (within a narrow spatial window whose size was determined by map uncertainty) ensued.

(b) if a fixation could not be attained, an alternative mapped or new feature was sought. The decision as to which feature to choose, was based on two criteria: expected visibility, and the value of the measurement. Once a measurable subset of features in the map was identified, the value of measuring each one is evaluated in terms of the uncertainty of its position relative to the

vehicle. The choice was then made on the basis of the highest innovation covariance (i.e., "the Vs rule" see section 5. 7).

(c) the map continued to be updated sequentially as the robot moved about in its environment, making measurements of features and updating the state vector and related covariances, according to the rules of the Extended Kalman Filter.

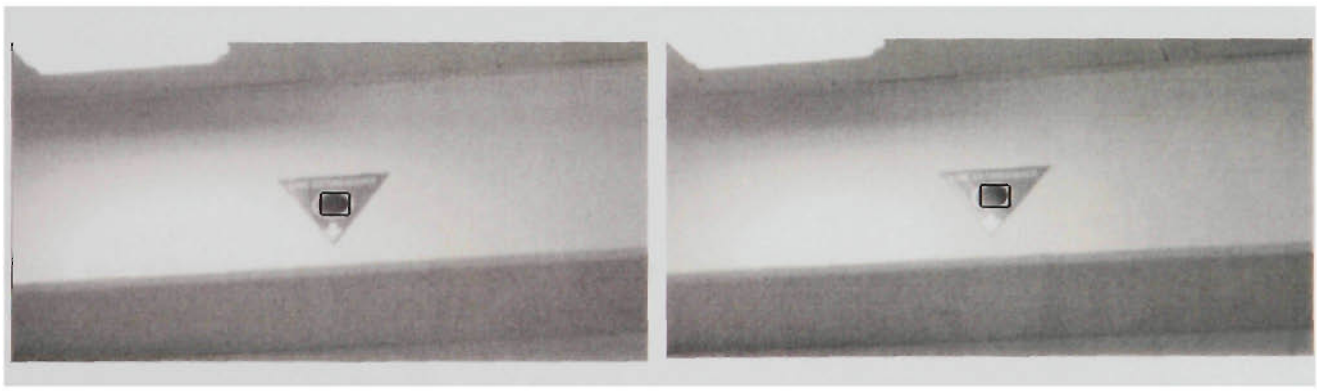
Several views of MROLR navigating to the "Table" waypoint are shown in Fig 7.2.1. The subsequent "lock on to" known-feature the " fire extinguisher sign" is shown in Fig 7.2.2. Additional features used for mapping the laboratory environment are also displayed in Fig 7.2.3.

(a) Robot navigating towards specified waypoint

(b) Robot closer to specified waypoint

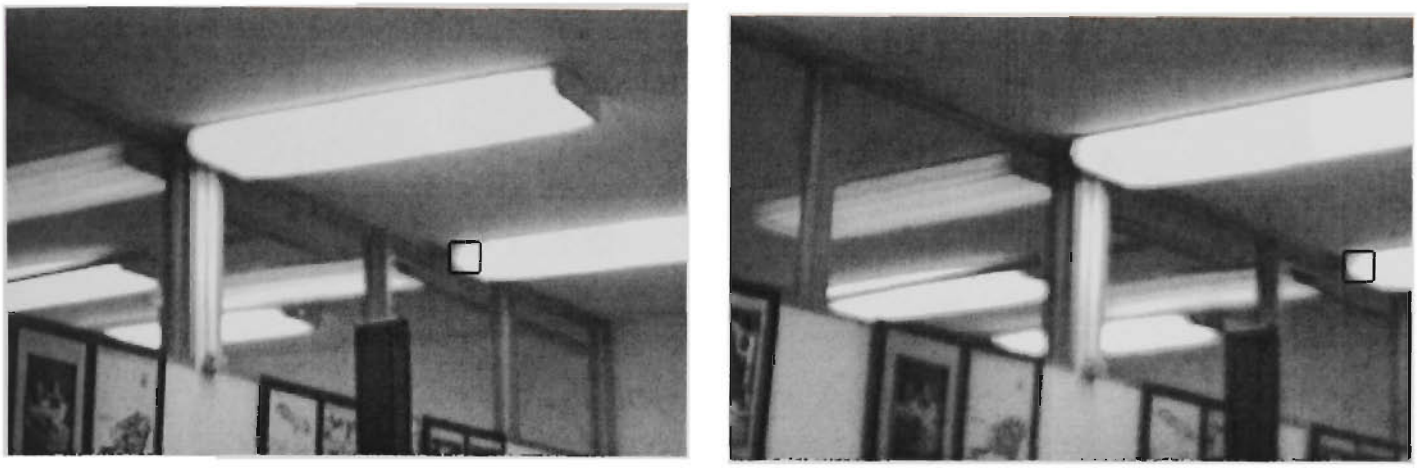
(c) Head pointing at known-feature "Fire Extinguisher"

**Fig 7.2. 1** **(a), (b) and (c) Several views of MROLR navigating towards specified waypoint, "the Table".**

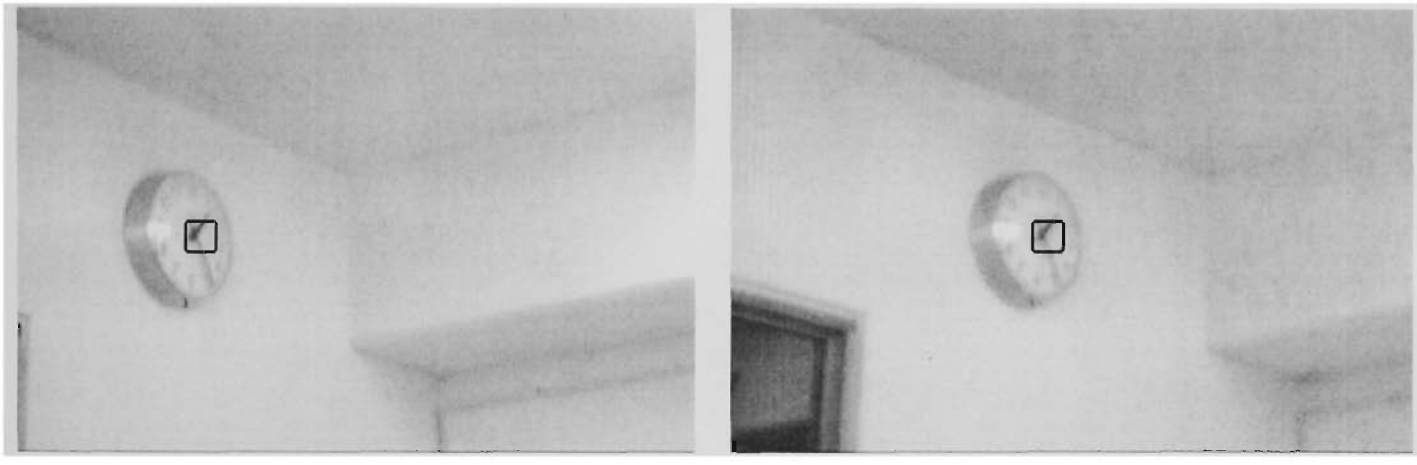


***Feature location( w.r.t Head) x=0.123 ,y=1.345, z=1.732 (metres)***

**Fig 7.2. 2** **Lock on achieved for "Fire Extinguisher " sign feature patch.**



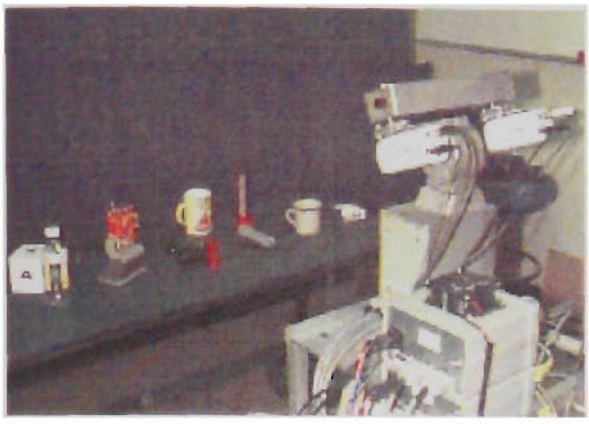
(a)



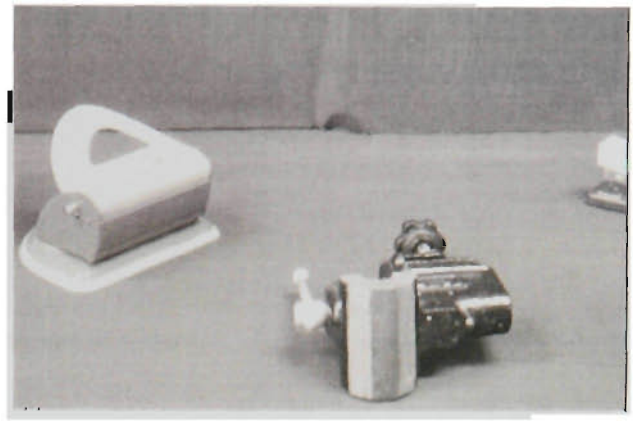
(b)

**Fig 7.2. 3 Two additional mapped features (a) light fitting edge (b) clock face , intersection of hands**

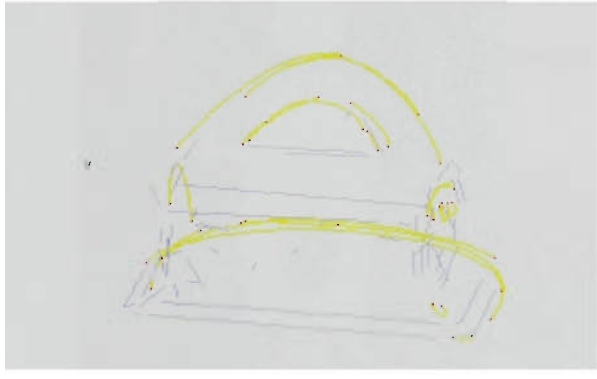
In Fig 7.2. 4 (a), the robot is at a waypoint panning for the hole-punch. Fig (b) shows the left image of a scan sequence search for the hole-punch in the scene, while Fig (c) shows the database model of the hole-punch. Fig (d) illustrates the resulting created scene-model corresponding to the scene-view in Fig (b), and Fig (e) the model transformed into the scene following recognition. Pose information relating to the found "hole-punch" is given in Fig 7.2. 5. The process leading to the recognition and grasping of the "bottle" and "cup" are shown in sequences Fig 7.2. 6 through Fig 7.2.12.



(a)



(b)



(c)

(d) (e)

*(a) Panning robot at a waypoint,(b) Table scene: Captured left image, (c) Model of hole-punch (d) Scene-model of (b), (e) Hole-punch found and model transformed into the scene.*

**Fig 7.2. 4 Hole-Punch search**

```
0.555833 0.315053 -0.769280
-0.351152 0.927769 0.126241
0.753487 0.199965 0.626316

493.338318 -154.193161 270.673859

Rotation about x axis= 17.706822  degrees
Rotation about y axis= -48.893348  degrees
Rotation about z axis= -32.282967  degrees
```

***Transformation matrix and related rotations***

**Fig 7.2. 5 Pose information of "Hole Punch"**

**Fig 7.2. 6 Left image of "Bottle" in scene**

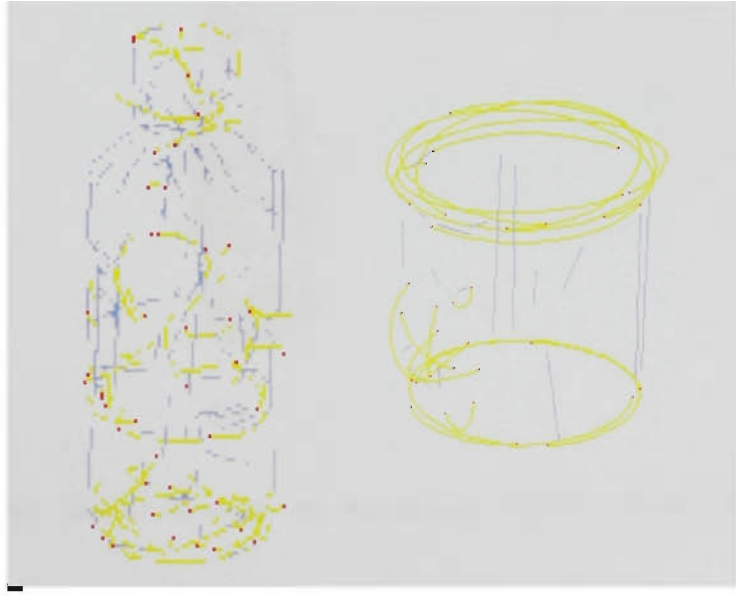
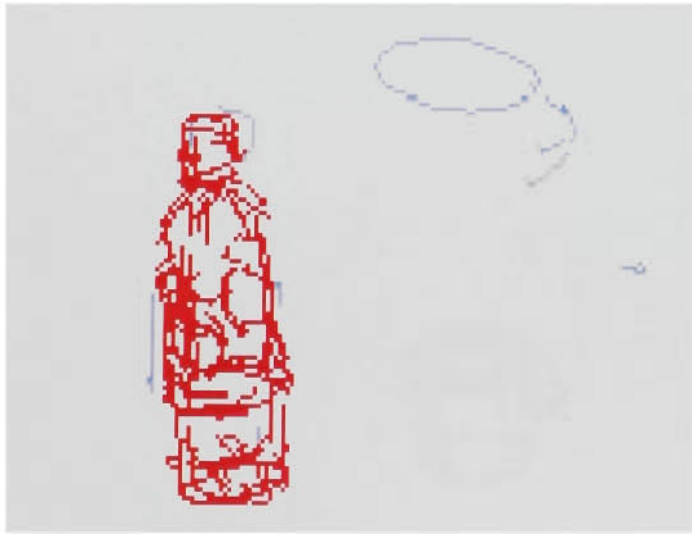


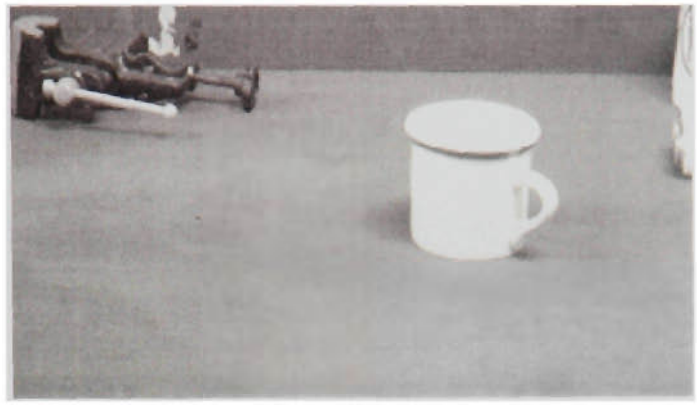
**Fig 7.2. 7 3D models of "Bottle" and "Cup"**

(a)

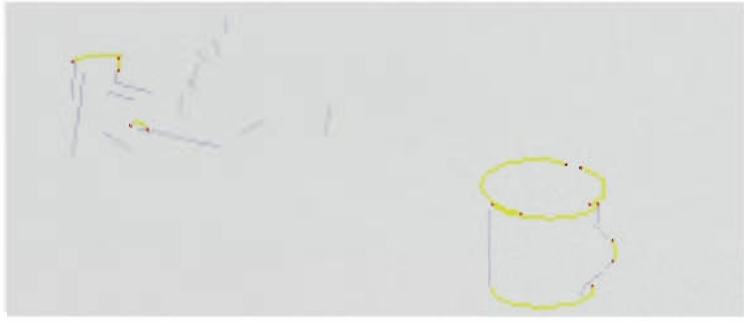


(b)

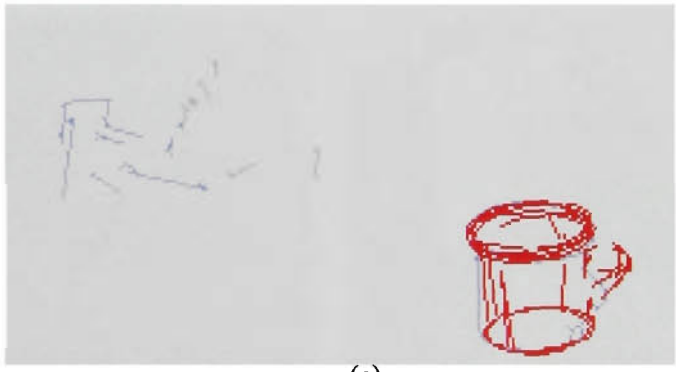
**Fig 7.2. 8** **(a) 3D model of "Bottle" in scene (b) Recognised "Bottle" transported into the scene**



(a)



(b)



(c)

**Fig 7.2. 9** **(a) Left image of "Cup" in scene, (b) 3D model of "Cup" in scene, (c) Recognised "Cup" transported into the scene**

```
computed transform wrt camera frame
computed tranform is
-0.769674 0.220012 -0.599330
-0.294048 0.711089 0.638662
0.566689 0.667793 -0.482613

563.264954 -596.585205 1412.442627
Scene Object Found with Head at X=-47.397186 Y=-411.613678
Z=980.958069
```

(a)

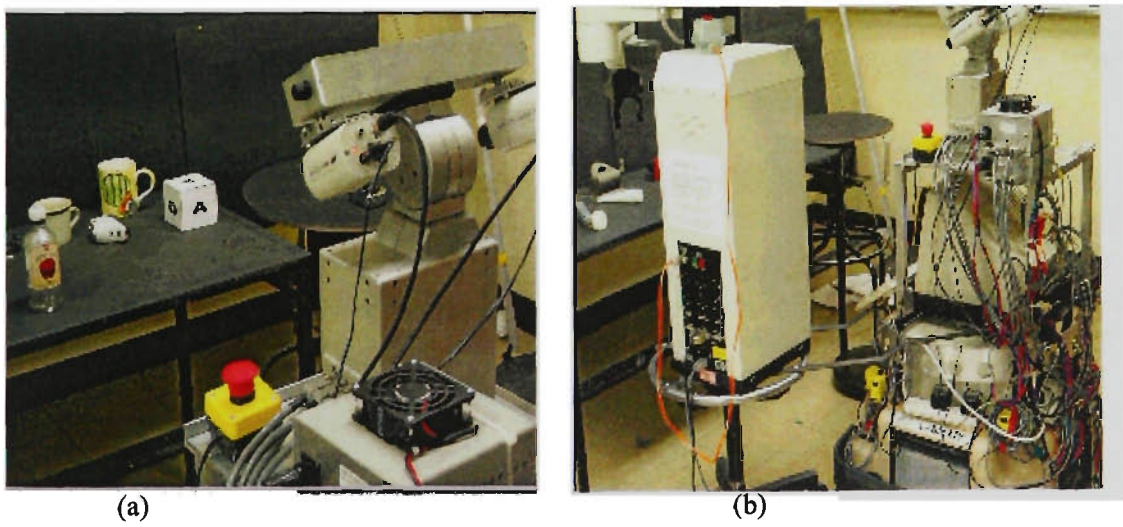
$$\begin{bmatrix} -1.00 & 0.00 & 0.00 & 353.96 \\ 0.00 & 1.00 & 0.00 & -664.40 \\ 0.00 & 0.00 & -1.00 & 299.39 \end{bmatrix}$$

(b)

(a) Pose of Cup in Scene

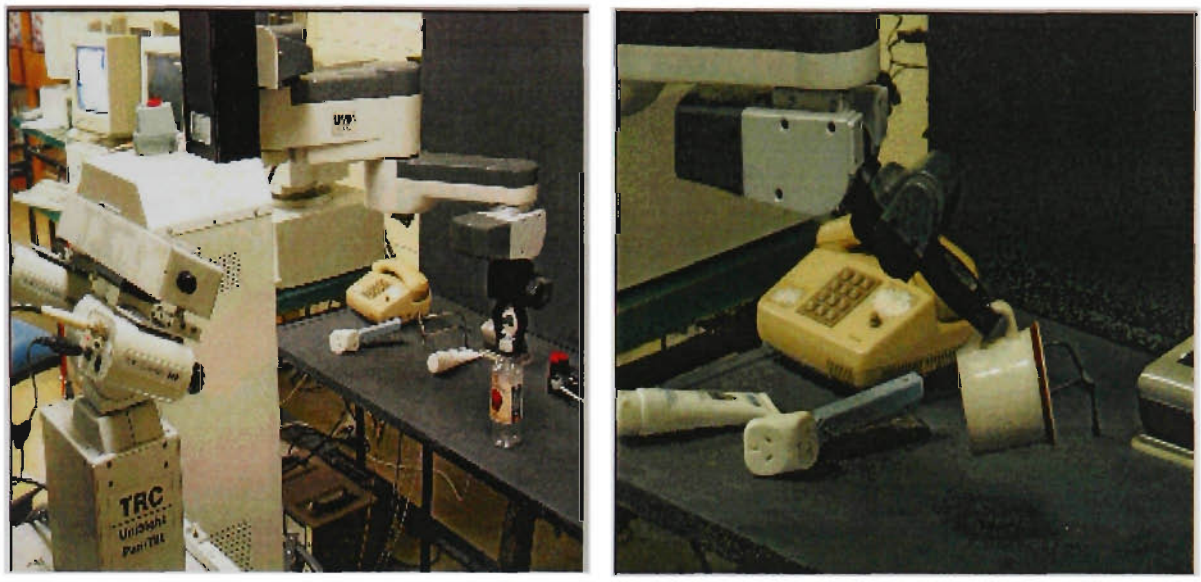
(b) $T_{Arm}^{Cup}$

**Fig 7.2. 10 Cup transforms for transportation into scene and grasping**



(a) (b)

**Fig 7.2. 11 (a) Head gazing at scene (b) Docked robot arm**




(a) (b)

**Fig 7.2. 12 Robot retrieving located objects**

### 7.2.1 Processing Times

At present the average navigation time for a trip to waypoint 2 (approx. 1.49m), including recognition and object retrieval, is 13.5 minutes.[1]

The journey consisted of 2 angular base rotations and 9 translations. The linear step size chosen for the mobile base was 200mm. Final linear step size was halved until the remaining distance less than 32mm, i.e., for a linear movement of 1490mm, linear steps consist of 7 x 200mm 1 x 45mm, 2 x 22.5 mm). EKF measurements were performed at each step, these included up to 3 head adjustments to achieve feature lock on. Objects sought were the bottle, cup and cube (the cube can not be grasped with current grippers.

[1] The average time is based on 6 runs, time to manually dock the arm is not included.

**Average processing times:**

| | |
|---|---|
| (i) feature measurement and localisation during navigation | 40 sec |
| (ii) object recognition[2] | 4 min |
| (iii) object retrieval[3] | 35sec |

For a viable commercial system, fast processing and operating times are crucial. The work carried out in this research is essentially only at a prototype stage and processing times have not been a major consideration for the following reasons. Emphasis has been on proving algorithms, integrating system modules and establishing a working system with the hardware resources available. The fastest PC used for MROLR has only a 90MHz processor. This however has not been the greatest limiting factor to processing speed. Our BiSight head has an undesirable low frequency mechanical resonance that is often excited following either the head or mobile base movement. A delay of several seconds following either the head movement or base movement is necessary to allow such head oscillations to dissipate and prevent blurred images.

In addition, several images formats are currently being used and converted on the fly to the required format. Images are also transmitted serially from the on board PC to a stationary one.

Each of these overheads can be eliminated with new hardware and software rewritings and improvements. It is worth noting that Davison [Dav98] attained real-time tracking of approximately 5 measurements/second during navigation and map building. MROLR should be able to achieve a similar performance. Given that PC processors now operate in excess of 1.5 GHz, it is not difficult to envisage a possible time reduction of at least an order of magnitude (i.e., from 13.5 minutes to 1.35minutes).

### 7.2.2 How Reliably can Target-objects be Retrieved?

Successful recognition and retrieval outcomes will in general occur if the following rules are adhered to:

- objects are be with reach of the arm (approx. 500mm) and graspable
- spacing between objects is sufficient to allow the gripper to move between them.
- spacing between objects is at least as great as the largest linear dimension of the object. This will ensure clique features of one object, do not overlapped with cliques of another during the matching process
- ambient lighting is sufficient and remains relatively constant
- the surface on which the objects lie is of a dark non reflective texture that minimises glare and formation of shadows

[2] From time the head is in the correct position (i.e., pan search times not included)

[3] Following recognition

Following recognition the object's pose is determined via precise geometric computations and well established analytical methods, therefore given perfect data, its location and orientation will be accurately found.

This data consists of:

- camera calibration parameters,
- stereo head calibration parameters,
- 3D object models, 3D scene-models,
- a homogeneous (4 x 4) transform of the object-model with respect to its position in the scene,
- grasp location data (in terms of x, y, z, yaw, pitch, roll), obtained from the 3D object-model,
- a homogeneous (4 x 4) transform of the head coordinate frame with respect to the robot arm's base coordinate frame,
- a homogeneous (4 x 4) transform relating the object's grasping location with respect to the robot arm's base coordinate frame. This final transform directs the arm (driven by its controller) to retrieve the object.

It is not feasible to accommodate all possible requested gripper grasping orientations, as the robot arm has a limited reach and range of movements. In addition the propagation of data errors that will inevitably accumulate, may also on some occasions preclude successful object retrieval. Quantification of this uncertainty is of course desirable, the considerable number of variables that may affect outcomes however make this task difficult. Consider for example just 3 variables, lighting conditions, object local surface features, and textures. These affect the quality of the scene-model obtained, and consequently the accuracy of recognition and the resulting transform.

The dependence on a transform of high precision to solely guide the robot arm is undoubtedly a fundamental weakness of the retrieval system. In effect the object retrieval system is an open loop one. An improvement in the form of visual feedback (visual servoing) could be added to make the retrieval process more robust, either by more effective use of the existing stereo head and/or with the supplementation of a miniature camera attached to the gripper (i.e., an eye in the hand).

Further testing to obtain comprehensive quantitative and qualitative performance metrics of MROLR, i.e., bounds on recognition and retrievablity, will be performed once the automated docking transport base with arm, is an integral working part of the system. It is likely that visual servoing, as described above, will also have been added by then. These enhancements are further discussed in Chapter 8.

## *7.3 CD with Video Clips of MROLR Performing*

Three short video clips of MROLR performing several routines are provided on the accompanying CD.

The first shows the automated model production of the alphabet cube on the lighter of the portable modeling table. The completed model is finally rotated about its axes to show typical

duplication of edge lines following integration. The cleaner model, after their removal, is also shown.

The second shows MROLR navigating to the table of objects, waypoint 2, tracking two known-features as it travels. On reaching the table the target-object, in this instance, the alphabet cube is recognised and the model is correctly transported into the scene. The mobile base then navigates home (waypoint 0) again tracking the same two known features. A black coloured ellipse of measurement uncertainty is displayed in close proximity to the feature being tracked.

The third video clip commences with the mobile robot at the table of objects; it then proceeds to recognise the target object, the bottle in this instance. The robot arm is manually docked and the bottle retrieved. Next the bottle is manually relocated to two additional positions on the table, and the process of recognition and retrieval repeated. Of these two additional attempts, only one retrieval was successful, the other being close (within 1 cm of success).

A fourth video clip showing the "transport mobile robot" under test, carrying the arm has also been provided. This vehicle is not yet able to be remotely communicated with or to dock with the "scout master mobile base".

## 7.4 Conclusion

The tests carried out in this chapter commenced with the creation of a small database of ten 3D models of commonly found household items. MROLR was then directed to autonomous navigate to selected waypoints in search of objects selected from the database. It successfully performed these navigational tasks, mapping its way while simultaneously tracking several known landmark features.
On reaching the waypoint destination (in each case a table laden with objects), the active head panned the scene in search of the object. With the object present and in range of the arm, recognition, pose determination and retrieval occurred. Obstacles avoidance was not attempted during these tests as MROLR has only limited ability to detect obstacles (refer section 5.8.1.2). These results confirm the methodology of this research and are sufficiently encouraging to support continuation of the project.

The contributions of this work, and planned future improvements and extensions (several of which are currently in progress), are summarised in Chapter 8.

# Chapter 8 Conclusions and Further Work

## 8.0 Introduction

In this, the final chapter, the main contributions of this work are summarised and proposed extensions for continuation in the future outlined.

## 8.1 Main Contributions of This Work

The major contribution made by this research is in the design and implementation of a new type of "service robot", comprising a navigating mobile robotic platform with active vision and object retrieval abilities. Given a specified destination (with reference to a home base coordinate frame), its navigational capability enables it to reliably localise and map-build in an unknown indoor environment, using information from measurements of arbitrary landmark-features and robot-odometry. The resulting map is also available for future navigational purposes such as the return journey, or recovering from accumulated positional errors that are a result of estimation errors and wheel slippage.

The specified destination is required to be in close proximity (< 0.5 metres) to a surface, such as a table, on which a desired object is to be located and retrieved. The recognition ability of the MROLR enables it to compare a 3D (a prior developed) model to scene-objects modeled while surveying the location, and if a suitable match is obtained (based on 3D geometric features) verify its pose by aligning the 3D model with the 3D matched scene-model. Using information from pose measurements, a transform is established locating the sought object with respect to the active head. This transform is then available to guide the docked robot arm to grasp, retrieve, and to place the object on the platform, for transport back to the base home position.

### 8.1.1 Related Contributions

Related contributions stem from:

- Development of automated 3D object-model creating facilities. This consisted of the design and construction of two, portable motorized turntables, and related software modules. An object to be modeled is placed on the table and the active stereo head forms 3D view dependent wire frame models as the table is accurately rotated. These sequences are finally integrated to form a complete (view independent) 3D model suitable for storage in a database. The establishment of a model database of sought objects is a prerequisite for their use in the recognition process.

- Development of a model editor to facilitate the manual removal and/or addition of 3D segments in formed models. The editor was built on the incomplete "geomstat" module of TINA. The major contribution to the geomstat software module was the addition of an

"edited model" output facility. The need for the addition of 3D lines to models arises on occasion as a result of poor lighting conditions, or edges that are difficult to match. More commonly, line duplication occurs due to slight misalignments during the integration process, and the editor now allows these to be removed.

- Preliminary design and construction of a "slave transport" mobile robot base and related software has commenced. This mobile robot will carry the robot arm that ultimately grasps and retrieves the target object. Once the "master scout" mobile base has accomplished its task of navigating to and locating the desired object, the "transport" base will be sent a command to dock with the "master". Docking ensures that a known, predetermined distance exists between the gripper and the head coordinate frame.

- Extension and modification of existing recognition algorithms to facilitate automated recognition while searching and scanning of the scene. The added algorithm utilises nested iterative loops in which the seven parameters are systematically modified (in a "fuzzified" like, fine-to-course adjustment) to reflect a reduction in data “confidence levels”.

- Extension and modification of existing active vision based navigation software to work with the designed mobile base. This included the writing of an obstacle detection and avoidance module.

- Development of kinematic solutions from object pose information, to enable the robot arm to grasp and retrieve the desired object.

- Preliminary implementation of a "3D Virtual Environment" for simulating MROLR. This will be useful for evaluating alternative map-building strategies, object grasping locations, as well as for demonstration and educational purposes.

## *8.2 Future Directions and Work*

As stated at the conclusion of Chapter 1, motivation for this work stems from the desire to ultimately develop a system capable of rapid and reliable retrieval of objects in both domestic and industrial environments. Potential applications for such an improved system range from simple domestic and industrial “robotic aids”, to “assistants” for the severely physically disabled or visually impaired.

The work completed to date goes a considerable way to meeting the objectives of this thesis. However, MROLR is a prototype, and to develop it into a useful product requires improvements in several areas. Presently, self-localisation and map building is quite a slow process. There are several reasons for this. Navigational feature measurements are only performed while the base is stationary, and several image format conversions are required. The stationary base requirement is in partly dictated by the Extended Kalman Filter algorithm requiring constant distant increments for measurements. Also occasionally severe stereo head motion-vibration occurs, producing

blurred images when the head moves concurrently with the base. Elimination of the need for several image formats is a relatively straightforward task, and the existing software modules will be modified to accept a single standard image formats in the near future. The stereo head vibrations appear to be a structural problem resulting from natural harmonic motions. The purchase of an improved head may be the simplest solution to overcome this problem. With faster computers, it is not unreasonable to expect measurement updates at better than eight per second. Of course these time delays do not arise if the mobile robot utilises the navigational mode that relies solely on odometry. For most indoor environments where floor levels are even and wheel slippage negligible, this simpler mode of operation is acceptable and quite reliable.

Currently only one scan and search sequence has been implemented. Effectively the target object must reside on a surface in close proximity to the specified destination. An experiment with multiple scan and search sequences has been done. The mobile robot was guided around the perimeter of a table while searching for an object. This proved successful when the object was eventually located at the far end of the table. The ability to move around the boundary area of benches and other objects will require additional navigational algorithms to be incorporated, to determine and negotiate complex path trajectories -- a challenging future project.

In the more immediate future, it is intended to improve on the current method of object recognition. One simple improvement would be to add colour to the object feature list. This could greatly help discriminate between objects of similar shape (i.e., white cup instead of green cup). More complex model building algorithms will also be considered, perhaps using Generalised Cylinders or Octrees.

The use of a separate "slave transport" mobile base has both potential advantages and drawbacks. Placing a robot arm on the master mobile base was not a feasible option with the current equipment, due to size and weight constraints. The advantage of using two vehicles is that a relatively lightweight fast acting, low energy mobile base, equipped with active cameras, can initially be used to navigate about, in a search, recognition and location phase. This is followed by a second phase, during which the slower mobile-crane like vehicle is instructed to automatically dock with the "master" and then guided to pick up the object (a" brain and brawn" collaboration). Substantial progress on the design and construction of the "transport" mobile base has been made (details are given in appendix B).

Adding "gesture recognition" capabilities to MROLR has also been considered. This would build on the work of Wingate and Stoica relating to Apprentice Robots [WS96], [WS97]. A Master could point towards the target-object thus making it easier to request the retrieval of an object. There is little doubt that as the methods of communicating with and directing robots becomes more "natural" for humans, such as with speech and hand gestures, the use and appeal of robots will grow. Speech control trials on several robots have already been undertaken by Moyssidis and Wingate [MW02].

As acknowledged in chapter 7, the sole dependence on a transform of high precision to guide the robot arm is a fundamental weakness of the retrieval system. In effect the object retrieval system is an open loop one. An improvement in the form of visual feedback (visual servoing) to make the retrieval process more effective and robust, was suggested in chapter 7. This could be

accomplished by more effective use of the existing stereo head and/or with the addition of a miniature camera attached to the gripper (i.e., an eye in the hand). The concept of using cameras attached to robots is not new. Researchers such as Smith et al [SBP97], Hauck et al [HPRSF99], Wijesoma et al [WWR93] and others have successfully utilised them for end effector guidance. A future software module and micro camera could be added to incorporate direct "hand-eye" coordination.

In their current form the simulation modules described in Chapter 6 would be useful for teaching and demonstration purposes. A possible extension to the simulation modules (briefly mentioned in Chapter 6) would be for all created objects to be positioned in the "virtual" scene (in close proximity to their real world x y z location). Mouse selecting the target object (or verbally selecting it, if voice recognition was added) could then initiate a simulation run. If the simulation run proved successful this could be linked with the physical task of navigation and object retrieval. On retrieval by the "physical" docked arm, the objects would be removed from the "virtual" scene. This would provide a direct way of target object selection as well as a visual means of readily establishing the proximity of the desired object position, thus considerably simplifying the navigation, search and retrieval tasks.

## *8.3 What Follows ?*

Three appendices follow this final chapter, provided to expand upon the work outlined in the preceding chapters, as well as to outline some interesting preliminary work undertaken in search of suitable algorithms to meet the needs and objectives of this thesis.

Appendix A describes early approaches and methods considered and evaluated, but ultimately not followed.

Appendix B provides specific details of ancillary software and hardware developed in support of this continuing work.

Appendix C continues with the algorithmic development for continuous tracking of multiple features using active vision commenced in Chapter 5.

A comprehensive Bibliography follows the appendices. This is a collection of references maintained and used by the author.

# Appendix A Early Attempts to Obtain Object Modeling Algorithms

## *A 1.0 Introduction*

The work of this thesis commenced in 1995 and at that time, obtaining object boundary outlines with metric information suitable for 3D object modeling and recognition was still relatively in its infancy. The problem still attracting a great deal of attention from eminent researchers, as is evident from the vast array of publications being produced, for example Z. Zhang and others, [Zha02b], [YZ02], [SZ00], [SNWGH99]. In the process of obtaining an effective recognition module for MROLR a number of novel approaches were considered. Although algorithms were developed and tested, and results published and presented, ultimately these were not utilised in the recognition module.

This section reviews the work carried out and contributions made during that research period.

## *A 1. 1 Structure Recovery of Objects Using Multiple Camera Views.*

An early approach using "line features in multiple camera views", Wingate [Win97a], [Win97b], was formulated around the algorithm of Taylor and Kriegman [TK95]. This work differed to that of Taylor and Kriegman in that it did not rely on a priori knowledge of the cameras focal length.

An abstract from "3D Structure Recovery of Objects Using Line Features in Multiple Camera Views", [Win97a] is reproduced below:

*Abstract*
*The work carried out represented an initial attempt to recovery the 3D structure of rigid objects from multiple views. The concept involved images captured from a roving mobile robot equipped with a single camera whose pan and tilt are controllable. The algorithms formulated use straight-line feature correspondence obtained from a minimum of 3 views and an objective function that minimises the squared difference between projected edge segments and image segments. The structure of the object in terms of a scaled 3D-line drawing as well as the position of the cameras is returned and available for use as a scene model. Edge outlines were obtained using a Canny edge detector and straight lines fitted to these using a recursive line fitting algorithm. Results of simulated data are provided using a pin hole camera model and these are compared with results using real images in which line correspondence matching is carried out manually.*

## *A 1. 2 Trinocular Stereo*

A trinocular stereo vision, rectifying images to enhance epipolar matching was another method investigated. Algorithms were developed and results compared with those of two camera stereo.

The work performed was presented in Wingate [Win99]. Details are outlined in Section A 1.2. 1 below.

### A 1.2. 1 Trinocular Vision (3 Camera Stereo)

The notion of using trinocular stereo vision is appealing because the additional camera provides a second epipolar line in each image and the intersection of these lines correspond to a match. False matches are readily eliminated by verification of correspondences in each image (A 1.2. 1 below). Rectification of the images such that the image planes are coplanar and parallel to the lines joining the focal centres results in the epipolar lines between images 1 and 2 becoming horizontal, and also vertical epipolar lines between images 1 and 3. The advantage of using trinocular stereo over binocular stereo comes at the cost of increases in algorithmic complexity and overhead processing time. Ayache [Aya91] states that the rectification of k images necessitates storing k 3x3 matrices, and forming six products, six additions and two divisions per rectified point. R. Jarvis and others [Jar94] have used quad stereo. An advantage of four cameras is that testing for vertical, horizontal and diagonal matches can be performed simultaneously thereby reducing the likelihood of matching outliners even further.

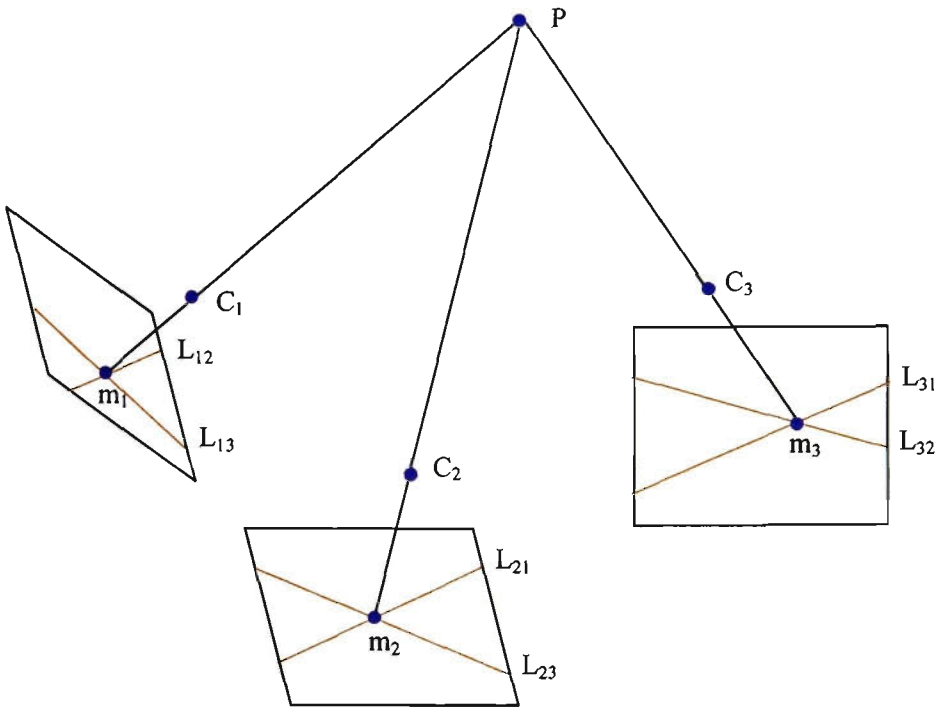



**Fig A 1.2. 1 Trinocular stereo: corresponding matches at intersection of epipolar lines**

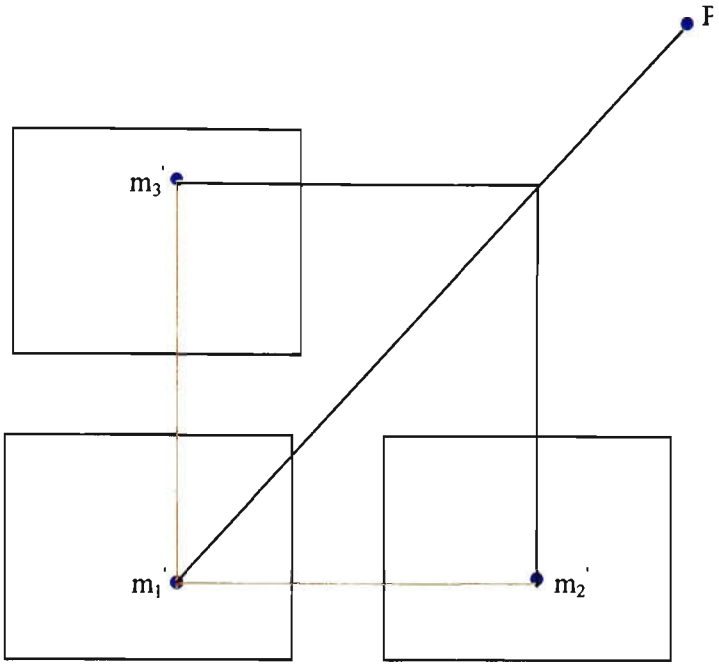



**Fig A 1.2. 2 Epipolar lines and corresponding matches following image rectification**

Ayache's procedure [Aya91, pp.30-41, (outlined below)] was implemented for trinocular rectification. Initially coding was carried out in Matlab and subsequently in 'C'. The theoretical basis of the approach is outlined in the following sections.

### A 1.2. 2 Image Modeling

The standard pin hole camera is used, and for clarity of notation this is redrawn in Fig A 1.2.2.1.

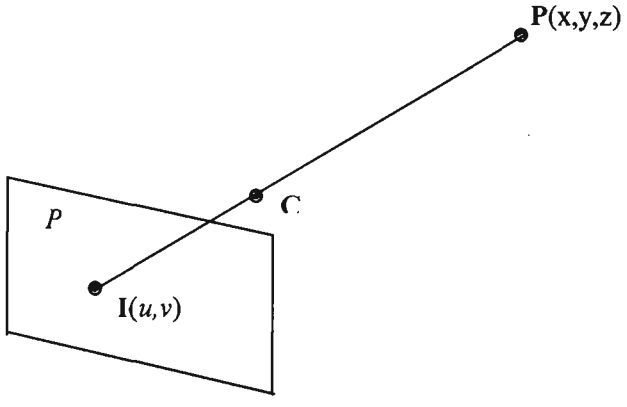



**Fig A 1.2.2. 1 Standard pin hole camera, optical centre C, image point I focal plane *P***

The transformation from **P** to **I** is modeled by a linear transformation **T** in projective coordinates. Letting $\mathbf{I}^* = (U, V, S)'$ be the projective coordinates of **I** and $(x, y, z)'$ be the coordinates of **P**,

$$\mathbf{I}^* = \begin{pmatrix} U \\ V \\ S \end{pmatrix} = \mathbf{T} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \qquad \text{A 1}$$

Where **T** is a 3 x 4 matrix, generally called the *perspective matrix* of the camera.

For $S \neq 0$ the image coordinates of **I** are defined by:

$$\mathbf{I} = \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} U/S \\ V/S \end{pmatrix} \qquad \text{A 2}$$

**T** is generally determined by analysing the image of a calibration grid where the positions in the image of the intersection points are known with precision. Various established calibration procedures exist for determining **T** (The Tsai routine [Tsa87] was used for this work). The 3x4 matrix **T** consists of twelve parameters, but these are defined up to a scale factor only. Thus, an additional constraint must be found. The simplest is to assume that $T_{34}$ is non-zero and to set it to 1. Following calibration, the 11 parameters comprise of the intrinsic and extrinsic parameters of the pinhole model.

For later use the following notation introduced by Ayache is defined.

$t_{ij}$ is the $(i, j)$ element of **T**, and $\mathbf{t}_i$ is the vector composed of the first three elements of the row $i$ of **T**:

$$\mathbf{t}_i = (t_{i1}, t_{i2}, t_{i3})' \qquad \text{A 3}$$

The coordinates of the optical centres $\mathbf{C}_i$ $(xc_i, yc_i, zc_i)$ are obtained by solving the system

$$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} = \mathbf{T}_i \begin{pmatrix} xc_i \\ yc_i \\ zc_i \\ 1 \end{pmatrix} \qquad \text{A 4}$$

### A 1.2. 3 Image Rectification

Image rectification is performed by a linear transformation in projective coordinates normally resulting in either horizontal or vertical conjugate epipolar lines, points with coordinates $(u_i, v_i)$ are transformed to new coordinates $(u_i', v_i')$. By appropriate choice of coordinate frames, a point $(u_i', v_i')$ in image 1 has the line segment $v_2' = v_1'$ of image 2 as its conjugate epipolar line, Fig A 1.2.3.1.

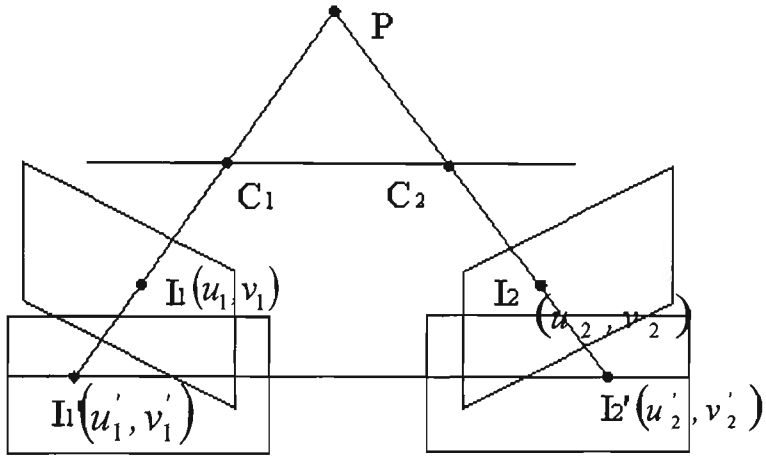



**Fig A 1.2.3. 1 Rectification of two images.**

Initially rectification for binocular stereo will be considered and extended for a trinocular stereo system. To begin it is necessary to define two new perspective matrices **M** and **N** which preserve the two optical centers $\mathbf{C}_1$ and $\mathbf{C}_2$. These facilitate the transformation from coordinates $(u_i, v_i)$ to new coordinates $(u_i', v_i')$ in each image $i$.

The following constraints on the new perspective matrices **M** and **N** are imposed:

1. The optical centers of **M** and **N** are $\mathbf{C}_1$ and $\mathbf{C}_2$ respectively (to give a unique match between image points $\mathbf{I}_i$ and $\mathbf{I}_i'$ before and after rectification).

2. The focal plane of **M** is identified with that of **N** (to produce parallel epipolar lines in both images).

3. For any point **P** (not in the optical plane), the image points $\mathbf{I}_1'$ and $\mathbf{I}_2'$ obtained by **M** and **N** respectively are such that $v_1' = v_2'$.

Let

$$\mathbf{M} = \begin{pmatrix} m_1^t & m_{14} \\ m_2^t & m_{24} \\ m_3^t & m_{34} \end{pmatrix} \qquad \mathbf{N} = \begin{pmatrix} n_1^t & n_{14} \\ n_2^t & n_{24} \\ n_3^t & n_{34} \end{pmatrix} \qquad \text{A 5}$$

The application of these constraints lead to matrices $\mathbf{M}$ and $\mathbf{N}$ defined by:

$$\mathbf{M} = \begin{pmatrix} ((\mathbf{C}_1 x \mathbf{C}_2) x \mathbf{C}_1)' & 0 \\ (\mathbf{C}_1 x \mathbf{C}_2)' & 0 \\ ((\mathbf{C}_1 - \mathbf{C}_2) x (\mathbf{C}_1 x \mathbf{C}_2))' & \|\mathbf{C}_1 x \mathbf{C}_2\|^2 \end{pmatrix} \qquad \text{A 6}$$

$$\mathbf{N} = \begin{pmatrix} ((\mathbf{C}_1 x \mathbf{C}_2) x \mathbf{C}_2)' & 0 \\ (\mathbf{C}_1 x \mathbf{C}_2)' & 0 \\ ((\mathbf{C}_1 - \mathbf{C}_2) x (\mathbf{C}_1 x \mathbf{C}_2))' & \|\mathbf{C}_1 x \mathbf{C}_2\|^2 \end{pmatrix} \qquad \text{A 7}$$

By considering a point in image 1, $\mathbf{I}_1(u_1, v_1)$ that is the projection of a point $\mathbf{P}(x, y, z)$ with coordinates:

$$\mathbf{P} = \mathbf{C}_1 + \lambda \mathbf{n} \qquad \text{A 8}$$

Where $\mathbf{n}$, is a direction vector of the segment $\mathbf{I}_1\mathbf{C}_1$. The projective coordinates of the new image $\mathbf{I}_1'$ of point $\mathbf{P}$ are then given by:

$$\mathbf{I}_1' * = \begin{pmatrix} U_1' \\ V_1' \\ S_1' \end{pmatrix} = \mathbf{M} \begin{pmatrix} \mathbf{C}_1 + \mathbf{n} \\ 1 \end{pmatrix} \qquad \text{A 9}$$

Now, since $\mathbf{C}_1$ is the optical centre of $\mathbf{M}$, $(\mathbf{C}_1, 1)' = 0$. As a consequence, the computation of $I'$ reduces to:

$$\mathbf{I}_1' * = \begin{pmatrix} U_1' \\ V_1' \\ S_1' \end{pmatrix} = \mathbf{M}' \mathbf{n} \qquad \text{A 10}$$

Where $\mathbf{M}'$ is the 3x3 matrix obtained by removing the last column of $\mathbf{M}$. Since $n$ is computed by an affine transformation of the coordinates $(u_1, v_1)$, it is sufficient to put

$$\mathbf{Q}_1 = \begin{pmatrix} ((\mathbf{C}_1 x \mathbf{C}_2) x \mathbf{C}_1)' \\ (\mathbf{C}_1 x \mathbf{C}_2)' \\ ((\mathbf{C}_1 - \mathbf{C}_2) x (\mathbf{C}_1 x \mathbf{C}_2))' \end{pmatrix} \begin{bmatrix} \mathbf{t}_2^1 x \mathbf{t}_3^1 & \mathbf{t}_3^1 x \mathbf{t}_1^1 & \mathbf{t}_1^1 x \mathbf{t}_2^1 \end{bmatrix} \qquad \text{A 11}$$

a 3x3 matrix, to obtain:

$$\begin{pmatrix} U_1' \\ V_1' \\ S_1' \end{pmatrix} = \mathbf{Q}_1 \begin{pmatrix} u_1 \\ v_1 \\ 1 \end{pmatrix} \qquad \text{A 12}$$

Proceeding symmetrically, putting

$$\mathbf{Q}_2 = \begin{pmatrix} ((\mathbf{C}_1 x \mathbf{C}_2) x \mathbf{C}_2)' \\ (\mathbf{C}_1 x \mathbf{C}_2)' \\ ((\mathbf{C}_1 - \mathbf{C}_2) x (\mathbf{C}_1 x \mathbf{C}_2))' \end{pmatrix} \begin{bmatrix} \mathbf{t}_2^2 x \mathbf{t}_3^2 & \mathbf{t}_3^2 x \mathbf{t}_2^2 & \mathbf{t}_1^2 x \mathbf{t}_2^2 \end{bmatrix} \qquad \text{A 13}$$

gives:

$$\begin{pmatrix} U_2' \\ V_2' \\ S_2' \end{pmatrix} = \mathbf{Q}_2 \begin{pmatrix} u_2 \\ v_2 \\ 1 \end{pmatrix} \qquad \text{A 14}$$

Rectification of images 1 and 2 thus reduces to the application of the two linear transformations in projective coordinates given by equations A12 and A14.

The elegance of Ayache's derivation for **Q**1 and **Q**2 is that it readily extends to multiple cameras, as is evident from the following.

#### A 1.2. 4 Three Cameras

For the case of three cameras, it is preferable to rectify the images to have horizontal epipolar lines between images 1 and 2, and vertical epipolar lines between images 1 and 3. For analysis the same steps as used for two cameras are adopted to compute new perspective matrices $\mathbf{M}, \mathbf{N}$ and $\mathbf{Q}$ from the optical centers $\mathbf{C}_1, \mathbf{C}_2$ and $\mathbf{C}_3$.
This results in:

$$\mathbf{M}=\begin{pmatrix}(\mathbf{C}_3 x\mathbf{C}_1)' & 0\\ (\mathbf{C}_1 x\mathbf{C}_2)' & 0\\ (\mathbf{C}_1 x\mathbf{C}_2+\mathbf{C}_2 x\mathbf{C}_3+\mathbf{C}_3 x\mathbf{C}_1)' & -(\mathbf{C}_1,\mathbf{C}_2,\mathbf{C}_3)\end{pmatrix} \qquad \text{A 15}$$

$$\mathbf{N}=\begin{pmatrix}(\mathbf{C}_2 x\mathbf{C}_1)' & 0\\ (\mathbf{C}_1 x\mathbf{C}_2)' & 0\\ (\mathbf{C}_1 x\mathbf{C}_2+\mathbf{C}_2 x\mathbf{C}_3+\mathbf{C}_3+\mathbf{C}_1)' & -(\mathbf{C}_1,\mathbf{C}_2,\mathbf{C}_3)\end{pmatrix} \qquad \text{A 16}$$

$$\mathbf{Q}=\begin{pmatrix}(\mathbf{C}_3 x\mathbf{C}_1)' & 0\\ (\mathbf{C}_2 x\mathbf{C}_3)' & 0\\ (\mathbf{C}_1 x\mathbf{C}_2+\mathbf{C}_2 x\mathbf{C}_3+\mathbf{C}_3 x\mathbf{C}_1)' & -(\mathbf{C}_1,\mathbf{C}_2\mathbf{C}_3)\end{pmatrix} \qquad \text{A 17}$$

Where $(\mathbf{C}_1,\mathbf{C}_2,\mathbf{C}_3)$ is the triple product of vectors $\mathbf{C}_1,\mathbf{C}_2$ and $\mathbf{C}_3$.

In equations A15 to A.17, $\mathbf{Q}$ is chosen to produce equality between the ordinate $v_3'$ of $\mathbf{I}_3'$ and the abscissa $u_2'$ of $\mathbf{I}_2'$, i.e., $v_3'=u_2'$.

Rectification is performed in exactly the same way as previously but now with three 3 x 3 rectification matrices $\mathbf{R}_1,\mathbf{R}_2$ and $\mathbf{R}_3$.

$$\mathbf{R}_i=\begin{pmatrix}(\mathbf{C}_{i-1} x\mathbf{C}_i)'\\ (\mathbf{C}_i x\mathbf{C}_{i+1})'\\ (\mathbf{C}_1 x\mathbf{C}_2+\mathbf{C}_2 x\mathbf{C}_3+\mathbf{C}_3 x\mathbf{C}_1)'\end{pmatrix}\begin{bmatrix}t_2^i x t_3^i & t_3^i x t_1^i & t_1^i x t_2^i\end{bmatrix} \qquad \text{A 18}$$

Note for $\mathbf{R}_i$ the convention that $i+1=1$ if $i=3$ and $i-1=3$ if $i=1$ applies.
Thus for three homologous points $\mathbf{I}_1',\mathbf{I}_2'$, and $\mathbf{I}_3'$,

$$\begin{aligned} v_2' &= v_1'\\ u_3' &= u_1'\\ v_3' &= u_2' \end{aligned} \qquad \text{A 19}$$

**A 1.2. 5 3D Reconstruction**

In principle a knowledge of $\mathbf{T}_1$ and $\mathbf{T}_2$ is sufficient to compute the three coordinates of any point P, given its two images $\mathbf{I}_1$ and $\mathbf{I}_2$. However Ayache points out that in the absence of an objective criterion, solving the whole system either by least squares or by Kalman filtering is desirable. Either approach extends naturally to trinocular stereo vision, and more generally to reconstruction based on an arbitrary number of cameras.

For this work least squares was used, and the solution for n cameras follows.

For $n$ cameras, set

$$\mathbf{A}\alpha = \mathbf{b} \qquad \text{A 20}$$

With $\alpha = (x, y, z)'$ and

$$\mathbf{A} = \begin{pmatrix} A_1 \\ \vdots \\ A_n \end{pmatrix} \quad \text{and } \mathbf{b} = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix} \qquad \text{A 21}$$

For which

$$\mathbf{A}_i \begin{pmatrix} (\mathbf{t}_1^i - u_i \mathbf{t}_3^i)' \\ (\mathbf{t}_2^i - v_i \mathbf{t}_3^i)' \end{pmatrix} \text{ and } \mathbf{b}_i = \begin{pmatrix} u_i \mathbf{t}_{34}^i - \mathbf{t}_{14}^i \\ v_i \mathbf{t}_{34}^i - \mathbf{t}_{24}^i \end{pmatrix} \qquad \text{A 22}$$

The least squares solution is then given by

$$\alpha = (\mathbf{A}'\mathbf{A})^{-1}\mathbf{A}'\mathbf{b} \qquad \text{A 23}$$

provided $\mathbf{A}'\mathbf{A}$ is invertible.

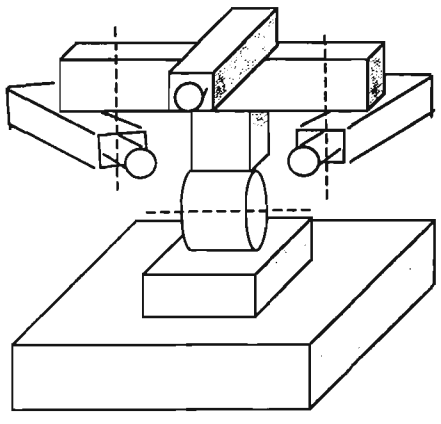


**Fig A 1.2.5. 1 4 Degree of freedom stereo head with third camera mounted**

An example of results obtained using three Pulnix colour cameras is given below.


**Fig A 1.2.5. 2 Left, centre and right camera images of a scene**

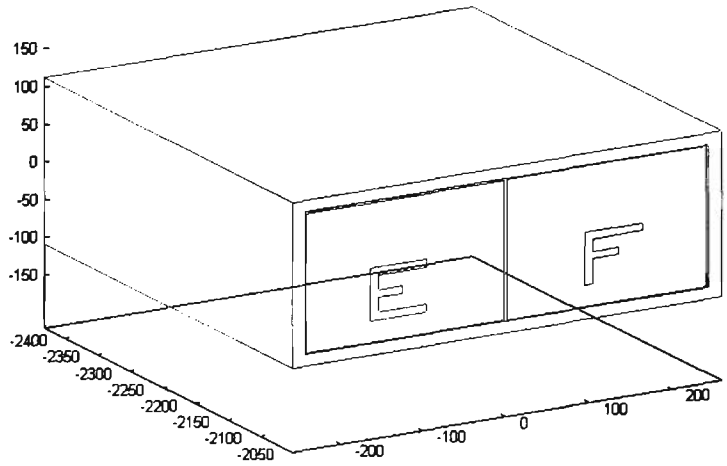



**Fig A 1.2.5. 3 Resulting 3D trinocular reconstruction**

## A 1. 3 Cluster Prototype Centring by Membership (CPCM)

A fuzzy probabilistic method for object boundary detection resulting in the development of a new progressive fuzzy clustering algorithm was also tested. The algorithm was given the acronym CPCM (Cluster Prototype Centering by Membership). Full details and results were published in [IWQ95], [ILW96].

The extent of this work is best summarized by the following extract from P.T.Im [Im97]

*"This work applies an enhanced progressive clustering approach, involving clustering algorithms and fuzzy neural networks, to solve some practical problems of pattern recognition. A new fuzzy clustering framework referred to as Cluster Prototype Centring by Membership (CPCM) has been developed. A Possiblistic Fuzzy c-Means algorithm (PFCM), which is also new, has been formulated to investigate properties of fuzzy clustering. PFCM extends the useability of the Fuzzy c-Means (FCM) algorithm by generalisation of the membership function. CPCM provides a flexible framework to integrate clustering methods that detect cluster substructures. Application development is focused on three problem contexts: (i) detection of contaminants in wool and paint defect on tile surface (region segmentation), (ii) identification of real object lines and circles (boundary detection) and (iii) recognition of a notched feature on an armature housing (general pattern recognition). Results obtained from these algorithms indicate robust clustering and accurate identification of cluster parameters (circle centre, radius, line gradient and corners) from real data silhouettes characterised by the presence of noise, fragmentation and partial obscurity."*

The bulk of this work was carried out by P.T. Im with the aid and supervision of M.Wingate. The following five papers on CPCM and applications were co-authored by M.Wingate: [ILW96],[IQWH95],[IQWH95b],[IWQ95].

# Appendix B Hardware and Software

## B 1.0 Introduction

To clarify and extend information provided so far in various chapters, specific details relating to ancillary hardware and software developed is provided in this appendix.

MROLR's system software comprises of numerous integrated platform modules (Linux, Windows, DOS) and many megabites of 'C/C++' based source code. Much of it is hardware specific and/or proprietary produced (i.e., written for the mobile-base controller, active stereo head-controller, robot arm-controller, etc.) and is of little use for other systems, or is not licensed for inclusion in this appendix. Ultimately, it is expected that source code written specifically for this project will be available on a CD for interested parties.

Accurate 3D object-model creation is essential for reliable object recognition. Several portable computer-controlled rotating tables needed to be designed and constructed to facilitate the object-model creations discussed in chapters 6 and 7. The motion controller requirements for the (partially complete proposed) mobile transport base, are similar to those of the rotating tables, thus for standardisation and ease of manufacture, identical designs were used. Section B 2.0 gives the circuit schematics of the controller and details of the electronic boards constructed. Specific details of the procedure for 3D object-model creation and its implementation in software are outlined in section B 2.1.

Kinematic analysis and solutions for a large range of robotic arms are well documented and understood. Driver binaries for the RTX UMI robot were provided by the manufacturer, however to incorporate kinematic and inverse kinematic solutions for the MROLR Simulation Module, required the development and coding of the relevant mathematical solutions. Forward kinematic and inverse kinematic solutions coded for the RTX robot arm, are detailed in sections B 3.

## B 2.0 Rotating Table and Mobile Transport Robot Controller Circuit Design

Motion controllers designed specifically for MROLR were standardised around the Motorola HC11 microprocessor and National Semiconductors LM629 motion-control IC. Thus, controllers for the model creating turn tables, are the same as used for the mobile transport base, except for the number of motor drivers utilised. All motors are fitted with precision optical encoders permitting velocity and acceleration against time profiles to be specified. This control is accomplished using the LM629, 32 bit motion controller, see list of features below.

LM629 features[2]

- 32-bit position, velocity, and acceleration registers
- Programmable digital PID filter with 16-bit coefficients
- Programmable derivative sampling interval
- 8-bit sign-magnitude PWM output data Internal trapezoidal velocity profile generator
- Velocity, target position, and filter parameters may be changed during motion
- Position and velocity modes of operation
- Real-time programmable host interrupts
- 8-bit parallel asynchronous host interface
- Quadrature incremental encoder interface with index pulse input

Circuit hardware and schematic details are shown in Figs B.2.0.3 to B.2.0.9. Interface to PCs is via an RS232 serial port.

Two portable, model creating tables were built, one robust with level adjusting legs to ensure a precise horizontal planar surface, the other lighter and more portable for ease of transport to different locations. Prior to the integration of the single view models formed, a rotation of each model about the Z-axis of the table, equivalent to the minus the angle the table was rotated, is required. The closer to horizontal the table surface is the better the model produced. Both motorised tables, previously shown in Fig 6.1.5 (a) and (b), are redisplayed reduced in Fig B.2.0.1

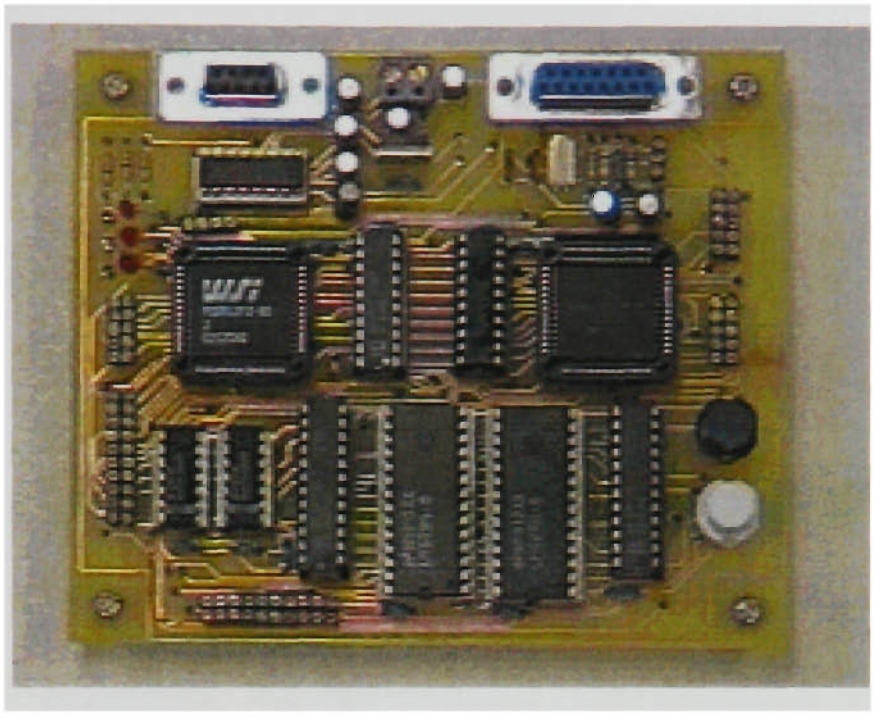
The transport mobile base (Fig 6.7.2) was designed and constructed to carry the robot arm and to ultimately dock with the scout master mobile base, as outlined in earlier chapters. In its present "carnation" it is accurately able to move to specified coordinates. This is demonstrated in the accompanying CD which shows the transport base carrying the robot arm. For convenience Fig 6.7.2 is also redisplayed reduced in Fig B.2.0.2

Wireless communication, and laser controlled docking still need to be added to make this facility a useful component of MROLR.

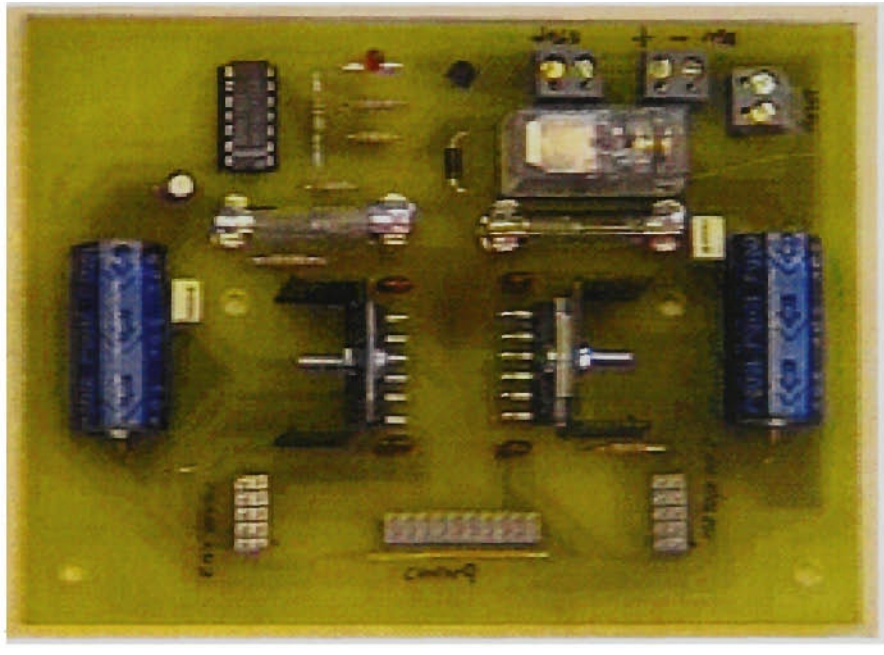
[2] Extracted from National Semiconductors application notes

**Fig B.2.0. 1 Portable motorised turn tables**

**Fig B.2.0. 2 Mobile Transport base and Arm**



(a) Double layer, microprocessor and precision motor controller circuit board. This board comprises 5 basic circuit sections, schematic circuit details are given in Figs B.2.0.4 to B.2.0.8.



(b) Motor Driver Board : Schematic circuit details are given in Fig B.2.0.9

**Fig B.2.0. 3 Controller hardware**

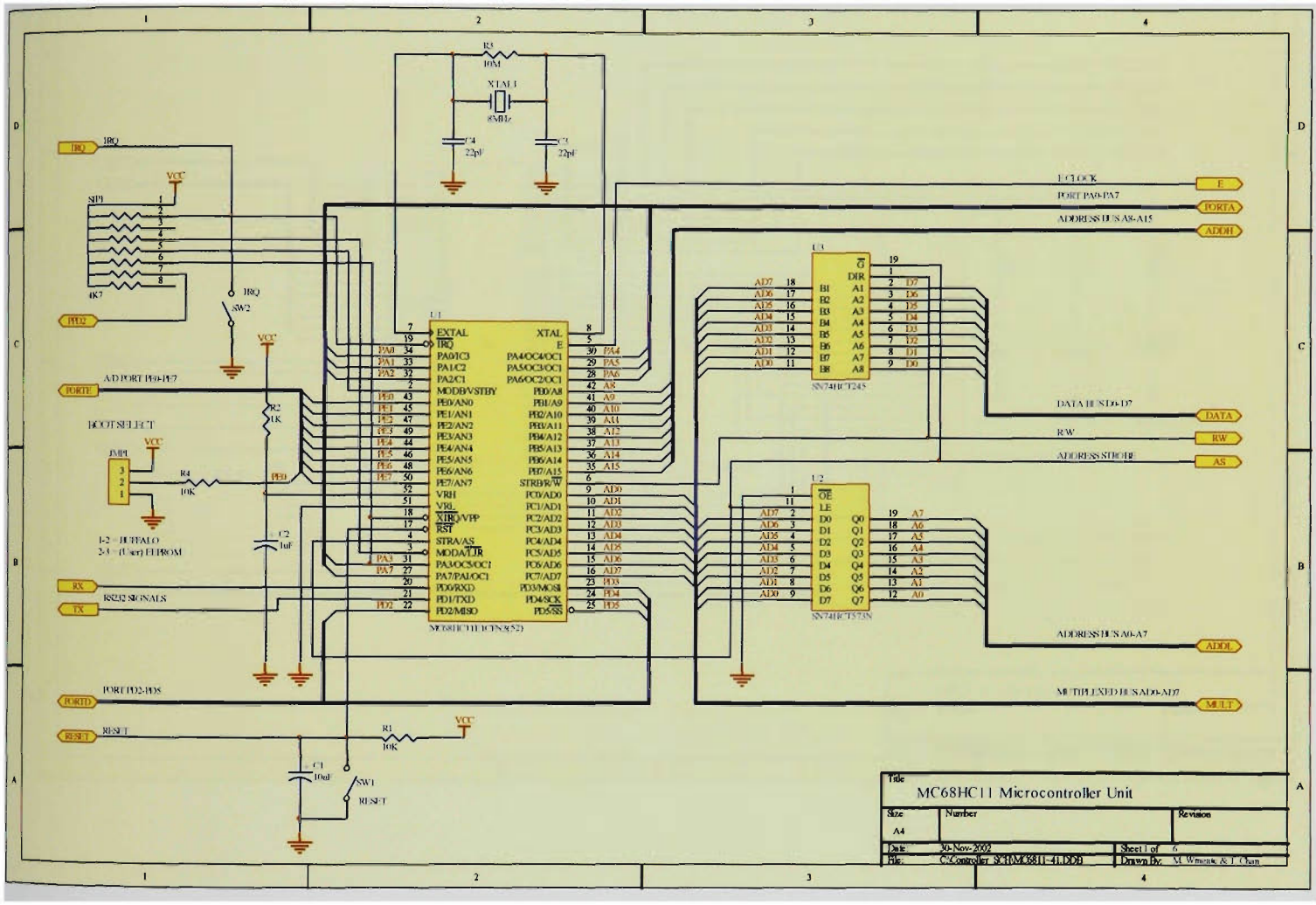



**Fig B.2.0. 4 MC68HC11 Microprocessor and interface circuit**

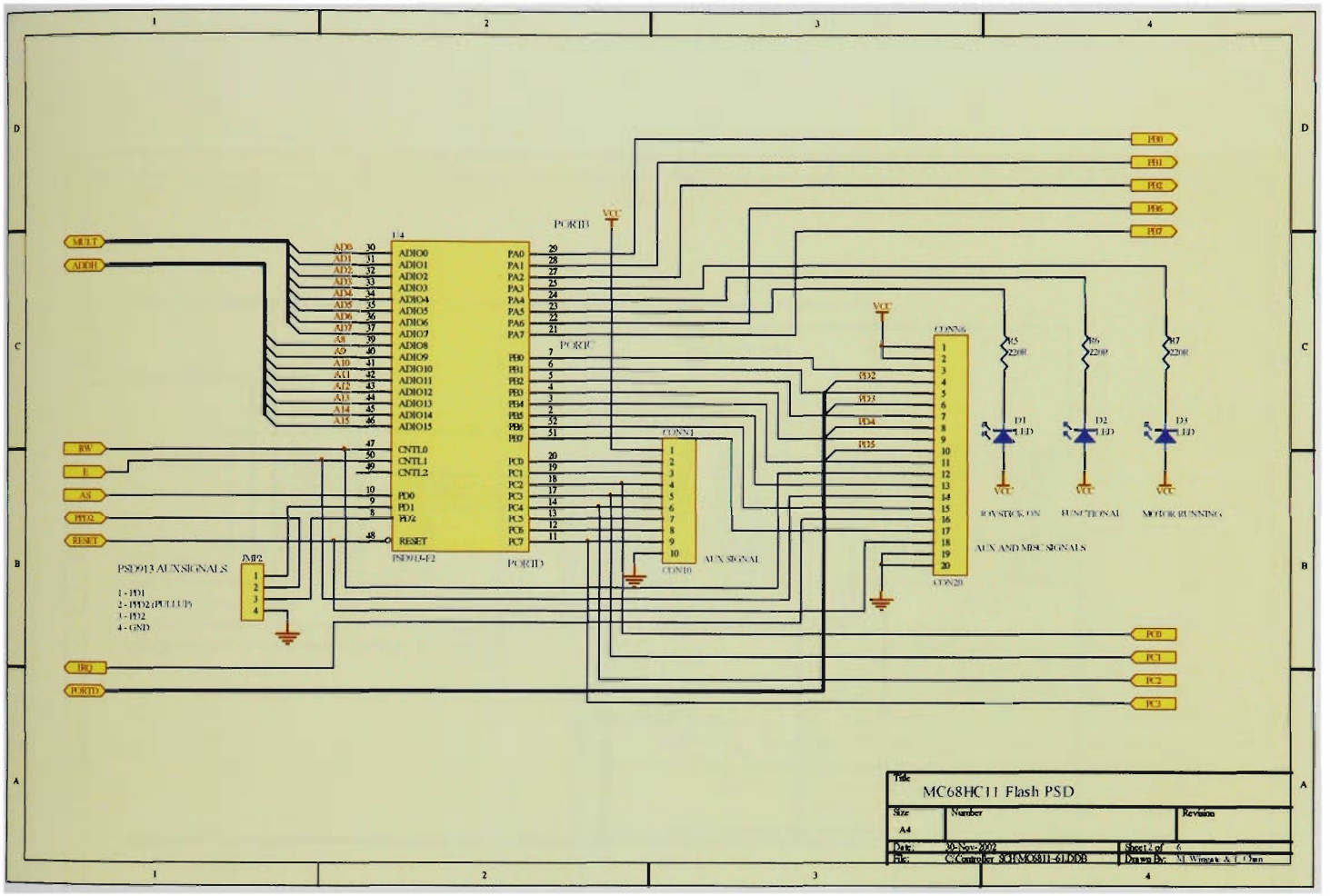



**Fig B.2.0. 5 Flash PSD memory circuit**

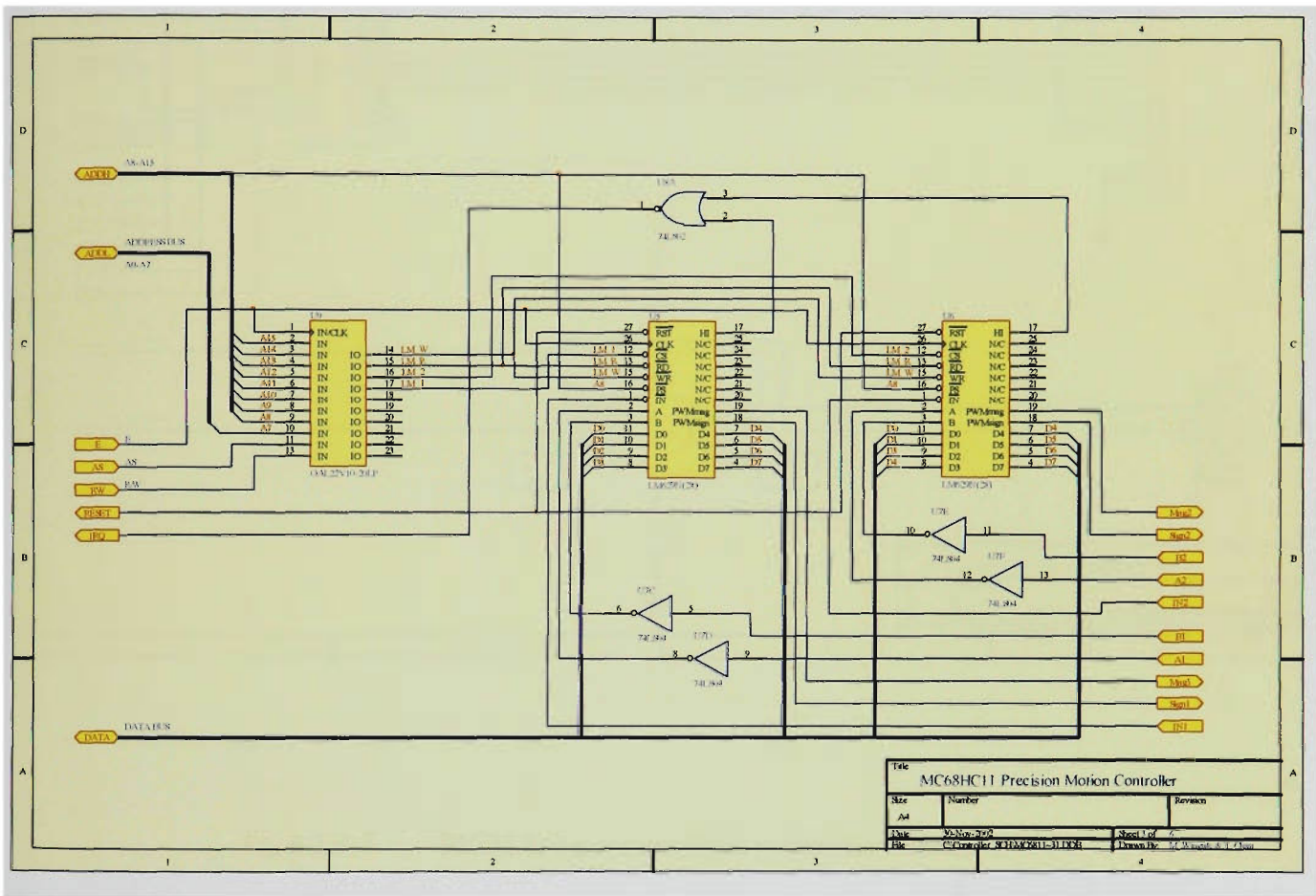



**Fig B.2.0. 6 Motion controller circuit, utilises LM629 motion controllers**

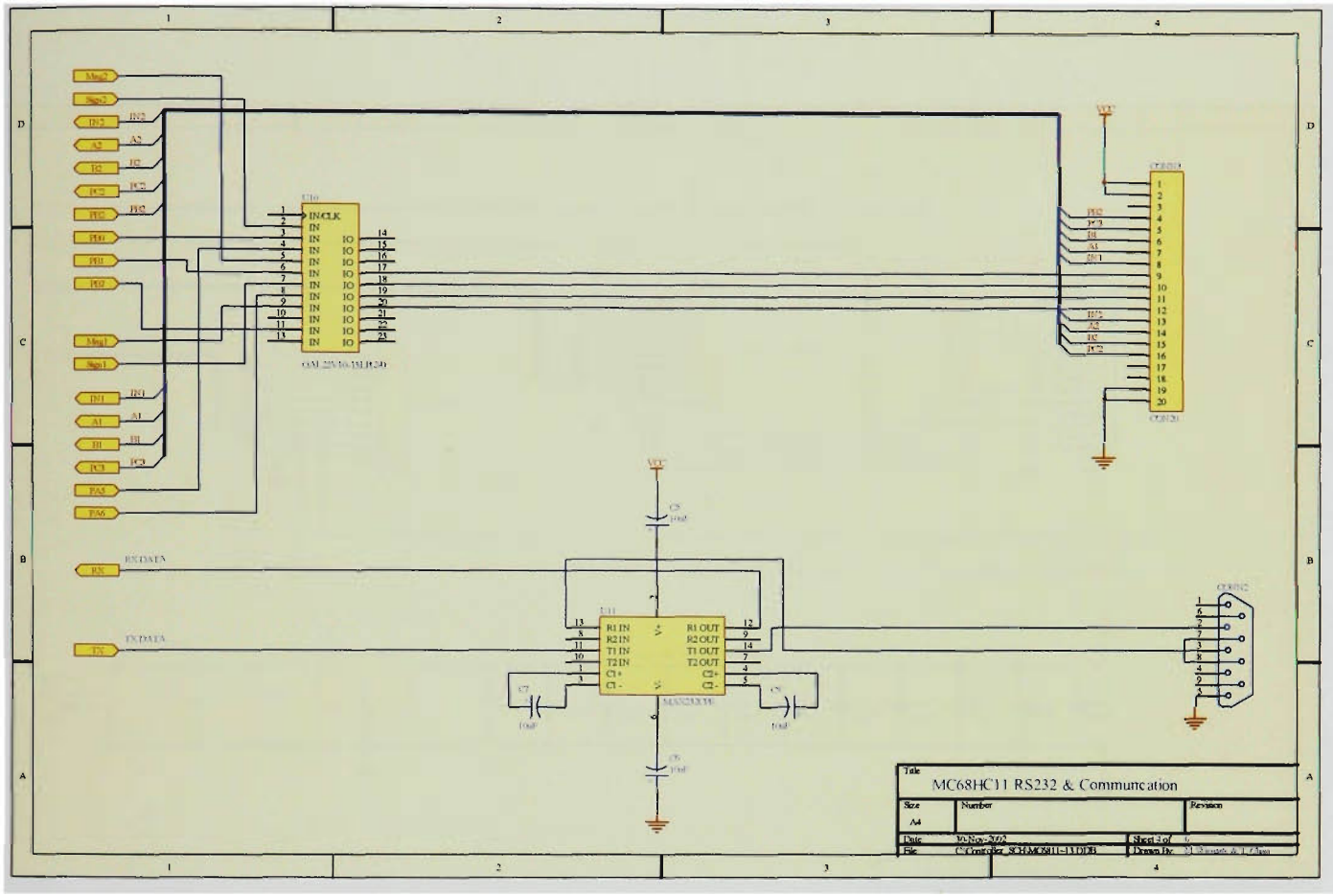



**Fig B.2.0. 7 RS232 Serial communication interface circuit**

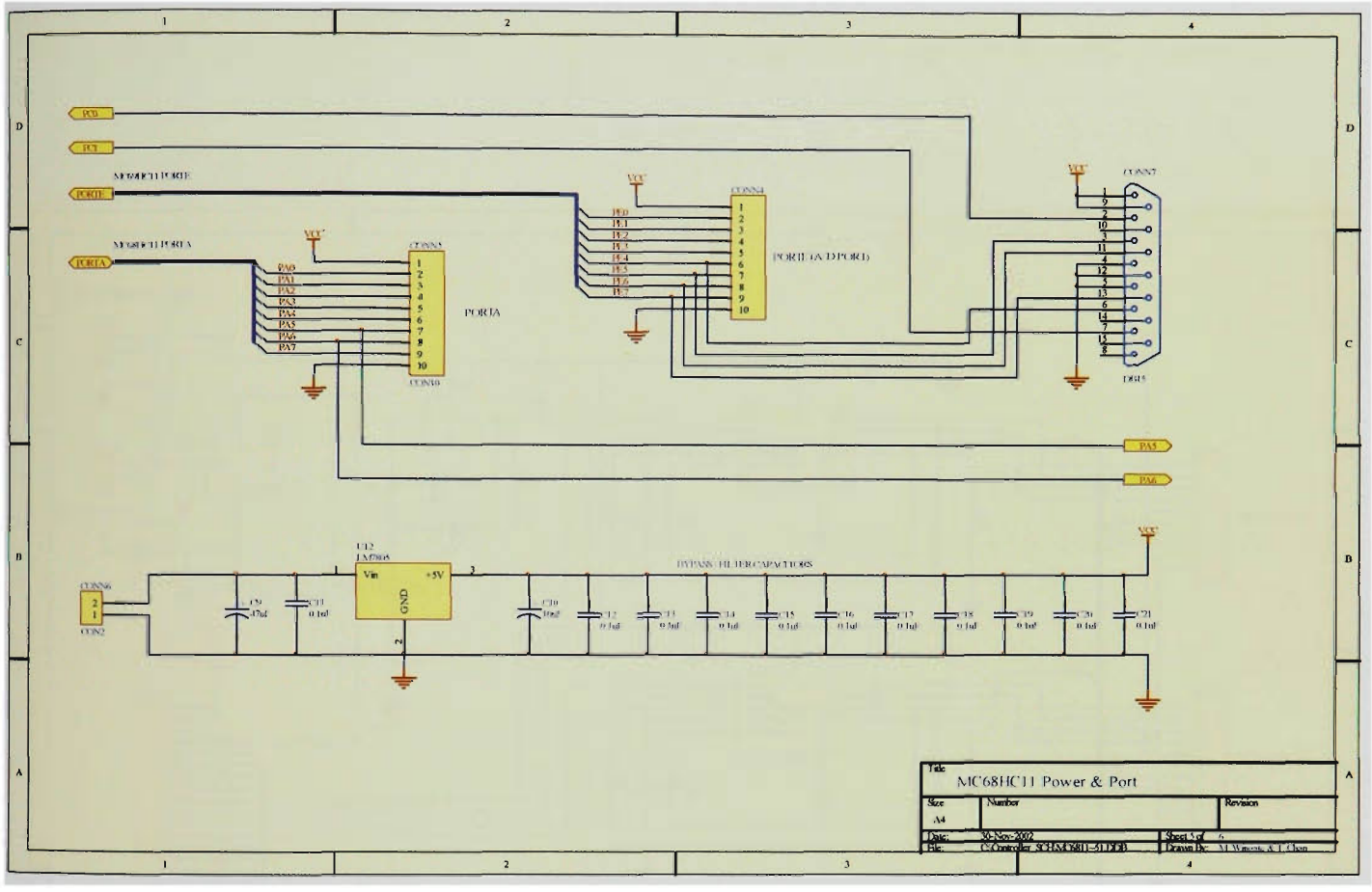



**Fig B.2.0. 8 Power and ports circiut**

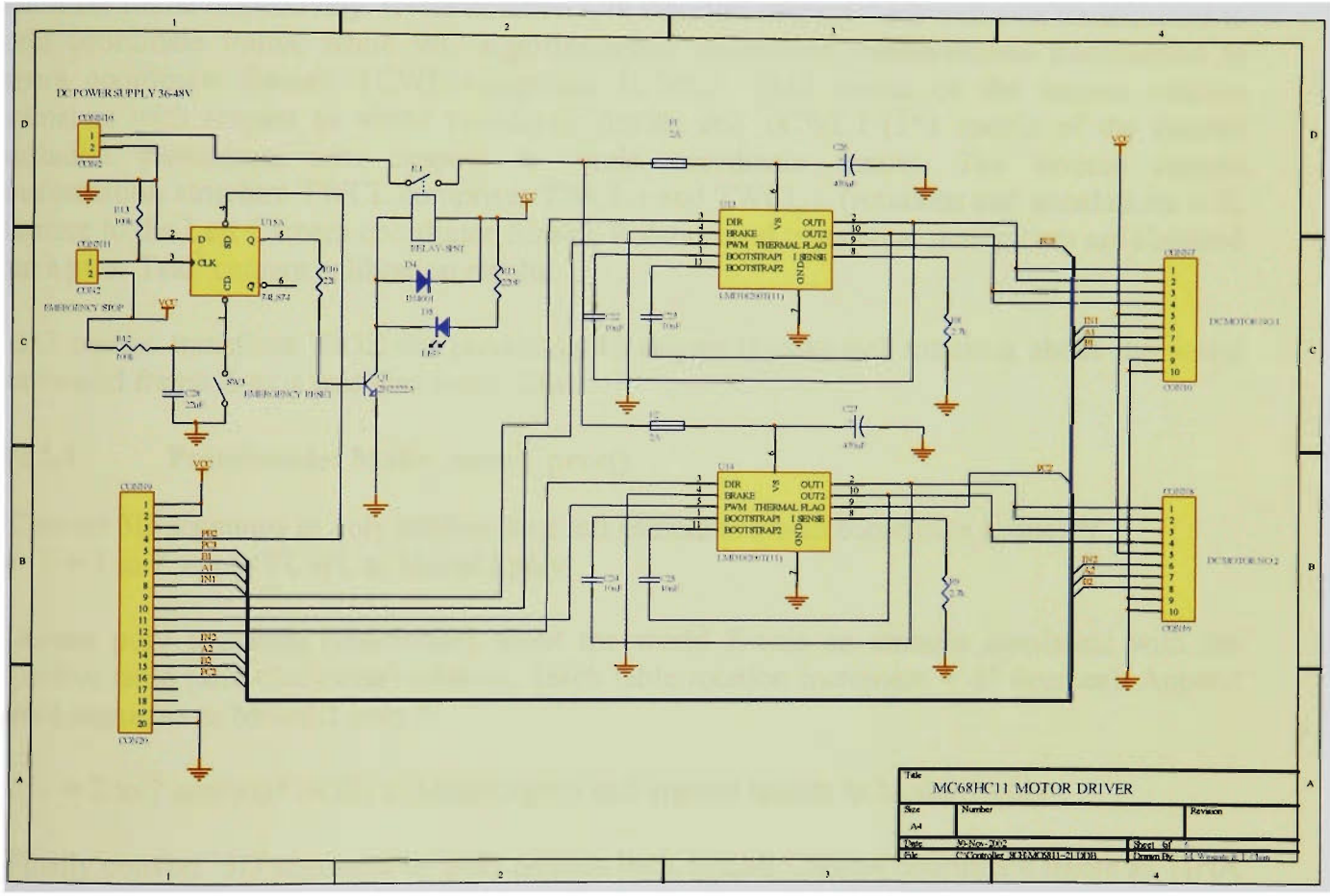



**Fig B.2.0. 9 Motor driver circuit**

## *B 2.1 Model Creating Procedure Details*

The procedure for creating a model is as follows:

3D edge models of each rotated view are automatically provided via the stereo head, computerised table, and modified TINA software. These are Model.i.poly, with i incrementing from 1 to 7, respectively representing edge model views of: 0, 45, 90, 135, 180, 225, 270, 315 degree rotation, about the Z axis of the table (i.e., world coordinate frame). Each view comprise 3D line and 3D conic edge segment location descriptors referenced to the Left Camera coordinate frame. Left Camera homogeneous transforms TCWL[1] and TWCL are established to facilitate transformations of "3D segments" to either world coordinate frame or the Left camera coordinate frame respectively. (Note notation CW signifies camera measurements transformed to world coordinate frame, while WC signifies world coordinate measurements transformed to camera coordinate frame). TCWL comprises TCWL.r (3x3 matrix of the camera rotation parameters with respect to world coordinate frame) and TCWL.t (3*1 matrix of the camera translation parameters with respect to world coordinate frame). The inverse camera transformation structure TWCL comprises TWCL.r and TWCL.t (rotations and translations with reference to the Left Camera coordinate frame). Rotation and translation matrix data are obtained from a prior Tsai[5] camera calibration results.

A 3x3 matrix transform T45Z for producing 45 degree (clockwise) rotations about the world coordinated frame Z-axis was also established.

### B 2.1.1 Pseudocode: Make_model_proc()

/* Convert 3D segments in poly buffers from left camera to world coordinate frame) */
For i = 1 to 7 apply TCWL to Model.i.poly

/* Rotate poly segments (clockwise) about the world Z-axis an amount consistent with the respective table (anti-clockwise) rotation (each table rotation increment = 45 degrees). Append rotated segments to Model.1.poly */

For i = 2 to 7 apply (i*T45Z) to Model.i.poly and append results to Model.1.poly

/* Finally convert 3D segments in .poly models back to Left Camera coordinate frame as TINA TV displays camera coordinate views) */

Apply TWCL to Model.1.poly

Display resulting Model.1.poly

Note for most cases only 4 models (instead of 7) are necessary. For these cases i=1 to 4, and a T90Z = 2x T45Z is used etc.

---

[1] TCWL is the 4x4 homogeneous transform of the Left Camera w.r.t. the World Coordinate Frame.
[5] The Tsai calibration routine is provided in TINA's Calibration tool.

## B 3.0 Forward Kinematics and Inverse Kinematics of the UMI RTX 6 DOF Robot.

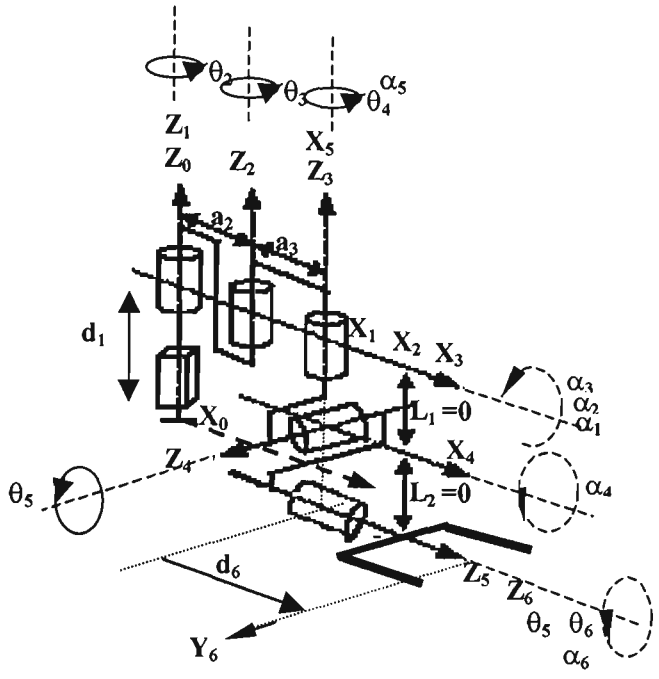



**Fig B 3.0. 1 DH link coordinate systems and joint assignment for the RTX robot arm**

## B 3.1 Denavit-Hartenberg Representation (D-H)

coordinate frames:

1. $z_{i-1}$ axis lies along the axis of motion of the $i^{th}$ joint.
2. $x_i$ axis is normal to the $z_{i-1}$ axis, and points away from it.
3. $y_i$ axis completes the right-hand coordinate system as required.

link parameters:

1. $\theta_i$ is the joint angle from the $x_{i-1}$ axis to the $x_i$ axis about the $z_{i-1}$ axis (using the right hand rule)
2. $d_i$ is the distance form the origin of the $(i-1)^{th}$ frame to the intersection of the $z_{i-1}$ axis with the $x_i$ axis along the $z_{i-1}$ axis
3. $a_i$ is the shortest distance between the $z_{i-1}$ and the $z_i$ axes (offset distance from the intersection of the $z_{i-1}$ axis with $x_i$ axis to the origin of the $i^{th}$ frame along the $x_i$ axis)
4. $\alpha_i$ is the offset angle from the $z_{i-1}$ axis to the $z_i$ axis about the $x_i$ axis (using the right hand rule)

$$^{i-1}A_i = T(z,d)R(z,\neg)T(x,a)R(x,\nabla) = \begin{vmatrix} \cos\theta_i & -\cos\alpha_i \sin\theta_i & \sin\alpha_i \sin\theta_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i \cos\theta_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

| *Links* | $d_i$ | $a_{i-1}$ | $\theta i$ | $\alpha_{i-1}$ |
|---|---|---|---|---|
| *1* | var** 0 | *0* | *0* | 0 |
| *2* | 0 | 254mm | var* | 0 |
| *3* | 0 | 254mm | var* | 90° |
| *4* | 0 | 0 | var* | 90° |
| *5* | 0 | 0 | var* | 0 |
| *6* | 177mm | 0 | var* | 0 |

Note var = variable, ** signify a prismatic joint while * a rotary joint.

Initial variable joint values are:
$d_1$=922mm, $\theta_2 = \theta_3 = \theta_4 = 0°$, $\theta_5 = 90°$, $\theta_6$=0

**Fig B 3.1. 2 Denevit Hartenberg link/joint parameter table**

## The A Matricies

```
        | Cos[q1]  -Sin[q1]  0  0    |      *note q is used in place of θ
        | Sin[q1]  Cos[q1]   0  0    |
A[1]=   | 0        0         1  d1   |
        | 0        0         0  1    |

        | Cos[q2]  -Sin[q2]  0  254 Cos[q2]   |
        | Sin[q2]  Cos[q2]   0  254 Sin[q2]   |
A[2]=   | 0        0         1  0             |
        | 0        0         0  1             |

        | Cos[q3]  -Sin[q3]  0  254 Cos[q3]   |
        | Sin[q3]  Cos[q3]   0  254 Sin[q3]   |
A[3]=   | 0        0         1  0             |
        | 0        0         0  1             |

        | Cos[q4]  0  Sin[q4]   0   |
        | Sin[q4]  0  -Cos[q4]  0   |
A[4]=   | 0        1  0         0   |
        | 0        0  0         1   |

        | Cos[q5]  0  Sin[q5]   0   |
        | Sin[q5]  0  -Cos[q5]  0   |
```

```
A[5]=  | 0        1  0        0   |
       | 0        0  0        1   |

       | Cos[q6]  -Sin[q6]  0  0    |
       | Sin[q6]  Cos[q6]   0  0    |
A[6]=  | 0        0         1  177  |
       | 0        0         0  1    |
```

## B 3.2 Forward Kinematic Solution ${}^{0}_{6}T$

The Forward Kinematic Solution ${}^{0}_{6}T$ is obtained from:

$${}^{0}_{6}T={}^{0}_{1}A * A^{*2}_{3}A^{*3}_{4}A^{*4}_{5}A^{*5}_{6}A = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Because of its size ${}^{0}_{6}T$=T06 is printed one Row at a time, note also S1=Sin($q_1$), C1=Cos($q_1$), etc.

${}^{0}_{6}T$:

```
T06[1,1] = -(S1 (S2 (-(S3 (C5 C6 S4 - C4 S6)) + C3 (C4 C5 C6 + S4 S6)) +
>         C2 (C3 (C5 C6 S4 - C4 S6) + S3 (C4 C5 C6 + S4 S6)))) +
>    C1 (C2 (-(S3 (C5 C6 S4 - C4 S6)) + C3 (C4 C5 C6 + S4 S6)) -
>       S2 (C3 (C5 C6 S4 - C4 S6) + S3 (C4 C5 C6 + S4 S6)))
T06[2,1] = C1 (S2 (-(S3 (C5 C6 S4 - C4 S6)) + C3 (C4 C5 C6 + S4 S6)) +
>       C2 (C3 (C5 C6 S4 - C4 S6) + S3 (C4 C5 C6 + S4 S6))) +
>    S1 (C2 (-(S3 (C5 C6 S4 - C4 S6)) + C3 (C4 C5 C6 + S4 S6)) -
>       S2 (C3 (C5 C6 S4 - C4 S6) + S3 (C4 C5 C6 + S4 S6)))
T06[3,1] = C6 S5
T06[4,1] = 0
T06[1,2] = C1 (-(S2 (S3 (C6 S4 - C4 C5 S6) + C3 (-(C4 C6) - C5 S4 S6))) +
>       C2 (C3 (C6 S4 - C4 C5 S6) - S3 (-(C4 C6) - C5 S4 S6))) -
>    S1 (C2 (S3 (C6 S4 - C4 C5 S6) + C3 (-(C4 C6) - C5 S4 S6)) +
>       S2 (C3 (C6 S4 - C4 C5 S6) - S3 (-(C4 C6) - C5 S4 S6)))
T06[2,2] = S1 (-(S2 (S3 (C6 S4 - C4 C5 S6) + C3 (-(C4 C6) - C5 S4 S6))) +
```

```
>       C2 (C3 (C6 S4 - C4 C5 S6) - S3 (-(C4 C6) - C5 S4 S6)) +
>    C1 (C2 (S3 (C6 S4 - C4 C5 S6) + C3 (-(C4 C6) - C5 S4 S6)) +
>       S2 (C3 (C6 S4 - C4 C5 S6) - S3 (-(C4 C6) - C5 S4 S6)))
T06[3,2] = -(S5 S6)
T06[4,2] = 0
T06[1,3] = Cos[q1 + q2 + q3 + q4] S5
T06[2,3] = Sin[q1 + q2 + q3 + q4] S5
T06[3,3] = -C5
T06[4,3] = 0
T06[1,4] = -(S1 (254 S2 + S2 (254 C3 + 177 C3 C4 S5 - 177 S3 S4 S5) +
>         C2 (254 S3 + 177 S34 S5))) +
>    C1 (254 C2 + C2 (254 C3 + 177 C3 C4 S5 - 177 S3 S4 S5) -
>       S2 (254 S3 + 177 S34 S5))
T06[2,4] = C1 (254 S2 + S2 (254 C3 + 177 C3 C4 S5 - 177 S3 S4 S5) +
>       C2 (254 S3 + 177 S34 S5)) +
>    S1 (254 C2 + C2 (254 C3 + 177 C3 C4 S5 - 177 S3 S4 S5) -
>       S2 (254 S3 + 177 S34 S5))
T06[3,4] = d1 - 177 C5
T06[4,4] = 1
```

## B 3.3 Inverse Kinematics

The above transforms may be used to obtain the inverse kinematic solution for the robot arm. Alternatively the inverse kinematic solution may be obtained by considering the joint movement geometry directly. This later geometrical approach was taken and the resulting function (coded in C) is given in Section 3.3.1 below. A more general solution (suitable for arms with 5 or more degrees of freedom) was also adopted. This extends the use of the simulator to include arms with a variety of configurations and joints (i.e., Articulated, Spherical, Cylindrical joints). This second solution uses the pseudo-inverse of the manipulator's Jacobian and is completely general. Recall the forward kinematic solution is:

$${}^{0}_{n}\mathbf{T}={}^{0}_{1}\mathbf{A}*{}^{1}_{2}\mathbf{A}*{}^{2}_{3}\mathbf{A}*{}^{3}_{4}\mathbf{A}*{}^{4}_{5}\mathbf{A}*{}^{5}_{6}\mathbf{A}..{}^{n-1}_{n}\mathbf{A} = K(\mathbf{q}),$$

The inverse kinematic solution is $(\mathbf{q}) = K^{-1}(\mathbf{T})$ where $(\mathbf{q})$ is a vector array of the variable joint parameters.

Because the solution is obtained iteratively, it is less efficient than specific inverse kinematic solutions derived symbolically or from direct joint geometry. A further drawback is that while this approach allows a solution to be obtained at a singularity, the joint angles within the null space are arbitrarily assigned.

The code for the pseudo-inverse was ported from Peter Corke's Matlab "Robotics Toolbox"[1] to 'C' code.

[1] publicly available from www.brb.dmt.csiro.au/dmt/programs/autom/pic/matlab.html

### B 3.3.1 'C' written function to compute inverse kinematic solution for RTX robot arm

(Note portions of this code were ported from pascal code generously provided by Geoff West of Curtain University, W.A. Australia.)

```
int compute_xyz(double *x_coord, double *y_coord,double  *z_coord,double *pitch,double *roll,double *yaw,\\
int *s_ec, int *e_ec,int  *z_ec,int *w1_ec,int *w2_ec,int *wy_ec)
{
double   x_wrist,y_wrist,z_wrist;
double   theta,phi,radius;
double   opp;
int   flag;
double   r_xy;

/* compute roll, pitch and yaw motor counts */
/* note -  pitch - rotate about -Z (was x)
      roll  - rotate about X (was y)
      yaw   - rotate about Y  (was z)
w.r.t. wrist.


w1_ec -- wrist motor1 encoder counts
w2_ec --wrist motor2 encoder counts
s_ec --- shoulder motor "    "
e_ec --  elbow  motor  "    " etc.
 */
*w1_ec= (int)(-*pitch/0.07415);
*w2_ec=*w1_ec;

*w1_ec=*w1_ec+ (int)(*roll/(0.07415));
*w2_ec=*w2_ec- (int)(*roll/(0.07415));
/*
compute position of the wrist from the position of the gripper,
this depends on the pitch,roll and yaw angles
 */
r_xy=177 * cos(rads(*pitch));
x_wrist=*x_coord - (r_xy * sin(rads(*yaw)));
y_wrist=*y_coord - (r_xy * cos(rads(*yaw)));
z_wrist=*z_coord + (177 * sin(rads(*pitch)));

/*
note default configuration is LH
 */
flag=0;
*z_ec= (int)((z_wrist-z_orig)/0.2667);
if ((*z_ec > 0) || (*z_ec < -3554))
  flag=1;
radius=sqrt((x_wrist*x_wrist)+(y_wrist*y_wrist));
if (radius > 507)
  {
  print_to_screen("ERROR attempt to place wrist outside the limits! radius=%f\n",radius);
```

```
  print_to_screen("will jump out of program");
  }
else
  {
  opp=sqrt((arm_1*arm_1)-((radius/2)*(radius/2)));
  if(radius < 0.0001)
    theta=90;
  else
    theta=atan(opp*2/radius)*360.0/(2*pi);


  *s_ec= (int)(theta/0.03422);
  *e_ec= (int)((-theta/0.06844)*2);
  if ((y_wrist < 0.0001) && (x_wrist > 0))
    phi=90;
  else if ((y_wrist < 0.0001) && (x_wrist <= 0))
    phi=-90;
  else
    phi=atan(x_wrist/y_wrist)*360/(pi*2);

    print_to_screen( "phi= %f \n",phi);
  *s_ec=*s_ec- (int)(phi/0.03422);
/*
compute required yaw angle w.r.t position determined by
the angle of the arm with the y axis
 */
  *wy_ec=- (int)((*yaw - phi)/0.10267);


/*
if limits exceeded, try RH config
 */
  if((*s_ec < -2630) || (*s_ec > 2630) || (*e_ec < -2630) || (*e_ec > 2206))
    {
    *s_ec=- (int)(theta/0.0342);
    *e_ec=- (int)((-theta/0.06844)*2);
    *s_ec=*s_ec- (int)(phi/0.03422);
    };
  if((*s_ec < -2630) || (*s_ec > 2630) || (*e_ec < -2630) || (*e_ec > 2206)) flag=1;
  if (flag == 1 )
    {
    print_to_screen("WARNING- arm will exceed limits if you continue \n");
    print_to_screen("z_ec=%d , s_ec=%d ,e_ec=%d  \n",*z_ec,*s_ec, *e_ec);
    print_to_screen("hit <cr> to continue\n");

    }

  }

return 0;
}
```

# Appendix C Development of a Feature Tracking Strategy

## C 1.0 Introduction

This appendix builds on the feature tracking analysis commenced in Chapter 5, developing the methodology and strategies for continuous featuring tracking while at the same time maintaining a manageable sized feature-map. The objective is to maximise the capabilities of the TRC stereo head to make measurements, and provide navigational information.

Davison [Dav98] develops a means of maintaining a large map of landmark features without the computational burden of having to update the full covariance matrices following a measurement. Moreover the solution permits continuous tracking without compromising the integrity of the filter, and during motion, requires only those parts of the filter that are needed for the current tracking task to be updated.

Davison's derivations are adopted, making the necessary modifications to accommodate the differences in MROLR's control architecture and stereo head outlined in Section 4.2. Building on the expressions (equations 5.1 through 5.52) developed in Chapter 5, the methodology that necessitates only those parts of the state vector and covariance matrix which are directly involved in the tracking process at that time to be updated is outlined. These are the estimated states of the sensor and observed $i^{th}$ feature, $\hat{\mathbf{x}}_v$ *and* $\hat{\mathbf{y}}_i$ respectively, and the covariances $\mathbf{P}_{xx}, \mathbf{P}_{xy_i}$, and $\mathbf{P}_{y_i y_j}$. Recall that $\mathbf{P}_{xy_i}$ is the covariance matrix between the estimated mobile robot state $\hat{\mathbf{x}}_v$ and feature $\hat{\mathbf{y}}_i$. By storing a small amount of information at each transition step, the state and covariances can be updated in a generic way at the completion of each tracking motion.

To emphasis the equations developed in Chapter 5 are being extended, the same numbering sequence is used.

For MROLR the system state transition function $\mathbf{f}$ (Equation 5.53) comprises of function $\mathbf{f}_v$ which models the head sensor movements in terms of the mobile-base state $\mathbf{x}_v$ and the control vector $\mathbf{u}$, and $\mathbf{y}_i$ the scene-feature states.

The system analysis can be simplified by imposing the following restrictions:

- during the platform's motion the position of the scene-features (relative to the world coordinate frame) do not change, and consequently $\mathbf{y}_i$ remains constant during a state transition, and
- a scene-feature's measurement depends only the current states of the sensor and the feature undergoing measurement (equation 5.54).

With the above restrictions the state transition has the form

$$\mathbf{f}(\mathbf{x}) = \begin{pmatrix} \mathbf{f}_v(\mathbf{x}_v, \mathbf{u}) \\ \mathbf{y}_1 \\ \mathbf{y}_2 \\ \vdots \end{pmatrix} \qquad 5.53$$

and a measurement has the form

$$\mathbf{m}(\mathbf{x}) = \mathbf{m}_i(\mathbf{x}_v, \mathbf{y}_i) \qquad 5.54$$

### C 5.5.9 System State Prediction

The prediction of the state following a motion and the corresponding covariance is readily written down from consideration of equations 5.32 and 5.33. In the following the "k" time-step notation will be omitted.

$$\hat{\mathbf{x}}_{new} = \begin{pmatrix} \mathbf{f}_v(\hat{\mathbf{x}}_v, \mathbf{u}) \\ \hat{\mathbf{y}}_1 \\ \hat{\mathbf{y}}_2 \\ \vdots \end{pmatrix} \qquad 5.55$$

$$\mathbf{P}_{new} = \begin{bmatrix} \nabla(\mathbf{f}_v)_{\mathbf{x}_v} \mathbf{P}_{xx} \nabla(\mathbf{f}_v)^T_{\mathbf{x}_v} + \mathbf{Q} & \nabla(\mathbf{f}_v)_{\mathbf{x}_v} \mathbf{P}_{xy1} & \nabla(\mathbf{f}_v)_{\mathbf{x}_v} \mathbf{P}_{xy2} & \cdots \\ \mathbf{P}_{y1x} \nabla(\mathbf{f}_v)^T_{\mathbf{x}_v} & \mathbf{P}_{y1y1} & \mathbf{P}_{y1y2} & \cdots \\ \mathbf{P}_{y2x} \nabla(\mathbf{f}_v)^T_{\mathbf{x}_v} & \mathbf{P}_{y2y1} & \mathbf{P}_{y2y2} & \cdots \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \end{bmatrix} \qquad 5.56$$

### C 5.5.10 Filter Update

Following measurement $\mathbf{z}_i$ of scene-feature $i$ and recalling that measurement vector $\mathbf{m}_i$ is a function of $\mathbf{x}_v$ and $\mathbf{y}_i$ only, the following equations may be derived:

$$\nabla(\mathbf{m}_i)_{\mathbf{x}} = \begin{pmatrix} \nabla(\mathbf{m}_i)_{\mathbf{x}_v} & \nabla(\mathbf{m}_i)_{\mathbf{y}_1} & \nabla(\mathbf{m}_i)_{\mathbf{y}_2} & \cdots & \nabla(\mathbf{m}_i)_{\mathbf{y}_i} & \cdots \end{pmatrix}$$

$$= \begin{pmatrix} \nabla(\mathbf{m}_i)_{\mathbf{x}_v} & \mathbf{0} & \cdots & & \cdots & \mathbf{0} & \nabla(\mathbf{m}_i)_{\mathbf{y}_i} & \cdots \end{pmatrix} \qquad 5.57$$

$$\mathbf{P}\nabla(\mathbf{m}_i)_{\mathbf{x}}^T = \begin{bmatrix} \mathbf{P}_{xx} & \mathbf{P}_{xy1} & \mathbf{P}_{xy2} & \cdots \\ \mathbf{P}_{y1x} & \mathbf{P}_{y1y1} & \mathbf{P}_{y1y2} & \cdots \\ \mathbf{P}_{y2x} & \mathbf{P}_{y2y1} & \mathbf{P}_{y2y2} & \cdots \\ \vdots & \vdots & \vdots & \end{bmatrix} \begin{pmatrix} \nabla(\mathbf{m}_i)_{\mathbf{x}_v}^T \\ \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \nabla(\mathbf{m}_i)_{\mathbf{y}_i}^T \\ \vdots \end{pmatrix}$$

or

$$\mathbf{P}\nabla(\mathbf{m}_i)_{\mathbf{x}}^T = \begin{pmatrix} \mathbf{P}_{xx} \\ \mathbf{P}_{y_1x} \\ \mathbf{P}_{y_2x} \\ \vdots \end{pmatrix} \nabla(\mathbf{m}_i)_{\mathbf{x}_v}^T + \begin{pmatrix} \mathbf{P}_{xy_i} \\ \mathbf{P}_{y_1x_i} \\ \mathbf{P}_{y_2y_i} \\ \vdots \end{pmatrix} \nabla(\mathbf{m}_i)_{\mathbf{y}_i}^T \qquad \textbf{5. 58}$$

And the innovation covariance **S** is:

$$\begin{aligned} \mathbf{S} &= \nabla(\mathbf{m}_i)_{\mathbf{x}} \mathbf{P} \nabla(\mathbf{m}_i)_{\mathbf{x}}^T + \mathbf{R} \\ &= \nabla(\mathbf{m}_i)_{\mathbf{x}_v} \mathbf{P}_{xx} \nabla(\mathbf{m}_i)_{\mathbf{x}_v}^T + \nabla(\mathbf{m}_i)_{\mathbf{y}_i} \mathbf{P}_{y_1x} \nabla(\mathbf{m}_i)_{\mathbf{x}_v}^T + \nabla(\mathbf{m}_i)_{\mathbf{x}_v} \mathbf{P}_{xy_i} \nabla(\mathbf{m}_i)_{\mathbf{y}_i}^T \\ &\quad + \nabla(\mathbf{m}_i)_{\mathbf{y}_i} \mathbf{P}_{y_1y_1} \nabla(\mathbf{m}_i)_{\mathbf{y}_i}^T + \mathbf{R} \end{aligned} \qquad \textbf{5. 59}$$

Recall that in equation 5.59 **R** is the covariance matrix of the measurement noise.

From equation 5.27 the Kalman gain **W** is:

$$\mathbf{W} = \mathbf{P}\nabla(\mathbf{m})_{\mathbf{x}}^T \mathbf{S}^{-1} = \begin{pmatrix} \mathbf{P}_{xx} \\ \mathbf{P}_{y_1x} \\ \mathbf{P}_{y_2x} \\ \vdots \end{pmatrix} \nabla(\mathbf{m}_i)_{\mathbf{x}_v}^T \mathbf{S}^{-1} + \begin{pmatrix} \mathbf{P}_{xy_i} \\ \mathbf{P}_{y_1y_i} \\ \mathbf{P}_{y_2y_i} \\ \vdots \end{pmatrix} \nabla(\mathbf{m}_i)_{\mathbf{y}_i}^T \mathbf{S}^{-1} \qquad \textbf{5. 60}$$

The $\mathbf{WSW}^T$ required to update **P** following a measurement is:

$$\mathbf{WSW}^T = (\mathbf{P}\nabla(\mathbf{m})_{\mathbf{x}}^T \mathbf{S}^{-1})\mathbf{S}(\mathbf{P}\nabla(\mathbf{m})_{\mathbf{x}}^T \mathbf{S}^{-1})^T = \mathbf{P}\nabla(\mathbf{m})_{\mathbf{x}}{}^T \mathbf{S}^{-1}\mathbf{S}\mathbf{S}^{-1}{}^T \nabla(\mathbf{m})_{\mathbf{x}} \mathbf{P}^T$$
$$= \mathbf{P}\nabla(\mathbf{m})_{\mathbf{x}}^T \mathbf{S}^{-1}{}^T \nabla(\mathbf{m})_{\mathbf{x}} \mathbf{P}^T \qquad \textbf{5. 61}$$

Substituting for $\mathbf{P}\nabla(\mathbf{m})_{\mathbf{x}}^T$ from equation 5.58 gives equation 5.62.

In equation 5.62 recall **S** is symmetric $\mathbf{S}^{-1} = \mathbf{S}^{-T}$

Further details on updating components of the Estimated State Vector and Covariance Matrix are given below. The derivations assume that the feature observed in the measurement stage has label i, and there are also two unobserved features j and k.

$$\mathbf{WSW}^T = \begin{pmatrix} \mathbf{P}_{xx} \\ \mathbf{P}_{y_1 x} \\ \mathbf{P}_{y_2 x} \\ \vdots \end{pmatrix} \nabla(\mathbf{m}_i)_{\mathbf{x}_v}^T \mathbf{S}^{-1} \nabla(\mathbf{m}_i)_{\mathbf{x}_v} (\mathbf{P}_{xx} \quad \mathbf{P}_{x\,y_1} \quad \mathbf{P}_{xy_2} \quad \cdots)$$

$$+ \begin{pmatrix} \mathbf{P}_{xx} \\ \mathbf{P}_{y_1 x} \\ \mathbf{P}_{y_2 x} \\ \vdots \end{pmatrix} \nabla(\mathbf{m}_i)_{\mathbf{x}_v}^T \mathbf{S}^{-1} \nabla(\mathbf{m}_i)_{\mathbf{y}_i} (\mathbf{P}_{y_i x} \quad \mathbf{P}_{y_i y_1} \quad \mathbf{P}_{y_i y_2} \quad \cdots)$$

$$+ \begin{pmatrix} \mathbf{P}_{xy_i} \\ \mathbf{P}_{y_1 y_i} \\ \mathbf{P}_{y_2 y_i} \\ \vdots \end{pmatrix} \nabla(\mathbf{m}_i)_{\mathbf{y}_i}^T \mathbf{S}^{-1} \nabla(\mathbf{m}_i)_{\mathbf{x}_v} (\mathbf{P}_{xx} \quad \mathbf{P}_{xy_1} \quad \mathbf{P}_{xy_2} \quad \cdots)$$

$$+ \begin{pmatrix} \mathbf{P}_{xy_i} \\ \mathbf{P}_{y_1 y_i} \\ \mathbf{P}_{y_2 y_i} \\ \vdots \end{pmatrix} \nabla(\mathbf{m}_i)_{\mathbf{y}_i}^T \mathbf{S}^{-1} \nabla(\mathbf{m}_i)_{\mathbf{y}_i} (\mathbf{P}_{y_i x} \quad \mathbf{P}_{y_i y_1} \quad \mathbf{P}_{y_i y_2} \quad \cdots) \qquad \textbf{5. 62}$$

Commencing with the Estimated Sensor State.

**The Prediction is:**

$$\hat{\mathbf{x}}_{v\,(new)} = \mathbf{f}_v(\hat{\mathbf{x}}_v, \mathbf{u}) \qquad 5.63$$

and **Update**

$$\hat{\mathbf{x}}_{v\,(new)} = \hat{\mathbf{x}}_v + \mathbf{P}_{xx}\nabla(\mathbf{m}_i)_{\mathbf{x}_v}^T \mathbf{S}^{-1}v + \mathbf{P}_{xy_i}\nabla(\mathbf{m}_i)_{\mathbf{y}_i}^T \mathbf{S}^{-1}v \qquad 5.64$$

The Observed Feature State Prediction is:

$$\hat{\mathbf{y}}_{i\,(new)} = \hat{\mathbf{y}}_i \qquad 5.65$$

**Update**

$$\hat{\mathbf{y}}_{i\,(new)} = \hat{\mathbf{y}}_i + \mathbf{P}_{y_i x}\nabla(\mathbf{m}_i)_{\mathbf{x}_v}^T \mathbf{S}^{-1}v + \mathbf{P}_{y_i y_i}\nabla(\mathbf{m}_i)_{\mathbf{x}_v}^T \mathbf{S}^{-1}v \qquad 5.66$$

For the Unobserved Feature State, the Prediction is:

$$\hat{\mathbf{y}}_{j\,(new)} = \hat{\mathbf{y}}_j \qquad 5.67$$

While the Update:

$$\hat{\mathbf{y}}_{j(new)} = \hat{\mathbf{y}}_j + \mathbf{P}_{y_j x}\nabla(\mathbf{m}_i)_{\mathbf{x}_v}^T \mathbf{S}^{-1}v + \mathbf{P}_{y_j y_i}\nabla(\mathbf{m}_i)_{\mathbf{y}_i}^T \mathbf{S}^{-1}v \qquad 5.68$$

Next the covariance matrix **P** will be considered. This matrix is partitioned with covariance components comprising: (a) the covariance between the sensor state and itself, (b) the sensor state and the observed feature state, (c) the observed feature state and itself, (d) the sensor state and the unobserved feature state, (e) the observed feature state and an unobserved feature state, and (f) two unobserved feature states.

Because of the considerable size of matrix P it is simpler to define component sub matrices as follows:

$$\mathbf{A} = \nabla(\mathbf{m}_i)_{\mathbf{x}_v}^T \mathbf{S}^{-1}\nabla(\mathbf{m}_i)_{\mathbf{x}_v} \qquad 5.69$$

$$\mathbf{B} = \nabla(\mathbf{m}_i)_{\mathbf{x}_v}^T \mathbf{S}^{-1}\nabla(\mathbf{m}_i)_{\mathbf{y}_i} \qquad 5.70$$

$$\mathbf{C} = \nabla(\mathbf{m}_i)_{y_i}^T \mathbf{S}^{-1}\nabla(\mathbf{m}_i)_{\mathbf{x}_v} \qquad 5.71$$

$$\mathbf{D} = \nabla(\mathbf{m}_i)_{y_i}^T \mathbf{S}^{-1}\nabla(\mathbf{m}_i)_{\mathbf{y}_i} \qquad 5.72$$

Now for each of the partitoned covariance states;

### C 5.5.11 Sensor State and Itself

**Prediction**

$$\mathbf{P}_{xx(new)} = \nabla(\mathbf{f}_v)_{x_v} \mathbf{P}_{xx} \nabla(\mathbf{f}_v)^T_{x_v} + \mathbf{Q} \qquad \text{5. 73}$$

**Update**

$$\mathbf{P}_{xx(new)} = \mathbf{P}_{xx} - (\mathbf{P}_{xx}\mathbf{A}\,\mathbf{P}_{xx} + \mathbf{P}_{xx}\mathbf{B}\,\mathbf{P}_{y_i x} + \mathbf{P}_{x\,y_i}\mathbf{C}\,\mathbf{P}_{xx} + \mathbf{P}_{x\,y_i}\mathbf{D}\,\mathbf{P}_{y_i x}) \qquad \text{5. 74}$$

### C 5.5.12 Sensor State and Observed Feature State

**Prediction**

$$\mathbf{P}_{xy_i(new)} = \nabla(\mathbf{f}_v)_{x_v} \mathbf{P}_{xy_i} \qquad \text{5. 75}$$

**Update**

$$\mathbf{P}_{xy_i(new)} = \mathbf{P}_{xy_i} - (\mathbf{P}_{xx}\mathbf{A}\,\mathbf{P}_{xy_i} + \mathbf{P}_{xx}\mathbf{B}\,\mathbf{P}_{y_i y_i} + \mathbf{P}_{xy_i}\,\mathbf{C}\,\mathbf{P}_{xy_i} + \mathbf{P}_{x\,y_i}\mathbf{D}\,\mathbf{P}_{y_i y_i}) \qquad \text{5. 76}$$

### C 5.5.13 Observed Feature State and Itself

**Prediction**

$$\mathbf{P}_{y_i y_i(new)} = \mathbf{P}_{y_i y_i} \qquad \text{5. 77}$$

**Update**

$$\mathbf{P}_{y_i y_i(new)} = \mathbf{P}_{y_i y_i} - (\mathbf{P}_{y_i x}\mathbf{A}\,\mathbf{P}_{xy_i} + \mathbf{P}_{y_i x}\mathbf{B}\,\mathbf{P}_{y_i y_i} + \mathbf{P}_{y_i y_i}\,\mathbf{C}\,\mathbf{P}_{xy_i} + \mathbf{P}_{y_i y_i}\mathbf{D}\,\mathbf{P}_{y_i y_i}) \qquad \text{5. 78}$$

### C 5.5.14 Sensor State and an Unobserved Feature State

**Prediction**

$$\mathbf{P}_{xy_j(new)} = (\mathbf{f}_v)_{x_v} \mathbf{P}_{xy_j} \qquad \text{5. 79}$$

**Update**

$$\mathbf{P}_{xy_j(new)} = \mathbf{P}_{xy_j} - (\mathbf{P}_{xx}\mathbf{A}\,\mathbf{P}_{xy_j} + \mathbf{P}_{xx}\mathbf{B}\,\mathbf{P}_{y_i y_j} + \mathbf{P}_{xy_i}\mathbf{C}\,\mathbf{P}_{xy_j} + \mathbf{P}_{x\,y_j}\mathbf{D}\,\mathbf{P}_{y_i y_j}) \qquad \text{5. 80}$$

### C 5.5.15 Observed Feature State and an Unobserved Feature State

**Prediction**

$$\mathbf{P}_{y_i y_j(new)} = \mathbf{P}_{y_i y_j} \qquad \textbf{5. 81}$$

**Update**

$$\mathbf{P}_{y_i y_j(new)} = \mathbf{P}_{y_i y_j} - (\mathbf{P}_{y_i x}\mathbf{A}\mathbf{P}_{xy_j} + \mathbf{P}_{y_i x}\mathbf{B}\mathbf{P}_{y_i y_j} + \mathbf{P}_{y_i y_i}\mathbf{C}\mathbf{P}_{xy_j} + \mathbf{P}_{y_i y_i}\mathbf{D}\mathbf{P}_{y_i y_j}) \qquad \textbf{5. 82}$$

### C 5.5.16 And finally Two Unobserved Feature States

**Prediction**

$$\mathbf{P}_{y_j y_k(new)} = \mathbf{P}_{y_j y_k} \qquad \textbf{5. 83}$$

**Update**

$$\mathbf{P}_{y_j y_k(new)} = \mathbf{P}_{y_j y_k} - (\mathbf{P}_{y_j x}\mathbf{A}\,\mathbf{P}_{xy_k} + \mathbf{P}_{y_j x}\mathbf{B}\,\mathbf{P}_{y_i y_k} + \mathbf{P}_{y_j y_i}\mathbf{C}\,\mathbf{P}_{xy_k} + \mathbf{P}_{y_j y_i}\mathbf{D}\,\mathbf{P}_{y_i y_k}) \qquad \textbf{5. 84}$$

Equation 5.84 also applies for determination of the covariance between an unobserved feature state and itself (ie for j = k)

## *C 5.6 Multiple Steps*

The development of the above equations permits the efficient filtering for the specific case of tracking a single feature over multiple incremental steps. Assume for the moment that the robot is at the start position and is able to observe feature i:

$$\hat{\mathbf{x}}(0) = \begin{pmatrix} \hat{\mathbf{x}}_v(0) \\ \hat{\mathbf{y}}_1(0) \\ \hat{\mathbf{y}}_2(0) \\ \vdots \end{pmatrix}, \quad \mathbf{P}(0) = \begin{bmatrix} \mathbf{P}_{xx}(0) & \mathbf{P}_{xy_1}(0) & \mathbf{P}_{xy_1}(0) \cdots \\ \mathbf{P}_{y_1 x}(0) & \mathbf{P}_{y_1 y_1}(0) & \mathbf{P}_{y_1 y_1}(0) \cdots \\ \mathbf{P}_{y_2 x}(0) & \mathbf{P}_{y_2 y_1}(0) & \mathbf{P}_{y_2 y_2}(0) \cdots \\ \vdots & \vdots & \vdots \end{bmatrix} \qquad \textbf{5. 85}$$

During these multiple steps while the single feature i is being tracked, equations 5.63 to 5.66 and 5.73 to 5.78 could to be used to update the state vector and covariance matrix. However by storing a small amount of information at each filter step it is possible to reduce the computations to that of only modifying

$\hat{\mathbf{x}}$, $\hat{\mathbf{y}}_i$, $\mathbf{P}_{xx}$, $\mathbf{P}_{xy_i}$, *and* $\mathbf{P}_{y_iy_i}$ directly. Other parts need to be updated only at the completion of the single feature tracking in a systematic and generic manner.

To outline the process first consider:

$\mathbf{P}_{xy_j}$ *and* $\mathbf{P}_{y_iy_j}$

From examination of equations 5.79 to 5.82 it is evident that they may be expressed in the form

$$\mathbf{P}_{xy_j} = \mathbf{E}_T\mathbf{P}_{xy_j}(0)+\mathbf{F}_T\mathbf{P}_{y_iy_j}(0) \qquad \textbf{5. 86}$$

$$\mathbf{P}_{y_iy_j} = \mathbf{G}_T\mathbf{P}_{y_iy_j}(0)+\mathbf{H}_T\mathbf{P}_{xy_j}(0) \qquad \textbf{5. 87}$$

For a single measurement update, using Equations 5.80 and 5.82,

$$\mathbf{P}_{xy_j(new)} = \mathbf{P}_{xy_j} - \left[\mathbf{P}_{xx}\mathbf{A}+\mathbf{P}_{xy_i}\mathbf{C}\right]\mathbf{P}_{xy_j} - \left[\mathbf{P}_{xx}\mathbf{B}+\mathbf{P}_{xy_i}\mathbf{D}\right]\mathbf{P}_{y_iy_j} \qquad \textbf{5. 88}$$

$$\mathbf{P}_{y_iy_j(new)} = \mathbf{P}_{y_iy_j} - \left[\mathbf{P}_{y_ix}\mathbf{A}+\mathbf{P}_{y_iy_i}\mathbf{C}\right]\mathbf{P}_{xy_i} - \left[\mathbf{P}_{y_ix}\mathbf{B}+\mathbf{P}_{y_iy_i}\mathbf{D}\right]\mathbf{P}_{y_iy_j} \qquad \textbf{5. 89}$$

To obtain a form similar to equations 5.86 and 5.87 matrices **E, F, G** and **H** are defined as:

$$\mathbf{E} = \mathbf{I} - \left[\mathbf{P}_{xx}\mathbf{A}+\mathbf{P}_{xy_i}\mathbf{C}\right] \qquad \textbf{5. 90}$$

$$\mathbf{F} = -\left[\mathbf{P}_{xx}\mathbf{B}+\mathbf{P}_{xy_i}\mathbf{D}\right] \qquad \textbf{5. 91}$$

$$\mathbf{G} = \mathbf{I} - \left[\mathbf{P}_{y_ix}\mathbf{B}+\mathbf{P}_{y_iy_i}\mathbf{D}\right] \qquad \textbf{5. 92}$$

$$\mathbf{H} = -\left[\mathbf{P}_{y_ix}\mathbf{A}+\mathbf{P}_{y_iy_i}\mathbf{C}\right] \qquad \textbf{5. 93}$$

Equations 5.88 and 5.90 can now be written as:

$$\mathbf{P}_{xy_j(new)} = \mathbf{E}\mathbf{P}_{xy_j} + \mathbf{F}\mathbf{P}_{y_iy_j} \qquad \textbf{5. 94}$$

$$\mathbf{P}_{y_iy_j(new)} = \mathbf{G}\mathbf{P}_{y_iy_j} + \mathbf{H}\mathbf{P}_{xy_j} \qquad \textbf{5. 95}$$

which is in the desired form. Coefficient matrices $\mathbf{E}_T$, $\mathbf{F}_T$, $\mathbf{G}_T$, $\mathbf{H}_T$ of equations 5.88 and 5.89 for a measurement update become:

$$\mathbf{E}_{T(new)} = \mathbf{E}\mathbf{E}_T + \mathbf{F}\mathbf{H}_T \qquad 5.96$$

$$\mathbf{F}_{T(new)} = \mathbf{E}\mathbf{F}_T + \mathbf{F}\mathbf{G}_T \qquad 5.97$$

$$\mathbf{G}_{T(new)} = \mathbf{G}\mathbf{G}_T + \mathbf{H}\mathbf{F}_T \qquad 5.98$$

$$\mathbf{H}_{T(new)} = \mathbf{G}\mathbf{H}_T + \mathbf{H}\mathbf{E}_T \qquad 5.99$$

At a prediction (From equations 5.79 and 5.81) $\mathbf{E}_T$, $\mathbf{F}_T$ become:

$$\mathbf{E}_{T(new)} = \nabla(\mathbf{f}_v)_{xv}\mathbf{E}_T \qquad 5.100$$

$$\mathbf{F}_{T(new)} = \nabla(\mathbf{f}_v)_{xv}\mathbf{F}_T \qquad 5.101$$

$\mathbf{G}_T$, $\mathbf{H}_T$ remaining unchanged.

Consider next $\mathbf{P}_{y_j y_k}$. To express the prediction and measurement updates for $\mathbf{P}_{y_j y_k}$ (the covariance between the position estimate of two of the unobserved features) in the same form as the equations just derived, it can be deduced from equations 5.83 and 5.84 that:

$$\begin{aligned}\mathbf{P}_{y_j y_k} = \mathbf{P}_{y_j y_k}(0) - \big[&\mathbf{P}_{y_j x}(0)\mathbf{A}_T\mathbf{P}_{xy_k}(0) + \mathbf{P}_{y_j x}(0)\mathbf{B}_T\mathbf{P}_{y_i y_k}(0) \\ &+ \mathbf{P}_{y_j y_i}(0)\mathbf{C}_T\mathbf{P}_{xy_k}(0) + \mathbf{P}_{y_j y_i}(0)\mathbf{D}_T\mathbf{P}_{y_i y_k}(0)\big]\end{aligned} \qquad 5.102$$

For a single measurement update, re-expressing in the form of equation 5.84 gives,

$$\begin{aligned}\mathbf{P}_{y_j y_k(new)} = \mathbf{P}_{y_j y_k} - \big[&\mathbf{P}_{y_j x}\mathbf{A}\mathbf{P}_{xy_k} + \mathbf{P}_{y_j x}\mathbf{B}\mathbf{P}_{y_i y_k} \\ &+ \mathbf{P}_{y_j y_i}\mathbf{C}\mathbf{P}_{xy_k} + \mathbf{P}_{y_j y_i}\mathbf{D}\mathbf{P}_{y_i y_k}\big]\end{aligned} \qquad 5.103$$

substituting for $\mathbf{P}_{xy_j}$, $\mathbf{P}_{y_i y_j}$, $\mathbf{P}_{xy_k}$, *and* $\mathbf{P}_{y_i y_j}$,

$$\begin{aligned}\mathbf{P}_{y_j y_k(new)} = \mathbf{P}_{y_j y_k} &- \mathbf{P}_{y_j x}(0)\left[\mathbf{E}_T^T\mathbf{A}\mathbf{E}_T + \mathbf{E}_T^T\mathbf{B}\mathbf{H}_T + \mathbf{H}_T^T\mathbf{C}\mathbf{E}_T + \mathbf{H}_T^T\mathbf{D}\mathbf{H}_T\right]\mathbf{P}_{xy_k}(0) \\ &- \mathbf{P}_{y_j x}(0)\left[\mathbf{E}_T^T\mathbf{A}\mathbf{F}_T + \mathbf{E}_T^T\mathbf{B}\mathbf{G}_T + \mathbf{H}_T^T\mathbf{C}\mathbf{F}_T + \mathbf{H}_T^T\mathbf{D}\mathbf{G}_T\right]\mathbf{P}_{y_i y_k}(0)\end{aligned}$$

$$-\mathbf{P}_{y_j y_l}(0)\left[\mathbf{F}_\mathbf{T}^T\mathbf{A}\mathbf{E}_\mathbf{T}+\mathbf{E}_\mathbf{T}^T\mathbf{B}\mathbf{H}_\mathbf{T}+\mathbf{G}_\mathbf{T}^T\mathbf{C}\mathbf{E}_\mathbf{T}+\mathbf{G}_\mathbf{T}^T\mathbf{D}\mathbf{H}_\mathbf{T}\right]\mathbf{P}_{xy_k}(0)$$
$$-\mathbf{P}_{y_j y_l}(0)\left[\mathbf{F}_\mathbf{T}^T\mathbf{A}\mathbf{E}_\mathbf{T}+\mathbf{F}_\mathbf{T}^T\mathbf{B}\mathbf{G}_\mathbf{T}+\mathbf{G}_\mathbf{T}^T\mathbf{C}\mathbf{F}_\mathbf{T}+\mathbf{G}_\mathbf{T}^T\mathbf{D}\mathbf{G}_\mathbf{T}\right]\mathbf{P}_{y_l y_k}(0) \quad \textbf{5. 104}$$

$\mathbf{A}_\mathbf{T}$, $\mathbf{B}_\mathbf{T}$, $\mathbf{C}_\mathbf{T}$, **and** $\mathbf{D}_\mathbf{T}$ at a measurement update become:

$$\mathbf{A}_{\mathbf{T}(new)}=\mathbf{A}_\mathbf{T}+\mathbf{E}_\mathbf{T}^T\mathbf{A}\mathbf{E}_\mathbf{T}+\mathbf{E}_\mathbf{T}^T\mathbf{B}\mathbf{H}_\mathbf{T}+\mathbf{H}_\mathbf{T}^T\mathbf{C}\mathbf{E}_\mathbf{T}+\mathbf{H}_\mathbf{T}^T\mathbf{D}\mathbf{H}_\mathbf{T} \quad \textbf{5. 105}$$

$$\mathbf{B}_{\mathbf{T}(new)}=\mathbf{B}_\mathbf{T}+\mathbf{E}_\mathbf{T}^T\mathbf{A}\mathbf{F}_\mathbf{T}+\mathbf{E}_\mathbf{T}^T\mathbf{B}\mathbf{G}_\mathbf{T}+\mathbf{H}_\mathbf{T}^T\mathbf{C}\mathbf{F}_\mathbf{T}+\mathbf{H}_\mathbf{T}^T\mathbf{D}\mathbf{G}_\mathbf{T} \quad \textbf{5. 106}$$

$$\mathbf{C}_{\mathbf{T}(new)}=\mathbf{C}_\mathbf{T}+\mathbf{F}_\mathbf{T}^T\mathbf{A}\mathbf{E}_\mathbf{T}+\mathbf{F}_\mathbf{T}^T\mathbf{B}\mathbf{H}_\mathbf{T}+\mathbf{G}_\mathbf{T}^T\mathbf{C}\mathbf{E}_\mathbf{T}+\mathbf{G}_\mathbf{T}^T\mathbf{D}\mathbf{H}_\mathbf{T} \quad \textbf{5. 107}$$

$$\mathbf{D}_{\mathbf{T}(new)}=\mathbf{D}_\mathbf{T}+\mathbf{F}_\mathbf{T}^T\mathbf{A}\mathbf{F}_\mathbf{T}+\mathbf{F}_\mathbf{T}^T\mathbf{B}\mathbf{G}_\mathbf{T}+\mathbf{G}_\mathbf{T}^T\mathbf{C}\mathbf{F}_\mathbf{T}+\mathbf{G}_\mathbf{T}^T\mathbf{D}\mathbf{G}_\mathbf{T} \quad \textbf{5. 108}$$

In a prediction, equation 5.83 shows $\mathbf{P}_{y_j y_k}$ remains unchanged and therefore it is unnecessary to change $\mathbf{A}_\mathbf{T}$, $\mathbf{B}_\mathbf{T}$, $\mathbf{C}_\mathbf{T}$ and $\mathbf{D}_\mathbf{T}$ for a prediction

Consider next $\hat{\mathbf{y}}_j$. Proceeding as above the estimated state $\hat{\mathbf{y}}_j$ of an unobserved feature is:

$$\hat{\mathbf{y}}_j=\hat{\mathbf{y}}_j(0)+\mathbf{P}_{y_j x}(0)\mathbf{m}_{1\mathbf{T}}+\mathbf{P}_{y_j y_i}(0)\mathbf{n}_{1\mathbf{T}} \quad \textbf{5. 109}$$

(with $\mathbf{m}_{1\mathbf{T}}$ and $\mathbf{n}_{1\mathbf{T}}$ yet to be defined), and for a measurement update:

$$\hat{\mathbf{y}}_j(new)=\hat{\mathbf{y}}_j+\mathbf{P}_{y_j x}\nabla(\mathbf{m}_i)_{x_v}^T\,\mathbf{S}^{-1}\nu+\mathbf{P}_{y_j y_i}\nabla(\mathbf{m}_i)_{y_i}^T\,\mathbf{S}^{-1}\nu \quad \textbf{5. 110}$$

Substituting for $\mathbf{P}_{y_j x}$ and $\mathbf{P}_{y_j y_i}$ from equations 5.88 and 5.89,

$$\hat{\mathbf{y}}_{j(new)}=\hat{\mathbf{y}}_j+\mathbf{P}_{y_j x}(0)\left(\mathbf{E}_\mathbf{T}^T\nabla(\mathbf{m}_i)_{x_v}^T+\mathbf{H}_\mathbf{T}^T\nabla(\mathbf{m}_i)_{y_i}^T\right)\mathbf{S}^{-1}\nu$$
$$+\mathbf{P}_{y_j y_i}(0)\left(\mathbf{F}_\mathbf{T}^T\nabla(\mathbf{m}_i)_{x_v}{}^T+\mathbf{G}_\mathbf{T}^T\nabla(\mathbf{m}_i)_{y_i}^T\right)\mathbf{S}^{-1}\nu \quad \textbf{5. 111}$$

Thus in a measurement step:

$$\mathbf{m}_{1\mathbf{T}(new)}=\mathbf{m}_{1\mathbf{T}}+\left(\mathbf{E}_\mathbf{T}^T\nabla(\mathbf{m}_i)_{x_v}^T+\mathbf{H}_\mathbf{T}^\mathbf{T}\nabla(\mathbf{m}_i)_{y_i}^T\right)\mathbf{S}^{-1}\nu \quad \textbf{5. 112}$$

$$\mathbf{n}_{1\mathbf{T}(new)}=\mathbf{n}_{1\mathbf{T}}+\left(\mathbf{F}_\mathbf{T}^T\nabla(\mathbf{m}_i)_{x_v}^T+\mathbf{G}_\mathbf{T}^T\nabla(\mathbf{m}_i)_{y_i}^T\right)\mathbf{S}^{-1}\nu \quad \textbf{5. 113}$$

Reference to equation 5.67 shows that in a prediction step $\hat{\mathbf{y}}_j$ remains unchanged and therefore there is no need to alter $\mathbf{m}_{1T}$ and $\mathbf{n}_{1T}$.

To clarify and summarise the above procedures, consider the situation where the robot has mapped multiple features but is currently only making measurements of feature i. j and k are two unobserved features. The parts of the estimated state vector and covariance matrix that are directly involved in the motion and measurement are continuously updated. The information needed to generically update the other sections can be conveniently and efficiently stored, so that a single step, at the end of the motion, can bring them up to date.

Prior to motion the estimated state vector and covariance of the system are designated by $\hat{\mathbf{x}}(0)$ *and* $\mathbf{P}(0)$ and the following matrices and vectors are initialised to:

$$\mathbf{A}_T = 0, \mathbf{B}_T = 0, \mathbf{C}_T = 0, \mathbf{D}_T = 0, \mathbf{E}_T = \mathbf{I}, \mathbf{F}_T = 0, \mathbf{G}_T = \mathbf{I}, \mathbf{H}_T = 0, \mathbf{m}_{1T} = 0, \mathbf{n}_{1T} = 0$$

Each of these store the generic update information referred to. The subscript T is used to signify "Total". Note $\mathbf{A}_T$ ...$\mathbf{H}_T$ are square matrices equal in size to $\mathbf{y}_j$ while $\mathbf{m}_{1T}$ and $\mathbf{n}_{1T}$ are column vectors of the same dimension.

**Prediction**

While the cameras are in motion, the components of the estimated state vector and covariance are updated as below:

$$\hat{\mathbf{x}}_{v(new)} = \mathbf{f}_v(\hat{\mathbf{x}}_v, \mathbf{u}) \qquad \textbf{5. 114}$$

$$\hat{\mathbf{y}}_{i(new)} = \hat{\mathbf{y}}_i \qquad \textbf{5. 115}$$

$$\mathbf{P}_{xx(new)} = \nabla(\mathbf{f}_v)_{x_v} \mathbf{P}_{xx} \nabla(\mathbf{f}_v)_{x_v}^T + \mathbf{Q} \qquad \textbf{5. 116}$$

$$\mathbf{P}_{xy_i(new)} = \nabla(\mathbf{f}_v)_{x_v} \mathbf{P}_{xy_i} \qquad \textbf{5. 117}$$

$$\mathbf{P}_{y_i y_i(new)} = \mathbf{P}_{y_i y_i} \qquad \textbf{5. 118}$$

And stored matrices:

$$\mathbf{E}_{T(new)} = \nabla(\mathbf{f}_v)_{x_v} \mathbf{E}_T \qquad \textbf{5. 119}$$

$$\mathbf{E}_{T(new)} = \nabla(\mathbf{f}_v)_{x_v} \mathbf{F}_T \qquad \textbf{5. 120}$$

## Measurement Update

The following matrices and updates are calculated for measurement of feature i:

$$\mathbf{A} = \nabla(\mathbf{m}_i)^T_{\mathbf{x_v}} \mathbf{S}^{-1} \nabla(\mathbf{m}_i)_{\mathbf{x_v}} \quad \text{5. 121}$$

$$\mathbf{B} = \nabla(\mathbf{m}_i)^T_{\mathbf{x_v}} \mathbf{S}^{-1} \nabla(\mathbf{m}_i)_{\mathbf{y}_i} \quad \text{5. 122}$$

$$\mathbf{C} = \nabla(\mathbf{m}_i)^T_{\mathbf{y}_i} \mathbf{S}^{-1} \nabla(\mathbf{m}_i)_{\mathbf{x_v}} \quad \text{5. 123}$$

$$\mathbf{D} = \nabla(\mathbf{m}_i)^T_{\mathbf{y}_i} \mathbf{S}^{-1} \nabla(\mathbf{m}_i)_{\mathbf{y}_i} \quad \text{5. 124}$$

$$\mathbf{E} = \mathbf{I} - \left[\mathbf{P}_{\mathbf{xx}}\mathbf{A} + \mathbf{P}_{\mathbf{xy}_i}\mathbf{C}\right] \quad \text{5. 125}$$

$$\mathbf{F} = -\left[\mathbf{P}_{\mathbf{xx}}\mathbf{B} + \mathbf{P}_{\mathbf{xy}_i}\mathbf{D}\right] \quad \text{5. 126}$$

$$\mathbf{G} = \mathbf{I} - \left[\mathbf{P}_{\mathbf{y}_i\mathbf{x}}\mathbf{B} + \mathbf{P}_{\mathbf{y}_i\mathbf{y}_i}\mathbf{D}\right] \quad \text{5. 127}$$

$$\mathbf{H} = -\left[\mathbf{P}_{\mathbf{y}_i\mathbf{x}}\mathbf{A} + \mathbf{P}_{\mathbf{y}_i\mathbf{y}_i}\mathbf{C}\right] \quad \text{5. 128}$$

$$\hat{\mathbf{x}}_{\mathbf{v}(new)} = \hat{\mathbf{x}}_{\mathbf{v}} + \mathbf{P}_{\mathbf{xx}}\nabla(\mathbf{m}_i)^T_{\mathbf{x_v}}\,\mathbf{S}^{-1}\nu + \mathbf{P}_{\mathbf{xy}_i}\nabla(\mathbf{m}_i)^T_{\mathbf{y}_i}\mathbf{S}^{-1}\nu \quad \text{5. 129}$$

$$\hat{\mathbf{y}}_{\mathbf{i}(new)} = \hat{\mathbf{y}}_i + \mathbf{P}_{\mathbf{y}_i\mathbf{x}}\nabla(\mathbf{m}_i)^T_{\mathbf{x_v}}\,\mathbf{S}^{-1}\nu + \mathbf{P}_{\mathbf{y}_i\mathbf{y}_i}\nabla(\mathbf{m}_i)^T_{\mathbf{y}_i}\mathbf{S}^{-1}\nu \quad \text{5. 130}$$

$$\mathbf{P}_{\mathbf{xx}(new)} = \mathbf{P}_{\mathbf{xx}} - \left(\mathbf{P}_{\mathbf{xx}}\mathbf{A}\mathbf{P}_{\mathbf{xx}} + \mathbf{P}_{\mathbf{xx}}\mathbf{B}\mathbf{P}_{\mathbf{y_i x}} + \mathbf{P}_{\mathbf{xy}_i}\mathbf{C}\mathbf{P}_{\mathbf{xx}} + \mathbf{P}_{\mathbf{xy}_i}\mathbf{D}\mathbf{P}_{\mathbf{y}_i\mathbf{x}}\right) \quad \text{5. 131}$$

$$\mathbf{P}_{\mathbf{xy}_i(new)} = \mathbf{P}_{\mathbf{xy}_i} - \left(\mathbf{P}_{\mathbf{xx}}\mathbf{A}\mathbf{P}_{\mathbf{xy}_i} + \mathbf{P}_{\mathbf{xx}}\mathbf{B}\mathbf{P}_{\mathbf{y}_i\mathbf{y}_i} + \mathbf{P}_{\mathbf{xy}_i}\mathbf{C}\mathbf{P}_{\mathbf{xy}_i} + \mathbf{P}_{\mathbf{xy}_i}\mathbf{D}\mathbf{P}_{\mathbf{y}_i\mathbf{y}_i}\right) \quad \text{5. 132}$$

$$\mathbf{P}_{\mathbf{y}_i\mathbf{y}_i(new)} = \mathbf{P}_{\mathbf{y}_i\mathbf{y}_i} - \left(\mathbf{P}_{\mathbf{y}_i\mathbf{x}}\mathbf{A}\mathbf{P}_{\mathbf{xy}_i} + \mathbf{P}_{\mathbf{y}_i\mathbf{x}}\mathbf{B}\mathbf{P}_{\mathbf{y}_i\mathbf{y}_i} + \mathbf{P}_{\mathbf{y}_i\mathbf{y}_i}\mathbf{C}\mathbf{P}_{\mathbf{xy}_i} + \mathbf{P}_{\mathbf{y}_i\mathbf{y}_i}\mathbf{D}\mathbf{P}_{\mathbf{y}_i\mathbf{y}_i}\right) \quad \text{5. 133}$$

Stored matrices and vectors:

$$\mathbf{m}_{\mathbf{1T}(new)} = \mathbf{m}_{\mathbf{1T}} + \left(\mathbf{E}^T_{\mathbf{T}}\nabla(\mathbf{m}_i)^T_{\mathbf{x_v}} + \mathbf{H}^T_{\mathbf{T}}\nabla(\mathbf{m}_i)^T_{\mathbf{y}_i}\right)\mathbf{S}^{-1}\nu \quad \text{5. 134}$$

$$\mathbf{n}_{1T(new)} = \mathbf{n}_{1T} + \left(\mathbf{F}_T^T \nabla(\mathbf{m}_i)_{x_v}^T + \mathbf{G}_T^T \nabla(\mathbf{m}_i)_{y_i}^T\right)\mathbf{S}^{-1}\nu \quad \text{5. 135}$$

$$\mathbf{B}_{T(new)} = \mathbf{B}_T + \mathbf{E}_T^T\mathbf{A}\mathbf{F}_T + \mathbf{E}_T^T\mathbf{B}\mathbf{G}_T + \mathbf{H}_T^T\mathbf{C}\mathbf{F}_T + \mathbf{H}_T^T\mathbf{D}\mathbf{G}_T \quad \text{5. 136}$$

$$\mathbf{C}_{T(new)} = \mathbf{C}_T + \mathbf{F}_T^T\mathbf{A}\mathbf{E}_T + \mathbf{F}_T^T\mathbf{B}\mathbf{H}_T + \mathbf{G}_T^T\mathbf{C}\mathbf{E}_T + \mathbf{G}_T^T\mathbf{D}\mathbf{H}_T \quad \text{5. 137}$$

$$\mathbf{D}_{T(new)} = \mathbf{D}_T + \mathbf{F}_T^T\mathbf{A}\mathbf{F}_T + \mathbf{F}_T^T\mathbf{B}\mathbf{G}_T + \mathbf{G}_T^T\mathbf{C}\mathbf{F}_T + \mathbf{G}_T^T\mathbf{D}\mathbf{G}_T \quad \text{5. 138}$$

$$\mathbf{E}_{T(new)} = \mathbf{E}\mathbf{E}_T + \mathbf{F}\mathbf{H}_T \quad \text{5. 139}$$

$$\mathbf{F}_{T(new)} = \mathbf{E}\mathbf{F}_T + \mathbf{F}\mathbf{G}_T \quad \text{5. 140}$$

$$\mathbf{G}_{T(new)} = \mathbf{G}\mathbf{G}_T + \mathbf{H}\mathbf{F}_T \quad \text{5. 141}$$

$$\mathbf{H}_{T(new)} = \mathbf{G}\mathbf{H}_T + \mathbf{H}\mathbf{E}_T \quad \text{5. 142}$$

Note:
(1) The order of calculating $\mathbf{m}_{1T}$, $\mathbf{n}_{1T}$ and $\mathbf{A}_T$ ... $\mathbf{D}_T$ is important as the expressions make use of old values of $\mathbf{E}_T$ .... $\mathbf{H}_T$
(2) Similarly for equations 5.139 to 5.141, matrices on the right hand side of expressions use old versions of $\mathbf{E}_T$ .... $\mathbf{H}_T$.

To accommodate these requirements careful storage of $\mathbf{E}_T$ to $\mathbf{H}_T$ is required. The benefits are that the computational expense in making updates to the stored matrices and vectors at prediction and measurement is constant, independent of the number of features in the map. To achieve this, parts of the total state vector and covariance matrix not related to the current update are not processed until later at the end of the motion. The information required for this later processing being stored in a generic way.

**Final Full Update**

Once the motion has been completed, the full estimated state vector can be readily brought up to date using the following equation.

For each unobserved feature j:

$$\hat{\mathbf{y}}_j = \hat{\mathbf{y}}_j(0) + \mathbf{P}_{y_j x}(0)\mathbf{m}_{1T} + \mathbf{P}_{y_j y_i}(0)\mathbf{n}_{1T} \quad \text{5. 143}$$

The update of the covariance between a pair of unobserved features j and k is:

$$\mathbf{P}_{y_j y_k} = \mathbf{P}_{y_j y_k}(0) - \left[\mathbf{P}_{y_j x}(0)\mathbf{A}_T\mathbf{P}_{xy_k}(0) + \mathbf{P}_{y_j x}(0)\mathbf{B}_T\mathbf{P}_{y_i y_k}(0) + \mathbf{P}_{y_j y_i}(0)\mathbf{C}_T\mathbf{P}_{xy_k}(0) + \mathbf{P}_{y_j y_i}(0)\mathbf{D}_T\mathbf{P}_{y_i y_k}(0)\right]$$

5. 144

To update the parts of the covariance matrix relating unobserved feature j to the sensor state and the unobserved feature state (equations 5.145 and 5.146):

$$\mathbf{P}_{xy_j} = \mathbf{E}_T\mathbf{P}_{xy_j}(0) + \mathbf{F}_T\mathbf{P}_{y_i y_j}(0)$$

5. 145

$$\mathbf{P}_{y_i y_j} = \mathbf{G}_T\mathbf{P}_{y_i y_j}(0) + \mathbf{H}_T\mathbf{P}_{xy_j}(0)$$

5. 146

**Choice of Feature**

Following a feature measurement **m**, the innovation covariance matrix **S** formed provides a measure of how much the actual measurement is expected to vary from that predicted in angular measurement coordinates the (α, e, γ) (see equation 5.40). This knowledge can be used as the basis for deciding which feature to track. The volume in (α, e, γ) space of the ellipsoid represented by **S** at the $n_\sigma\sigma$ level can be calculated for each visible feature: eigenvalues $\lambda_1$, $\lambda_2$, $\lambda_3$, of S mean that the ellipsoid has axes of length $n_\sigma\sqrt{\lambda_1}$, $n_\sigma\sqrt{\lambda_2}$ *and* $n_\sigma\sqrt{\lambda_3}$. If this volume is labelled $V_s$ (equation 5.147), then its calculation for each of the features currently visible, provides a criterion for selection, based on the feature with the highest value.

$$V_S = \frac{4}{3}\pi n_\sigma^3\sqrt{\lambda_1\lambda_2\lambda_3}$$

5. 147

This criterion takes no account of the time the head would take to saccade to an alternative feature. For instance, if two features are widely apart (one in the front of the robot and the other at 90 degrees), then the time for the head to moves to a position where it can commence the measurement of a new feature can be quite long (perhaps 1 second). This impinges significantly on the total navigational time. An optimal strategy would incorporate a penalty for this saccading time into the choice-of –feature criterion. The Davison criterion adopted for this work does so, and is outlined below.

The decision, as to whether to switch to a new feature or stay tracking the existing one, is made while the robot is moving. The choice is based on the predicted robot and map state that would exist in a short future interval of time if (a) the saccade and measure of the chosen feature was made or (b) tracking of the existing feature continued. The referred to "short future interval of time", is the estimated time it would take to saccade to and measure any feature under consideration.

Formally the decision process is implemented according to:

1. $N_i$, the number of measurements that would be lost while saccading to each of the visible features is calculated. To determine this, an estimate of head movement time, to correctly move to and locate each feature, is required.

2. Ascertain the largest $N_i$ ($N_{max}$); this represents the largest number of measurements lost during the longest saccade.

3. If an immediate saccade is to be made, estimate for each feature i the state that would exist after $N_{max} + 1$ steps.

4. Evaluate Vs(max) for each of the above estimated states and saccade to the feature with the lowest Vs(max), unless the Vs(max) of the feature being tracked is already the lowest.

# Bibliography

[AA91] F. Arman and J. K. Aggarwal. "Automatic generation of recognition strategies using CAD models," *Workshop on Directions in Automated CAD Based Vision*, pp. 124-133, Los Alamitos, California, IEEE, Computer Society Press, June 1991.

[AA93] F. Arman and J. K. Aggarwal, "Model-based object recognition in dense-range images -- A review," *ACM Computing Surveys*, 25(1), pp. 5-43, 1993.

[AL91] M. Andersson and A. Lundquist, "Tracking lines in a stereo image sequence," *Technical Report TRITANAP9116*, Dept. of Numerical Analysis and Computing Science, Royal Institute of Technology, July 1991.

[AN93a] M. Andersson and P. Nordlund, "Modeling, Matching and Tracking for the Stereovision II project," *Technical Report TRITANAP9322*, Dept. of Numerical Analysis and Computing Science, Royal Institute of Technology, June 1993.

[AN93b] M. Andersson and P. Nordlund, "Modeling, Matching and Tracking for the Stereovision II project: Program Description", *Technical Note CVAP TN12*, Dept. of Numerical Analysis and Computing Science, Royal Institute of Technology, June 1993.

[ANE93] M. Andersson, P. Nordlund, and J.O. Eklundh, "Modeling, matching and tracking for the Stereovision II project: State-of-the-Art review", R*eport, ISRN* KTH/NA/P-93/21-SE, June 1993.

[Ark98] R.C. Arkin, "Behavior-Based Robotics," Cambridge, MA*: MIT Press*, 1998.

[AS97] C.G. Atkeson and S. Schaal, "Robot learning from demonstration", *Jr. D. H. Fisher, editor, in Proc. of the Int. Conf. on Machine Learning*, pp. 12-20, Morgan Kaufmann, 1997.

[Aya91] N. Ayache, "Artificial Vision for Mobile Robots: Stereo Vision and Multi-sensory Perception," *MIT Press*, Cambridge MA,1991.

[AZH96] M. Armstrong, A. Zisserman and R. Hartley, "Selfcalibration from image triplets," *B. Buxton and R. Cipolla, editors, in Proc. 3rd European Conf. on Computer Vision,* Cambridge, Volume 1, pp. 3-16, April 1996.

[BA02] D.C. Bentivegna and C.G. Atkeson, "Learning How to Behave from Observation," *SAB'02-Workshop on Motor Control in Humans and Robots: on the interplay of real brains and artificial devices*, Edinburgh, UK, August, 2002.

[Baj88] R. Bajcsy, "Active perception," *in Proc. IEEE, 76(8)*, pp. 996-1005, 1988.

[BCFHLSST98] W. Burgard, A.B. Cremers, D. Fox, Hahnel , G. Lakemeyer, D. Schulz, W. Steiner and S. Thrun, "The interactive museum tour-guide robot," *in Proc. of the Fifteenth National Conf. on Artificial Intelligence*, 1998.

[BDFC98] W. Burgard, A. Derr, D. Fox, and A.B. Cremers, "Integrating global position estimation and position tracking for mobile robots: The dynamic Markov localization approach," *in Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 1998.

[BEF96] J. Borenstein, H. Everett and L. Feng, "Navigating Mobile Robots: Systems and Techniques," A.K. Peters Ltd., Wellesley, MA, 1996.

[Ben96] A. Bennet, "Do animals have cognitive maps ?," *Journal of Experimental Biology*, pp. 219-224, 1996.

[Bes88] P.J. Besl. "Geometric modeling and computer vision," *in Proc. of the IEEE, 76(8)*, pp. 936-958, August 1988.

[Bet96] A. Bethell, "A hippocampally-inspired connectionist model for mobile robot localisation," *Master's thesis,* Department of Computer Science, University of Manchester, 1996.

[BHH84] R.C. Bolles, P. Horaud and M.J. Hannah, "3DPO: A three dimensional part orientation system," *Michael Brady and Richard Paul, editors, Robotics Research*, pp. 413-424, Cambridge, Massachusetts, USA, National Science Foundation and System Development Corporation, The MIT Press, 1984

[Bie85] I. Biederman, "Human image understanding: recent research and theory," *Computer Graphics and Image Processing*, 32, pp. 29-73, 1985.

[Bin82] T.O. Binford, "Survey of Model-Based Image Analysis Systems," *The Int. Journal of Robotics Research Vol.1 No.1,* 1982.

[Bin87] T.O. Binford, "Generalized Cylinder Representation," *Encyclopedia of Artificial Intelligence*, pp. 321-323, John Wiley & Sons, 1987.

[Bis95] C. Bishop, "Neural Networks for Pattern Recognition," Oxford, England: *Oxford University Press*, 1995.

[BJ85] P.J. Besl and R.C. Jain, "Three-dimensional object recognition," *ACM Computing Surveys, 17(1)*, pp. 75-145, 1985.

[BK96] P. Bakker and Y. Kuniyoshi, "Robot see, robot do: An overview of robot imitation," *AISB96 Workshop on Learning in Robots and Animals*, pp. 3-11, 1996.

[BK96b] C. Balkenius and L. Kopp, "The XT-1 vision architecture," *in Proc. of the Symposium on Image Analysis*, Lund University, Sweden, pp. 39-43, 1996.

[BMRM94] K.J. Bradshaw, P.F. McLauchlan, I.D. Reid and D.W. Murray, "Saccade and pursuit on an active head/eye platform," *Image and Vision Computing*, 12(3), pp. 155-163, 1994.

[BNA89] J.P. Brady, N. Nandhakumar and J.K. Aggarwal, "Recent progress in object recognition from range data," *Image and Vision Computing*, 7(4), pp. 295-307, November 1989.

[BO88] D.H. Ballard and A. Ozcandarli, "Eye fixation and kinetic depth," *in Proc. 2nd Int. Conf. on Computer Vision*, Tampa, pp. 524-532,1988.

[Bow91] K. Bowyer, "Why aspect graphs are not (yet) practical for computer vision," *Workshop on Directions in Automated CAD Based Vision*, pp. 98-104, Los Alamitos, California, IEEE, Computer Society Press, June 1991.

[BP95] J.Y. Bouget and P. Perona, "Visual navigation using a single camera," *ICCV5*, pp. 645-652, Los Alamitos, CA, IEEE Computer Society Press, 1995.

[Bro81] R.A. Brooks, "Symbolic reasoning among 3D models and 2D images," *J. of Artificial Intelligence*, 17(1-3), pp. 285-348, 1981.

[Bro86] R.A. Brooks, "A robust layered control system for a mobile robot," *Technical Report* 864, Massachusetts Institute of Technology, 1986.

[Bro87] R.A. Brooks. "Intelligence without representation," *Workshop on the Foundations of Artificial Intelligence*, Dedham, MA, 1987.

[Bro90] R.A. Brooks and L.A. Stein. "Builing brains for bodies," *Technical Report A.I. Memo 1439*, Massachusetts Institute of Technology, August 1993.

[Bro90b] C.M. Brown, "Gaze control with interactions and delays," *IEEE Trans. Sys. Man and Cyb., 63,* pp. 61-70, 1990.

[BRZW95] P.A. Beardsley, I. D. Reid, A. Zisserman and D.W. Murray, "Active visual navigation using nonmetric structure," *in Proc. 5th Int. Conf. on Computer Vision*, Boston, pp. 58-65. IEEE Computer Society Press, 1995.

[Bur94] T.P.H. Burke, "Design of a Modular Mobile Robot," *PhD thesis*, University of Oxford, 1994.

[Bur96] W. Burgard, D. Fox, D. Henning and T. Schmidt, "Estimating the absolute position of a mobile robot using position probability grids," *in Proc. of the Thirteenth National Conf. of Artificial Intelligence* (AAAI'96), pp. 896-901, 1996.

[BW92] D. Ballard and S. Whitehead, "Learning visual behaviours," *Wechsler, H., ed., Neural Networks for Perception*, New York: Academic Press, 1992

[BW02] N.Burley and M.Wingate, "An Active Camera Target Surveillance Tracker," *4th Int. on Conf. Modelling and Simulation,* pp. 440-445, Australia, Nov 2002.

[BWB94] M. Brady H. Wang and C. Bowman, "Architectures and algorithms for 3D vision: the parallel Droid system," *S. Cameron and P. Probert, editors, Advanced Guided Vehicles,* pp. 125-139. World Scientific, Singapore, 1994.

[BZC92] A. Blake, A. Zisserman and R. Cipolla, "Visual exploration of free space," *A. Blake and A. Yuille, editors, Active Vision,*" MIT Press, Cambridge, MA, 1992.

[BZM94a] P.A. Beardsley, A. Zisserman and D. W. Murray, "Navigation using affine structure from motion," *in Proc. 3rd European Conf. on Computer Vision,* Stockholm, vol. 2, pp. 85-96, 1994.

[BZM94b] P.A. Beardsley, A. Zisserman and D. W. Murray, "Sequential update of projective and affine structure and motion," *Technical Report OUEL report 2012/94*, Dept. of Engineering Science, University of Oxford, 1994.

[Can83] J.F. Canny, "Finding edges and lines in images," Master's thesis, MIT, 1983.

[Cap99] F. Caparrelli, "View-Based Three-Dimensional Object Recognition Using Pairwise Geometric Histograms," *Ph.D Thesis,* University Sheffield, Sept 1999.

[Car02] O.Carmichael, "Discriminant Filters for Object Recognition," T*ech. Report CMU-RI-TR-02-09*, Robotics Institute, Carnegie Mellon University, March, 2002.

[CB90] R. Cipolla and A. Blake, "The dynamic analysis of apparent contours," *in Proc. 3rd Int. Conf. on Computer Vision*, Osaka, 1990.

[CCV85] I.Carlbom, I. Chakravarty and D. Vanderschel, "A Hierarchical Data Structure for Representing the Spatial Decomposition of 3D Objects," *in Frontiers in Computer Graphics (T.L. Kunii ed.),* pp. 2-12. Springer-Verlag: New York, 1985

[CD86] R.T. Chin and C.R. Dyer, "Model-based recognition in robot vision," *ACM Computing Surveys*, 18(1), pp. 67-108, 1986.

[CDT00] J.A. Castellanos, M. Devy, J.D Tardos "Simultaneous Localization and Map Building for Mobile Robots: A Landmark-based Approach," *IEEE Int. Conferenceon Robotics and Automation: Workshop on Mobile Robot Navigation and Mapping*, San Francisco, *2000.*

[CG87] G. Carpenter and S. Grossberg, "ART2 : Self organization of stable category recognition codes for analog input patterns," *Applied Optics* 26(23), pp. 4919-4930, 1987.

[CH02] O.Carmichael and M.Herbert, "Object Recognition by a Cascade of Edge Probes," *British Machine Vision Conf. 2002*, British Machine Vision Association, Vol. 1, September, pp. 103-110, 2002.

[CJ94] J.D. Courtney and A.K. Jain, "Mobile robot localization via classification of multi-sensor maps," *in Proc. of the IEEE Int. Conf. on Robotics and Automation*, pp. 1672-1678, 1994.

[CKK96] A. Cassandra, L.P. Kaelbling and J. Kurien, "Acting under uncertainty: Discrete Bayesian models for mobile-robot navigation," *in Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, 1996.*

[CL94] I.J. Cox and J.J. Leonard, "Modeling a dynamic environment using a Bayesian multiple hypothesis approach," *Artificial Intelligence 66*, pp. 311-344, 1994.

[Cl98] J.C. Clarke, "Applications of Sequence Geometry to Visual Motion," *PhD thesis,* University of Oxford, 1998.

[CM93] A.Y.S Chong and M.Wingate, "A Colour Feature Based Stereo Vision System for Robot Assembly," *ICA/ACME Int. Conf. on Assembly combined with the Australian Conf. on Manufacturing Engineering, Nov. 1993.*

[Cox 91] I.J. Cox, " 'BLANCHE', an experiment in guidance and navigation of an autonomous robot vehicle," *IEEE Transactions on Robotics and Automation 7*, pp. 193-204, 1991.

[CP94] S.A. Cameron and P.J. Probert, editors, "Advanced Guided Vehicles --- Aspects of the Oxford AGV Project," *World Scientific*, Singapore, 1994.

[Cro95] J.L. Crowley, "Mathematical foundations of navigation and perception for an autonomous mobile robot," *Workshop on Reasoning With Uncertainty in Robotics. Tutorial paper*, 1995.

[Cso97] M. Csorba, "*Simultaneous Localisation and Map Building*," PhD thesis, Oxford University, 1997.

[CUD96] M. Csorba, J. K. Uhlmann and H. F. DurrantWhyte, "A new approach to simultaneous localization and dynamic map building," *SPIE in Proc.*, pp. 26-36, 1996.

[Dav98] A.J.Davison, "Mobile Robot Navigation Using Active Vision," *Ph.D. thesis,* University of Oxford, 1998.

[Dav01] A.J.Davison, "Models and State Representation in Scene: Generalised Software for Sequential Map-Building and Localisation," Feb. 6, 2001: http://www.robots.ox.uk/~ajd/ .

[DC98] M. Dorigo and M. Columbetti, "Robot Shaping: an experiment in behavior engineering," MIT Bradford Press, 1998.

[DFBT99] F. Dellaert, D. Fox, W. Burgard and S. Thrun, "Monte Carlo localization for mobile robots," *in Proc. of the IEEE Int. Conf. on Robotics and Automation*, pp. 1322-1328,1999.

[DHart73] R. Duda and P. Hart, "Pattern Classification and Scene Analysis," J. Wiley and Sons, 1973.

[dHB98] L. de Agapito, D.Q. Huynh and M.J. Brooks, "Self-calibrating a stereo head: an error analysis in the neighbourhood of degenerate configurations," *in Proc. 6th Int. Conf. on Computer Vision,* Bombay, pp. 747-753, 1998.

[Dic91] S.J. Dickinson, "The recovery and recognition of three dimensional objects using part-based aspect matching," *Technical Report CARTR572*, CS

[Dic99] E.D. Dickmanns, "Computer Vision and Highway Automation," *Vehicle System Dynamics,* vol. 31, no 5, pp. 325-343, 1999.

[DK00] A. J. Davison and N. Kita,. "Sequential localisation and map-building," *Computer vision and robotics. Int. in Proc. of the 2nd Workshop on Structure from Multiple Images of Large Scale Environments (SMILE)," in conjunction with ECCV* 2000, Dublin, Ireland. Springer-Verlag LNCS, 2000.

[DK02] G.N.DeSouza, A.C. Kak, "Vision for Mobile Robot Navigation:A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24, No. 2, pp. 237-264, 2002.

[DKH98] H.A.L. van Dijck, M.J. Korsten F. van der Heijden, " 2 and 3Dimensional Geometric Hashing with Points and Lines," *in Proc. of the Fourth Annual Conf. of the Advanced School for Computing and Imaging, ASCI'98,* pp. 51-57, Lommel, Belgium, ISBN 09-803086-3-3, 1998

[DM92] E.D. Dickmanns and B. Mysliwetz, "Recursive 3D Road and Relative Egostate Recognition," *IEEE trans. Pattern Analysis and Machine Intelligence, vol 14, no. 2,* pp. 499-507, Feb, 1992

[DN96] T. Duckett and U. Nehmzow, "A robust perception-based localisation method for a mobile robot," *Technical Report UMCS-96-11-1*, Department of Computer Science, University of Manchester, 1996.

[DN97a] T. Duckett and U. Nehmzow, "Experiments in evidence based localisation for a mobile robot," *in Proc. of the AISB'97 Workshop on Spatial Reasoning in Animals and Robots,* pp. 25-34, Department of Computer Science, Manchester University, Technical Report Series, ISSN 1361-6161, Report UMCS-97-4-1, 1997.

[DN97b] T. Duckett and U. Nehmzow, "Knowing your place in the real world," *ECAL-97 Fourth European Conf. on Artificial Life*, 1997.

[DN98] T. Duckett and U. Nehmzow, "Mobile robot self-localisation and measurement of performance in middle scale environments," *Robotics and Autonomous Systems 24(1{2)*, pp. 57-69, 1998.

[DN99] T. Duckett and U. Nehmzow, "Learning to predict sonar readings for mobile robot landmark selection," *Report of the 4th visit in the Academic Research Collaboration* (ARC) funded by the British Council and Deutscher Akademischer Austauschdienst, June 24 to July 13, Bremen University, Germany, 1999.

[DNDCC99] M.W. Dissanayake, P. Newman, H.F. Durrant-Whyte, S. Clark, and M. Csorba. "A solution to the simultaneous localisation and map building (slam) problem," *Internal Tech Report ACFR-TR-01-99, Australian Center for Field Robotics*, University of Sydney, Australia, January 1999.

[Don93] J. Donnett "Analysis and Synthesis in the Design of Locomotor and Spatial Competences for a Multisensory Mobile Robot," *Ph.D. Dissertation*, Edinburgh University, 1993.

[Dra93] B. Draper "Learning from the Schema Learning System," In *AAAI Symposium on Machine Learning and Computer Vision*, pp. 75-79, 1993.

[DRLR88] M. Dhome, M. Richetin, J.T. Lapreste and G. Rives, "The inverse of perspective problem from a single view for polyhedra location," *in Proc. of IEEE Computer Vision and Pattern Recognition*, pp. 61-66, June 1988.

[DRLR89] M. Dhome, M. Richetin, J.T. Lapreste and G. Rives, "Determination of the attitude of 3d objects from a single perspective view," *IEEE Trans. Pattern Anal. and Machine Intell.*, 11(12), pp. 1265-1278, 1989.

[DRM95] A.J. Davison, I.D. Reid and D.W. Murray, "The active camera as a projective pointing device," *in Proc. 6th British Machine Vision Conf.*, Birimingham, pp. 453-462, 1995.

[Duc00] T.D. Duckett, "Concurrent Map Building and Self Localisation for Mobile Robot Navigation," *Ph.D Thesis,* Univ. of Manchester, U.K., 2000.

[Dur94] H.W. Durrant-Whyte, "Where am I? A tutorial on mobile vehicle localization," *Industrial Robot*, 21(2), pp. 11-16, 1994.

[DZ86] E.D. Dickmanns and A. Zapp, "A Curvature-Based Scheme for Improving, Road Vehicle Guidance by Computer Vision," *in Proc. SPIE-The Int. Soc. Optical Eng-Mobile Robots, vol. 727*, pp. 161-168, Oct. 1986.

[DZ87] E.D. Dickmanns and A. Zapp, "Autonomous High Speed Road Vehicle Guidence by Computer Vision," *in Proc. 10th World Congress on Automatic Control, vol 4,* pp. 232-237, 1987.

[EBDCG92] D.W. Eggert, K.W. Bowyer, C. R. Dyer, H. I. Christensen and D. B. Goldgof, "Applying the scale space concept to perspective projection aspect graphs," *Technical Report LIA 92-02,* Laboratory of Image Analysis, Institute of Electronic Systems, Fredrik Bajers Vej 7D, DK 9220 Aalborg O, Denmark, January 1992. Extract from the book from 7th Scandinavian Conference on Image Analysis.

[Elfes 1987] A. Elfes, "Sonar-based real-world mapping and navigation," *IEEE Journal of Robotics and Automation,* RA-3(3), pp. 249-265,1987.

[EM92] S.P. Engelson and D.V. McDermott, "Error correction in mobile robot map learning," *in Proc. of the IEEE Int. Conf. on Robotics and Automation,* pp. 2555-2560, 1992.

[Eng94] S. Engelson, "Passive Map Learning and Visual Place Recognition," *Ph.D. Dissertation*, Department of Computer Science, Yale University, 1994.

[ETM93a] A.C.Evans, N.A.Thacker and J.E.W.Mayhew, 'The Use of Geometric Histograms for Model-Based Object Recognition," *Proc. 4th BMVC93*, Guildford, 21-23 pp. 429 - 438, Sept. 1993.

[ETM93b] A.C.Evans, N.A.Thacker and J.E.W.Mayhew, "A Practical View Based 3D Object Recognition System," *IEE conference on neural networks*, Brighton, pp. 6-10, 1993.

[Eva90] R. Evans, "Kalman filtering of pose estimates in applications of the RAPID video rate tracker," *in Proc. of the BMVC*, pp. 79-84, 1990.

[EW95] T. Edlinger and G. "Weiss, Exploration, navigation and self localisation in an autonomous mobile robot, " *Autonomous Mobile Systems*, pp. 142-151, 1995.

[Fau86] O.D. Faugeras, "The representation, recognition, and locating of 3D objects," *Int. Journal of Robotic Research*, 5(3): pp. 27–52, Fall 1986.

[Fau92] O.D. Faugeras, "What can be seen in three dimensions with an uncalibrated stereo rig?," *G. Sandini, editor, in Proc. 2nd European Conf. on Computer Vision*, Santa Margharita Ligure, Italy, pp. 563-578, Springer-Verlag, 1992.

[Fau93] O.D. Faugeras, "Three Dimensional Computer Vision," MIT Press, 1993.

[Fau95] O.D. Faugeras, "Stratification of three-dimensional vision: projective, affine, and metric representations," *J. Opt. Soc. Am.* A, 12(3) pp. 465-484 (1995).

[FBT98] D. Fox, W. Burgard and S. Thrun, "Active Markov localisation for mobile robots," *Robotics and Autonomous Systems 25*: pp. 195-207, 1998.

[Fer91] N.J. Ferrier, "The harvard binocular head," *Technical Report 919*, Harvard Robotics Laboratory, 1991.

[FHPP84] Faugera, O D,Hebert, M, Ponce J, and Pauchon, E., "Object representation, identification, and positioning from range data," *Proc 1st Int. Symp. On Robotics Res.* MIT Press, Cambridge, MA, USA, 1984, pp. 425 – 446.

[Fis89] Robert B. Fisher, "From Surfaces to Objects," *John Wiley & Sons*, Baffins Lane, Chichester, West Sussex, PO19 1UD England, 1990.

[FJ91a] P.J. Flynn and A.K. Jain, "Bonsai: 3d object recognition using constrained search," *IEEE Trans. Pattern Anal. and Machine Intell., 13(10)*, pp. 1066-1075, October 1991.

[FJ91b] P.J. Flynn and A.K. Jain, "CAD based computer vision: From CAD models to relational graphs," *IEEE Trans. Pattern Anal. and Machine Intell.*, 13(2), pp. 114-132, 1991.

[FMN88] T.J. Fan, G. Medioni and R. Nevatia, "3D object recognition using surface descriptions," *in Proc. of the Image Understanding Workshop,* pp. 383-397. DARPA, April 1988.

[FP02] D. Forsyth and J. Ponce, "Computer Vision -- A modern approach," *To be* Published Prentice Hall 2002 ?) Currently On Line.

[FRM95] S.M. Fairley, I.D. Reid and D.W. Murray, "Transfer of fixation for an active stereo platform via affine structure recovery," *in Proc. 5th Int. Conf. on Computer Vision*, Boston, pp. 1100-1105, 1995.

[Fri95] B.Fritze, "A Growing Neural Gas Network Learns Topologies," *Advances in Neural Information Processing Systems*, 7, 1995.

[FSMB98] M. Franz, B. Schoelkopf, H. Mallot and H. Bueltho, "Learning view graphs for robot navigation," *Autonomous Robots* 5, pp. 111-125,1998.

[Gall90] C.R. Gallistel, "The Organisation of Learning," MIT Press, Cambridge MA, 1990.

[Gat98] E. Gat, "Three layer architectures," In D. *Kortenkamp, R. Bonasso, and R. Murphy, eds., AI-based Mobile Robots: Case Studies of Successful Robot Systems*. Cambridge, MA: MIT Press, 1998.

[GBFK98] J.S. Gutmann, W. Burgard, D. Fox and K. Konolige, "An experimental comparison of localization methods," *in Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 1998.

[Gel96] A. Gelb, "Applied Optimal Estimation," MIT Press, 14th Edition, 1996.

[GH91] W.E.L. Grimson and Daniel P. Huttenlocher, "On the verification of hypothesized matches in model based recognition," *IEEE Trans. Pattern Anal. and Machine Intell., 13(12)*, pp. 1201-1213, December 1991.

[Gib85] A. Gibbons, "Algorithmic Graph Theory," Cambridge, England: Cambridge University Press, 1985.

[GLP84] W.E.L. Grimson and T. LozanoPerez, "Mode-based recognition and localization from sparse range or tactile data," *Int. Journal of Robotics Research, 3(3)*, pp. 3-35, 1984.

[GLP87] W.E.L. Grimson and T. LozanoPerez, "Localizing overlapping parts by searching the interpretation tree," *IEEE Trans. Pattern Anal. and Machine Intell., 9(4)*, pp. 469-482, July 1987.

[GMR98] M. Golfarelli, D. Maio and S. Rizzi, "Elastic correction of dead-reckoning errors in map building," *in Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems,* pp. 905-911, 1998.

[GN98] J. Gluckman and S.K. Nayar, "Egomotion and omnidirectional cameras," *in Proc. 6th Int. Conf. on Computer Vision*, Bombay, pp. 999-1005, 1998.

[Goa83] C. Goad, "Special purpose automatic programming for 3d model-based vision," *in Proc. of DARPA Image Understanding Workshop*, pp. 94-104,DARPA, 1983.

[Gra89] V. Graefe, "Dynamic Vision System for Autonomous Mobile Robots," *Proc. IEEE Int. Conf. Intelligent Robots and Systems*, pp. 12-23 1989.

[Gra92a] V. Graefe, "Vision for Autonomous Mobile Robots," *Proc. IEEE Workshop Advanced Motion Control*, pp. 57-64, 1992.

[Gra92b] V. Graefe, "Driverless Highway Vehicles," Proc. Int. Hi-Tech Forum, pp. 86-95, 1992.

[Gra93] V. Graefe, "Vision for Intelligent Road Vehicles," *Proc. IEEE Symp. Intelligent Vehicles, pp. 135-140, 1993.*

[Gri90a] W.E.L. Grimson, "Object Recognition by Computer: the role of geometric constraints," The MIT Press, Cambridge, Massachusetts, USA, 1990.

[Gri90b] W.E.L. Grimson, "The combinatorics of object recognition in cluttered environments using constrained search," *J. of Artificial Intelligence, 44*, pp. 121-165, 1990.

[GS96] J.S. Gutmann and C. Schlegel, "AMOS: Comparison of scan matching approaches for self-localization in indoor environments," *in Proc. EUROBOT '96, 1st European Workshop on Advanced Mobile Robots. IEEE Computer Society*, 1996.

[GS99] J. Gasos, and A. Saffotti, "Integrating fuzzy geometric maps and topological maps for robot navigation," *in Proc. of the 3rd Int. ICSC Symposium on Soft Computing (SOCO'99)*, pp. 754-760, 1999.

[Guz69] A. Guzman, "Decomposition of a visual scene into three dimensional bodies. In A. Grasseli, editor, Automatic Interpretation and Classification of Images," *Academic Press*, New York, 1969.

[HGC92] R.I. Hartley, R. Gupta and T. Chang, "Stereo from uncalibrated cameras," *in Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 761-764, 1992.

[HH89] C. Hansen and T.C. Henderson, "CAD based computer vision," *IEEE Trans. Pattern Anal. and Machine Intell.*, 11(11), pp. 1181-1193, November 1989.

[HHC96] S. Hutchinson, G.D. Hager and P. I. Corke, "A tutorial on visual servo control," *IEEE Trans. Robotics and Automation,* 12(5), pp. 651-669, 1996.

[HK88] R. Hinkel and T. Knieriemen, "Environment perception with a laser radar in a fast moving robot," *Symposium on Robot Control,* pp. 68.1-68.7, 1988.

[HK95] E. Huber and D. Kortenkamp, "Using stereo vision to pursue moving agents with a mobile robot," *IEEE Int. Conf. on Robotics and Automation*, May 1995.

[HK97] J. Hertzberg and F. Kirchner, "Landmark based autonomous navigation in sewerage pipes," *in Proc. EUROBOT '97, 2nd European Workshop on Advanced Mobile Robots,* 1997.

[HK98] E. Huber and D. Kortenkamp, "A behavior based approach to active stereo vision for mobile robots," *Engineering Applications of Artificial Intelligence, vol.11*, pp. 229-243,1998.

[HLM02] J.B. Hayet, F.Lerasle and M.Davy, "A Visual Landmark Framework for Indoor Mobile Robot Navigation," *in Proc. of Int. Conf. on Robotics and Automation*, Washington, 2002.

[HM92] K. Hughes and R. Murphy, "Ultrasonic robot localization using dempster-shafer theory," *Neural and Stochastic Methods in Image and Signal Processing*. SPIE vol. 1766, pp. 2-11, 1992.

[HORATIO] Libraries for Vision Applications, P.F. McLauchlan, University of Surrey, 1995

[HP87] C.G. Harris and J. M. Pike, "3D positional integration from image sequences," *in Proc. 3rd Alvey Vision Conf.,* Cambridge, pp. 233-236, 1987.

[HPRSF99] A. Hauck, Georg Passig, J. Rüttinger, M.Sorg, and G. Färber, "Biologically Motivated Hand-Eye Coordination for the Autonomous Grasping of Unknown Objects," *Autonome Mobile Systeme*, pp. 295-300, November 1999.

[HRSF99] A. Hauck, J. Rüttinger, M. Sorg, and G. Färber, "Visual Determination of 3D Grasping Points on Unknown Objects with a Binocular Camera System," *in Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'99)*, October 1999

[HS88] C.G. Harris and M. Stephens, "A combined corner and edge detector," *in Proc. 4th Alvey Vision Conf.,* Manchester, pp. 147-151, 1988.

[HS90] C. Harris and C. Stennett. " "RAPID" a video rate tracker," *Proceedings of the BMVC*, pp. 73-77, 1990.

[HU88] D.P. Huttenlocher and S. Ullman. "Recognizing solid objects by alignment," *in Proc. of the Image Understanding Workshop*, pp. 1114-1124. DARPA, April 1988.

[HW96] J.J. Heuring and D.W. Murray, "Visual head tracking and slaving for visual telepresence," *in Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 515-524,1996.

[IB96] M. Isard and A. Blake, "Contour tracking by stochastic propagation of conditional density," *in Proc. European Conf. on Computer Vision*, pp. 343-356, 1996.

[IK88] K. Ikeuchi and T. Kanade, "Automatic generation of object recognition programs," *in Proc. of the IEEE*, 76(8), pp. 1016-1035, August 1988.

[ILW96] P.T. Im, W.S. Lee, and M. Wingate, "A progressive fuzzy clustering approach for the detection of linear boundary and corners", *Australian Journal of Intelligent Information Processing Systems*, Vol.3 No3 pp. 42-58 Australia, Spring 1996.

[Im97] P.T. Im, "An Enhanced Progressive Fuzzy Clustering Approach To Pattern Recognition," *Ph.D Thesis, Victoria University*, Australia, 1997.

[IQWH95] P.T. Im, B. Qiu, M. Wingate and L. Herron, "Linear boundary detection by cluster prototype centring based on fuzzy memberships", *Australia New Zealand Intelligent Information Processing Conf.*, Perth, Australia, pp. 210-212, 1995.

[IQWH95b] P.T. Im, B. Qiu, M. Wingate and L. Herron, "Implementing a neural network for a progressive fuzzy clustering algorithm", *IEEE Int. Conf. on Neural Networks*, Perth, Australia, vol. III, pp. 1369-1372, 1995.

[IWQ95] P.T. Im, M. Wingate, and B. Qiu, "A cluster prototype centring by membership algorithm for fuzzy clustering", *2nd Asian Conf. on Computer Vision*, Singapore, vol. 2, pp. 67-70, 1995.

[Jar94] R A Jarvis, "Colour Edge Based Quad Vision Ranging," *Australian Pattern Recognition Society's Workshop on Colour Imaging and Applications* Canberra, pp. 83-87, 1994

[Jens00] P. Jensfelt, D. Austin, O. Wijk and M. Anderson, "Feature based condensation for mobile robot localization," *in Proc. of the IEEE Int. Conf. on Robotics and Automation,* (ICRA), pp. 2531-2537, 2000.

[JF93] "Three-Dimensional Object Recognition Systems," *Jain and Flynn (Editors), Elsevier, 1993.*

[JFN97] M. Jagersand, O Fuentes, R. Nelson, "Experimental Evaluation of Uncalibrated Visual Servoing for Precision Manipulation," *in Proc. of Int. Conf. on Robotics and Automation,* pp. 2874-2880, *1997.*

[JGCWS97] J. Janet, R. Gutierrez-Osuna, T. Chase, M. White and J. Sutton, "Autonomous mobile robot self-localization using Kohonen and region-feature neural networks," *Journal of Robotic Systems*, 14(4), pp. 263–282,1997.

[JPT95] T.M. Jochem, D.A. Pomerleau and C.E. Thorpe, "Vision guided lane transition," *IEEE Symposium on Intelligent Road Vehicles*, Detroit, MI, pp. 30-35,1995.

[JW97] C. Jones, M. and Wingate, "Robotic Harvesting in a Simulated Environment," *3rd Int. Conf. on Modelling and Simulation*, Melbourne, pp. 363 - 367, Oct. 1997.

[Kal60] R.E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME Journal of Basic Engineering,* pp. 35-45, March 1960.

[KB61] R.E. Kalman and R.S. Busy, "New results in linear filtering and prediction theory," *Transactions of the ASME Journal of Basic Engineering,* pp. 95-108, March 1961.

[KB91] B. Kuipers and Y. Byun, "A robot exploration and mapping strategy based on a semantic hierarchy of spatial representation," *Robotics and Autonomous Systems 8*, pp. 47-63, 1991

[KBM98] D. Kortenkamp, R.P. Bonasso and R. Murphy, "Artificial Intelligence and Mobile Robots: Case Studies of Successful Robot Systems," Cambridge, MA: MIT Press, 1998.

[KG96] S. Koenig, R. Goodwin and R. Simmons, "Robot navigation with Markov models: A framework for path planning and learning with limited computational resources," *Dorst, L., van Lambalgen, M., and Voorbraak, L., eds., Lecture Notes in Artificial Intelligence, Volume 1093.* Springer, Berlin. pp. 322-337, 1996.

[KHOROS] Khoros: Commercial image processing software package.

[KHS95] E. Krotkov, M. Hebert and R. Simmons, "Stereo perception and dead reckoning for a prototype lunar rover," *Autonomous Robots, 2(4)*, pp. 313-331, 1995.

[KK92] A. Kosaka and A.C. Kak, "Fast Vision-Guided Mobile Robot Navigation Using Model-Based Reasoning and Prediction of Uncertainties," *Computer Vision, Graphics, and Image Processing - Image Processing, vol. 56, no. 3,* pp. 271-329, *1992.*

[Koh93] T. Kohonen, "Self-Organization and Associative Memory," 3rd ed. Springer, Berlin, 1993.

[Kor93] D. Kortenkamp, "Cognitive Maps for Mobile Robots: A Representation for Mapping and Navigation," *Ph.D. Dissertation*, Department of Electrical Engineering and Computer Science, University of Michigan, 1993.

[Kos93] A. Koschan, "What is New in Computational Stereo Since 1989: A Survey on Current Stereo Papers," *Technical Report 93-22,* Technical University of Berlin, Department of Computer Science, August 1993.

[Kos93b] A. Koschan, "Chromatic block matching for dense stereo correspondence," *in Proc. 7th Int. Conf. on Image Analysis and Processing 7ICIAP,* Capitolo, Monopoly, Italy, pp. 641-648, 1993.

[KS96] S. Koenig and R. Simmons, "Unsupervised learning of probabilistic models for robot navigation," *in Proc. of the IEEE Int. Conf. Robotics and Automation*, pp. 2301-2308,1996.

[KS98] S. Koenig and R. Simmons, "Xavier: A robot navigation architecture based on partially observable markov decision process models," *Kortenkamp, D., Bonnasso, R., and Murphy, R., eds., Artificial Intelligence Based Mobile Robots: Case Studies of Successful Robot Systems.* MIT Press, 1998.

[Kur96] A. Kurz, "Constructing maps for mobile robot navigation based on ultrasonic range data," *IEEE Transactions on Systems, Man and Cybernetics | Part B: Cybernetics 26(2),* pp. 233-242, 1996.

[KW90] S. King and C. Weiman, "HELPMATE autonomous mobile robot navigation system," *in Proc. of the SPIE Conf. on Mobile Robots 2352*, pp. 190-198 (1990).

[KW94] D. Kortenkamp and T. Weymouth, "Topological mapping for mobile robots using a combination of sonar and vision sensing," *in Proc. of the Twelth National Conf. on Artificial Intelligence*, pp. 979-984, 1994.

[KWN99] C. Kunz, T. Willeke and I. Nourbakhsh, "Automatic mapping of dynamic office environments," *Autonomous Robots*, pp. 131-142,1999.

[LBLH02] F. Lerasle, M. Briot, C. Lemaire, and J.B. Hayet. "A Smart Sensor based Visual Landmarks Detection for Indoor Robot Navigation," *in Proc. of Int. Conf. on Pattern Recognition*, Quebec, 2002.

[LD92a] J.J. Leonard and H. Durrant-Whyte, "Directed Sonar Sensing for Mobile Robot Navigation," Kluwer Academic Publishers, 1992.

[LD92b] J. J. Leonard, H. Durrant-Whyte and I.J. Cox, "Dynamic map building for an autonomous mobile robot," *Int. Journal of Robotics Research, 11(4),* pp. 286-298, 1992.

[Lee95] D.C. Lee, "The Map-Building and Exploration Strategies of a Simple, Sonar-Equipped, Mobile Robot, An Experimental, Quantitative Evaluation," *Ph.D. Dissertation,* University College London, 1995.

[Lee96] W.Y. Lee, "Spatial Semantic Hierarchy for a Physical Robot," *Ph.D. Dissertation,* The University of Texas at Austin, 1996.

[Leo90] J.J. Leonard, "Directed Sonar Sensing for Mobile Robot Navigation," *PhD thesis, University of Oxford*, 1990.

[Li96] F. Li. "Active Stereo for AGV Navigation," *PhD thesis,* University of Oxford, 1996.

[Lin88] R. Linsker, "Self-organization in a perceptual network," *IEEE Computer 21(3)*, pp. 105-117, 1988.

[LL90] T. Levitt and D. Lawton, "Qualitative navigation for mobile robots," *Artificial Intelligence 44*, pp. 305-360, 1990.

[LL94] M.F. Land and D.N. Lee, "Where we look when we steer," *Nature, 369*, pp. 742-744, 1994.

[LM97] F. Lu and E. Milios, "Robot pose estimation in unknown environments by matching 2D range scans," *Intelligent and Robotics Systems 18*, pp. 249-275, 1997.

[LNH98] T.M. Lee, U. Nehmzow, and R. Hubbold, "Mobile robot simulation by means of acquired neural network models," *European Simulation Multiconference*, pp. 465-469,1998.

[LOLA95] "LOLA:. Visual Object Detection and Retrieval," *A project undertaken at the Centre for Robotics and Intelligent Machines,* NC State University USA. Also, IJCAI-95 Robot Competion and Exhibition, Montreal, Canada. 1995.

[Lon81] H.C. Longuet-Higgins, "A computer algorithm for reconstructing a scene from two projections," *Nature, 293,* pp. 133-135, 1981.

[Low85] D.G. Lowe, "Perceptual Organization and Visual Recognition," *Chapter7, Kluwer, Boston*, 1985.

[Low87] D.G. Lowe, "Three dimensional object recognition from single two dimensional images," *J. of Artificial Intelligence, 31*, pp. 355-395, 1987.

[Low91] D.G. Lowe, "Fitting parameterized three dimensional models to images," *IEEE Trans. Pattern Anal. and Machine Intell., 13(5)*, pp. 441-450, May 1991.

[LR97] D. Lee and M. Recce, "Quantitative evaluation of the exploration strategies of a mobile robot," *Int. Journal of Robotics Research 16(4)*, pp. 413-447, 1997.

[LRH94] D. Langer, J.K. Rosenblatt and M. Hebert, "A behaviour based system for off road navigation," *IEEE Trans. Robotics and Automation, 10(6)*, pp. 776-782, 1994.

[LSL98] G. Li, B. Svensson and A. Lansner, "Self-orienting with on-line learning of environmental features," *Adaptive Behaviour 6(3/4)*, pp. 535-566, 1998.

[Lun91] A. Lundquist. "Tracking lines using a rigid motion assumption," *Master's thesis*, Royal Institute of Technology, Report no. TRITANA P9104, Jan 1991.

[LW88] S. Lu and A. K. C. Wong, "Analysis of 3D scene with partially occluded objects for robot vision," *in Proc. of the 9th Int. Conf. on Pattern Recognition, Int. Association for Pattern Recognition,* The Computer Society Press, pp. 303-308, November 1988.

[MAFMAKHO99] T. Matsui, H. Asoh, J. Fry, Y. Motomura, F. Asano, T. Kurita, I. Hara, and N. Otsu, "Integrated natural spoken dialogue system of JIJO-2 mobile robot for office services," *Proceedings of Sixteenth National Conference on Artificial Intelligence (AAAI-99)*, pp. 621-627, Florida, July, 1999.

[Man82] J. Manyika. "An Information Theoretic Approach to Data Fusion and Sensor Management," *PhD thesis*, University of Oxford, 1993.

[Mar82] D. Marr, "Vision," MIT Press, Cambridge MA, 1982.

[Mat90] M. Mataric, "Navigating with a rat brain: A neuro-biologically inspired model for robot spatial representation," *From Animals to Animats* 1, pp. 169-175,1990.

[May90] P.S. Maybeck, "The Kalman filter: An introduction to concepts," *Cox, I., and Wilfong, P., eds., AI-based Mobile Robots: Case Studies of Successful Robot Systems,* Springer, Berlin, pp. 194-204, 1990.

[McL92] P.F. McLauchlan, "Horatio: Libraries for vision applications," *Technical Report OUEL 1967/92*, Dept. Engineering Science, University of Oxford, October 1992.

[MD94] P. MacKenzie and G. Dudek, "Precise positioning using model-based maps," *in Proc. of the Int. Conf. on Robotics and Automation, Los Alamitos, CA:* IEEE Computer Society Press, pp. 1615-1621, 1994.

[ME85] H. Moravec and A. Elfes, "High resolution maps from wide angle sonar," *in Proc. of the IEEE Int. Conf. on Robotics and Automation.* St. Louis, Missouri: IEEE Computer Society Press, pp. 116-121, 1985.

[MF92] S.J. Maybank and O. Faugeras, "A theory of self-calibration of a moving camera," *Int. Journal of Computer Vision, 8(2),* pp. 123-151, 1992.

[MF96] F. Mura and N. Franceschini, "Obstacle avoidance in a terrestrial mobile robot provided with a scanning retina," *in Proc. of the 1996 Intelligent Vehicles Symposium,* pp. 47-52, 1996.

[MK93a] M. Meng and A.C. Kak, "NEURO-NAV: A Neural Network Based Architecture for Vision-Guided Mobile Robot Navigation Using Non-Metrical Models of the Environment." *in Proc. IEEE Int. Conf. Robotics and Automation,* vol. 2, pp. 750-757, 1993.

[MK93b] M. Meng and A.C. Kak, "Mobile Robot Navigation Using Neural Networks and No metrical Environment Models," *IEEE Control Systems,* pp. 30-39, Oct. 1993.

[MM96] P.F. McLauchlan and D.W. Murray, "Active camera calibration for a head-eye platform using a variable state dimension filter," *PAMI,* pp. 15-22, 1996.

[MM95] P.F. McLauchlan and D.W. Murray, "A unifying framework for structure and motion recovery from image sequences," *in Proc. 5th Int. Conf. on Computer Vision, Boston.* IEEE Computer Society Press, pp. 314-320, 1995.

[MMD95] V. Morellas, J. Minners and M. Donath, "Implementation of real time spatial mapping in robotic systems through self-organizing neural networks," *in Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems.* pp. 277-284,1995.

[MMS93] D.W. Murray, P.F. McLauchlan, I.D. Reid and P.M. Sharkey, "Reactions to peripheral image motion using a head/eye platform," *in Proc. 4th Int. Conf. on Computer Vision, Berlin, Los Alamitos, CA,* IEEE Computer Society Press , pp. 403-411, 1993.

[Mor80] H. Moravec. "Obstacle avoidance and navigation in the real world by a seeing robot rover," *Technical Report CMURITR3, Carnegie* Mellon University, Robotics Institute, September 1980.

[MRD96] D.W. Murray, I.D. Reid and A.J. Davison, "Steering and navigation behaviours using fixation," *in Proc. 7th British Machine Vision Conf.,* Edinburgh, pp. 635-644, 1996.

[MW02] H. Moyssidis and M.Wingate, "A Speech Guided Robot Controller with 3D Graphical Positioning," *4th Int. on Conf. Modelling and Simulation, Australia,* pp. 428-433, *Nov 2002.*

[MZC92] J. Mayhew, Y. Zhang and S. Cornell, "The adaptive control of a four degrees of freedom stereo camera head," *Phil. Trans. Royal Society London B, 337,* pp. 315-326, 1992.

[Neh92] U. Nehmzow, "Experiments in Competence Acquisition for Autonomous Mobile Robots," *Ph.D. Dissertation, Department of Artificial Intelligence,* University of Edinburgh, 1992.

[Neh97] U. Nehmzow, "Scientific methods in mobile robotics," *in Proc. of TIMR'97, Towards Intelligent Mobile Robots, Manchester*, Department of Computer Science, Manchester University, Technical Report Series, ISSN 1361-6161, Report UMCS-97-9-1, 1997.

[New99] P.M Newman, "On the Structure and Solution of the Simultaneous Localisation and Map Building Problem," *Ph.D Thesis* Oxford University 1999.

[Nie85] L. Nielsen, "Simplifications in Visual Servoing," *Ph.D thesis*, Department of Automatic Control, Lunds Institute of Technology, September 1985.

[Nil80] N. Nilsson, "Principles of Artificial Intelligence," *Palo Alto*, CA: Tioga. 1980.

[Nou98] I. Nourbakhsh, "Dervish: An office-navigating robot," *Kortenkamp, D., Bonasso, R., and Murphy, R., eds., AI-based Mobile Robots: Case Studies of Successful Robot Systems,* Cambridge, MA: MIT Press, pp. 73-90, 1998.

[NS92] U. Nehmzow and T. Smithers, "Using motor actions for location recognition," *Varela, F., and Bourgine, P., eds., Toward a Practice of Autonomous Systems.* MIT Press, Cambridge MA, 1992.

[NSH91] U. Nehmzow and T. Smithers, and J. Hallam, "Location recognition in a mobile robot using self-organising feature maps," *In Schmidt, G., ed., Information Processing in Autonomous Mobile Robots. Springer-Verlag*, 1991.

[OHD97] S. Oore, G.E. Hinton and G. Dudek, "A mobile robot that learns its place," *Neural Computation 9*, pp. 683-699, 1997.

[ON97] C. Owen, and U. Nehmzow, "Middle scale navigation - a case study," *in Proc. AISB 97 Workshop on Spatial Reasoning in Animals and Robots*. Department of Computer Science, Manchester University, Technical Report Series, ISSN 1361-6161, Report UMCS-97-4-1, 1997.

[OUV98] G. Oriolo, G. Ulivi and M. Vendittelli, "Real-time map building and navigation for autonomous robots in unknown environments," *IEEE Transactions on Systems, Man and Cybernetics 28(3)* pp. 316-333, 1998.

[Owe97] R. Owens, "Lecture Series, Computer Vision Representations," *CVonline/Owens/LECT13 1997*

[PBF90] J.M. Pichon, C. Blanes and N. Franceschini. "Visual guidance of a mobile robot equipped with a network of self motion sensors," *SPIE Conf. on Mobile Robots 4,* pp. 44-53, 1990.

[Pen87] A.P. Pentland. "Recognition by parts," *in Proc. 1st Int. Conf. on Computer Vision*, pp. 612-620,1987.

[PJ96] D.A. Pomerleau and T. Jochem, "Rapidly Adapting Machine Vision for Automated Vehicle Steering," *IEEE Expert, Intelligent Systems and Their Applications, vol. 11, no.2,* Apr., pp. 19-27, 1996.

[PM88] H.D. Park and O. R. Mitchell, "CAD based planning and execution of inspection," *in Proc. of the Computer Vision and Pattern Recognition Conf., The Computer Society, The Computer Society Press, pp. 858-863,* June 1988.

[PND96] D. Pagac, E.M. Nebot and H. Durrant-Whyte, "An evidential approach to probabilistic map building," *Reasoning with Uncertainty in Robotics, number 1093 in Lecture Notes in AI*, pp. 164-170. Springer, 1996.

[Pom89] D.A. Pomerleau, "ALVINN: An Autonomous Land Vehicle in a Neural Network," *Technical Report CMU-CS-89-107*, Carnegie Mellon Univ., 1989.

[Pom89] D.A. Pomerleau, "ALVINN: An autonomous land vehicle in a neural network", In Touretzky, D., ed., *Advances in Neural Information Processing Systems 1.* Morgan Kaufman, 1989.

[Pom93] D.A. Pomerleau, "Neural Network Perception for Mobile Robot Guidance" Kluwer Academic Publishers, 1993.

[Pom94] D.A. Pomerleau, "Reliability Estimation for Neural Network Based Autonomous Driving," Robotics and Autonomous Systems, vol. 12, pp. 113-119, 1994.

[Pop94] A.R. Pope, Model-Based Object Recognition, "A survey of recent research," *Technical Report 94-04,* January 1994.

[PPKK95] J. Pan, D.J. Pack, A. Kosaka, and A.C. Kak, "FUZZY-NAV: A Vision-Based Robot Navigation Architecture Using Fuzzy Inference for Uncertainty-Reasoning," *Proc. IEEE World Congress Neural Networks*, vol. 2, pp. 602-607, July 1995

[PPM90] Pollard, S.B., J. Porrill, and J.E.W. Mayhew, "Experiments in vehicle control using predictive feed-forward stereo," *IVC(8),* pp. 63-70, 1990,

[PPM91] S.B. Pollard, J. Porrill and J.E. Mayhew, "Recovering partial 3D wire frames descriptions from stereo data," *Image and Vision Computing*, vol 9 No 1 pp. 58-65, 1991.

[PPMF87] S.B. Pollard, J. Porrill, J.E. Mayhew and J.P Frisby, "Matching geometrical descriptions in three-space," *Image and Vision Computing*, vol 5 No 2, pp. 73-78, 1987.

[PPPBMF87] Porrill, J, Pollard, S B Pridmore, T P Bowen, J, Mayhew, J E W and Frisby J P, "TINA: the Sheffield AIVRU vision System," *IJCAI9*, Milan, Italy 1987, pp. 1138-1144.

[PPPMF88] S.B. Pollard, J. Porrill, T.P. Pridmore, J.E.W. Mayhew and J.P. Frisby, "Components of a stereo vision system," *Robert Bolles and Bernard Roth, editors, Robotics Research, pp. 229-235, Cambridge, Massachusetts, USA, August 1988. National Science Foundation and System Development Corporation,* The MIT Press.

[PPT99] S.Pollard, J. Porrill, N. Thacker, "Tina Programmer's Guide," University of Sheffield/Manchester, England, 1999.

[PZ98] P. Pritchett and A. Zisserman, "Wide baseline stereo matching," *in Proc. 6th Int. Conf. on Computer Vision,* Bombay, pp. 754-760, 1998.

[Rab89] L.R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *in Proc. of the IEEE, 77(2)*, pp. 257-286,1989.

[Ram90] V.S. Ramachandran, "Visual perception in people and machines," *A. Blake, editor, A.I. and the Eye,* Chapter 3. Wiley and Sons, 1990.

[RB95] I.D. Reid and P. A. Beardsley, "Self alignment of a binocular robot," *in Proc. 6th British Machine Vision Conf.,* Birimingham, pp. 443-452, 1995.

[RB95b] S.M. Rowe and A. Blake "Statistical background modeling for tracking with a virtual camera," *in Proc. 6th British Machine Vision Conf., Birimingham*, 1995.

[RBFT99] N. Roy, W. Burgard, D Fox, and S. Thrun, "Coastal navigation: Robot navigation under uncertainty in dynamic environments," *in Proc. of the IEEE Int. Conf. on Robotics and Automation*, pp. 35-40, 1999.

[RCE82] D. Reily, L. Cooper and C. Elbaum, "A neural model for category learning," *Biological Cybernetics 45,* pp. 34-51, 1982.

[RD95] J. Racz and A. Dubrawski, "Artificial neural network for mobile robot topological localisation," *Robotics and Autonomous Systems 16*, pp. 73-80, 1995.

[Req80] A.A.G. Requicha, "Representations for rigid solids: Theory, methods, and systems." *Computing Surveys, 12(4),* pp. 437-464, December 1980.

[RG94] U. Regensburger and V. Graefe, "Visual Recognition of Obstacles on Roads," *Proc. IEEE Int. Conf. Intelligent Robots and Systems*, pp. 980-987, 1994.

[RJ86] L.R. Rabiner, B.H. Juang, "An Introduction to Hidden Markov Models," *IEEE Acoustics Speech and Signal Processing ASSP Magazine, ASSP*-3(1): pp. 4-16,. 1986.

[RK97] S. Rougeaux and Y. Kuniyoshi, "Robust real time tracking on an active vision head," *in Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 1-6, 1997.

[RM93] I.D. Reid and D.W. Murray, "Tracking foveated corner clusters using affine structure," *in Proc. 4th Int. Conf. on Computer Vision, Berlin, Los Alamitos, CA,* pp. 76-83, *IEEE Computer Society Press*, 1993.

[RMB94] I.D. Reid, D.W. Murray and K.J. Bradshaw, "Towards active exploration of static and dynamic scene geometry," *IEEE Int. Conf. on Robotics and Automation,* pp. 718-723, San Diego, May 1994.

[Rob65] L.G. Roberts, "Machine perception of threedimensional solids," *J.T. Tippett et al., editor, Optical and ElectroOptical Information Processing, chapter 9,* pp. 159-197. *MIT Press*, 1965.

[Rot96] C.A. Rothwell, "Object Recognition Through Invariant Indexing," *Oxford University Press*, Oxford, 1996

[RPB90] M. Rygol, S. Pollard and C. Brown, "A multiprocessor 3d vision system for pick and place," *in Proc. of the British Machine Vision Conf.,* pp. 169-174, 1990.

[Ruc97] W. Rucklidge, "Efficient guaranteed search for gray level patterns," *in Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 717-723, 1997.

[RV81] A.A.G. Requicha and H. B. Voelcker, "An Introduction to Geometric Modeling and Its Application," *Mechanical Design and Production, volume 8, chapter 5,* pp. 293-328. Plenum Publishing Corporation, 1981.

[RV83] A.A.G. Requicha and H. B. Voelcker, "Solid modeling: Current status and research directions," *Computer Graphics and Applications*, pp. 25-37, October 1983.

[RZFM91] C.A. Rothwell, A. Zisserman, J.L. Munday, and D.A. Forsyth, "Efficient Model Library Access By Projectively Invariant Indexing Functions," *CVPR(92)*, pp. 109-114. 1992.

[SA98] A. Schultz and W. Adams, "Continuous localization using evidence grids," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pp. 2833-2839, 1998.

[Saf97] A. Saffiotti, "The uses of fuzzy logic in autonomous robot navigation: A catalogue raisonne," *Soft Computing 1(4)*, pp. 180-197, 1997.

[SAY99] A. Schultz, W.Adams, and B. Yamauchi, "Integrating Exploration, Localization, Navigation and Planning With a Common Representation," *Autonomous Robots, vol 6 No3,* IEEE Press, pp. 293-308, May 1999.

[SB91] F. Solina and Z. Bjelogrlic, "Methodolgies and techniques for interpretation of 3d range images," *P. Plancke et al., editor, First ESA Workshop on Computer Vision and Image Processing for Spaceborne Applications. European Space Agency, European Space Agency,* June 1991.

[SBK00] O. Schreer, N. Brandenburg, P. Kauff: "A Comparative Study on Disparity Analysis Based on Convergent and Rectified Views," *in Proc. of BMVC 2000, 11. British Machine Vision Conf.,* Bristol, United Kingdom, Sept. 2000.

[SBP97] C. E. Smith, S. A. Brandt, and N. P. Papanikolopoulos, "Eye-In-Hand Robotic Tasks In Uncalibrated Environments," *IEEE Journal of Robotics and Automation, Vol. 13, No. 6,* pp. 903-914, 1997

[SD92] G. Schoner and M. Dose, "A dynamical systems approach to task-level system integration used to plan and control autonomous vehicle motion," *Robotics and Autonomous Systems* 10, pp. 253-267, 1992.

[SD95] A. Stevens, M. Stevens and H. Durrant-Whyte, "`OxNav': Reliable autonomous navigation," *in Proc. of the IEEE Int. Conf. on Robotics and Automation*, pp. 2607-2612, 1995.

[SD98] S. Simhon and G. Dudek, "Selecting targets for local reference frames," *in Proc. of the IEEE Int. Conf. on Robotics and Automation,* pp. 2840-2845, 1998.

[Shat98] H. Shatkay, "Learning Models for Robot Navigation," *Ph.D. Dissertation*, Department of Computer Science, Brown University, 1998.

[Sin99] Virtual Robotics Lab (Programmer P.G Singh). http://members.tripod.com/gurvinder_pec

[SK95] R. Simmons and S. Koenig, "Probabilistic robot navigation in partially observable environments," *in Proc. of the 14th Int. Joint Conf. on Artificial Intelligence*, pp. 1080-1087, 1995.

[SK97] H. Shatkay and L.P. Kaelbling, "Learning topological maps with weak local odometric information," *in Proc. of the 15th Int. Joint Conf. on Artificial Intelligence*, pp. 920-929, 1997.

[Smi92] S. Smith, "A new class of corner finder," *in Proc. 3rd British Machine Vision Conf.,* Leeds, pp. 139-148, 1992.

[SMVM93] P. M. Sharkey, D. W. Murray, S. Vandevelde, I. D. Reid and P. F. McLauchlan, "A modular head/eye platform for real-time reactive vision," *Mechatronics, 3(4)*, pp. 517-535, 1993.

[SN00]A. Selinger and R. C. Nelson, "Improving Appearance-Based Object Recognition in Cluttered Backgrounds," *TR725,* Computer Science Dept., U. Rochester, January 2000.

[SNSDRCCC97] S. Scheding, E. M. Nebot, M. Stevens, H. F. DurrantWhyte, J. Roberts, P. Corke, J. Cunningham and B. Cook, "Experiments in autonomous underground guidance," *in Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 1898-1903, 1997.

[SNWGH99] V. Sequeira, K. Ng, E. Wolfart, J. G. M. Goncalves, and D. Hogg, "Automated reconstruction of 3d models from real world environments," *ISPRS Journal of Photogrammetry and Remote Sensing*, no. 54, pp. 1–22, 1999.

[Sor85] H.W. Sorenson, editor, "Kalman Filtering: Theory and Application," *IEEE Press 1985.*

[SSC90] R. Smith, M. Self and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," *Cox, I., and Wilfong, P., eds., AI-based Mobile Robots: Case Studies of Successful Robot Systems.* Springer, Berlin. pp. 167-193, 1990.

[SSD95] A. Stevens, M. Stevens and H. F. Durrant-Whyte, "OxNav: Reliable autonomous navigation," *in Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 2607-2612, 1995.

[ST94] J. Shi and C. Tomasi. "Good features to track," *in Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 593-600, 1994.

[Sto96] H.W. Stone. "Mars pathfinder micro-rover: A low cost, low power spacecraft," *in Proc. of the 1996 AIAA Forum on Advanced Developments in Space Robotics,* Madison, WI., 1996.

[SWG97] E. Shilat, M. Werman and Y. Gdalyahu, "Ridge's corner detection and correspondence," *in Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition,* pp. 976- 982, 1997.

[SZ97] C. Schmid and A. Zisserman, "Automatic line matching across views," *in Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 666-672, 1997.

[SZ00] Y. Shan and Z. Zhang, "Corner Guided Curve Matching and its Application to Scene Reconstruction," *IEEE Conf. Computer Vision and Pattern Recognition* (CVPR'00), South Carolina, Vol.I, pp. 796-803, June 2000.

[TBBCDFHRRSS99] S. Thrun, M. Benewitz, W. Burgard, A.B. Cremers, F. Dellaert, D. Fox, D. Haehnel, C. Rosenberg, N. Roy, J. Schulte and D. Schulz, "MINERVA: A second generation museum tour-guide robot," *in Proc. of the IEEE Int. Conf. on Robotics and Automation*, 1999.

[TBBFFHHKS98a] S. Thrun, A. Bucken, W. Burgard, D. Fox, T. Frohlinghaus, D. Henning, T. Hofmann, M. Krell and T. Schmidt, "Map learning and high-speed navigation in RHINO", *Kortenkamp, D., Bonasso, R., and Murphy, R., eds., AI-based Mobile Robots: Case Studies of Successful Robot Systems. Cambridge*, MA: MIT Press, 1998.

[TBF98b] S. Thrun, W. Burgard and D. Fox, "A probabilistic approach to concurrent mapping and localisation for mobile robots," *Machine Learning 31(5)*, pp. 1-25, 1998.

[TC92]D.C. Tseng and C.H Chang, "Color Segmentation Using Perceptual Attributes," *Proc. 11th Conf. Pattern Recognition*, vol. 3, pp. 228-231, 1992.

[TFB00] S. Thrun, D. Fox and W. Burgard, "Monte Carlo localization with mixture proposal distribution," *in Proc. of the National Conf. on Artificial Intelligence*, 2000.

[TFZ98] P. H. S. Torr, A. W. Fitzgibbon and A. Zisserman, "Maintaining multiple motion model hypotheses over many views to recover matching and structure," *in Proc. 6th Int. Conf. on Computer Vision,* Bombay, pp. 485-491, 1998.

[TH84] R.Y. Tsai and T.S. Huang, "Uniqueness and estimation of three dimensional motion parameters of rigid objects with curved surfaces," *IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI6*, pp. 13-27, 1984.

[Tha99] N.Thacker, "Tina Algorithms's Guide University of Sheffield/Manchester, England, 1999.

[THK88] C. Thorpe, M.H Herbert, T. Kanade, and S.A. Shafer, " Vision and Navigation for the Carnegie-Mellon Navlab," *IEEE Tran. Pattern Analysis and Machine Intelligence*, vol. 10, no. 3, pp. 362-372, May 1988.

[THO93] J.I. Thomas, A. Hanson and J. Oliensis, "Understanding noise: The critical role of motion error in scene reconstruction," *in Proc. 4th Int. Conf. on Computer Vision*, Berlin, pp. 691-695, 1993.

[THO94] J.I. Thomas, A. Hanson and J. Oliensis, "Refining 3d reconstructions: A theoretical and experimental study of the effectt of cross-correlation," *Computer Vision, Graphics, and Image Processing*, 60(3), pp. 359-370, 1994.

[Thr98a] S. Thrun, "Bayesian landmark learning for mobile robot localisation," *Machine Learning 33(1)*, pp. 41-76,1998.

[Thr98b] S. Thrun, "Learning metric-topological maps for indoor mobile robot navigation," *Artificial Intelligence 99*, pp. 21-71,1998.

[Tin99] S.Pollard, J. Porrill, N. Thacker, "Tina Programmer's Guide," University of Sheffield/Manchester, England, 1999.

[Tin99a] N.Thacker, "Tina Algorithms's Guide," University of Sheffield/Manchester, England, 1999.

[Tin97b] N.Thacker,T. Lacey, D. Prendergast, "Tina User's Guide," University of Sheffield/Manchester, England, 1999.

[TJ] TargetJr - A C++ Computer Vision Environment - C++ programming environment with libraries to support: image processing; image segmentation; camera modeling; 2-d and 3D geometry; a graphical user interface based on FRESCO. TargetJr has been developed over the last 10 years, starting at GE's Corporate R&D Center. Currently TargetJr is used by a number of vision research groups with emaphasis on geometric algorithms and object recognition. TargetJr is written in C++ and organized into a number of libraries including: numerics; spatial objects; image; image processing; segmentation; computational geometry; 3D modeling; and user interface. *(by* Joseph Mundy, William Hoffman, Andrew Fitzgibbon, Peter Vanroose and Rupert Curwen / *GE Corp. R&D,* Oxford University, University of Leuven

[TK92] C. Tomasi and T. Kanade, "Shape and motion from image streams under orthography: A factorization approach," *Int. Journal of Computer Vision, 9(2),* pp. 137-154, 1992.

[TK95] C.J. Taylor and D. Kriegman. "Structure and motion from line segments in multiple images," *IEEE Trans. On Pattern Analysis and Machine Intelligence*, Vol. 17 No. 11, pp. 1021-1033, November 1995

[TKS87] C. Thorpe, T. Kanade, and S.A. Shafer, "Vision and Navigation for the Carnegie-Mellon Navlab," *Proc. Image Understand Workshop*, pp. 143-152, 1987.

[TLP99] N.Thacker,T. Lacey, D. Prendergast, "Tina User's Guide," University of Sheffield/Manchester, England, 1999.

[TO93] J.I. Thomas and J. Oliensis, "Automatic position estimation of a mobile robot," *IEEE Conf. on AI Applications*, pp. 438-444, 1993.

[Tsa87] Roger Y. Tsai, "A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses," *IEEE Journal of Robotics and Automation* RA-3(4): pp. 323-344, August 1987.

[TZ98] P.H.S. Torr and A. Zisserman, "Robust computation and parameterization of multiple view relations," *in Proc. 6th Int. Conf. on Computer Vision,* Bombay, pp. 727-732, 1998.

[Ull96] S. Ullman, "High-level vision: object recognition and visual cognition," *MIT Press, 1996 TR2732, Computer Vision Laboratory, Center for Automation Research*, University of Maryland, Aug 1996.

[VXL01] VXL - C++ libraries for Computer Vision - The Vision-something-Libraries are a collection of C++ libraries designed for computer vision research. It was created from TargetJr and the Image Understanding Environment (IUE) with the aim of making a lighter, faster and more consistent system. VXL is written in ANSI/ISO C++ and is designed to be portable over many platforms. It is developed and used by a consortium including groups from the Universities of Leuven, Oxford, Manchester, and RPI, GE CRD. *(* VXL Consortium).

[Wal72] D.I. Waltz, "Generating semantic descriptions from drawings of scenes with shadows," *PhD thesis,* AI Lab, MIT, 1972.

[WD02a] M.Wingate and A.J.Davison, "3D Recognition of Remote Objects Utlizing Stereo Vision on a Roving Platform," *ACCV2002,* pp. *682-688, Australia, January 2002.*

[WD02b] M.Wingate and A.J.Davison, "Object Retrieval using an Active Stereo Head and a Robot Arm," *4th Int. Conf. on Modeling and Simulation,* Australia, pp. 417-422, *Nov 2002.*

[Weh96] R. Wehner, "Middle-scale navigation: The insect case," *Journal of Experimental Biology*, 1996.

[Wei93] I. Weiss, "Geometric invariants and object recognition," *Int. Journal of Computer Vision*, 10(3), pp. 207-231,1993.

[WF97] J. Weng, T.S. Huang, and N. Ahuja, "Motion and structure from two perspective views: Algorithms, error analysis and error estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 11(5), pp. 451-476, 1989.

[WHA89] H. Wang and J.M. Brady. "Corner detection for 3D vision using array processors," *in Proc. BARNAIMAGE91,* Barcelona, Springer-Verlag, 1991.

[Win97a] M. Wingate, "3D Structure Recovery of Objects Using Line Features in Multiple Camera Views" 3rd *Int. Conf. on Modelling and Simulation,* Melbourne, pp. 386-391, Oct. 1997.

[Win97b] *M.* Wingate, "3D Structure Recovery of Rigid Objects Using Pan-Tilt-Vergence Stereo," *DICTA '97 Conf.* Auckland, N.Z. pp. 83-88, Dec 1997.

[Win99] *M.* Wingate, "Structure Recovery of Objects Using Multiple Camera Views," *Post Graduate Seiminar Presentation,* School of Electrical Engineering, Victoria University, 1999.

[Win02] M.Wingate, "3D Modeling for the Recognition of Remote Objects," pp. 405-410, *4th Int. on Conf. Modelling and Simulation,* Australia, Nov 2002.

[WMMW97] P. Whaite and F.P. Ferrie, "Autonomous exploration: Driven by uncertainty," *IEEE Transactions on Pattern Analysis and Machine Intelligence, 19(3),* pp. 193-205, 1997.

[WS96] M. Wingate and A. Stoica, "A Step Towards Robot Apprectices- Visual Learning," *in Proc. of the IASTED Int. Conf. Robotics and Manufacturing*, pp. 186-189, 1996.

[WS97] M. Wingate and A. Stoica, "An Application of Fuzzy Neurons to Visual Learning," *in Proc. of the MS'97 Int. Conf. On modelling and Simulation,* pp. 45-50, 1997.

[WTRM98] S. Waldherr S. Thrun, R. Romero and D. Margaritis, "Template-Based Recognition of Pose and Motion Gestures On a Mobile Robot," *AAAI/IAAI*, pp. 977-982, 1998.

[Wv95] G. Weiss and E. von Puttkamer, "A map based on laser scans without geometric interpretation," *in Proc. of Intelligent Autonomous Systems 4, Karlsruhe,* Germany, pp. 403-407, 1995.

[WWR93] S. W.Wijesoma, D. F. H. Wolfe, R. J. Richards "Eye-to-Hand Coordination for vision guided Robot Control Applications" *Int. J. of Robotics Research*, v 12, No 1, pp. 65-78, 1993.

[Yam97] B. Yamauchi, "A frontier-based approach for autonomous exploration," *in Proc. of the IEEE Int. Symposium on Computational Intelligence in Robotics and Automation*, pp. 146-151, 1997.

[YB96] B. Yamauchi, and R, Beer, "Spatial learning for navigation in dynamic environments," *IEEE Transactions on Systems, Man and Cybernetics Section B* 26(3), pp. 496-505, 1996.

[Yea88] W. Yeap, "Towards a computational theory of cognitive maps," *Artificial Intelligence 34(3)*, pp. 297-360, 1988.

[YL97] B. Yamauchi and P. Langley, P. "Place recognition in dynamic environments," *Journal of Robotic Systems, Special Issue on Mobile Robots* 14(2), pp. 107-120,1997.

[YSA98] B. Yamauchi, A. Schultz and W. Adams, "Mobile robot exploration and map-building with continuous localization," *in Proc. of the 1998 IEEE Int. Conf. on Robotics and Automation, Leuven,* Belgium, pp. 3715-3720, 1998.

[YSAGGP97] B. Yamauchi, A. Schultz W., Adams, K. Graves, J. Grefenstette and D. Perzanowski, "ARIEL : Autonomous robot for integrated exploration and localization," *in Proc. of the Fourteenth National Conf. on Artificial Intelligence,* pp. *804-805*, 1997.

[YZ02] R. Yang and Z. Zhang. "Model-based Head Pose Tracking With Stereovision," In *in Proc. Fifth IEEE Int. Conf. on Automatic Face and Gesture Recognition* (FG2002), pp. 255-260, Washington, DC, May 20-21, 2002.

[Zel92] A. Zelinsky, "Mobile robot exploration algorithm," *IEEE Robotics and Automation 8(6)*, pp. 707-717,1992.

[ZF92] Z. Zhang and O. Faugeras, "3D Dynamic Scene Analysis," Springer-Verlag, 1992.

[ZF94] C. Zeller and O. Faugeras, "Applications of non-metric vision to some visual guided tasks," *Technical Report INRIA Rapport de recherche 2308*, INRIA SophiaAntipolis, 1994.

[ZF94] C.S. Wiles, A. Maki, N. Matsuda and M. Watanabe, "Hyper-patches for 3D model acquisition and tracking," *in Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition,* pp. 1074-1080, 1997.

[Zha02a] Cha Zhang "A Survey on Stereo Vision for Mobile Robots," *Technical Report*, Dept.of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh USA, 2002

[Zha02b] Z. Zhang, "Vision-Based Modeling and Interaction," *in the Fifth Asian Conf. on Computer Vision (*ACCV*)*, Invited paper, Melbourne, Australia, January 2002.

[Zim95a] U.R. Zimmer, "Adaptive Approaches to Basic Mobile Robot Tasks," *Ph.D. Dissertation, University of Kaiserslautern*, 1995.

[Zim95b] U.R. Zimmer, "Self localisation in dynamic environments," *IEE/SOFT Int. Workshop,* 1995.

[Zim96] U.R. Zimmer, "Robust world modeling and navigation in a real world," *Neurocomputing, Special Issue 13* pp. 2-4, 1996.

[Zim97] U.R. Zimmer, "General and specific models in complex robotics systems - a critique and a proposal," *Technical Report 1063,* GMD, 1997.

[ZW02] M.Zulli and M.Wingate, "A Micro Mobile Surveillance System with Wireless Video," *4th Int. Conf. Modelling and Simulation, Australia,* pp. 452-457, Nov 2002.