

**Model Dependent Inference of  
3D Information From  
a Sequence of 2D Images**

**Rakesh Kumar**

**COINS TR92-04**

**February 1992**

**This work has been supported in part by the Defense Advanced Research Projects Agency (via TACOM) under Contract Number DAAE07-91-C-R035, and by the National Science Foundation under Grant Number CDA-8922572.**

**MODEL DEPENDENT INFERENCE OF 3D INFORMATION FROM  
A SEQUENCE OF 2D IMAGES**

A Dissertation Presented

by

**RAKESH KUMAR**

Submitted to the Graduate School of the  
University of Massachusetts in partial fulfillment  
of the requirements for the degree of

**DOCTOR OF PHILOSOPHY**

February 1992

Department of Computer and Information Science

RIGHTS RESERVED BY THE AUTHOR AND PUBLISHERS. ALL RIGHTS RESERVED.

© Copyright by: RAKESH KUMAR 1992

All Rights Reserved

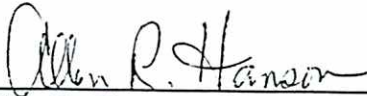
MODEL DEPENDENT INFERENCE OF 3D INFORMATION FROM  
A SEQUENCE OF 2D IMAGES

A Dissertation Presented


by

RAKESH KUMAR

Approved as to style and content by:



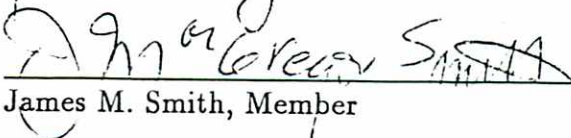
Allen R. Hanson, Chair of Committee



Edward M. Riseman, Member



Roderic Grupen, Member



James M. Smith, Member



W. Richards Adrion, Department Chair  
Computer and Information Science

Dedicated to my parents  
Mitter Sen and Krishna Kumar

His line tracking algorithm was very useful to me for both collecting data and building an overall system. John Oliensis was also present to remind me that what is new in computer vision is not necessarily new in physics. Ross Beveridge provided me with many of the data sets used in my experiments. He was a good colleague to work with. Bob Collins often helped me with technical and programming problems. I had good discussions and interactions with Claude Fennema, Rabi Dutta, Martin Herbordt, Mike Scudder, Mark Snyder, Joe Thomas and Rich Weiss. Laurie Waskiewicz was very helpful and a nice, cheerful spirit to interact with. Janet Turnbull and Val Conti provided good administrative support. Bob Heller was always present to answer any system questions.

I have immensely enjoyed and learned a lot during my stay as a graduate student in the Vision's group at the University of Massachusetts. It is to the credit of Professors Ed Riseman and Al Hanson that they have been able to gather such a diverse set of researchers with interests in almost all aspects of computer vision. There is some fear that the latest round of funding may alter the diverse character of research performed at the Visions group. I hope this does not happen.

I would not have been able to complete my PhD without the *shakti* provided by my wife Renee. She encouraged me to be my best. My sister Aarti and friend Devesh Kapur inspired me to do my PhD. Aarti helped lay the foundations of my education by teaching me basic algebra and geometry. I would like to acknowledge the support and love of my parents. They provided me with an excellent education and gave me the freedom and wealth to roam the world. Finally, thanks to my newborn son Nikhil for teaching me the the value of time and our cat Chewbaka for cuddling with me.

## ACKNOWLEDGEMENTS

I wish to thank Professors Al Hanson and Ed Riseman for their support and advice during my tenure as a graduate student at the University of Massachusetts. Al Hanson was the chair of my committee and I truly enjoyed working with him. He was always very insightful and patient in our many discussions. He painstakingly read and gave me excellent feedback on many drafts of papers, reports, thesis etc.. Ed Riseman was also very inspiring and gave me many good suggestions. He encouraged me to do my best and produce the most robust algorithms. Finally, I would like to thank my other two thesis committee members: Professors Rod Grupen and Jim Smith. Rod Grupen is probably one of the few thesis committee members who has closely examined the internals of the code developed by a PhD student. Jim Smith provided me with a good background on non-linear optimization techniques.

A large part of my thesis was accomplished by debates, discussions and good advice from my friend Harpreet Singh Sawhney. I was lucky that he was finishing his PhD at the same time and his untiring spirit also helped to keep up my momentum. Over the years, he has helped me in a countless number of ways. I will miss our many conversations on almost every topic under the sun. I will also miss him telling me: "That's obvious, Teddy". Harpreet, Vindhi and Dheerja are like family to us.

Raghavan Manmatha was always willing to read drafts of my papers and provided me with excellent feedback and comments. It was also fun to beat him in scrabble. Brian Burns was one of the inspiring vision researchers when I joined the visions group at the University of Massachusetts. He is very knowledgeable about a large range of topics in vision and was a good sounding board. Lance Williams was ever ready to espouse on what is and what isn't the real thing: perceptual organization.

## ABSTRACT

# MODEL DEPENDENT INFERENCE OF 3D INFORMATION FROM A SEQUENCE OF 2D IMAGES

FEBRUARY 1992

RAKESH KUMAR

B.TECH., INDIAN INSTITUTE OF TECHNOLOGY AT KANPUR

M.S., STATE UNIVERISTY OF NEW YORK AT BUFFALO

PH.D., UNIVERSITY OF MASSACHUSETTS

Directed by: Professor Allen R. Hanson

In order to autonomously navigate through a complex environment, a mobile robot requires sensory feedback. This feedback will typically include the 3D motion and location of the robot and the 3D structure and motion of obstacles and other environmental features. The general problem considered in this thesis is how this 3D information may be obtained from a sequence of images generated by a camera mounted on a mobile robot.

The first set of algorithms developed in this thesis are for robust determination of the 3D pose of the mobile robot from a matched set of model and image landmark features. Least-squares techniques for point and line tokens, which minimize both rotation and translation simultaneously are developed and shown to be far superior to the earlier techniques which solved for rotation first and then translation. However, least-squares techniques fail catastrophically when outliers (or gross errors) are present in the match data. Outliers arise frequently due to incorrect correspondences or gross errors in the 3D model. Robust techniques for pose determination are developed to handle data contaminated by fewer than 50.0 % outliers.



To make the model based approach widely applicable, it is necessary to be able to automatically build the landmark models. The approach adopted in this thesis is one of model extension and refinement. A partial model of the environment is assumed to exist and this model is extended over a sequence of frames. As will be shown in the experiments, the prior knowledge of the small partial model greatly enhances the robustness of the 3D structure computations. The initial 3D model may have errors and these too are refined over the sequence of frames.

Finally, the sensitivity of pose determination and model extension to incorrect estimates of camera parameters is analyzed. It is shown that for small field of view systems, offsets in the image center do not significantly affect the location of the camera and the location of new 3D points in a world coordinate system. Errors in the focal length significantly affect only the component of translation along the optical axis in the pose computation.

# TABLE OF CONTENTS

	<u>Page</u>
ACKNOWLEDGEMENTS . . . . .	v
ABSTRACT . . . . .	vii
LIST OF TABLES . . . . .	xii
LIST OF FIGURES . . . . .	xiv
1. INTRODUCTION . . . . .	1
1.1 Mobile Robot Project . . . . .	3
1.2 Thesis Outline . . . . .	6
1.2.1 Pose Determination: Least-Squares Approach . . . . .	6
1.2.2 Robust Pose Determination: Outliers . . . . .	8
1.2.3 Model Extension and Refinement . . . . .	9
1.2.4 Sensitivity to Camera Parameters . . . . .	9
2. LEAST-SQUARES METHODS FOR POSE DETERMINATION . . . . .	11
2.1 Introduction . . . . .	11
2.2 Rotation and Translation Constraints . . . . .	13
2.2.1 Pose Constraint for Points . . . . .	13
2.2.2 Pose Constraint for Lines. . . . .	16
2.2.3 Minimum Number of Lines/ Points. . . . .	20
2.3 Previous Work . . . . .	21
2.3.1 Special Configurations . . . . .	22
2.3.2 Three Line/ Point Correspondences . . . . .	22
2.3.3 Arbitrary Number of Line/ Point Correspondences . . . . .	24
2.3.4 Camera Calibration . . . . .	27
2.3.5 Our Approach . . . . .	28

2.4	Least-Squares Solution Methods . . . . .	32
2.4.1	Objective Functions for line data . . . . .	33
2.4.2	Objective Functions for point data . . . . .	37
2.4.3	Non-linear Technique for "R_and_T" . . . . .	38
2.4.4	Initial Estimates of Rotation and Translation . . . . .	44
2.4.5	Prior Estimates . . . . .	45
2.5	Results Using Point Data . . . . .	46
2.6	Results Using Line Data . . . . .	49
2.6.1	Synthetic Data Results for "R_and_T" and "R_then_T" . . . . .	51
2.6.2	Results for "R_and_T_mod" and "R_and_T_img" . . . . .	53
3.	<b>ROBUST METHODS FOR POSE DETERMINATION . . . . .</b>	<b>65</b>
3.1	Introduction . . . . .	65
3.2	Previous Work . . . . .	66
3.3	M-Estimation Techniques . . . . .	70
3.3.1	The "Tuk_wts" Algorithm . . . . .	74
3.4	Least Median of Squares (LMS) Technique . . . . .	77
3.5	Results and Discussion . . . . .	81
3.5.1	The First Seven Indoor Frames . . . . .	83
3.5.2	Outdoor Frame 1 . . . . .	84
3.5.3	Indoor Frame 8 . . . . .	87
3.5.4	Outdoor Frame 3 . . . . .	88
3.5.5	Discussion . . . . .	92
3.6	Conclusions . . . . .	93
4.	<b>MODEL EXTENSION AND REFINEMENT . . . . .</b>	<b>98</b>
4.1	Introduction . . . . .	98
4.1.1	Our Approach . . . . .	101
4.2	Pose Determination . . . . .	104
4.2.1	Experimental Results: Pose Algorithm . . . . .	106
4.3	Induced Stereo . . . . .	109
4.3.1	Model Extension and Refinement Algorithm . . . . .	112

4.4	Experimental Results . . . . .	113
4.4.1	BOX Sequence . . . . .	113
4.4.2	PUMA Sequence . . . . .	117
4.4.3	A211 Sequence . . . . .	121
4.4.4	COMP Sequence . . . . .	124
4.5	Conclusions . . . . .	127
5.	<b>SENSITIVITY TO CAMERA PARAMETERS</b> . . . . .	129
5.1	Introduction . . . . .	129
5.2	Errors in the Pose Determination Problem from Center Offsets . . . . .	132
5.2.1	Experimental Results . . . . .	135
5.3	Errors in Induced Stereo from Center Offsets . . . . .	137
5.3.1	Experimental Results for Induced Stereo . . . . .	141
5.3.2	Structure from Motion . . . . .	143
5.4	Inaccurate Estimates of the Focal Length . . . . .	144
6.	<b>CONCLUSIONS</b> . . . . .	148
6.1	Pose Determination . . . . .	148
6.2	Model Acquisition . . . . .	151
6.3	Sensitivity Analysis . . . . .	154
<b>APPENDICES</b>		
A.	<b>DIFFERENT WAYS OF REPRESENTING 3D ROTATION</b> . . . . .	155
B.	<b>UNIFORM SAMPLING OF THE ROTATION</b> . . . . .	159
C.	<b>LINEAR SYSTEM THEORY</b> . . . . .	160
C.1	Hat Matrix . . . . .	160
<b>BIBLIOGRAPHY</b> . . . . .		162

## LIST OF TABLES

Table	Page	
2.1	Estimated Errors of Location in world coordinates for “R_and_T” algorithm for point data. . . . .	48
2.2	Estimated Pose for 12 different input rotation samples using a combination of algorithms “R_then_T” and “R_and_T” (point) data. . . . .	49
2.3	Average Absolute Error of Translation and Rotation in camera coordinates for algorithm “R_and_T”. . . . .	54
2.4	Average Absolute Error of Translation and Rotation in camera coordinates for algorithm “R_then_T”. . . . .	54
2.5	Standard deviation of Translation and Rotation error in world coordinates for algorithms “R_and_T_img” and “R_and_T_mod” with high prior covariance estimates for translation. . . . .	56
2.6	Standard deviation of Translation and Rotation error in world coordinates for algorithms “R_and_T_img” and “R_and_T_mod” with low prior covariance estimates for translation. . . . .	56
2.7	Estimated Errors of Translation in world coordinates for algorithm “R_and_T_img” and “R_and_T_mod”. . . . .	58
3.1	Estimated Errors of Translation in world coordinates for algorithms “R_and_T_img”, “MED_R_and_T_img” and “Tuk_Wts”; High Covariance Case. . . . .	85
3.2	Estimated Errors of Translation in world coordinates for algorithms “R_and_T_mod”, “MED_R_and_T_mod”; High Covariance Case. . . . .	85
3.3	Estimated errors of translation in world coordinates for algorithms “R_and_T_img”, “MED_R_and_T_img”; Low Covariance Case. . . . .	86
3.4	Estimated errors of translation in world coordinates for algorithms “R_and_T_mod”, “MED_R_and_T_mod”; Low covariance case. . . . .	86
3.5	Residual Errors (pixels) and Diagonal Hat Matrix entries for outdoor frame 2 for data with outliers. . . . .	90

4.1	Experimental and Computed standard deviation of Translation and Rotation error in world coordinates for algorithms "R_and_T" for point data. . . . .	107
4.2	Box Sequence, Model Extension Results . . . . .	115
4.3	Absolute and Percentage 3D location errors for points in PUMA sequence. . . . .	119
4.4	Absolute and Percentage 3D location errors for points in A211 room sequence. . . . .	122
4.5	Absolute and Percentage 3D location errors for points in COMP sequence. . . . .	125
5.1	Computed location of camera in world coordinates using different center offsets. . . . .	137
5.2	Predicted percentage average depth errors versus computed average depth errors for different center offsets. . . . .	142
5.3	Computed average depth errors for synthetic uniform noise data with and without center offsets. . . . .	142
5.4	Computed (pose) rotation and translation with different focal lengths. . . . .	147

## LIST OF FIGURES

Figure	Page
1.1 Block Diagram for Milestone Verification. . . . .	5
2.1 Perspective projection. . . . .	15
2.2 Error function based on the "Infinite Image Line" constraint. . . . .	35
2.3 Error function based on the "Infinite Model Line" constraint. . . . .	35
2.4 Image Lines used for convergence experiment. . . . .	43
2.5 Projection of model using initial estimate for convergence experiment.	43
2.6 Projection of model using estimate after first iteration for convergence experiment. . . . .	43
2.7 Projection of model using estimate after second iteration for conver- gence experiment. . . . .	43
2.8 Projection of model using estimate after third and final iteration for convergence experiment. . . . .	43
2.9 A211 sequence Frame 1. . . . .	47
2.10 Outdoor frame 2 with input image lines. . . . .	61
2.11 Projection of model (Outdoor frame 2) using final estimate of algorithm "R_and_T_img" . . . . .	61
2.12 Projection of model (Outdoor frame 2) using final estimate of algorithm "R_and_T_mod" . . . . .	61
2.13 Outdoor frame 4 with input image lines. . . . .	62
2.14 Projection of model (Outdoor frame 4) using final estimate of algorithm "R_and_T_img" . . . . .	62
2.15 Projection of model (Outdoor frame 4) using final estimate of algorithm "R_and_T_mod" . . . . .	62

2.16	Indoor frame 1 with input image lines. . . . .	63
2.17	Projection of model for Indoor frame 1 using final estimate of algorithm "R_and_T_img" . . . . .	63
2.18	Projection of model for Indoor frame 1 using final estimate of algorithm "R_and_T_mod" . . . . .	63
2.19	Indoor frame 3 with input image lines. . . . .	64
2.20	Projection of model for Indoor frame 3 using final estimate of algorithm "R_and_T_img" . . . . .	64
2.21	Projection of model for Indoor frame 3 using final estimate of algorithm "R_and_T_mod" . . . . .	64
3.1	Huber's minimax function and its derivative. . . . .	72
3.2	Tukey's biweight redescending function and its derivative. . . . .	72
3.3	Indoor frame 7 with input image lines. . . . .	94
3.4	Projection of model for indoor frame 7 using final estimate of Algorithm "R_and_T_mod" with high prior covariance matrix. . . . .	94
3.5	Projection of model for indoor frame 7 using final estimate of Algorithm "Med_R_and_T_mod" with high prior covariance matrix. . . . .	94
3.6	Projection of model for indoor frame 7 using final estimate of Algorithm "Tuk_wts" . . . . .	94
3.7	Outdoor frame 1 with input image lines. . . . .	95
3.8	Projection of model for outdoor frame 1 using final estimate of Algo- rithm "R_and_T_mod" with low prior covariance matrix. . . . .	95
3.9	Projection of model for outdoor frame 1 using final estimate of Algo- rithm "Med_R_and_T_mod" with low prior covariance matrix. . . . .	95
3.10	Projection of model for outdoor frame 1 using final estimate of Algo- rithm "Tuk_wts". . . . .	95
3.11	Indoor frame 8 with input image lines. . . . .	96
3.12	Projection of model for indoor frame 8 using final estimate of Algorithm "R_and_T_mod" with high prior covariance matrix. . . . .	96



3.13	Projection of model for indoor frame 8 using final estimate of Algorithm "Med_R_and_T_mod" with high prior covariance matrix. . . . .	96
3.14	Projection of model for indoor frame 8 using final estimate of Algorithm "Tuk_wts" . . . . .	96
3.15	Outdoor frame 3 with input image lines. . . . .	97
3.16	Projection of model for outdoor frame 3 using final estimate of Algorithm "R_and_T_mod" with high prior covariance matrix. . . . .	97
3.17	Projection of model for outdoor frame 3 using final estimate of Algorithm "Med_R_and_T_mod" with high prior covariance matrix. . . . .	97
3.18	Projection of model for outdoor frame 3 using final estimate of Algorithm "Tuk_wts" . . . . .	97
4.1	Model Extension and Refinement. . . . .	103
4.2	Box Image. . . . .	116
4.3	Box Sequence, Model Extension and Refinement Plot. . . . .	116
4.4	Puma Image. . . . .	118
4.5	Puma Sequence, Model Extension Plot. . . . .	120
4.6	A211 Image. . . . .	123
4.7	COMP Image. . . . .	126
5.1	Hallway image with input 2D-image lines. . . . .	138
5.2	Hallway image, projection of model after estimation of pose, image center assumed to be frame center. . . . .	138
5.3	Hallway image, projection of model after estimation of pose, image center assumed to be offset from frame center by 20 pixels along each axis. . . . .	138

# CHAPTER 1

## INTRODUCTION

In order to autonomously navigate through a complex environment, a mobile robot must be able to plan and follow a smooth path, detect and avoid obstacles, and finally recognize when it has reached its destination. An industrial robot which has to perform complex assembly tasks must be able to recognize, locate and manipulate various tools and objects in its workspace. The visual feedback required to accurately accomplish tasks such as these will typically include the 3D motion and location of the robot, and the 3D structure and motion of obstacles and other environmental features. The general problem considered in this thesis is how this 3D information may be obtained from a sequence of images generated by a camera mounted on a mobile robot. It is assumed that rough estimates of the motion of the robot between consecutive frames are known. Many different photometric and geometric visual cues may be used to infer 3D information from a sequence of images. We restrict our attention to geometric cues such as the image flow of tokens (2D points and lines) and correspondence between modeled 3D tokens and image tokens.

In the Mobile Robot Research Project at the University of Massachusetts, a model based approach to navigation has been adopted. The premise is that the higher level goals of navigation and the degree of accuracy required would benefit greatly from knowledge of objects in the environment. For instance, visual measurements of modeled 3D landmarks provide strong constraints on the 3D position and orientation (3D pose) of the robot in a world or model coordinate system. The first set of algorithms

developed in this thesis are for robust determination of the 3D pose of the mobile robot from a matched set of model and image landmark features (*Pose Determination*). First, existing least-squares techniques for pose determination from both point and line tokens are improved upon. Least-squares techniques which minimize both rotation and translation simultaneously are developed and shown to be far superior to the earlier techniques which solved for rotation first and then translation. However, least-squares techniques fail catastrophically when outliers (or gross errors) are present in the match data. Outliers arise frequently due to incorrect correspondences or gross errors in the 3D model. Robust techniques for pose determination are developed to handle data contaminated by fewer than 50.0 % outliers.

To make the model based approach widely applicable, it is necessary to be able to automatically build the landmark models. A substantial amount of effort has been invested by computer vision researchers over the past ten years on developing robust methods for computing 3D structure from a sequence of 2D images. However, robust computation of 3D structure, with respect to even small amounts of input image noise, has remained an open problem. The approach adopted in this thesis is one of model extension. A partial model of the environment is assumed to exist and this model is extended over a sequence of frames. As will be shown in the experiments, the prior knowledge of the small partial model greatly enhances the robustness of the 3D structure computations. It also provides a stable world coordinate system within which all new measurements can be fused. The initial 3D model may have small errors and these too are refined over the sequence of frames.

Finally, the sensitivity of pose determination and model extension to incorrect estimates of camera parameters is analyzed. The camera parameters studied are focal length and image center. It is shown that for small field of view systems, offsets in the image center do not significantly affect the location of the camera and the

location of new 3D points in a world coordinate system. Errors in the focal length significantly affect only the component of translation along the optical axis in the pose computation.

## 1.1 Mobile Robot Project

In this section, a brief overview of the Mobile Robot Research Project, conducted at the University of Massachusetts, is presented. The goal is to briefly describe an overall navigation system and to place within it the algorithms and analysis developed in this thesis. Note, however, that the potential applications of the algorithms developed here are not limited to the navigation domain.

At the University of Massachusetts, a model-based approach has been adopted for the robot navigation project. The robot is supplied with a hierarchical description of the environment in terms of locales [21]. The hierarchical organization of the map gives it a functional structure and facilitates high-level goals such as path planning. For instance, in the global locale of the UMass campus, different buildings will be sub-locales and each building will be further sub-divided into rooms, hallways etc.. A basic command given to the mobile robot is to go to a certain destination from a certain start or home position. The robot then uses its map of the environment to plan a path. Details of the locale structure and path planning techniques employed can be found in [21, 22]. To ensure the robot is following its planned path and to correct any deviations from the path, visual feedback is employed. This is necessary because the actual motion of the robot is not described completely by its modeled kinematic equations. Hence from dead-reckoning alone, the actual motion of the robot cannot be accurately predicted. To reach its goal, the navigation system sets up a sequence of milestone locations along its planned path. If the robot is able

to successfully reach all its milestones, it will eventually reach its goal destination. A primary goal of the vision system is to verify if a milestone location has been successfully reached. If the milestone has been achieved, the robot proceeds to the next milestone. If not, the robot replans another path from its current location as determined from the computation of its pose. Since a milestone may not be reachable because of some unmodeled obstacles in the scene, another task of the vision system is to detect unmodeled obstacles.

Figure 1.1 shows a block diagram of the flow chart for milestone verification. To verify a milestone, the robot takes an image from its current location. 2D lines or points are extracted from the image. These lines or points serve as the visual features used for locating landmarks. From previous milestone verification and dead-reckoning over the intervening motion, the robot is expected to be at a new location. Given the expected pose of the robot, a set of visible landmarks are predicted. A selection procedure chooses the optimal set of visible landmarks to be used for the milestone verification step. These 3D landmarks are projected using the expected pose to create a 2D model data set. Correspondences are then established between the 2D model and extracted image features. Given these correspondences as input, the pose determination algorithms developed in this thesis return the location and orientation of the robot. This completes the sensory feedback loop for milestone verification. The first set of algorithms developed in this thesis are for robust determination of robot location and orientation.

The experiments reported herein have used the output of two different matching algorithms. The first matching technique is based on local search and was developed by Beveridge *et. al.* [8]. The second technique uses optic-flow based token tracking and was developed by Williams *et. al.* [76]. The matching algorithms produce a list of correspondences between 3D model features and 2D image features. Some of the

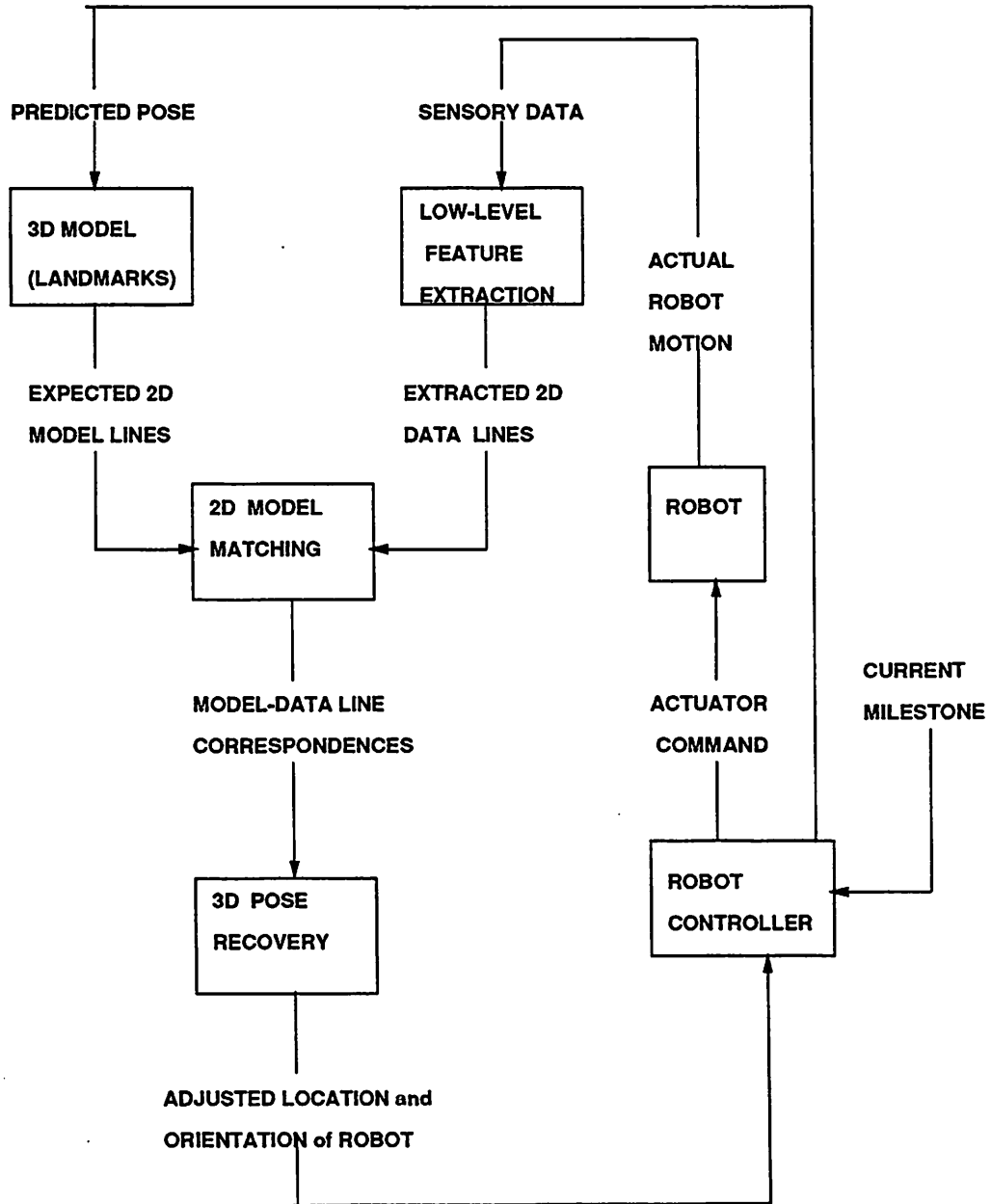


Figure 1.1: Block Diagram for Milestone Verification.

correspondences returned by the matching algorithms may be wrong and therefore the pose determination process must be insensitive to these matching errors.

In this framework, a key step for navigation is milestone verification. For the robot to verify a milestone, however, it need not have a complete map of the environment. The process of navigation can be begun with a partial map of the environment. However, as the robot explores the environment, it is desirable to enhance the initial map. Also, the initial model may contain errors and these too should be refined over time. Finally, as noted above there may be unmodeled obstacles in the robot's path; these must be detected and their locations returned to the planner to replan a path around them. This leads us to the second set of algorithms developed in this thesis; those for model extension and refinement. The initial partial model is used to develop a more complete model of the environment over time; in the process, the initial model is also refined.

## 1.2 Thesis Outline

Each of the next four chapters is self-contained with its own literature survey and a results section with extensive experiments reported on both simulated and real data. Each consecutive chapter builds upon and uses the theories and algorithms developed in the previous chapters.

### 1.2.1 Pose Determination: Least-Squares Approach

Chapter 2 presents techniques for determination of pose using least-squares methods. The pose determination problem is defined as follows: "Given correspondences between 3D lines (or points) in the model and 2D lines (or points) found in the image, find the rotation and translation matrices which map the world coordinate system

to the camera coordinate system.” Most existing techniques minimize the sum of squares of a non-linear error function. The goal is to find the pose which best aligns the perspective projection of the 3D model with the image data. Previously developed techniques solved for rotation first and then translation (“R\_then\_T”). In Chapter 2 a new *least-squares* algorithm called “R\_and\_T” is developed which simultaneously solves for both rotation and translation; this algorithm performs much better than “R\_then\_T”.

In the case of line token data, it is assumed that end-points of image lines cannot be reliably extracted. This gives rise to two different sets of error functions: the first aligns a model line segment with an infinitely extended image line while the second aligns an image line segment with an infinitely extended model line. We show in the experiments that if there is significant line fragmentation (which produces unreliability in the end-point estimate), then the “infinite model line” method performs much more better than the “infinite image line” method.

Both of the above algorithms are iterative in nature and require initial estimates of the rotation and translation matrices. However, experiments show that there is rapid convergence even with significant errors in the initial estimates. These initial estimates can be obtained in several ways. In the case of autonomous navigation, for example, we usually know an approximate pose of the robot from the last verified robot pose and dead reckoning. In those cases where such data is not available, it is possible to sample the rotation space and use the different sample points as initial estimates for the “R\_then\_T” algorithm. The output estimates from the different “R\_then\_T” runs are fed into the “R\_and\_T” algorithms and the best fitting pose chosen as the final answer. Hence, the pose determination process can be bootstrapped by selecting initial estimates from a regular sampling of the space of all orientations.



### 1.2.2 Robust Pose Determination: Outliers

Visual processes have often been modeled in a parameter estimation paradigm, although rarely is the assumed model valid over the entire range of the visual data. A central problem therefore is to know when the underlying process model breaks down. Parameter estimation methods based on least-squares fitting break down when there are gross errors or outliers in the data. In the pose determination problem, incorrect correspondences or gross errors in the 3D model can give rise to outliers.

In Chapter 3, we investigate two different statistical techniques for the “robust” estimation of pose with data having outliers. The first set of techniques, called M-Estimation techniques, minimize non-convex functions of the individual residual errors. The effect of large errors is bounded by saturating the minimization function. Experimentally, the M-estimate methods seem to be susceptible to initial estimates and are not able to handle a large number of outliers (a maximum of approximately 20%). However, there are efficient computational methods for minimizing the associated error functions.

The second set of algorithms developed are capable of performing correctly in situations where the number of outliers is less than 50% of the number of data points. Also, they are not as sensitive as the M-Estimate techniques to initial estimates. These minimize the Least Median Square (LMS) of the residual error functions across all landmarks. The LMS pose computed is used to detect and remove outliers, and then a least-squares fit on the remaining data is used to obtain the rotation and translation matrix estimates. These algorithms are computationally much slower than those based on M-Estimation techniques. Using random subset selection methods, the average time complexity of the LMS-based algorithms can be substantially reduced with a minimal loss in robustness.

### 1.2.3 Model Extension and Refinement

An important problem in vision is to automatically build 3D models of objects. The approach we have adopted in Chapter 4 is to first begin with a partial model and to then extend and refine it by viewing the object over a sequence of frames. Both modeled and unmodeled features of the object are tracked over the image sequence by using an optic flow based token-tracking algorithm. Correspondences are obtained between the modeled 3D features and their image projections. New features are located by triangulation using the displacement of image tokens and the poses of the object computed from model-image feature correspondences for a sequence of image frames. The estimation of the new 3D points is done using both batch and sequential Kalman filter based methods. New 3D points are located in four real data sequences with average errors less than 1.7% . These results are far superior to those obtained by the traditional structure from motion techniques employed in computer vision. This supports the premise that prior knowledge of a partial model greatly extends the robustness of the structure estimates.

The initial partial model may have errors in some applications. We therefore extend the pose determination techniques to optimize for errors in both the image and model data. Triangulation is done using the poses computed by the extended technique to obtain new measurements for both new and old model points. The new measurements and old estimates are then combined using their respective covariance matrices. This is essentially an exercise in data fusion and leads to both model refinement and extension.

### 1.2.4 Sensitivity to Camera Parameters

Camera calibration is an important and difficult task in vision. The goal in camera calibration is to make accurate estimates of intrinsic camera parameters such as focal

length, image center, lens distortion, etc. Chapter 5 discusses how critical this task is for the pose determination and model extension problems. We show that the sensitivity to errors in various intrinsic camera parameters depends on the particular 3D inference to be made and the amount of noise in the system. In particular, for “small” field of view imaging systems, incorrect knowledge of the camera center (or “principal point”) fortunately does not significantly affect the determination of the location of the robot camera or the location of new points in the world coordinate system. However, the orientation of the robot is affected. The amount of error in the orientation is linearly related to the error in the estimate of the center. These results are extended to analyze the effect of center offsets on the structure from motion problem. It is also shown that incorrect estimates of the focal length only significantly affects the z-component (i.e. the component parallel to the optical axis) of the translation in camera coordinates.

Finally in Chapter 6, the main contributions of this thesis are summarized. We also present open problems and propose future directions for research.

## CHAPTER 2

# LEAST-SQUARES METHODS FOR POSE DETERMINATION

### 2.1 Introduction

This chapter mathematically analyzes and proposes a new solution for the problem of estimating camera location and orientation (*Pose Determination*) from a set of recognized landmarks appearing in the image. Given correspondences between 3D lines (or points) and 2D lines (or points) found in the image, the goal is to find the rotation and translation matrices which map the world coordinate system to the camera coordinate system. Algorithms are developed for handling both 3D line and 3D point landmarks. The line landmarks employed are real or virtual 3D lines represented in a world coordinate system. A minimum of three pairs of point or line correspondences are needed to solve the pose problem. In practice, however, many more correspondences may be available and all of them should be used to obtain the best possible estimate of the pose.

This chapter presents a sequence of least-squares techniques to solve the pose determination problem, each performing better than the previous one. The least-squares techniques minimize non-linear functions, are iterative in nature and require an initial estimate. For those cases in which an initial estimate of rotation and translation is not available, techniques based on sampling the rotation space to provide initial estimates are developed. Least-squares techniques are known to be sensitive

to gross errors or *outliers* in the data; techniques based on robust statistics to handle outliers are developed in Chapter 3.

The camera model assumed is perspective projection. Intrinsic camera parameters, such as focal length, field of view, center of the image, size of image etc. are assumed to have been estimated by a camera calibration procedure [19, 49, 73]. In Chapter 5, the sensitivity of the pose parameters to errors in the intrinsic camera parameters such as focal length and image center are studied.

A mathematical analysis of an uncertainty measure is developed, which relates the variance in the output parameters to the noise present in the input parameters. This analysis is used to compute a covariance matrix which represents the uncertainty in the estimated pose parameters. In Section 2.6 and Section 4.2.1, results of Monte-Carlo analysis over thousands of experiments are reported and the experimentally derived covariance matrices are compared with the computed covariance matrix. In certain scenarios there are strong prior expectations of the pose of the robot. For instance, in the hallway domain, the height of the robot (the z-coordinate of the sensor location vector) changes very little over a sequence of frames. The prior expectation can be captured by an initial estimate with an associated prior covariance matrix. An extended Kalman filter combines both the new measurements and the initial estimate weighted with its covariance matrix to obtain new pose estimates. For the experiments and analysis in this chapter, it is assumed that there is no noise in the 3D model data and the only input noise is in the image data. In Chapter 4, the pose determination algorithm is extended to handle errors in the model.

The remainder of the chapter is organized as follows: Section 2.2 discusses the geometry of perspective projection and the rotation and translation constraints. Section 2.3 critiques previous approaches and motivates our approach. Section 2.4 presents the least-squares non linear technique and solution methods for situations when there

is no good initial estimate. Section 2.5 presents results for the pose algorithms using point data and Section 2.6 presents results using line data. For real image data results are presented for environments in which the landmarks are on the order of hundreds of feet distant from the camera. For point data more results are also presented in Chapter 4. Appendix A discusses various representations for rotation and motivates the particular choice of quaternions for large rotations and the 3D rotation vector for small angles.

## 2.2 Rotation and Translation Constraints

In this section, the basic constraints for pose determination are developed. Given correspondences between 3D lines (or points) and 2D lines (or points) found in the image, the goal in pose determination is to find the rotation and translation matrices which map the world coordinate system to the camera coordinate system. The constraints developed in this chapter relate the rotation and translation parameters to the 3D model line (or point) coordinates and the corresponding 2D image line (or point) coordinates. These constraints are used in Section 2.4 to develop the objective functions, which are minimized to find the optimum pose parameters given noisy input data.

### 2.2.1 Pose Constraint for Points

Points in 3D space are represented by 3D vectors  $\vec{p}$ . Lines in 3D are represented by two end-points  $\vec{p}_1$  and  $\vec{p}_2$ . The unit vector corresponding to the direction  $\hat{d}$  of the 3D line is determined by subtracting the two end-point vectors. The rigid body transformation from the world coordinate system to the camera coordinate system can be represented as a rotation ( $R$ ) followed by a translation ( $\vec{T}$ ). The point  $\vec{p}$  in

world coordinates gets mapped to the point  $\vec{p}_c$  in camera coordinates. The mapping is given by:

$$\vec{p}_c = R(\vec{p}) + \vec{T} \quad (2.1)$$

In this equation, except for the rotation  $R$ , all the terms are column vectors with 3 components each, referred to by the subscripts  $x$ ,  $y$  and  $z$ .  $R$  is the rotation operator and can be expressed in many ways, e.g. orthonormal matrices, quaternions, axis and angle, etc. The various representations for rotation are discussed in Appendix A. Based on the discussion there, quaternions were chosen to represent large rotations and 3D vectors chosen for small rotations.

Figure 2.1 shows the camera and world coordinate systems with  $X_w$ ,  $Y_w$  and  $Z_w$  representing the axes of the world coordinate system.  $O$  is the optical center of the lens and also the origin of the camera coordinate system  $Ox_cY_cZ_c$ .  $OZ_c$  is the optical axis of the camera. In equation (2.1) the translation vector  $\vec{T}$  represents the location of the origin of the world coordinate system in camera coordinates. Equation (2.1) can be rewritten to map points in the camera coordinate system to the world coordinate system:

$$\vec{p} = R^T(\vec{p}_c) + \vec{T}_w \quad (2.2)$$

In this equation,  $R^T$  is the inverse of the rotation operator (transpose if rotation is expressed as an orthonormal matrix) in equation (2.1). " $\vec{T}_w$ " represents the location of the origin of the camera coordinate system in world coordinates. " $\vec{T}_w$ " is related to " $\vec{T}$ " by the following equation:

$$\vec{T}_w = -R^T(\vec{T}) \quad (2.3)$$

The 3D line "AB" in (Figure 2.1) projects to the image line "ab". Image points  $\vec{I}, \vec{J}$  are represented by 2D vectors. A 3D point  $\vec{p}_c$  projects to an image pixel  $\vec{I}$  by the

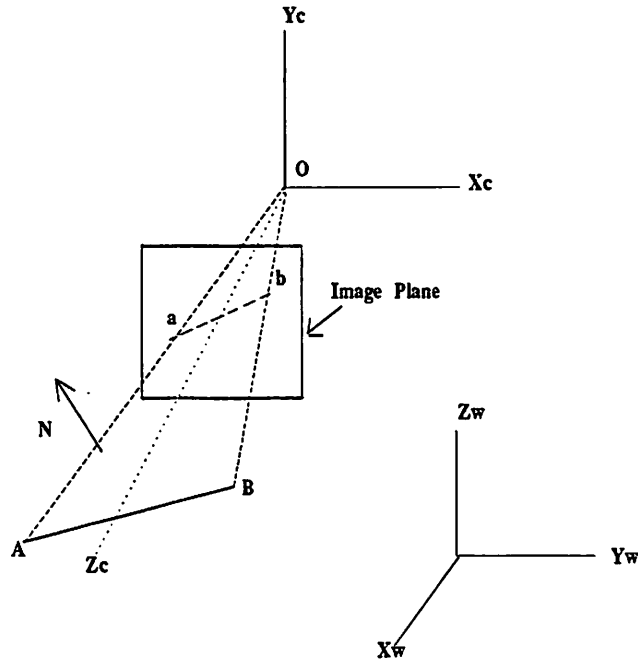


Figure 2.1: Perspective projection of image lines.

following equations:

$$I_x = s_x \frac{p_{cx}}{p_{cz}} \quad I_y = s_y \frac{p_{cy}}{p_{cz}} \quad (2.4)$$

where  $s_x$  and  $s_y$  are the scale factors along the "X" and "Y" directions respectively. They are related to the field of view angles  $\phi_x, \phi_y$  and the image size  $N_r$  (number of rows) and  $N_c$  (number of columns) by:

$$s_x = \frac{N_c}{2} \cot\left(\frac{\phi_x}{2}\right) \quad s_y = \frac{N_r}{2} \cot\left(\frac{\phi_y}{2}\right) \quad (2.5)$$

Using equations (2.1) and (2.4) the constraint equations for the pose parameters given point correspondences are:

$$I_x = s_x \frac{(R\vec{p} + \vec{T})_x}{(R\vec{p} + \vec{T})_z} \quad (2.6)$$

$$I_y = s_y \frac{(R\vec{p} + \vec{T})_y}{(R\vec{p} + \vec{T})_z} \quad (2.7)$$



Each point correspondence gives two constraint equations. The “x” and “y” coordinates of the projected model point must equal the “x” and “y” coordinates of the measured image point, respectively. In Section 2.4, this constraint is used to develop an objective function given point correspondences as input. The objective function is minimized to find the optimal pose parameters.

### 2.2.2 Pose Constraint for Lines.

In the case of line correspondences, it is assumed that model and image line end-point correspondences cannot be established. Constraint equations for the pose parameters can be developed in two ways. In the first case (“Infinite Image Line”) a model line segment is aligned with an infinitely extended image line, whereas in the second case (“Infinite Model Line”) an image line segment is aligned with an infinitely extended model line.

#### The “Infinite Image Line” Case

In the “Infinite Image Line” case an image line is represented by its  $(\rho, \theta)$  parameters. Any image point  $(I_x, I_y)$  on the  $i$ 'th line must satisfy the following equation:

$$I_x \cos \theta + I_y \sin \theta = \rho \quad (2.8)$$

Substituting  $I_x$  and  $I_y$  from equation (2.4) into equation (2.8), the equation of the projection plane formed by the image line and the optical center is obtained :

$$\frac{(s_x \cos \theta)p_{cx} + (s_y \sin \theta)p_{cy} - \rho p_{cz}}{p_{cz}} = 0 \quad (2.9)$$

In Figure 2.1 , the projection plane formed by the image line “ab” is given by the plane “Oab” and the 3D line “AB” must lie in this plane. The normal  $\vec{N}$  to the projection plane is given by:

$$\vec{N} = (s_x \cos \theta, s_y \sin \theta, -\rho)^T \quad (2.10)$$

Using equations (2.1) and (2.10), equation (2.9) can be rewritten as follows:

$$\frac{\vec{N} \cdot (R(\vec{p}) + \vec{T})}{(R(\vec{p}) + \vec{T})_z} = 0 \quad (2.11)$$

This equation is the basic constraint for the “Infinite Image Line” case. The projected model end-point must lie on the infinitely extended image line. If the denominator is dropped and the vector  $\vec{N}$  made into a unit vector  $\hat{N}$ , the equivalent 3D constraint is formulated:

$$\hat{N} \cdot (R(\vec{p}) + \vec{T}) = 0 \quad (2.12)$$

The 3D constraint represents the fact that any point on the 3D line in camera coordinates ideally must lie in the projection plane. The vector formed from the origin (optical center) to this point must be perpendicular to the normal of the projection plane.

These constraint equations (2.11 and 2.12) relate both rotation and translation pose parameters to the 3D model and 2D image coordinates. A separate constraint involving just rotation was formulated by Liu et.al. [53]:

$$\hat{N} \cdot R(\hat{d}) = 0 \quad (2.13)$$

The constraint reflects the fact that the 3D line must lie in the projection plane formed by its corresponding image line. Rigid body transformation can be represented as a rotation followed by a translation. Since translation does not change the direction of the line, the direction of the 3D line after rotation in equation (2.13) must be perpendicular to the normal of the projection plane of the image line.

From the constraints represented in equations (2.12) and (2.13), two algorithms have been developed to solve for rotation and translation. In the first algorithm (“R.then\_T”) the constraint in equation (2.13) is used to solve for rotation. Then the rotation result returned from this step is used in conjunction with equation (2.12) to solve for translation.

In the second algorithm (“R.and.T”), only equation (2.12) is used and both rotation and translation are solved for simultaneously. For each line, the two end points must satisfy equation (2.12). The tables in Section 2.6 will show that “R.and.T” performs much better than “R.then.T”. Finally, a third algorithm (“R.and.T.ing”) is developed which uses the image-based constraint equation (2.11) instead of the 3D-based constraint equation (2.12) to solve for rotation and translation simultaneously.

### The “Infinite Model Line” Case

In the “Infinite Model Line” case, an image line segment is aligned with the infinite extension of a projected model line. The  $(\rho, \theta)$  parameters in the line equation (2.8) are now used to represent the infinite extension of the projected model line<sup>1</sup>. The image line end-points ( $\vec{I}_1$  and  $\vec{I}_2$ ) must lie on this projected model line and satisfy equation (2.8).

Expressions are derived for  $(\cos(\theta), \sin(\theta)$  and  $\rho)$  in equation (2.8) in terms of the model end-points  $(p_1, p_2)$  and the pose  $(R, \vec{T})$ .  $J_1$  and  $J_2$  are the image projections of the 3D model line end-points  $(p_1$  and  $p_2)$ . The expressions for  $(\cos(\theta), \sin(\theta)$  and  $\rho)$  for the image line between ( $\vec{J}_1$  and  $\vec{J}_2$ ) are:

$$\cos \theta = \frac{(J_{2y} - J_{1y})}{\sqrt{((J_{1x} - J_{2x})^2 + (J_{2y} - J_{1y})^2)}} \quad (2.14)$$

$$\sin \theta = \frac{(J_{1x} - J_{2x})}{\sqrt{((J_{1x} - J_{2x})^2 + (J_{2y} - J_{1y})^2)}} \quad (2.15)$$

$$\rho = \frac{(J_{1x}J_{2y} - J_{2x}J_{1y})}{\sqrt{((J_{1x} - J_{2x})^2 + (J_{2y} - J_{1y})^2)}} \quad (2.16)$$

The projections of the 3D model line end-points into the image plane points ( $\vec{J}_1$  and  $\vec{J}_2$ ) are given by equations (2.6) and (2.7). Substituting these expressions for

---

<sup>1</sup>This is in contrast to the “Infinite Image Line” case where the  $(\rho, \theta)$  parameters in equation (2.8) were used to represent the image line.

points  $(\vec{J}_1, \vec{J}_2)$  into the equations for  $(\cos(\theta), \sin(\theta)$  and  $\rho)$  and using equation(2.3) the following is obtained:

$$\cos \theta = \frac{1}{s_x M} (R(\vec{p}_2 - \vec{T}_w) \times R(\vec{p}_1 - \vec{T}_w))_x \quad (2.17)$$

$$\sin \theta = \frac{1}{s_y M} (R(\vec{p}_2 - \vec{T}_w) \times R(\vec{p}_1 - \vec{T}_w))_y \quad (2.18)$$

$$\rho = -\frac{1}{M} (R(\vec{p}_2 - \vec{T}_w) \times R(\vec{p}_1 - \vec{T}_w))_z \quad (2.19)$$

where the vector  $\vec{A}$  and scalar  $M$  are defined as follows:

$$\vec{A} = R(\vec{p}_2 - \vec{T}_w) \times R(\vec{p}_1 - \vec{T}_w) \quad (2.20)$$

$$M = \sqrt{\left(\frac{A_x}{s_x}\right)^2 + \left(\frac{A_y}{s_y}\right)^2} \quad (2.21)$$

The line equation (2.8) can be rewritten as the dot-product of two vectors:

$$(I_x, I_y, 1) \cdot (\cos \theta, \sin \theta, -\rho) = 0.0 \quad (2.22)$$

We now define vector  $\vec{B}$  corresponding to the projection ray from the focal point to an image point  $\vec{I}$  as:

$$\vec{B} = \left(\frac{I_x}{s_x}, \frac{I_y}{s_y}, 1\right)^T \quad (2.23)$$

Using the above definition and substituting equation (2.17), (2.18) and (2.19) into the line equation (2.22) the ‘‘Infinite Model Line’’ constraint equation becomes:

$$\frac{1}{M} (\vec{B} \cdot R(\vec{p}_2 - \vec{T}_w) \times R(\vec{p}_1 - \vec{T}_w)) = 0 \quad (2.24)$$

or

$$\frac{1}{M} (R^T \vec{B} \cdot (\vec{p}_2 - \vec{T}_w) \times (\vec{p}_1 - \vec{T}_w)) = 0 \quad (2.25)$$

or

$$\frac{1}{M} (R^T \vec{B} \cdot (\vec{p}_2 \times \vec{p}_1 + \vec{T}_w \times (\vec{p}_2 - \vec{p}_1))) = 0 \quad (2.26)$$

For each image line end-point, the constraint equation (2.26) is developed. It represents the constraint that the image line end-points  $\vec{I}_1, \vec{I}_2$  must lie on the projection of the 3D model line. From this constraint, another least-squares algorithm

(“R\_and\_T\_mod”) is developed in Section 2.4 for computing the pose parameters given line correspondences. In Section 2.6, it is shown that algorithm (“R\_and\_T\_mod”) performs more robustly than algorithm (“R\_and\_T\_img”) when there is significant line fragmentation in the image data.

### 2.2.3 Minimum Number of Lines/ Points.

Both rotation and translation in the 3D world can be represented by three parameters each. Each line or point data gives us 2 constraint equations (2.6,2.7). Thus, a minimum of three lines or points are needed. However, in many cases, with three lines or points, there is no unique solution. If the three lines are parallel in 3D space or lie on the same projection plane, then an infinite number of solutions can be found. If the three lines meet at a common point in 3D space, then there are two solutions for rotation (the Necker cube phenomena) and an infinite number of solutions for translation.

In general, there are eight possible solutions for the 3 point or line case, four of which correspond to the 3D points lying in front of the camera and four corresponding to the 3D points lying behind the camera. The fact that there are eight possible solutions for the three line case can be seen by representing the rotation operator in equation (2.13) as quaternions. Let the directions of the 3 lines in the world coordinate space be  $\vec{d}_1, \vec{d}_2, \vec{d}_3$  respectively. Representing the rotation operator  $R$  in equation (2.13) by the quaternion  $q$  (see Appendix A), the following three equations can be written:

$$\vec{N}_1 \cdot (q \circ d_1 \circ q^*) = 0 \quad (2.27)$$

$$\vec{N}_2 \cdot (q \circ d_2 \circ q^*) = 0 \quad (2.28)$$

$$\vec{N}_3 \cdot (q \circ d_3 \circ q^*) = 0 \quad (2.29)$$

Since quaternions are 4-tuples, each of these equations is a second degree polynomial

equation with four unknown variables. Due to the fact that quaternions must be a unit vector, a fourth polynomial equation of degree two (Equation (A.8) in Appendix A) can be written. From elementary algebra, it is well known that a system of four second degree equations in four variables can have a maximum of eight solutions. Thus, for the three line pose problem using only line direction information, there are a maximum of 8 solutions. The same result holds for the three point case because three lines can be drawn between the three points. However, another property of the imaging process is that the viewed scene must lie in front of the camera. If this constraint is enforced, that then the three line/ point pose problem has a maximum of four solutions [16, 23, 29, 36]. Wolfe et. al. [78] also provide geometric explanations for 3 point configurations that cause 1, 2 or 4 solutions.

### 2.3 Previous Work

The problem of “pose determination” has been referred to by various other names including “exterior orientation”, “determination of camera location and orientation”, “pose refinement”, “perspective inversion” and “model alignment”. There have been many papers on pose determination, but most assume only point data is available; only a few have presented techniques for line data. Previous work can be classified into three categories: (1) Special configurations, (2) Three (minimum number) lines/ points, and (3) Arbitrary set of lines/ points. A related problem to pose determination is “camera calibration” or “interior orientation”. The last subsection of our review will briefly discuss some of these methods and their relevance to pose determination techniques.

### 2.3.1 Special Configurations

The first category includes papers which deal with pose determination from special configurations of landmarks such as cubes, rectangles, crosses etc.. For instance, Kite and Magee [40] present algorithms for pose determination from rectangular landmarks. Tseng et.al. [74] determine the pose of a mobile robot in 3D space from a single 2D image of a cube. They derive initial estimates from a vanishing point analysis. The estimated pose is then refined using the constraints imposed by the structure of the cube. Paquette et.al. [62] determine the orientation of a robot given the image projections of a set of three orthogonal lines in 3D space. However, this kind of work is not extensible to an arbitrary set of landmarks and therefore is not very useful to us.

### 2.3.2 Three Line/ Point Correspondences

A fair amount of research has been done on the problem of finding closed form solutions for the pose problem using the minimal set of three lines or points. The goal is to find computationally efficient techniques which are robust with respect to noise. The first known solution for the three point pose problem<sup>2</sup> was obtained by a German photogrammetrist in 1841 [27, 29]. Fischler and Bolles [23] were among the first in the computer vision literature to provide a closed form solution for the pose problem using three pairs of point correspondences. They solve for the “legs” of the points (the lengths of rays from the optical center of the camera to the points in 3D space). The closed form solution they present is quite complex and involves solving a quartic equation. Linnainmaa et.al. [52] provide another solution for the three point pose problem. Their solution involves solving an eighth order equation. Recently, Haralick et.al. [29] have done a comprehensive survey of six different tech-

---

<sup>2</sup>Referred to in the photogrammetry literature [77] as the “three point space resection problem.”

niques (including the above two) for the three point pose problem. They compare the six techniques with respect to numerical stability and robustness towards noise, finite word lengths etc.. Each of the methods reduces the problem to solving a polynomial equation in one variable. Those methods which require solving smaller degree polynomial equations rather than larger degree equations perform better. For instance, Fischler et.al.'s [23] technique which solves a quartic equation performs better than Linnainmaa et.al.'s [52] technique, which requires that an eighth order equation be solved. Haralick et.al. [29] also report that for similar degree equations "depending on the order of substitutions realized to obtain the equation, the relative error can change over a thousand to one". Their analysis technique guarantees obtaining a method which provides stable solutions.

In the case of lines, Horaud [36] devised a closed form solution for 3 line segments meeting at a point. Like the solution of Fischler et.al. [23] for the three point case, Horaud must solve a quartic equation. Dhome et.al. [16] find a solution for a general set of three lines. Their solution involves solving an eighth order equation.

In the literature, two advantages are given for devising methods to compute pose which use the minimum size set of three pairs of line/ point correspondences. The first advantage is that initial estimates are not needed, since it is possible to develop closed form solutions. It was noted in the previous section that there is a maximum of four possible solutions for an input set of three line or point correspondences. Note that in ensuing sections, techniques are developed for an arbitrary configuration of lines/ points which likewise do not need initial estimates. The techniques depend on sampling the rotation space to find initial estimates for rotation, from which the initial estimate of translation can be determined. Thus, while these techniques do not have a closed form solution, they share some advantages of those that do.

The second advantage in using solutions to the three point pose problem is that



these methods are typically used in conjunction with a matching algorithm. Using a small size input set for pose estimation reduces the computational complexity of the algorithm. For instance, two common techniques used for matching are Hypothesize/ Verify and the Hough Transform. In the Hypothesize/ Verify procedure [54], a pose computed from an hypothesized seed set of three line/ point matches is used to extend the match. Using the computed pose, if a sufficient number of plausible and consistent correspondences can be found, then the match has been verified; otherwise another triplet is used to hypothesize the next pose. In Hough-based techniques [52], poses computed from all possible sets of correspondences of line/ point triplets are clustered in the pose-space. Viable matches correspond to peaks found in this space. In either matching technique, the combinatorial complexity is reduced by using just the minimal set to compute the pose.

However, there is a need for pose estimation techniques which work with an arbitrary configuration and number of lines/ points. First, at the end of the matching cycle there will typically be many more than three correspondences available and all of them should be used to obtain the best possible estimate of the pose. Second, based on experiments discussed further in Chapter 4, it seems doubtful whether "good enough" poses can be generated in noisy situations by using just a subset of 3 line/ point correspondences. In Chapter 4 this point plays an important part in selecting the parameters of the median-based robust algorithm.

### 2.3.3 Arbitrary Number of Line/ Point Correspondences

No closed form solution has yet been found for the general pose problem with an arbitrary set of lines/ points in an arbitrary configuration. Most techniques minimize non-linear error functions, are iterative in nature and require an initial estimate.

Iterative techniques for point data based on linearization of a non-linear error

function are presented in the photogrammetry literature [77]. Wu et.al. [80] use extended Kalman filtering to solve for the pose and motion of an object given point correspondences from a matching algorithm. They track the pose of the object over a sequence of frames and have as their state vector both the position and motion of the object. Their dynamic model assumes that the velocity of the object remains constant. Mitchie et.al. [59] use constraints based on conservation of distance under rigid body transformation to solve for the length of the “legs” of an arbitrary set of points. Once the points have been located in the 3D camera coordinate system, the pose can be computed by employing a simpler absolute orientation (3D-3D pose) algorithm. This technique is employed by them to solve the related problem of relative orientation between two camera coordinate systems (structure from motion) given correspondences between points in the two image frames. However, their method involves minimizing a non-linear error function with as many variables as the number of points. In contrast, the techniques presented here solve directly for the six pose parameters and the dimension of the error function does not increase with the number of lines/ points.

Some researchers [8, 76] have examined the use of image lines as an alternative to point data; the idea is that lines provide a more stable image feature to match. It is assumed that line end-points cannot be reliably extracted and hence end-point correspondences cannot be established. This has led to the formulation of two sorts of constraint equations for pose determination: the “Infinite Image Line” case and the “Infinite Model Line” case. Constraint equations for both these cases were developed in Section 2.2. In the “Infinite Image Line” case, a model line segment is aligned with an infinitely extended image line, whereas in the “Infinite Model Line” case extracted image line segments are aligned with infinitely extended model lines. Beveridge et.al. [8] report that, in the case of a 2D-2D pose problem, if extracted

image lines have significant line breaks, then the “Infinite Model Line” algorithms perform much better than the “Infinite Image Line” algorithms. In Section 2.6, we report similar results for the 3D-2D pose problem.

Liu, Huang and Faugeras present a solution to the “camera location determination” problem which works for both point and line data [53]. Their constraint is based on the “Infinite Image Line” case and on the observation that three-dimensional lines in the camera coordinate system must lie on the projection plane formed by the corresponding image line and the optical center. Using this fact, constraints for rotation can be separated from those of translation. Equations (2.12) and (2.13) express these constraints. They first solve for the rotation and then use the rotation result to solve for the translation. Two methods are suggested to solve for the rotation constraint. In the first method, rotation is represented as an orthonormal matrix and a smallest eigenvalue based solution technique is presented. However, the six orthonormality constraints for an orthonormal matrix are not enforced. It is not clear how they would find the nearest orthonormal matrix to the matrix their algorithm returns, and whether that would be a solution to the original problem. The second method represents rotation by Euler angles and is a non-linear iterative solution obtained by linearizing the problem about the current estimate of the output parameters. The translation constraint is solved for by a linear least-squares method. Liu, Huang and Faugeras extend their technique to point data by drawing virtual lines between pairs of points [53]. They use the same rotational constraint; however, the translation constraint is different from that for lines.

Lowe [54] presents iterative techniques for both point and line data. His camera model is an approximation to perspective projection and therefore the solution presented in his book is not applicable to our problem of camera location determination, where the typical scenes that we consider have landmarks spread over a large range

in depth. For line data, the solution presented in his book is based on the “Infinite Image Line” case. However, in his recently published papers [55, 56], the camera model adopted is the exact perspective projection and the error functions minimized are based on the “Infinite Model Line” case. Lowe’s current least-squares technique for pose estimation is very similar to ours, although the two pieces of work were done independently. His papers, however, do not discuss the relative merits of the “Infinite Model Line” based error functions versus the “Infinite Image Line” based error functions.

In [56], Lowe presents techniques for stabilizing the iterative minimization method for “near singular” data sets. Lowe also integrates matching and pose estimation. His matching technique is based on the Hypothesize/ Verify paradigm discussed earlier. Based on smoothness of motion, image lines are predicted in the next frame and potential matches are weighted by their Mahalanobis Distance from the image predictions. The final matches are computed by a best-first tree search with the pose being computed at each node of the search tree [55].

Worrall et.al. [79] present a least-squares technique for line data which minimizes an error measure derived from the “Infinite Image Line” constraint equation (2.12). They compare results with one of Lowe’s solutions and report similar performance. However, they represent rotations as Euler angles and use a different non-linear technique from that presented here. They also do not handle outliers or provide a mathematical analysis of the errors.

#### 2.3.4 Camera Calibration

It is important to draw the distinction here between techniques for “camera calibration” [19, 49, 73], also called “interior orientation”, versus the techniques for “camera location determination”. Camera calibration techniques solve for intrinsic

camera parameters (such as focal length, image center, lens distortion) along with the pose parameters (rotation and translation). The techniques for camera calibration require very precise image measurements and are less tolerant to noise. Pose determination techniques are less susceptible to image noise but one needs to know the intrinsic camera parameters. In Chapter 5, the effect of errors in the intrinsic camera parameters on the pose determination problem is studied.

Ganapathy [25] presents a linear closed form solution for point data. In addition to solving for the rotation and translation parameters, he also solves for the center of the image and scaling along the “x” and “y” directions in the image. We have an implementation of his technique and find it extremely susceptible to noise, probably due to the use of a linear least-squares minimization where it is assumed that all the parameters are independent when they are not. Faugeras et.al. [19] have come up with a technique to solve a similar system of equations with appropriate constraints and report better results. Tsai et.al. [73, 49] have also developed state of the art camera calibration techniques. They assume that the lens distortion is mostly radial and solve for the lens distortion parameters and the pose parameters using a radial alignment constraint equation. The image center and the relative scale along the image “x” and “y” axes are solved for separately using a variety of special methods [49]. For their technique to work image points must be located within 0.1 pixel accuracy.

### 2.3.5 Our Approach

The first algorithm developed in this thesis is called “R\_then\_T” and is similar to the line based algorithm developed by Liu et.al. [53]. First, rotation parameters are solved for by minimizing an error function based on the “Infinite Image Line” constraint equation (2.13). The rotation estimate obtained is used to minimize an error function based on equation (2.12) to solve for translation. However a different non-

linear technique from Liu et.al. is used for the minimization process. The technique used was adapted from one used by Horn [35] to solve the problem of relative orientation (also called structure from motion). It is based on the Gauss-Newton method for minimizing non-linear functions. We believe that the application of Horn's technique gives us much better convergence properties than Liu's solution using Euler angles.

One of the results of this thesis is that decomposition of the solution into the two stages, of first solving for rotation and then for translation, does not use the set of constraints effectively [47]. This same observation was made by other researchers working on the structure from motion problem [57]. The rotation and translation constraints (refer to equations (2.12, 2.13)), when used separately, are very weak constraints. When solving for them separately, small errors in the rotation stage are amplified into large errors in the translation stage. This is particularly true with the large distances of the landmarks from the camera in our application. Much better noise immunity is obtained if both rotation and translation are solved for simultaneously. That is the approach adopted here. Using Horn's technique, another algorithm ("R\_and\_T") was developed to solve for the rotation and translation simultaneously. Section 2.6 presents results that show algorithm "R\_and\_T" performs much better than "R\_then\_T".

For the techniques developed in this chapter, the major source of noise in the input data is assumed to be gaussian noise in the image measurements<sup>3</sup>. The algorithms "R\_then\_T" and "R\_and\_T" minimize error functions which lie in the 3D projection plane and they also give more weight to lines whose end-points are further away from the camera. Therefore the "R\_and\_T" technique described above is modified to use equation (2.11) instead of equation (2.12) to develop the objective function. The resulting algorithm, called "R\_and\_T\_img", based on the "Infinite Image Line"

---

<sup>3</sup>In Chapter 5, pose determination techniques are developed where significant noise is assumed to be in both the image and model data.

constraint equations, minimizes the sum of squares of the perpendicular distances in the image plane between projected model end-points and the infinitely extended image line. This does not give more weight to distant lines, and also the error terms for each line is weighted by the inverse of the standard deviation of the expected image noise for optimal estimation.

Another algorithm "R\_and\_T\_mod" is developed which minimizes an error function in the image plane based on the "Infinite Model Line" constraint equation (2.26). In contrast to "R\_and\_T\_img", algorithm "R\_and\_T\_mod" minimizes the sum of squares of the perpendicular distance from the image line end-points to the infinitely extended projected model line. Section 2.6 presents results that show algorithm "R\_and\_T\_mod" performs significantly better than "R\_and\_T\_img" when there are lots of line breaks in the input image data. This is because image line breaks cause the alignment error from model end-point to the infinitely extended image line to be magnified, whereas the perpendicular distances from image end-point to the projected model lines are not so severely affected. If there is not significant noise along the length of the image lines (i.e. not severe line breaks) then the two algorithms perform comparably.

All the above algorithms are iterative and require initial estimates. "R\_and\_T", "R\_and\_T\_img" and "R\_and\_T\_mod" need initial estimates for both rotation and translation while "R\_then\_T" needs initial estimates just for rotation. For applications, for which no initial estimate is available, an algorithm is developed which samples the rotation space to provide initial estimates for the "R\_then\_T" algorithm. The output of this algorithm is then used as initial estimates for the "R\_and\_T\_img" or "R\_and\_T\_mod" algorithms. The sample which results in the smallest alignment error is returned as the best estimate. In practice, it is found that 12 initial samples of the rotation space suffice to find the optimal estimate.

A mathematical analysis of an uncertainty measure is developed which relates the variance in the output parameters to the noise present in the input parameters. In our experiments, it is assumed that the noise is the same for all lines. Given the input noise variance, the covariance matrix of the output parameters is computed. Results of Monte-Carlo analyses over thousands of experiments are reported in Section 2.6 and Section 4.2.1 and the experimentally derived covariance matrices are compared with the computed covariance matrix.

In certain scenarios there are strong prior expectations of the pose of the robot. For instance, in the hallway domain, the distance of the robot camera sensor from the floor is fairly constant over multiple frames. This prior expectation can be captured by an initial estimate with an associated prior covariance matrix. It is shown that this prior information about the pose estimate can easily be incorporated into the system of equations developed by the non-linear minimization technique used in this thesis. Effectively the optimization technique corresponds to an extended static Kalman filter where both new measurements and the initial estimate weighted with its covariance matrix are combined to make the new pose estimates.

Finally, the techniques and mathematical analysis developed in this chapter apply equally well to 3D/2D point data. In the point case, the total error between the projection of the model points to the extracted image points is to be minimized. In section 2.4, an objective function based on the constraint equations (2.6,2.7) is developed for point data. This function is minimized in the same manner as the optimization technique used for line data. Both rotation and translation parameters are solved for simultaneously and the algorithm is referred to as the "R\_and\_T" algorithm for points. The line and point algorithms can be extended to deal with combinations of point and line data. Similar comments can be made regarding the use of point data as were made for lines. For example, solving for rotation and translation si-



multaneously instead of separately gives much better results. Comparatively, a point algorithm using “n” points seems to be more robust than the corresponding line algorithm using “n” lines, although in any specific case, the results for both points and lines depends on the particular data set available.

## 2.4 Least-Squares Solution Methods

Ideally, if there was no noise, the minimal set of constraint equations (2.6, 2.7) or (2.11,2.26) for three sets of point or line correspondences respectively could be used to solve for the unknown pose parameters. The correct pose parameters would then cause perfect alignment between the projected model landmarks and the image measurements. However, in practice, measurements are noisy and perfect alignment cannot be realized. Therefore an objective function is minimized to find the pose parameters. The objective function is typically some function of the alignment error between the transformed model landmarks and image measurements. Most of the objective functions constructed in this thesis are non-linear in nature. Therefore a crucial factor in the choice of the objective function is the ability to construct suitable minimization algorithms for it. Suitability of minimization techniques is gauged by the following parameters: (1) speed of minimization, (2) convergence from a suitably large distribution of initial estimates, (3) numerical stability of the algorithm with respect to noise, finite length calculations etc..

In this section, a set of different objective functions for line and point data are presented. The objective functions are sum of squares of a function of the alignment error between each 3D model feature and its corresponding image measurement. To optimally estimate the pose (rotation and translation) parameters, the error for each landmark feature must be appropriately weighted based on the assumptions of noise in

the measurement process. In the least-squares algorithms presented in this section, it is assumed that the image measurements are corrupted with zero-mean independent gaussian noise<sup>4</sup>. Thus the error terms corresponding to each image line or point measurement are weighted by the inverse of the squared standard deviation of the measurement noise.

#### 2.4.1 Objective Functions for line data

The first set of objective functions developed is based on the line constraint equations (2.12) and (2.13):

$$E_R = \sum_{i=1}^n w_i (\hat{N}_i \cdot R(\hat{d}_i))^2 \quad (2.30)$$

$$E_1 = \sum_{i=1}^{2n} w_i (\hat{N}_i \cdot (R(\vec{p}_i) + \vec{T}))^2 \quad (2.31)$$

If there was no noise, the normal of the projection planes formed using the image lines would be perpendicular to the direction of the rotated model lines and thus the cosine of the angle between them would be zero.  $E_R$  is the sum of squares of these cosines.  $E_1$  is the weighted sum-of-the-squares of the perpendicular distances from the end-points of the 3D lines to their corresponding projection planes. For each line two 3D end-points are used and therefore, each line contributes twice to the objective function  $E_1$ . In these equations,  $w_i$  is the weight applied to the error for each line for optimal estimation.

In the “R\_then\_T” algorithm for line data, the rotation objective function  $E_R$  is minimized to solve for rotation “R”. The estimated rotation is used to minimize the objective function  $E_1$  to determine the translation “ $\vec{T}$ ”. In contrast, the “R\_and\_T” algorithm minimizes only “ $E_1$ ” to determine rotation “R” and translation “ $\vec{T}$ ” simultaneously. The particular non-linear iterative technique used to minimize these

---

<sup>4</sup>It is assumed in this chapter that the input noise in the 3D model is not significant.

functions and the ones discussed below, is given in Section 2.4.1.. Objective function  $E_1$  is a 3D alignment error between model line end-points and the infinitely extended image line. However, the significant noise is assumed to be in the image measurements. Thus to optimally weight the error terms in  $E_1$ , the weight terms  $w_i$  must be computed by reverse projecting the standard deviation value of the image noise onto the 3D projective plane. A similar “optimal” objective function can more simply be obtained by minimizing the sum of squares of the residual error function defined in the image plane:

$$E_2 = \sum_{i=1}^{2n} w_i \left( \frac{\vec{N}_i \cdot (R(\vec{p}_i) + \vec{T})}{(R(\vec{p}_i) + T)_z} \right)^2 \quad (2.32)$$

This error function is based on the constraint equation (2.11). The resulting algorithm which minimizes  $E_2$  is called “R.and.T\_img”. The objective function<sup>5</sup>  $E_2$  is the sum of squares of the perpendicular distance between projected model end-points to the infinitely extended image line. Figure (2.2) shows the above alignment error for one line pair.  $E_2$  is also a rational function and therefore it is more difficult to optimize. To minimize  $E_2$ , at each iteration in our non-linear technique, the denominator term  $(R(\vec{p}_i) + T)_z^2$  for each point is held constant (see section 2.4.1). In the next iteration, the denominator is updated with the new “R” and “ $\vec{T}$ ”. Therefore, we are able to employ the same algorithm as used for  $E_1$ . This seems to work for all the cases the algorithm has been run on. The same technique is applied for other error functions which have a denominator term.

In contrast to  $E_1$  and  $E_2$ , which are based on the “Infinite Image Line” constraints given in equations (2.12) and (2.11) respectively, an objective function ( $E_3$ ) is constructed based on the “Infinite Model Line” constraint equation (2.26):

$$E_3 = \sum_{i=1}^n \sum_{j=1}^2 \frac{w_i}{M_i^2} (R^T \left( \frac{I_{xj}}{s_x}, \frac{I_{yj}}{s_y}, 1 \right) \cdot (\vec{p}_{2i} - \vec{T}_w) \times (\vec{p}_{1i} - \vec{T}_w))^2 \quad (2.33)$$

---

<sup>5</sup>Note in  $E_2$  the normal vector  $\vec{N}$  is not a unit vector. It is defined by equation (2.10).

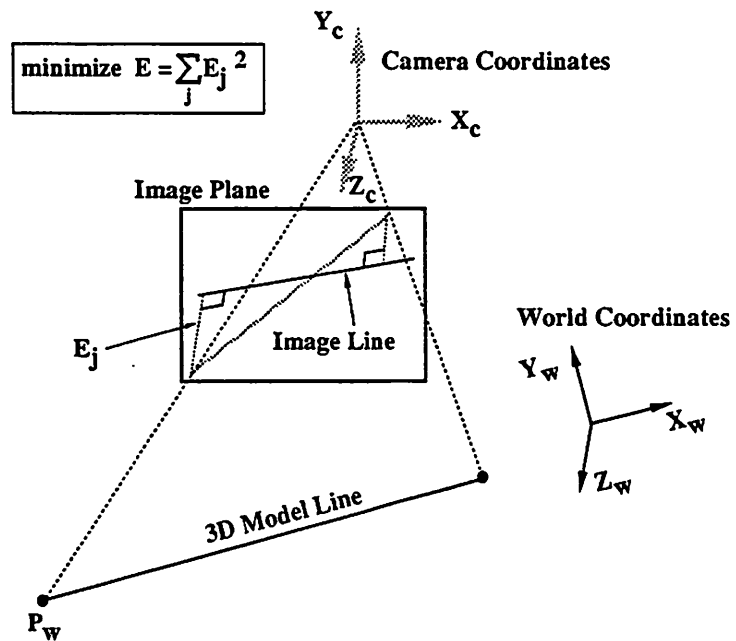


Figure 2.2: Error function based on the “Infinite Image Line” constraint.

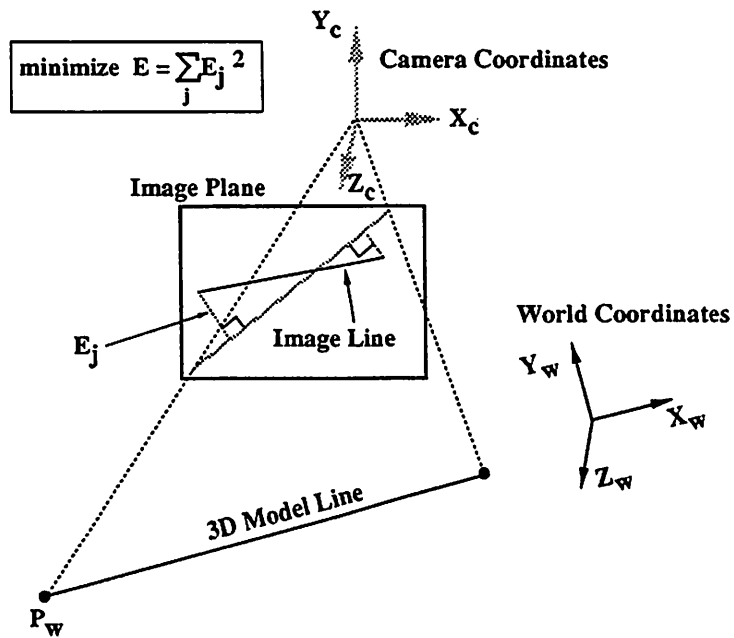


Figure 2.3: Error function based on the “Infinite Model Line” constraint.

where  $\vec{T}_w$  and  $R^T$  are the translation and rotation in the world coordinate system (2.2).  $M$  is defined in equation (2.21) in Section 2.2.  $E_3$  is the sum-of-the-squares of the perpendicular distances of the end-points of the image lines to the projected model line. Figure (2.3) shows the alignment error for one line pair. It is minimized by a method similar to the techniques used for  $E_1$  and  $E_2$ ; the algorithm which minimizes  $E_3$  is called "R.and.T.mod".

In Section 2.6, it is shown that algorithm ("R.and.T.mod") performs more robustly than algorithm ("R.and.T.img") when there is significant line fragmentation in the image data. This is because of the difficulty in optimally weighting the error terms in the objective function  $E_2$  minimized by algorithm ("R.and.T.img"). This can be understood by the discussion in the following paragraph on noise in the measurement of image lines.

The noise in the measurement of image lines can be decomposed into two components: noise in measuring the location of the image line end-points perpendicular to the line and noise in measuring the location of the image line end-points along the length of the line. We model the noise in locating the image line end-points perpendicular to the length of the line as a gaussian random variable. However, it seems intuitively not plausible that fragmentation of image lines is reasonably modeled as a gaussian process. In fact, the noise in locating line end-points along the length of the line seems significantly dependent upon factors such as the particular line extraction algorithm used, the image contrast across the line length, intensity structures neighboring the line in the image etc.. Thus it is difficult to develop a general model for the noise in locating line end-points along the length of the line.

In algorithm "R.and.T.img" the error in aligning model line segments with infinitely extended image lines is optimized (objective function  $E_2$ ). In order to derive the optimal weights for the error terms, the alignment error for each line must be

approximated by a function of the measurement noise. If the model line segments and the corresponding measured image line segments are of similar length, then the perpendicular noise component in locating image line end-points reasonably approximates the alignment error. However, if the model line segments are much larger or smaller than the measured image line segments<sup>6</sup>, the alignment error can only be “well” approximated by a function of both components of the noise in locating image line end-points. This, as we noted earlier is very difficult to do. In contrast, in algorithm “R\_and\_T\_mod”, the objective function  $E_3$  optimized is the error in aligning image line segments with infinitely extended model lines. The alignment error considered in this case depends only on the perpendicular noise component in locating the image line end-points. Thus, to optimally weight the error terms in  $E_3$  it is not necessary to know (or erroneously model) the measurement noise along the length of the image lines. Note, in experiments reported in Section 2.6, for both algorithms “R\_and\_T\_img” and “R\_and\_T\_mod”, the weights used in the respective objective functions are based only on the perpendicular noise components.

#### 2.4.2 Objective Functions for point data

For point data, we could draw virtual lines between all pairs of points and then use any of the line algorithms. This however would give us an objective function which has  $O(n^2)$  terms where “ $n$ ” is the number of points. Instead the following objective function based on constraint equations (2.6) and (2.7) has only  $O(n)$  terms:

$$E_{p1} = \sum_{i=1}^n \frac{w_i}{(R(\vec{p}_i) + \vec{T})_z^2} (((s_x, 0, -I_{ix}) \cdot (R(\vec{p}_i) + \vec{T}))^2 + ((0, s_y, -I_{iy}) \cdot (R(\vec{p}_i) + \vec{T}))^2) \quad (2.34)$$

$E_{p1}$  is the sum of squares of the distance of the projected model point to the measured image point in the image plane.  $E_{p1}$  is again minimized by keeping the denominator

---

<sup>6</sup>Note, this would be the case when image lines are highly fragmented.

$(R(\vec{p}_i + \vec{T}))_z^2$  constant during an iteration and then updating it for the next iteration using the new “R” and “ $\vec{T}$ ”. The resulting algorithm is referred to as the “R\_and\_T” algorithm for point data.

$E_1, E_2, E_3, E_R$  and  $E_{p_1}$  are all minimized by modifying the same basic non-linear technique. Therefore, only the technique for minimizing  $E_1$  is presented in the next section. For algorithm “R\_then\_T”,  $E_1$  is minimized by a straight-forward linear least squares algorithm. Appendix A discusses various representations for rotation and motivates our particular choice of quaternions for large rotations and the 3D rotation vector for small angles.

### 2.4.3 Non-linear Technique for “R\_and\_T”

To minimize “ $E_1$ ”, we adapt an iterative technique formulated by Horn [35] to solve the problem of relative orientation. An initial estimate is required for both “R” and “ $\vec{T}$ ”. The technique linearizes the error terms about the current estimate for “R” and “ $\vec{T}$ ”. At each iteration, the linearized error function is minimized to determine adjustment vectors for the rotation and translation terms. The iterative adjustments are made to the rotation and translation terms until the objective function “ $E_1$ ” converges to a minimum. Note that the algorithm, like all such descent algorithms, does not guarantee a global minimum.

Assume we have a current estimate “R” for rotation. The coordinates  $\vec{p}_i'$  of a rotated 3D point is given by  $\vec{p}_i' = R(\vec{p}_i)$ . An incremental rotation vector  $\delta\omega$  is added to the rotation estimate “R”; the direction of this incremental vector is parallel to the axis of rotation, while its magnitude is the angle of rotation.

This incremental rotation takes  $\vec{p}_i'$  to  $\vec{p}_i''$ :

$$\vec{p}_i'' = \vec{p}_i' + \delta\omega \times \vec{p}_i' \quad (2.35)$$

This follows from Rodrigue’s formula [35] for the rotation of a vector  $r$  to  $r'$  by angle

“ $\theta$ ” about the three dimensional axis vector “ $\omega$ ”:

$$r' = r(\cos\theta) + \sin\theta(\omega \times r) + (1 - \cos\theta)(\omega \cdot r)\omega \quad (2.36)$$

where  $\theta = \|\delta\omega\|$  and  $\omega = \delta\omega/\|\delta\omega\|$

Let  $\Delta\vec{T}$  represent a small translation added to the current translation estimate  $T$ . Thus, the linearized energy function about the current estimate “ $R$ ” and “ $\vec{T}$ ” becomes:

$$E = \sum_{i=1}^{2n} w_i (\vec{N}_i \cdot (\vec{p}_i' + \delta\vec{\omega} \times \vec{p}_i' + T + \Delta\vec{T}))^2 \quad (2.37)$$

Let  $\vec{b}_i = \vec{p}_i' \times \vec{N}_i$ . Using the chain rule of triple scalar product for vectors, differentiating the objective function with respect to  $\Delta\vec{T}$  and  $\delta\vec{\omega}$  respectively, and setting the results equal to 0, the following two equations are obtained after some manipulation:

$$\sum_{i=1}^{2n} w_i (\vec{N}_i \cdot \Delta\vec{T} + \delta\vec{\omega} \cdot \vec{b}_i) \vec{N}_i = - \sum_{i=1}^{2n} w_i (\vec{N}_i \cdot (\vec{p}_i' + \vec{T})) \vec{N}_i \quad (2.38)$$

$$\sum_{i=1}^{2n} w_i (\vec{N}_i \cdot \Delta\vec{T} + \delta\vec{\omega} \cdot \vec{b}_i) \vec{b}_i = - \sum_{i=1}^{2n} w_i (\vec{N}_i \cdot (\vec{p}_i' + \vec{T})) \vec{b}_i \quad (2.39)$$

Together, these two vector equations constitute 6 linear scalar equations in the 6 unknown components of  $\Delta\vec{T}$  and  $\delta\vec{\omega}$ . They can be rewritten in the more compact matrix form:

$$A \begin{bmatrix} \Delta\vec{T} \\ \delta\vec{\omega} \end{bmatrix} = \vec{f} \quad (2.40)$$

In the above equation,  $A$  is a 6 x 6 matrix and  $f$  is a 6 x 1 vector.  $A$  and  $\vec{f}$  are defined in the following manner:

$$A = \begin{bmatrix} C & F \\ F^T & D \end{bmatrix} \quad (2.41)$$

$$\vec{f} = \begin{bmatrix} \vec{c} \\ \vec{d} \end{bmatrix} \quad (2.42)$$

$$\text{where } C = \sum_{i=1}^{2n} w_i \vec{N}_i \vec{N}_i^T \quad D = \sum_{i=1}^{2n} w_i \vec{b}_i \vec{b}_i^T \quad F = \sum_{i=1}^{2n} w_i \vec{N}_i \vec{b}_i^T$$



while  $\bar{c} = \sum_{i=1}^{2n} w_i (\vec{N}_i \cdot (\vec{p}_i' + \vec{T})) \vec{N}_i$  and  $\bar{d} = \sum_{i=1}^{2n} w_i (\vec{N}_i \cdot (\vec{p}_i' + \vec{T})) \vec{b}_i$ .

Solving the above set of 6 linear equations gives a way of finding small changes in rotation and translation that reduce the overall objective function. The algorithm can therefore be expressed in the following four steps.

**Step 1** Given an initial estimate for rotation “R” and translation “ $\vec{T}$ ” and a list of correspondences between 3D modeled lines and their image measurements.

**Step 2** Compute the coefficients of the matrices in equation (2.40). Solve the linear system for  $\Delta \vec{T}$  and  $\delta \vec{\omega}$ .

**Step 3** Compose  $\delta \vec{\omega}$  with the current estimate R of rotation to get the new estimate. Add  $\Delta \vec{T}$  to  $\vec{T}$  to get the next estimate for translation.

**Step 4** Stop if the algorithm has converged or has exceeded a maximum number of iterations, else go back to Step 2.

The current rotation estimate “R” is represented as a quaternion. At each iteration, the rotation increment  $\delta \vec{\omega}$  is transformed into a quaternion and composed with the current estimate to form the new rotation estimate [35]. The iteration procedure is terminated when either a maximum number of iterations is exceeded or when the difference in the result between two successive iterations is less than a pre-specified minimum. The computational complexity of the algorithm is  $O(kn)$  where there are “n” points or lines and “k” iterations are needed to converge to the optimal solution.

The covariance matrix  $\Lambda_P$  of the estimated pose parameters is related to the coefficient matrix A (defined in equation 2.41):

$$\Lambda_P = A^{-1} \quad (2.43)$$

The above formula for the covariance matrix is only valid if the residual error terms of the objective function are optimally weighted. The covariance matrix is computed

using the final pose parameters estimated at the last iteration. The matrix  $A$  is known as the information matrix. It becomes singular when there are an infinite number of solutions e.g. a data set of less than three lines, all lines are parallel, all lines meet at a point etc.. For these cases, incremental adjustments to the current pose estimate cannot be computed. In Chapter 4, we will present an alternative method to derive the expression for the covariance matrix.

### Performance of the Non-linear Algorithm

The non-linear algorithm described above requires the user to specify an initial estimate for translation and rotation. How close the initial estimate must be to the final values, in order to ensure convergence, depends on the particular data set. The algorithm seems to converge for initial estimates which differ considerably from the correct solution. Generally, the rotation estimate is more important than the translation estimate. For some data sets, convergence seems to be almost independent of the starting point; for others the initial rotation estimate must be within 40 degrees for all the three Euler angles representing the rotation.

Another important question asked about non-linear iterative algorithms is speed of convergence. In our experiments, for "good" data sets of about 10 or more lines, the algorithm typically converges in 3 or 4 iterations for initial estimates that may be more than 40 degrees off in rotation and 100 feet off in translation. For instance, Figure (2.4) shows an initial set of input image lines used for an experiment to demonstrate the convergence properties of the above minimization technique. Figure (2.4) is the first frame of a set of outdoor images on which the pose algorithms were tested. Figure (2.5) is the projection of the model using the initial estimate of the pose parameters. The initial estimate is off by more than 100 feet in translation along the walkway direction, 20 feet under the walkway in the vertical direction and about 15

degrees off from the axis for rotation. Figure (2.6) shows the projection of the model using the estimate of the pose parameters obtained after the first iteration. The pose is now within 10 feet of the correct answer. Figure (2.7) shows the projection of the model using the estimate of the pose parameters after the second iteration; the pose is now almost correct. The algorithm converges in the third iteration and the projection using the final estimate is shown in Figure (2.8). Note that in Figure (2.8) additional model lines have been projected to show the accuracy of the final projection.

Finally, for certain near singular data sets, it is possible that the technique can diverge rather than converge. Iterative minimization techniques can be looked upon as moving in a multi-dimensional space, searching for the bottom of the nearest valley. Ideally, at each iteration or move, the function value would decrease until the valley bottom is reached. The Gauss-Newton minimization method adopted here is a second-order technique. Second-order methods have the property of extremely fast convergence under normal conditions. Unlike first-order (or gradient based) methods, they are not guaranteed to descend in every iteration.

For the near singular cases where the technique might diverge, a simple solution exists to guarantee convergence. This solution is motivated by the Levenberg-Marquardt method [64] for minimizing non-linear functions and its application to this minimization technique for pose was first noted by Lowe [56]. The Levenberg-Marquardt method combines first- and second-order methods. At any current iteration, the increment is first calculated by second-order methods. If the new estimate causes the objective function to increase, then the increment is re-estimated by adding a component of the gradient to the old estimate of the increment. Note that moving along the gradient guarantees descent and hence convergence. But gradient-descent algorithms are slow to converge. The Levenberg-Marquardt method attempts to combine the best of both methods by moving along directions close to the gradient only if moving

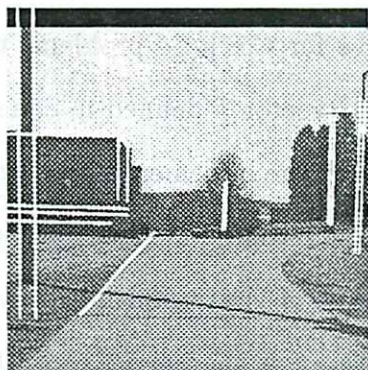


Figure 2.4: Image Lines for Outdoor Frame 1 used for convergence experiment.

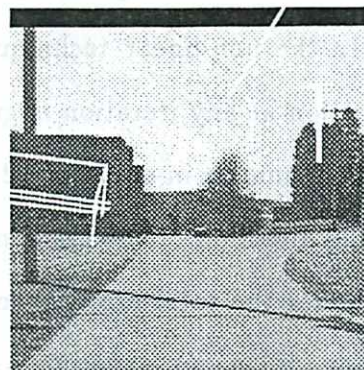


Figure 2.5: Projection of model using initial estimate for convergence experiment.

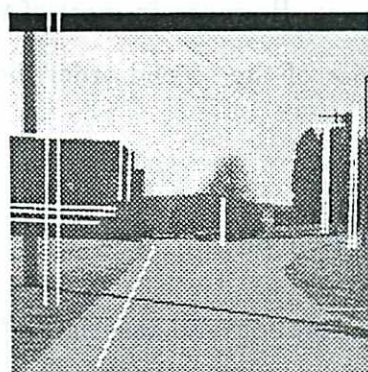


Figure 2.6: Projection of model using estimate after first iteration for convergence experiment.

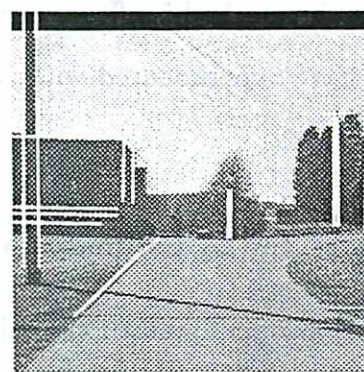


Figure 2.7: Projection of model using estimate after second iteration for convergence experiment.

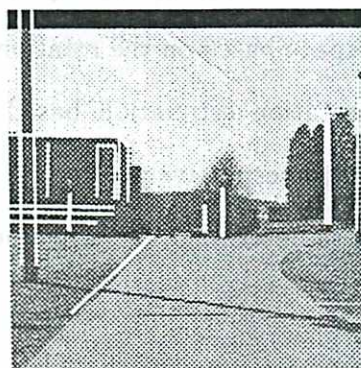


Figure 2.8: Projection of model using estimate after third and final iteration for convergence experiment.

in the direction computed by the second-order method causes divergence.

In the non-linear technique described above for the pose problem, the gradient direction at any iteration is given by the  $(6 \times 1)$  vector  $\vec{f}$  defined in equation (2.40). If the  $A$  matrix in equation (2.40) is diagonal then the incremental move calculated is in the direction of the gradient. Thus, if the new estimate at the end of an iteration increases the total error, then the diagonal terms of  $A$  are multiplied by a constant factor (10 in our experiments). This biases the new increment to be towards the gradient direction. This procedure is repeated until the error function decreases after the addition of the increments to the current estimate. Experiments have proven that the method is effective in forcing convergence. However, in most of the data sets we have experimented with, divergence behavior of the second-order method is not observed.

#### 2.4.4 Initial Estimates of Rotation and Translation

For some applications, initial estimates for rotation and translation may not be available. In that case, the rotation space can be sampled; each of the samples is used as an initial estimate for the rotation estimation part of "R\_then\_T". The rotation and translation estimates made by the algorithm "R\_then\_T" are then used as initial estimates for "R\_and\_T". We have successfully tried this procedure with 12 uniform samples of the rotation space based on the rotation group of a tetrahedron. Appendix B provides the initial rotation estimation based on sampling the tetrahedron and octahedron rotation groups respectively.

The algorithm for pose determination, when no initial estimate is available, is the following:

**Step 1:** Pick a rotation estimate from a uniform sampling of the rotation space.

(Refer to Appendix B.)

**Step 2:** Run Algorithm “R\_then\_T” to get estimates for both “R” and “ $\vec{T}$ ”.

**Step 3:** Use the output estimates from Step 2 as initial estimates to algorithm “R\_and\_T”.

**Step 4:** Repeat Steps 1-3 until all rotation samples have been used.

**Step 5:** Return the estimate which gives the smallest alignment error in Step 3 as final estimate.

It is possible to speed up the computation by running algorithm “R\_then\_T” in step 2 for all initial rotation samples. The best<sup>7</sup> estimate is then used as the initial estimate for a single run of “R\_and\_T”. In practice, this has been found to be good enough most of the time. Finally, in Step 3, the algorithms “R\_and\_T\_img” or “R\_and\_T\_mod” can replace “R\_and\_T”.

#### 2.4.5 Prior Estimates

In many applications a prior estimate for the pose parameters is available; in many case a prior covariance (or uncertainty) matrix of the pose parameters is also available. For instance, when tracking independently moving objects their location and orientation can be predicted by incorporating a motion model such as constant velocity or constant acceleration. Similarly in the mobile robot navigation domain, the location of the robot can be predicted by dead-reckoning. Some pose parameters may remain relatively constant. The height of a robot navigating indoor hallways may change only by negligibly small amounts. The strength of belief in such predictions on the robot pose are captured by the prior covariance matrix. In this section, the iterative methods developed earlier to estimate pose are extended to handle the

---

<sup>7</sup>Best is defined in terms of final alignment errors.

information represented by the prior estimate and its covariance matrix. The basic tool used is akin to Extended Kalman Filtering [26].

Let the initial estimate be denoted by the (6 x 1) vector  $\vec{Q}$  and the associated prior covariance matrix by the (6 x 6) matrix  $\Lambda_Q$ . In the iterative system developed earlier, a linear system of equations (2.40) is solved at each iteration to determine the pose increments  $\Delta\vec{T}$  for translation and rotation  $\delta\vec{\omega}$ . To incorporate the additional information available in the covariance matrix, this linear system of equations is modified in the following way:

$$(A + \Lambda_Q^{-1}) \begin{bmatrix} \Delta\vec{T} \\ \delta\vec{\omega} \end{bmatrix} = \vec{f} - \Lambda_Q^{-1}(\vec{\theta} - \vec{Q}) \quad (2.44)$$

In the above equation,  $\vec{\theta}$  is the current pose estimate. Note that  $A$  is a (6 x 6) matrix while  $\vec{f}$  and  $\vec{\theta}$  are (6 x 1) vectors. Therefore, when prior estimates and covariance matrices are available equation (2.44) instead of equation (2.40) is solved at each iteration. The output covariance matrix of the pose parameters is given by  $(A + \Lambda_Q^{-1})^{-1}$  evaluated at the final pose estimate. In the Section 2.6, we shall describe monte-carlo experiments where the pose estimation is done with initial estimates having high and low initial prior covariance matrices respectively.

## 2.5 Results Using Point Data

In this section, pose estimation results using point data are presented for the A211 room sequence. The A211 sequence was generated by taking images from a camera mounted on a mobile robot. The robot was translated roughly along the optical axis of the camera and 10 image frames (Frame 1 to Frame 10) were taken; the translation for each step is approximately 0.38 feet. The field of view of the camera was approximately  $29.27^\circ \times 22.86^\circ$ . Objects in the scene ranged from 8 feet to 20 feet away in the first image frame. The depths of some corner points was measured by

using a tape measure. Using the image measurement of these points, a 3D model was built in the first frame's coordinate system. The corner points were tracked over the 10-frame sequence using William's [76] optic-flow based line tracking system.

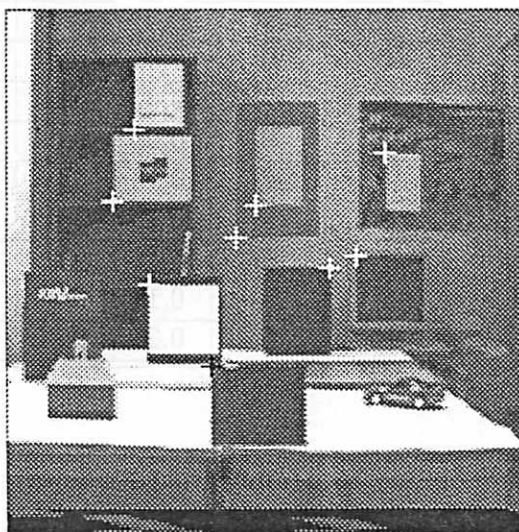


Figure 2.9: A211 sequence Frame 1.

Figure 2.9 shows frame 1 of the sequence. The 9 points used to compute the pose are marked by crosses in the figure. The location estimation results for frame 2 to frame 10 are presented in Table 2.1. In the table, the x and y-axes are parallel to the image plane (see Figure 2.9) and the z-axis is the optical axis. Ground truth was not available for the orientation estimate. For the location estimation, the ground truth was obtained from dead-reckoning. The exact amount of movement in the forward direction was measured again by tape measure. However no accurate measurement was made of any sideways drift in the robot motion. As can be seen from Table 2.1 the robot is located to within 0.21 feet for all 9 frames. However, there is a systematic increase in error in location along the x-axis from frame 2 to frame 10. We believe



Table 2.1: Estimated Errors of Location in world coordinates for “R\_and\_T” algorithm for point data. Real Data results for A211 sequence.

Frame No.	Num. points	Location Error		
		$\Delta L_x$ feet	$\Delta L_y$ feet	$\Delta L_z$ feet
2	9	-0.01	0.01	0.01
3	9	0.02	-0.02	0.04
4	9	0.07	-0.03	0.03
5	9	0.09	-0.04	0.04
6	9	0.13	-0.05	0.03
7	9	0.16	-0.05	0.01
8	9	0.16	-0.06	0.00
9	9	0.19	-0.06	0.00
10	9	0.21	-0.06	0.06

this systematic error is due to a unmeasured sideways drift in the robot from the specified forward motion. Note along the measured forward direction (z-axis) and the vertical direction (y-axis) the robot is located within 0.06 feet.

Finally, we report results for frame 2 when no initial pose estimate is available. In this case, the algorithm developed in Section 2.4.2 is used. Twelve rotation samples were used as input for the “R\_then\_T” algorithm the output of which was fed to the “R\_and\_T” algorithm. Finally, the estimate with the smallest average fitting error for all nine points was chosen as the optimal estimate. Table 2.2 shows the estimated pose for each of the rotation samples. The rotation samples and the final rotation estimates are specified as quaternions. The rotation samples are based on the tetrahedron group (see Appendix B). Each of the twelve initial rotation samples led to one of four final pose estimates (P0 – P3). The translation and rotation values of each of the pose estimates is given in the caption of Table 2.2. As can be seen from the table, pose estimate P0 has the lowest average fitting error of 0.26 pixels and it

Table 2.2: Estimated Pose for 12 different input rotation samples using a combination of algorithms “R\_then\_T” and “R\_and\_T” (point) data. Results for frame 2 of A211 sequence; the 12 initial rotation samples for “R\_then\_T” lead to one of the following four pose estimates:

P0: Translation ( 0.01 -0.02 0.40) Rotation (0.9999 -0.0017 -0.0018 -0.0010)

P1: Translation (-0.18 7.00 31.6) Rotation (0.2041 -0.9735 -0.1010 0.0179)

P2: Translation ( 0.40 -2.19 3.95) Rotation (0.0027 0.0869 -0.9891 0.1186)

P3: Translation ( 1.36 -1.69 37.0) Rotation (0.0076 -0.0376 0.0476 -0.9981)

Trial No.	INITIAL ROTATION QUATERNION				FINAL POSE Estimated	FITTING ERROR pixels
	$Q_0$	$Q_1$	$Q_2$	$Q_3$		
1	1.0	0.0	0.0	0.0	P0	0.26
2	0.0	1.0	0.0	0.0	P1	37.88
3	0.0	0.0	1.0	0.0	P2	36.66
4	0.0	0.0	0.0	1.0	P3	10.49
5	0.5	0.5	0.5	0.5	P3	10.49
6	-0.5	0.5	0.5	0.5	P1	10.49
7	0.5	-0.5	0.5	0.5	P0	0.26
8	0.5	0.5	-0.5	0.5	P2	36.66
9	0.5	0.5	0.5	-0.5	P3	10.49
10	0.5	0.5	-0.5	-0.5	P0	0.26
11	0.5	-0.5	0.5	-0.5	P2	36.66
12	0.5	-0.5	-0.5	0.5	P1	37.87

was obtained from three of the rotation samples. This is the correct pose for image frame 2 and corresponds to an forward motion of 0.38 feet from the position of the robot at frame 1.

## 2.6 Results Using Line Data

The development of the algorithms presented in this thesis are part of a larger effort to enable the UMASS robot “Harvey” to navigate the sidewalks and interior

hallways of a part of the UMASS campus [20]. Consequently, results using line data are presented for both indoor hallway images and outdoor sidewalk images. Figures 2.10 and 2.13 are examples of the outdoor images. Figures 2.16 and 2.19 are examples of indoor hallway images.

The indoor model was built by measuring distances with a tape measure and is accurate to approximately 0.1 feet [20]. The outdoor 3D model was built over two passes. In the first pass, blueprints of the campus, drawn to a scale of 40 feet to an inch, were used. Errors of up to 10 feet were found in the resulting 3D model; errors of this magnitude are unacceptable for our navigation goals. An error of 1 foot in the location of a 3D landmark, 50 feet away from the camera, can cause its projection to be displaced by 24 pixels in the image. In the second pass, landmarks were surveyed using theodolites. We believe most of our 3D model is now accurate to within 0.3 feet. Some landmarks, such as poles and posts, are difficult to position accurately, because of their cylindrical shape and their lack of any distinguishing points.

The images were acquired using a Sony B/W model AVC-D1 camera mounted on the robot vehicle. Linked to a Gould frame grabber, 512 by 484 pixel images are obtained, with field of view of  $24.0^\circ$  by  $23.0^\circ$ . Calibration was not done for the image center; it was assumed to be at the frame center. In Chapter 5, the sensitivity of pose determination to errors in the estimate of intrinsic camera parameters is studied. It is shown that errors in estimation of the image center do not affect the location of the camera in a world coordinate system.

Experiments were conducted on both real image data and simulated data with noise added to it. The landmarks used for the outdoor scene experiments were the 3D lines forming the visible corner of the building, window lines, lampposts, telephone poles and one sidewalk line (see Figure 2.4). The experiments for both synthetic and real data for the outdoor scenes were conducted with the camera about 300 feet from

the building in Figure 2.4. The synthetic data experiments were conducted with projections of 3D lines from the model. The camera was assumed to be placed at the same location as the first frame of the real data experiments. For that frame, there was one telephone pole 50 feet away from the camera. The rest were in the range of 150 to 300 feet away. For indoor hallway scenes (Figure 2.16) the camera was anywhere from 40 to 23 feet from the door at the far end of the hallway. A typical set of indoor landmarks used for both the synthetic and real data experiments can be seen by the highlighted lines in Figure 2.16.

### 2.6.1 Synthetic Data Results for “R\_and\_T” and “R\_then\_T”

The synthetic data experiments were conducted for both algorithms “R\_and\_T” and “R\_then\_T” using the outdoor 3D model. The two algorithms were run with four different sets of data lines, each set being perturbed by at least two different amounts of noise. Zero mean uniform noise was added to the  $\rho$  and  $\theta$  of each image line. In Tables 2.3 and 2.4 the noise for each simulation is specified in the  $\rho$  and  $\theta$  columns. One pixel noise in  $\rho$  means that to the correct  $\rho$  of each line, we added a  $\Delta\rho$ , which was a random number anywhere in the range  $[-1,+1]$ . Similarly, one degree of noise in  $\theta$  means that to the correct  $\theta$  of each line, we added a  $\Delta\theta$ , which was a random number anywhere in the range  $[-1,+1]$ . Simulations were performed for a maximum of  $1^\circ$  or  $5^\circ$  noise in  $\theta$ , and a maximum of 1 pixel or 5 pixel noise in  $\rho$ . For each set of lines and each specification of input noise, 100 data samples were created; for the generation of each sample the random number generator was initialized with a different seed point.

The results presented in the tables are the average absolute error of the computed rotation and translation over these 100 data samples, for each set of lines and each noise specification. The results for the “R\_and\_T” and “R\_then\_T” algorithms are

shown in Table 2.3 and Table 2.4, respectively. Rotation and translation errors in these tables for synthetic data are specified with respect to the camera coordinate system (Figure 2.1). The rotation errors are specified in terms of the error in degrees of the axis-angle 3D rotation vector.  $\Delta T_x$  corresponds to error in translation in the direction along the rows in the image plane (in camera coordinates, see Figure 2.4).  $\Delta T_y$  corresponds to error in translation in the vertical direction in camera coordinates.  $\Delta T_z$  corresponds to error in translation along the direction of the optical axis in camera coordinates.

The first set of 5 lines consisted of the 4 corner edges of the building visible in Figure 2.4 and one window line in that same building. The second set of 10 lines consisted of the 4 corner edges of the building, as above, and 6 lampposts and telephone pole lines. The third set of 14 lines consisted of these 10 lines plus three more lines on the building and one side walk line. The fourth set of 30 lines consisted of the above set of 14 lines plus a set of 16 virtual lines that were drawn between 6 real vertices in the scene.

A comparison of the results shows that the performance of the “R\_and\_T” algorithm (Table 2.3) is much better than the “R\_then\_T” algorithm (Table 2.4). With zero noise specified, both algorithms gave the correct result. For each set of lines and each specification of noise, “R\_and\_T” performs much better than “R\_then\_T”. The results for “R\_then\_T” are particularly bad for the 5 and 10 line simulations. This can be explained by the observation that, in the 5 line case, 3 of the lines form a trihedral junction. As noted before, trihedral junctions can give rise to an infinite number of translations. Thus, the translation result is determined from this infinite set solely by the two remaining lines, both of which are vertical and not too far from each other. With noise, therefore, we would expect large errors in translation. Similarly, in the 10 line simulation, most of the lines are vertical. Vertical lines do not disambiguate

rotations about the x-axis and translations along the y-axis. This problem is compounded even further when the rotation stage is separated from the translation stage as is the case in algorithm “R\_then\_T”.

In the results for both algorithms, the error decreases appreciably as the number of lines increases. In the “R\_and\_T” case (Table 2.3) results are shown for experiments with the 5 line data set for two extra cases of noise. Examination of the results for these two cases in Table 2.3 shows an appreciably larger error when the noise in  $\theta$  is  $5^\circ$ . However, when the noise in  $\rho$  is 5 pixels and the noise in  $\theta$  is  $1^\circ$ , the errors are much smaller; in general noise in  $\theta$  for lines is much more harmful than noise in  $\rho$ . Finally, in all experiments, the error in  $\Delta T_y$  was often found to be larger than the errors in  $\Delta T_x$  and  $\Delta T_z$ . This is due to the fact that in the data sets for these experiments, the majority of the 3D lines are vertical.

## 2.6.2 Results for “R\_and\_T\_mod” and “R\_and\_T\_img”

### Synthetic Data

In this subsection, synthetic data results for algorithms “R\_and\_T\_mod” and “R\_and\_T\_img” are discussed. The results for the experiments are presented in Table 2.5. The 3D line model used for these experiments is same as that built for the indoor hallway images (see Figure 2.16). The camera was assumed to be 40 feet from the door in Figure 2.16. The field of view and other intrinsic camera specifications are the same as those used for the the synthetic data experiments discussed in the previous section. Given an input pose, the 3D model of 10 lines is projected to create a set of 2D data lines. The end-points of the 2D lines are then corrupted by 2D gaussian noise. The gaussian noise has two components; the first is perpendicular to the 2D line and the second is along the length of the line. Noise for each end-point was assumed to be independent. For all the experiments reported in Table 2.5 the

Table 2.3: Average Absolute Error of Translation and Rotation in camera coordinates for algorithm “R\_and\_T”. The average for each experiment is taken over 100 samples of uniform noise.

NOISE			ROTATION ERROR			TRANSLATION ERROR		
No. Lines	$\theta$ deg.	$\rho$ pixels	$\delta\omega_x$ deg.	$\delta\omega_y$ deg.	$\delta\omega_z$ deg.	$\Delta T_x$ feet	$\Delta T_y$ feet	$\Delta T_z$ feet
Correct			0.00	0.00	0.00	0.00	0.00	0.00
5	1.0	1.0	0.24	0.15	0.04	0.21	2.03	1.16
5	5.0	5.0	1.20	0.79	0.19	1.08	10.14	6.20
5	1.0	5.0	0.24	0.16	0.04	0.21	2.04	1.18
5	5.0	1.0	1.19	0.78	0.19	1.08	10.14	6.20
10	1.0	1.0	0.21	0.08	0.05	0.02	1.73	0.08
10	5.0	5.0	0.72	0.27	0.31	0.18	6.33	0.48
14	1.0	1.0	0.07	0.06	0.08	0.03	0.77	0.02
14	5.0	5.0	0.34	0.30	0.39	0.17	3.80	0.12
30	1.0	1.0	0.03	0.05	0.06	0.06	0.48	0.06
30	5.0	5.0	0.16	0.24	0.31	0.32	2.39	0.32

Table 2.4: Average Absolute Error of Translation and Rotation in camera coordinates for algorithm “R\_then\_T” The average for each experiment is taken over 100 samples of uniform noise.

NOISE			ROTATION ERROR			TRANSLATION ERROR		
No. Lines	$\theta$ deg.	$\rho$ pixels	$\delta\omega_x$ deg.	$\delta\omega_y$ deg.	$\delta\omega_z$ deg.	$\Delta T_x$ feet	$\Delta T_y$ feet	$\Delta T_z$ feet
Correct			0.00	0.00	0.00	0.00	0.00	0.00
5	1.0	1.0	1.08	5.06	0.62	11.44	13.96	51.16
5	5.0	5.0	3.19	14.65	1.62	32.69	39.85	149.40
10	1.0	1.0	0.50	2.26	0.31	9.24	8.83	8.84
10	5.0	5.0	2.44	10.45	1.28	40.83	40.03	38.65
14	1.0	1.0	0.29	0.29	0.18	0.35	2.37	0.23
14	5.0	5.0	1.50	1.56	0.91	1.92	12.44	1.27
30	1.0	1.0	0.09	0.10	0.13	0.40	1.01	0.36
30	5.0	5.0	0.45	0.50	0.66	2.09	5.05	1.82

standard deviation of the component of the noise perpendicular to the line was 1 pixel. The standard deviation of the component of the noise along the length of the line is specified as percentage of the length of the line. This is reported in column 2 in Table 2.5. For each noise specification, 1000 noisy sample sets are created and the two algorithms run on each of the noisy 2D data sets. First-order and second-order statistics are collected for each set of 1000 runs. From the first order statistics (i.e. estimation of the mean) it is observed that the final estimates are unbiased. The second-order statistics are the experimentally derived covariance matrices of the output pose parameters. The square root of the diagonal values (or standard deviations) of each of the pose parameters for each noise specification and each algorithm are reported in Table 2.5.

From Table 2.5, it can be seen that the two algorithms perform comparably as long the component of noise along the length of the line is small. But when the standard deviation of the noise along the length of the line becomes 20 % or more of the length, then the results for algorithm "R\_and\_T\_mod" sharply improve over algorithm "R\_and\_T\_img". This confirms what was predicted in the discussion in Section 2.4.

In Section 2.4.4, the use of prior knowledge in estimating the pose parameters was discussed. Table (2.6) shows the advantage of having prior knowledge. Prior knowledge of the location of the robot is specified in terms of an input estimate and an associated covariance matrix. The same experiment as described above is repeated for both algorithms "R\_and\_T\_img" and "R\_and\_T\_mod". In this case, however, the initial estimate is supplied with a small prior covariance matrix. For these experiments, the input covariance matrix is assumed to be a diagonal matrix. Thus only the diagonal values corresponding to the square of the input standard deviation of the pose parameters needed to be specified. As noted before for hallway scenes, the height



Table 2.5: Standard deviation of Translation and Rotation error in world coordinates for algorithms “R\_and\_T\_img” and “R\_and\_T\_mod” with high prior covariance estimates for translation. The statistics for each experiment is taken over 1000 samples of gaussian noise.

NOISE			ROTATION ERROR			TRANSLATION ERROR		
No. Lines	length %	perp. pixels	$\delta\omega_x$ deg.	$\delta\omega_y$ deg.	$\delta\omega_z$ deg.	$\Delta T_x$ feet	$\Delta T_y$ feet	$\Delta T_z$ feet
Algorithm “R_and_T_img”								
Prior Estimate			57.30	57.30	57.30	94.86	94.86	94.86
10	1.0	1.0	2.77	0.46	2.48	0.25	0.82	0.70
10	5.0	1.0	2.79	0.47	2.50	0.25	0.83	0.70
10	10.0	1.0	2.83	0.48	2.53	0.26	0.84	0.71
10	20.0	1.0	3.03	0.54	2.71	0.27	0.89	0.77
10	30.0	1.0	6.68	1.35	5.27	0.96	1.72	1.49
10	40.0	1.0	10.33	2.32	8.45	2.00	2.44	2.12
Algorithm “R_and_T_mod”								
Prior Estimate			57.30	57.30	57.30	94.86	94.86	94.86
10	1.0	1.0	2.69	0.46	2.40	0.25	0.79	0.68
10	5.0	1.0	2.69	0.46	2.40	0.25	0.79	0.68
10	10.0	1.0	2.70	0.46	2.40	0.25	0.80	0.68
10	20.0	1.0	2.79	0.48	2.48	0.27	0.82	0.70
10	30.0	1.0	2.92	0.48	2.59	0.28	0.86	0.73
10	40.0	1.0	3.06	0.45	2.69	0.27	0.90	0.76

Table 2.6: Standard deviation of Translation and Rotation error in world coordinates for algorithms “R\_and\_T\_img” and “R\_and\_T\_mod” with low prior covariance estimates for translation. The statistics for each experiment is taken over 1000 samples of gaussian noise.

NOISE			ROTATION ERROR			TRANSLATION ERROR		
No. Lines	length %	perp. pixels	$\delta\omega_x$ deg.	$\delta\omega_y$ deg.	$\delta\omega_z$ deg.	$\Delta T_x$ feet	$\Delta T_y$ feet	$\Delta T_z$ feet
Algorithm “R_and_T_img”								
Prior Estimate			57.30	57.30	57.30	3.0	3.0	0.001
10	1.0	1.0	0.30	0.44	0.19	0.24	0.12	0.0
10	10.0	1.0	0.30	0.45	0.20	0.24	0.12	0.0
10	20.0	1.0	0.34	0.51	0.23	0.25	0.14	0.0
10	30.0	1.0	0.70	1.11	0.52	0.39	0.30	0.0
10	40.0	1.0	0.83	1.51	0.67	0.77	0.37	0.0
Algorithm “R_and_T_mod”								
Prior Estimate			57.30	57.30	57.30	3.0	3.0	0.001
10	1.0	1.0	0.30	0.44	0.19	0.24	0.12	0.0
10	10.0	1.0	0.30	0.44	0.19	0.24	0.18	0.0
10	20.0	1.0	0.30	0.43	0.19	0.24	0.17	0.0
10	30.0	1.0	0.30	0.42	0.19	0.24	0.15	0.0
10	40.0	1.0	0.29	0.41	0.18	0.24	0.13	0.0

of the robot ( $T_z$  component of location of camera) does not change much. Therefore the  $T_z$  component of the input location vector is given a very small standard deviation value of 0.001 feet, effectively pinning it to the input estimate. The  $T_x$  and  $T_y$  components of the input location estimate are given standard deviations of 3 feet. This is reasonable because in normal runs of the robot for navigation purposes, the robot can usually be located to within 3 feet just by dead-reckoning. All components of rotation are given standard deviations of 1 radian or  $57.30^\circ$ . Comparing corresponding rows in Table 2.5 and Table 2.6, it can be seen that specifying lower prior covariance matrices greatly improves the results for both algorithms “R\_and\_T\_mod” and “R\_and\_T\_img”.

### Real Data Results

In this subsection results for algorithms “R\_and\_T\_mod” and “R\_and\_T\_img” over two real data sequences are compared. The first sequence consists of 6 outdoor frames. The first, second and fourth frames are shown in Figures 2.4, 2.10 and 2.13 respectively. For the outdoor sequence, the camera was moved in an approximate forward motion 25 feet along the walkway. Each subsequent frame was taken after a movement of 5 feet down the walkway. The sidewalk line is close to parallel to the x-axis in the world coordinate system. The z-axis is the vertical axis in the world coordinate system. The 2D images lines were taken from the output of a 2D line matching system [8]. For each frame, column 2 in Table 2.7 gives the number of lines the 2D line matcher was able to correctly match. The 2D matcher does not return the original image lines, rather it returns the location of the matched 2D lines as predicted by the final 2D-2D pose. One consequence of this is that the input lines used in our experiment for the outdoor sequence are not broken and fragmented like the original extracted image lines may be. Figures 2.10 and 2.13 show the input 2D lines as returned by the line

matcher for frames 4 and 6 of the outdoor image sequence. These were used as input to our algorithm along with the 3D model.

Table 2.7: Estimated Errors of Translation in world coordinates for algorithm “R\_and\_T\_img” and “R\_and\_T\_mod”. Real Data results for Outdoor and Indoor frames without outliers.

Frame No.	Num. Lines	“R_and_T_img”			“R_and_T_mod”		
		TRANSLATION ERROR			TRANSLATION ERROR		
		$\Delta T_x$ feet	$\Delta T_y$ feet	$\Delta T_z$ feet	$\Delta T_x$ feet	$\Delta T_y$ feet	$\Delta T_z$ feet
Outdoor Frames							
1	17	0.47	0.02	0.54	0.42	0.05	0.54
2	15	0.41	0.56	0.59	0.37	0.86	0.85
3	12	2.03	0.48	0.41	2.34	0.58	0.50
4	7	0.64	1.02	0.88	0.77	1.16	0.90
5	13	1.30	0.87	0.61	1.32	0.87	0.61
6	13	1.72	1.23	0.79	1.41	1.43	0.88
Indoor Frames							
1	22	0.24	0.04	0.04	0.24	0.05	0.01
2	22	0.10	0.17	0.02	0.12	0.08	0.01
3	12	1.58	4.87	4.25	0.11	0.11	0.03
4	10	2.95	1.06	0.79	0.10	0.03	0.01

Table 2.7 gives the error in the estimated location by algorithms “R\_and\_T\_img” and “R\_and\_T\_mod”. Ground truth for rotation was not available so the algorithms were compared only with respect to location estimation. In most cases, the robot is located to within one or two feet. The measurement errors in Table 2.7 are approximate to 0.5 feet. The precise location of the camera is not known. It is better to judge the performance of the algorithm by looking at the projections of the 3D landmarks on the image after the pose has been estimated. Figures 2.11 and 2.12 are the projection of the 3D model lines using the poses estimated by the algorithm “R\_and\_T\_img” and “R\_and\_T\_mod” for outdoor frame 2 respectively; similar results

for outdoor frame 4 are shown in Figures 2.14 and 2.15 respectively. As can be seen, in all cases there is fairly good alignment between the 3D model and the original image. The two algorithms perform comparably for this sequence. This is probably due to the fact that the output of the 2D matcher returns whole lines and there is no fragmentation of the 2D image lines.

The results for the outdoors can be improved by the following three strategies :

(1) Use more line correspondences, some of which can be obtained by drawing virtual lines between modeled 3D points, in this case from the corner of the building to the tops of lampposts.

(2) Use closer landmarks. The most accurate result is for frame 1; this is probably because it is the only frame in which there is an object close to the camera (a telephone pole 50 feet away).

(3) Improve the 3D positioning of the lampposts and poles in the 3D model.

The results of the two algorithms "R\_and\_T\_img" and "R\_and\_T\_mod" on four indoor hallway frames are also given in Table 2.7. The camera location for these frames ranged from 32 feet to 23 feet from the door. The results in Table 2.7 are with no outliers in the input data. Figures 2.16 and 2.19 show the input 2D lines for the first and third frame of the indoor sequence respectively. The line correspondences for the four frames were obtained from a motion line tracking system [76]. In this case, the input lines used are the extracted image lines. As a result, some of the input lines are fragmented. If more than one image line was matched to a model line, then the longest matched line was selected for the input set given to the pose algorithms. The projection of the 3D model lines using the poses estimated by the algorithm "R\_and\_T\_img" and "R\_and\_T\_mod" for indoor frame 1 are shown in Figures 2.17 and 2.18 respectively, and for indoor frame 3 in Figures 2.20 and 2.21 respectively.

Algorithm "R\_and\_T\_mod" is able to locate the camera within 0.3 feet for all

four frames. In contrast, algorithm "R\_and\_T\_img" performs poorly for the last two frames. This is due to the severe and noisy fragmentation of the image lines in the last two frames. The final estimate of algorithm "R\_and\_T\_img" for indoor frame 3 is wrong by more than 4 feet (see Table 2.7). In Figure 2.20, it can be seen that the projected model lines (especially, the baseboard lines on the left hand side) for indoor frame 3 are not at all aligned with the input image. The line extraction algorithm has recovered only a small and noisy fragment of the lower left baseboard line (see Figure 2.19). This input line is an outlier for algorithm "R\_and\_T\_img" and causes the final projection to be skewed. Note that if the lower left baseboard line is removed from this data set, the algorithm "R\_and\_T\_img" locates the robot within an inch of the correct location.

From the above experiment for indoor frame 3, two facts are demonstrated. The first, of course, is that algorithm "R\_and\_T\_mod" is more robust than algorithm "R\_and\_T\_img". The second fact is that even a single outlier can cause a least-squares algorithm to fail catastrophically. Thus, we must develop algorithms which are robust with respect to outliers; this is the subject of the next chapter.

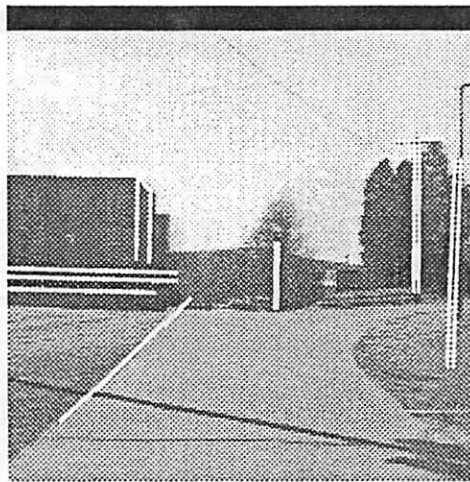


Figure 2.10: Outdoor frame 2 with input image lines.

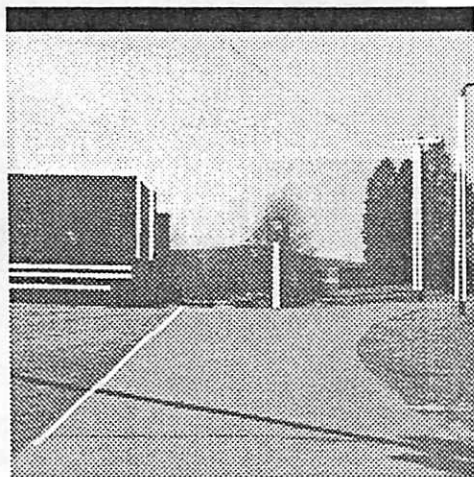


Figure 2.11: Projection of model (Outdoor frame 2) using final estimate of algorithm "R\_and\_T\_img".

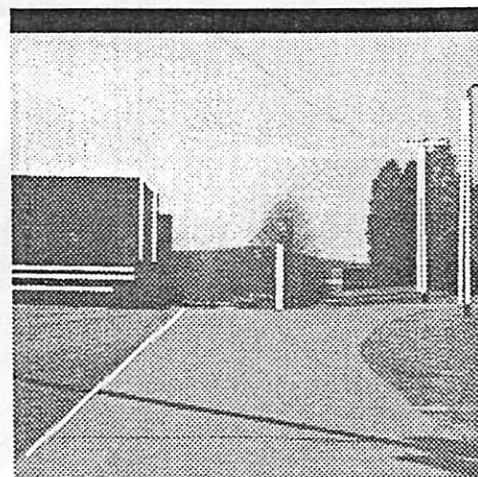


Figure 2.12: Projection of model (Outdoor frame 2) using final estimate of algorithm "R\_and\_T\_mod".

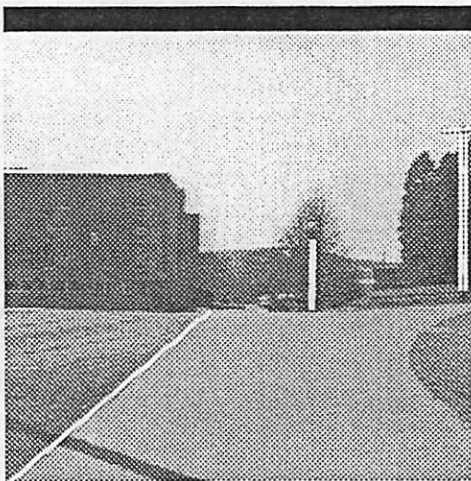


Figure 2.13: Outdoor frame 4 with input image lines.

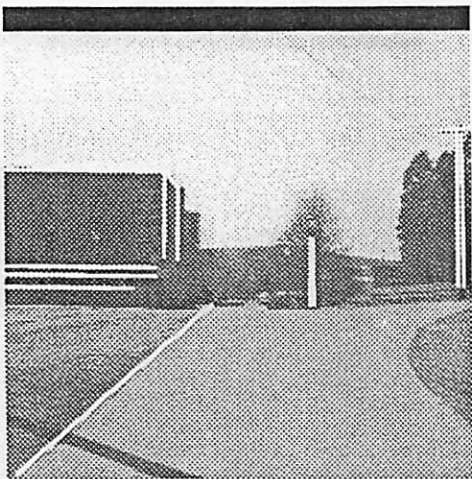


Figure 2.14: Projection of model (Outdoor frame 4) using final estimate of algorithm "R\_and\_T\_img".

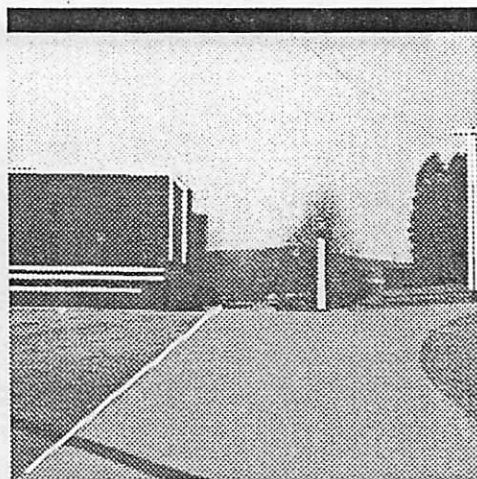


Figure 2.15: Projection of model (Outdoor frame 4) using final estimate of algorithm "R\_and\_T\_mod".

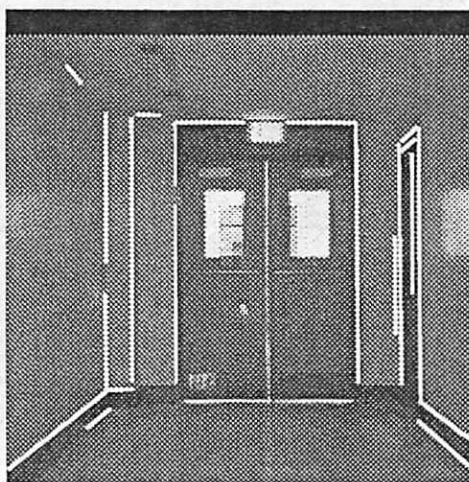


Figure 2.16: Indoor frame 1 with input image lines.

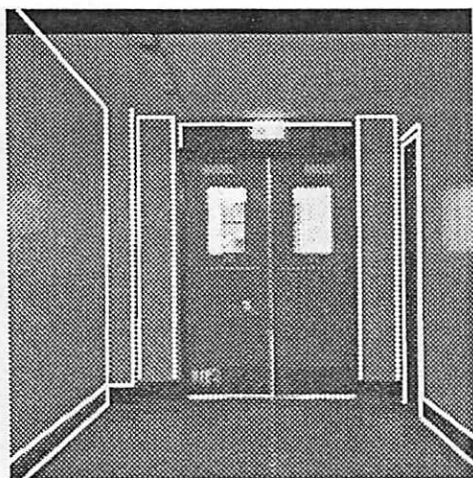


Figure 2.17: Projection of model for Indoor frame 1 using final estimate of algorithm "R\_and\_T\_img".

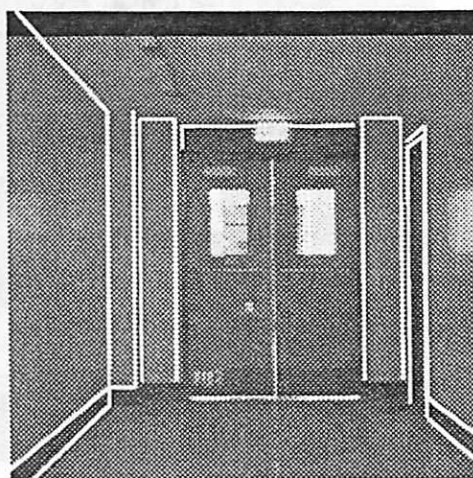


Figure 2.18: Projection of model for Indoor frame 1 using final estimate of algorithm "R\_and\_T\_mod".



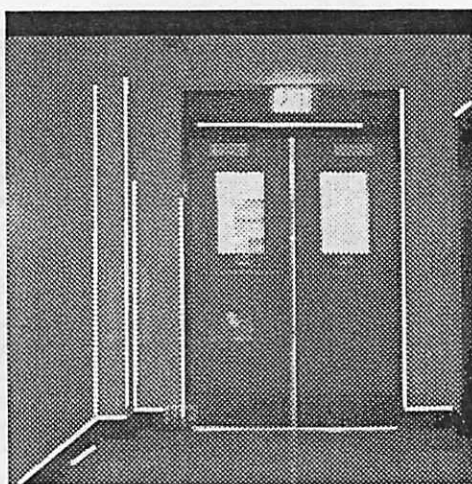


Figure 2.19: Indoor frame 3 with input image lines.

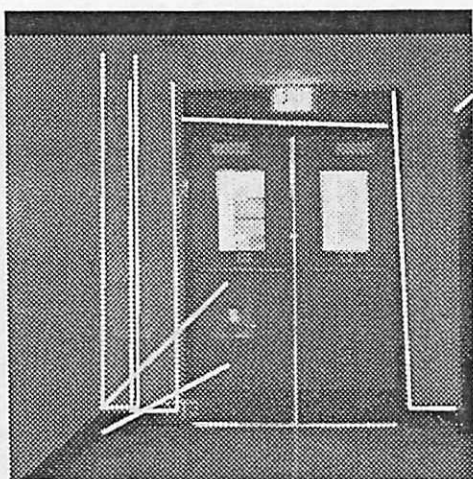


Figure 2.20: Projection of model for Indoor frame 3 using final estimate of algorithm "R\_and\_T\_img".

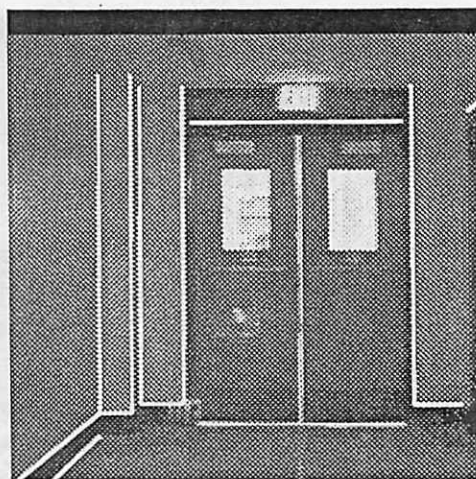


Figure 2.21: Projection of model for Indoor frame 3 using final estimate of algorithm "R\_and\_T\_mod".

## CHAPTER 3

# ROBUST METHODS FOR POSE DETERMINATION

### 3.1 Introduction

This chapter develops and analyzes pose determination techniques which are robust with respect to outliers or gross errors in the data. Given correspondences between 3D lines (or points) represented in a world coordinate system and 2D image lines (or points) represented in a camera coordinate system, the goal is to find the rotation and translation matrices (the pose) which map the world coordinate system to the camera coordinate system. Some of the image to world correspondences may be incorrect and hence outliers or gross errors may be expected to occur. Gross errors or outliers may also occur if parts of the 3D model are incorrect.

Traditionally, least squares techniques have been used for regression analysis or model fitting. In Chapter 2, least squares techniques were presented to solve the pose problem. Least squares is optimum and reliable when the underlying noise in the data is gaussian. However, when outliers are present in the data, the gaussian assumption is violated and the least squares result is skewed in order to make the data approximate a gaussian. Because of the skewing of the result, trying to detect outliers by comparing the residual errors of each line with a threshold will not work. Throwing away one line at a time and doing least squares on the remaining subset also does not work when more than one outlier is present.

Statisticians have suggested many different “robust” techniques [24, 28, 38, 51, 60, 65, 66, 81] to handle outliers and these techniques are currently gaining popularity

in computer vision [7, 23, 24, 31, 39, 45, 46]. A measure to analyze these “robust” algorithms is the breakdown point : the smallest fraction of outliers present in the input data which may cause the output estimate to be arbitrarily wrong. Algorithms based on minimizing L1, L2 or L<sub>n</sub> error measures have breakdown points of 1/n where “n” is the number of data items. Another measure of robust statistical procedures is their “relative efficiency” [39, 65]. It is defined in Kim et. al. [39] as the “ratio between the lowest achievable variance for the estimated parameters (the Cramer-Rao bound) and the actual variance provided by the given method”, so that the best possible value is 1. Kim et. al. also note that “the least mean square estimator in the presence of gaussian noise has an asymptotic (large sample) efficiency of 1 while the median’s efficiency is only 0.637” [39, 60]. As we shall see, there is a trade-off between algorithms with high breakdown points versus those with high efficiency. Finally most research in robust statistics appears to have been done for linear problems. When applying these techniques to non-linear problems, another important consideration is the initial estimate. Non-linear problems are often solved iteratively using an initial estimate. How close must this initial estimate be for the robust technique to work?

In this chapter, two different techniques for robust determination of pose are presented. These techniques are analyzed with respect to the above measures and extensive experimental results on real data are presented.

### 3.2 Previous Work

There are two major sets of statistical techniques for handling outliers. The first attempts to detect outliers before forming a robust estimate. The goal is to find “leverage points” i.e. data points which are on the outskirts of the data cluster. These points, if wrong, can have the largest influence on the final estimate. Standard

outlier detection techniques are based on the diagonal entries of the “Hat” matrix<sup>1</sup>, Mahalanobis distance etc. [24]; these generally work for only certain kind of outliers and often cannot handle more than one outlier. This fact will be demonstrated experimentally in Section 3.5, where the hat matrix values for a contaminated data set with outliers will be given. However, when no outliers are present, the hat matrix is useful for analyzing the data set to determine high leverage data elements and the effect of each of the input data elements on the final estimate. Rousseeuw and Leroy present additional outlier detection techniques in Chapter 6 of their book [65].

The second set of robust statistical techniques detects outliers and computes robust estimates simultaneously. Most of these techniques attempt to define objective functions whose global minimum would not be significantly affected by outliers. The techniques analyzed in this chapter are of this kind. Standard among these are M-estimates (Maximum likelihood type estimates), L-estimates (linear combination of order statistics) and R-estimates (estimates based on rank transformations). M-estimation techniques, first proposed by Huber [38], minimize some function of the individual residual errors. Least square techniques, for instance, minimize the sum of squares of the residual error. The error value for outlying data can be arbitrarily large and hence to accommodate it the least squares fit is skewed.

Functions minimized by M-Estimate techniques attempt to bound the maximum possible error value (e.g. the biweight “redescending” function suggested by Tukey) or bound its rate of change (e.g. Huber’s Minimax function). Figures 3.1 and 3.2 shows Huber’s and Tukey’s function respectively. For small values of “u”, both functions are quadratic up to a point, then Tukey’s function becomes constant (corresponding to the maximum possible error value) whereas Huber’s function becomes linear (corresponding to the maximum rate of change.).

---

<sup>1</sup>The hat matrix is defined in Appendix C.

Inherent in this attempt to bound the error values is a computation of “scale” or the point in these functions where the switch from quadratic to constant or linear occurs. The underlying noise model is based on the assumption that the data is locally contaminated with gaussian noise and then some data elements have gross errors. The standard deviation of the gaussian noise corresponds to the scale. Data points lying beyond a few standard deviations from the mean are classified as outliers and their error is bounded. In some applications, the standard deviation of the gaussian noise may be known whereas in others it is concurrently computed along with the robust estimate. Most of these M-Estimate techniques have been shown to have breakdown points of  $1/(p+1)$  or lower [51, 65] where “p” is the number of unknowns ( $p = 6$  for the pose refinement problem). They have high efficiencies however, typically more than 0.9. Huber [38] suggests iterative algorithms to minimize these error functions and these algorithms are relatively fast to compute. However, it is important to have good initial estimates because of the number of local minima.

Haralick and Joo [31] adapt these M-Estimate techniques for the pose problem using point data. They use the “redescending” function suggested by Tukey [60] and the “minimax” function suggested by Huber [38], respectively. These techniques are also adapted here for the pose problem using line correspondences. Better results were obtained using Tukey’s function as compared to Huber’s. An algorithm “Tuk\_wts” based on using Tukey’s error function and the “Infinite Image Line” constraint for pose is presented in Section 3.3 of this chapter.

Another robust technique suggested by Rousseeuw and Leroy [65] is based on the minimization of the median of the squares of the residual errors (LMS: Least Median Square). This method has a breakdown point of 0.5 and consequently is able to handle data sets which contain less than 50 % outliers. However, since the median is not a differentiable function, it has to be minimized by a combinatorial method and

is comparatively very slow compared to the M-Estimate techniques. To minimize the median square error, a brute-force technique is employed that computes the median square error using all “p” size data subsets, where “p” is the number of unknown parameters. Rousseeuw and Leroy [65] believe that only techniques of this brute force nature will be able to achieve this high breakdown point. They suggest that by sacrificing 100% probability of correctness, a large gain in computational speed can be obtained by considering only a small random set of the minimal subsets (this point is discussed further in Section 3.4). Using the best median pose, data points whose residual error is greater than a certain threshold are weeded out as potential outliers. The threshold may be either fixed a-priori or determined based on the computed scale (standard deviation) of the non-outlier gaussian noise. Finally a reduced weighted least squares (RLS) or a one step M-Estimate <sup>2</sup> is done, using the median estimate as the initial guess. This greatly improves the relative efficiency of the median-based estimate.

In this chapter, two algorithms based on the Least Median Squares technique are developed. The first (“Med\_R\_and\_T\_img”) minimizes the median of the square of the alignment error given by the “Infinite Image Line” constraint discussed in Chapter 2 (see equation 2.11). The second (“Med\_R\_and\_T\_mod”) minimizes the median of the square of the alignment error given by the “Infinite Model Line” constraint (see equation 2.26). In both cases the random sampling techniques suggested by Rousseeuw and Leroy [65] to achieve higher computational speed are implemented and finally a weighted reduced least squares is done to improve the efficiency.

Yohai [81] attempts to combine the high efficiency of M-Estimate methods with the high breakdown of median-based methods by suggesting a fully iterated M-Estimate (as opposed to a one-step M-Estimate) method using a redescending type function

---

<sup>2</sup>A one step M-Estimate is the estimate obtained after one step of the iterative M-Estimation based algorithm.

for which the initial estimate is obtained by minimizing the least squares median. We find, empirically, that local minima of the Median based methods are also the local minima of the M-Estimate methods and therefore there is no benefit in following this procedure.

Fischler and Bolles [23], with their RANSAC paradigm, were among the first to present robust techniques for the pose problem given point data. They find a pose, using a minimal set of three points, and then attempt to grow the solution by successively adding points which are satisfied by the same pose. If a sufficient number of points can be explained by a pose, then it is chosen as the final estimate. Their technique is similar to the median-based technique in that it finds the best pose by looking for a consensus. However, it is adhoc in terms of what is being optimized and the search mechanism to find the estimate. The median based techniques provide an explicit error function to minimize and also provide a mechanism for calculating the scale of the gaussian noise to detect and remove outliers.

In the next section, the M-Estimate based “Tuk\_wts” algorithm is presented. The median based algorithms “Med\_R\_and\_T\_img” and “Med\_R\_and\_T\_mod” are described in section 3.4. Results and discussion are presented in section 3.5.

### 3.3 M-Estimation Techniques

A least squares optimization technique minimizes the sum of squares of an error function “ $e$ ” of an unknown parameter vector “ $\theta$ ” over all “ $n$ ” data elements:

$$\text{Minimize } \sum_i^n e_i^2 \quad (3.1)$$

Least square techniques are optimum and reliable when the underlying noise in the data is gaussian. However, when outliers are present in the data, the gaussian assumption is violated. The error values for outlying data can be arbitrarily large and

hence to accommodate even a single outlier the least squares result can be arbitrarily skewed. Least squares fitting therefore has a break down point of  $\frac{1}{n}$ .

In contrast, M-Estimation techniques, developed by Huber [38] and other statisticians, minimize the sum of a function  $\rho(e_i/s)$  where  $e_i$  is the error function for the  $i$ 'th data vector and  $s$  is a scaling factor:

$$\text{Minimize } \sum_i^n \rho(e_i/s) \quad (3.2)$$

The function  $\rho$  is designed to bound the influence of outliers and it must satisfy the following assumptions for efficient optimization [38, 81]:

1.  $\rho(0) = 0$
2.  $\rho(u) = \rho(-u)$
3.  $0 \leq u \leq v$  implies  $\rho(u) \leq \rho(v)$
4.  $\rho$  is continuous
5. let  $b = \sup \rho(u)$  then  $0 < b < \infty$
6. if  $\rho(u) < b$  and  $0 \leq u < v$  then  $\rho(u) < \rho(v)$

Essentially these conditions guarantee that  $\rho(u)$  is a continuous, symmetric function with minimum value at  $u = 0$ . Also  $\rho(u)$  must be monotonically increasing from  $u = 0$  to  $\infty$  and from  $u = 0$  to  $-\infty$ .

There have been many  $\rho$  functions proposed in the literature. One of the first was proposed by Huber [38]:

$$\rho(u) = \begin{cases} 0.5u^2 & \text{if } |u| \leq a \\ a|u| - 0.5a^2 & \text{otherwise} \end{cases} \quad (3.3)$$

Figure 3.1 shows Huber's function and its derivative plotted with "a" set to 2.0. In this function, for small values of "u",  $\rho(u)$  varies as the square of "u". Above a certain



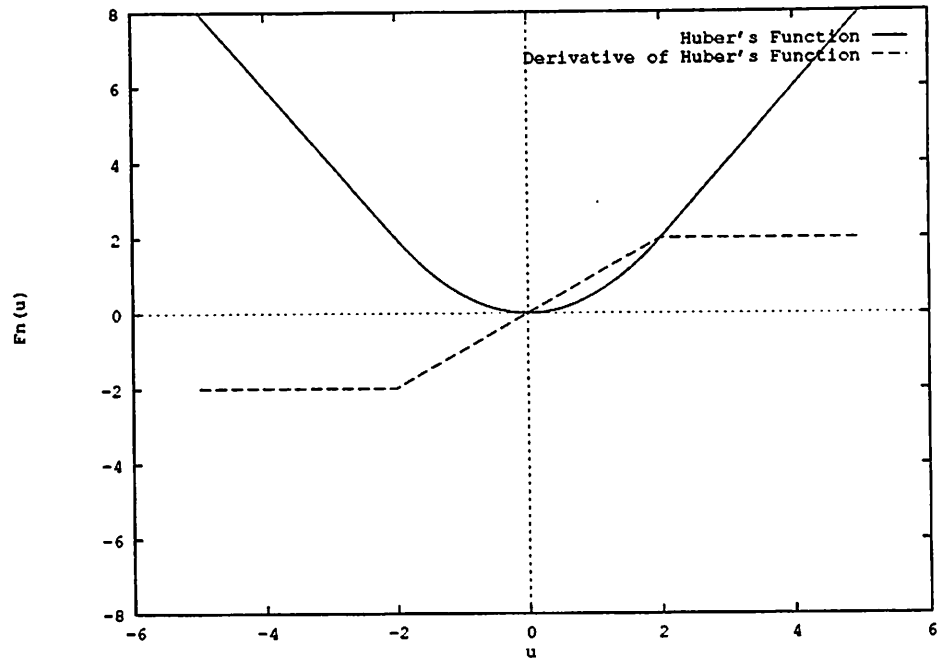


Figure 3.1: Huber's minimax function and its derivative.

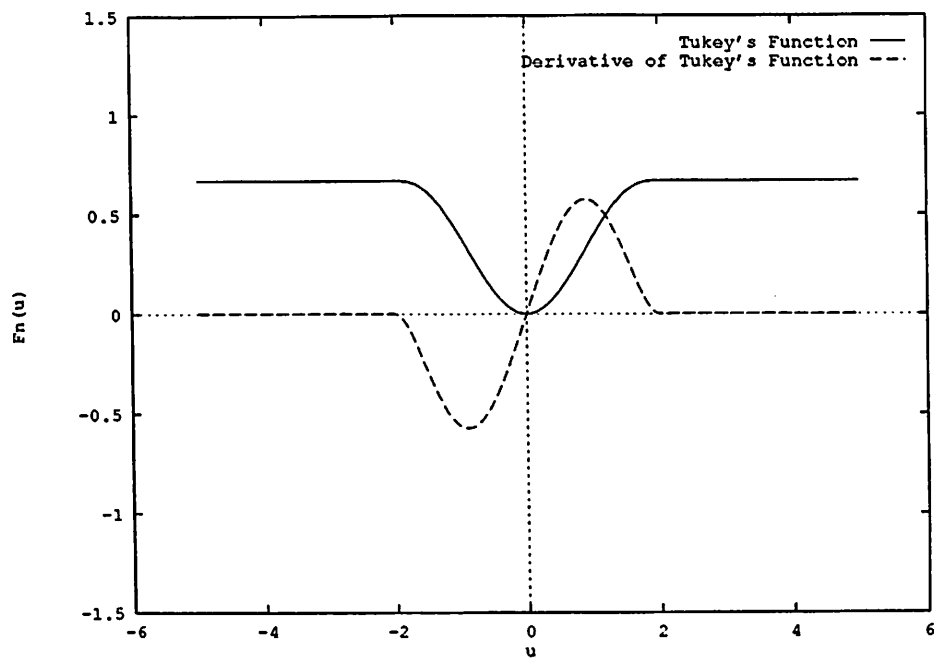


Figure 3.2: Tukey's biweight redescending function and its derivative.

threshold value “ $a$ ”,  $\rho(u)$  varies linearly; thus the rate of change of the function is bounded. The function is therefore a combination of L2 and L1 error norms. However, it still has a break-down point of  $\frac{1}{n}$ , since both the L1 and L2 norms have break-down points of  $\frac{1}{n}$ .

Another  $\rho$  function was proposed by Tukey [60]:

$$\rho(u) = \begin{cases} u^2/2 - u^4/2a^2 + u^6/6a^4 & \text{if } |u| \leq a \\ a^2/6 & \text{otherwise} \end{cases} \quad (3.4)$$

Figure 3.2 shows Tukey’s function and its derivative plotted with “ $a$ ” set to 2.0. In this function,  $\rho(u)$  approximately varies as the square of “ $u$ ” for small values and then tapers off to a constant maximum value of “ $a^2/6$ ”. Thus the error for any outlier data element cannot be arbitrarily large. A rationale for this function is that for small error values, “ $u$ ” corresponds to gaussian noise and thus for optimum relative efficiency their square value must be minimized. However, as the error grows, the data element is probably an outlier and therefore its influence must be bounded. Blake and Zisserman [10] use a variation of Tukey’s function in solving the problem of fitting piece-wise smooth surfaces to range data. They provide another rationale for the above function: the constant maximum value is assumed to be the cost of assigning a data element to be an outlier. Thus non-outlier data elements are minimized as the sum of squares of their error values and there is a cost of assigning a data element to be an outlier. The final estimation is a trade-off between minimizing the number of outliers and the fitting error for the non-outlier data points.

Inherent in these M-Estimation techniques is a simultaneous computation of a scale  $s$ . The scale corresponds to the standard deviation of the residual errors. If a good estimate of the standard deviation of the errors of non-outlier data can be made, then data points whose error lies beyond a certain number of standard deviations from the center (that is, in the “tails” of the distribution) can be classified as outliers. The estimate of scale therefore itself must be robust and not affected by outliers. M-

Estimation techniques therefore vary both on their choice of the  $\rho$  function and the method used to compute scale. In the algorithm presented below, scale  $s$  is computed by the following equation:

$$s = \frac{\text{median}_i |e_i|}{0.6745} \quad (3.5)$$

where 0.6745 is one half of the interquantile range of the Gaussian Normal distribution  $N(0,1)$ . This equation relates the scale (standard deviation) of a gaussian distribution to the median of the absolute values of a sampled set of data points. It is based on the fact that the median of the absolute values of random numbers sampled from the Gaussian Normal distribution  $N(0,1)$  is estimated to be approximately 0.6745.

### 3.3.1 The “Tuk\_wts” Algorithm

Huber [38] suggests two methods for minimizing the error functions in equations (3.2), (3.3) and (3.4). The two methods are the modified residual method and the modified weights method [38, 31]. In our experiments better performance was obtained using the modified weights method; consequently only this version will be presented here.

The modified weights method is an iterative reweighting least squares (IRLS) algorithm. Let  $\theta_j$  represent the set of unknown parameters (rotation and translation in our case) to be estimated. Differentiating the error expression in equation (3.2) with respect to each parameter  $\theta_j$  we get the set of equations:

$$\sum_i \Psi\left(\frac{e_i}{s}\right) \frac{\delta e_i}{\delta \theta_j} = 0.0 \quad (3.6)$$

where  $\Psi$  is the derivative of the  $\rho$  function with respect to  $u$ . The  $\Psi$  function used in the “Tuk\_wts” algorithm is obtained by differentiating Tukey’s  $\rho$  function as given in equation (3.4):

$$\Psi(u) = \begin{cases} u(1 - \frac{u^2}{a^2})^2 & \text{if } |u| \leq a \\ 0.0 & \text{otherwise} \end{cases} \quad (3.7)$$

The  $\Psi$  function is also sketched in Figure 3.2. Note, that for small values of “ $u$ ” around  $u = 0.0$  it is almost linear and then it slowly tapers off to zero on both ends.

Equation (3.6) can be written in the standard weighted form as:

$$w_i = \frac{\Psi\left(\frac{e_i}{s}\right)}{\frac{e_i}{s}} \quad (3.8)$$

$$\sum_i w_i e_i \frac{\delta e_i}{\delta \theta_j} = 0.0 \quad (3.9)$$

If the weights  $w_i$  in the above equation (3.9) are held constant and the residual error  $e_i$  is a linear function of the unknown set of parameters, then a linear system of equations in the unknown parameters  $\theta_j$  is obtained. The modified weights algorithm solves the equations (3.9) iteratively. At each iteration the weights  $w_i$  are held constant and equal to the values computed using the previous iteration’s estimates of the unknown parameters  $\theta_j$  and the linear system of equations is solved to get new estimates. The iterative procedure is repeated until the parameter values converge to a final value. Huber [38] proves that the above iterative procedure leads to a minimum (possibly a local minimum) of the objective function as given in equation (3.2) for linear  $e_i$  residual error functions<sup>3</sup>.

In this iterative procedure, if the error of a data element is greater than “ $a$ ” for a current estimate, then by the weighting function defined in equation (3.8) its weight will be zero and therefore it will not contribute to the new estimation. Data elements with small errors will have almost unit weight while data elements with slightly larger errors (but less than “ $a$ ”) will contribute partially (between 0 and 1) to the final fit. This partial contribution of these data elements greatly improves the relative efficiency of the final estimation. This can be intuitively understood by recalling that the noise model assumes that most of the data is contaminated by gaussian noise and only some of the data elements are outliers. The important issue

---

<sup>3</sup>Note, although  $e_i$  may be linear,  $\rho(e_i)$  is not a quadratic function.

is the criteria for deciding when a data element should be classified as an outlier (i.e., the number of standard deviations to be used as an outlier threshold). The relative efficiency is improved by using all data elements which are not outliers for the final fit. The above procedure does not automatically cut-off all data elements beyond a certain threshold; instead it gives increasingly small weight to data elements with larger errors until a final weight of zero is given for data elements whose error is larger than “ $a$ ” standard deviations.

We now turn to the problem of applying this technique to the minimization of the pose error functions, which are non-linear. In Chapter 2, an iterative method was developed to minimize the sum of squares of the error function. This method entailed linearizing the error function about the current estimate and then solving for the small increments in translation ( $\Delta T$ ) and rotation ( $\delta\omega$ ). Equations 2.38 and 2.39 shown in chapter 2.4 are the linear system of equations solved at each iteration of the “R\_and\_T” algorithm described there. An equivalent set of equations can be developed for algorithms “R\_and\_T\_img” and “R\_and\_T\_mod”. The error for each data element is weighted by  $w_i$ . To incorporate the robust techniques described above into the minimization procedure for “R\_and\_T” (and by extension for “R\_and\_T\_img” and “R\_and\_T\_mod”) described in Section 2.4, the only change required is to replace the weights in equations 2.38 and 2.39 by the weighting function given in equation (3.8). This procedure leads to rapid convergence and a robust estimate, as will be seen in the experiments section 3.4. For most of our experiments, convergence is reached in about 10 iterations.

The “Tuk\_wts” algorithms developed here solves the system of equations (2.38) and (2.39) corresponding to algorithm “R\_and\_T\_img” at each iteration and composes the results with the previous estimates of rotation and translation. The weights are calculated using equation (3.8). The steps used in the algorithm are identical to the

steps used in algorithm “R\_and\_T\_img”. At each iteration there is also an estimation of scale using equation (3.5). Finally, although not done here, the “R\_and\_T\_mod” algorithm can also be similarly modified to give a robust M-Estimate version of it.

### 3.4 Least Median of Squares (LMS) Technique

In this section another set of robust techniques for the pose determination problem are developed. These techniques were developed by Rousseeuw et. al. [65] for the linear regression problem. They are based on minimizing the median of the square of the error function over all data elements:

$$\text{Minimize } \underline{\text{Median}}_i e_i^2 \quad (3.10)$$

Earlier, it was noted that in least square systems the large error values of the outlier data elements causes a skew of the final fit. If the data elements are ranked in ascending order according to error values, the median corresponds to the error of the middle data element. Minimizing the median, therefore ignores the errors of the larger ranked half of the data elements. Thus, this method automatically can perform robustly in situations where less than 50 % of the data elements are outliers; it has a breakdown point of  $\frac{1}{2}$ .

This procedure can be modified to incorporate a-priori knowledge of the level of contamination of the data set with outliers. If for example, in a particular application, it is guaranteed that no more than 30 % of the points are outliers, then the 70 % ranked error could be minimized rather than the median. In absence of any such guarantees, the median is the best choice to minimize. This corresponds to finding that pose which fits (with minimum error) at least half of the data elements. In other words, the final estimated pose is the best consensus fit over all subsets of size equal to half the number of the data elements.

To develop robust pose algorithms, any of the pose alignment error functions developed in Chapter 2.4 can be substituted for  $e_i$  in equation (3.10). In this section, two such robust algorithms “Med\_R\_and\_T\_img” and “Med\_R\_and\_T\_mod” are developed, based on the alignment errors used in the least square algorithms “R\_and\_T\_img” and “R\_and\_T\_mod”, respectively. In “Med\_R\_and\_T\_img” the following objective function “ $E_{m1}$ ” is minimized:

$$E_{m1} = \text{Median}_i \left( \left( \frac{\vec{N}_i \cdot (R(\vec{p}_{i1}) + \vec{T})}{(R(\vec{p}_{i1}) + T)_z} \right)^2 + \left( \frac{\vec{N}_i \cdot (R(\vec{p}_{i2}) + \vec{T})}{(R(\vec{p}_{i2}) + T)_z} \right)^2 \right) \quad (3.11)$$

$E_{m1}$  is a modification of the objective function  $E_2$  given in equation (2.32) in Section 2.4. The right hand side corresponds to the median of the total alignment error for each line. The total alignment error is based on the “Infinite Image Line” constraint and is the sum of squares of the perpendicular distances from the projected model end-points to the infinitely extended image line.

The “Med\_R\_and\_T\_mod” algorithm is based on the “Infinite Model Line” constraint and the following objective function “ $E_{m2}$ ” is minimized:

$$\vec{C}_i = (\vec{p}_{2i} - \vec{T}_w) \times (\vec{p}_{1i} - \vec{T}_w) \quad (3.12)$$

$$\vec{B}_j = \left( \frac{I_{xj}}{s_x}, \frac{I_{yj}}{s_y}, 1 \right)^T \quad (3.13)$$

$$E_{m2} = \text{Median}_i \frac{1.0}{M_i^2} \left( (R^T(\vec{B}_{i1}) \cdot \vec{C}_i)^2 + (R^T(\vec{B}_{i2}) \cdot \vec{C}_i)^2 \right) \quad (3.14)$$

$E_{m2}$  is a modification of the objective function  $E_3$  given in equation (2.33) of Section 2.4;  $M_i$  is defined in Section 2.4. Again, the right hand side corresponds to the median of the total alignment error for each line. In this case, however, the total alignment error is based on the “Infinite Model Line” constraint and is the sum of squares of the perpendicular distances from the image line end-points to the infinitely extended projected model line.

Since the median is not a differentiable function,  $E_{m1}$  and  $E_{m2}$  must be minimized by combinatorial methods. In the method adopted here, candidate poses are

generated by using the least squares algorithms developed in chapter 2 on subsets of the data elements. The pose which gives the minimum median error across all data elements is chosen as the optimal median pose. The goal is to find at least one subset which has no outliers in it; this should give the minimum median error. The “pose determination problem” needs a minimum of 3 input lines. Thus subsets of line data elements used to generate the candidate poses must be of size  $m$  ( $m \geq 3$ ). Typically, poses are generated from all subsets of size  $m$  of the data elements. Experimentally, we have found the choice of “ $m$ ” is important. The larger the size of the subsets, the greater the probability of them having an outlier. However, choosing “ $m$ ” = 3, the minimum as suggested by Rousseeuw [65], often leads to local minima. Typically good results are obtained with  $m = 6$  or higher. This is because the poses estimated by using just 3 line data sets have large variances and some of the non-outlier lines can get labeled as outliers. This point is further elaborated in the results section of this chapter (Section 3.4).

To speed up the computation, instead of using all subsets, only a random set of all size  $m$  subsets is used. If  $\epsilon$  is the fraction of contaminated data and we choose “ $k$ ” different random subsets of size “ $m$ ”, then the probability “ $P$ ” that all “ $k$ ” different subsets will contain at least one or more outliers is:

$$P = (1 - (1 - \epsilon)^m)^k \quad (3.15)$$

The probability that at least one random subset has no outliers is given by  $(1 - P)$ . This is the probability that the correct answer will be found by the median algorithm. For example, if we want the correct answer with 99% probability and expect no more than 30% outliers when using subsets of size 3 ( $m = 3$ ), then only 37 out of the 1140 subsets (for a set of 20 lines) need to be randomly chosen. In practice, however, we find that a much larger set needs to be chosen (again, this is shown in the results section). Finally, some subsets will lead to degenerate solutions because all lines are



parallel, etc. This can be detected before any further processing of the subset is done. A simple method to detect degenerate subsets in the case of three lines is to threshold on the determinant of the matrix whose rows are the unit direction vectors of the 3D lines.

Using the best median pose, data points whose residual error is greater than a certain threshold are weeded out as potential outliers. The threshold may be either fixed a-priori or determined based on the computed scale (standard deviation) of the non-outlier gaussian noise. In many of our experiments, it is assumed that the scale of the line fitting process to edge data is 2 pixels. Equation (3.5) developed in the previous section may also be used to compute the scale. Finally, the weighted least squares algorithm (“R\_and\_T\_img”) or (“R\_and\_T\_mod”) are run on the remaining lines. This last step greatly improves the “relative efficiency” of the robust algorithm.

The algorithms “Med\_R\_and\_T\_img” and “Med\_R\_and\_T\_mod” are summarized as follows:

**Step 1:** Select “k” random subsets of size “m” from the input data.

**Step 2:** For each subset, determine the pose by using algorithm “R\_and\_T\_img” or “R\_and\_T\_mod”. Estimate the residual error for all “n” lines given this pose and find the median square error.

**Step 3:** Select the pose which gives the minimum median error  $E_{m1}$  or  $E_{m2}$  and compute the scale “s” (if not known a-priori) using equation (3.5).

**Step 4:** Filter out lines as outliers whose squared residual error for that pose is greater than  $(a \times s)^2$ ;  $a$  is an algorithm parameter and is set equal to 2.0 for all experiments discussed in Section 3.5.

**Step 5:** Minimize the error function given in equation (2.32) or equation (2.33) on the remaining lines using the least square algorithm “R\_and\_T\_img” or algorithm

“R\_and\_T\_mod” and return the estimated translation and rotation as the final output.

### 3.5 Results and Discussion

In this section we shall present and analyze the results of running the three algorithms “Tuk\_Wts”, “Med\_R\_and\_T\_img” and “Med\_R\_and\_T\_mod” on real image data. The results are presented for both the indoor hallway images (Figures 3.3 and 3.11) and the outdoor sidewalk images (Figures 3.7 and 3.15). The indoor and outdoor images are similar to the ones used in Chapter 2 for the least-squares experiments. In some cases, the data used for experiments in chapter 2 was artificially altered to create outliers. In other cases, the data is directly an output of the the 2D matching system developed by Beveridge et. al. [8]. The camera parameters and the indoor and outdoor models used for the experiments are exactly the same as those described in Chapter 2. In the indoor scenes the door is about 20 to 40 feet from the camera. In the outdoor scenes, the building is about 300 feet from the camera.

Tables 3.1–3.4 shows the results of the three algorithms on eight indoor hallway images and three outdoor walkway images. In the tables only the translation results (in world coordinates) are listed. This is because we have no way to measure accurately the true orientation of the camera with respect to the world coordinate system. However the 3D model can be projected into the image plane using the final computed pose. A qualitative estimation of the orientation accuracy can be obtained by seeing how well the projected model aligns with the original image data (e.g. as in Figures 3.5 and 3.9) In the tables, the translation direction “x” is the horizontal direction oriented with the long side of the hallway (for indoor images) or the walkway (for outdoor images), the “y” direction is the horizontal direction perpendicular

to the long sides of the hallway or walkway, and the “z” direction is aligned with the direction of gravity. The data listed under the column “No. Ln.” is the number of input data lines and the data under the column “Outliers Fnd.” is the number of outliers found by the median based algorithm.

The “Med\_R\_and\_T\_img” and “Med\_R\_and\_T\_mod” algorithms were run on both the indoor and outdoor data sets with the same set of parameters. In each case, 500 random sample sets of size 6 were generated. The scale of the gaussian noise was assumed to be 2 pixels for all lines. Using the best median pose, any line whose alignment error was more than 4 pixels was declared to be an outlier and it was given zero weight for the final least-squares fit. Similar to Chapter 2, each experiment was performed with two different settings of the input prior covariance matrix of the pose parameters. In each case, the prior covariance matrix was a diagonal matrix. In Tables 3.1 and 3.2, the prior covariance matrix specified for the least-squares and median based algorithms had high diagonal values corresponding to standard deviations of 94.86 feet for translation terms and 1 radian or 57.3 degrees for the rotation terms. In the experiments whose results are presented in Tables 3.3 and 3.4, the standard deviations of the rotation terms in the prior covariance matrix was set at the same value of 57.3 degrees. However much smaller standard deviations are specified for the translation terms; the input estimate of the height of the robot (the translation term  $T_z$ ) was set to have a standard deviation of 0.001 feet and the estimates of the horizontal location of the robot (the translation terms  $T_y$  and  $T_x$ ) were set at standard deviations of 3 feet each. This corresponds to the typical values obtainable from dead-reckoning in the robot-navigation domain. It was ensured for all experiments that the initial estimates used were within a standard deviation of the correct estimate. Finally, the “Tuk\_wts” algorithm was always run with the threshold “a” in equation (3.7) set to 4.0.

For comparison, the results of the least-squares algorithms “R\_and\_T\_img” and “R\_and\_T\_mod” on the same data sets are also presented in the tables. The performance of the least-squares algorithms gives a measure of the severity of the outliers in each of the data sets. Generally, the more gross an outlier is with respect to the non-corrupt data the easier it is detect and remove<sup>4</sup>. However, the robust algorithms must perform comparably or better than the least-squares algorithms both when the least squares algorithms completely fail and when they do reasonably well. The results presented in the Tables 3.1–3.4 document both these cases. The first case is the first 7 frames of the indoor hallway images. From any of the four tables, it can be observed that the least-squares algorithms performed reasonably well for the first seven indoor frames. The outliers in these cases have not very significantly affected the performance of the least-squares algorithms and the robust algorithms perform only slightly better. In contrast to the first seven indoor frames, the least-squares algorithms totally fail in locating the robot for indoor frame 8 and outdoor frames 1 and 3. Each of these cases is discussed in more detail in the next few sections.

### 3.5.1 The First Seven Indoor Frames

The average error in locating the robot for the first seven indoor frames by the least-squares algorithms “R\_and\_T\_img” and “R\_and\_T\_mod” when run with high initial covariance matrices is 0.676 feet and 0.585 feet respectively<sup>5</sup>. Note, that “R\_and\_T\_mod” performs slightly better than “R\_and\_T\_img”. The robust algorithms perform better than both the least squares algorithms. From the data shown in the Tables 3.1 and 3.2, the average error in locating the robot for the seven indoor frames by algorithms “Med\_R\_and\_T\_img”, “Tuk\_wts” and “Med\_R\_and\_T\_mod” (run with

---

<sup>4</sup>This is actually more true for the median algorithms. Severe outliers can affect the “Tuk.Wts” algorithm because of its low break down point.

<sup>5</sup>Note, the average values are calculated from data in Tables 3.1 and 3.2

high initial covariance matrices) is 0.405 feet, 0.387 feet and 0.363 feet respectively. Note again, “Med\_R\_and\_T\_mod” performs slightly better than “Med\_R\_and\_T\_img”.

Similar statistics can be calculated from the data shown in Tables 3.3 and 3.4 for the cases when the algorithms were run with low initial covariance matrices. The least-squares algorithms “R\_and\_T\_img” and “R\_and\_T\_mod” located the robot in these cases to an average error of 0.626 feet and 0.488 feet respectively. The robust algorithms “Med\_R\_and\_T\_img” and “Med\_R\_and\_T\_mod” located the robot for the same seven frames to an average error of 0.366 feet and 0.280 feet respectively. Based on the above comparisons, algorithm “Med\_R\_and\_T\_mod” performed the best. It located the robot within 0.20 feet for five of the seven frames (see Table 3.2 or Table 3.4). Figure 3.3 shows the initial set of lines for Indoor frame 7. Figure 3.4 shows the projection of the model using the pose estimated by the least-squares algorithm “R\_and\_T\_mod” and there is some mis-alignment of the two baseboard lines running parallel to the length of the hallway. The projections of the model using the poses estimated by algorithms “Med\_R\_and\_T\_mod” and “Tuk\_wts” is shown in figures 3.5 and 3.6 respectively. In these two cases, the alignment between model and image data has improved.

### 3.5.2 Outdoor Frame 1

Figure 3.7 shows the input data set of 17 lines for the outdoor frame 1. In this case, the street-light has been mismatched to the telephone pole. The top bar of the telephone pole is also located incorrectly. Finally, the left edge of the nearby telephone pole (in the left of the image) is slightly out of alignment and also the walkway line has been slightly skewed. Thus, of the 17 input lines, 5 are clear outliers and 2 may or may not be outliers. The least-squares algorithms are off in their estimate of location by about 10 feet for this data set (for all cases of initial covariance). Figure 3.8 shows

Table 3.1: Estimated Errors of Translation in world coordinates for algorithms “R\_and\_T\_img”, “MED\_R\_and\_T\_img” and “Tuk\_Wts”; High Covariance Case. Prior Standard Deviation ( $\sigma$ ) of the initial robot pose estimate is 94.86 feet and 1 radian for each axis of location and orientation, respectively.

Fr.	No.	“R_and_T_img”			“Med_R_and_T_img”			Out-Liers Fnd.	“Tuk_Wts”		
		TRANS. ERR.			TRANS. ERR.				TRANS. ERR.		
No.	Ln.	$\Delta T_x$ feet	$\Delta T_y$ feet	$\Delta T_z$ feet	$\Delta T_x$ feet	$\Delta T_y$ feet	$\Delta T_z$ feet		$\Delta T_x$ feet	$\Delta T_y$ feet	$\Delta T_z$ feet
INDOOR FRAMES											
1	24	0.35	0.26	0.00	0.08	0.01	0.01	7	0.11	0.01	0.02
2	26	0.87	0.48	0.08	0.19	0.02	0.02	11	0.47	0.32	0.06
3	24	0.63	0.09	0.02	0.01	0.42	0.35	10	0.13	0.01	0.01
4	15	0.57	0.03	0.23	0.55	0.22	0.10	7	0.64	0.25	0.29
5	16	0.43	0.23	0.03	0.27	0.01	0.07	5	0.27	0.01	0.10
6	16	0.74	0.37	0.47	0.65	0.18	0.23	4	0.59	0.26	0.30
7	46	0.38	0.25	0.41	0.11	0.21	0.35	17	0.06	0.02	0.07
8	19	5.67	0.27	0.56	0.05	0.38	0.32	9	0.04	0.00	0.01
OUTDOOR FRAMES											
1	17	10.42	2.76	2.38	0.95	0.36	1.22	7	1.16	0.36	0.14
3	15	40.91	13.56	10.19	2.39	0.21	0.12	5	36.72	13.41	13.46

Table 3.2: Estimated Errors of Translation in world coordinates for algorithms “R\_and\_T\_mod”, “MED\_R\_and\_T\_mod”; High Covariance Case. Prior Standard Deviation ( $\sigma$ ) of the initial robot pose estimate is 94.86 feet and 1 radian for each axis of location and orientation, respectively.

Fr.	No.	“R_and_T_mod”			“Med_R_and_T_mod”			Out-Liers Fnd.
		TRANS. ERR.			TRANS. ERR.			
No.	Ln.	$\Delta T_x$ feet	$\Delta T_y$ feet	$\Delta T_z$ feet	$\Delta T_x$ feet	$\Delta T_y$ feet	$\Delta T_z$ feet	
INDOOR FRAMES								
1	24	0.26	0.20	0.03	0.02	0.47	0.44	6
2	26	0.75	0.47	0.05	0.20	0.03	0.02	9
3	24	0.14	0.12	0.02	0.04	0.04	0.01	3
4	15	0.50	0.04	0.10	0.02	0.08	0.00	4
5	16	0.34	0.34	0.12	0.15	0.01	0.07	3
6	16	0.77	0.59	0.11	0.85	0.49	0.33	4
7	46	0.31	0.20	0.61	0.14	0.01	0.03	17
8	19	5.47	0.30	0.78	0.05	0.38	0.32	9
OUTDOOR FRAMES								
1	17	10.41	2.82	2.28	1.00	0.38	1.27	7
3	15	41.23	13.90	7.91	2.53	0.19	0.10	5

Table 3.3: Estimated errors of translation in world coordinates for algorithms “R\_and\_T\_img”, “MED\_R\_and\_T\_img”; Low covariance case. Prior standard deviation ( $\sigma$ ) of initial robot pose estimate is 3 feet for “x” and “y” axis of location, 0.001 feet for height of robot and 1 radian for each axis of orientation.

Fr. No.	No. Ln.	“R_and_T_img”			“Med_R_and_T_img”			Out-Liers Fnd.
		TRANS. ERR.			TRANS. ERR.			
		$\Delta T_x$ feet	$\Delta T_y$ feet	$\Delta T_z$ feet	$\Delta T_x$ feet	$\Delta T_y$ feet	$\Delta T_z$ feet	
INDOOR FRAMES								
1	24	0.34	0.26	0.00	0.09	0.02	0.00	7
2	26	0.84	0.46	0.00	0.18	0.03	0.00	11
3	24	0.64	0.08	0.00	0.13	0.02	0.00	7
4	15	0.54	0.21	0.00	0.84	0.42	0.00	5
5	16	0.48	0.20	0.00	0.27	0.05	0.00	5
6	16	0.68	0.13	0.00	0.68	0.52	0.00	5
7	46	0.50	0.26	0.01	0.07	0.05	0.00	17
8	19	5.65	0.25	0.00	0.03	0.00	0.00	8
OUTDOOR FRAMES								
1	17	8.84	1.63	0.00	1.04	0.34	0.00	7
3	15	9.86	1.16	0.00	2.43	0.08	0.00	5

Table 3.4: Estimated errors of translation in world coordinates for algorithms “R\_and\_T\_mod”, “MED\_R\_and\_T\_mod”; Low covariance case. Prior standard deviation ( $\sigma$ ) of initial robot pose estimate is 3 feet for “x” and “y” axis of location, 0.001 feet for height of robot and 1 radian for each axis of orientation.

Fr. No.	No. Ln.	“R_and_T_mod”			“Med_R_and_T_mod”			Out-Liers Fnd.
		TRANS. ERR.			TRANS. ERR.			
		$\Delta T_x$ feet	$\Delta T_y$ feet	$\Delta T_z$ feet	$\Delta T_x$ feet	$\Delta T_y$ feet	$\Delta T_z$ feet	
INDOOR FRAMES								
1	24	0.26	0.20	0.00	0.13	0.04	0.00	5
2	26	0.73	0.45	0.00	0.30	0.00	0.00	10
3	24	0.13	0.12	0.00	0.03	0.03	0.00	3
4	15	0.47	0.04	0.00	0.08	0.05	0.00	3
5	16	0.43	0.28	0.00	0.16	0.05	0.00	3
6	16	0.62	0.15	0.00	0.95	0.57	0.00	4
7	46	0.40	0.17	0.04	0.07	0.09	0.00	17
8	19	5.56	1.53	0.00	0.04	0.01	0.00	7
OUTDOOR FRAMES								
1	17	7.09	1.53	0.01	1.07	0.35	0.00	7
3	15	10.96	1.54	0.00	2.92	0.21	0.00	5

the projection of the model using the pose returned by algorithm “R\_and\_T\_mod” for this data set and as can be observed, there is significant mis-alignment. The median algorithms and “Tuk\_wts” are able to locate the robot to within 1.5 feet for this data set. This result is considered quite reasonable given that the 3D landmark lines are all in the range of 300 feet distant from the camera except for the two model lines of the nearby telephone pole (50 feet distant) and the walkway line. The final projection of the model for algorithms “Med\_R\_and\_T\_mod” and “Tuk\_wts” are shown in Figures 3.9 and 3.10 respectively. All 7 lines (discussed above) are detected as outliers. However, as can be noticed from the figures, the final projections of all model lines are fairly well aligned.

### 3.5.3 Indoor Frame 8

Figure 3.11 shows the input set of lines for indoor frame 8. This data set of 19 lines has 8 outliers, i.e. approximately 40% of the data is contaminated. Only the 11 lines on or close to the left wall are correct. Both the least-squares algorithms “R\_and\_T\_img” and “R\_and\_T\_mod” fail (for all variance settings) on this data set. Their final pose estimates are more than 5 feet off from the correct location. Figure 3.12 shows the projection of the model using the pose estimated by algorithm “R\_and\_T\_mod” (run with the low initial variance setting) and as can see from the figure, it is quite skewed.

From Tables 3.1 and 3.2, the error in locating the robot for this frame by the robust algorithms “Med\_R\_and\_T\_img” and “Med\_R\_and\_T\_mod” (run with high initial covariance) is about 0.38 feet in the y-axis and 0.32 feet in the z-axis (or vertical direction). Figure 3.13 shows the projection of the model using the pose estimated by “Med\_R\_and\_T\_mod” when run with high initial covariance. In that figure, it can be noticed that the line in depth in the top left corner of the image is slightly skewed.



All other lines are fairly well aligned. This line has been detected as an outlier. Note that in the Table 3.2, 9 lines have been labeled as outliers. Of the 11 lines in the data set for indoor frame 8, only 2 are lines in depth. The rest of the lines are coplanar and are clustered near the door. The best pose returned by the median is a consensus of only half the number of lines. As a result, one of the lines in depth remains out of the best consensus set and becomes an outlier.

In contrast, when algorithm “Med\_R\_and\_T\_mod” is run with the low prior covariance setting<sup>6</sup>, the height of the robot is pinned and the final estimate of location returned by the algorithm is within 0.05 feet and only the 8 outlier lines have been detected as outliers (see Table 3.4). Figure 3.14 shows the projection of the model using the pose estimated by “Med\_R\_and\_T\_mod” when run with low initial covariance. In this case, all lines, including the top left line in depth, are well aligned.

Finally, the “Tuk\_wts” algorithm performed very well on this data set, locating the robot to within 0.04 feet. The projection of the model for indoor frame 8 using the pose estimated by the “Tuk\_wts” algorithm is almost identical to the projection shown in Figure 3.14.

### 3.5.4 Outdoor Frame 3

The outdoor frame 3 data set (shown in Figure 3.15) is another case where the least-squares algorithms fails completely. The outlier here is due to the telephone pole being mismatched to the street light. Thus, 3 of the 15 input lines are clear outliers. The location of the robot returned by the least-squares algorithm is off by about 40 feet for the high initial covariance case (see Tables 3.1 and 3.2) and by about 10 feet for the low initial covariance case (see Tables 3.3 and 3.4). Figure 3.16 shows the projection of the 3D model using the pose estimated by algorithm “R\_and\_T\_mod”

---

<sup>6</sup>For subsets of size 6 or 8.

for the high covariance case.

### Analysis Using Hat Matrix

It was noted in Section 4.1 that data sets can be analyzed for outliers by examining the diagonal entries of the hat matrix<sup>7</sup>  $h_{ii}$ . These correspond to the relative contribution an observation has on its current fit<sup>8</sup>. Typically, observations with  $h_{ii}$  greater than two or three times the average value are considered observations with “high leverage” [65]. Outliers which have a significant effect on the final fit therefore should be detected as observations with “high leverage”. Unfortunately, the Hat matrix diagonal entries are not very robust and the values of  $h_{ii}$  for outliers can be masked by other “leverage” observations [65]. This fact is demonstrated by examining the residual errors and diagonal hat matrix values estimated by the least-squares algorithm “R\_and\_T\_img” on the outdoor frame 3 data set. In Table 3.5, the residual errors and diagonal hat matrix values are shown for the two end-points of each line. Lines 3-5 in Tables 3.5 correspond to the three outlier telephone lines in outdoor frame 3 (see Figure 3.15). Lines 6 and 7 correspond to the two lines of the street-light and line 8 corresponds to the walkway line. Table 3.5 is the result of running the algorithm “R\_and\_T\_img” on the complete data set (including three outlier lines) and hence the residual errors as seen in Table 3.5 are quite large. The near end-point of the walkway line in outdoor frame 3 (line 8 in Table 3.5) has a diagonal hat matrix value of 0.99 implying that it is having a very significant effect on the final fit. Meanwhile, the three lines corresponding to the outlier telephone pole (lines 3-5 in Table 3.5) have hat matrix values ranging from 0.16 to 0.20. These lines are therefore not characterized as having “high leverage” on the final fit. However, we

---

<sup>7</sup>See Appendix C for a definition of the Hat Matrix.

<sup>8</sup> $h_{ii}$  range from 0 to 1. Higher values of  $h_{ii}$  imply that the observation has a more significant effect on its final fit compared to the effect of other observations.

know that they are outliers and are causing the pose estimate to be offset by 40 feet from Table 3.1. This demonstrates the fact that the hat matrix values themselves are corrupted when outliers are present and therefore they cannot be used to detect outliers. Note also in Table 3.5, the telephone outlier lines (lines 3-5) do not have the largest residual errors. This demonstrates the fact that the highest residual errors need not belong to the outlier lines.

Table 3.5: Residual Errors (pixels) and Diagonal Hat Matrix entries for outdoor frame 2 for data with outliers. Outlier lines are marked with an asterik(\*).

Line No.	Residual Errors		Hat Matrix Diagonals	
	endpoint 1	endpoint 2	endpoint 1	endpoint 2
	pixels	pixels	$h_{ii}$	$h_{ii}$
1	-12.88	-20.28	0.18	0.16
2	-10.94	-19.85	0.17	0.15
3*	-11.89	-29.34	0.17	0.19
4*	-9.09	-28.57	0.17	0.20
5*	20.42	15.90	0.32	0.39
6	20.26	3.96	0.24	0.29
7	27.44	4.76	0.23	0.29
8	-1.68	-2.37	0.11	0.99
9	-9.96	-28.94	0.16	0.10
10	1.15	-7.66	0.12	0.13
11	-18.76	-34.60	0.10	0.13
12	-15.04	-22.32	0.12	0.13
13	14.22	-7.09	0.16	0.09
14	15.37	-5.68	0.16	0.10
15	17.37	-0.37	0.16	0.10

## Performance of the Robust Algorithms

The estimates of the robot pose returned by the two median algorithms are much better than the least-squares algorithms, but are still off by about 2.5 feet along the walkway direction for the both high and low initial covariance cases. The best pose returned by the median classifies the two lines of the street lamp on the right side of the image also as outliers (see Fig. 3.17). This is because in this image there are two sets of data lines that have significant “leverage” on the resultant pose: the walkway line (as mentioned earlier, lines in depth are important for these scenes) and the street lamp (which is the only non-outlier data in the right side of the image). Since both these sets of lines are slightly incorrect, the median chooses a pose which best explains the walkway line together with the rest of the data (which is mostly 300 feet away).

In contrast to its performance for indoor frame 8, the “Tuk\_wts” algorithm failed completely on outdoor frame 3. The results for it are shown in Table 3.1. The algorithm is extremely sensitive to initial estimates for outdoor frame 3 (see Fig 3.15). As noted earlier, outdoor frame 3 has three outlier lines (the telephone pole lines). For algorithm “Tuk\_wts” to converge to the right answer, the initial estimate must be within 1 foot of the correct translation. However, if one of the outlier telephone lines is removed from the data set, the algorithm will converge to the correct answer from initial estimates up to 10 feet or so away. Finally, if two of the telephone outlier lines are removed convergence to the correct answer is obtained from initial estimates upto 20 feet away. This problem of an accurate initial estimate is due to the presence of multiple local minima in the objective function minimized by the “Tuk\_Wts” algorithm. The iterative algorithms for M-Estimation techniques weight lines based on their residual errors, and therefore if the initial estimate is off, the outlier lines may get higher weights and the algorithm will then converge to a local

minima. A possible solution would be to use a global minimization technique for minimizing the M-Estimate error functions with some prior estimate of scale [10]. The algorithm “Med\_R\_and\_T”, on the other hand, is not affected by these initial estimates and produces the same answer as shown in the tables for all of them.

### 3.5.5 Discussion

The output of the pose refinement process depends on the quality of the input provided to it. The best data sets have many “good” lines in all directions and lines running both near and far from the camera. In contrast indoor frame 7 and outdoor frame 3 are impoverished data sets where most of the input lines are at approximately the same distance from the camera and one or two lines are much closer to the camera. As a result these close lines have a significantly larger effect in determining the final pose estimation and have “high leverage” in the final estimation. Note that the “high leverage” data lines can be detected by examining the diagonal values in the hat matrix when there are no outliers. For instance, in Table 3.5, the walkway line in Outdoor frame 3 had a hat matrix diagonal value of 0.99<sup>9</sup>.

A consensus based algorithm tries to find the best pose which explains a significant proportion of the set of lines. Given that the observations for the non-outlier data are noisy, it is conceivable that the pose returned by the consensus algorithm explains a significant set of observations with “low leverage” quite well and makes a non-outlier observation with “high leverage” an outlier. This is what happened in the indoor frame 8 and outdoor frame 3 case, where some of the lines with “high leverage” become outliers. A consequence (and an inherent danger) of the incorrect removal of the “high leverage” lines as outliers from an impoverished data set is that the output

---

<sup>9</sup>Note, this high value of the diagonal entry in the hat matrix pertaining to the the walkway line in Outdoor frame 3 is obtained even when the computation is done with the outlier lines removed *from the data set*.

covariance matrix of the computed pose parameters will be much higher than what is optimally obtainable for that data set. The consensus-based algorithms will work best when there are no observations with “high leverage” and the data set is well rounded with a sufficient number of input data lines in all directions both close and far to the camera.

### 3.6 Conclusions

To conclude, we quote from Li [51] “No one robust regression technique has proved superior to all others in all situations, partly because of the challenge of handling many forms of influential observations”. We have observed this especially with the performance of the median and other similar consensus-based algorithms using many different error functions<sup>10</sup>. For instance, in another consensus algorithm we experimented with, instead of minimizing the median, the optimization criterion was to find that pose which fits the maximum number of lines under a certain error value. The results of this algorithm were of a similar nature to the median based algorithm. There appears to be no one algorithm which works best for all data sets. The robust statistical algorithms presented in this chapter, given data with outliers, can locate the robot in the indoor scenes in a range of 0.4 feet (on the average) for the high prior covariance cases and within 0.3 feet (on the average) for the low prior covariance cases. This was sufficient for most of our robot navigation tasks. However, the question remains can we do better?

---

<sup>10</sup>Although only two are reported here.

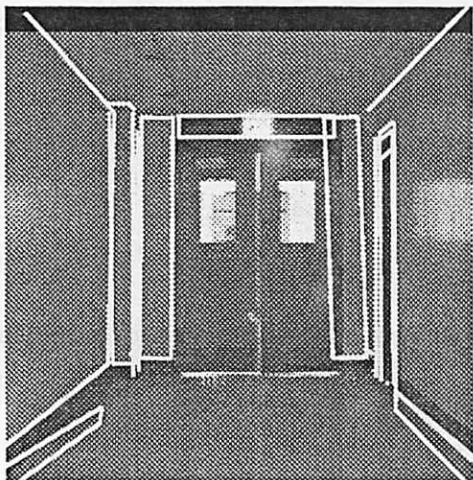


Figure 3.3: Indoor frame 7 with input image lines.

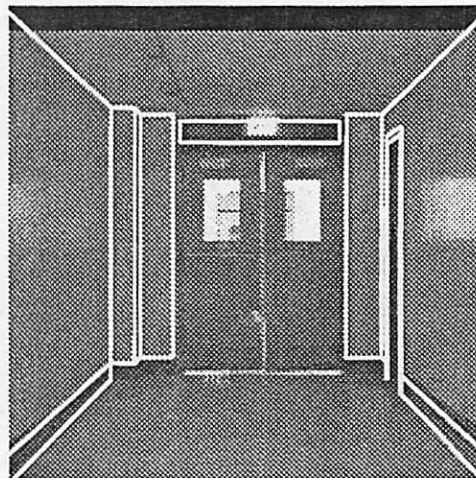


Figure 3.4: Projection of model for indoor frame 7 using final estimate of Algorithm "R\_and\_T\_mod" with high prior covariance matrix.

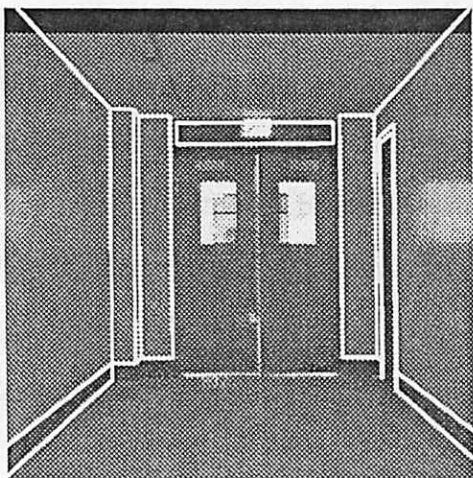


Figure 3.5: Projection of model for indoor frame 7 using final estimate of Algorithm "Med\_R\_and\_T\_mod" with high prior covariance matrix.

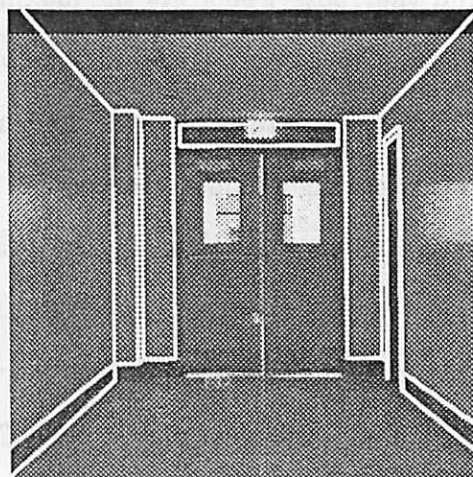


Figure 3.6: Projection of model for indoor frame 7 using final estimate of Algorithm "Tuk\_wts".

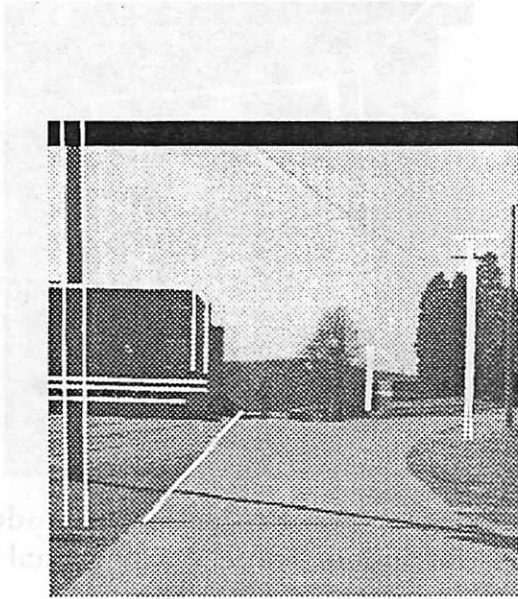


Figure 3.7: Outdoor frame 1 with input image lines.

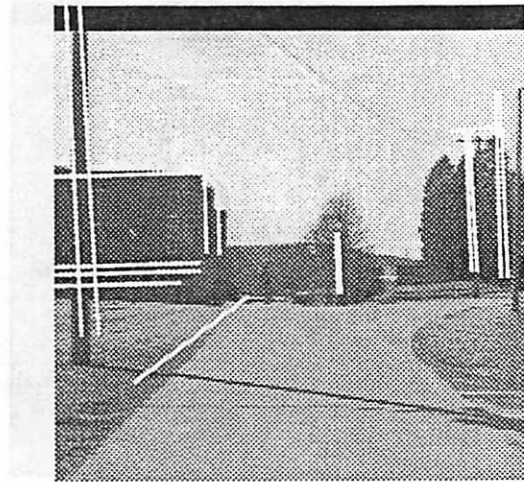


Figure 3.8: Projection of model for outdoor frame 1 using final estimate of Algorithm "R\_and\_T\_mod" with low prior covariance matrix.

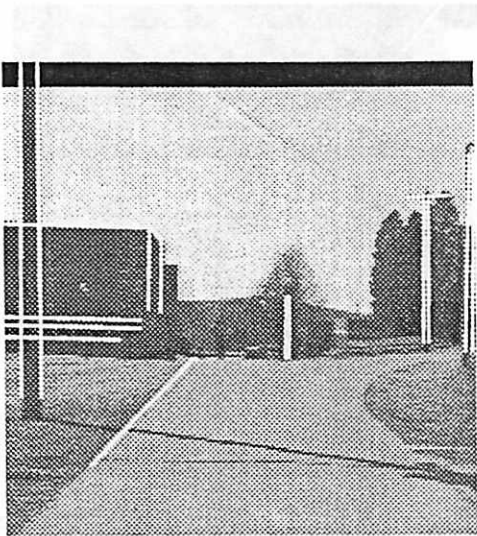


Figure 3.9: Projection of model for outdoor frame 1 using final estimate of Algorithm "Med\_R\_and\_T\_mod" with low prior covariance matrix.

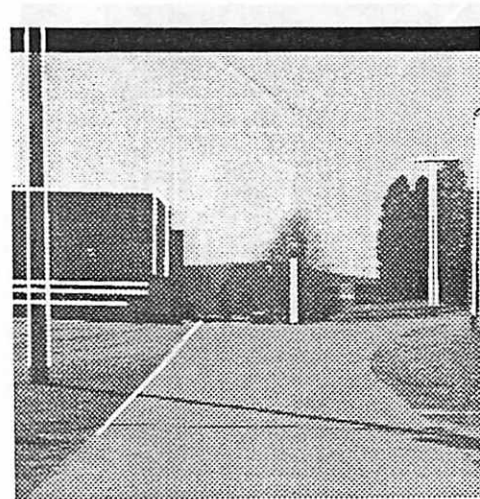


Figure 3.10: Projection of model for outdoor frame 1 using final estimate of Algorithm "Tuk\_wts".



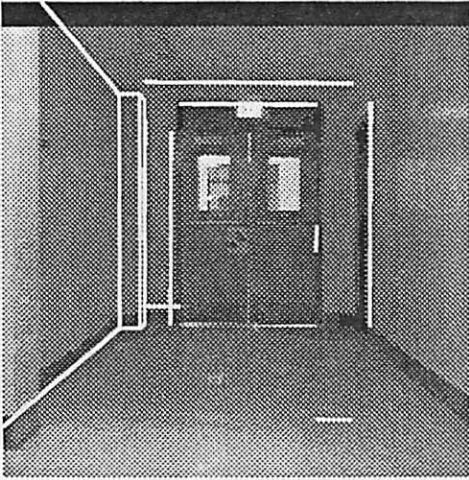


Figure 3.11: Indoor frame 8 with input image lines.

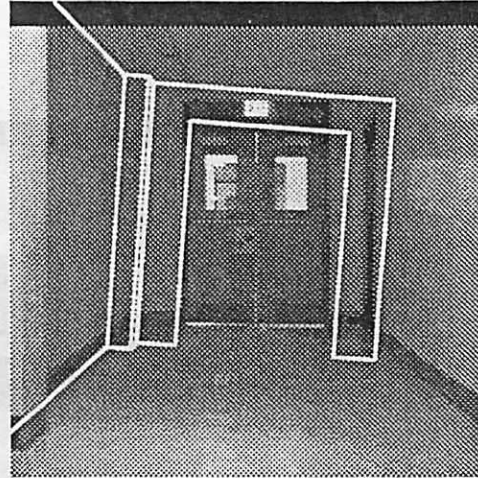


Figure 3.12: Projection of model for indoor frame 8 using final estimate of Algorithm "R\_and\_T\_mod" with high prior covariance matrix.

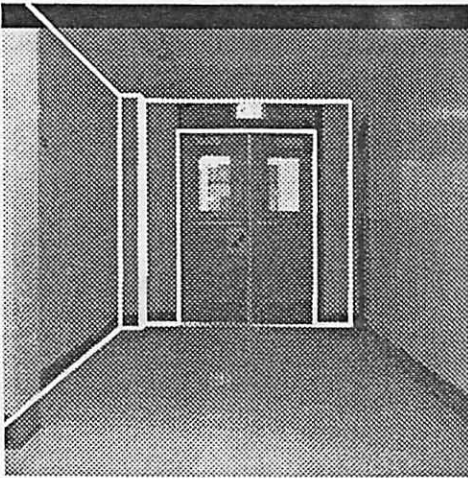


Figure 3.13: Projection of model for indoor frame 8 using final estimate of Algorithm "Med\_R\_and\_T\_mod" with high prior covariance matrix.

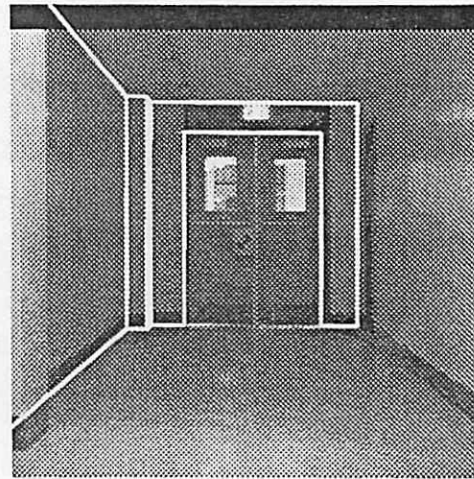


Figure 3.14: Projection of model for indoor frame 8 using final estimate of Algorithm "Tuk\_wts". Note that the projection of model using final estimate of "Med\_R\_and\_T\_mod" with low prior covariance is almost identical.

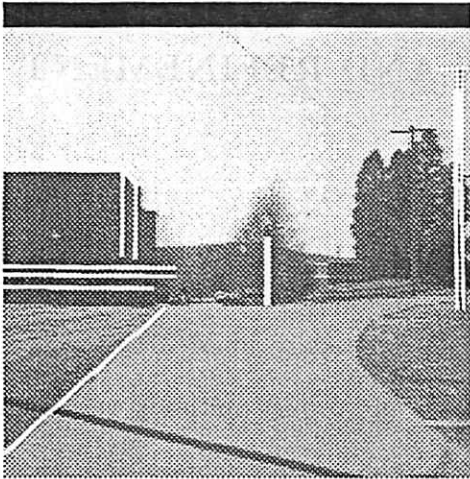


Figure 3.15: Outdoor frame 3 with input image lines.

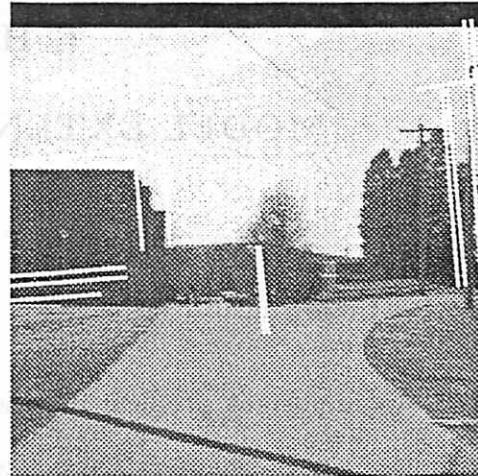


Figure 3.16: Projection of model for outdoor frame 3 using final estimate of Algorithm "R\_and\_T\_mod" with high prior covariance matrix.

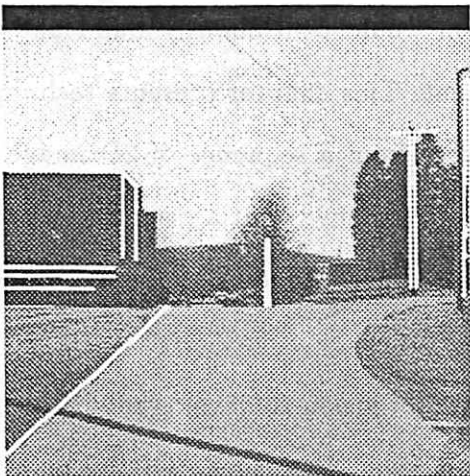


Figure 3.17: Projection of model for outdoor frame 3 using final estimate of Algorithm "Med\_R\_and\_T\_mod" with high prior covariance matrix.

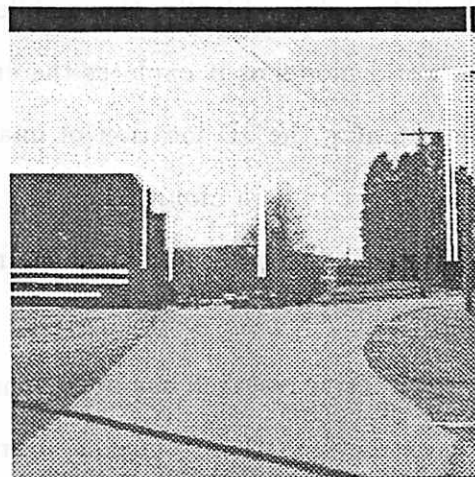


Figure 3.18: Projection of model for outdoor frame 3 using final estimate of Algorithm "Tuk\_wts".

## CHAPTER 4

### MODEL EXTENSION AND REFINEMENT

#### 4.1 Introduction

An important problem in vision is to automatically build 3D models of objects and scenes. In Chapters 2 and 3, least-squares and robust methods were presented for determining the location and orientation of a mobile robot from visual measurements of modeled 3D landmarks. However, building the 3D landmark models is a time consuming and tedious affair. For the landmark-based navigation methods to be widely applicable, automatic methods have to be developed to build and enhance the 3D models. Ideally, the mobile platform would continuously build and update its world model as it explores the environment. This chapter presents techniques for determining the 3D location of image features from a sequence of 2D image frames taken by a camera mounted on the robot. It is assumed that a prior partial model is available. The goal is to have the robot extend and refine this model as it explores the world.

Extensive research has been done in computer vision to develop robust algorithms for extracting 3D information from a sequence of 2D images. Of the many different visual cues for extracting 3D information, the two most extensively researched are stereo and motion. The basic principle exploited in both cues is triangulation (see Figure 4.1). New points are located by triangulating the projection rays from *corresponding* points in two or more frames.

In applications involving stereo, two cameras separated by a baseline are used to provide the two image frames. The relative orientation of the two fixed cameras is determined during a prior calibration stage. Thus, the main problem and focus of stereo research has been to establish correspondences [6, 15, 58, 63].

In two-frame motion analysis both the correspondences and the relative orientation between the two camera frames are unknown. Research in motion analysis has classically been divided into two steps. In the first step inter-frame image displacements of image pixels and/or higher level tokens are computed. The second step, also known as “structure from motion” or “relative orientation”, is the interpretation of these displacements (or correspondences between image tokens) into 3D structure and relative orientation (rotation and translation) between frames [1, 35, 37].

However, due to noise in the measurement process, results for both stereo and motion analysis using just two frames are not very robust [1, 17]. To improve the robustness of the results, the traditional stereo and structure from motion techniques have been extended to deal with multi-frame image sequences [3, 5, 11, 32, 61, 69, 82], under the assumption that temporal integration would lead to more robust results. The multi-frame research can be categorized into two broad classes or strategies. The first class assumes that a model of 3D inter-frame motion is known [11, 12, 67, 68], rather than assuming independent motion parameters between consecutive frames. Broida [11] assumes constant velocity motion and estimates the 3D location of a set of points tracked over a monocular image sequence. Recently, Chandrasekhar et.al. [12] have extended Broida’s technique to deal with data sets where the 3D location of a few points is known. The objective function, which Broida and Chandrasekhar et.al. minimize has the motion model parameters and the unknown structure location parameters as unknowns. Thus the dimension of the objective function grows with the number of unknown points. An even more basic limitation of this approach lies

in the model of motion being employed and its adequacy in cases with more general motion.

The second class of techniques does not assume any model of motion. The rigid structure of the world is carried forward by the depth estimates from frame to frame. These techniques are incremental in nature and typically use Kalman-filtering to compute the depth estimates [5, 13, 61, 69, 75, 82].

Both Ayache et.al. [5] and Zhang et.al. [82] build world models using multi-frame stereo sequences. Zhang et.al. [82] track 3D line segments over a sequence of stereo image frames and use a Kalman-filter to integrate the results for a final estimate of the 3D line segment. To do the temporal integration, the absolute orientation between successive stereo pair coordinate frames is determined.

Oliensis and Thomas [61] use Horn's relative orientation algorithm [35] to solve for the motion parameters between consecutive image frames in a monocular image sequence. With each image pair, new measurements are made for depth values of features and these are integrated with previous estimates in the Kalman-filter framework. The new observation Oliensis and Thomas [61] make is that the depth estimates of different feature points are correlated since the same noisy motion parameters are used to compute the depth. Because of this correlation, they estimate the depth parameters of all points simultaneously. This gives them fairly good depth estimates for camera motions having some translation component along the optical axis. The cost, however, is that for estimating the depths of "m" points, a covariance matrix of size  $(3m \times 3m)$  must be inverted with each new frame.

Sawhney et.al. [69] also use Kalman-filtering to estimate the depths of "shallow structures" over a monocular sequence of multiple image frames. Shallow structures are those whose extent in depth is small compared to their average depth from the camera. The algorithm, however, cannot handle non-shallow structures. The image

motion of shallow structures can be described by an affine transform. Based on the affine trackability of an object, they are able to segment out different shallow structures in the scene and hence can potentially handle multiple moving objects. In some of the experiments reported in the results section of this chapter, an initial model is built using the 3D points lying on some of the shallow structures recovered by their algorithm. Using this initial model, the 3D location of other points in the scene is estimated by the techniques developed in this chapter. Thus with a combination of techniques presented in this chapter and Sawhney et.al.'s [69] technique for 3D recovery of shallow structures, a fairly robust general motion technique [42] may be constructed.

#### 4.1.1 Our Approach

The approach adopted here is to first begin with a partial model (possibly noisy) and to then extend and refine it by viewing the object over a sequence of frames [43]. Both modeled and unmodeled features of the object are tracked over the image sequence by using an optic flow based line tracking algorithm [4, 76]. Correspondences are obtained between the modeled 3D features and their image projections. Using the flow of image tokens and the poses of the object computed from model-image feature correspondences for a sequence of image frames, new points are located by triangulation (see Figure 4.1). The triangulation process is also used to make new 3D measurements of the initial model points. These measurements are then fused with the previous estimates to refine the set of initial model points. The approach adopted here is basically induced stereo. Tracking image features over a large sequence effectively leads to a large baseline for stereo and improves the robustness of the 3D reconstruction. Note that this approach does not require any models of inter-frame motion.

The key assumption made is that a partial model is available at the beginning of the process. Due to the availability of the partial model, new points are located in a stable world coordinate system. The pose computed from each image frame is independent of the other frames, so each frame provides an independent measure to the whole process<sup>1</sup>. This does not lead to the cascading problems which most of the sequential multi-frame “structure from motion” techniques suffer from because noisy prior estimates in the previous frame’s coordinate system are integrated with new estimates in the current frame’s coordinate system.

The estimation of the new 3D points is done using both batch and quasi-batch or sequential methods. Triangulation requires at least two frames and therefore the minimum batch size is two. Results from batch to batch are integrated by the standard Kalman-filter covariance based updating equations. Results are presented for four real data sequences where new 3D points are located with average errors less than 1.7 %. These results are far superior to those obtained by the traditional structure from motion techniques employed in computer vision. This supports the earlier stated premise that prior knowledge of a partial model greatly extends the robustness of the structure estimates.

The errors in the initial partial model are assumed to be either gross errors or gaussian noise. If gross errors are present in the 3D model, these would be detected as outliers by the robust pose recovery techniques developed in Chapter 3 and would not be used for the final step of least-squares fitting to the remaining non-outlier data. Note that outliers can also arise due to incorrect correspondences. However, if a modeled landmark appears as an outlier over a large number of frames, then it probably is due to a gross error in the 3D model and it could eventually be removed from the 3D model database. Thus for the remainder of this chapter, the noise in the

---

<sup>1</sup>Note that this would not be true if there was significant noise in the initial partial model.

input 3D model is assumed to be gaussian. Section 4.2 extends the least-squares algorithms for pose determination (presented in Chapter 2) to handle gaussian noise both in the 3D model and image measurements. In Section 4.2.1, results of monte-carlo experiments for the new pose determination algorithm are presented. Experimentally derived standard deviations of the error in the estimated pose parameters are compared with the computed standard deviations. Section 4.3 presents the mathematics for locating new points and refining old points using the computed poses and their respective variances. Finally, Section 4.4 presents and analyzes results from real data experiments. Some concluding remarks are presented in Section 4.5.

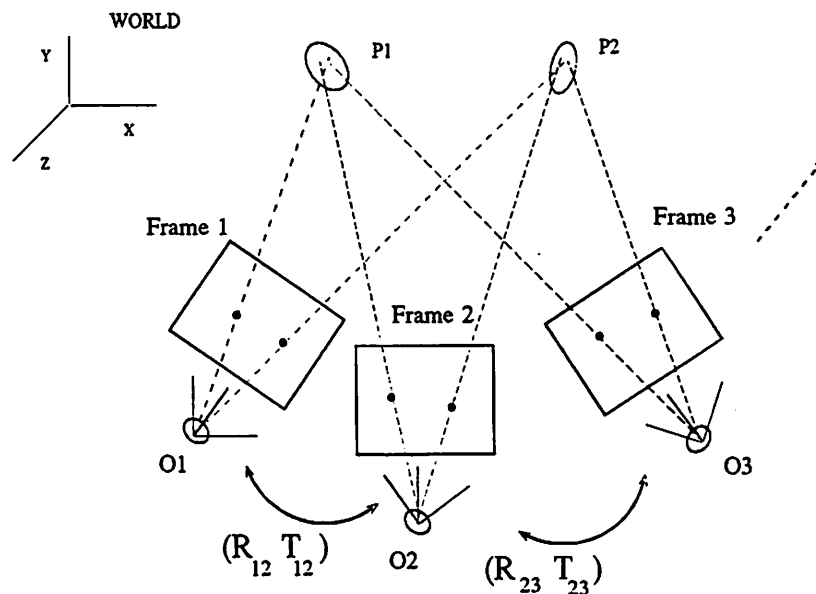


Figure 4.1: Model Extension and Refinement.



## 4.2 Pose Determination

In Chapter 2, least-squares techniques for pose determination were developed. These techniques are optimal with respect to gaussian noise in the input image measurements. In this section, the least-squares techniques are extended to handle gaussian noise in the 3D model. The techniques presented in this section assume point correspondences but are easily modified for line correspondences.

The rigid body transformation from the world coordinate system to the camera coordinate system can be represented as a rotation ( $R$ ) followed by a translation ( $\vec{T}$ ). The point  $\vec{p}$  in world coordinates gets mapped to the point  $\vec{p}_c$  in camera coordinates:

$$\vec{p}_c = R(\vec{p}) + \vec{T} \quad (4.1)$$

Using the equation (4.1) and assuming perspective projection, the pose constraint equations for the  $i^{th}$  point  $\vec{p}_i$  in a set of “m” points can be written in the following manner:

$$\frac{1}{p_{czi}} \vec{C}_{xi} \cdot (R\vec{p}_i + \vec{T}) = 0 \quad (4.2)$$

$$\frac{1}{p_{czi}} \vec{C}_{yi} \cdot (R\vec{p}_i + \vec{T}) = 0 \quad (4.3)$$

$$\vec{C}_{xi} = (s_x, 0, -I_{xi}) \quad (4.4)$$

$$\vec{C}_{yi} = (0, s_y, -I_{yi}) \quad (4.5)$$

$$p_{czi} = (R\vec{p} + \vec{T})_z \quad (4.6)$$

$(I_{xi}, I_{yi})$  is the image projection of the point and  $(s_x, s_y)$  is the focal length in pixels along each axis.

Since both the image measurements and the 3D model locations are assumed to be noisy, it will not be possible to satisfy the above constraint equations exactly. Let the measurement error in pixels of image point locations be given by  $(\Delta X, \Delta Y)$  and

the error in the 3D model points be given by  $\vec{\Delta p}$ . Given a current estimate  $R, \vec{T}$ , the constraint equations (4.2,4.3) are linearized about the estimate:

$$\frac{1}{p_{czi}}(\vec{C}_{xi} \cdot \vec{\Delta T} + \delta\vec{\omega} \cdot \vec{b}_{xi}) = -\frac{1}{p_{czi}}\vec{C}_{xi} \cdot \vec{p}_{ci} + \eta_x \quad (4.7)$$

$$\frac{1}{p_{czi}}(\vec{C}_{yi} \cdot \vec{\Delta T} + \delta\vec{\omega} \cdot \vec{b}_{yi}) = -\frac{1}{p_{czi}}\vec{C}_{yi} \cdot \vec{p}_{ci} + \eta_y \quad (4.8)$$

where  $\vec{b}_{xi} = R\vec{p}_i \times \vec{C}_{xi}$  and  $\vec{b}_{yi} = R\vec{p}_i \times \vec{C}_{yi}$ . The noise terms in the two equations,  $\eta_x$  and  $\eta_y$  are functions of both model noise  $\vec{\Delta p}$  and image noise  $\Delta X, \Delta Y$ :

$$\eta_x = \Delta X + \frac{1}{p_{czi}}\vec{C}_{xi} \cdot (R(\vec{\Delta p}_i)) \quad (4.9)$$

$$\eta_y = \Delta Y + \frac{1}{p_{czi}}\vec{C}_{yi} \cdot (R(\vec{\Delta p}_i)) \quad (4.10)$$

Therefore for the  $i^{th}$  point, two linear equations (4.7 and 4.8) can be written and for a set of “m” points, a total of “2m” equations is obtained. This linear system of equations relate the pose increments  $\delta\omega$  (rotation) and  $\Delta T$  (translation) to the computed measurement errors using the current pose estimate. The system of “2m” equations is similar to the linear system of equations (C.1) described in Appendix C. Using the formula for the best linear unbiased estimate described in equation(C.2) in Appendix C, the linear system of equations is solved at each iteration to find the best increment vector. This increment is added to the current pose estimate and the process repeated until there is convergence.

In the above system of equations,  $(\eta_x, \eta_y)$  represents the measurement noise. If the correct estimate of pose were known,  $\eta_x$  and  $\eta_y$  would be equal to the sum of the measurement error of the image point location and the projection of the error in the model point along the image x-axis and y-axis respectively. The measurement of the image point location is assumed to be corrupted with zero-mean independent gaussian noise. In our case, for lack of a better noise model, it is assumed that the image noise in the measurements is independent across all points and is also the same.

The 3D model points are also assumed to be corrupted by zero-mean independent gaussian noise. Therefore, in the “2m” system of linear equations, the noise in the two equations for every point is correlated. Thus the covariance matrix “V” corresponding to the noise in the linear system of equations (C.1) in Appendix C is a band matrix in which the non-zero entries are  $(2 \times 2)$  matrices about the diagonal. If the model noise was zero and the noise in the image measurements were assumed to be same for all points, then the input covariance matrix would be an identity matrix scaled by the standard deviation of image noise. The output covariance matrix for the pose rotation and translation parameters is given by equation (C.3) evaluated at the final pose estimate.

#### 4.2.1 Experimental Results: Pose Algorithm

In this subsection, results of Monte-Carlo experiments using the pose algorithm described above are presented. The synthetic data used for the experiments was generated from the 3D model constructed for the A211 sequence described in Section 2.5. The nine points marked in Figure 2.9 were used for the experiments. For each experiment, given an input pose, the 3D model of the nine points is projected to create a 2D image data set. The field of view of the virtual camera was set to be  $29.27^\circ \times 22.86^\circ$  and image size of the sensor was assumed to be  $(256 \times 256)$  pixels. The 3D model points and the projected 2D image points are corrupted with gaussian noise. Noise for each point was assumed to be zero-mean, identically distributed, and independent. For each noise specification, 1000 noisy sample sets were created and the pose algorithm was run on each of the samples. From each sample run, both the estimated pose and the estimated pose error covariance matrix were collected. The estimated standard deviations of the pose parameters are computed from the diagonal terms of the error covariance matrix. The actual standard deviations of

Table 4.1: Experimental and Computed standard deviation of Translation and Rotation error in world coordinates for algorithms “R\_and\_T” for point data. The algorithm was run on a data set generated from the model created for the A211 sequence. The statistics for each experiment is taken over 1000 samples of gaussian noise.

NO.	NOISE $\sigma$		TRANSLATION ERR.			ROTATION ERR.		
PTS.	IMG. pixels	3D feet	$\Delta T_x$ feet	$\Delta T_y$ feet	$\Delta T_z$ feet	$\delta\omega_x$ deg.	$\delta\omega_y$ deg.	$\delta\omega_z$ deg.
EXPERIMENTAL STANDARD DEVIATIONS								
9	1.0	0.0	0.031	0.032	0.119	0.124	0.118	0.334
9	3.0	0.0	0.092	0.094	0.355	0.369	0.351	1.001
9	5.0	0.0	0.151	0.154	0.583	0.607	0.579	1.668
9	1.0	0.5	1.540	1.640	3.395	2.679	2.492	5.029
9	1.0	1.0	1.539	1.635	3.382	5.466	5.034	10.224
9	3.0	1.0	1.540	1.640	3.395	5.485	5.035	10.254
COMPUTED STANDARD DEVIATIONS								
9	1.0	0.0	0.031	0.032	0.118	0.122	0.118	0.321
9	3.0	0.0	0.092	0.095	0.352	0.364	0.353	0.958
9	5.0	0.0	0.152	0.156	0.579	0.599	0.581	1.583
9	1.0	0.5	0.762	0.799	1.704	2.648	2.487	5.026
9	1.0	1.0	1.498	1.574	3.402	5.194	4.846	9.016
9	3.0	1.0	1.502	1.577	3.414	5.206	4.861	9.051

the pose parameters are also calculated from the sample set of 1000 pose estimates. These are both shown in Table 4.1. Under the heading “Experimental Standard Deviations”, the actual standard deviations of the estimated pose parameters over the 1000 samples are reported. Under the heading “Computed Standard Deviations”, the average of the computed standard deviations over 1000 samples are reported. The statistics ( $\sigma$ ) of the input 2D and 3D gaussian noise is specified in Columns 2 and 3 of Table 4.1. Columns 4-6 and 7-9 list the standard deviations of the translation and rotation errors respectively.

From Table 4.1, various observations can be made about the performance of the algorithm. The first three rows report experiments with no input 3D model noise. The pose recovery algorithm is able to locate the robot with 0.6 feet and  $1.7^\circ$  when the image noise has a standard deviation as large as 5 pixels. However, when even moderate amount of 3D model noise is introduced, the performance of the algorithm rapidly degrades. This is because, for this experiment, the 3D model points ranged from 8 feet to 20 feet away from the camera. Thus noise of 0.5 feet or 1 feet has a very large effect on the estimated pose. For instance, 3D noise of 0.5 feet in a model point 20 feet away can cause the projection error in the image to be approximately 15 pixels. The corresponding error for points closer to the camera would be much larger. Finally, comparing corresponding rows in Table 4.1, it is noted that the “Computed Standard Deviations” are fairly close to the “Experimental Standard Deviations”. This provides experimental confirmation of the claim that the estimated standard deviations are quite reliable. Note that the standard deviations of the 1000 sample standard deviations was also calculated (although not reported here) and found to be quite small.

### 4.3 Induced Stereo

In this section, we present techniques for computing 3D estimates of new points in the world coordinate system from their tracked image locations over a multi-frame sequence. The mathematics for both extending the model and refining the initial modeled points is presented. Computed with the estimate of each new model point is an estimate of its error covariance. These covariances are functions of the input image measurement covariances and the initial 3D model point covariances.

Image features (both new features and modeled image features appearing in the images) are tracked over a sequence of frames using the computed optic flow between pairs of successive frames [76]. Typically corners (defined by the intersection of two image lines) are tracked although any image feature which can be reliably tracked may be used. The initial matching of image features to the partial model for the first frame may be done by a matching process such as in [8]. Combining the results of the initial matching and the feature tracking, correspondences between image features and the partial model for each frame are established. Using these correspondences, pose estimation is done for each frame using the method presented in the previous section.

The image projection ray for an image point in a particular frame is defined as the ray originating from that frame's optic center and passing through the image point. Given the pose estimates for each frame, the vectors corresponding to these projection rays in the world coordinate system can be obtained. The 3D estimate of the point is the pseudo-intersection of all the image projection rays for a tracked image point (see Figure 4.1). In order to combine 3D measurements from a sequence of frames, a stable coordinate frame should be used; a nice property of the system described here is that the pose estimation process provides the world coordinate frame as this *stable* coordinate frame. Independent measurements can be made relating the coordinate

system of each frame in the sequence to the world coordinate frame.

Points are located by the pseudo-intersection process in two steps. In the first step, a 3D error function is minimized to find an initial estimate of the point's location. This step, however, does not yield the optimal estimate since the various error terms are not weighted by the input covariances. In the second step, an image-based error function is optimized in which the error terms are inversely weighted by a combination of the input covariances of the pose estimate and the image measurements.

Let  $r_i$  be the unit vector corresponding to the image projection ray for an image point in the  $i$ 'th frame. The pose estimation for this frame is given by the rotation  $R_i$  and translation  $\vec{T}_i$  (see equation (2.1)). Since the image projection rays do not intersect at a unique point<sup>2</sup>, the 3D pseudo-intersection point  $p_i$  is obtained by minimizing an error function  $E$ :

$$E = \sum_{i=1}^n \|(R_i(\vec{p}) + \vec{T}_i) \times r_i\|^2 \quad (4.11)$$

Therefore the 3D error function  $E$  (used in the first step) is the sum of squares of the perpendicular distances from the pseudo-intersection point  $\vec{p}$  to the image projection rays. Differentiating  $E$  with respect to the unknown variable  $\vec{p}$  leads to a set of linear equations, which are then solved to give the initial estimate for  $\vec{p}$ .

In the second step, the pose constraint equations (4.2, 4.3) are used to formulate image-based error equations for the X and Y projections of the model points.

$$\frac{1}{p_{cz}} \vec{C}_{xi} \cdot R_i(\vec{p}) = -\frac{1}{p_{cz}} \vec{C}_{xi} \cdot \vec{T}_i + \zeta_X \quad (4.12)$$

$$\frac{1}{p_{cz}} \vec{C}_{yi} \cdot R_i(\vec{p}) = -\frac{1}{p_{cz}} \vec{C}_{yi} \cdot \vec{T}_i + \zeta_Y \quad (4.13)$$

where  $\zeta_X$  and  $\zeta_Y$  are the noise terms in the two equations.  $\zeta_X$  and  $\zeta_Y$  are functions of both noise in pose  $\Delta\vec{T}_i$  and  $\delta\vec{\omega}_i$  and image noise ( $\Delta X, \Delta Y$ ):

$$\zeta_X = \Delta X + \frac{1}{p_{cz}} \vec{C}_{xi} \cdot \Delta\vec{T}_i + \frac{1}{p_{cz}} \vec{b}_{xi} \cdot \delta\vec{\omega}_i \quad (4.14)$$

---

<sup>2</sup>Due to noise both in image measurements and pose estimates.

$$\zeta_Y = \Delta Y + \frac{1}{p_{cz}} \vec{C}_{yi} \cdot \Delta \vec{T}_i + \frac{1}{p_{cz}} \vec{b}_{yi} \cdot \delta \vec{\omega}_i \quad (4.15)$$

In this case the 3D model point  $\vec{p}$  is the unknown variable. The denominator  $p_{cz}$  in the equations (4.12 and 4.13) corresponds to the depth of the point and is a function of the unknown variable  $\vec{p}$ . Therefore for each frame over which the point is tracked, two non-linear constraint equations (4.12 and 4.13) are obtained<sup>3</sup>. An iterative procedure is employed to solve the system of non-linear equations. At each iteration, the denominator  $p_{cz}$  is held constant using the previous estimate of  $\vec{p}$  and the resulting linear system of equations is solved using equation (C.2) (see Appendix C). The iterative procedure is repeated until there is convergence. In practice, we have found one iteration is sufficient for good results. The input covariance matrix  $V$  required in equation (C.2) is obtained from the expressions derived above for the noise terms  $\zeta_X, \zeta_Y$ . The output covariance of the 3D point estimate is given by equation (C.3) in Appendix C.

In the batch method, information from all frames is used simultaneously to estimate the 3D locations of tracked image points. However, it may be desired to sequentially update the location of new points after every pair (or a larger set) of frames. In the sequential or quasi-batch mode, equations (4.12 and 4.13) are again used to estimate the 3D location of image points tracked over the current set of frames. However, these new estimates must be fused with the previous estimates to obtain the current optimal estimate. Associated with each estimate is a covariance matrix representing the uncertainty in the estimate. These covariance matrices are used to fuse the two estimates and provide a new uncertainty matrix using the standard Kalman Filtering equations.

Let the estimate of the point's 3D location and its covariance at frame " $t_1$ " be  $\vec{p}(t_1)$  and  $\Lambda_p(t_1)$  respectively. A new 3D location estimate  $\vec{Q}$  with uncertainty estimate

---

<sup>3</sup>A minimum of two frames is needed to solve the system of equations.



(covariance matrix  $\Lambda_Q$ ) is computed from a batch of “n” image frames. The fused location estimate  $\vec{p}(t_n)$  and updated covariance matrix  $\Lambda_p(t_n)$  at frame “ $t_n$ ” are given by:

$$\vec{p}(t_n) = \Lambda_p(t_n)(\Lambda_p(t_1)^{-1}\vec{p}(t_1) + \Lambda_Q^{-1}\vec{Q}) \quad (4.16)$$

$$\Lambda_p(t_n) = (\Lambda_p(t_1)^{-1} + \Lambda_Q^{-1})^{-1} \quad (4.17)$$

This same method is used for model refinement. Initial model points have associated with them their input covariance matrices. When the model is tracked over a new batch of frames, 3D measurements can also be made for the model points by the above pseudo-intersection procedure. These new measurements are fused with the old estimate using the above equation.

#### 4.3.1 Model Extension and Refinement Algorithm

The algorithm for model extension and refinement using a current batch size of “n” ( $n \geq 2$ ) frames can be summarized as follows:

**Step 1** Given a partial 3D model and an image, establish correspondences between model points and image points using a matching technique such as in [8].

**Step 2** Track image points over the batch of “n” frames using the computed optic flow between successive pairs of images [76].

**Step 3** Using the correspondences established above between model points and image points, compute the pose for each image frame using the method described in Section 4.2.

**Step 4** Estimate the 3D location of both new points and initial model points in world coordinates using the two-step approach developed in Section 4.3 and

the feature correspondences established in Step 2 for the current batch of "n" frames.

**Step 5** Fuse initial estimates of both the new points and the model points with any previous estimates using equations (4.16) and (4.17).

#### 4.4 Experimental Results

The model extension and refinement algorithm has been applied to four image sequences. Figures 4.2, 4.4, 4.6 and 4.7 show example images from the BOX, PUMA, A211 and COMP sequences respectively. In all experiments the image center was assumed to be at the center of the image frame and the effective focal length was calculated from the manufacturers specification sheets. Since we show in Chapter 5 that errors in the image center do not significantly affect the location of new points in a world coordinate system, calibration for the image center has not been done. The image sequences were captured with a SONY B/W AVC-D1 camera, with an effective FOV of approximately  $(23^\circ \times 23^\circ)$  and  $(40^\circ \times 40^\circ)$  for the BOX and PUMA sequences respectively. The images in the A211 and COMP sequence had an approximate FOV of  $(29.27^\circ \times 22.86^\circ)$ . The images in all sequences were digitized to 256-by-242 pixels.

##### 4.4.1 BOX Sequence

The first sequence (referred to as the BOX sequence) was generated by rotating the box (in Fig. 4.2) about its central vertical axis, while the camera was kept stationary. Consecutive images in the sequence were taken after a rotation of approximately 3.6 degrees. In the first frame, the camera was about 650 mm distant from the top front corner of the box. The location of 30 points (marked in Fig.4.2 by circles and crosses)

in a world coordinate system was measured to an accuracy of approximately 1 mm along each axis. The depth of the points (in the first frame's coordinate system) used in our experiment varied from 575 mm to 700 mm. The thirty points were tracked over the set of 8 frames.

The fifteen points marked by crosses in Figure 4.2 were used as the initial model to do pose estimation for each frame. Various experiments were performed with different amounts of synthetic uniform noise added to the measured 3D locations of the cross points. Using the computed poses, 3D estimates of the remaining 15 points (marked by circles in Figure 4.2) were computed. In addition, the initial model of 15 (cross marked) points was refined. The algorithm described in Section 4.3 was run in a batch mode over all 8 frames to perform these experiments; the results are reported in Table 4.2. The first column of Table 4.2 gives the range of noise added to the initial model points. Thus a 10 mm entry in the first column means uniform noise in the range of  $\pm 10$  mm was added to each of the 3D coordinates of the model points. The average error<sup>4</sup> of the 15 initial model points for each experiment (prior to any refinement) is given in the second column of Table 4.2. The third column in the table shows the results of the model refinement process; it gives the average output error of the 15 (now refined) initial model points. The fourth column in the table shows the results of the model extension process; it gives the average output error of the 15 new (circle) points.

As can be seen from the first row in Table 4.2, the average error for model extension when there is no noise in the initial model is 1.38 mm. The maximum error was 2.6 mm and the minimum error was 0.44 mm. The average percentage error was 0.25 %. The percentage error is calculated by dividing the absolute 3D error by the depth of the point from the origin of the camera in the first image's coordinate frame. As

---

<sup>4</sup>The average error is the root mean square (RMS) value of the 3D location error of all points.

Table 4.2: Computed average output 3D location errors for model extension process with noisy input model points for the Box Sequence of 8 frames. Input Noise to model is synthetic uniform noise.

Uniform 3D Input Noise	RMS Input Noise	RMS Output Noise	
		Initial Points	New Points
$\pm$ mm	mm	mm	mm
0	0.00	0.00	1.38
1	1.02	1.01	1.69
2	1.95	1.52	1.92
3	3.06	2.00	2.23
5	4.49	3.00	3.78
7	6.96	3.32	3.84
10	10.25	4.16	6.31
20	17.29	10.32	16.23

the noise in the initial model increases, the errors in model extension and refinement also increases. However, except for the first two cases in Table 4.2, the average output error for both model extension and refinement were significantly lower than the average input error of the initial model points.

The model extension and refinement algorithm was also run in a sequential mode, where new 3D locations were computed after every new pair of frames and the results were fused with previous estimates. Figure 4.3 shows the results of such an experiment. For this experiment, the range of input noise was 5mm and the average error of the initial model points was 4.49 mm (corresponding to the fifth row in Table 4.2). The average output error in location of both the initial model points and the new (circle) 3D points is plotted for every image frame in the sequence. As can be noticed in the figure, the 3D error in both the initial model points and the unknown points monotonically decreases between each frame. The average error of the new points is reduced from 6.5 mm after the first pair of frames to about 3.7 mm at the end. The average error of the initial points is reduced from 4.49 mm to about 2.8 mm.

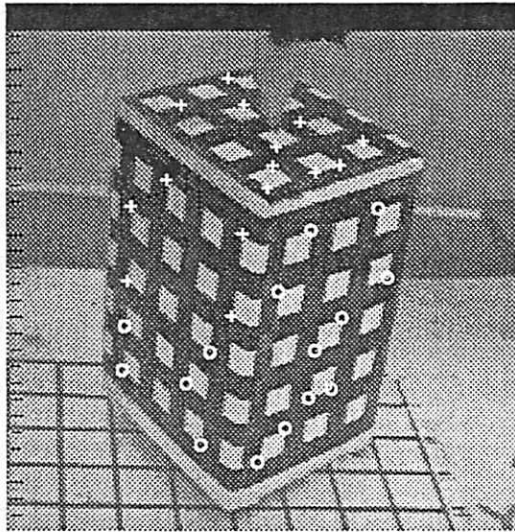


Figure 4.2: Box Image. The points marked by crosses were used to compute the 3D pose for each frame. Using these poses, the 3D location of the numbered points marked by circles is computed.

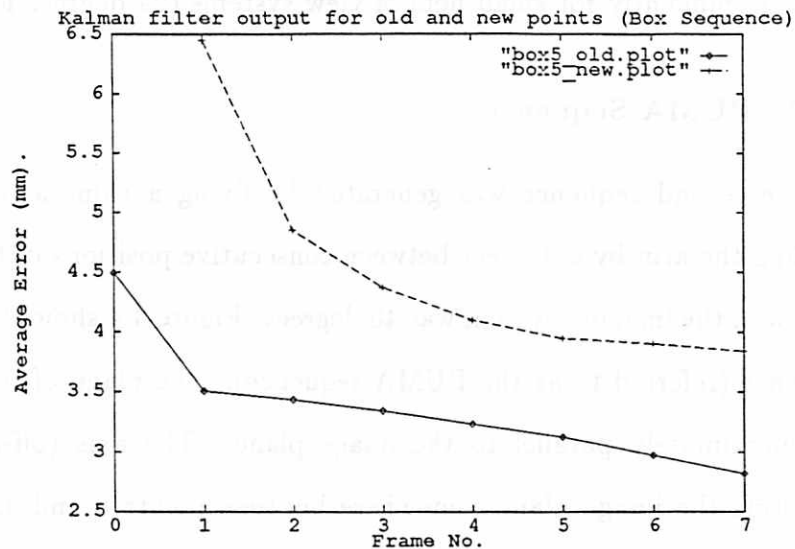


Figure 4.3: Box Sequence. Plot of average error over the frame sequence for for the new points (Model Extension) and for the initial model points (Model Refinement).

In this experiment, the high accuracy with which 3D parameters of the new points were computed is due primarily to the fact that the motion over the sequence is approximately parallel to the image plane. Such motion is best for accurate triangulation. Moreover, due to the rotation about an off-centered axis, image features remain in the image plane for the entire sequence and large image disparities are obtained.

In the first experiment (first row of Table 4.2) described above for the box sequence, the image center was assumed to be at the frame center. In another experiment, the image center was assumed to be displaced by 15 pixels along each axis from the frame center. The experiment was repeated and the 3D locations of the points obtained; comparing these locations to the previously computed locations, we found that the new estimates of the 3D points differed from the previously computed estimates by an average distance of 0.261 mm. This supports the claim made in Chapter 5 that incorrect estimates of the center do not affect the 3D estimation of points significantly for small field of view systems (24 degrees for this sequence).

#### 4.4.2 PUMA Sequence

The second sequence was generated by fixing a camera to a PUMA arm and rotating the arm by 4 degrees between consecutive positions of the camera. The field of view of the imaging system was 40 degrees. Figure 4.4 shows the 14'th frame of this sequence (referred to as the PUMA sequence). The plane of rotation of the camera is approximately parallel to the image plane. The axis (off-centered) of rotation intersects the image plane somewhere between points 8 and 18 in Figure 4.4. The radius of rotation is approximately 2 feet. Thirty frames were taken over a total angular displacement of 116 degrees. The maximum displacement of the camera in these thirty frames is approximately 2 feet along the world y-axis (vertical direction) and 1 feet along the world x-axis (parallel to the x-axis of the image in Figure 4.4).

This corresponds to the longest baseline over these 30 frames. The location of 32 points (marked in Figure 4.4) in a world coordinate system was measured to an accuracy of approximately 0.2 feet along each axis. The depth of the points (in the first frame's coordinate system) used in our experiment varied from 13 feet to 33 feet. Most of the 32 points were tracked over the entire set of 30 frames.

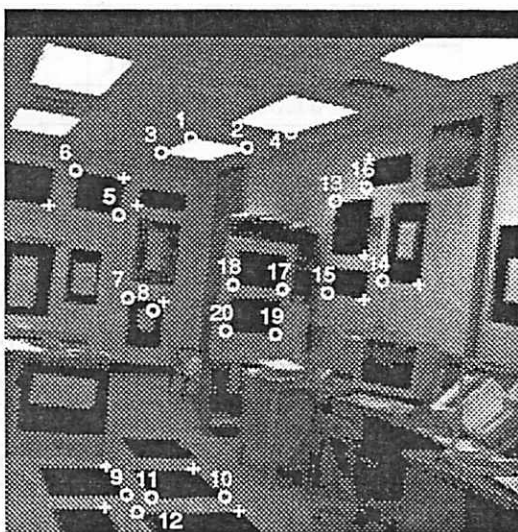


Figure 4.4: Puma Image. The points marked by crosses were used to compute the 3D pose for each frame. Using these poses, the 3D location of the numbered points marked by circles is computed.

The twelve points marked by crosses in Figure 4.4 were used to do pose estimation for each frame. For this experiment, no noise was added to the initial twelve model points. Table 4.3 shows the errors in computing the 3D locations of the remaining 20 points (marked by numbered circles in Figure 4.4). The results shown in Table 4.3 are the output of the algorithm when run in a batch mode using all 30 frames. Figure 4.5 is a graph of the same experiment when run in a sequential mode using a batch size of 2 frames to generate 3D locations. The y-axis in Figure 4.5 is the average error in

Table 4.3: Absolute and Percentage 3D location errors for points in PUMA sequence (see Fig. 5.)

Point Num.	Depth feet	Absolute Error feet	Percentage Error
1	24.59	0.616	2.50 %
2	26.02	0.355	1.36 %
3	28.32	0.373	1.32 %
4	22.06	0.440	1.99 %
5	30.20	0.217	0.72 %
6	28.62	0.281	0.98 %
7	31.56	0.472	1.50 %
8	32.61	0.038	0.12 %
9	14.33	0.125	0.87 %
10	15.34	0.279	1.82 %
11	14.46	0.019	0.13 %
12	13.50	0.081	0.60 %
13	21.75	0.054	0.25 %
14	18.81	0.022	0.12 %
15	21.73	0.036	0.17 %
16	20.28	0.104	0.51 %
17	21.26	0.402	1.89 %
18	20.28	0.731	3.60 %
19	21.55	0.234	1.09 %
20	20.42	0.594	2.91 %

locating the 20 new points and the x-axis is the frame number. Again, the average error is reduced from about 1.5 feet after the first pair of frames to about 0.3 feet at the end of 30 frames.

The point numbers in Table 4.3 correspond to the numbered circled points in Figure 4.4. The depth of each point from the first camera coordinate frame is also shown<sup>5</sup>. The average error for the twenty points was 0.27 feet. The maximum error was 0.731 feet and the minimum error was 0.019 feet. The average percentage error

---

<sup>5</sup>Since the plane of motion was roughly parallel to the image plane, these depths are approximately constant for the entire sequence.



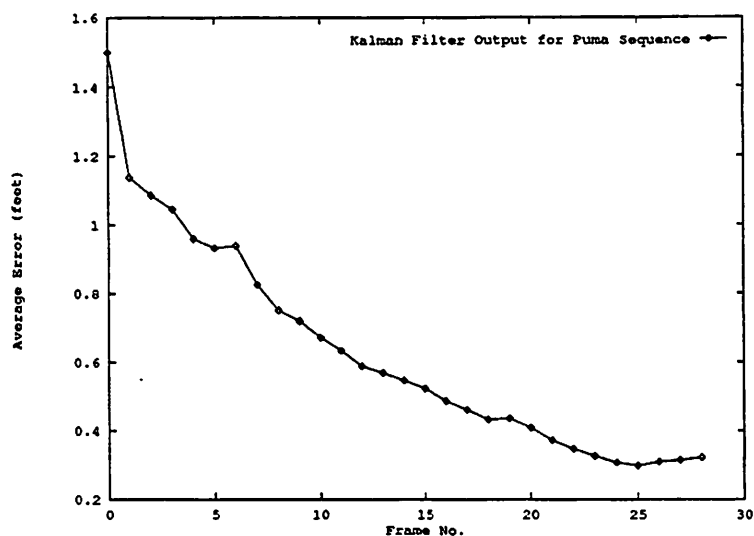


Figure 4.5: Puma Sequence. Plot of average error over the frame sequence for for the new points (Model Extension).

was 1.22 %. The reader must note that this average is just over the set of unknown 20 points. There are points in the sequence for which the error is much larger than 1.2 %. Points 1-4 in Table 4.3 have large errors because they were not localized accurately. The line-finding algorithm was not able to correctly find the borders of the lights. Points 18 and 20 have large errors because they are close to the point where the rotation axis pierces the image plane. These points therefore do not have large disparities. Points 17 and 19, which are a little further away, have correspondingly smaller errors. Finally, as noted above, the imaging system has not been calibrated. Since we used a higher field of view lens for this experiment (40 degrees as compared to 24 degrees for the BOX sequence), the 3D results are more sensitive to errors in locating the image center.

### 4.4.3 A211 Sequence

The A211 sequence (see Figure 4.6) is described in Section 2.5 and Section 4.2.1. There are 10 image frames in the sequence, each taken after a forward motion of the camera by approximately 0.38 feet. Thus the total translation of the camera is 3.42 feet. Objects in the scene ranged from 8 feet to 20 feet away in the first image frame. For this experiment, the initial model was built using Sawhney's [69] algorithm for segmenting and locating shallow structures<sup>6</sup>. The image motion of shallow structures can be described by an affine transform. Based on the affine-trackability of an object, Sawhney [69] is able to segment out different shallow structures in the scene. A Kalman-filter is used to estimate the depths of "shallow structures" over a sequence of multiple image frames.

Seven points (the points marked by crosses in Figure 4.6) lying on shallow structures recovered by this algorithm were used as the initial model points. The 3D model locations were constructed by extending the image projection rays in the first image's coordinate frame of the seven points to the depth computed by Sawhney's algorithm. Thus, the model coordinate frame is the same as the first image's coordinate frame.

The model extension and refinement algorithm was run in a sequential mode. Table 4.4 shows the result of locating the 13 new points (circled and numbered from 8 to 20 in the Figure 4.6) and refining the seven initial model points. The ground truth available for the experiment was only the depths (as opposed to 3D location) of the points in the first image's coordinate frame. Thus the results shown in Table 4.4 compare the measured depth value (ground truth) with the recovered depth value. Column 2 in the table shows the measured depth of the point in the first image coordinate frame. Columns 3 and 4 show the input error and percentage error in

---

<sup>6</sup>Shallow structures are those whose extent in depth is small compared to their average depth from the camera.

Table 4.4: Absolute and Percentage 3D location errors for points in A211 room sequence.

Pt. No.	Depth ft.	INPUT		OUTPUT	
		Abs. Err. ft.	% Err.	Abs. Err. ft.	% Err.
Initial Points					
1	13.4	0.24	1.80 %	0.24	1.78 %
2	14.6	0.19	1.31 %	0.20	1.34 %
3	19.0	0.74	3.88 %	0.66	3.46 %
4	19.0	0.16	0.86 %	0.11	0.60 %
5	20.4	0.13	0.62 %	0.17	0.86 %
6	20.4	0.39	1.90 %	0.32	1.60 %
7	20.4	0.49	2.38 %	0.46	2.25 %
New Points					
8	13.4	-	-	0.11	0.79 %
9	13.4	-	-	0.00	0.01 %
10	14.6	-	-	0.53	3.65 %
11	19.0	-	-	0.73	3.86 %
12	19.0	-	-	0.54	2.82 %
13	19.0	-	-	0.11	0.59 %
14	19.0	-	-	0.07	0.34 %
15	20.4	-	-	0.23	1.13 %
16	20.4	-	-	0.27	1.32 %
17	20.4	-	-	0.12	0.57 %
18	20.4	-	-	0.34	1.65 %
19	20.4	-	-	0.62	3.02 %
20	20.4	-	-	0.59	2.92 %

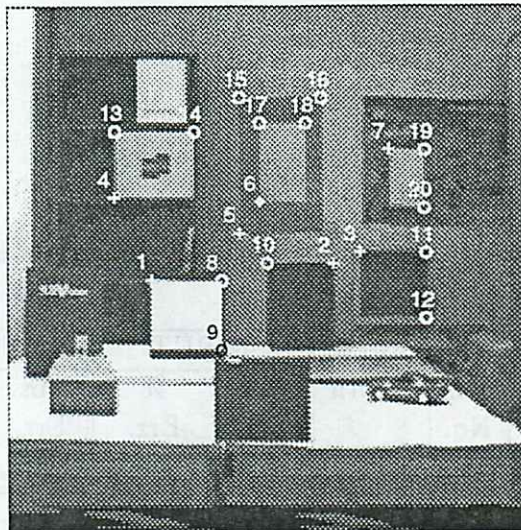


Figure 4.6: A211 Image. The points marked by crosses were used to compute the 3D pose for each frame. Using these poses, the 3D location of the numbered points marked by circles is computed.

depth (before model refinement and extension) respectively. Thus, for the new points (Nos. 8 to 20) these two columns are blank, since no prior estimate is assumed for them. Columns 5 and 6 show the input error and percentage error in depth (after model refinement and extension) respectively. The percentage error in depth is computed with respect to the depth in the first image's coordinate frame.

The average input error in depths of the seven model points was 0.4 feet (1.85 % error). At the end of the ten frames, the average error of the 7 initial points was 0.37 feet (1.76 %). The thirteen new points were located to an average accuracy of 0.4 feet (1.63 %). Thus, in this experiment there was only slight improvement for the model refinement process. The model extension process was, however, fairly accurate in locating new points. If the initial model given to the model extension process is noise free, then the average error in recovering the thirteen new points is 0.2 feet (0.94 %).

#### 4.4.4 COMP Sequence

The COMP sequence was generated by taking images from a camera mounted on a mobile robot. The robot was translated roughly along the optical axis of the camera and 6 image frames were taken after every 1.4 feet (approximately). The total translation of the camera was 7 feet. Figure 4.7 shows the first frame in the image sequence. Objects in the scene ranged from 20 feet to 40 feet away in the first image frame. The depth of some salient structures was measured with a tape measure.

Sawhney's [69] tracking algorithm was applied to the image sequence to identify the shallow structures in the scene. Nine points (the points marked by crosses in Figure 4.7) lying on the recovered shallow structures were used as the initial model points. These points are defined by the intersection of some of the pairs of lines belonging to shallow structures. The 3D model locations were constructed by extending the image projection rays in the first image's coordinate frame of the nine points to the depth computed by Sawhney's algorithm [69]. Thus, the model coordinate frame is the same as the first image's coordinate frame.

The model extension and refinement algorithm was run in a sequential mode. Table 4.5 shows the result of locating the 18 new points (circled and numbered from 10 to 27 in the Figure 4.7) and refining the nine initial model points. The ground truth available for the experiment was only the depths (as opposed to 3D location) of the points in the first image's coordinate frame. Thus the results shown in Table 4.5 compare the measured depth value (ground truth) with the recovered depth value. For this experiment the measured depth values are only approximate to about 0.5 feet for some points. This is especially true for points lying on the left side wall (points 1, 2, 3 etc. in Figure 4.7). Column 2 in the table shows the measured depth of the point in the first image coordinate frame. Columns 3 and 4 show the output error and percentage error in depth for the nine model points as recovered by the affine-

Table 4.5: Absolute and Percentage 3D location errors for points in COMP sequence (see Fig. 4.7.)

Pt. No.	Depth ft.	INPUT		OUTPUT	
		Abs. Err. ft.	% Err.	Abs. Err. ft.	% Err.
Initial Points					
1	29.3	-0.23	0.80 %	-0.11	0.36 %
2	31.3	0.26	0.84 %	0.17	0.55 %
3	34.2	-0.10	0.29 %	-0.07	0.20 %
4	25.7	-0.26	1.03 %	-0.23	0.88 %
5	35.8	1.59	4.43 %	1.54	4.31 %
6	28.7	0.39	1.36 %	0.39	1.35 %
7	43.2	-1.65	3.82 %	-1.63	3.76 %
8	43.2	1.18	2.73 %	1.15	2.66 %
9	28.7	1.46	5.08 %	1.41	4.91 %
New Points					
10	29.3	-	-	0.25	0.86 %
11	29.3	-	-	-0.35	1.19 %
12	31.3	-	-	0.51	1.63 %
13	31.3	-	-	0.28	0.89 %
14	34.2	-	-	0.93	2.70 %
15	34.2	-	-	1.31	3.82 %
16	25.7	-	-	-0.02	0.07 %
17	25.7	-	-	0.03	0.11 %
18	35.8	-	-	1.05	2.93 %
19	35.8	-	-	0.50	1.40 %
20	28.7	-	-	-0.11	0.39 %
21	28.7	-	-	0.08	0.29 %
22	43.2	-	-	0.46	1.07 %
23	43.2	-	-	1.77	4.10 %
24	43.2	-	-	-0.45	1.04 %
25	43.2	-	-	0.13	0.30 %
26	28.7	-	-	0.80	2.77 %
27	28.7	-	-	0.25	0.88 %

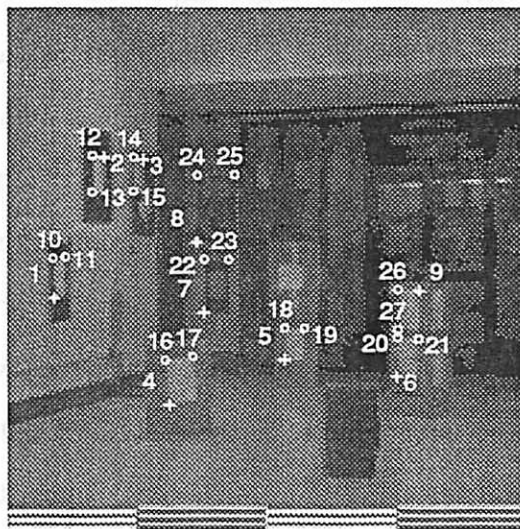


Figure 4.7: COMP Image. The points marked by crosses were used to compute the 3D pose for each frame. Using these poses, the 3D location of the numbered points marked by circles is computed.

based tracking algorithm respectively. Columns 5 and 6 show the output error and percentage error in depth (after model refinement) respectively. For points numbered 10 to 27, it was assumed that no initial model was available, therefore columns 3 and 4 are blank. Note that these points also belong to the reconstructed shallow structures. However, their reconstructed locations were not used as a part of the initial partial model. Instead, these points were used to demonstrate model extension because the ground truth was available only for these structures. Columns 5 and 6 show the output error and percentage error in depth after the model extension process. In the table, the percentage error in depth is computed with respect to the depth in the first image's coordinate frame.

The average input error in depths of the seven initial model points (as recovered by the affine-based tracking algorithm) was 1.01 feet (2.27 % error). At the end of the ten frames, the average error of the 9 initial points was 0.98 feet (2.11 %). In this

experiment also, there was only slight improvement in the initial model as a result of the model refinement process. The model extension process was, however, fairly accurate in locating new points. The eighteen new points were located to an average accuracy of 0.69 feet (1.46 %). Finally, the reader is reminded that for this sequence, the measured depth values for some points are approximate to 0.5 feet.

The robust recovery of the location of new 3D points depends on the camera motion. Optimal angles for triangulation are achieved when there is significant translation parallel to the image plane. In the A211 and COMP sequence, the translation of the camera is mostly along the optical axis. Thus, the FOE (focus of expansion) lies on the image plane. Points close to the FOE have hardly any disparity and their depths cannot be reliably estimated. For this reason, the best results obtained by the model extension and refinement process were for the BOX sequence.

#### 4.5 Conclusions

The techniques presented in this chapter are preliminary efforts for model extension and refinement of point data. The experimental results show that knowledge of a few points can greatly increase the accuracy of 3D recovery in comparison to traditional algorithms from motion and stereo analysis. However, the accuracy of the model extension process depends on the initial accuracy of the model points. To make the system less sensitive to the initial accuracy of the model points, one possible solution would be to couple methods of motion analysis with those of pose recovery.

If the initial model points have a large amount of noise, then the poses determined for any batch of frames will be highly correlated. In this case, the 3D location estimates of new points will be correlated both across all points and also all frames. To fully account for this correlation, assuming  $n$  points and  $f$  frames, ( $n \times f$ ) size



covariance matrices have to be inverted. In our case, it is assumed that the initial points do not have significant noise and hence the cross-correlations can be ignored. But for larger amounts of noise, it may not be possible to ignore these effects. These cross-terms are exactly what Oliensis and Thomas [61] incorporate in their motion analysis paper.

Finally, the terms model extension and refinement are slightly abused in this paper. Model extension and refinement are not limited to just locating new points in the scene. Ultimately, it is desired to build 3D surface and volumetric models and to integrate the new 3D measurements with the existing higher order models; this has been left for future work.

# CHAPTER 5

## SENSITIVITY TO CAMERA PARAMETERS

### 5.1 Introduction

The standard model adopted for imaging 3D scenes by CCD and other cameras is perspective projection. A ray from the camera focal point to a 3D point intersects the image plane at the image location of the 3D point under perspective projection. The optical axis is defined as the perpendicular line from the focal point to the imaging plane and the image center is defined as the point where the optical axis pierces the image plane. Two important camera parameters which often need to be calibrated are the focal length and the image center. In this chapter, we study the effect of errors in estimates of the image center and focal length on pose determination and other related (3D inference from 2D images) problems [44]. The pose determination algorithms used in the experiments are described in Chapter 2 and in [46]. The conclusions drawn, however, are independent of the particular algorithm used.

The image center is often assumed to lie at the center of the image frame. This default center has been reported to be off by as much as 30 pixels for some standard camera and frame grabber combinations [49]. Calibration techniques using either lasers or high precision calibration plates have been used to locate the center to within a few pixels [19, 49, 73]. Is this precise calibration necessary? The analysis presented here shows that it depends on three factors:

1. The particular 3D output or inference one is interested in.

2. The level of accuracy desired in the results.

3. The amount of noise in the input data.

The goal in pose determination is to find the rotation and translation matrices which map the world coordinate system to the camera coordinate system. Given the rotation (or orientation) and translation, the location of the camera with respect to the world coordinate system can be computed. We will show that for *small field of view* imaging systems, an error in the estimation of the camera center does not affect the location of the camera significantly. The rotation or orientation is affected, however, and the amount of error in the orientation is linearly related to the error in the estimate of the center.

An application of the pose determination process is model extension. Given a partial model of the scene, it can be used to obtain robust 3D estimates of new image features, effectively extending the model. From the poses computed using the partial model for two images taken from the same camera, the relative orientation between the two image coordinate frames is computed as a prelude to “induced stereo” analysis<sup>1</sup>. Using the computed relative orientation, the 3D depth and location of points ( in the coordinate frame of one of the cameras) is computed using triangulation. In the third section of this chapter a model of error for this depth is constructed based on the amount of error in locating the image center. The errors predicted by this model are consistent with the errors obtained when applying the pose determination algorithm [46] to both synthetic and real data. We show that these errors are small compared to the errors caused by image noise of 0.5 pixels or more for 512 x 512 images with 24 degrees field of view and approximately 2 feet long stereo baseline; note that the baseline is in the lateral direction. Therefore, image noise is the most significant

---

<sup>1</sup>We use the term “induced stereo” to refer to the process of estimating 3D locations of points from triangulation given the relative orientation between the same camera in two different locations.

factor in determining accurate 3D depths. Furthermore, if the 3D coordinates of the triangulated point are transformed to the world coordinates using the computed pose, the error in 3D location is only due to second order effects, and hence negligible for small field of view systems.

The results derived for induced stereo, showing the effect of errors in locating the image center on the relative orientation between pairs of frames, are also applicable to recovery of structure from motion algorithms<sup>2</sup> [35]. Experiments for motion in depth show that these formulae were able to predict moderately well changes in relative orientation<sup>3</sup> as computed by Horn's algorithm [35]. However, the formulae did not predict the errors well for experiments with motion parallel to the image plane. In the case of induced stereo, the formulae were accurate in their predictions for both kinds of motions. Note that structure from motion algorithms are especially non-robust when the motion is parallel to the image plane [1].

Finally, in the last section of this chapter, the effect of incorrect estimation of the focal length on the pose determination problem is studied. We show that incorrect estimates of the focal length only significantly affects the z-component (i.e. parallel to the optical axis) of the translation. The x and y components of the translation and the rotation are not affected significantly. However, the location of the camera in world coordinates will be affected since the z-component of the translation changes. Again, experimental results on real data are presented to support the theoretical claims.

---

<sup>2</sup>In structure from motion, both relative orientation and depths of image points are unknown.

<sup>3</sup>Due to errors in locating the image center.

## 5.2 Errors in the Pose Determination Problem from Center Offsets

The question asked is this: given two input data sets to the pose refinement problem (the first with the correct image center and the second with an offset image center), how are the two resulting poses related? The only difference between the two data sets is a constant offset of all the image pixels in one data set by the amount the center estimate is offset. Associated with each of the input data sets is a camera coordinate frame. The result of the pose refinement process is to determine the rigid body transformation between the world coordinate frame and the camera coordinate frame. Let “W” represent the world coordinate frame, “C1” the camera coordinate frame with the correct center and “O1” the camera coordinate frame with the offset center; then

$$X_{c1} = R_{c1}(X_w) + T_{c1} \quad (5.1)$$

In this equation, the rotation  $R_{c1}$  and translation  $T_{c1}$  relate a 3D point  $X_{c1}$  in the first camera coordinate frame “C1” to its coordinates  $X_w$  in the world coordinate frame “W”. Points in the camera coordinate frame “O1” are related to points in the world coordinate frame “W” by equation:

$$X_{o1} = R_{o1}(X_w) + T_{o1} \quad (5.2)$$

We would like to find the relationship between the two camera coordinate frames “C1” and “O1”. As noted earlier the only difference between the image data associated with the two frames is a constant shift of all the pixels. Let these be  $\Delta C_x$  and  $\Delta C_y$  in the X and Y image frame directions, respectively; these shifts correspond to the offset of the image center for the second image data set relative to the first image. The displacement of image points between two frames due to rigid motion [2] is given by the following equation:

$$\alpha = \frac{x_1 y \Omega_x}{f} - \left(f + \frac{x x_1}{f}\right) \Omega_y + y \Omega_z + \frac{(f T_x - x T_z)}{Z} \quad (5.3)$$

$$\beta = \left(f + \frac{y y_1}{f}\right) \Omega_x - \frac{y_1 x \Omega_y}{f} - x \Omega_z + \frac{(f T_y - y T_z)}{Z} \quad (5.4)$$

where

$\alpha, \beta$  are the image displacements in the  $x, y$  axis respectively.

$(\Omega_x, \Omega_y, \Omega_z)$  are the small angle approximations to rotation about the  $X, Y$  and  $Z$  axis respectively.

$(T_x, T_y, T_z)$  is the translation along the  $(X, Y, Z)$  axis respectively.

$Z$  is the depth of the point in the first coordinate frame.

$f$  is the focal length of the camera in pixels.

$(x, y)$  is the location of the point in the first image frame ("C1") and  $(x_1, y_1)$  is the location of the point in the second image frame ("O1").

Between the two frames "C1" and "O1",  $\alpha = \Delta C_x$  and  $\beta = \Delta C_y$  i.e. both are constant for all points in the image. What transformation can account for this constant shift? If we assume the field of view of the camera is small, then second order terms such as  $x x_1, x_1 y$  etc. can be neglected. If the scene being imaged is not a frontal plane, i.e. " $Z$ " is not constant for all points<sup>4</sup> then the only transformations that can cause a constant change for a general set of points are the rotations  $\Omega_x$  and  $\Omega_y$  about the  $X$  and  $Y$  axis; everything else (i.e.  $\Omega_z, T_x, T_y$  and  $T_z$ ) will be zero. The following two equations express this relationship:

$$\alpha = \Delta C_x = -f \Omega_y \quad (5.5)$$

$$\beta = \Delta C_y = f \Omega_x \quad (5.6)$$

Let the rotation operator  $\Delta_R$  represent the overall rotation composed of the rotations  $\Omega_x$  and  $\Omega_y$  about the  $X$  and  $Y$  axis. The two coordinate frames "C1" and

---

<sup>4</sup>Frontal planes are dealt with later on.

“O1” are therefore hypothesized to be related by a rotation  $\Delta_R$ :

$$X_{o1} = \Delta_R(X_{c1}) \quad (5.7)$$

Combining equation (5.7) with equation (5.1) we get:

$$X_{o1} = \Delta_R R_{c1}(X_w) + \Delta_R(T_{c1}) \quad (5.8)$$

Comparing equation (5.8) with equation (5.2) we see that:

$$R_{o1} = \Delta_R R_{c1} \quad (5.9)$$

$$T_{o1} = \Delta_R(T_{c1}) \quad (5.10)$$

The above equations reflect how the orientation  $R_{o1}$  and location of the world origin in camera coordinates  $T_{o1}$  are altered with incorrect knowledge of the center. The location of the camera origin in world coordinates  $T_w$  is given by the following equation:

$$T_{wc1} = -R_{c1}^T(T_{c1}) \quad \text{for camera frame } C1. \quad (5.11)$$

$$T_{wo1} = -R_{o1}^T(T_{o1}) \quad \text{for camera frame } O1. \quad (5.12)$$

Using equations (5.9, 5.10) and the above equation for  $T_{wo1}$  we get:

$$T_{wo1} = -R_{c1}^T \Delta_R^T \Delta_R(T_{c1}) = -R_{c1}^T(T_{c1}) = T_{wc1} \quad (5.13)$$

*Therefore an error in estimating the image center significantly affects the location of the camera in world coordinates only if the second order terms in the motion displacement equations (5.3,5.4) are large.* For small field of view imaging systems, the location of the camera is not affected since the second order terms are negligibly small.

The orientation of the robot, on the other hand, is affected; the amount depends on the values of  $(\Delta C_x, \Delta C_y)$ . For instance, for a camera with field of view 24

degrees and a 512 x 512 image, a 30 pixel offset in the camera center in either x or y coordinate would cause a rotation error of 1.427 degrees about the corresponding axis (using equations (5.6,5.6)). Whether changes in orientation of this order are significant or not depends on the application.

Finally, in the case of frontal planes, the depth value "Z" is the same for all points. Therefore in the motion displacement equations (5.3,5.4) both the translation components  $T_x$ ,  $T_y$  and rotation terms  $\Omega_x$ ,  $\Omega_y$  can account for the constant displacement. In this case, the model of change in pose as given in equation (5.7) may not be correct. However, the reader is reminded that frontal planes are typically a degenerate case for pose. Even if we have a correct estimate of center, since "Z" is constant, there could be an incorrect pose related to the correct pose by a transformation composed of translation components  $T_x$ ,  $T_y$  and rotation components  $\Omega_x$ ,  $\Omega_y$ . The image transformations caused by rotation ( $\Omega_x$ ,  $\Omega_y$ ) can be canceled by the transformation due to translation ( $T_x$ ,  $T_y$ ) in equations (5.3,5.4) leading to approximately zero values of  $\alpha$  and  $\beta$  and therefore more than one pose can explain the same input data. The same observation has been made for the structure from motion problem by other researchers [57]. The above model will also break down for large field of view imaging systems (e.g. beyond 45 degrees field of view), when the second order effects cannot be ignored.

### 5.2.1 Experimental Results

In chapter 2, we described algorithms for pose estimation given correspondences for 3D model and 2D image points and lines. We show results from our pose algorithms for two image sets with different errors in the location of the image center. The images (512 x 484 pixels) were acquired using a SONY B/W camera (model AVC-D1) interfaced to a Gould frame grabber. The field of view of the imaging system is



approximately 24.0 degrees. For each set of image data, a new data set was created by adding a constant pixel offset to the x and y coordinates of the image data of the original set.

The first image (Figure 5.1) is of a hallway (similar to the hallway images used in Chapters 2 and 3). The large door in the image is 40 feet distant from the camera. Figure 5.1 shows the first set of input image lines to the pose algorithm. Two more sets of input data were created by adding center offsets of (10,10) and (20,20) pixels respectively to the assumed center. Figure 5.2 shows the projected lines after estimation of pose for the first (original) set of input image data. Figure 5.3 shows the projected lines after estimation of pose for the third set (center offset of (20,20) pixels) of input image data. Note that to display the data in Figure 5.3, the original intensity image was shifted by 20 pixels on each axis (corresponding to the center offset). It is clear from Figure 5.2 and Figure 5.3 that the projections align with their respective input images in a very similar manner. The results for location of the camera in world coordinates for the three different center offsets is given in Table 5.1 under the heading "HALLWAY IMAGE". *The final location (in feet) in world coordinates (for scenes and images as shown in the figure above) changes only by a few tenths of an inch.* In Table 5.1 the (0,0) offset corresponds to the projected model in Figure 5.2 and the (20,20) offset corresponds to the projected model in Figure 5.3.

The second image is from the BOX sequence described in the results section (Section 4.4) of the previous chapter. The image is shown in Figure 4.2. The fifteen points marked by crosses in Figure 4.2 were provided as input to the pose refinement algorithm. Three new image data sets were created by adding center offsets of (10,0), (10,10) and (20,20) respectively. The results of locating the camera for these different data sets are shown in Table 5.1 under the heading "BOX IMAGE". As can be seen from the table, the location of the camera changes by only 1 or 2 mm for different

Table 5.1: Location of camera in world coordinates as computed by the pose refinement algorithm for two sets of real image data with different center offsets.

Center Offset X	Center Offset Y	LOCATION in WORLD		
		$L_x$	$L_y$	$L_z$
HALLWAY IMAGE				
pixels	pixels	feet	feet	feet
Measured	Location	40.00	4.00	3.57
0	0	39.98	4.09	3.57
10	10	40.00	4.09	3.58
20	20	40.02	4.09	3.58
BOX IMAGE				
pixels	pixels	mm	mm	mm
0	0	418.23	260.52	381.37
10	0	417.94	260.49	381.72
10	10	417.27	260.56	380.85
20	20	416.68	260.71	380.51

center offsets. Although results from only two images are presented here, the above behavior has been observed for numerous other images.

### 5.3 Errors in Induced Stereo from Center Offsets

Given two image frames from the same camera at two different positions, the relative orientation between the camera coordinate systems for the two frames can be computed using a pose recovery algorithm. The relationship of the two cameras with respect to the world coordinate system is found and from that the relative orientation is computed. Let the two frames with the correct center be "C1" and "C2"; their relationship to the world coordinate system is:

$$X_{c1} = R_{c1}(X_w) + T_{c1} \quad (5.14)$$

$$X_{c2} = R_{c2}(X_w) + T_{c2} \quad (5.15)$$

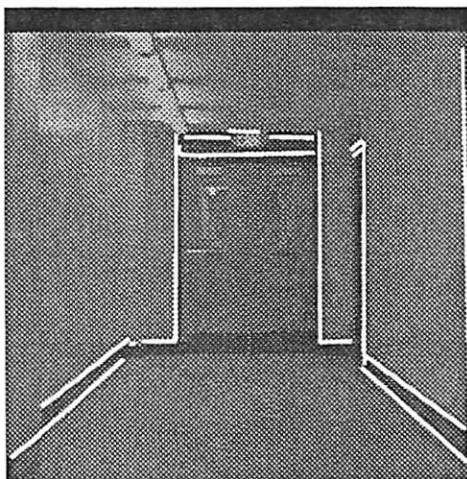


Figure 5.1: Hallway image with input 2D-image lines.

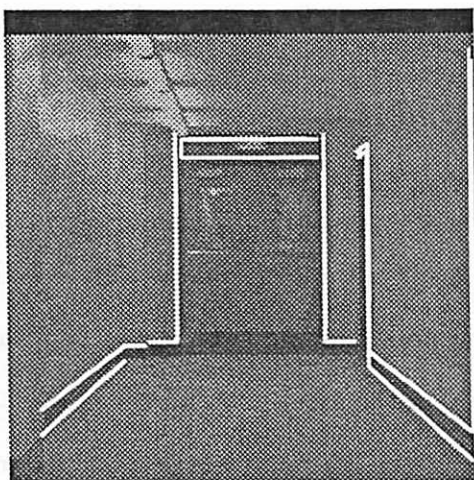


Figure 5.2: Hallway image, projection of model after estimation of pose, image center assumed to be frame center.

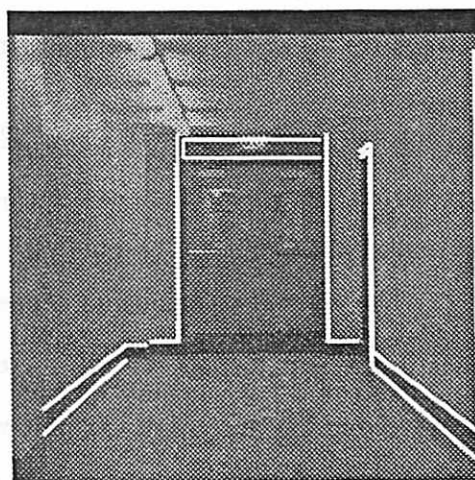


Figure 5.3: Hallway image, projection of model after estimation of pose, image center assumed to be offset from frame center by 20 pixels along each axis.

Combining these equations, the relative orientation between the frames “C1” and “C2” can be expressed by:

$$X_{c1} = R_{c12}(X_{c2}) + T_{c12} \quad (5.16)$$

$$R_{c12} = R_{c1}R_{c2}^T \quad (5.17)$$

$$T_{c12} = (T_{c1} - R_{c1}R_{c2}^T(T_{c2})) \quad (5.18)$$

Similarly, let the frames with the incorrect center be “O1” and “O2”. The relative orientation between these frames is given by:

$$X_{o1} = R_{o12}(X_{o2}) + T_{o12} \quad (5.19)$$

$$R_{o12} = R_{o1}R_{o2}^T \quad (5.20)$$

$$T_{o12} = (T_{o1} - R_{o1}R_{o2}^T(T_{o2})) \quad (5.21)$$

Using equations (5.9,5.10) and some algebraic manipulation, we can rewrite equations (5.20) and (5.21) for  $R_{o12}$  and  $T_{o12}$  in terms of  $R_{c12}$ ,  $T_{c12}$  and  $\Delta_R$ :

$$R_{o12} = \Delta_R R_{c12} \Delta_R^T \quad (5.22)$$

$$T_{o12} = \Delta_R(T_{c12}) \quad (5.23)$$

The above two equations represent the error in the relative orientation if the center estimate is incorrect. If two corresponding points in the two frames “C1” and “C2” are given and assuming the unit vectors<sup>5</sup> from the focal point to them are  $r_{c1}$  and  $r_{c2}$  respectively, the formula for the depth  $D_c$  of the 3D point obtained by triangulation in the first camera coordinate frame is:

$$D_c = s(r_{c1} \cdot \hat{z}) \quad (5.24)$$

$$s = \frac{(r_{c1} \times R_{c12}(r_{c2})) \cdot (T_{c12} \times R_{c12}(r_{c2}))}{\|(r_{c1} \times R_{c12}(r_{c2}))\|^2} \quad (5.25)$$

---

<sup>5</sup>We define the rays corresponding to these vectors as projection rays.

In this equation  $s$  is the length along the 3D ray corresponding to the vector  $r_{c1}$  from the origin of the 3D point and  $\hat{z}$  is the unit vector along the z-axis.

For the frames "O1" and "O2", the unit vectors for the same points  $r_{o1}$  and  $r_{o2}$  corresponding to points  $r_{c1}$  and  $r_{c2}$  in frames "C1" and "C2" can be approximated as before<sup>6</sup> by:

$$r_{o1} = \Delta_R(r_{c1}) \quad (5.26)$$

$$r_{o2} = \Delta_R(r_{c2}) \quad (5.27)$$

Combining these two equations and equation (5.24), the depth of the same 3D point in the image frame "O1" coordinate system is:

$$D_o = s(r_{o1} \cdot \hat{z}) = s(\Delta_R(r_{c1}) \cdot \hat{z}) \quad (5.28)$$

Note that the length along the 3D ray has not changed in the offset center case, i.e. frame "O1-O2" as compared to the correct center pair "C1-C2". However, the depth of the point changes because of the rotation of the unit vector  $r_{c1}$ . The X and Y coordinates of the 3D point are also similarly affected.

From the above derivation the percentage error in depth can be predicted by the following formula:

$$\%D_{err} = \frac{-\Delta C_y r_{c1x} + \Delta C_x r_{c1y}}{f r_{c1z}} 100.0\% \quad (5.29)$$

In the next section, this error is used to predict the percentage depth error due to incorrect center; and the prediction is compared with the actual depth errors found when running pose and the triangulation algorithm on synthetic data. The errors in depth computation due to an incorrect estimate of center and noise in the image locations versus data with a correct estimate of center but no noise being present are also compared. As results will show, the error for even small amounts of image noise are much larger than error due to incorrect center placement.

---

<sup>6</sup>The approximation ignores the second order terms for small field of view systems.

Note that if the triangulated 3D point is transformed to world coordinates, then the only error will be due to second order effects. The center offset causes the 3D point to be rotated by  $\Delta_R$  in the camera coordinate system and the subsequent transformation back to world coordinates cancels out the  $\Delta_R$  rotation.

### 5.3.1 Experimental Results for Induced Stereo

Experimental results are presented in this section for synthetic data and real image data. A pair of images is required for each experiment in this section. Synthetic data was created by taking a model of a 3D scene very similar to the hallway shown in Figure 5.1 and projecting a set of 3D points onto the image plane for two different positions of the camera (induced stereo baseline in lateral direction was approximately 2.0 feet). Twenty points in total were used, out of which only 9 were used for pose calculation. Depth computations were done for all twenty points. The imaging frame was assumed to be 512 x 512 with a field of view equal to 24 degrees.

The box image shown in Figure 4.2 was the first frame used for the real image data (see description of BOX sequence in Section 4.4). The 8'th frame of the BOX sequence was used as the second image. It corresponds to a rotation of the box by approximately 25 degrees about its central vertical axis. The fifteen points marked by crosses in Figure 4.2 were used to compute the pose for each frame. Depths were computed for the fifteen points marked by circles in Figure 4.2.

In Table 5.2, the predicted average depth errors are compared with the computed average depth errors for various different center offsets for both the synthetic data and the box data. As can be seen, the predicted depth errors compare quite favorably to the computed ones. The very small difference between the predicted and computed errors can be attributed to the second order effects which were ignored.

For the comparison of error due to incorrect center versus error due to noisy image

Table 5.2: Predicted percentage average depth errors versus computed average depth errors for different center offsets for synthetic and real image data.

Center Offset X pixels	Center Offset Y pixels	Predicted % depth error	Computed % depth error
<b>SYNTHETIC IMAGE</b>			
10	0	0.063	0.047
10	10	0.085	0.091
20	20	0.169	0.183
30	30	0.254	0.277
50	50	0.423	0.469
<b>BOX IMAGE</b>			
10	0	0.082	0.078
10	10	0.141	0.172
20	20	0.283	0.337
30	30	0.424	0.495
50	50	0.707	0.789

Table 5.3: Computed average depth errors for synthetic uniform noise data with and without center offsets. 512 x 512 image with 24 deg field of view, center offset by 30 pixels for each axis, 2 feet long stereo baseline.

Image Noise pixels	Noise only % Depth Err	Center Offset plus Noise % Depth Error
0.0	0.000	0.277
0.1	0.124	0.300
0.2	0.247	0.350
0.5	0.623	0.661
1.0	1.366	1.432
1.5	1.453	1.424
2.0	2.133	2.240
3.0	3.398	3.418
5.0	5.653	5.676
10.0	11.638	11.765

locations, various amounts of uniform pixel noise were added to the synthetic image data. The center was offset by a constant amount of 30 pixels along each axis for this experiment. From Table 5.3, it can be seen that at noise levels greater than 0.5 pixels, the error with and without center offset are comparable. *It is only at image noise levels of less than 0.5 pixels that the error due to incorrect center is significant.* Of course, if we increase the induced stereo baseline and are able to make more accurate 3D measurements from induced stereo, then the 3D error caused by incorrect center estimates will become comparable to 3D errors at larger levels of image noise. To conclude, given a particular stereo configuration and expectation of image noise, we can calculate the significance of 3D error due to an error in estimating the center.

### 5.3.2 Structure from Motion

The equations which show the effect of errors in locating the image center on the relative orientation between pairs of frames, derived in the case of induced stereo, are also applicable to recovery of structure from motion algorithms. The error function  $E_h$  minimized by Horn [35] in his relative orientation algorithm, given point correspondences for a pair of frames, is:

$$E_h = \sum_{i=1}^n ((r_{c1i} \times R_{c12}(r_{c2i})) \cdot T_{c12})^2 \quad (5.30)$$

where

$r_{c1i}$ ,  $r_{c2i}$  are the vector representations of the projection rays of corresponding points.

$R_{c12}$  and  $T_{c12}$  are the relative orientation parameters: rotation and translation respectively.

If two new frames are created by shifting the original image data by an offset



corresponding to the error in locating the center, the error function  $E_h^o$  minimized is:

$$E_h^o = \sum_{i=1}^n ((r_{o1i} \times R_{o12}(r_{o2i})) \cdot T_{o12})^2 \quad (5.31)$$

Substituting in equation (5.31) for the new projection rays  $(r_{o1i}, r_{o2i})$  using equations (5.26) and (5.27) and for the rotation  $R_{o12}$  and translation  $T_{o12}$  using equations (5.22) and (5.23) respectively, we can show that:

$$E_h^o = E_h \quad (5.32)$$

Therefore, if  $E_h$  is minimum for the rotation  $R_{c12}$  and translation  $T_{c12}$ , then  $E_h^o$  is minimum for the rotation  $R_{o12}$  and translation  $T_{o12}$  which are related to  $(R_{c12}, T_{c12})$  by equations (5.22) and (5.23). The change in relative orientation caused by errors in estimating the center are predicted by equations (5.22) and (5.23).

Experimentally, these formulae were able to predict moderately well the changes in relative orientation parameters due to errors in locating the image center<sup>7</sup> for camera motions having a significant translation component in the  $T_z$  (i.e. along the optical axis) direction. Less success was achieved for predicting errors for camera motions parallel to the image plane. Note that structure from motion algorithms are especially non-robust when the motion is chiefly parallel to the image plane.

#### 5.4 Inaccurate Estimates of the Focal Length

The focal length of the lens supplied by lens manufacturers are generally quite accurate. However, when the lens is focused on points close to the camera (i.e. when the camera is not focused at infinity) the *effective* focal length of the system must be established by a calibration procedure [73]. In this section, the effects of

---

<sup>7</sup>Relative Orientation parameters were computed by Horn's algorithm [35]

incorrect estimates of the focal length on the output of the pose refinement process are examined.

The image projection  $(x, y)$  of a world point  $X_w$  given an estimate of translation  $T_c$  and rotation  $R_c$  is:

$$x = f \frac{(R_c(X_w) + T_c)_x}{(R_c(X_w) + T_c)_z} \quad (5.33)$$

$$y = f \frac{(R_c(X_w) + T_c)_y}{(R_c(X_w) + T_c)_z} \quad (5.34)$$

Dividing these two equations, we obtain:

$$\frac{x}{y} = \frac{(R_c(X_w) + T_c)_x}{(R_c(X_w) + T_c)_y} \quad (5.35)$$

The rotation operator can be represented as a (3x3) matrix:

$$R = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \end{bmatrix} \quad (5.36)$$

where  $s_i$ ,  $i = 1, 2, 3$  are the vectors corresponding to the rows of the rotation matrix  $R_c$ . Substituting (5.36) into (5.35), equation (5.35) can be rewritten as:

$$\frac{x}{y} = \frac{(s_1 \cdot X_w + T_{cx})}{(s_2 \cdot X_w + T_{cy})} \quad (5.37)$$

This is a linear equation in the pose parameters  $s_1, s_2, T_{cx}$  and  $T_{cy}$  which can be rewritten as:

$$x(s_2 \cdot X_w + T_{cy}) - y(s_1 \cdot X_w + T_{cx}) = 0.0 \quad (5.38)$$

One such equation is obtained for each world/ image point correspondence. Given 5 or more point correspondences, we can therefore solve the system of equations and get estimates of the parameters  $s_1, s_2, T_{cx}$  and  $T_{cy}$ . The rotation parameters  $s_1, s_2$ , however, have quadratic constraints and therefore the system of equations must be solved by non-linear techniques. Tsai [73] uses the same system of equations in his camera calibration algorithm. Since the rotation matrix is an orthonormal matrix,

estimates of its first two rows  $s_1$  and  $s_2$  can be used to obtain the third row  $s_3$ , using the symmetric and other orthonormal properties of the matrix. The only pose refinement parameter not determined by the system of equations is therefore the translation along the optical axis  $T_z$ .

Our goal in this section is to examine the effect of incorrect estimates of the focal length on the output of a pose refinement algorithm. Equations (5.37) and (5.38) do not depend on the focal length and consequently an incorrect estimate of the focal length would not affect the solution of these equations. Therefore, an incorrect estimate of focal length should affect only the  $T_z$  parameter of the pose; all other parameters should not change since their estimation does not depend on knowledge of the focal length.

In practice [46] we may minimize other error functions to do pose refinement. Based on the above analysis we hypothesize that an incorrect estimate of the focal length would only significantly affect the  $T_z$  component of the pose parameters for other pose refinement methods as well<sup>8</sup>. This hypothesis has been supported by experiments using both synthetic and real data and the pose refinement algorithms described in Chapter 2 and in [46]. Results of some of these experiments are shown in Table 5.4.

The experiments were performed using the synthetic, hallway and box image data sets described earlier. In each case, the pose refinement algorithm was applied using both the correct focal length and incorrect estimates of the focal length. The incorrect estimates of the focal length were obtained by multiplying the correct focal length by a scale. Thus, in Table 5.4, entries in rows with focal length scale 1.0 correspond to experiments with the correct focal length and entries with rows corresponding to scale not equal to 1.0 correspond to experiments with incorrect focal lengths. Both

---

<sup>8</sup>That is, methods where the pose is not estimated by solving the system of equations defined by (5.38) but by some other system of equations.

Table 5.4: Rotation and translation as computed by the pose refinement algorithm for the same sets of images with different focal lengths.

FOCAL LENGTH SCALE	TRANSLATION			ROTATION			
	$T_x$	$T_y$	$T_z$	ANGLE	AXIS		
				deg.	$A_x$	$A_y$	$A_z$
SYNTHETIC DATA							
1.000	4.004	-3.994	60.011	120.015	-0.577	0.577	0.577
0.928	4.013	-4.002	58.048	120.016	-0.577	0.577	0.578
HALLWAY IMAGE							
1.000	4.055	-3.942	39.926	119.808	-0.576	0.574	0.582
0.928	4.023	-3.962	37.382	119.344	-0.579	0.574	0.580
1.045	4.072	-3.932	41.509	120.066	-0.575	0.574	0.583
BOX IMAGE							
1.000	-8.256	74.647	620.313	132.833	-0.178	0.952	-0.250
1.100	-8.344	74.801	684.354	132.627	-0.177	0.952	-0.249

the translation and rotation results of the pose are shown in Table 5.4. The rotation is shown by its angle-axis representation. The axis vector is a unit vector. As can be seen from Table 5.4, the only large change in any of the pose parameters for any of the experiments is in the  $T_z$  component of the translation.

Although poses can be obtained whose projection fits the original image data fairly well in the case of incorrect estimates of the image center, this is not the case for incorrect estimates of focal length. Changing the focal length causes the projection of 3D points to be dilated or contracted by a constant amount while changing the  $T_z$  component of the translation causes the image projections to dilate or contract based on their depth from the camera. As we have seen, however, the minimum of the pose error functions, given incorrect estimates of focal length, leads only to a significant change in  $T_z$ . This property of poor fits makes it comparatively easier to calibrate imaging systems for focal length as compared to calibrations for the image center.

## CHAPTER 6

### CONCLUSIONS

The goal of this thesis was to develop algorithms to infer 3D information from a sequence of 2D images. Algorithms were developed to *robustly* determine the 3D location and orientation of a robot (*pose determination*) and to locate new points in a 3D world coordinate system (*model acquisition*). A key assumption made was that the knowledge of a partial 3D model would greatly benefit the accuracy with which the robot, and new points or landmarks in the 3D world, could be located. We believe this assumption was borne out by the experimental results presented earlier. The robot was located with an average accuracy of 0.3 feet in the indoor environment and new points were located with an average accuracy of 1.7 %. These results are far superior to the accuracy obtained by traditional structure from motion techniques. In this chapter, the main contributions of this thesis to both the pose determination problem and the model acquisition problem are summarized. Also discussed are related unsolved problems and possible avenues for future research.

#### 6.1 Pose Determination

A sequence of algorithms were presented in this thesis for pose determination using both point and line correspondences. Each successive algorithm was able to perform more robustly than the previous algorithms over a wider range of input noise scenarios. First, existing least-squares techniques for pose determination from point

and line tokens were improved upon. Least-squares techniques which minimize rotation and translation simultaneously (“R\_and\_T”) were developed and shown to be far superior to the earlier techniques which solved for rotation first and then translation (“R\_then\_T”). In the case of line token data, it was assumed that end-points of image lines cannot be reliably extracted. Algorithm “R\_and\_T\_img” minimized the error in aligning model line segments with corresponding infinitely extended image lines. In contrast, algorithm “R\_and\_T\_mod” minimized the error in aligning image line segments with corresponding infinitely extended model lines. We showed experimentally that algorithm “R\_and\_T\_mod” performed more robustly than algorithm “R\_and\_T\_img” when there was significant fragmentation in the line data.

However, the least-squares algorithms failed catastrophically when outliers were present in the match data. It was noted that outliers in the pose determination problem arise frequently due to either incorrect correspondences or gross errors in the 3D model. Two sets of robust algorithms for pose determination were developed to handle data contaminated by outliers. The first algorithm “Tuk\_wts” minimized a non-convex objective function in which the effect of outliers was bounded by saturating the objective function after a threshold error value. The algorithm converged to the correct answer within ten iterations for many of our experiments. However, the “Tuk\_wts” algorithm was susceptible to initial estimates and could settle into local-minima. This is due to the local optimization technique used by the “Tuk\_wts” algorithm to minimize Tukey’s error function.

A useful task for future work would be to develop a global optimization method for minimizing Tukey’s error function. The optimization method must be computationally efficient and thus techniques based on simulated annealing would not be useful. Two possible directions to investigate are Blake and Zisserman’s “Graduated Non-Convexity Algorithm” [10] and the continuation methods used by Leclerc [48], both

developed for optimizing similar error functions but for different problem domains.

The second set of algorithms “Med\_R\_and\_T\_img” and “Med\_R\_and\_T\_mod” developed here were not as sensitive to initial estimates. These minimize the Least Median Square (LMS) of the residual error functions across all landmarks. They are capable of performing correctly in situations where the number of outliers is less than 50% of the number of data points. The average time complexity of the LMS-based algorithms was substantially reduced using random subset selection methods.

The robust pose determinations algorithms were formulated by extending the least-squares algorithms. Thus, the computational performance of the least-squares algorithms is an important issue. All the least-squares algorithms developed here used the Gauss-Newton technique for minimizing non-linear objective functions. In most of our experiments this technique required only a few iterations (typically 4 or 5) to converge to the correct answer for reasonable initial estimates. It was also easy to modify this technique to incorporate prior uncertainties of the initial pose estimate using covariance matrices. Finally, for applications where no initial estimate is available, a composite algorithm using both the “R\_then\_T” and the “R\_and\_T” algorithms was developed. The initial estimates were obtained from a uniform sampling of the rotation space. Experimentally, it was found that 12 initial samples of the rotation space sufficed to find the optimal estimate.

To determine the optimal pose parameters it is important to accurately model the image measurement noise. The pose algorithms presented in this thesis can be improved by more accurate modeling of this noise. The least-squares algorithms presented earlier are optimal when the measurement noise is gaussian. The robust algorithms also assume that the non-outlier data are corrupted with gaussian noise. In the experiments, the standard deviation of the measurement noise was assumed to be same for all data elements, although this assumption may not be valid everywhere

in the image. For instance, in the outdoor images (see Figure 2.10), the measurement noise in locating the walkway line is considerably higher than the measurement noise in locating the roof line of the building. Therefore, it is important to model the feature extraction (e.g. line-extraction) process and compute estimates of the noise parameters. This is not a trivial task and it depends on various factors, such as image sensor noise, intensity structures neighboring the image feature, the particular line extraction algorithm used, etc.. Deriche[14], for instance, computes the noise in locating image lines from the noise in locating the component edge elements. He assumes the location of edges in the image to be corrupted by zero-mean gaussian noise. Using the intensity contrast across the edges, the second order statistics of the noise is calculated. His scheme however, assumes the noise across neighboring image edges to be independent; in addition no influence from neighboring image lines or other intensity structures is taken into account. It remains to be demonstrated whether such assumptions can be reasonably made.

## 6.2 Model Acquisition

Model acquisition is an important and rich problem in computer vision. The approach adopted in this thesis was one of model extension and refinement. A partial model of the environment is assumed to exist and this model is extended and refined over a sequence of frames. New features are located by triangulation using the displacement of image tokens and the poses of the object computed from model-image feature correspondences for the sequence of image frames. Since the partial model may be noisy, the least-squares pose determination techniques were extended to optimize for errors in both the image and model data. The estimation of the 3D points was done using both batch and sequential Kalman filter based methods. An



advantage of using pose recovery was that the world coordinate system provided a stable and independent coordinate system in which integration of data from multiple image frames was done. New 3D points were located in four real data sequences with average errors less than 1.7%. These results as noted earlier are far superior to those obtained by the traditional structure from motion techniques employed in computer vision.

The experimental results showed that knowledge of a few points can greatly increase the accuracy of 3D structure recovery in comparison to traditional algorithms from motion and stereo analysis. However, the accuracy of the model extension process depends on the initial accuracy of the model points. To make the system less sensitive to the initial accuracy of the model points, one possible solution is to couple methods of motion analysis with those of pose recovery. In the approach adopted in this thesis, new points are located by triangulation and the relative orientation between consecutive image frame pairs is determined from the computed poses. However, the relative orientation can also be computed from the image correspondences of unmodeled points. This information was not exploited here and could potentially be used to develop algorithms minimizing composite objective functions having both pose determination and motion analysis error terms. The problem lies in constructing optimal objective functions which are not too complex for efficient minimization.

If the initial model points have a large amount of noise, then the poses determined for any batch of frames will be highly correlated. In this case, the 3D location estimates of new points will be correlated both across all points and also all frames. To fully account for this correlation, covariance matrices equal in size to the number of points times number of frames will have to be inverted. In our case, it is assumed that the initial points do not have large noise and hence the cross-correlations can be ignored. But for larger amounts of noise, it may not be possible to ignore these

effects. These cross-terms are exactly what Oliensis and Thomas [61] incorporate in their motion analysis paper and the same could be done for the model extension and refinement algorithms presented here.

The noise in the 3D model was assumed to be gaussian. However, this assumption depends significantly on how the initial model was constructed. For instance, in some scenarios, the gaussian assumption is met for parts of the model but the various parts may be skewed with respect to each other. Thus the techniques for model extension and refinement need to be broadened to handle various kinds of model noise.

The model extension and refinement algorithms presented in this thesis were for point tokens. The point tokens were corner points obtained from intersection of image lines. However, some of the image corner points do not correspond to actual 3D points. Thus techniques to detect these invalid corner points must be developed. A method suggested by Harris and Pike [32] is to examine the estimated covariances of the tracked corner points. The uncertainty ellipsoids of the valid corner points are aligned while those of the invalid corner points are skewed. This is related to the notion by Oliensis and Thomas [61] that the errors in the 3D reconstruction of points are correlated. This problem could also be avoided if 3D lines and curves were reconstructed instead of 3D points.

The terms model extension and refinement are slightly abused in this paper. Model extension and refinement are not limited to just locating new tokens in the scene. Ultimately, it is desired to build 3D surface and volumetric models and integrate the new 3D measurements with the existing higher order models. For distant objects moving closer to the mobile platform, the surface model must evolve over time as the detail present in the object emerges.

### 6.3 Sensitivity Analysis

Finally, the sensitivity of pose determination and model extension to incorrect estimates of camera parameters (focal length and image center) was analyzed. It was theoretically claimed and demonstrated experimentally that for small field of view systems, offsets in the image center do not significantly affect the location of the camera and the location of new 3D points in a world coordinate system. Errors in the focal length significantly affect only the component of translation along the optical axis in the pose computation. Future work in this direction would be to analyze the sensitivity of the pose determination and model extension process to lens distortion and other large field of view effects.

## APPENDIX A

### DIFFERENT WAYS OF REPRESENTING 3D ROTATION

The rotation operator in equation (2.31) can be expressed in many ways [41, 71]. Each gives rise to different non-linear expressions for “E”. Some ways would require non linear equality constraints to be satisfied. Some are unconstrained but give rise to complex objective functions, which must be minimized. A minimum of 3 parameters is needed to represent rotation, but this leads to non-unique representations. It has been proven that to represent 3D rotation uniquely at least 6 parameters are needed [71]. We now describe some of the common representations and build the motivation for our final choice of representation.

**Orthonormal Matrix :** This is one of the most common ways of representing 3D rotation. Here rotation is specified by a  $3 \times 3$  orthonormal matrix:

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \quad (\text{A.1})$$

To be an orthonormal matrix, the following equality constraints must be satisfied.

$$\sum_{j=1}^3 r_{ij}^2 = 1.0 \quad (\text{A.2})$$

$$\sum_{j=1}^3 r_{ij}r_{kj} = 0.0 \quad \text{where } i \neq k. \quad (\text{A.3})$$

$$\det(R) = 1.0 \quad (\text{A.4})$$

Note that for any matrix "R" which satisfies the first two constraints, its determinant will be either +1.0 or -1.0. The determinant of "R" must be +1.0 to represent a rotation operator. When the determinant of "R" is -1.0 it represents a reflection. In our solution process, the determinant constraint need not be explicitly observed. If we obtain a solution where the determinant of the "R" matrix is -1.0, all the elements of "R" and "T" in equation (2.31) can be multiplied by -1.0 to obtain a solution in which "R" represents a rotation. Thus, if "R" is represented as an orthonormal matrix, the objective function given by equation (2.31) becomes a quadratic function of 12 unknowns with 6 quadratic equality constraints to be satisfied. The rotation matrix gives us 9 of the 12 unknowns; the translation vector gives the remaining 3.

**Axis and Angle :** The rotation of a vector "p" by angle " $\theta$ " about axis " $\omega$ " is given by Rodrigue's formula :

$$p' = (\cos\theta)p + \sin\theta(\omega \times p) + (1 - \cos\theta)(\omega \cdot p)\omega \quad (\text{A.5})$$

Here, " $\omega$ " must be a unit vector. So, a quadratic equality constraint must be satisfied. Therefore, if "R" is represented by axis and angle, an objective function given by equation (2.31) is obtained, which contains trigonometric terms of power 2 and higher and a quadratic equality constraint must be satisfied.

There are two variations of the above method for representing rotations. As described above, we require 4 terms to represent the rotation. Both of the following variations require only 3. First, the axis could be represented by two spherical coordinates, in which case, a high order trigonometric objective function is obtained with no constraints. The three terms would be the two spherical coordinate angles representing the axis and the angle of rotation about the axis. Second, the constraint requiring the axis vector to be a unit vector could be

removed. The magnitude of the axis vector would then be the angle of rotation about the axis. This is also an unconstrained representation of rotation. For large rotations it leads to a complicated trigonometric objective function. However, for small rotations, it leads to a unconstrained linear operator and is therefore used by us to represent small rotations. Note, in the axis angle representation, negating both the axis and the angle represents the same rotation.

**Euler angles :** Here, 3D rotation is represented by 3 successive rotations about each of the 3 coordinate axis, respectively. The order of the rotations is: first rotate by angle  $\psi$  about the z-axis, then by angle  $\phi$  about the y-axis and finally by angle  $\theta$  about the x-axis. Each of these rotations can be specified by a  $3 \times 3$  matrix, whose elements are trigonometric functions of the angle of rotation. The final rotation matrix is given by the following equation:

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} \cos(\phi) & 0 & -\sin(\phi) \\ 0 & 1 & 0 \\ \sin(\phi) & 0 & \cos(\phi) \end{bmatrix} \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (\text{A.6})$$

Representing rotations by Euler angles results in an unconstrained minimization problem, although the objective function becomes an extremely complicated trigonometric function. Euler angles are often used in engineering applications as they can be measured by instruments and also are easy to picture.

**Quaternions :** Here rotations are represented by a 4D vector  $q$ . The quaternion "q" can be thought of as a scalar  $q_o$  and a vector  $q_v$ . It is related to the axis  $\omega$  and angle  $\theta$  representation in the following manner:

$$q_o = \cos(\theta/2) \quad q_v = \sin(\theta/2)\omega \quad (\text{A.7})$$

Rotation of a vector  $p$  using quaternions is given by  $p' = q \circ p \circ q^*$ , where "o" denotes quaternion multiplication.

The complex conjugate of  $q$  is  $q^* = (q_o, -q_v)$ . Quaternion multiplication of two quaternions is given by the following equations:

$$\begin{aligned} (p \circ q)_o &= (p_o q_o - p_{v1} q_{v1} - p_{v2} q_{v2} - p_{v3} q_{v3}) \\ (p \circ q)_{v1} &= (p_o q_{v1} + p_{v1} q_o + p_{v2} q_{v3} - p_{v3} q_{v2}) \\ (p \circ q)_{v2} &= (p_o q_{v2} - p_{v1} q_{v3} + p_{v2} q_o + p_{v3} q_{v1}) \\ (p \circ q)_{v3} &= (p_o q_{v3} + p_{v1} q_{v2} - p_{v2} q_{v1} + p_{v3} q_o) \end{aligned}$$

$(q_o, q_v)$  and  $(-q_o, -q_v)$  represent the same rotation, while  $(q_o, q_v)$  and  $(q_o, -q_v)$  represent opposite rotations. For a 4-tuple to represent rotation it must be a unit vector, i.e., the following constraint must be satisfied:

$$q_o^2 + q_{v1}^2 + q_{v2}^2 + q_{v3}^2 = 1.0 \quad (\text{A.8})$$

Representing rotations by quaternions, the objective function becomes a fourth degree polynomial. The solution also has to satisfy a quadratic equality constraint. With quaternions [34] however, it is easy to compose rotations. Another advantage of using quaternions is that given a 4D vector which does not satisfy the unit magnitude constraint, it is easy to find the closest unit quaternion to it. This is useful for us. In our formulation, large rotations are represented by quaternions. At each iteration in the search for the minimum of the objective function, small rotations are represented by a 3D vector for rotation axis and angle. Quaternions are formed from these 3D vectors and composed with the current estimates for the rotation. This results in an unconstrained minimization problem.

## APPENDIX B

### UNIFORM SAMPLING OF THE ROTATION

Horn [35] presents a uniform way of sampling the rotation space. His sampling is based on the rotation groups of regular polyhedra. Two sets of samplings of the rotation space in quaternion form are presented below (derived from [35]). The components of the unit quaternions may take on the values 0 and 1, as well as the following:

$$b = \frac{1}{2} \quad c = \frac{1}{\sqrt{2}} \quad (\text{B.1})$$

In the first set, there are 12 unit quaternion samples and they form the twelve elements of the rotation group of the tetrahedron:

$$\begin{array}{cccc} (1, & 0, & 0, & 0) & (0, & 1, & 0, & 0) & (0, & 0, & 1, & 0) & (0, & 0, & 0, & 1) \\ (b, & b, & b, & b) & (b, & b, & b, & -b) & (b, & b, & -b, & b) & (b, & b, & -b, & -b) \\ (b, & -b, & b, & b) & (b, & -b, & b, & -b) & (b, & -b, & -b, & b) & (b, & -b, & -b, & -b) \end{array}$$

In the second set, there are 24 unit quaternion samples and they form the twenty four elements of the rotation group of the octahedron and the hexahedron(cube):

$$\begin{array}{cccc} (1, & 0, & 0, & 0) & (0, & 1, & 0, & 0) & (0, & 0, & 1, & 0) & (0, & 0, & 0, & 1) \\ (0, & 0, & c, & c) & (0, & 0, & c, & -c) & (0, & c, & 0, & c) & (0, & c, & 0, & -c) \\ (0, & c, & c, & 0) & (0, & c, & -c, & 0) & (c, & 0, & 0, & c) & (c, & 0, & 0, & -c) \\ (c, & 0, & c, & 0) & (c, & 0, & -c, & 0) & (c, & c, & 0, & 0) & (c, & -c, & 0, & 0) \\ (b, & b, & b, & b) & (b, & b, & b, & -b) & (b, & b, & -b, & b) & (b, & b, & -b, & -b) \\ (b, & -b, & b, & b) & (b, & -b, & b, & -b) & (b, & -b, & -b, & b) & (b, & -b, & -b, & -b) \end{array}$$

Horn [35] also gives the 64 rotation samples for the rotation groups of the icosahedron and the dodecahedron.



## A P P E N D I X   C

### L I N E A R   S Y S T E M   T H E O R Y

Some facts from linear system estimation theory are reviewed [70]. An unknown parameter vector  $\vec{x}$  with “p” elements is related to a set of “n” noisy observations  $\vec{y}$  by the following equation:

$$A\vec{x} = \vec{y} + \vec{\eta} \tag{C.1}$$

where  $\vec{\eta}$  is zero-mean Gaussian noise with covariance matrix  $V$ . Assume that this set of equations is an over-constrained system. Then the Best Linear Unbiased Estimate (BLUE) of the unknown vector  $\vec{x}$  is given by:

$$\hat{x} = (A^T V^{-1} A)^{-1} A^T V^{-1} y \tag{C.2}$$

The covariance matrix “P” of the output parameters is given by:

$$P = (A^T V^{-1} A)^{-1} \tag{C.3}$$

#### C.1 Hat Matrix

If the covariance matrix  $V$  is an identity matrix then equation (C.2) can be rewritten as:

$$\hat{x} = (A^T A)^{-1} A^T \vec{y}$$

$\hat{x}$  is estimated by the psuedo-inverse of matrix  $A$ .

Substituting for  $\bar{x}$  in equation (C.1), the final fitted value of  $\bar{y}$  is:

$$\hat{y} = A(A^T A)^{-1} A^T \bar{y} = H \bar{y}$$

where  $H$  is the Hat Matrix. The Hat Matrix is useful for detecting leverage or influential observations in a linear system. It is a square matrix with dimension equal to the length of the vector  $\bar{y}$  (i.e. the number of observations: "n"). The Hat Matrix  $H$  relates the final fitted values of  $\hat{y}$  to the original measurements. The diagonal entries of the hat matrix, which must lie between 0 and 1, signify the influence an observation has on its own final fit. The sum of diagonal entries equals "p" (the number of unknown variables). Ideally all observations should contribute equally to every observation's final fit. If there is equal contribution by all observations, then each diagonal values of the hat matrix would equal  $\frac{p}{n}$ . However, if some observations are more influential in the final fit, then their diagonal values are higher. Typically, observations are considered to have high leverage if  $h_{ii} > \frac{2p}{n}$ .

## BIBLIOGRAPHY

- [1] Adiv, G., "Interpreting Optical Flow," *PhD thesis, COINS Technical Report 85-35*, University Of Massachusetts at Amherst, MA., 1985.
- [2] Adiv, G. and E.M. Riseman, "Recovery of 3-D Motion and Structure from Image Correspondences using a Directional Confidence Measure," *COINS Technical Report 88-105*, University Of Massachusetts at Amherst, MA., 1988.
- [3] Aloimonos, J., "Unification and Integration of Visual Modules: an extension of the Marr paradigm," *Proceedings DARPA Image Understanding Workshop*, Morgan Kaufman Publishers, Palo Alto, CA., pp. 507-552, May 1989.
- [4] Anandan, P., "Measuring Visual Motion from Image Sequences," *PhD Thesis, COINS Technical Report TR 87-21*, University of Massachusetts at Amherst, MA., 1987.
- [5] Ayache, N. and O.D. Faugeras, "Building, Registrating and Fusing Noisy Visual Maps," *The International Journal of Robotics Research*, Vol. 7, No. 6, Dec. 1988.
- [6] Barnard, S.T., "Stochastic Stereo Matching over Scale," *International Journal of Computer Vision*, Vol. 3(1), pp:17-32, May 1989.
- [7] Besl, P.J., J. B. Birch and L. T. Watson, "Robust Window Operators," *Proceedings 2nd IEEE International Conference on Computer Vision*, Tampa, Florida, 591-600, Dec. 1989.
- [8] Beveridge, J.R., R. Weiss and E. Riseman, "Optimization of 2-Dimensional Model Matching," *Proceedings IEEE International Conference on Pattern Recognition*, Atlantic City, N.J., June 1990.
- [9] Bevington, P.R., *Data Reduction and Error Analysis for the Physical Sciences*, McGraw-Hill, NY, 1969.
- [10] Blake, A. and A. Zisserman, *Visual Reconstruction*, MIT-Press, Cambridge, Massachusetts 1987.
- [11] Broida, T.J. and R. Chellappa, "Estimating the Kinematics and Structure of a Rigid Object from a Sequence of Monocular Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 13, No. 6, pp. 497-513, 1991.

- [12] Chandrashekhara, S. and R. Chellapa, "A Two-Step Approach to Passive Navigation Using a Monocular Image Sequence," *USC-SIPI Technical Report 170*, University of Southern California, Electrical Engineering-Systems, 1991.
- [13] Cui, N., J. Weng and P. Cohen, "Extended structure and motion analysis from monocular image sequences," *Proceedings 3rd IEEE International Conference on Computer Vision*, Osaka, Japan, Dec. 1990.
- [14] Deriche, R., R. Vaillant and O.D. Faugeras, "From Noisy Edge Points to 3D Reconstruction of a Scene: A Robust Approach and its Uncertainty Analysis," *Proceedings 7th Scandinavian Conference on Image Analysis*, Aalborg, Denmark, Aug. 1991.
- [15] Dhondt, U.R. and J. K. Aggarwal, "Structure from Stereo - A Review," *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 19, No. 6, pp 1489 - 1510, Nov./Dec. 1989.
- [16] Dhome, M., M. Richetin, J.T. Lapreste and G. Rives, "Determination of the attitude of 3-D objects from a single perspective view," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 11, No. 12, Dec. 1989.
- [17] Dutta, R. and M. Snyder, "Robustness of Correspondence-Based Structure from Motion," *Proceedings 3rd IEEE International Conference on Computer Vision*, Osaka, Japan, Dec. 1990.
- [18] Faugeras, O.D., F. Lustman and G. Toscani, "Motion and structure from point and line matches," *Proceedings 1st International Conference of Computer Vision*, London, England, June 1987.
- [19] Faugeras, O.D. and G. Toscani, "Camera Calibration for 3D Computer Vision," *Proceedings International Workshop on Machine Vision and Machine Intelligence*, Tokyo, Japan, Feb 2-5, 1987.
- [20] Fennema, C., A. Hanson, and E. Riseman, "Towards Autonomous Mobile Robot Navigation," *Proceedings DARPA Image Understanding Workshop*, Morgan Kaufman Publishers, Palo Alto, CA., May 1989.
- [21] Fennema, C., A. Hanson, E. Riseman, J. R. Beveridge and R. Kumar, "Model-Directed Mobile Robot Navigation," *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 20, No. 6, Nov./Dec. 1990.
- [22] Fennema, C., "Interweaving Reason, Action and Perception," *PhD Thesis, COINS Technical Report 91-56*, University of Massachusetts at Amherst, 1991.
- [23] Fischler, M.A. and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, Vol. 24, pp. 381-395, 1981.

- [24] Forstner, W., "Reliability Analysis of Parameter Estimation in Linear Models with Applications to Mensuration Problems in Computer Vision," *Computer Vision, Graphics, and Image Processing*, 40, pp. 273-310, 1987.
- [25] Ganapathy, S., "Decomposition of transformation matrices for robot vision," *Proceedings 1st IEEE Conference on Robotics*, pp. 130-139, 1984.
- [26] Gelb, A. (ed.), *Applied Optimal Estimation*, MIT Press, Cambridge, MA, 1974.
- [27] Grunert, J.A., "Das Pothenotische Problem in erweiterter Gestalt nebst Bber seine Anwendungen in der Geodäsie," *Grunerts Archiv für Mathematik und Physik*, Band 1, pp. 238-248, 1841.
- [28] Hampel, F.R., E. M. Ronchetti, P. J. Rousseeuw and W. A. Stahel, *Robust Statistics, The approach based on Influence Functions*, John Wiley & Sons, N.Y. 1986.
- [29] Haralick, R.M., C.N. Lee, K. Ottenberg and M. Nolle, "Analysis and Solutions of the three point perspective pose estimation problem," *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, pp. 592-598, Lahiana, Maui, Hawaii, June 1991.
- [30] Haralick, R.M., H. Joo, C.N. Lee, X. Zhuang, V.G. Vaidya, and M.B.Kim, "Pose estimation from corresponding point data," *IEEE Transactions on systems, man and cybernetics*, Vol. 19, No. 6, Nov.Dec. 1989.
- [31] Haralick, R.M. and H. Joo, "2D-3D pose estimation," *Proceedings 9th IEEE International Conference on Pattern Recognition*, Rome, Italy, pp. 385-391, Nov. 1988.
- [32] Harris, C.G. and J. M. Pike, "3D Positional Integration from Image Sequences," *Image Vision and Computing*, pp. 87-90, May 1988.
- [33] Heel, J., "Dynamic Motion Vision," *Proceedings DARPA Image Understanding Workshop*, pp. 702-713, May 1989.
- [34] Horn, B.K.P., "Closed-form solution of absolute orientation using unit quaternions," *Journal Optical Society of America*, Vol. 4, pp. 629-642, 1987.
- [35] Horn, B.K.P., "Relative Orientation," *International Journal of Computer Vision*, Vol. 4, pp. 59-78, 1990.
- [36] Horaud, R., B. Conio, O. Laboullex and B. Lacolle, "An analytic solution for the perspective 4-point problem," *Computer Vision, Graphics and Image Processing*, Vol. 47, No. 1, pp. 33-45, July 1989.

- [37] Huang, T.S., J. Weng and N. Ahuja, "Motion and structure from two perspective views: Algorithms, error analysis and error estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 11(5), 1989.
- [38] Huber, P.J., *Robust Statistics*, John Wiley & Sons, N.Y. 1981.
- [39] Kim, D.Y., J. J. Kim, P. Meer, D. Mintz and A. Rosenfeld, "Robust Computer Vision: A Least Median of Squares Based Approach," *Proceedings DARPA Image Understanding Workshop*, Morgan Kaufman Publishers, Palo Alto, CA., May 1989.
- [40] Kite, D.H., and M. Magee, "Determining the 3D position and orientation of a robot camera using 2D monocular vision," *Pattern Recognition* Vol. 23, No. 8, pp. 819-831, 1990.
- [41] Konopinski, E.J., *Classical Descriptions of Motion*, Ch. 9, pp. 234-280, W. H. Freeman and Co., San Francisco.
- [42] Kumar, R., Sawhney, H.S. and Hanson, A.R., "3D Model Acquisition from Monocular Image Sequences," *Submitted for review to the IEEE Conference on Computer Vision and Pattern Recognition*, Champaign, Illinois, June 1992.
- [43] Kumar, R. and Hanson, A.R., "The Application of Pose Determination Techniques to Model Extension and Refinement," *Proceedings DARPA Image Understanding Workshop*, San Diego, California, Jan. 1992.
- [44] Kumar, R. and Hanson, A.R., "Sensitivity of Pose Refinement to Accurate Estimation of Camera Parameters," *Proceedings 3rd IEEE International Conference on Computer Vision*, Osaka, Japan, Dec. 1990.
- [45] Kumar, R. and Hanson, A.R., "Analysis of Different Robust Methods for Pose Refinement," *Proceedings IEEE International Workshop on Robust Computer Vision*, International Photogrammetric Society, Seattle, Washington, Oct. 1990.
- [46] Kumar, R., and A.R. Hanson, "Robust Estimation of Camera Location and Orientation from Noisy Data with Outliers," *Proceedings IEEE Workshop on Interpretation of 3D scenes*, Austin, Texas, Nov. 1989.
- [47] Kumar, R. "Determination of Camera Location and Orientation," *Proceedings DARPA Image Understanding Workshop*, Palo Alto, California, 1989.
- [48] Leclerc, Y.G., "Constructing Simple Stable Descriptions for Image Partitioning," *International Journal of Computer Vision*, Vol. 3, No. 1, 1989.
- [49] Lenz, R.K. and R.Y. Tsai, "Techniques for calibration of the scale factor and image center for high accuracy 3-D machine vision metrology," *IEEE Transactions on Pattern Analysis Machine Intelligence*, Vol. 10 # 5, pp. 713-719, 1988.

- [50] Lee, C.N., R. M. Haralick and X. Zhuang, "Recovering 3-D Motion Parameters from Image Sequences with Gross Errors," *Proceedings IEEE Workshop on Visual Motion*, Irvine, California, March 1989.
- [51] Li, G., "Robust Regression," *Exploring Data Tables, Trends and Shapes*. D.C. Hoaglin, F. Mosteller and J. W. Tukey (eds.), John Wiley & Sons, 281-343, 1985.
- [52] Linnainmaa, S., D. Harwood and L.S. Davis, "Pose determination of a three-dimensional object using triangle pairs," *IEEE Transactions on Pattern Analysis Machine Intelligence*, Vol. 10 # 5, pp. 634-647, 1988.
- [53] Liu, Y., T. S. Huang and O. D. Faugeras, "Determination of camera location from 2D to 3D line and point correspondences," *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, pp. 82-88, 1988.
- [54] Lowe, D.G., *Perceptual Organization and Visual Recognition*, Kluwer Academic Publishers, Hingham, MA, 1985.
- [55] Lowe, D.G., "Integrated treatment of matching and measurement errors for robust model-based motion tracking," *Proceedings IEEE 3rd International Conference on Computer Vision*, Osaka, Japan, Dec. 1990.
- [56] Lowe, D. G., "Fitting parametrized three-dimensional models to images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 13, No. 5, May 1991.
- [57] Manmatha, R., R. Dutta, E.M. Riseman and M. Snyder, "Issues in Extracting Motion Parameters and Depth from Approximate Translational Motion," *Proceedings IEEE Workshop on Visual Motion*, March 1989, pgs 264-272.
- [58] Matthies, L.H., "Dynamic Stereo Vision," *Ph.D. thesis*, Carnegie Mellon University, Oct. 1989.
- [59] Mitchie, A., S. Seida and J. K. Aggarwal, "Using constancy of distance to estimate position and displacement in space," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 10, No. 4, July 1988.
- [60] Mosteller, F., and J. W. Tukey, *Data Analysis and Regression*, Addison-Wesley, Reading, MA., 1977.
- [61] Oliensis, J. and J. I. Thomas, "Incorporating motion error in multi-frame structure from motion," *Proceedings IEEE Workshop on Visual Motion*, Princeton, N.J., Oct. 1991.
- [62] Paquette, L., R. Stampfer, Y. Dube and M. Roussel, "A New Approach to Robot Orientation by Orthonormal Lines," *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, Ann Arbor, Michigan, June 1988.

- [63] Pollard, S.B., T. P. Pridmore, J. Porrill, J. E. W. Mayhew and J.P. Frisby, "Geometric Modeling from Multiple Stereo Views," *The International Journal of Robotics Research*, Vol. 8, No. 4, Aug. 1989.
- [64] Press, W.H., B. P. Flannery, S. A. Teukolsky, W. T. Vetterling, *Numerical Recipes*, Cambridge University Press, Cambridge, MA, 1986.
- [65] Rousseeuw, P.J. and A. M. Leroy, *Robust Regression & Outlier Detection*, John Wiley & Sons, N.Y., 1987.
- [66] Samarov, A.M., "Bounded Influence Regression via Local Minimax Mean Squared Error," *Journal of the American Statistical Association*, Vol. 80, No. 392, Dec. 1985.
- [67] Sawhney, H.S. and A. R. Hanson, "Comparative Results of Some Motion Algorithms on Real Image Sequences," *Proceeding DARPA Image Understanding Workshop*, Morgan Kaufman Publishers, Pittsburgh, PA, 1990.
- [68] Sawhney, H.S., J. Oliensis and A. R. Hanson, "Description and Reconstruction from Images Trajectories of Rotational Motion," *Proceedings 3rd IEEE International Conference on Computer Vision*, pp.494-498, Osaka, Japan, Dec. 1990.
- [69] Sawhney, H.S. and A. R. Hanson, "Identification and 3D description of 'shallow' environmental structure in a sequence of images," *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, pp. 179-186, Hawaii, June 1991.
- [70] Strang, G., *Introduction to Applied Mathematics*, Wellesley-Cambridge Press, Wellesley, Massachusetts, 1986.
- [71] Stuelpnagel, J., "On the parametrization of the three-dimensional rotation group," *SIAM Review*, Vol. 6, No. 4, pp. 422-430, Oct. 1964.
- [72] Tsai, R.Y. and T. S. Huang, "Uniqueness and estimation of 3d motion parameters and surface structures of rigid objects," *Image Understanding 1984*, pp. 135-171, 1984.
- [73] Tsai, R.Y., "An Efficient and Accurate Camera Calibration Technique for 3D Machine Vision," *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, pp. 364-374, 1986.
- [74] Tseng, D.C., Z. Chen and J. Y. Lin, "An Effective Method for Determining the Robot Pose," *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, Ann Arbor, Michigan, June 1988.
- [75] Weng, J., N. Ahuja and T. S. Huang, "Optimal Motion and Structure Estimation," *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, pp. 144-152, 1989.



- [76] Williams, L.R. and A. R. Hanson, "Translating Optical Flow into Token Matches and Depth from Looming," *Proceedings 2nd International Conference on Computer Vision*, pp. 441-448, Tampa, Florida, 1989.
- [77] Wolf, P.R., *Elements of Photogrammetry*, McGraw Hill, New York, 1974.
- [78] Wolfe, W.J., D. Mathis, C.W. Sklair and M. Magee, "The Perspective View of Three Points," *IEEE Transactions on Pattern Recognition and Machine Intelligence*, Vol. 11, No. 1, Jan. 1991.
- [79] Worrall, A.D., K. D. Baker and G. D. Sullivan, "Model based perspective inversion," *Image and Vision Computing*, Vol. 7, No. 1, pp. 17-23, Feb. 1989.
- [80] Wu, J.J., R. E. Rink, T. M. Caelli and V. G. Gourishankar, "Recovery of the 3d location and motion of a rigid object through camera image (an extended kalman filter approach)," *International Journal of Computer Vision*, Vol. 3. pp. 373-394, 1988.
- [81] Yohai, V.J., "High Breakdown-Point and High Efficiency Robust Estimates for Regression," *The Annals of Statistics*, Vol. 15, No. 20, 1987.
- [82] Zhang, Z. and O. D. Faugeras, "Building a 3D World Model with a Mobile Robot: 3D Line Segment Representation and Integration," *Proceedings IEEE International Conference on Pattern Recognition*, Atlantic City, N.J., June 1990.
- [83] Zhang, Z. and O. D. Faugeras, "Tracking and grouping 3D line segments," *Proceedings 3rd International Conference on Computer Vision*, pp. 577-580, Osaka, Japan, 1990.