3D RECONSTRUCTION UNDER VARYING CONSTRAINTS ON CAMERA GEOMETRY FOR ROBOTIC NAVIGATION SCENARIOS

A Dissertation Presented

by

ZHONGFEI ZHANG

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

DOCTOR OF PHILOSOPHY

February 1996

Department of Computer Science

3D RECONSTRUCTION UNDER VARYING CONSTRAINTS ON CAMERA GEOMETRY FOR ROBOTIC NAVIGATION SCENARIOS

A Dissertation Presented

by

ZHONGFEI ZHANG

Approved as to style and content by:

Allen R. Hanson, Chair of Committee

Edward M. Riseman, Member

Richard S. Weiss, Member

Robin J. Popplestone, Member

Richard Hartley, Member

David W. Stemple, Department Chair
Computer and Information Science

Dedicated to my parents,

Prof. Yukun Zhang and Ms. Ming Song

ACKNOWLEDGEMENTS

When I was an undergraduate, I was determined to have a Ph.D. someday. Now as this day is approaching, I realize that every step of progress towards this goal is not only due to my own effort, but also due to the kind help and strong support of many people around me. I am very fortunate to have such a group of wonderful people making it possible for me to complete my Ph.D. study.

I feel deeply indebted to Professors Allen Hanson and Edward Riseman for their consistent support, inspiring advice, and friendly encouragement over these years. Al Hanson is the chair of my thesis committee. I greatly appreciate his tireless effort of painstakingly reading and commenting many drafts of this thesis many times over, in addition to helping me with our previous papers and reports. He is always available whenever I have problems or difficulties, and is very helpful in our many discussions from which I have greatly benefited. I have enjoyed working with him very much. Ed Riseman has also read several drafts of this thesis and given me very good comments on all of my work at UMass. Ed first taught me how to write a technical article; I still have the comments he gave me for the very first conference paper I wrote since coming to the UMass Vision Lab. I am very impressed by the broad vision and deep insight that Al and Ed have in computer vision area, and am amazed by their ability to manage the Vision group as a world-renowned research group. I have to say that I am very lucky to be a member of this group under their direction. Through these years, they have helped develop my interest in computer vision research, allowing me to mature as a serious researcher in this field.

I am also very grateful to the other three members of my thesis committee: Professors Richard Weiss and Robin Popplestone, both at the Dept. of Computer Science,

UMass, and Dr. Richard Hartley at GE CRD. Rich was the first person I worked with when I came to the Vision Lab at UMass. He gave me lots of help when I was experiencing hard times during the first few years. We had lots of discussions, and I am glad to see that our cooperation was turned fruitful. Robin introduced me to lots of robotics terminology and showed me many differences between British English and American English; it was a fun talking with him. Richard gave me lots of mathematical insight into computer vision research. I greatly appreciate his correction of several errors in my work and truly enjoyed the discussions I had with him.

During the years I was in the Graduate School at UMass, I have enjoyed staying and working at the Vision Group very much. I will always remember here and everyone in the lab. All the vision group alumni and current students whom I was with are worth acknowledging. In particular, I would like to express my warm gratitude to Dr. John Oliensis at NEC Research Institute. When he was with the group, I enjoyed lots of good conversations with him. The idea of the work in Chapter 4 was partly inspired by one of those conversations. He also kindly served as my reference when I was applying for jobs. I would like to thank Bob Collins and Yanxi Liu for kindly hosting many delicious dinners, which provided such good relaxation while I was busy studying or working. Every time I went to their cozy home, I felt as though I were at my own home. I also enjoyed a lot of discussions with Bob Collins and appreciate much of his help. I would like to thank Lance Williams for many of the discussions I had with him and also for the help he gave me; he is a real buddy. I would also like to thank Bruce Draper for his comments and discussions when I was working on visual servoing controlled navigation, Teddy Kumar for providing his data for the experiments in Chapter 4, and my ex-officemate R. Manmatha for many interesting discussions, both academic and non-academic. Many thanks also to Poornima Balasubramanyam, Ross Beveridge, Shashi Buluswar, Brian Burns,

Yongqing Cheng, Chris Connolly, Madi Das, John Dolan, Rabi Dutta, Claude Fennema, Chris Jaynes, Gökhan Kutlu, Jonathan Lim, Brian Pinette, Chandu Ravela, Harpreet Sawhney, Howard Schultz, Michael Scudder, Frank Stolle, Inigo Thomas, Xiaoguang Wang, and Victor Wu for their numerous discussions and help. I would like to thank Yongqing, Xiaoguang, and Victor for their hospitality. My thesis would never be at this stage without technical help from Bob Heller and Jonathan Lim. Also I owe Laurie Downey and Janet Turnbull a big favor for their administrative help over these years.

My academic life in the Graduate School wouldn't have been so enjoyable if I hadn't had opportunities to work with people outside the University. Here I would like to thank Dr. David Jacobs at NEC Research Institute for sponsoring me for an internship in the Summer of 1994. I would also like to thank Drs. Frank Glazer, Charlie Kohl, and John Dolan at Amerinex Applied Imaging when I worked there during Fall, 1994 for the Mystic project, and partly in Spring, 1995 for the IUE project. Last but not the least, I would like to thank Professors Sargur Srihari and Rohini Srihari at SUNY Buffalo for graciously offering me such a good position as a research scientist at the Center of Excellence for Document Analysis and Recognition (CEDAR), and for generously allowing me to use CEDAR's resources to complete my thesis. Thanks also to Ajay Shekhawat for his technical support at CEDAR.

Finally, I owe my parents, Prof. Yukun Zhang and Ms. Ming Song, boundless gratitude for their tireless effort in caring for and educating me to be what I am now. They have not only provided me with an excellent education, but also helped me develop good character. I am sure that my Ph.D. is a good reward for them. I am also deeply grateful to my sister, Xuefei Zhang, my brother-in-law, Xiaolin Pei, and my nephew, Liang, for their help and care over these years, and for always providing me a warm place whenever I needed one.

ABSTRACT

3D RECONSTRUCTION
UNDER VARYING CONSTRAINTS ON CAMERA GEOMETRY
FOR ROBOTIC NAVIGATION SCENARIOS

FEBRUARY 1996

ZHONGFEI ZHANG

B.S., ZHEJIANG UNIVERSITY

M.S., ZHEJIANG UNIVERSITY

PH.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor Allen R. Hanson

3D reconstruction is an important research area in computer vision. With the wide spectrum of camera geometry constraints, a general solution is still open. In this dissertation, the topic of 3D reconstruction is addressed under several special constraints on camera geometry, and the 3D reconstruction techniques developed under these constraints have been applied to a robotic navigation scenario. The robotic navigation problems addressed include automatic camera calibration, visual servoing for navigation control, obstacle detection, and 3D model acquisition and extension.

The problem of visual servoing control is investigated under the assumption of a structured environment where parallel path boundaries exist. A visual servoing control algorithm has been developed based on geometric variables extracted from this structured environment. This algorithm has been used for both automatic camera calibration and navigation servoing control. Close to real time performance is achieved.

The problem of qualitative and quantitative obstacle detection is addressed with a proposal of three algorithms. The first two are purely qualitative in the sense that they only return yes/no answers. The third is quantitative in that it recovers height information for all the points in the scene. Three different constraints on camera geometry are employed. The first algorithm assumes known relative pose between cameras; the second algorithm is based on completely unknown camera relative pose; the third algorithm assumes partial calibration. Experimental results are presented for real and simulated data, and the performance of the three algorithms under different noise levels are compared in simulation.

Finally the problem of model acquisition and extension is studied by proposing a 3D reconstruction algorithm using homography mapping. It is shown that given four coplanar correspondences, 3D structures can be recovered up to two solutions and with only one uniform scale factor, which is the distance from the camera center to the 3D plane formed by the four 3D points corresponding to the given four correspondences in the two camera planes. It is also shown that this algorithm is optimal in terms of the number of minimum required correspondences and in terms of the assumption of internal calibration.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Robotic navigation is an interdisciplinary research area that involves techniques from robotics and computer vision. 3D reconstruction, one of the most important and active research areas in computer vision, plays a central role in robotic navigation. First, a successful robotic vehicle navigation scenario requires the ability to follow a road. To achieve this goal, one needs to reconstruct either an implicit or explicit 3D road model from 2D images. For example, an implicit 3D road model may be represented by a set of parameters computed from servoing parameters (see Chapter 2) while an explicit 3D road model may involve a direct representation of what a 3D road looks like [16, 90]. Second, a successful robotic navigation system must be able to detect obstacles during navigation. These two minimal requirements imply the need for some form of 3D reconstruction since obstacle detection is typically based on either partial (see Chapter 3) or complete 3D reconstruction [86, 105]. Third, in many applications, successful robotic navigation scenarios may require automatic 3D model acquisition and extension, especially in changing environments. This is particularly necessary in model-based navigation systems [43]. The problem of model acquisition and extension *per se* involves 3D reconstruction, and thus, it is just another application of 3D reconstruction techniques. Therefore, 3D reconstruction techniques are

central to research in robotic navigation, and it is necessary for a robotic navigation system to be equipped with different 3D reconstruction techniques in order to handle different navigation scenarios successfully (e.g. road following, obstacle detection, etc.).

In the next two sections, we give a brief introduction to 3D reconstruction and robotic navigation, respectively. Then we introduce the notation used in this dissertation. Finally we outline the organization of this dissertation.

1.1 3D Reconstruction

3D reconstruction, as the term implies, refers to recovering 3D structures from 2D images. It is a more mature, yet still very active subfield of computer vision. The most common techniques for recovery of 3D information from 2D images make use of stereo and/or motion information. Thus, 3D reconstruction is highly related to the fields of stereo vision and structure from motion.

The field of 3D reconstruction predates the advent of computer vision. Long before the development of computers, the necessity and importance of 3D reconstruction was recognized, and many techniques were proposed. These techniques were, of course, based on manual and mechanical constructions. A good example is in the area of tissue pathology. In order to analyze and examine human or animal tissues, scientists took pictures of the tissue. But in many situations, the information contained in the 2D images was inconvenient and/or insufficient to do scientific analysis. Thus, it became necessary to reconstruct 3D models of the tissue based on 2D images. 3D reconstruction has since become an important research direction in biological

and medical science. Gaunt and Gaunt [33] surveyed many techniques proposed for 3D reconstruction in biology. Most of them were manually based, including serial reconstruction techniques [55, 112], graphical reconstruction techniques [97, 117, 108, 87, 88], solid reconstruction techniques [10, 107, 37, 94], and more modern stereophotography reconstruction techniques [4, 42, 30, 127, 76].

Another area that showed early development and application of 3D reconstruction techniques is photogrammetry [104]. Back to the 19th century, owing to the development of industrialization, people found it was necessary to perform geographical surveys and terrestrial measurement. Terrestrial surveys were achieved by using photographs to reconstruct the 3D terrain map. In 1843, the French painter and physicist Louis Daguerre pioneered this work by creating a successful survey of a harbor from photographs taken from a ship, a submarine survey at a depth of 900 meters, and a cloud measurement in England. In 1859, the French caricaturist, photographer, and fashionable sports balloonist Tournachon created a countryside map from photographs taken from a balloon. In 1864, a French designer of terrestrial equipment for photographic surveying and the father of photogrammetry, Aime Laussedat, pioneered the work of reconstruction of classical architecture and created a survey of Paris from rooftop photography. Based on their work, many scientists in Austria, Germany, Britain, and Italy followed up and did many similar photography-based surveys in the early 1900's. In the United States, the first triangulation-based survey method was suggested by C.B. Adams in 1893, and was actually used in Canada to perform an Alaskan boundary survey in 1894.

In computer vision, one of the major goals is to understand 3D scenes through one or more 2D images. Here we will focus on stereo and motion research. Binford [9],

Quam and Hannah [93], Shapira [98], Turner [118], and Moravec [84] are among the early efforts in recovering 3D structure information based on stereo vision. Marr and Poggio [77] proposed a computational model based on human stereopsis to reconstruct 3D information. This model was later refined to agree better with psychological data [78, 17].

In the area of reconstruction of 3D structure using motion, Ullman [119] was one of the first to address the issue of the uniqueness of the 3D structure reconstructed from motion information. He showed that given four non-coplanar correspondences of three orthographic projections, the corresponding 3D structure could be uniquely determined. Shapira [98], Shapira and Freeman [99], and Wesley and Markovsky [124] investigated 3D reconstruction with unknown correspondences.

3D reconstruction is still a very active area in computer vision because many of its problems are still open. Unlike the problem of 2D reconstruction based on 1D projections, which is relatively simple and for which closed form solutions [95] exist, 3D reconstruction is much more complicated. The complexity of the general 3D reconstruction problem stems from the wide spectrum of the varying constraints on camera geometry. In the one extreme, if we have two cameras with known internal and external calibration, then simple triangulation methods may apply. In this case, as pointed out by Ballard and Brown [5], the problem reduces to one of determining the correspondences between the two images robustly and reliably. This is still a very difficult problem, and often additional assumptions must be made to find correspondences. Even if the correspondences are known, how to robustly reconstruct the 3D scene with the noise-corrupted data is still a non-trivial problem, and is an on-going

research topic [53]. At the other extreme, neither internal calibration nor external calibration of the cameras is known. In recent years, this problem has become more and more interesting to many researchers [24, 25, 82, 44, 52, 83, 100, 91, 96, 69, 48, 47, 51]. So far, research efforts have shown that given two completely uncalibrated views, the best solution to 3D reconstruction is only up to a projective or affine transform. In other words, it is impossible to reconstruct a 3D scene in Euclidean space with only two completely uncalibrated views. Consequently, the question of how to directly reconstruct a 3D scene in Euclidean space from completely uncalibrated cameras with more views and/or fewer correspondences still remains open. Between the two extremes, there may be different constraints on the camera geometry, and hence many different problems of 3D reconstruction may be proposed under these constraints. This thesis is aimed at discussing 3D reconstruction problems under some of the varying constraints on camera geometry between the two extremes described above, and proposes solutions to these problems in the application domain of robotic navigation.

1.2 Robotic Navigation

Navigation is also a very old topic, and its history mostly likely parallels that of the human race. Robotic navigation, as the term implies, refers to the whole process of automatic movement of a mobile robot and any potentially related action(s) taken by the robot. Specifically, according to Leonard and Durrant-Whyte [72], the problem of robotic navigation can be summarized by the following three questions:

- *Where am I?*

- *Where am I going?*
- *How should I get there?*

The first question addresses the problem of localization, i.e. what is the current pose (position and orientation) of the robot, and how to determine this pose w.r.t. some prespecified world coordinate system. The second question is simply to determine the goal of this particular navigation course, which may be represented by another pose in the same world coordinate system. The third one is the most complicated question. In general, it may include automatic servoing, road following, obstacle detection and avoidance, path planning, landmark detection and recognition, information retrieval and indexing in a map database, correspondence between the current working data and map database data, etc. Owing to the wide spectrum of problems, and owing to the fact the all these problems are nontrivial, a general solution to robotic navigation is still an open problem, although many researchers have made numerous efforts in trying to solve the problem in different special cases [2, 11, 14, 16, 20, 21, 22, 28, 32, 56, 62, 67, 70, 71, 74, 75, 89, 90, 116, 121, 134, 135].

In this dissertation, we will address some of the problems related to robotic navigation in some special cases by applying the techniques developed in 3D reconstruction and geometric analyses directly to the robotic navigation scenario. Again, our solutions are only valid under the special cases assumed. The applications of these techniques to the more general scenario of robotic navigation is a future research direction.

1.3 Outline of the Dissertation

This dissertation is composed of five chapters, including this introductory chapter. Chapter 2 presents a visual servoing system based on geometric analyses of a structured environment; the techniques developed are used to solve automatic calibration and navigation servoing problems. In a structured environment, we assume that the ground plane is horizontal and two locally parallel side-lines demarcate the boundaries of the "road". This assumption holds in many indoor environments, such as hallways, where the system has been successfully demonstrated. The geometric variables explored and used in this chapter include vanishing points, directional axis, and looming distance. The automatic calibration problem refers to how to continuously determine a path and associated robot motion from an arbitrary pose to a desired pose, so that the robot can maneuver to the desired pose accurately. The navigation problem of concern here is how to maintain the robot on a given path while it is under motion. Both theoretical analyses and experimental results show that the system works robustly and accurately.

Chapter 3 addresses the problem of obstacle detection during robot navigation. In this chapter, we discuss the trade-offs between qualitative and quantitative obstacle detection. Specifically, three different algorithms for obstacle detection are presented in this chapter. Each one is based on different assumptions. The first two algorithms are completely qualitative in the sense that they only return yes/no answers about the presence of obstacles in a view; this is done without reconstructing actual 3D structures. They have the advantage of fast determination of the existence of obstacles in a scene based on the solvability of a linear system. The first algorithm

uses information about the location of the ground plane. The second algorithm only assumes that the ground is planar, but may be unknown. While the first two algorithms assume that the ground plane is planar, which may not be true in the outdoor navigation with very rough ground, a third, more quantitative algorithm is developed which continuously estimates the local ground plane using a Kalman Filter. In each navigation step, based on the estimated ground plane, this algorithm reconstructs partial 3D structures by determining the height of each point in the scene. In each of the algorithms, we assume that the correspondences have been computed. The first two algorithms may be applied to either monocular motion data or stereo, while the third one can only be applied to stereo. Experimental results are presented for real and simulated data, and the performance of the three algorithms under different noise levels are compared in simulation. We conclude that in terms of the robustness of performance, the third algorithm is superior to the other two.

In Chapter 4, 3D reconstruction based on a homography mapping is proposed, and this algorithm is applied to model acquisition and extension for robot navigation scenarios. Previous work shows that based on the fundamental matrix from two views, 3D reconstruction can be achieved under an unknown projectivity. In this chapter, we show that based on four *coplanar* correspondences of two externally uncalibrated cameras, 3D reconstruction can be achieved in Euclidean space with only one uniform scale factor and up to two real solutions. It is shown that this scale factor is the physical distance from the camera center to the plane formed by the four points in 3D space. Consequently, if this distance is known *a priori*, then the 3D structure can be determined completely up to two solutions. In order to disambiguate the two solutions, a third view is required in general. The basic idea of this approach is that,

given four coplanar correspondences, a homography mapping between two views may be established, and the relative geometry between the two cameras, together with the relationship to the 3D reference plane formed by the four points in 3D space, can be obtained by explicitly decomposing the homography matrix. Therefore, the Euclidean coordinates of any 3D point can be obtained in a camera coordinate system based on the solved relative geometry between the two cameras. In practice, since the real data are always corrupted with noise, more coplanar correspondences are needed and a least squares solution is applied to obtain the estimate of the homography matrix. Results on both simulated and real data show that the reconstruction algorithm works reasonably robustly. We conclude this chapter with a proof that this algorithm is optimal.

Finally, Chapter 5 summarizes the main contributions of this dissertation, and outlines future research directions.

1.4 Notation

Throughout this dissertation, we use the following notation. Boldface symbols denote vectors or matrices, and non-boldface symbols denote scalar variables. The symbol T appearing at the upper right corner of a vector or a matrix means the transpose of that vector or that matrix. The relative pose between two cameras is represented by a translation vector $\boldsymbol{t} = (t_X, t_Y, t_Z)^T$, and a 3×3 rotation matrix $\boldsymbol{R}$. Sometimes, a rotation is represented by an explicit angle vector $\boldsymbol{\Omega} = (\Omega_X, \Omega_Y, \Omega_Z)^T$, whose three components are the rotation angles with respect to the three coordinate axes, while other times a rotation is represented as a quaternion.

We will define $\boldsymbol{p}$ to be a point in the first image (in a motion sequence) or in the left image (in stereo) and $\boldsymbol{p}'$ to be the corresponding point in the second image (in a motion sequence) or in the right image (in stereo). We always use upper case to denote 3D vectors or coordinates, and use lower case to denote 2D image vectors or coordinates. Thus, $\boldsymbol{P}$ denotes the corresponding 3D point, and $\boldsymbol{p} = (x, y, 1)^T$, and $\boldsymbol{P} = (X, Y, Z)^T$, etc.

The motion parameters between two camera poses are represented by a vector $\boldsymbol{M} = (\boldsymbol{\Omega}^T, \boldsymbol{t}^T)^T$. The symbol $\Longleftrightarrow$ is used to represent a "correspondence" relationship, and the symbol $\cong$ is used to represent "projective equivalence".

CHAPTER 2

VISUAL SERVOING CONTROL IN STRUCTURED ENVIRONMENTS

2.1 Introduction

This chapter is devoted to the problem of visual servoing control. Servoing is one of the most important areas in robotics and control research [8]. In the literature, servoing typically refers to a feedback control process for an actuator (e.g. a robot, or its arm) in motion using an active and/or passive sensor [8, 15] to generate the error signal. Visual servoing thus refers to this type of feedback control process using visual sensors (e.g. typically, a camera). Using a camera for visual servoing in robotic navigation has many obvious advantages over other commonly used sensors (e.g. LADAR or FLIR): cameras are inexpensive, light, passive, rugged and easy to operate. Owing to these advantages, the use of visual servoing presents opportunities for greatly increasing the flexibility and accuracy of robotic automation tasks. The application of machine vision as an improvement to robot performance is becoming an essential element for the integration of robotic technology in many application areas, and is becoming one of the most important areas in robotics research [6].

Machine vision enhances the robot's interaction with its work environment and helps obtain greater system flexibility [31]. In general, the function of visual servoing

is to determine the spatial relationship between camera, workpiece, and the robot [7]. There are many practical scenarios that visual servoing control can be applied to. Robotic navigation, robotic arm control, robotic hand control, etc., are just a few examples.

The goal of the research in this chapter is to develop a visual servoing control system with *real time performance*. We assume that the robotic actuator is the robot itself, and that the visual sensor is a video camera mounted on top of the robot. Specifically, we address the visual servoing control problem by investigating two important application problems: automatic calibration and navigation servoing. To simplify further, we assume here that the application environment of the visual servoing control techniques proposed in this chapter is structured. By a structured environment, it is meant here that the robot is spatially constrained to a path defined by two locally parallel boundary markers that are visible to the robot. Examples of this kind of environment can be found in many places, such as hallways, city roads, etc. Although the environmental structure and the Denning mobile robot used in the experiments do not completely satisfy these constraints, it can be seen via theoretical analysis and the experimental results that these assumptions are a close approximation to the actual situation. This chapter is an extension and revision of the earlier work [130, 131, 132], which was motivated by the need for an automatic calibration system for model-based mobile navigation [28]. Two robust, real time systems for automatic calibration and navigation servoing have been built on the basis of the algorithm developed in this chapter [132].

The term "calibration" has been in several different contexts in robotics and computer vision. *Internal camera calibration* or *intrinsic camera calibration* [113, 38, 103]

refers to the process of determining the internal parameters of a camera, such as the coordinates of the principal point, the scale factors for both image axes, and the angle between the two axes. *Robot calibration* [120, 54, 125] refers to the process of determining a kinematic model of a robot to provide accurate positioning of the robot actuator. This is a purely mechanical calibration, and typically has nothing to do with the camera or vision component. From a robotic system point of view, calibrating the vision system sensors and the robot kinematic model separately can help improve the system accuracy, but does not eliminate errors in the transformations which relate the 3D scene to the 2D image [6]. Thus, *external camera calibration* or *extrinsic camera calibration* [65, 115] refers to the process of determining the transformation matrix between the camera coordinate system and a prespecified world coordinate system. The transformation matrix can be further decomposed into two components: a positional (or locational) component (which is represented as a translation vector), and an orientation component (which may be represented as a rotation matrix or a vector). Each component has three degrees of freedom. Thus, this transformation matrix has, in general, six degrees of freedom.

The combination of the position and orientation of a camera w.r.t. a world coordinate system is called its *pose*. In many application scenarios, the camera coordinate system is assumed to be the same as the robot coordinate system. In this case, the robot pose is the same as the camera pose. In this dissertation, we follow this assumption. Therefore, in the following text, we do not distinguish between camera pose and robot pose. Hence, a pose of a robot has six degrees of freedom in general, and the problem of external camera calibration is no different from the problem of pose determination of the robot.

In many robotic navigation applications, *visual servoing* is typically defined as the process of maintaining an error function within a controlled threshold by using visual sensor feedback. Different variables can be used to provide the servoing error function. For example, the robot can servo to a desired pose, or it can servo to a desired path and a desired direction along the path. The former is called *automatic external calibration*, or just *automatic calibration*, while the latter is called *navigation servoing*.

In general, determination of a pose requires visible reference points or landmarks [68]. Fig. 2.1 depicts the typical scenario for the visual servoing control problem investigated in this chapter. The environment is structured with visible path boundaries and a fixed landmark, and the ground plane is flat. Moreover, the robot can only rotate around its vertical axis (that is, it can only pan). Hence, the robot has three degrees of freedom in total (two location and one orientation). The robot local coordinate system ($X - Y - Z$) is set up in such a way that the Z axis is the camera focal axis, and the $X - Z$ plane is always parallel to the ground plane. The two particular visual servoing problems we shall solve in this chapter are defined below:

- **automatic calibration problem:** Assume that the initial pose of the robot is labeled A, and the desired pose is labeled B. The automatic calibration problem is to continuously determine a path and motion from pose(A) to pose(B) so that the robot can maneuver to pose(B) accurately.

- **navigation servoing problem:** Initially the robot is located at position A with an arbitrary orientation, and let line L be the desired navigation path.

The navigation servoing problem is to move the robot from A to the path L, and to servo the robot to stay on this path during navigation.

Conceptually, the problem of navigation servoing can be reduced to the problem of automatic calibration if a sequence of discrete points are hypothesized along the given path and the robot servos to those points in sequence. At the implementation level, however, a dynamic control strategy has to be applied to the solution to navigation servoing to "smooth" the motion at those hypothetical discrete points (see Sec. 2.5.2).

Owing to the practical interests of visual servoing control, many systems have been developed under different assumptions [2, 11, 14, 16, 20, 21, 22, 28, 32, 56, 62, 67, 70, 71, 74, 75, 89, 90, 116, 121, 134, 135]. The following two subsections briefly review the recent literature in the area of automatic calibration and navigation servoing.

2.1.1 Automatic Calibration

As mentioned earlier, automatic calibration is no different from pose determination. Fukui [32] was one of the earliest efforts to use pose determination in robotics. The method used a diamond-shaped mark that had its diagonals horizontally and vertically oriented. The camera lens and mark centers were assumed to be at the same height as the optical axis of the camera pointing at the center of the mark. By trigonometrically relating the lengths of the projected and unprojected vertical diagonals, the pose of the camera was determined. Courtney *et al* [14] modified the algorithm by using the same mark but relaxing the constraint of having the lens center at the same height as the mark center. Later Magee and Aggarwal [75] used a sphere with horizontal and vertical great circles marked on its surface. Kabuka *et*

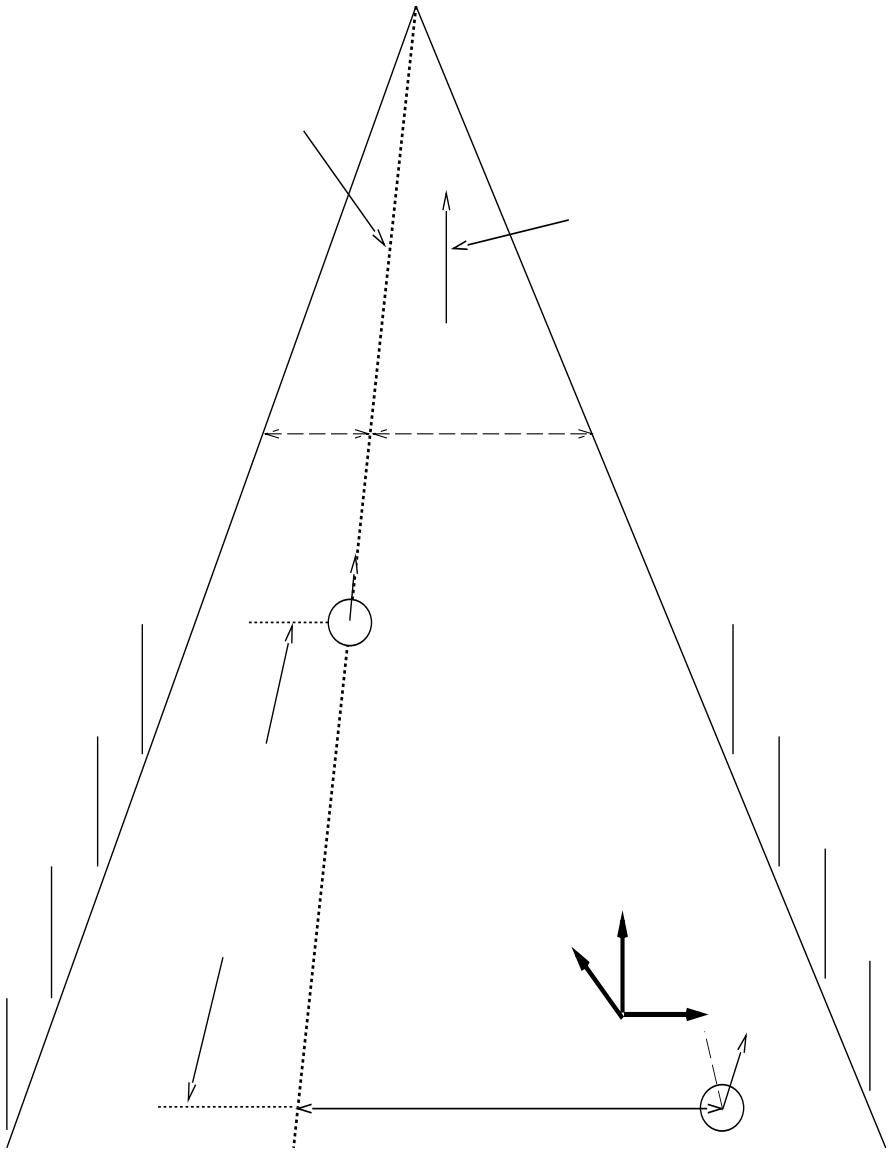

Figure 2.1. The general scenario of the visual servoing control problem.

al [62] used bar codes as landmarks for pose determination and verification. Lowe [74] used a viewpoint consistency constraint to match a set of characteristic features against models and achieved relatively good results for pose determination, although there was no report on error measures. Krotkov [67] reduced the pose determination problem to a global optimization problem to match a set of known landmarks and a set of rays such that each ray pierces at least one landmark. Kumar and Hanson [70] presented a pose estimation and refinement algorithm for computing the transformation matrix between the world coordinates and image coordinates. The algorithm works very robustly in the sense that it can handle a large percentage of outliers. The assumption made for this algorithm is that the correspondences of features between images are known *a priori*. Lee and Deng [71] presented an algorithm to estimate the pose information and camera parameters in a hallway environment. Their algorithm assumes that the ground plane is flat, and that there are some visible vertical landmarks in images. The pose determination part of this algorithm is similar to the algorithm proposed in this chapter. However, their algorithm requires three images, in general, to compute the pose information, whereas the algorithm proposed in this chapter uses only one image. Chenavier and Crowley [11] used an extended Kalman Filter to combine vision and odometry to determine pose.

2.1.2 Navigation Servoing

Waxman *et al* [121] reported some of the earliest efforts for developing a complete visual navigation system for an autonomous land vehicle. Dickmanns and Graefe [21, 22] applied Kalman Filter theory to develop a technique for using image features in a real-time feedback control loop to servo the motion of a vehicle. They applied

this technique to successfully drive a vehicle at high speeds on the German autobahn. Fennema *et al* [28] used 3D models to generate projections of landmarks expected to be seen from the estimated current location; the robot then servos directly on the image features, tracking them via correlation. At about the same time, three successful navigation systems were reported from CMU: the SCARF and UNSCARF systems for rural areas [16], the YARF system for city roads [2], and the ALVINN system based on neural networks [90]. Hong *et al* [56], Pinette [89] and Zhang *et al* [134, 135] developed homing-based navigation techniques for robot navigation using a spherical mirror.

2.1.3 Chapter Organization

This chapter is organized as follows. In the next section, a geometric analysis is conducted based on the assumptions made in this chapter. Then an algorithm for visual servoing control is proposed that exploits the geometric information under the assumptions to solve the problem. Two real time systems for automatic calibration and navigation servoing are developed based on this algorithm. This is followed by a theoretical error analysis of the preformance of these algorithms. Finally, experimental results of the real time performance of the two systems are presented. The chapter finishes with an analysis of the results, concluding that this proposed algorithm works robustly with real time performance.

2.2 Geometric Analysis

In this chapter, we develop an algorithm that exploits the geometric properties of the scene under the assumptions listed below to solve both the automatic calibration and navigation servoing problems simultaneously. First, we state the assumptions. In addition to the assumption of a structured environment (*two parallel path boundaries*), we also assume:

- the robot has one camera mounted on the center of the robot rotational axis, such that the focal center of the camera coincides with the rotational center of the robot;
- the intrinsic camera parameters are known;
- the ground plane is approximately flat;
- the camera focal axis is approximately parallel to the ground plane;
- for the automatic calibration problem, there is a visible landmark with known height.

Let us first define the following terminology:

- *vanishing point:* In general, if there are parallel lines in an environment (such as the two path boundaries in Fig. 2.1), their projections to the image domain under perspective transformation are a set of intersecting lines. This is true unless the image focal axis is perpendicular to the plane defined by the parallel lines in 3D. The intersection point in the image domain is called the vanishing

point [13, 12, 116, 64], as illustrated in Fig. 2.1. In the case when the camera focal axis is perpendicular to those lines, the vanishing point is at infinity.

- *directional axis:* For the automatic calibration problem, the directional axis is the *implicit* line parallel to the two path boundaries, and passing through the desired pose; for the navigation servoing problem, the directional axis is defined as the path that the robot is trying to follow as closely as possible. Referring to Fig. 2.1, the line L is the directional axis.

- *looming distance:* This is a geometric variable only used for the automatic calibration problem. Referring to Fig. 2.1, the looming distance is defined as $Z(B) - Z(A)$.

- *lateral distance:* Referring to Fig. 2.1, the lateral distance is defined as $X(B) - X(A)$.

- *orientation angle:* Referring to Fig. 2.1, the orientation angle is defined as $\theta(B) - \theta(A)$.

We begin by investigating geometric properties associated with the three geometric variables: *orientation angle*, *lateral distance*, and *looming distance*, and their relationships w.r.t. vanishing points and the directional axis.

2.2.1 Vanishing Point

Property 1 *The vanishing point is located at the principal point*[1] *of the image plane, no matter where the robot is positioned, if and only if the camera's axis is parallel to the path boundaries on the ground plane; if the camera is not parallel to the path boundaries, the locus of the vanishing point must be on the horizon line passing through the principal point of the image. The orientation angle (denoted as θ) is determined by:*

$$\theta = \arctan \frac{X_v - X_c}{S_X} \tag{2.1}$$

where X_v is the X coordinate of the vanishing point in the image, X_c is the X coordinate of the principal point of the image, and S_X is the scale factor along the X axis of the image plane.

In other words, this property says that the vanishing point in the image is *only dependent* on the *orientation* of the camera, and *independent* of the *position* of the robot.

It is easily seen that this property is true. As illustrated in Fig. 2.2(a), assuming the robot is positioned at an arbitrary location, as long as the camera axis is aligned with the path boundary direction, the optical axis of the camera is parallel to the path boundaries. In this case the intersection line of the projection planes of the two path boundaries coincides with the optical axis of the camera. If the image plane is perpendicular to the optical axis and also perpendicular to the ground plane, the

[1]The principal point or the center point of an image is that unique point in the image plane for which a ray through the focal point is perpendicular to the image plane.

intersection point of the optical axis of the camera and the image plane is exactly the intersection point of the two path boundaries in the image. Therefore, this point is always located at the principal point of the image plane.

If the camera's orientation makes an angle θ with the direction of the path boundaries, as illustrated in Fig. 2.2(b), this angle is exactly the angle between the intersection line of the two projection planes of the two path boundaries and the optical axis of the camera. Recall the assumption that the optical axis is parallel to the ground plane. Thus, the plane formed by this intersection line and the optical axis (say, plane P) is also parallel to the ground plane. Since the image plane is perpendicular to the ground plane, the vanishing point must lie on the intersection line between plane P and the image plane. Consequently, it is a horizon bisector line of the image plane. Since the optical axis of the camera is perpendicular to the image plane, from the obvious trigonometric relationship, Eq. 2.1 is easily obtained.

2.2.2 Directional Axis

Property 2 *The image of the directional axis is a vertical line in the image plane, no matter what direction the camera is oriented, if and only if the robot is positioned on the directional axis; if the robot is not positioned on the directional axis, the lateral distance is determined as:*

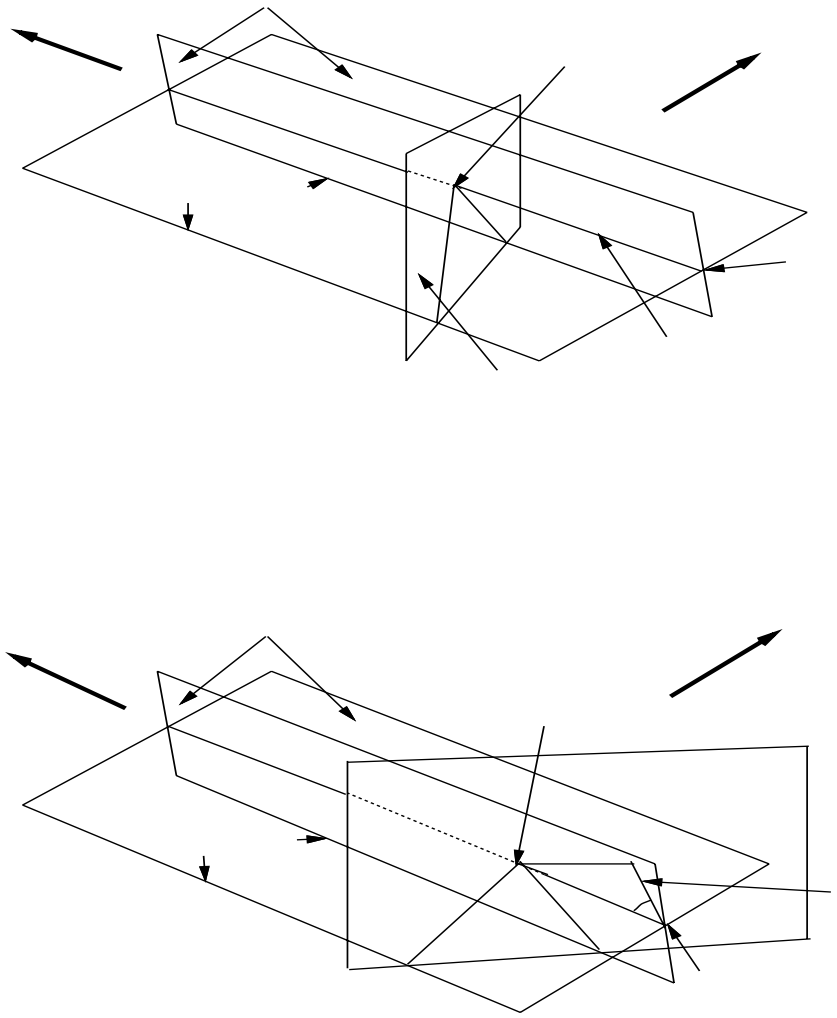

Figure 2.2. Illustration of **Property 1.** (a) when the camera orientation angle is 0, i.e., the focal axis coincides with the intersection line of the two projection planes of the path boundaries; (b) when the camera orientation angle is not 0; note that the plane formed by the camera focal axis and the intersection line of the projection planes is referred to as plane P in the text.

$$d = -\frac{b\cos(\beta+\gamma)\sin\beta}{\sin\gamma} \tag{2.2}$$

where α and β are the directional angles of the two path boundaries in the image plane, a and b are the distances from the directional axis to the two path boundaries in 3D, respectively, as illustrated in Fig. 2.1 and Fig. 2.3, and γ is determined by:

$$\gamma = \arctan\frac{b\sin\beta\sin(\alpha+\beta)}{a\sin\alpha - b\sin\beta\cos(\alpha+\beta)} \tag{2.3}$$

In other words, this property states that the verticalness of the directional axis in the image is *only dependent* on the *position* of the robot, and *independent* of the *orientation* of the camera. Clearly, this is a complementary property with respect to Property 1.

Here, the angle $\beta+\gamma$ indicates the orientation of the directional axis in the image. As shown in Fig. 2.3, if $\beta+\gamma < \frac{\pi}{2}$, d is negative, which means that the robot is on the right side of the directional axis at a distance $-d$; if $\beta+\gamma > \frac{\pi}{2}$, d is positive, which means that the robot is on the left side of the directional axis at a distance d; if $\beta+\gamma = \frac{\pi}{2}$, $d = 0$, then the robot is exactly on the directional axis.

To see that **Property 2** is true, refer to Fig. 2.4. First let us consider the case when the orientation is along the directional axis, i.e., $\theta = 0$. In this case, the intersection line formed by the two projection planes of the path boundaries coincides with the optical axis of the camera, and passes through the focal point, as shown in Fig. 2.4(a). The current directional axis *in the image plane* is formed by the intersection of the projection plane of the current directional axis on the ground and the image plane.

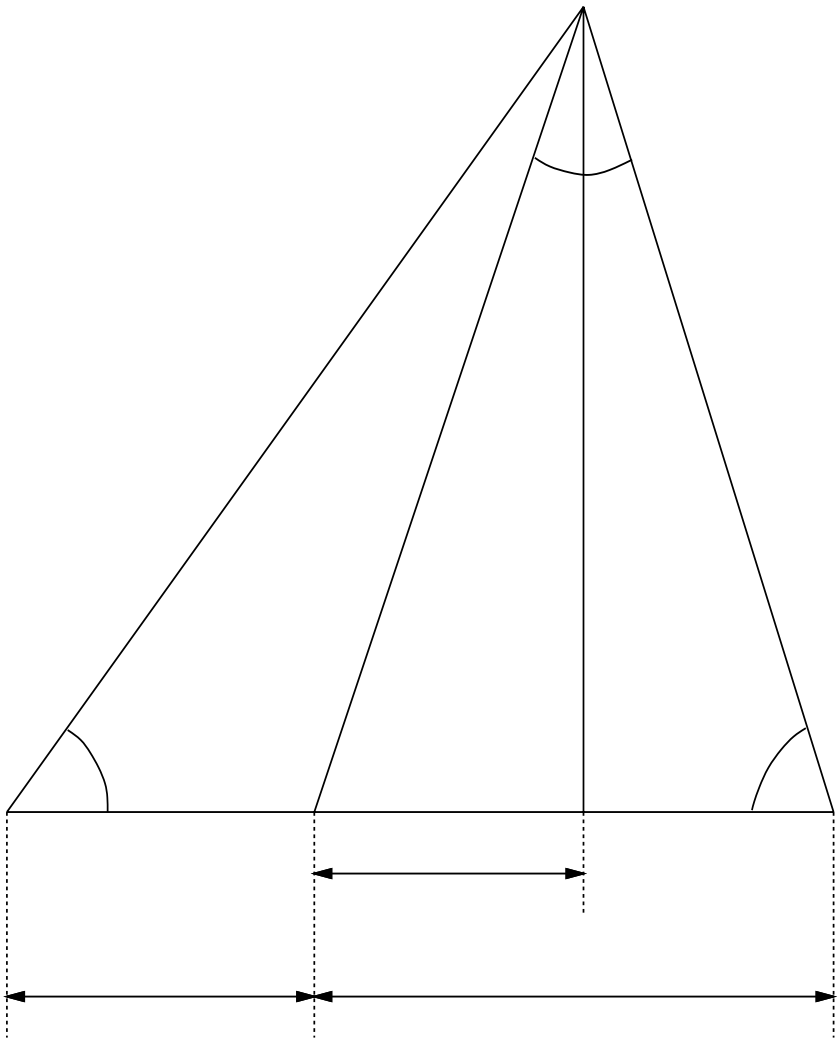

Figure 2.3. Illustration of the lateral distance from the desired directional axis. Here O denotes the vanishing point of the two path boundaries, and OA and OB are the path boundaries with directional angle α and β, respectively. AB is an arbitrary horizontal line so that a triangle OAB is formed. DO is the desired directional axis.

Since both this projection plane and the image plane are perpendicular to the ground plane, their intersection is also perpendicular to the ground plane. That is, the current directional axis in the image is always *the implicit vertical line* passing through the vanishing point in the image. Similarly, the desired directional axis in the image is formed by the intersection of the projection plane of the desired directional axis on the ground and the image plane, which is another line passing through the vanishing point in the image, but not necessarily vertical. Therefore, we can determine if the robot is located on the desired directional axis, in which case the current directional axis coincides with the desired one. Since the desired directional axis in the image can be determined from the 3D distance from the axis on the ground plane to the two path boundaries, we can figure out how far the current robot location is away from the desired directional axis by calculating the offset angle between the desired directional axis and the vertical line.

We now consider the problem of obtaining the lateral distance when the sensor is not at the desired directional axis. Refer to Fig. 2.3, where O denotes the vanishing point of the two path boundaries, and OA and OB are the path boundaries with directional angle α and β, respectively. AB is an arbitrary horizontal line so that a triangle OAB is formed.

Let OD be the desired directional axis in the image whose distances a and b from the path boundaries are known, and OE be the vertical line passing through the vanishing point O, indicating the current *position* of the robot. Then the distance $||DE|| = d$ is the lateral distance to be solved for.

Now, in $\triangle OAD$, we have

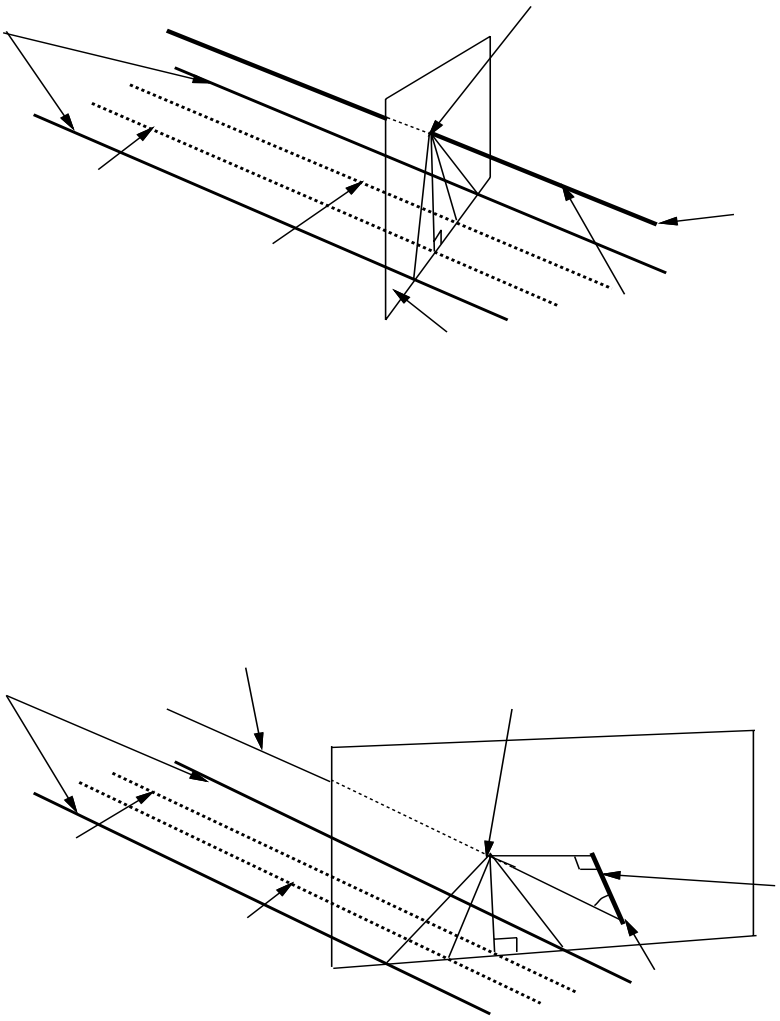

Figure 2.4. Illustration of **Property 2**. Note that to avoid confusion, the projection planes of the desired directional axis and the current directional axis, as well as those of the two path boundaries, are not shown. (a) when the camera orientation angle is 0. (b) when the camera orientation angle is not 0.

$$\frac{\|OD\|}{\sin\alpha} = \frac{a}{\sin(\angle AOD)}$$

Since $\angle AOD = \pi - \alpha - \beta - \gamma$, we have

$$\|OD\| = \frac{\sin\alpha}{\sin(\alpha+\beta+\gamma)}a$$

Similarly, in $\triangle ODB$, we have

$$\|OD\| = \frac{\sin\beta}{\sin\gamma}b$$

Thus, we have

$$\frac{a\sin\alpha}{\sin(\alpha+\beta+\gamma)} = \frac{b\sin\beta}{\sin\gamma}$$

Solving this equation, we have the solution for γ in Eq. 2.3.

Now, in $\triangle ODE$, since

$$\angle DOE = \gamma + \beta - \frac{\pi}{2}$$

we immediately have

$$d = \|OD\|\sin(\angle DOE)$$

By substituting for $\|OD\|$ and $\angle DOE$, Eq. 2.2 is obtained.

Next we show that the verticalness of the directional axis in an image is only dependent on the position of the robot in the environment, and is independent of the

orientation of the camera. To see this, let us assume that the robot now is located at an arbitrary position, say lateral distance d away from the desired directional axis, and with orientation angle θ, as shown in Fig. 2.4(b). Since both the projection plane of the current directional axis and the image plane are still perpendicular to the ground plane, their intersection is also perpendicular to the ground plane. In other words, the current directional axis in the image is still always the vertical line passing through the vanishing point in the image, except that in this case, the vanishing point is not at the center of the image. Similarly, the desired directional axis in the image is a line passing through the vanishing point. Thus, we have the same conclusion: if and only if the robot is located at the desired directional axis, will the desired directional axis in the image domain be vertical.

To obtain the lateral distance in the general case, consider Fig. 2.5, which is derived from Fig. 2.3 by replacing a, b, d with a', b', d'. Note that we have

$$a = a' \cos\theta$$

$$b = b' \cos\theta$$

$$d = d' \cos\theta$$

Substituting these equations into Eq. 2.3 and Eq. 2.2, it is easy to verify that we have very similar closed-form solutions for γ and d. Thus, we have proven **Property 2**.

2.2.3 Looming Distance

In the literature of robot vision, looming refers to the change in size or area of an object induced by motion and is a very useful cue to distance measurement during

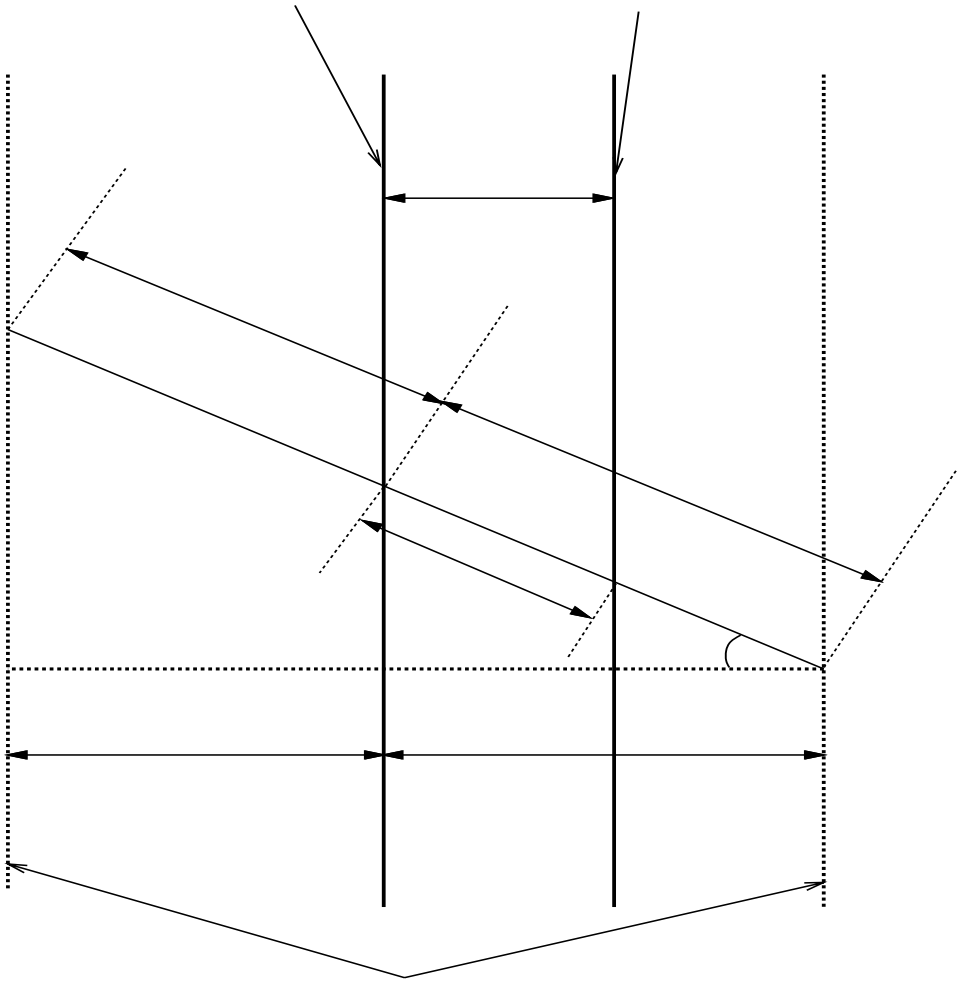

Figure 2.5. The figure used to derive the equation for the lateral distance from the desired directional axis in the general case.

robot motion [5, 57]. The problem of finding the looming distance refers to the determination of robot position along the directional axis from the size of landmark image features. This assumes that the robot has been positioned on the directional axis with the correct orientation as shown in Fig. 2.6(a).

There are many ways to calculate the looming distance. Here we use the "conventional" approach [126]. As illustrated in Fig. 2.6(b), assume that there is a landmark in front of the robot camera, with vertical extent (or "height") h. The image heights of the landmark when the robot is at the desired pose and when the robot is at an arbitrary pose are denoted as δY_t and δY_i, respectively. Let l be the looming distance, and let S_Y be the lens focal scale factor along the Y axis. Then, by similar triangles, we have

$$l = hS_Y(\frac{1}{\delta Y_t} - \frac{1}{\delta Y_i}) \tag{2.4}$$

Note that this approach requires that the landmark should have a "significant" height. Otherwise, there would be a very large error in the computation of l (see Sec. 2.4.3). Fortunately, in many indoor environments, there are landmarks such as doors and other objects which satisfy this criterion.

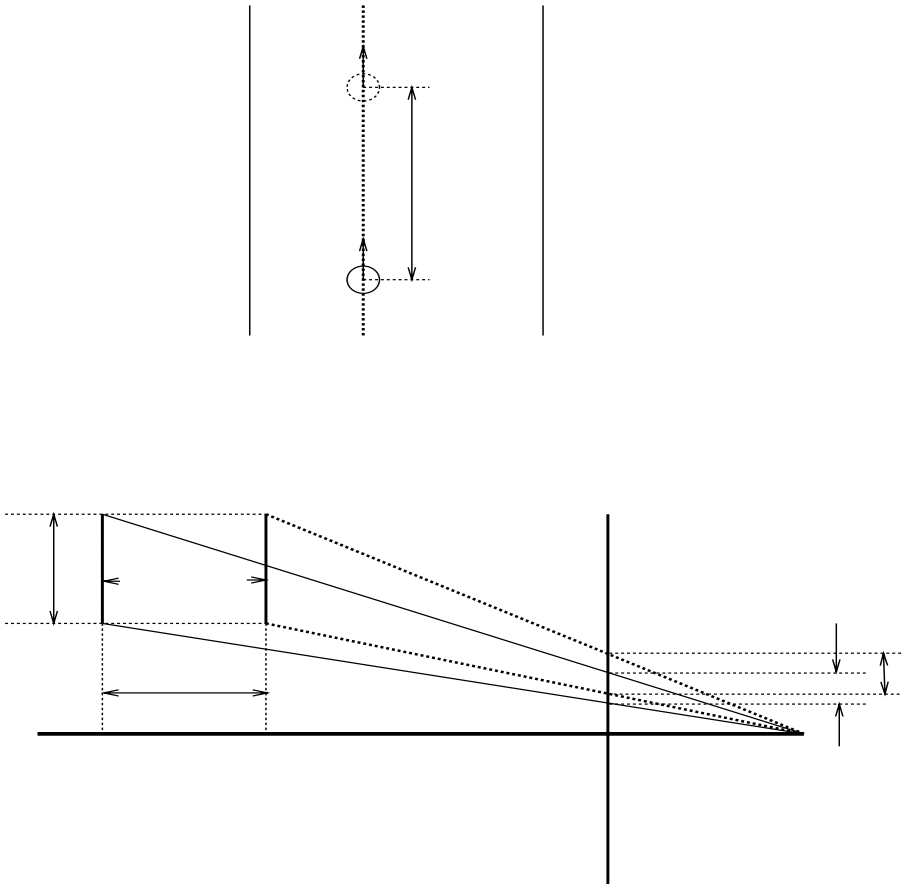

Figure 2.6. Illustration of looming. (a) Analysis of looming (top view). The dashed circle denotes the desired pose, and the solid circle denotes the current pose. The dashed straight line is the desired directional axis (which does not explicitly exist either in the environment or in the image). (b) Calculation of looming distance from side view. δY_i is the measurement of the height of the landmark in the image of the current pose, and δY_t is of the desired pose. Here in order to facilitate illustration, the motion of the camera in distance l is equivalent to the motion of the landmark in the same distance.

2.3 Calibration/Navigation Algorithms

2.3.1 Inference of the Three Geometric Variables from One Image

As stated in Section 2.1, we always use the robot local coordinate system in all the computation, as illustrated in Fig. 2.1, i.e. the robot moves in three degrees of freedom: it moves in lateral direction X, along the directional axis Z, and rotates in the orientation angle θ. By **Property 1** and **Property 2**, we know that orientation and position are independent of each other. Thus, given an arbitrary image, we can compute the orientation angle and the lateral distance with respect to a directional axis simultaneously. In this section, we show that the looming distance can also be computed from the same image, provided that we know *a priori* the "height" of the landmark in the image when the robot is in the desired pose.

Given an arbitrary pose of the robot, and the image plane which is shown as the solid-lined plane in Fig. 2.7(a), the landmark, which has height h, projects into the image plane as a line with length $\delta Y_i'$. Assume that the orientation angle of the camera is θ, which can be independently calculated from Eq. 2.1. If the robot is rotated by $-\theta$ so that the orientation is zero (as shown as the dashed image plane in Fig. 2.7(a)), a new image of the projected landmark, denoted as δY_i, is obtained. If we know δY_i, by applying Eq. 2.4, we can compute the looming distance l immediately.

Now, let us draw a diagram that shows the geometric relationship on the plane formed by the landmark and the focal point, as illustrated in Fig. 2.7(b). It is clear that

$$\frac{\delta Y_i}{\delta Y_I'} = \frac{S_Y}{\frac{S_Y}{\cos\theta}}$$

Hence, we have the following simple relation:

$$\delta Y_i = \delta Y_i' \cos\theta \tag{2.5}$$

Eq. 2.5 shows that by knowing the projected landmark height $\delta Y_i'$ in the image domain at the current pose, it can be transformed into the landmark measurement δY_i in the image domain when the pose orientation is zero. Then by applying Eq. 2.4, the looming distance l can be calculated directly. This completes the claim that the three geometric variables can be inferred from a single image. Given this, we can now develop the appropriate algorithms for the visual servoing problems discussed earlier.

2.3.2 Algorithms

In the automatic calibration problem, a target pose is given. The initial pose is arbitrary. The robot is expected to move from the initial pose to the target pose. In theory, as we showed earlier, one image is sufficient to solve the problem, because once the three geometric variables are calculated from the image, the robot may be actuated according to the three variables to move to the desired pose (e.g. rotate θ, move perpendicular to the directional axis in distance d, then finally move along the directional axis in distance l). In practice, owing to the mechanical error inherent in the operation of a typical robot vehicle (e.g. a robot may actually move 1.45 ft. although it is told to move 1.5 ft.), repeated application of the algorithm (in a feedback

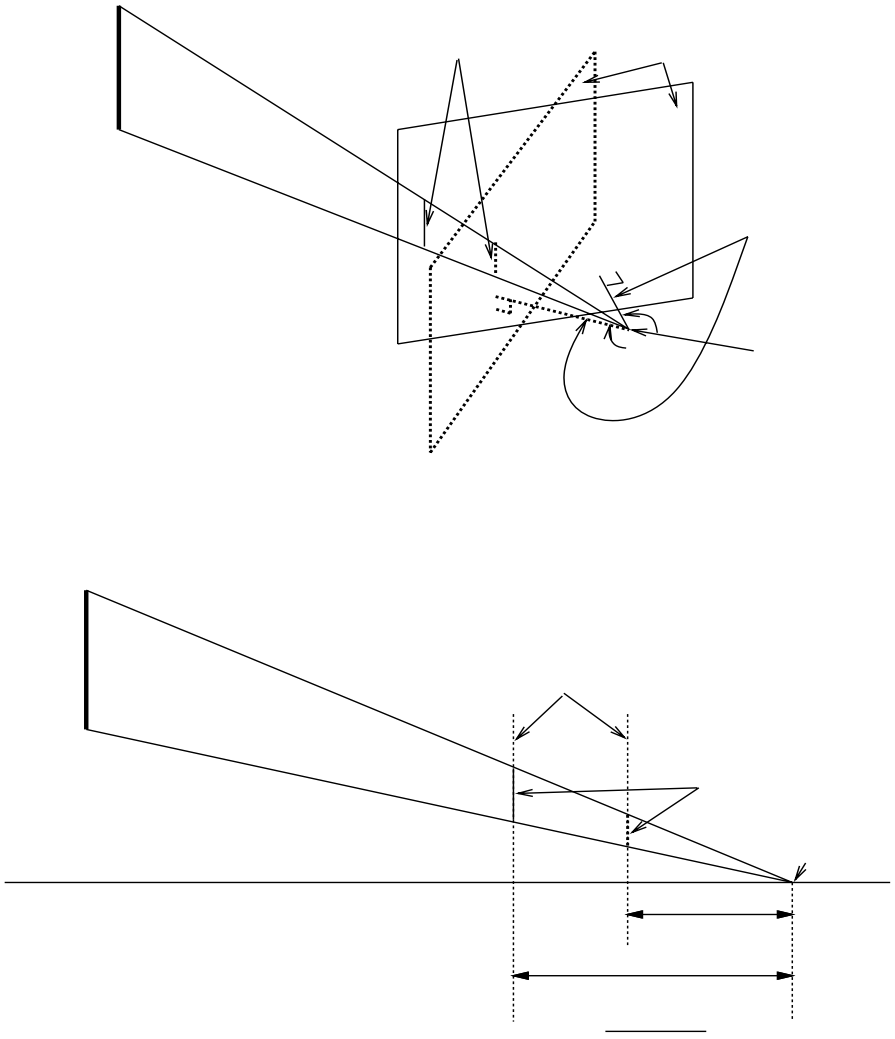

Figure 2.7. Illustration on how to infer the three geometric variables from one image. (a) The imaging system when given an arbitrary pose. The dashed plane is the imaginary image plane after the robot has rotated the angle θ. (b) The relationship between the landmark in the current arbitrary image plane and in the imaginary image plane. Here S_Y stands for the focal length of the camera.

control loop) is usually required. In this approach, the control loop reduces the difference between the desired pose and the robot's current pose until the difference is "small enough" (that is, below a user-specified tolerance threshold). Fig. 2.8 shows the algorithm for automatic calibration (abbreviated as **CAL**).

In the navigation servoing problem, the looming distance is irrelevant. A desired directional axis is given, and the robot is visually controlled to move along the directional axis. Similarly, owing to the mechanical error of the typical robot's operation, at any instant of the navigation course, there is always a difference between the current pose and the orientation and position of the desired directional axis. Thus, a feedback loop control is again necessary to servo the navigation course. The navigation servoing algorithm (abbreviated as **NAV**) is shown in Fig. 2.9.

Hardware/software implementation issues related to these two algorithms are discussed in the experimental results section (Sec. 2.5).

2.4 Error Analysis

In this section, we investigate the stability of the the closed-form solutions for the geometric variables used in the two algorithms with respect to noise or measurement error. The purpose of this is to show that the solutions obtained in this chapter are reliable in the presence of noise. This is also confirmed by the experimental results.

2.4.1 Orientation Angle with respect to the Directional Axis

Eq. 2.1 is obtained under the assumption that the ground plane is flat. In real situations, the ground plane may be locally uneven and the vanishing point might

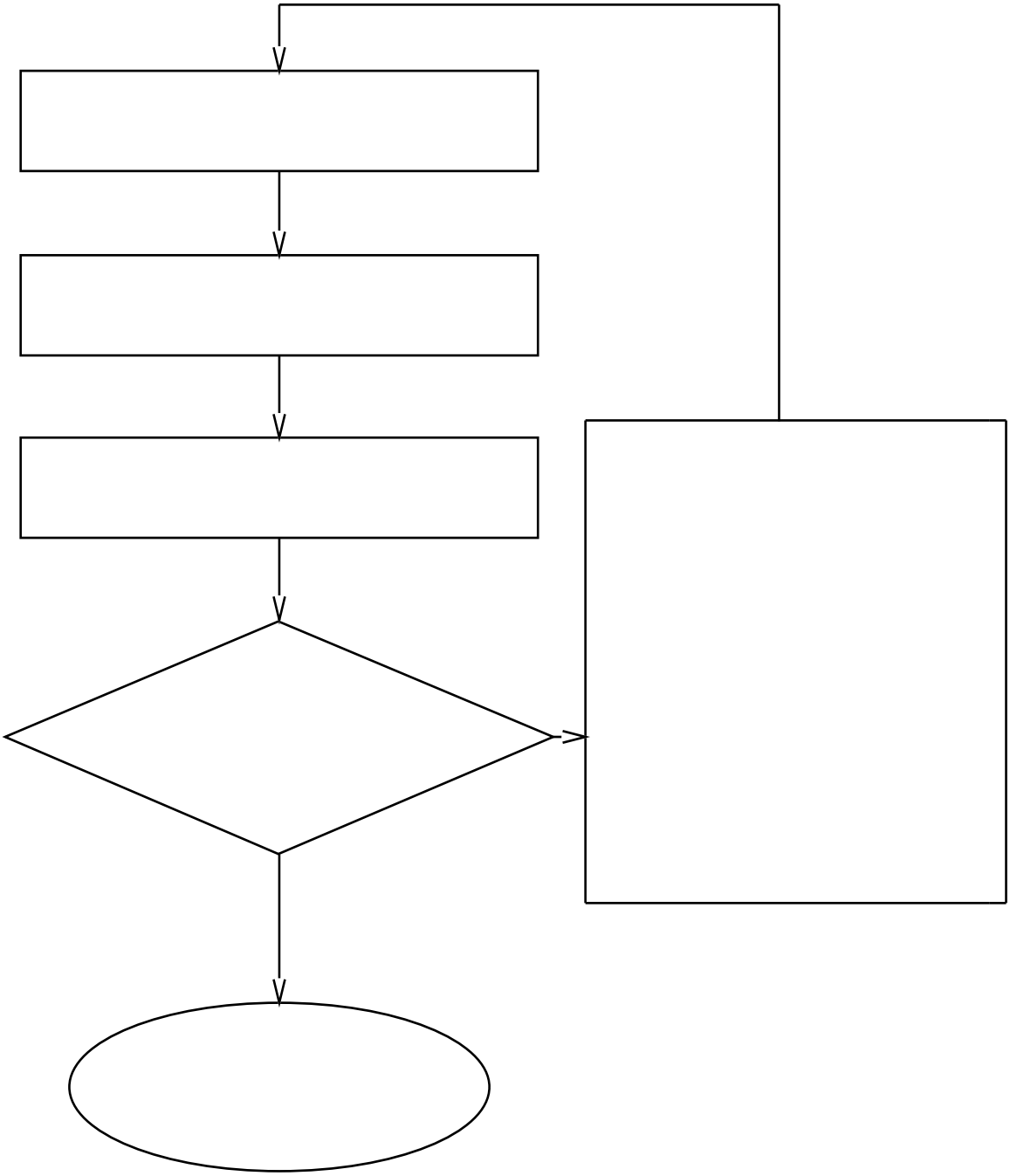

Figure 2.8. **CAL** algorithm.

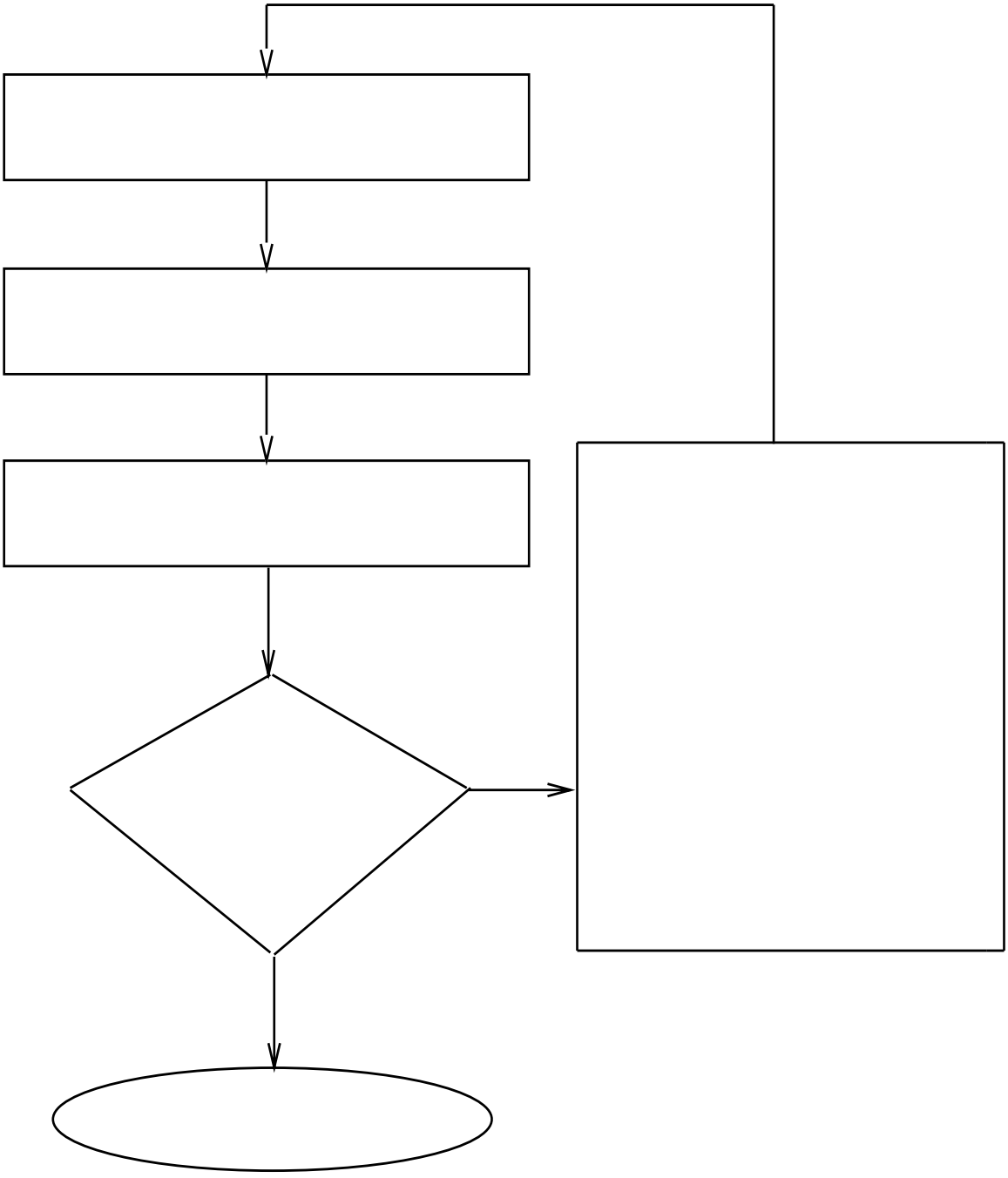

Figure 2.9. **NAV** algorithm.

not be on the horizon bisector line through the principal point. If $||Y_v - Y_c||$ is small, the orientation angle θ can be approximated as:

$$\theta \simeq \arctan \frac{\sqrt{(X_v - X_c)^2 + (Y_v - Y_c)^2}}{f} \tag{2.6}$$

where (X_v, Y_v) and (X_c, Y_c) are the coordinates of the current vanishing point and the principal point, respectively, and f is the focal length of the camera.

Now, we have

$$\triangle\theta = \frac{\partial\theta}{\partial X_v}\triangle X_v + \frac{\partial\theta}{\partial Y_v}\triangle Y_v \tag{2.7}$$

where

$$\frac{\partial\theta}{\partial X_v} = \frac{f}{f^2 + (X_v - X_c)^2 + (Y_v - Y_c)^2}\frac{X_v - X_c}{\sqrt{(X_v - X_c)^2 + (Y_v - Y_c)^2}} \tag{2.8}$$

$$\frac{\partial\theta}{\partial Y_v} = \frac{f}{f^2 + (X_v - X_c)^2 + (Y_v - Y_c)^2}\frac{Y_v - Y_c}{\sqrt{(X_v - X_c)^2 + (Y_v - Y_c)^2}} \tag{2.9}$$

Clearly, $\triangle\theta$ in general is a function of the vehicle geometry. In our specific application scenario, however, although the ground plane is uneven, the fluctuation is very small. This is typically true in all indoor situations. Thus, $|X_v - X_c|$ is normally in the range of 10^2 pixels, and $|Y_v - Y_c|$ is normally in the range of a few pixels. Consequently, $|X_v - X_c| \gg |Y_v - Y_c|$, and $f \simeq S_X = 991.0 \simeq 10^3$. That leads to $\frac{\partial\theta}{\partial X_v} \simeq 10^{-3}$, and $\frac{\partial\theta}{\partial Y_v} \simeq 10^{-5}$. This implies that $\frac{\partial\theta}{\partial X_v}$ plays a dominant role in determining the accuracy of the estimation of θ. Hence, the contribution to the noise

from $\triangle Y_v$, which is caused by the unevenness of the ground plane, can be ignored. This implies that Eq.2.1 can be safely applied to estimate the value of θ even in the case of an uneven ground plane. In order to give an example to show how stable the estimation of the orientation angle is from Eq. 2.1, we assume one pixel perturbation in X_v and Y_v, respectively, i.e., $\triangle X_v = 1$ pixel, $\triangle Y_v = 1$ pixel. Then the resulting error in the calculated orientation angle $\triangle\theta$ is only on the order of 10^{-3} radians.

2.4.2 Lateral Distance from the Directional Axis

Based on Eq. 2.3 and Eq. 2.2, the lateral distance d is determined by the two variables α and β. Therefore,

$$\triangle d = \frac{\partial d}{\partial \alpha}\triangle\alpha + \frac{\partial d}{\partial \beta}\triangle\beta \tag{2.10}$$

where

$$\frac{\partial d}{\partial \alpha} = \frac{\cos\gamma\sin(\gamma+\beta)\cos(\gamma+\beta)+\sin\gamma}{\sin^2\gamma\sin^2(\gamma+\beta)} b\sin\beta\frac{\partial\gamma}{\partial\alpha} \tag{2.11}$$

$$\frac{\partial\gamma}{\partial\alpha} = \frac{b\sin\beta\cos 2(\alpha+\beta) - a\sin\beta}{[a\sin\alpha - b\sin\beta\cos(\alpha+\beta)]^2 + b^2\sin^2\beta\sin^2(\alpha+\beta)} b\sin\beta \tag{2.12}$$

$$\begin{aligned}\frac{\partial d}{\partial\beta} &= \frac{\sin\beta\cos\gamma\sin(\gamma+\beta)\cos(\gamma+\beta)\dfrac{\partial\gamma}{\partial\beta} + \sin\beta\sin\gamma\dfrac{\partial\gamma}{\partial\beta}}{\sin^2\gamma\sin^2(\gamma+\beta)} b \\ &+ \frac{\sin\beta\sin\gamma - \sin\gamma\cos\beta\sin(\gamma+\beta)\cos(\gamma+\beta)}{\sin^2\gamma\sin^2(\gamma+\beta)} b\end{aligned} \tag{2.13}$$

$$\frac{\partial\gamma}{\partial\beta} = \frac{ab\sin\alpha\sin(\alpha+2\beta) - b^2\sin\beta\sin(2\alpha+3\beta)}{[a\sin\alpha - b\sin\beta\cos(\alpha+\beta)]^2 + b^2\sin^2\beta\sin^2(\alpha+\beta)} \tag{2.14}$$

For the typical data in our system, $\frac{\partial d}{\partial\alpha}$ and $\frac{\partial d}{\partial\beta}$ are in the range of $[10^{-3}, 1]$. If the calculation of α and β have 1° error, respectively (i.e. $\triangle\alpha$ and $\triangle\beta$ are on the

order of 10^{-2} radian), then the resulting range of error in the lateral distance from the directional axis is in the range of $[10^{-5}, 10^{-2}]$ ft. Although in real situations there may be other error sources that have not been considered here, and the error model may be more complicated, in our experiments (see Sec. 2.5), the actual error in the distance never exceeds 0.1 ft.

2.4.3 Looming Distance along the Directional Axis

From Eq. 2.4, we know that h, S_Y, and δY_t are constants. Thus, the looming distance l is only a function of the variable δY_i. Since

$$\frac{dl}{dY_i} = \frac{hS_Y}{(\delta Y_i)^2}$$

we have

$$\triangle l = \frac{hS_Y}{(\delta Y_i)^2} \triangle(\delta Y_i) \tag{2.15}$$

In our experiments, we used the hallway door as the reference landmark, (see Sec. 2.5). The height of the door is $h = 8.05$ ft. The camera focal scale factor along the Y axis is $S_Y = 1209.658$, measured in pixels. In the experiments, δY_i is usually around 300 pixels. Therefore, the typical value for $\frac{dl}{d\delta Y_i}$ is around 10^{-1}. Then, for one pixel error in the measurement of δY_i, i.e., $\triangle(\delta Y_i) = 1$, the resulting error in the looming distance could be up to 10^{-1} ft. Clearly, if subpixel measurement is used, the accuracy of the estimation of the looming distance may be increased proportionally. For instance, if $\triangle Y_i = 0.1$, then $\triangle l$ will be in the range of 10^{-2} ft. On the other hand, the accuracy of the looming distance is also inversely proportional to the square of

the measurement of the landmark height in the image domain. This means in order to keep a reasonable accuracy, the landmark height should be reasonably large in the image. Fortunately, since looming distance is only used in the automatic calibration problem (which is done once at robot start-up), a significantly large landmark can be placed in the vicinity of the robot such that the projected image "height" of this landmark will be large enough. With this requirement satisfied, even if subpixel measurement is not used, reasonable accuracy can still be achieved. This is shown in the experiments in Sec. 2.5.

2.4.4 Summary of the Error Analysis

Based on the analysis in the previous subsections, the calculation of the orientation angle θ has the highest stability, whereas the calculation of the looming distance l has the lowest, with the calculation of the lateral distance d lying between the two. The experimental results show that the calculations of θ and d are typically very accurate. Only the estimation of l has noticeable error, usually 0.1 ft. or so for the particular geometry used in the experiments. But even so, the relative deviation is still within 1% (see Sec. 2.5).

2.5 Experimental Results

2.5.1 CAL

We implemented the **CAL** algorithm first in LISP on a Sun IV SPARC workstation. A CCD camera was mounted on top of a Denning mobile robot in such a way

Table 2.1. Desired pose parameters and camera intrinsic parameters

X_c	Y_c	d	δY_t	S_X	S_Y
255.5	240.2	0.0	245.0	991.0	1209.7

that the camera focal center coincided with the center of rotation of the robot. Thus, the robot camera has three degrees of freedom: one rotation, and two translation. Black and white images acquired by the camera were digitized by a DigiMax system at a resolution of 512 by 480 pixels, with 8 bits per pixel.

The system was tested in a hallway environment, which satisfies the structured environment requirement. Fig. 2.10 is an image of the landmark (the hallway door) and the path boundaries (the baseboard) taken when the robot is at the desired pose. Note that the black/white target on the bottom of the door is not used during automatic calibration. It *is* used to check if the algorithm has worked correctly after the robot has been automatically calibrated. In other words, if the algorithm works correctly, once the robot has been calibrated, the orientation angle should be zero. In this case, the center of the cross should be on the center vertical line in the image. Table 2.1 lists the geometric parameters when the robot is correctly calibrated, i.e. it is in the desired pose, as well as the intrinsic parameters of the CCD camera. Table 2.2 gives the thresholds used to terminate the algorithm.

The key to the success of the system is the robust detection of the two path boundaries. Fortunately, in the experimental hallway environment, the two black baseboard lines have very strong contrast against the white wall, and thus are very easy to detect. In order to obtain a real-time performance, a simple Roberts operator

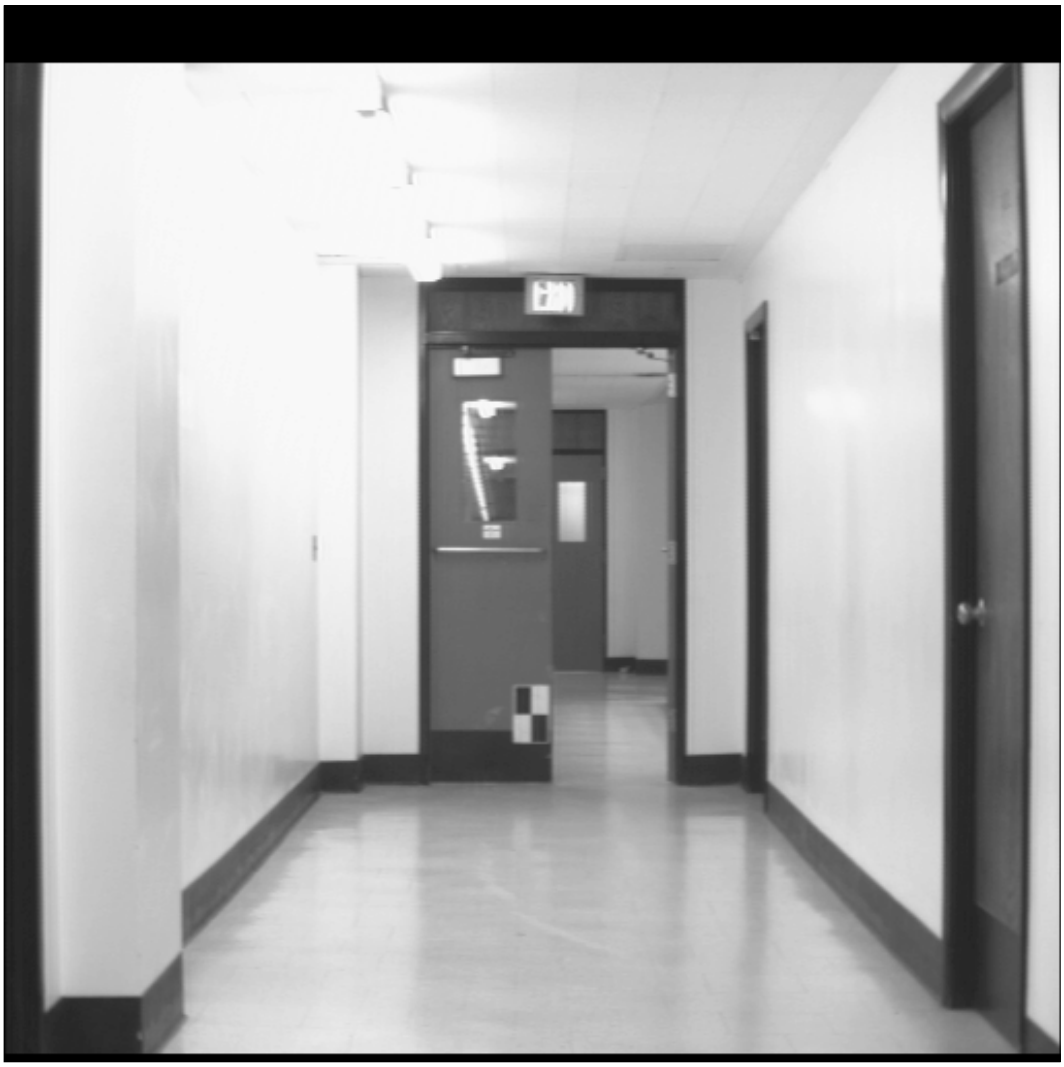

Figure 2.10. The view from the robot when located at the desired pose. Note that the black/white target on the bottom of the door is not used during the operation of the automatic calibration. It is used to check if the algorithm has worked correctly after the robot has been automatically calibrated.

Table 2.2. Thresholds

θ	d	l
0.02°	0.1 ft.	0.1 ft.

is used to detect all the baseboard edge pixels, and then a Least-Mean-Square technique (Linear Regression) is used to fit a straight line to each side. Once the two lines are obtained, the coordinates of the vanishing point are simply calculated by finding the intersection point of the two lines. We compared the vanishing point obtained by using this simple method with that obtained by using Collins *et al* [13]'s algorithms, and found that the differences were always within half a pixel. That means the simple method works very well in this particular environment.

Table 2.3 records the results of executing **CAL** in the hallway environment. The desired pose was defined as a pose which was located 41.1 ft. away from the door, 4.0 ft. from the left wall, and 4.13 ft. from the right wall, and oriented parallel to the baseboard direction. In this experiment, the hallway door was used as a landmark to infer the looming distance [126]. At the initial pose, the robot was oriented about 4° off to the right of the baseboard orientation (i.e. the desired orientation), and located about 5.3 ft. from the left wall (i.e. about -1.4 ft. lateral distance (X direction) from the directional axis), and about 5 ft. away from the desired pose on the directional axix (i.e. -5 ft. in Z direction). Fig. 2.11 shows the image taken at this initial pose. After the termination of the algorithm, the robot was in a pose very close to that of the target. The final actual lateral distance and the final actual looming distance were both within 0.05 ft. of the estimated values calculated by the algorithm, as shown in Table 2.3.[2] Fig. 2.12 is a schematic diagram of the execution of the algorithm. Based on the results of this experiment, it can be seen that even with this large difference between the initial pose and the desired one, it only took four iterations to

[2]Since the robot is not a perfectly round object, it is difficult to measure the coordinate of its center with a precision better than 0.05 ft.

Table 2.3. A recording of an execution using **CAL**.

loop #	X_c	θ	d	l	Action
1	145.8	$-6.28°$	-1.23 ft.	-5.50 ft.	Turn left 6.28° Move left 1.23 ft. Move Backward 5.50 ft.
2	252.2	$-0.191°$	0.033 ft.	-0.163 ft.	Turn left 0.191° Move Backward 0.163 ft.
3	253.8	$-0.102°$	0.042 ft.	0.0	Turn Left 0.102°
4	255.7	0.009°	0.043 ft.	0.0	Stop

finally calibrate the robot pose. Note that the very first iteration corrected most of the difference, which shows that the **CAL** algorithm *is* able to determine the three variables simultaneously from one image. The next three iterations are basically for refinement of the robot mechanical error incurred during execution of the first step.

2.5.2 NAV

NAV was modified from **CAL**, and was implemented in C in the Khoros environment [39, 40, 41]. Khoros has advantages of a unified parameter format, and it is easy to modify parameters without recompiling. Fig. 2.13 shows **NAV**'s dataflow diagram. All the modules are briefly described below:

- **inirobot** Initializes the Denning mobile robot to make sure it is ready to move.
- **grabplane** Invokes CCD camera and DigiMax to take an image and digitize it.
- **linefind** In order to obtain the two path boundaries more robustly, a *Fast Line Finder* module was used to replace the simple LMS based line detector

Figure 2.11. A hallway view from robot in an initial pose of an experiment.

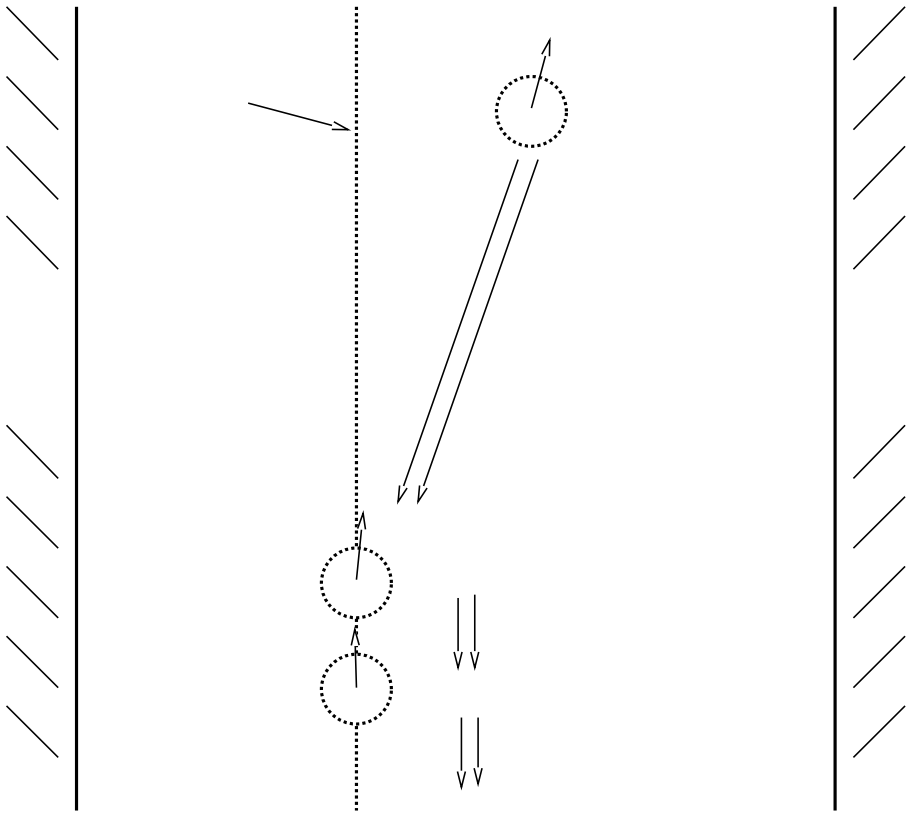

Figure 2.12. A diagram showing the scenario of an experiment.

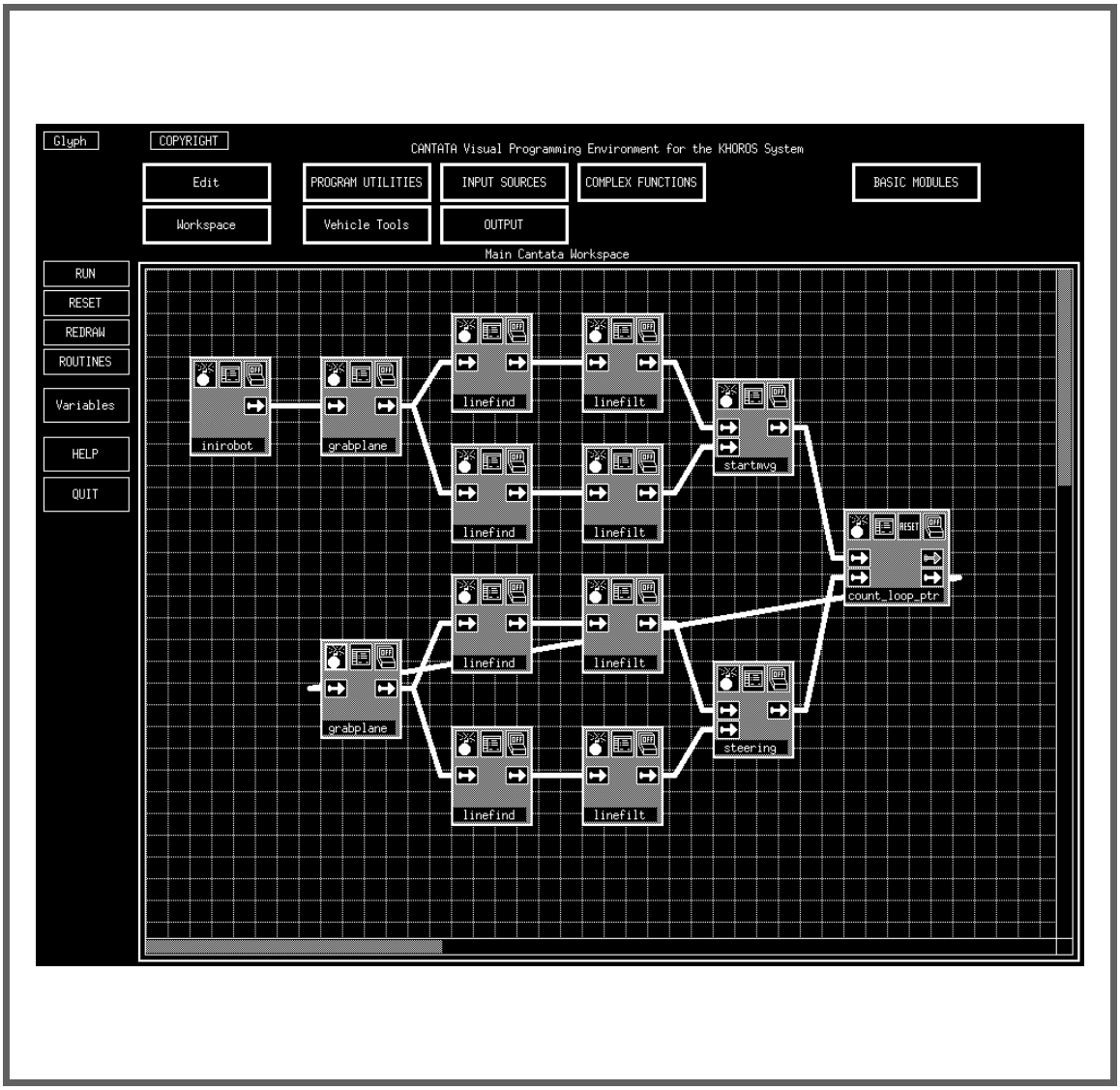

Figure 2.13. Dataflow of the implementation of **NAV**.

implemented in **CAL**. The fast line finder [63] groups edges into line segments based on certain geometric criteria, such as length and orientation.

- **linefilt** Filters out all the other line segments such that only the path boundaries are left, by using *a priori* knowledge and assumptions. Specifically, we know the orientation range of the path boundaries, and we know that the longest lines within this range must be a path boundary.

- **startmvg** Computes the vanishing point and lateral distance based on the detected path boundaries, and then starts the robot moving according to these two steering parameters.

- **count_loop_ptr** This is a control module. A termination condition is set in this module so that the robot stops moving when this condition is satisfied. In our current implementation, we set the total length of the navigation course as the termination condition.

- **steering** Computes the vanishing point and lateral distance based on detected path boundaries, and then steers the robot to perform continuous motion. The implementation of **steering** is more complicated than that of **startmvg**. **startmvg** handles the case of start-up from a stationary position. Thus, the pose information estimated at this time *is* the robot's current pose information. When in **steering**, however, since the robot is always in continuous motion, after the pose is estimated, it no longer represents the current pose information. The design of the control structure of the **steering** module needs to take this dynamic

change into account by applying classic automatic control theory [35] with the orientation angle and the lateral distance as the two feedback variables.

In order to achieve real-time performance, the two path boundaries are detected in parallel. That is why two pairs of **linefind** and **linefilt** work together with each image obtained from **grabplane**; each pair of modules feeds the path boundaries detected to **startmvg** or **steering**, as seen in Fig. 2.13.

The whole system gives a real-time performance, at the speed of 0.2 ft. per second [131].

2.6 Summary of Visual Servoing Control

In this chapter, we explored and used geometric variables in a structured environment. By a structured environment, we mean that there is a path defined *a priori* which is bounded by two parallel boundaries. Based on this assumption, we have shown that there is very rich geometric information that can be used for visual servoing. Specifically, we have used orientation angle, lateral distance, and looming distance as geometric variables. The estimation of these variables are based on vanishing points and directional axes. A visual servoing control algorithm was developed based on these geometric variables, and this algorithm was applied to solving automatic calibration and navigation servoing problems in a real time system. The two systems have been implemented on a Denning Mobile Robot, and have been studied in a set of experiments. A theoretical analysis was used to show that the estimations of the geometric variables are reasonably stable in the presence of measurement

noise. The results of the theoretical analysis were confirmed by the experimental results, which also show that the geometric variable estimation procedure is reasonably stable and robust.

CHAPTER 3

QUALITATIVE AND QUANTITATIVE OBSTACLE DETECTION

3.1 Introduction

In the previous chapter, algorithms were developed which control the motion of a vehicle from a given position and orientation (vehicle pose) to a desired target position and orientation. It was implicitly assumed that the vehicle path was clean — that is, it did not contain obstacles. In real world situations, however, this assumption cannot be made. Static unmodelled objects may exist in the environment and mobile objects may move into the path of the vehicle. Hence, obstacle detection (and ultimately avoidance) is an important issue in mobile robotics.

Before we proceed to propose any obstacle detection algorithms, there are two important questions that need to be answered. The first question is how to define an obstacle. Intuitively, anything that obstructs the motion of a vehicle is an obstacle. However, forming a precise definition is surprisingly difficult. If the ground were a perfect plane, then an obstacle point (as measured by feature points on the obstacle) could be defined as any point perpendicular to the ground plane greater than some fixed value (which depends on the capabilities of the vehicle). It could be either a

"normal" obstacle if it is higher than the ground plane, or an indentation or hole if it is lower than the ground plane. A more complete analysis would involve determining the extent of the obstacle in the plane (needed for path planning), its shape, and possibly its motion; none of these are considered here; rather, we only focus on the case in which static obstacles can be represented and characterized by their feature points.

In general, there will be variations in the ground height occuring at different scales. At the smallest scale, there are bumps and indentations which are on the order of a few inches high. At intermediate scales, the road may have a crown, may be banked, or may have large bumps, and at large scales, there are hills and long ramps. Depending on the environment and the vehicle, an obstacle detection algorithm may need to handle all of these variations.

The second question is the speed at which it is necessary to detect obstacles. Conventional obstacle detection algorithms are based on complete 3D reconstruction, either directly (e.g. through range data [18, 79, 80]), or indirectly (e.g. through computing optical flow maps [23, 86]). We believe that this is unnecessary because typically obstacles are rare events. That is, during most of the computation cycles underlying a navigation process, there are actually no obstacles. Thus, in each computation cycle, instead of generating a complete 3D reconstruction, it may be sufficient to compute whether or not there are obstacles in the path of the vehicle. This is what we call *qualitative obstacle detection*, because the reconstruction technique determines only whether or not an obstacle is present. It accomplishes this by determining whether or not a set of points in front of the vehicle can be approximated

by a plane (known or unknown); points not on the plane are considered potential obstacles. Clearly, qualitative obstacle detection methods should keep computational expense to a minimum.

This qualitative approach to obstacle detection is particularly applicable when the ground plane is locally flat, so that a mathematical plane (known or unknown) can be approximated. If the ground has large variations, this qualitative approach may not be appropriate. In this case, the height of these variations becomes the more relevant information for obstacle detection. Even in this case, it may not be necessary to recover the complete 3D reconstruction. Instead, we only need a more quantitative algorithm to compute the height map of the ground plane, which is the minimum level of information needed to detect obstacles in this case.

In this chapter, we propose three algorithms for obstacle detection. The first two algorithms are qualitative in the sense that they return only yes/no answers to the question of whether or not one or more obstacles are present; they do not recover 3D structure. The third algorithm is quantitative because it recovers partial 3D structure — height information. The three algorithms are compared with respect to robustness to noise.

One of the questions that is examined in this chapter is the role of knowledge in detecting obstacles. The three different algorithms make use of different knowledge of the visual sensors and the environment. Using knowledge can usually improve the robustness of a system provided the knowledge is accurate. The first algorithm assumes that the ground is planar and its equation is known (**KGP**). Using this information together with the intrinsic and extrinsic parameters of the cameras, a linear system of equations is derived for the relationship of the ground plane to the sensor. The

existence of a point not on the known ground plane (i.e. a potential obstacle) implies that the system is not solvable. One of the drawbacks of this algorithm is that errors in the equation of the ground plane due to camera tilt, grades, or hills in the road, as well as noise in image measurements, can cause the algorithm to fail. The second algorithm does not use quantitative knowledge about the environment. It assumes that the ground is planar, but its equation as well as the camera parameters are unknown (**UGP**). This algorithm also reduces the obstacle detection problem to determining the solvability of a linear system. The third algorithm adaptively updates its knowledge of the environment by estimating the ground plane over a sequence of partially calibrated stereo pairs (**EGP**). This algorithm estimates the height above the ground plane for all points in a region of interest in the image. Faugeras [24] has shown that from uncalibrated stereo one can recover 3D structure up to a family of projective transformations. In this chapter we show that it is possible to recover partial metric information from partial calibration.

From the experiments with real and simulated data which are presented in Section 3.5, it is seen that the adaptive algorithm is the most robust with respect to noise. However, there is still potential value in using the other algorithms. One of the limitations on the speed of a mobile robot is how fast it can safely detect obstacles given the available resources. These resources can be computational (how long it takes to run the algorithm) and sensor hardware (which cameras or other sensors are devoted to this task and for how long). The third algorithm requires stereo processing and computes the heights of each of the feature points, while the first two algorithms may use monocular views and compute a single statistic for all of the points combined. In each of the algorithms, we assume that the correspondences have been computed.

KGP and **UGP** are typically suitable to navigation in an indoor environment in which the ground plane is locally flat, and can be approximated by a mathematical plane. Since **KGP** and **UGP** both can be applied in either monocular motion or stereo, depending on whether or not there is a stereo pair of cameras available in the robot, qualitative obstacle detection can be accomplished either from two consecutive monocular views or from one stereo view.

If information about camera geometry is available (such as the camera orientation w.r.t. the ground plane, and the camera height from the ground plane), then **KGP** is the ideal algorithm to apply, because the ground plane equation can be derived from the geometric information. Note that the assumption that the parameters of the camera geometry are known may be valid in many situations of indoor robot navigation, or even in outdoor vehicle navigation. In case the parameters of the physical set-up are not available, or it is very difficult to measure those parameters, the **UGP** algorithm may be applied instead of **KGP**. In either case, the robot can quickly determine whether or not there are obstacles in the scene by applying **KGP** or **UGP**. When obstacles are detected, the robot may slow down or even completely stop, and then apply more sophisticated algorithms to locate where the obstacles are. This can be followed by path planning in order to avoid them. This is a very reasonable scenario even for human navigation. One could imagine that when a human is driving on a highway where presumably there should be no obstacles, the speed can be set to a reasonably high value. On the other hand, when driving on a narrow city street full of potential obstacles, speed must be relatively low and it may be necessary to completely stop occasionally to safely avoid obstacles and to plan paths around them.

While **KGP** and **UGP** both assume a locally flat ground plane, which is satisfied in most indoor environments, this constraint may not be valid in outdoor scenarios, especially when a vehicle is navigating over a rough terrain. In this case, the ground plane cannot be approximated as a mathemetical plane, and any "small" variations of the ground plane should be considered as part of the ground plane *per se*. Given this consideration, **EGP** is developed. During navigation, the ground plane is constantly estimated and updated based on previous estimates using Kalman Filtering techniques, and obtacles are detected only if they are significantly different in height from the updated ground plane threshold. Again, it is not necessary to recover complete 3D information for obstacle detection. Instead, **EGP** only computes the 3D height information. Consequently, computation can still be saved in each navigation cycle, and the vehicle navigation speed can be expedited.

Because of its practical impact, obstacle detection has received widespread attention in the research literature [18, 19, 23, 86, 105, 106, 128, 36, 79, 80]. Most existing algorithms for detecting obstacles use active range sensors, stereo, or optical flow. For example, Nelson and Aloimonos [86] used flow field divergence for obstacle detection and avoidance in visual navigation. Daily *et al* [18, 19] used a laser range sensor to detect obstacles. Enkelmann [23] approached this problem by evaluating the difference between the calculated optical flow and the predicted model-based flow. Young *et al* [128] developed another approach to obstacle detection based on the difference between the reference flow and the observed flow. The difference between this method and that of Enkelmann's is that, instead of detecting flow over the whole image, Young *et al* developed a localized scheme for detecting flow along a line, enabling fast and

parallel computation. Both of Enkelmann's and Young *et al*'s methods assume that the vehicle motion is pure translation, which may not be true in a real environment.

Recently, Matthies and Grandjean [36, 79, 80] addressed the problem of real-time vehicle navigation by using stereo maps. Like us, they assume all the obstacles in a scene are near-vertical, i.e. no consideration is given to the slope of an obstacle. However, they thresholded in the range image domain instead of in 3D space (as done here). In general, the error in range will be much greater than the error parallel to the image plane. For example, for a camera with a very small angle of elevation, one could measure the height very accurately and still have large errors in depth. Thus, thresholding on depth can produce more false positives and negatives than for height.

This chapter is based on Zhang *et al* [133] and is organized as follows. The three obstacle detection algorithms are presented in the next three sections. In Section 3.5, the sensitivity of the algorithms under different levels of noise error is investigated, and results on real images are presented. The chapter concludes with a summary.

3.2 Obstacle Detection with a Known Ground Plane (KGP)

In this section, we assume that the ground plane equation with respect to the first camera is known, the rotation between the first and the second camera is small, and the intrinsic calibration of the camera is known. This algorithm can be applied to stereo images, where the first and second images are the left and right images of the stereo pair. It can also be applied to a motion sequence where the two images

are taken at different time instants from a monocular camera in motion. We call this algorithm the **known ground plane** algorithm (**KGP**).

Given an arbitrary 3D ground plane point $\boldsymbol{P}$, with the small rotation assumption specified above, the velocity/displacement of $\boldsymbol{P}$ can be expressed as:

$$\dot{\boldsymbol{P}} = \boldsymbol{\Omega} \times \boldsymbol{P} + \boldsymbol{t} \tag{3.1}$$

where $\dot{\boldsymbol{P}}$ represents the velocity of $\boldsymbol{P}$, $\boldsymbol{\Omega}$ denotes the rotation angle vector, and $\boldsymbol{t}$ is the translation vector.

Given the intrinsic camera parameters, the mapping relationship between the 3D world and a 2D image can be modeled as a calibration transformation followed by a pin-hole camera projection. By projective geometry, the relationship between the 3D point $\boldsymbol{P}$ and its corresponding 2D image point $\boldsymbol{p}$ is:

$$x = \frac{X}{Z}, \quad y = \frac{Y}{Z} \tag{3.2}$$

Substituting these equations into Eq.3.1, together with the ground plane equation yields:

$$k_X X + k_Y Y + k_Z Z = 1, \tag{3.3}$$

from which the following matrix equation of the flow of point $\boldsymbol{p}$ can be obtained:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \begin{pmatrix} -xy & 1+x^2 & -y & K & 0 & -xK \\ -(1+y^2) & xy & x & 0 & K & -yK \end{pmatrix} \begin{pmatrix} \Omega_X \\ \Omega_Y \\ \Omega_Z \\ t_X \\ t_Y \\ t_Z \end{pmatrix} \tag{3.4}$$

where $K = k_X x + k_Y y + k_Z$. The above relation can be abbreviated as:

$$\dot{\boldsymbol{p}} = \boldsymbol{H}\boldsymbol{M} \tag{3.5}$$

Here $\boldsymbol{H}$ is a matrix linear function, which transforms a motion parameter vector $\boldsymbol{M}$ into a flow vector for $\boldsymbol{p}$, when the image of a ground plane point is given. Thus, if there are n ground plane points $\boldsymbol{p_1}, ..., \boldsymbol{p}_n$ in the image plane, then

$$\begin{pmatrix} \dot{\boldsymbol{p}}_1 \\ . \\ . \\ . \\ \dot{\boldsymbol{p}}_n \end{pmatrix} = \begin{pmatrix} \boldsymbol{H_1} \\ . \\ . \\ . \\ \boldsymbol{H}_n \end{pmatrix} \boldsymbol{M} \tag{3.6}$$

Let

$$\boldsymbol{D} = \begin{pmatrix} \boldsymbol{H_1} \\ . \\ . \\ . \\ \boldsymbol{H}_n \end{pmatrix}$$

$$\boldsymbol{b} = \begin{pmatrix} \dot{\boldsymbol{p}}_1 \\ . \\ . \\ . \\ \dot{\boldsymbol{p}}_n \end{pmatrix}$$

Eq.3.6 can then be written as:

$$\boldsymbol{DM} = \boldsymbol{b} \tag{3.7}$$

It is assumed throughout the chapter that there are at least three ground plane points. An obstacle point is one which is more than some fixed distance above the ground plane. For an obstacle point in the first image, there must be a corresponding ground plane point which shares the same line of sight, but is further away. These two points will have distinct locations in the second image unless the motion is a pure rotation around the line of sight of the point in the first image and/or a pure translation along the line of sight (i.e. the FOE coincides with this point). In either case, the flow value of the point under consideration will be zero, which is easily detected (other points will have non-zero flow values). Assuming that all such zero-flow points have been eliminated, then each obstacle point will have a different flow value from the corresponding ground plane point on that projection ray. Now suppose we partition the point set into two subsets: one composed of ground plane points, and the other of obstacle points. Under this assumption, Eq.3.7 becomes

$$\begin{pmatrix} \boldsymbol{D_1} \\ \boldsymbol{D_2} \end{pmatrix} \boldsymbol{M} = \begin{pmatrix} \boldsymbol{b_1} \\ \boldsymbol{b_2} \end{pmatrix} \tag{3.8}$$

where the subsystem with subscript 1 denotes ground plane points, and the subsystem with subscript 2 denotes obstacle points. Now if we replace the obstacle points with their corresponding ground plane points which share the same lines of sight with them, another linear system can be obtained:

$$\begin{pmatrix} \boldsymbol{D_1} \\ \boldsymbol{D_2} \end{pmatrix} \boldsymbol{M'} = \begin{pmatrix} \boldsymbol{b_1} \\ \boldsymbol{b_2}' \end{pmatrix} \tag{3.9}$$

If we assume that the ground plane points are non-collinear, then the linear system in Eq. 3.7 has a unique solution using only the ground plane points. This is a subsystem of both Eq.3.8 and Eq.3.9. If they both have solutions, then the linear systems must be the same. However, since $\boldsymbol{b_2} \neq \boldsymbol{b_2}'$, it follows that the two linear systems in Eq.3.8 and Eq.3.9 cannot be satisfied at the same time.

A linear system like Eq.3.7 has a solution *iff* Rank($\boldsymbol{D}$) = Rank($\boldsymbol{Db}$). Thus, we have proved the following proposition:

Proposition: *The n points are all ground plane points iff the linear system Eq.3.7 has a solution; that is, iff Rank($\boldsymbol{D}$) = Rank($\boldsymbol{Db}$).*

A practical problem with this proposition is how to calculate the rank of these matrices in the presence of noise. The rank of a matrix is equal to the number of non-zero singular values in its singular value decomposition (SVD) [92]. To test if the two matrices have the same number of non-zero singular values, the ratios of the appropriate corresponding singular values are used. We always assume that the number of feature points in a scene is no less than 3 and they are not collinear. It can be easily shown that if there are at least three non-collinear points, the matrix $\boldsymbol{D}$ will be full rank, i.e. Rank($\boldsymbol{D}$) = 6, so none of the singular values will be zero. If there is at least one obstacle point, Rank($\boldsymbol{Db}$) = 7, which means the linear system Eq.3.7 is inconsistent. Thus, the problem in this case is to decide whether or not the smallest singular value of $\boldsymbol{Db}$ is zero. However, since the real data are always corrupted with some degree of noise, together with the problems of numerical processing, the

computed singular values are almost always non-zero. Let $\sigma_{min}(\boldsymbol{D})$ be the smallest singular value of matrix $\boldsymbol{D}$, and $\sigma_{min}(\boldsymbol{Db})$ be the smallest singular value of matrix $[\boldsymbol{Db}]$. A feasible criterion for determining if Rank($\boldsymbol{D}$) = Rank($\boldsymbol{Db}$) is to check if the ratio $\frac{\sigma_{min}(\boldsymbol{D})}{\sigma_{min}(\boldsymbol{Db})}$ is sufficiently large. Based on our simulation analysis under different noise levels, and our experimental results on real images in Section 3.5, a threshold δ between 5 to 10 on this ratio value is sufficient to detect obstacle points 1 ft. or more off the ground plane at a distance of 20 ft.. Note that this criterion is the most conservative one, since a singular value of a matrix is always less than or equal to the corresponding singular value of its augmented matrix [34]. Let $\lambda_1 \geq \lambda_2 \geq ...\lambda_6$ be the six singular values of matrix $\boldsymbol{D}$, and $\eta_1 \geq \eta_2 \geq ...\eta_7$ be the seven singular values of the augmented matrix ($\boldsymbol{Db}$). Hence, we have

$$\eta_1 \geq \lambda_1 \geq \eta_2 \geq \lambda_2 \geq ...\lambda_6 \geq \eta_7$$

Therefore, the value of $\frac{\lambda_6}{\eta_7}$ is the most conservative criterion for detecting if the two matrices, $\boldsymbol{D}$ and ($\boldsymbol{Db}$), have rank 6.

Based on the above analysis, the basic steps of the algorithm are:

For each set of image correspondences (either stereo pair or motion pair),

- build the linear system defined in Eq.3.7;
- compute the singular values of matrices $\boldsymbol{D}$ and $[\boldsymbol{Db}]$;
- if $\frac{\sigma_{min}(\boldsymbol{D})}{\sigma_{min}(\boldsymbol{Db})} > \delta$, no obstacle is detected; else, report obstacle;

In the very rare case that all the n points are collinear in 3D, the above algorithm could be modified. In this case, matrix $\boldsymbol{D}$ would be singular, and Rank($\boldsymbol{D}$) =

Rank($\boldsymbol{Db}$) $= i < 6$ iff the n points lie on a line on the ground plane. If there are obstacle points, then Rank($\boldsymbol{Db}$) $= i+1$. Thus, instead of using the ratio between the smallest singular values of $\boldsymbol{D}$ and $[\boldsymbol{Db}]$, one would first determine i, then determine the consistency of the linear system by checking $\frac{\sigma_i(\boldsymbol{D})}{\sigma_{i+1}(\boldsymbol{Db})} > \delta$.

This algorithm assumes that the internal calibration is known, the ground plane is locally flat, and the plane equation with respect to the first camera is known (in other words, the external calibration of the first camera is known). It only returns a binary decision for the detection of obstacles from a pair of images. In Section 3.5, an analysis of the choice of the threshold δ is given for simulated data. The algorithm in the next section removes the requirement of knowing the ground plane equation.

3.3 Obstacle Detection with Unknown Ground Plane (UGP)

This algorithm uses a method similar to the algorithm in the previous section, although it does not require *a priori* knowledge of the ground plane equation, or the internal or external camera calibration. Like the previous algorithm, it assumes the ground plane is locally flat, it can be applied to either stereo pair or motion images, and it only returns a yes/no answer for obstacles. We call this algorithm the **unknown ground plane** algorithm, which is abbreviated **UGP**. Clearly, this algorithm is more general than the previous one, but as shown in Section 3.5, **KGP** performs better if good *a priori* estimates for the parameters are available.

Since we are using a pinhole camera model, the 3D coordinates of a point $\boldsymbol{P_i}$ can also be used to represent the image point $\boldsymbol{p_i}$ in homogeneous coordinates. Using the ground plane equation, it is easy to show (see Appendis A) that given an arbitrary ground plane point, $\boldsymbol{p_i} \Longleftrightarrow \boldsymbol{p_i'}$, there is an invariant 3 by 3 matrix:

$$\boldsymbol{A} = H\boldsymbol{R} + \boldsymbol{t}\boldsymbol{n}^T \tag{3.10}$$

such that

$$k_i\boldsymbol{p_i'} = \boldsymbol{A}\boldsymbol{p_i} \tag{3.11}$$

where H is the height of the focal point of the first camera above the ground plane, $\boldsymbol{R}$ is the rotation between the two cameras/instants, $\boldsymbol{t}$ is the translation between the two cameras/instants, $\boldsymbol{n}$ is the normal vector of the ground plane with respect to the first camera coordinate system, and k_i is a scale factor of the point pair $\boldsymbol{p_i} \Longleftrightarrow \boldsymbol{p_i'}$, which accounts for the fact that the representation in homogeneous coordinates is not unique.

Since A can be normalized up to a factor that is "absorbed" into k_i, let

$$\boldsymbol{A} = \begin{pmatrix} s_1 & s_2 & s_3 \\ s_4 & s_5 & s_6 \\ s_7 & s_8 & 1 \end{pmatrix} \tag{3.12}$$

Then Eq. 3.11 can be written as:

$$k_i \begin{pmatrix} x_i' \\ y_i' \\ 1 \end{pmatrix} = \begin{pmatrix} s_1 & s_2 & s_3 \\ s_4 & s_5 & s_6 \\ s_7 & s_8 & 1 \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} \tag{3.13}$$

Eliminating k_i, we have:

$$\begin{pmatrix} x_i & y_i & 1 & 0 & 0 & 0 & -x_i x_i' & -y_i x_i' \\ 0 & 0 & 0 & x_i & y_i & 1 & -x_i y_i' & -y_i y_i' \end{pmatrix} \begin{pmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \\ s_8 \end{pmatrix} = \begin{pmatrix} x_i' \\ y_i' \end{pmatrix} \tag{3.14}$$

The above equations were derived for the pinhole camera model, but they actually hold in general, if the two images are obtained from cameras with the same internal parameters (e.g. monocular motion or stereo with two identical cameras). Let $\boldsymbol{C}$ be the internal calibration matrix [26], and $\hat{\boldsymbol{p}}$ and $\hat{\boldsymbol{p}}'$ be the uncalibrated 2D vectors for the first and second cameras, respectively. We have

$$\boldsymbol{p} = \boldsymbol{C}\hat{\boldsymbol{p}} \tag{3.15}$$

and

$$\boldsymbol{p}' = \boldsymbol{C}\hat{\boldsymbol{p}}' \tag{3.16}$$

By substituting $\boldsymbol{p}$ and $\boldsymbol{p}'$ in Eq. 3.11 with $\hat{\boldsymbol{p}}$ and $\hat{\boldsymbol{p}}'$, respectively, we have

$$k_i \hat{\boldsymbol{p}'}_i = \boldsymbol{C^{-1}AC}\hat{\boldsymbol{p}}_i \tag{3.17}$$

Obviously, the linear system Eq. 3.14 is still valid. The only differences are that the definition of the unknown vector changes to the elements of $\boldsymbol{C^{-1}AC}$ instead of elements of $\boldsymbol{A}$, and now x_i, y_i and x_i', y_i' refer to the uncalibrated image vectors. This is why this algorithm does not need internal calibration.

Assuming that there are at least three ground plane points visible, the following proposition is the foundation for the algorithm. The proof is similar to that of the proposition in the previous section.

Proposition: *Given n point correspondences* $\boldsymbol{p_i} \Longleftrightarrow \boldsymbol{p_i}'$, $i = 1, ..., n$, *they are all ground plane points iff the following linear system is consistent:*

$$\begin{pmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x_1' & -y_1x_1' \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y_1' & -y_1y_1' \\ ... & & & & & & & \\ x_n & y_n & 1 & 0 & 0 & 0 & -x_nx_n' & -y_nx_n' \\ 0 & 0 & 0 & x_n & y_n & 1 & -x_ny_n' & -y_ny_n' \end{pmatrix} \begin{pmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \\ s_8 \end{pmatrix} = \begin{pmatrix} x_1' \\ y_1' \\ ... \\ x_n' \\ y_n' \end{pmatrix} \tag{3.18}$$

i.e. the rank of the coefficient matrix is equal to the rank of the augmented matrix of this linear system.

In this case, the equality in rank can be interpreted as a coplanarity constraint. The remainder of the algorithm is identical to the **KGP** algorithm; however, the dimensionality of the system is greater.

Note that both of these algorithms utilize a singular value decomposition [92] to determine if a linear system is consistent, under the assumption that the ground plane is a true plane. In practice, the road surface may have bumps and dents. Thus, the surface may not satisfy the coplanarity constraint. Although we can use a lower threshold to tolerate this kind of variation of the road surface, the simulation results in Section 3.5 show that when the noise increases, the ratio value decreases dramatically, from the order of 10^{-6} in noise-free case to the order of 10 with noise levels less than $\pm 10.0\%$. This implies that these two algorithms are sensitive to noise. The next section describes an algorithm based on partial 3D reconstruction that is more robust in the presence of noise, based on the experiments in Section 3.5.

3.4 Obstacle Detection Based on Ground Plane Estimation (EGP)

Since roads may have hills and curves, the orientation of the local ground plane may change with time. The algorithm presented in this section is one which can adapt to these changes. Estimation of the ground plane is done in such a way that the 3D heights of points with respect to the plane can be estimated from partially calibrated stereo. The previous two algorithms measure the deviation of the data from a planar configuration (known or unknown). This algorithm examines the heights of all of the points. Since ground plane points will not necessarily have height zero, it is necessary to use a threshold to distinguish between ground plane points and obstacle points. For sake of clarity, we call the estimated unknown ground plane the *reference plane*. The algorithm for computing the heights is based on the following assumptions:

- There is no translation component between the two uncalibrated stereo cameras along the Z direction (focal axis direction) of the first camera coordinate system, i.e. $t_Z = 0$. This is what is meant by *partially calibrated stereo*. Information about the absolute pose of these cameras (e.g. pan/pitch/tilt angles) and the other two translation components (e.g. baseline between the two cameras) is not required.

- The height H of the first camera above the reference plane is known.

In addition to the above assumptions, we also assume that if there is a nonzero rotation between the two cameras, then the internal camera parameters are known. In practice [3], two cameras are typically aligned, i.e. the optical axes of the two cameras are parallel to each other, and it will be shown in the next section that in this case no internal calibration is required.

3.4.1 Derivation of the Algorithm

This algorithm is called the **estimated ground plane** algorithm, and is abbreviated **EGP**. Each 3D point $\boldsymbol{P}_i$ is at a height h_i from the reference plane. If h_i is 0, this point is exactly on the plane; if h_i is positive, this point is above the reference plane; if h_i is negative, this point is below the plane. Now we can preset a threshold δ_h such that any point with its height $|h_i| \leq \delta_h$ is regarded as a ground plane point. In this way, we view the ground plane as a plank with thickness $2\delta_h$ instead of a precise plane. If the road actually lies within this plank, then the height of obstacle that can be detected will depend on δ_h. In the worst case, an obstacle which lies in an indentation will need to have height at least $2\delta_h$ to be outside this volume.

Let $\boldsymbol{p_i}$ and $\boldsymbol{p_i}'$ be the two corresponding images of $\boldsymbol{P_i}$ in the two image planes, respectively. Note that $\boldsymbol{P_i}$ has the same motion parameters $(\boldsymbol{R}, \boldsymbol{t})$ with respect to the two cameras and the same plane orientation $\boldsymbol{n}$ as those points on the reference plane. It can be shown [27] that the relationship between $\boldsymbol{p_i}$ and $\boldsymbol{p_i}'$ is:

$$k_i' d_i \boldsymbol{p_i}' = (d_i \boldsymbol{R} + \boldsymbol{t}\boldsymbol{n}^T)\boldsymbol{p_i} \tag{3.19}$$

where d_i is the distance from $\boldsymbol{P_i}$ to the origin of the camera coordinate system in the left image in the direction of $\boldsymbol{n}$, and k_i' is a scale factor, which is defined as the ratio of the two depths of the same physical point $\boldsymbol{P_i}$ viewed in the left and right camera coordinate systems, i.e.

$$k_i' = \frac{Z_i'}{Z_i} \tag{3.20}$$

Clearly, we have

$$d_i = H - h_i \tag{3.21}$$

where H is the height of the left camera above the reference plane. Thus,

$$k_i'(H - h_i)\boldsymbol{p_i}' = [(H - h_i)\boldsymbol{R} + \boldsymbol{t}\boldsymbol{n}^T]\boldsymbol{p_i} \tag{3.22}$$

We define the state vector $\boldsymbol{S}$ to be the vector consisting of the eight elements of the matrix $\boldsymbol{A}$ in Eq.3.12. Note that this vector combines information about the ground plane and the transformation between cameras. Assuming for a moment that the matrix $\boldsymbol{A}$ is known, then for an arbitrary 3D point $\boldsymbol{P_i} = (X_i, Y_i, Z_i)^T$, with its

corresponding image point at the left image plane $\boldsymbol{p}_i$, there must be a corresponding 3D reference plane point $\boldsymbol{Q}_i = (X_{Qi}, Y_{Qi}, Z_{Qi})^T$ which shares the same line of sight with $\boldsymbol{P}_i$ in the left image plane (see Fig. 3.1). The image point $\boldsymbol{p}_i''$ of this reference plane point $\boldsymbol{Q}_i$ in the right image plane can be obtained by using Eq. 3.11:

$$k_i \boldsymbol{p}_i'' = \boldsymbol{A}\boldsymbol{p}_i \tag{3.23}$$

After eliminating k_i, the right image coordinates are:

$$x_i'' = \frac{s_1 x_i + s_2 y_i + s_3}{s_7 x_i + s_8 y_i + 1} \tag{3.24}$$

$$y_i'' = \frac{s_4 x_i + s_5 y_i + s_6}{s_7 x_i + s_8 y_i + 1} \tag{3.25}$$

Thus, we have:

$$k_i'' H \boldsymbol{p}_i'' = (H\boldsymbol{R} + \boldsymbol{t}\boldsymbol{n}^T)\boldsymbol{p}_i \tag{3.26}$$

where k_i'' is the scale factor corresponding to $\boldsymbol{p}_i \Longleftrightarrow \boldsymbol{p}_i''$.

Combining Eq.3.22 and Eq.3.26:

$$k_i'(H - h_i)\boldsymbol{p}_i' - k_i'' H \boldsymbol{p}_i'' = -h_i \boldsymbol{R}\boldsymbol{p}_i \tag{3.27}$$

Let $\boldsymbol{R_1}, \boldsymbol{R_2}, \boldsymbol{R_3}$ be the row vectors of $\boldsymbol{R}$, i.e. $\boldsymbol{R} = (\boldsymbol{R_1}, \boldsymbol{R_2}, \boldsymbol{R_3})^T$, and $\boldsymbol{t} = (t_X, t_Y, t_Z)^T$. Since we have

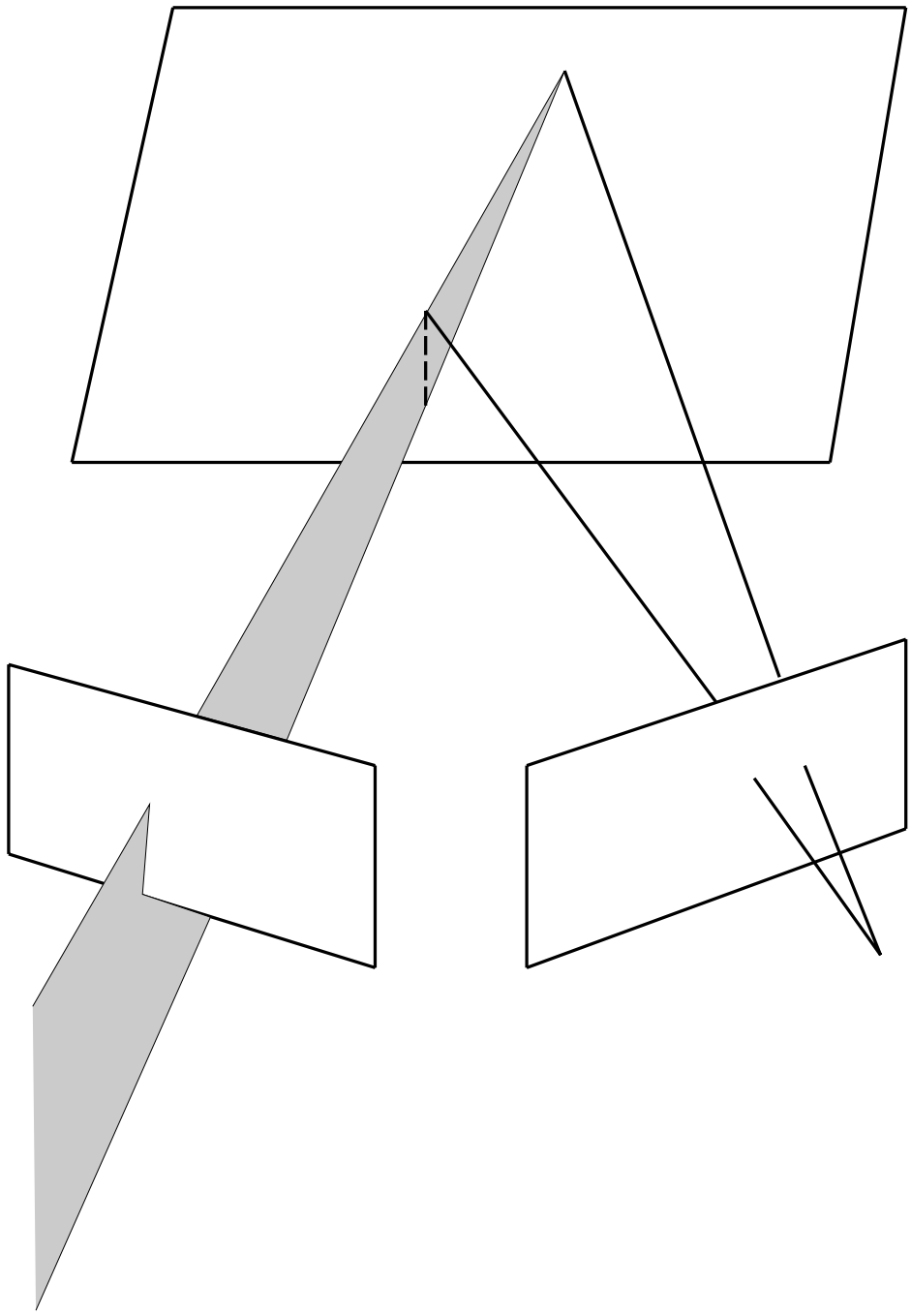

Figure 3.1. The geometry of an arbitrary 3D point $\boldsymbol{P_i}$ and its corresponding ground plane 3D point $\boldsymbol{Q_i}$.

$$\boldsymbol{P_i} \Longleftrightarrow \boldsymbol{p_i} \Longleftrightarrow \boldsymbol{p_i}'$$

$$\boldsymbol{Q_i} \Longleftrightarrow \boldsymbol{p_i} \Longleftrightarrow \boldsymbol{p_i}''$$

by the definition of k_i as defined in Eq. 3.20,

$$k_i' = \frac{Z_i'}{Z_i} = \boldsymbol{R_3} \cdot \boldsymbol{p_i} + \frac{t_Z}{Z_i} \tag{3.28}$$

$$k_i'' = \frac{Z_{Qi}'}{Z_{Qi}} = \boldsymbol{R_3} \cdot \boldsymbol{p_i} + \frac{t_Z}{Z_{Qi}} \tag{3.29}$$

Based on the matrix $\boldsymbol{A}$, we can solve for the rotation $\boldsymbol{R}$ and the translation $\boldsymbol{t}$ up to a scale factor (see Chapter 4). However, with unknown Z_i and Z_{Qi}, if $t_Z \neq 0$, k_i' and k_i'' remain unknown. Thus, Eq. 3.27 is not sufficient to solve for h_i, since the three equations are not independent. If $t_Z = 0$, which is the case we assume, with known relative rotation $\boldsymbol{R}$, k_i' and k_i'' can be determined from

$$k_i' = k_i'' = \boldsymbol{R_3} \cdot \boldsymbol{p_i} \tag{3.30}$$

Therefore, using Eq. 3.27, the solution for h_i is:

$$h_i = \frac{(\boldsymbol{R_3} \cdot \boldsymbol{p_i})\boldsymbol{p_i}' - (\boldsymbol{R_3} \cdot \boldsymbol{p_i})\boldsymbol{p_i}''}{(\boldsymbol{R_3} \cdot \boldsymbol{p_i})\boldsymbol{p_i}' - \boldsymbol{R}\boldsymbol{p_i}} H \tag{3.31}$$

Here we follow the convention that the quotient of two vectors is the vector of quotients of their corresponding elements. Thus, Eq. 3.31 can be written as follows:

$$\begin{aligned} h_i &= (x_i'' - x_i')(\frac{\boldsymbol{R_1}\cdot\boldsymbol{p_i}}{\boldsymbol{R_3}\cdot\boldsymbol{p_i}} - x_i')^{-1}H \\ &= (y_i'' - y_i')(\frac{\boldsymbol{R_2}\cdot\boldsymbol{p_i}}{\boldsymbol{R_3}\cdot\boldsymbol{p_i}} - y_i')^{-1}H \end{aligned} \tag{3.32}$$

This solution looks very simple. It does not require the absolute pose information of the cameras such as pan/pitch/tilt angles, nor does it require orientation or the distance of the baseline between the two cameras.

A special case (which is perhaps the most frequently encountered) occurs when the two cameras are aligned. Alignment means that the optical axes of the two cameras are parallel, but the baseline is not necessarily aligned with the horizontal or vertical axes of each of the cameras. In fact, the baseline can be any line in the $X-Y$ plane. In addition, there is no rotation about the optical axis of each camera, i.e. the image coordinate axes of the two cameras are in parallel. In this case, $\boldsymbol{R} = \boldsymbol{I}$, where $\boldsymbol{I}$ is a 3×3 identity matrix. Thus, Eq. 3.32 is simplified as:

$$h_i = \frac{x_i' - x_i''}{x_i' - x_i}H = \frac{y_i' - y_i''}{y_i' - y_i}H \tag{3.33}$$

Note in this case, even internal calibration is not necessary, because x_i, x_i', x_i'' and y_i, y_i', y_i'' can be represented in terms of any arbitrary uncalibrated camera coordinate system.

Now the remaining question is how to estimate the state vector $\boldsymbol{S}$, i.e. $\boldsymbol{A}$. Assuming there are no obstacles in the first few image pairs, an initial estimate of the state vector can be obtained by solving an overconstrained linear system using least mean squares. For every subsequent stereo frame, this state vector is updated based

on the information obtained from that frame. Hence, the algorithm can be expressed as:

- The first pair of images is assumed to have only ground plane points. The overconstrained linear system (Eq. 3.18) can be solved using least mean squares.
- For current frame i, use the current estimate of the state vector up to the last frame $\boldsymbol{A_{i-1}}^{best}$ to estimate the height of each point j in the current frame i, $\hat{h_{ij}}$, then form a ground plane point subset:

$$S_i = \{P_{ij} : |\hat{h_{ij}}| < \delta_h\}$$

  Based on S_i, compute the state vector $\boldsymbol{A_i}$ and covariance matrix $\boldsymbol{\Lambda_i}$ of the current frame; then refine the state vector $\boldsymbol{A_i}^{best}$ and its covariance matrix $\boldsymbol{\Lambda_i}^{best}$ using a Kalman Filter:

$$\boldsymbol{A_i}^{best} = (\boldsymbol{\Lambda_i}^{best})^{-1}[(\boldsymbol{\Lambda_{i-1}}^{best})^{-1}\boldsymbol{A_{i-1}}^{best} + \boldsymbol{\Lambda_i}^{-1}\boldsymbol{A_i}] \tag{3.34}$$

$$\boldsymbol{\Lambda_i}^{best} = [(\boldsymbol{\Lambda_i}^{best})^{-1} + \boldsymbol{\Lambda_i}^{-1}]^{-1} \tag{3.35}$$

- Based on $\boldsymbol{A_i}^{best}$, recalculate the height h_{ij} of each point j at this frame; report obstacles for any point P_{ij} if $|h_{ij}| > \delta_h$

There is another point that needs to be addressed. The above algorithm works well if the ground plane is flat globally. In practice, this may not be true, and there may be hills at different scales. Hills at large scales can be accommodated by weighting

previous estimates so that the Kalman Filter only accumulates its history from the last n frames. Experimental results (Section 3.5) show that this modified version works much better for real road scenes.

3.4.2 Error Analysis

In this section, the stability of the estimate of the height h_i for each point $\boldsymbol{P_i}$ based on Eq. 3.31 is analyzed. In general, there are four sources of error which contribute to the error in h_i:

- measurement error of the physical height of the camera, H;
- localization error of the 2D image points $\boldsymbol{p_i}, \boldsymbol{p_i}', \boldsymbol{p_i}''$;
- deviation of the ground plane points from the reference plane;
- measurement error of the relative rotation between the two cameras;

The relative error of camera height H is on the order of a percent or two in practice. Since h_i is linearly proportional to H, the contribution to the relative error in h_i will be the same. Thus, we do not take this into account. The contribution of the error of localization of the image points is complicated, especially the error of $\boldsymbol{p_i}''$, which is a function of the state vector in the Kalman Filter, as is the contribution of the error of the deviations of the ground plane points from the reference plane. To simplify the problem, we also do not consider these last two error sources here. Instead, we leave the analysis of these errors under different Gaussian noise levels to the simulation studies in Section 3.5. Thus, the following error analysis only considers the error in relative rotation. Since it is assumed that the stereo cameras are aligned,

i.e. the relative rotation is approximately zero, our analysis is done only for this case. An analysis for the general case of relative rotation is similar.

Expanding Eq. 3.31 in terms of Taylor's series at $\boldsymbol{R} = \boldsymbol{I}$, we have:

$$h_i = \left[\frac{(\boldsymbol{R_3} \cdot \boldsymbol{p_i})\boldsymbol{p_i}' - (\boldsymbol{R_3} \cdot \boldsymbol{p_i})\boldsymbol{p_i}''}{(\boldsymbol{R_3} \cdot \boldsymbol{p_i})\boldsymbol{p_i}' - \boldsymbol{R}\boldsymbol{p_i}} H\right]_{\boldsymbol{R}=\boldsymbol{I}} + \left[\frac{\partial h_i}{\partial \boldsymbol{R}}\right]_{\boldsymbol{R}=\boldsymbol{I}} (\boldsymbol{R}-\boldsymbol{I}) + O((\boldsymbol{R}-\boldsymbol{I})^2) \quad (3.36)$$

Let us define the rotation matrix as:

$$\boldsymbol{R} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix}$$

By ignoring the higher order terms in Eq. 3.36, we have an approximation to the error in the estimated height:

$$\begin{aligned} \triangle h_i &= \left[\frac{\partial h_i}{\partial \boldsymbol{R}}\right]_{\boldsymbol{R}=\boldsymbol{I}} (\boldsymbol{R}-\boldsymbol{I}) \\ &= (x_i(x_i \triangle r_{31} + y_i \triangle r_{32} + \triangle r_{33}) \\ &\quad -(x_i \triangle r_{11} + y_i \triangle r_{12} + \triangle r_{13}))(x_i - x_i')^{-2}(x_i'' - x_i')H \end{aligned} \quad (3.37)$$

There are nine elements in the matrix $r_{ij}, i, j = 1, 2, 3$. However, there are only three degrees of freedom for a rotation: the camera relative roll/pan/tilt angles [68]. Let ψ, ϕ, θ be the three independent Euler roll/pan/tilt angles. From [68], it can be shown that:

$$
\begin{aligned}
\triangle r_{11} &= -\sin\phi\cos\psi\triangle\phi - \cos\phi\sin\psi\triangle\psi \\
\triangle r_{12} &= \sin\phi\sin\psi\triangle - \cos\phi\cos\psi\triangle\psi \\
\triangle r_{13} &= -\cos\phi\triangle\phi \\
\triangle r_{21} &= -(\sin\theta\sin\psi + \cos\theta\sin\phi\cos\psi)\triangle\theta - \sin\theta\cos\phi\cos\psi\triangle\phi \\
&\quad +(\cos\theta\cos\psi + \sin\theta\sin\phi\sin\psi)\triangle\psi \\
\triangle r_{22} &= (\cos\theta\sin\phi\sin\psi - \sin\theta\cos\psi)\triangle\theta + \sin\theta\cos\phi\sin\psi\triangle\phi \\
&\quad +(\sin\theta\sin\phi\cos\psi - \cos\theta\sin\psi)\triangle\psi \\
\triangle r_{23} &= -\cos\theta\cos\phi\triangle\theta + \sin\theta\sin\phi\triangle\phi \\
\triangle r_{31} &= (\cos\theta\sin\psi - \sin\theta\sin\phi\cos\psi)\triangle\theta + \cos\theta\cos\phi\cos\psi\triangle\phi \\
&\quad +(\sin\theta\cos\psi - \cos\theta\sin\phi\sin\psi)\triangle\psi \\
\triangle r_{32} &= (\cos\theta\cos\psi + \sin\theta\sin\phi\sin\psi)\triangle\theta - \cos\theta\cos\phi\sin\psi\triangle\phi \\
&\quad -(\sin\theta\sin\psi + \cos\theta\sin\phi\cos\psi)\triangle\psi
\end{aligned}
\tag{3.38}
$$

When $\boldsymbol{R} = \boldsymbol{I}$, then $\theta = \phi = \psi = 0$. Assume that the maximum deviation of the three angles is δ. Thus,

$$\triangle r_{11} = \triangle r_{22} = \triangle r_{33} = 0.0$$

$$\triangle r_{21} = \triangle r_{31} = \triangle r_{32} = \delta$$

$$\triangle r_{12} = \triangle r_{13} = \triangle r_{23} = -\delta$$

This makes it possible to express the relative error in height from Eq. 3.37 as:

$$\frac{\triangle h_i}{H} = \frac{(x_i'' - x_i')\delta}{(x_i - x_i')^2} \tag{3.39}$$

Clearly, the relative error increases as depth increases, because the term $x_i - x_i'$ is inversely proportional to depth. Since we typically focus attention on the central part of each image, and since $|x_i'' - x_i'| \ll |x_i - x_i'|$, for the typical values within our focus-of-attention window, the relative error is:

$$|\frac{\triangle h_i}{H}| \sim 10^{-2}$$

The experimental results presented in Section 3.5 indicate that this bound is consistent with the actual relative error. A simulation analysis for the performance of this algorithm under different levels of noise, as well as the false-positive and false-negative probabilities with respect to the different thresholds on heights, is also presented there.

In order to obtain the relationship between the accuracy of the height computation and the depth, let us assume that the stereo camera has a tilt angle θ, the camera height is H, and there is an obstacle at depth d with height h. If the obstacle height changes by δh, the corresponding change in the image domain δy can be obtained from Fig. 3.2:

$$\delta y = S_y[\tan(\theta - \arctan\frac{H-h-\delta h}{d}) - \tan(\theta - \arctan\frac{H-h}{d})] \tag{3.40}$$

where δy is the resulted measurement error in the image owing to the 3D height change δh, S_y is the scale factor along the Y axis in the image, and H is the height of the camera from the ground plane.

Obviously, δy is always bounded by the image resolution, i.e. any change with $\delta y < P$ would not be detected in the image domain where P is the pixel size. Based on this constraint, we can solve for Eq. 3.40 to obtain the maximum change $\triangle h$, parameterized by depth d. Since Eq. 3.40 is non-linear and involves trigonometric functions, a closed-form solution does not seem to be possible. On the other hand,

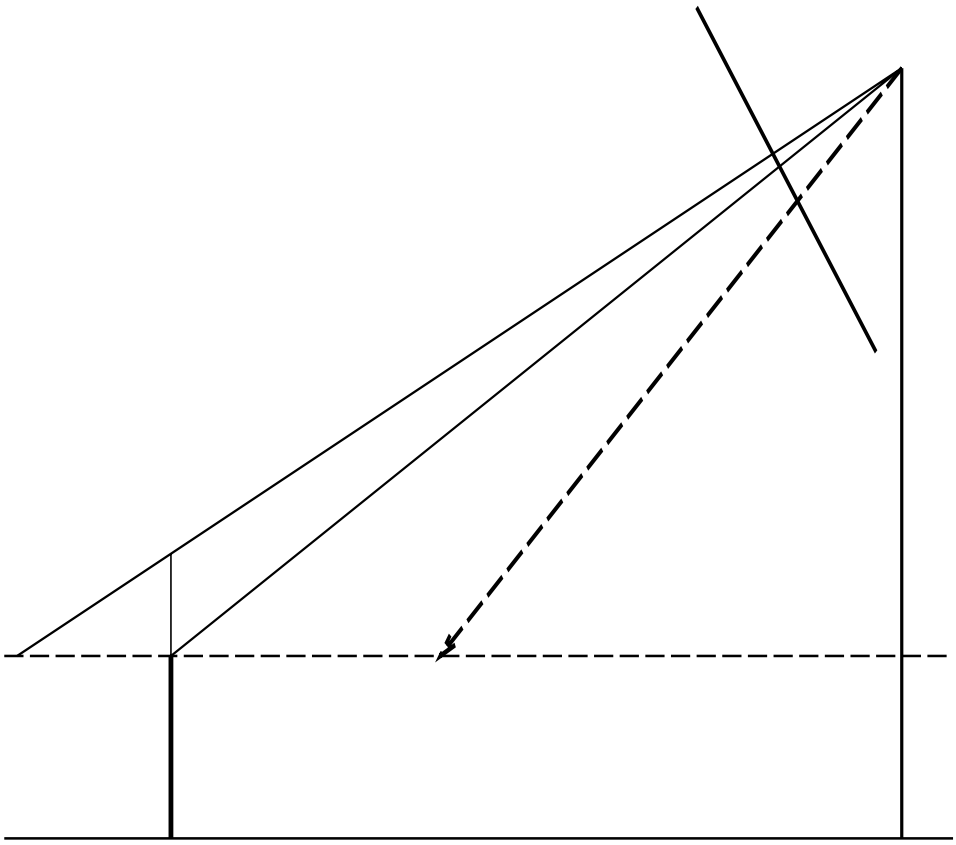

Figure 3.2. Geometric illustration of the relationship between the depth of an obstacle and its height estimation accuracy.

even without a closed-form solution, a general relationship between the accuracy of height computation and the obstacle depth can still be seen clearly from Eq. 3.40, i.e. when the depth increases, the accuracy decreases, which is again consistent with our intuition. In the next section, a simulation-based analysis for different heights of an obstacle with fixed depth is given.

3.5 Experiments

This section compares the three algorithms with respect to robustness in the presence of noise in real and simulated image sequences. We address the question of what is the smallest obstacle that can be reliably detected for a given level of noise. The simulation only adds noise of the type produced by small scale deviations in the ground plane. The real image sequences, of course, include all the usual sources of noise such as misalignment of cameras, errors in camera calibration, error in knowledge of the ground plane, as well as bumps, indentations and errors in the optic flow/disparity.

The simulated data consisted of 10 ground plane points and an obstacle point at a distance of 20 ft. from the camera. Fig. 3.3 shows this hypothetical scenario for this simulation with the distribution of the 10 ground plane points and the obstacle point. The height of the obstacle point was varied from 0 ft. to 6.5 ft. The stereo baseline distance is 0.66 ft., and the height of the camera was 3.55 ft.. These are the actual stereo baseline distance and the actual height of the stereo cameras when mounted on the vehicle used to generate the experimental results. In addition, the displacement vectors were multiplied by Gaussian noise, which is equivalent to random fluctuations

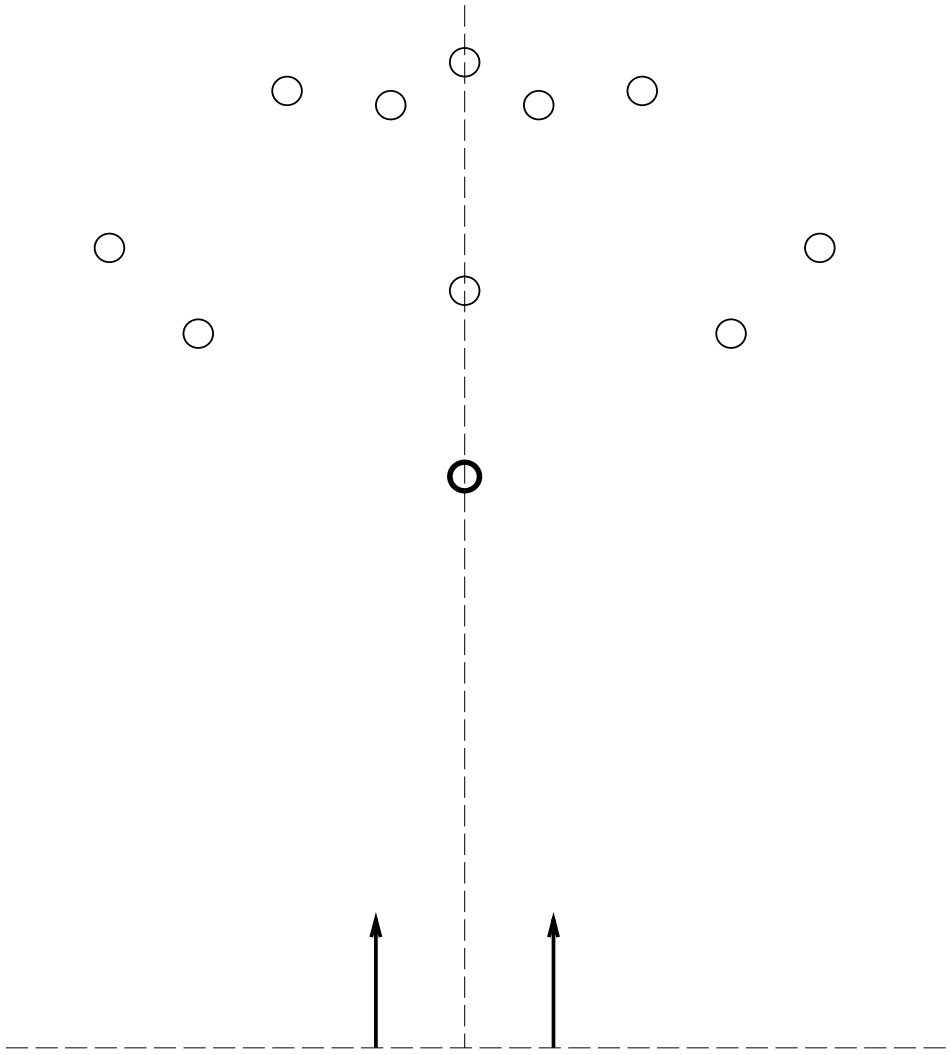

Figure 3.3. The scenario of the simulation. The stereo cameras point horizontally. The light circles represent the ground plane points while the dark circle represents the obstacle point. Note that this figure is only an illustration; in particular, it is not in scale relative to the actual measurements.

in the heights of the ground plane points, and thus could be interpreted as bumps and depressions in the ground plane. For the given height of the camera and a 10% noise level, the maximum variation in the height of the ground plane points induced by this noise would be 0.355 ft. The simulation was performed to *determine for each algorithm and each level of noise, the smallest obstacle that could be detected.* For an obstacle to be detectable, there must be a decision rule which will find all positive instances and will not produce too many false alarms. In other words, it is safe to allow a few false positives, but the probability of a false negative (failure to detect an obstacle) must be so close to zero so as to be quite unlikely to ever arise in practice.

The simulation results for the **KGP** algorithm are shown in Fig. 3.4, which shows the average of ten runs based on different seeds, all of which show very similar curves. The statistic for detecting obstacle points is the ratio value $\frac{\sigma_{min}(\boldsymbol{D})}{\sigma_{min}(\boldsymbol{Db})}$. The graph shows how this ratio varies with the height of the obstacle point and the level of noise. In order for an obstacle point of a given height (or greater) to be detectable, one should be able to draw a horizontal line that separates the ratio value for that height from the ratio value when the height is zero. We can see clearly from Fig.3.4 that for those points with heights greater than 1 ft., the ratio values are very close to 1, and are almost unaffected by noise. However, for levels of noise greater than 10%, it will be impossible to choose a threshold that would distinguish ground-plane points from obstacle points. Even for smaller levels of noise, it will be impossible to distinguish ground-plane points from obstacle points with heights less than 1 ft. Based on the simulation, a threshold on this ratio value of between 5 and 10 would be able to correctly detect obstacles with height above 1 ft. up to $\pm 10\%$ noise. For

obstacles with heights less than 1 ft., even $\pm 2.0\%$ noise would be a problem. From the graph, it is apparent that the ratio increases slightly with the height of the obstacle. However, the increase rate of this statistic is much slower than that of the obstacle height. Therefore, it is still very easy to find a threshold for detecting obstacles. More interestingly, this ratio statistic shows the largest rate of increase in the noise-free case. As the noise levels increases, this rate goes down. And when the noise level is at $\pm 10.0\%$, the ratio statistic stays at 1.0 and appears to be independent of the obstacle height. That implies that in real applications, which are always corrupted by noise, it is not necessary to take this increase into account.

Fig. 3.5 shows a similar simulation for the **UGP** algorithm. Clearly, the general performance of this algorithm with respect to noise is worse than that of **KGP**. This is because the **KGP** algorithm used knowledge about the environment while **UGP** must estimate it. However, for points higher than 2 ft., the **UGP** algorithm can correctly detect them, up to noise level of $\pm 10.0\%$. Satisfactory threshold values are between 5 and 10. For obstacle points with heights less than 2 ft., no satisfactory threshold can be found when the noise level is larger than $\pm 1.0\%$. As with **UGP**, the ratio statistic increases as the obstacle height increases in the noise-free case; when there is noise, this statistic stays around 1.0 and is roughly independent of the obstacle height.

Fig.3.6 shows the simulation results for the **EGP** algorithm. The plot shows the average of the absolute relative error of the estimated height as a percentage of the actual 3D camera height. Again, we use the hypothetical scenario in Fig. 3.3 for this simulation. From this figure, we can see that with an increase in noise level, the relative error also increases, which is consistent with intuition. 1000 trials of

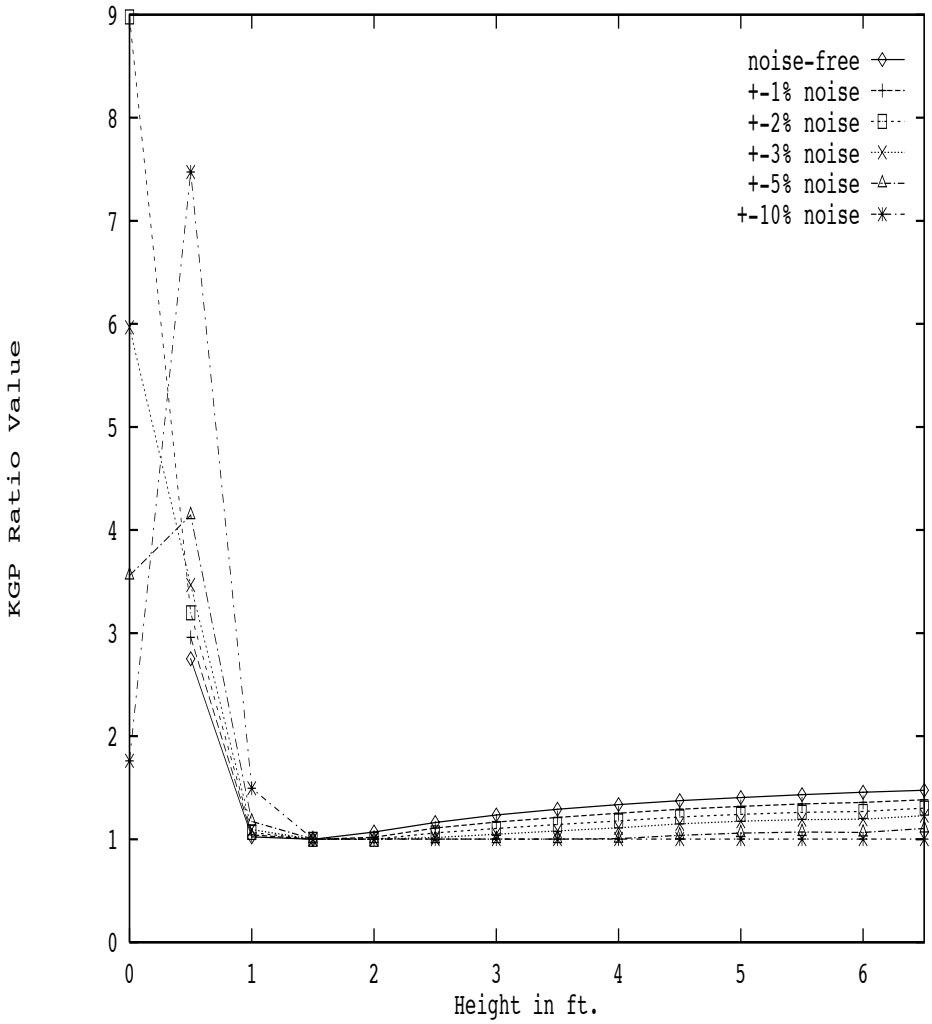

Figure 3.4. Simulation results of a 3D point located at 20 ft. based on **KGP**.

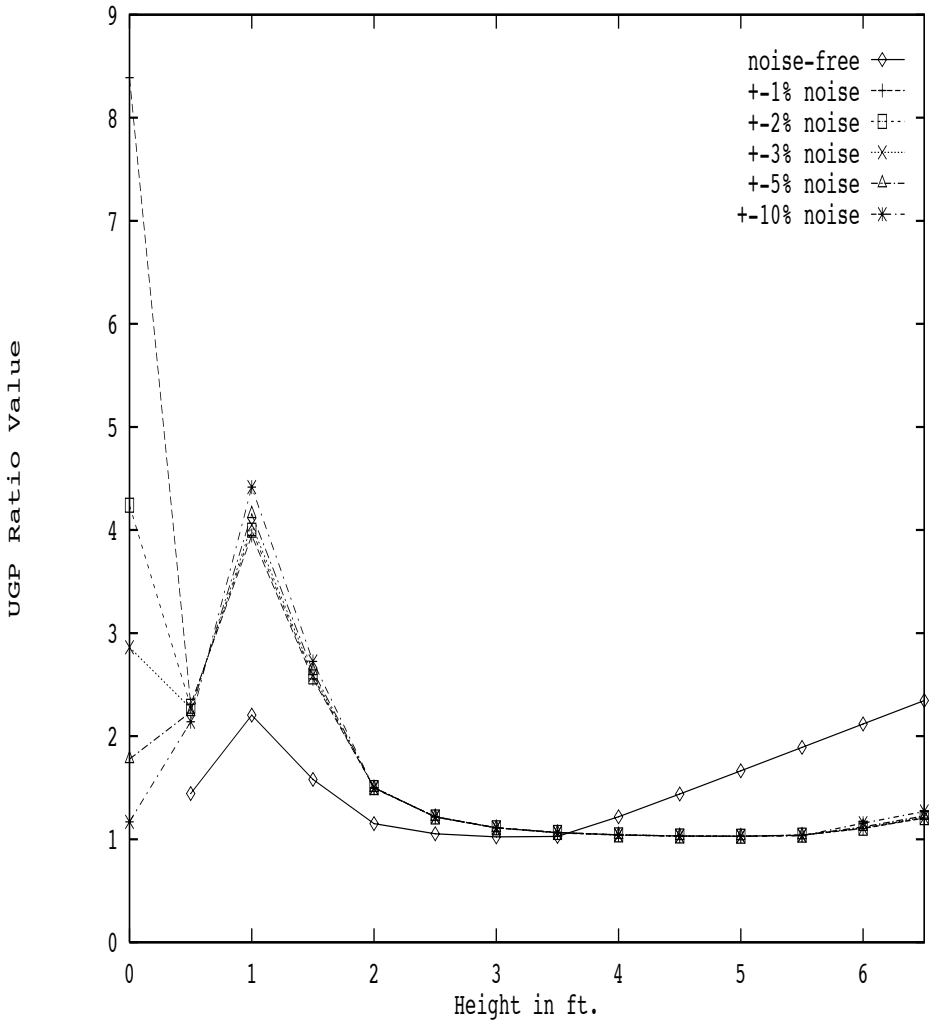

Figure 3.5. Simulation results of a 3D point located at 20 ft. based on **UGP**.

the simulation were run with different seeds for each noise level, and the absolute values of the relative errors were averaged over the 1000 trials to generate the figure. Note that for this experiment, the same noise distribution (Gaussian) was used in each of the experimental runs; only the amplitude of the noise changed (by changing the standard deviation). From this figure, with each noise level, the relative error increases as the obstacle height increases. This figure also shows that the relative error is a linear function (approximately) of the obstacle height after some threshold (1.5 ft. in height in this simulation). The reason for this linear relationship is easy to see. Under the assumption of the stereo alignment, from Eq. 3.33 the relative error defined above can be expressed as:

$$\frac{x' - x'' + \sigma}{x' - x + \eta} - \frac{x' - x''}{x' - x} \simeq \frac{\sigma}{x' - x} - \frac{\eta(x' - x'')}{(x' - x)^2}$$

where x, x' are the X correspondences in the stereo image pair, and $x' - x''$ is the "parallax" (see Fig. 3.1) arising from the non-ground plane points in one image plane, η is the given noise level, and σ is the noise level in the computation of the "parallax". Clearly, given a noise level η, with the situation in this simulation experiment, $x' - x$ is independent of height. If we assume σ is the same for a given noise level, then it is obvious that the relative error is a linear function of the "parallax" $x' - x''$, which is proportional to the obstacle height, as indicated in Eq. 3.33. Note that this linear function conclusion is also consistent with the analytical error analysis result in Eq. 3.39, which was obtained by analyzing the sensitivity to camera misalignment; in this simulation analysis, it is assumed that the noise is added to the localization of the image correspondences.

In order to determine the probabilities of false positives and false negatives, the simulation was run 10 times with different seeds for each obstacle height and for each noise level. The false-positive probability is defined as the probability of the maximum height of ground plane points being above the threshold. False-negative probability is defined as the probability of the height of the obstacle point being below the threshold. Fig.3.7 shows the frequencies of false positives and false negatives as a function of threshold for different obstacle heights and different noise levels. We are interested in finding a threshold such that the probabilities of false negatives and false positives are as small as possible. In general, one might allow some small number of false positives in order to detect smaller obstacles. For this experiment, only ten trials were run for each of the parameter settings, so we looked for a threshold such that no false positives or negatives were obtained. For example, Fig.3.7(a) shows these frequencies for a noise level of $\pm 1.0\%$ and for a variety of obstacle heights. A satisfactory threshold is 0.03 ft. because the frequency of false positive curve is zero for this value. If the obstacle height is 0.05 ft. or more, then the frequency of false negatives is also zero, and this height is defined as the smallest obstacle that can be effectively detected in this case[1]. On the other hand, if the noise level is increased to $\pm 10.0\%$, and the frequency of a false positive is still required to be zero, then one can only detect obstacles that are higher than 0.45 ft. Note this is a big improvement over the 1 ft. height for **KGP** and 2 ft. for **UGP**. This fact indicates that **EGP**

[1]The observation that no false negatives occurred in 10 trials means that one can state that the probability of such an event is at most 0.26 with a confidence level of 0.95. One would need more trials to get a better upper bound on this probability.

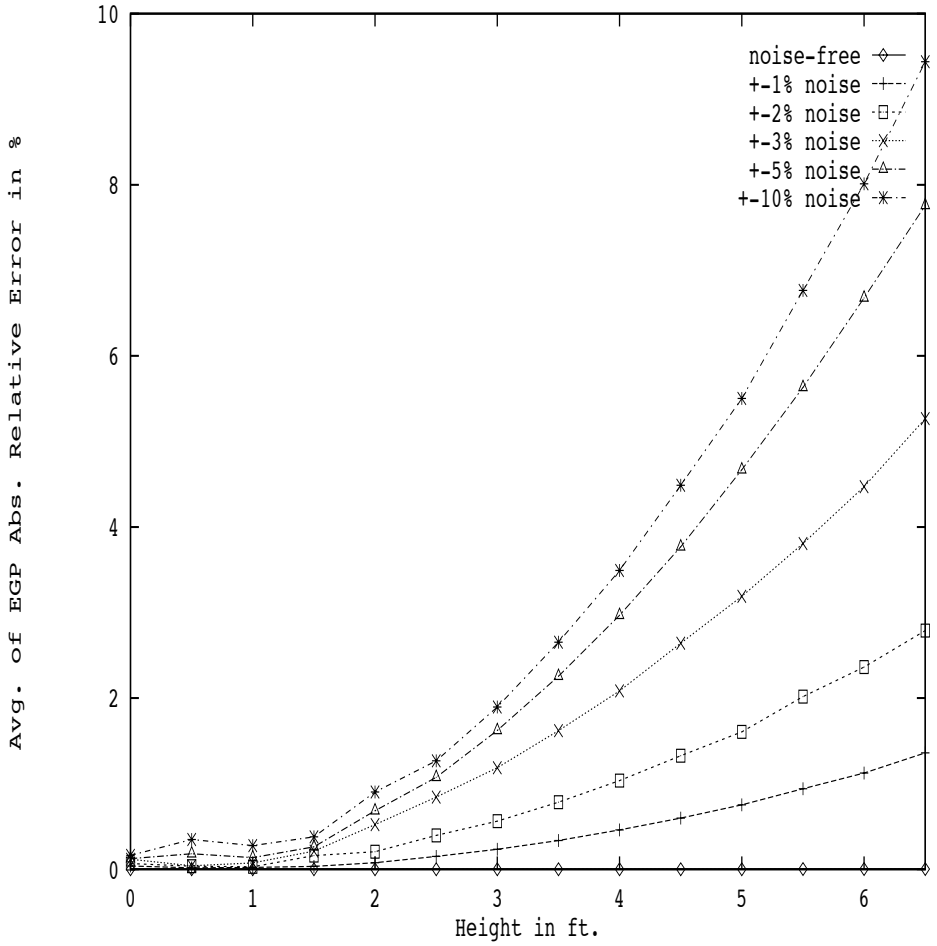

Figure 3.6. Simulation results of a 3D point located at 20 ft. based on **EGP**.

is the most robust algorithm of the three with respect to small-scale variation in the ground height.

All the algorithms have been tested on real image data. Fig.3.8 shows the left image of a hallway scene with some boxes as obstacles. The height of the cameras, H, is 3.55 ft. above the ground plane. The right image is similar and is not shown. We arbitrarily chose 38 feature points in this image, as shown in the figure, and obtained

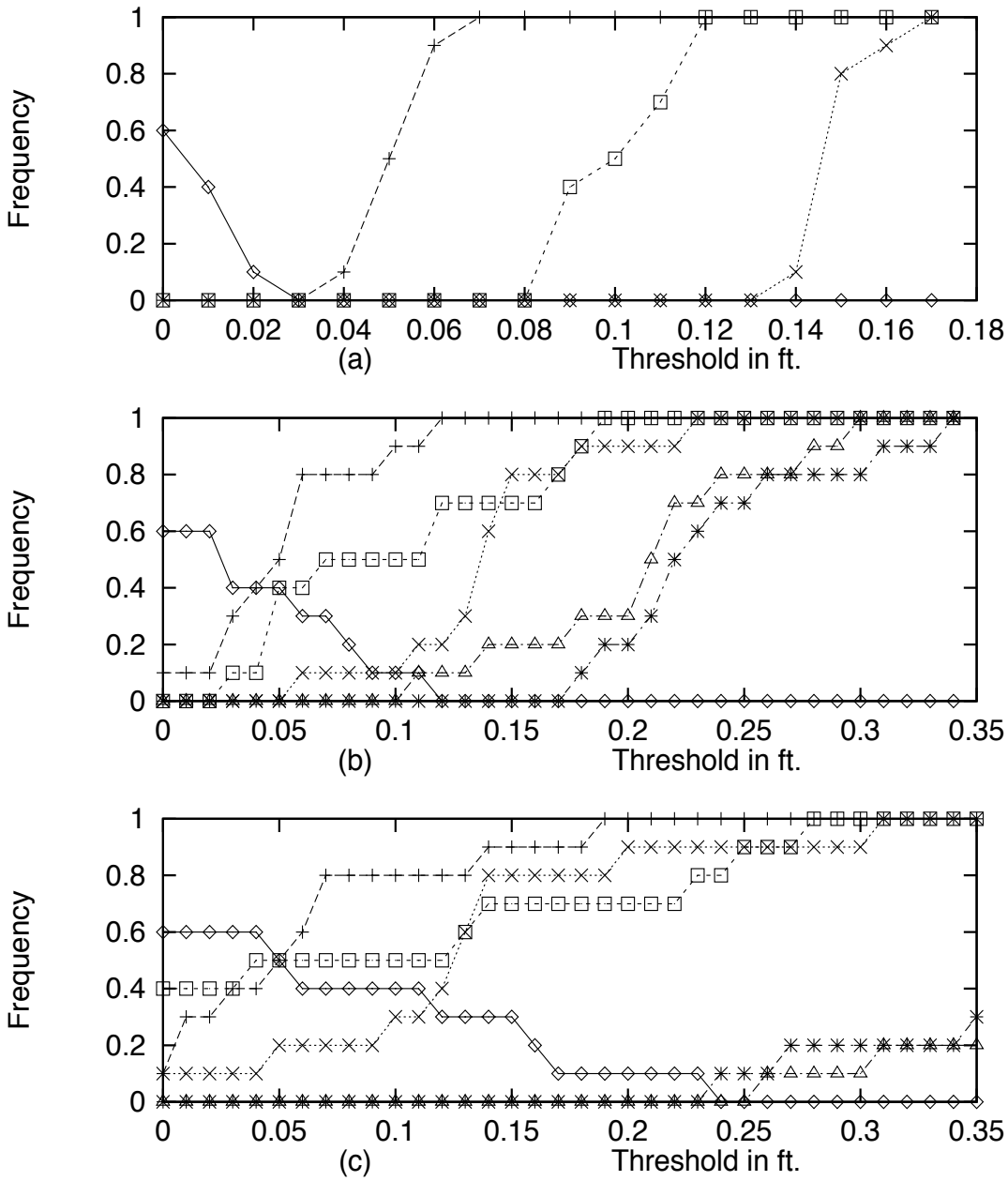

Figure 3.7. False-positives and false-negatives with respect to thresholds for different obstacle heights using **EGP**. (a) noise level $\pm 1.0\%$ (b) noise level $\pm 5.0\%$ (c) noise level $\pm 10.0\%$. See Table 3.1 for legends.

Table 3.1. Legends of the figure for false-positive and false-negative analysis of **EGP**.

Symbols	Interpretations
$\diamond$	false positive
$+$	false negative with obstacle height 0.05 ft.
$\square$	false negative with obstacle height 0.10 ft.
$\times$	false negative with obstacle height 0.15 ft.
$\triangle$	(b) false negative with obstacle height 0.20 ft. (c) false negative with obstacle height 0.40 ft.
$*$	(b) false negative with obstacle height 0.25 ft. (c) false negative with obstacle height 0.45 ft.

Figure 3.8. The left image of a hallway box scene

their displacement values with respect to the right image using Anandan's algorithm [1]. Using all 38 points in the linear system, the singular values of the matrices $\boldsymbol{D}$ and $[\boldsymbol{Db}]$ of **KGP** are listed in Table 3.2. From the table, $\sigma_{min}(\boldsymbol{D}) = 0.01689$ and $\sigma_{min}(\boldsymbol{Db}) = 0.01667$, respectively. Thus, the ratio value is 1.01, which is very close to 1. On the other hand, using only the ground plane points, the singular values are shown in Table 3.3. Now $\sigma_{min}(\boldsymbol{D}) = 0.01202$ and $\sigma_{min}(\boldsymbol{Db}) = 0.001744$, and their ratio value is 6.897.

Table 3.2. Singular values of **KGP** algorithm using 38 points in the hallway indoor obstacle scene.

SV's of $\boldsymbol{D}$	SV's of $[\boldsymbol{Db}]$
$\lambda_1 = 6.226\text{e}{+0}$	$\eta_1 = 6.227\text{e}{+0}$
$\lambda_2 = 6.219\text{e}{+0}$	$\eta_2 = 6.219\text{e}{+0}$
$\lambda_3 = 6.495\text{e}{-1}$	$\eta_3 = 6.497\text{e}{-1}$
$\lambda_4 = 1.095\text{e}{-1}$	$\eta_4 = 1.096\text{e}{-1}$
$\lambda_5 = 8.573\text{e}{-2}$	$\eta_5 = 8.723\text{e}{-2}$
$\lambda_6 = 1.689\text{e}{-2}$	$\eta_6 = 2.729\text{e}{-2}$
	$\eta_7 = 1.667\text{e}{-2}$

Table 3.3. Singular values of **KGP** algorithm using all ground points in the hallway indoor obstacle scene (coplanarity constraint).

SV's of $\boldsymbol{D}$	SV's of $[\boldsymbol{Db}]$
$\lambda_1 = 4.650\text{e}{+0}$	$\eta_1 = 4.651\text{e}{+0}$
$\lambda_2 = 4.634\text{e}{+0}$	$\eta_2 = 4.634\text{e}{+0}$
$\lambda_3 = 4.487\text{e}{-1}$	$\eta_3 = 4.488\text{e}{-1}$
$\lambda_4 = 4.057\text{e}{-2}$	$\eta_4 = 4.507\text{e}{-2}$
$\lambda_5 = 3.814\text{e}{-2}$	$\eta_5 = 4.041\text{e}{-2}$
$\lambda_6 = 1.202\text{e}{-2}$	$\eta_6 = 1.214\text{e}{-2}$
	$\eta_7 = 1.744\text{e}{-3}$

Table 3.4. Singular values of **UGP** algorithm using 38 points in the hallway indoor obstacle scene.

SV's of $\boldsymbol{D}$	SV's of $[\boldsymbol{Db}]$
$\lambda_1 = 7.215\text{e}{+5}$	$\eta_1 = 7.215\text{e}{+5}$
$\lambda_2 = 1.754\text{e}{+5}$	$\eta_2 = 1.754\text{e}{+5}$
$\lambda_3 = 1.950\text{e}{+3}$	$\eta_3 = 1.950\text{e}{+3}$
$\lambda_4 = 7.316\text{e}{+2}$	$\eta_4 = 7.839\text{e}{+2}$
$\lambda_5 = 3.008\text{e}{+2}$	$\eta_5 = 4.434\text{e}{+2}$
$\lambda_6 = 1.285\text{e}{+2}$	$\eta_6 = 1.286\text{e}{+2}$
$\lambda_7 = 1.602\text{e}{+0}$	$\eta_7 = 2.067\text{e}{+1}$
$\lambda_8 = 5.468\text{e}{-1}$	$\eta_8 = 1.062\text{e}{+0}$
	$\eta_9 = 5.335\text{e}{-1}$

We also tested the **UGP** algorithm using the same point set. With this algorithm, neither the camera internal parameters nor the external calibration information is required. Table 3.4 shows the SV's for the two matrices of the linear system with all 38 points. The ratio of the two minimum SV's of the two matrices is 1.025. Table 3.5 shows the SV's for the two matrices of the linear system using only the ground plane points; this yields a ratio of 6.594. From the data presented in Tables 3.2 to 3.5, it is clear that both algorithms are capable of detecting the obstacles in this case and, in fact, their performance are equally comparable. However, based on the simulation results shown in Figs. 3.4 and 3.5, the performance of **KGP** should be better than that of **UGP**, assuming that the *a priori* knowledge of the ground plane and camera calibration is available. In this case, though, the experimental result suggests that the available ground plane information and/or camera calibration information might not be accurate enough.

Table 3.5. Singular values of **UGP** algorithm using all ground points in the hallway indoor obstacle scene (coplanarity constraint).

SV's of $\boldsymbol{D}$	SV's of $[\boldsymbol{Db}]$
$\lambda_1 = 4.672\text{e+}5$	$\eta_1 = 4.672\text{e+}5$
$\lambda_2 = 6.183\text{e+}4$	$\eta_2 = 6.183\text{e+}4$
$\lambda_3 = 1.408\text{e+}3$	$\eta_3 = 1.408\text{e+}3$
$\lambda_4 = 3.838\text{e+}2$	$\eta_4 = 4.099\text{e+}2$
$\lambda_5 = 1.706\text{e+}2$	$\eta_5 = 2.469\text{e+}2$
$\lambda_6 = 7.016\text{e+}1$	$\eta_6 = 7.016\text{e+}1$
$\lambda_7 = 9.909\text{e}-1$	$\eta_7 = 1.317\text{e+}1$
$\lambda_8 = 2.888\text{e}-1$	$\eta_8 = 6.133\text{e}-1$
	$\eta_9 = 4.380\text{e}-2$

Fig. 3.9 is a sample of a sequence of stereo images of a real road scene with obstacles, taken by the stereo cameras onboard the Mobile Perception Lab [43]. The height of the cameras, H, is 7.8 ft. Again, only the left image is shown for each stereo frame. **EGP** was run on this sequence. The first three image pairs Fig. 3.9(a), (b), (c) are used to give an initial estimate of the state vector which includes information about the ground plane. It is assumed that there are no obstacles in these scenes. Fig. 3.9(d) is a frame with five cones (ground truth height is 2.35 ft.) and a box (ground truth height is 1.50 ft.) on the road as obstacles. Fig. 3.9(e) is the next frame with the same obstacles. Table 3.6 lists the results of the **EGP** algorithm for frames (d) and (e). The goal is an improvement over time of the iterative estimation of the ground plane and other parameters and a reduction in the error in the estimation of the heights of the six obstacle points that were detected. In this table, "predict" means using state vectors updated through the previous frame, and "estimate" means

Table 3.6. Height estimate errors. Note that absolute error is in ft. and relative error in %.

Point	(d) predict		(d) estimate		(e) predict		(e) estimate	
Labels	Abs	Rel	Abs	Rel	Abs	Rel	Abs	Rel
1	0.50	21.3%	0.30	12.8%	0.23	9.8%	0.11	4.7%
2	0.36	15.3%	0.14	6.0%	0.20	8.5%	0.13	5.5%
3	0.32	13.6%	0.09	3.8%	0.21	8.9%	0.15	6.4%
4	0.21	8.9%	0.07	3.0%	-0.01	-0.4%	0.0	0.0
5	0.23	9.8%	0.10	4.3%	-0.08	-3.4%	-0.05	-2.1%
6	0.19	12.7%	0.13	8.7%	-0.11	-7.3%	-0.07	-4.7%

using state vectors updated up to and including the current frame. The first pair of columns shows the estimates of the absolute and relative errors (respectively) in the heights of the obstacles using the initial estimate of the state vector from the first three frames. The second pair of columns shows the estimates based on the updated state vector from frame (d). The third pair of columns shows the estimates of the heights of the points in frame (e) using the state vector estimated from frame (d). The last pair of columns shows the estimates of the points in frame (e) based on the state vector updated from frame (e). From the data shown in this table, we can see that both absolute and relative errors generally decrease as the vehicle approaches the obstacles and as more frames are used. The relative error is of the same order as in the simulation.

We did not run **KGP** and **UGP** with this image sequence because there was no ground truth data available. In order to compare the three algorithms with the same data, another obstacle sequence was acquired in the robotics lab of the University of

(a) (b)

(c) (d)

(e)

Figure 3.9. A sample of a stereo sequence. The right images are not shown here. (a) - (c) are the first three images which show the scene without obstacles. (d) is a scene with obstacles. (e) is the next frame with the same obstacles.

Massachusetts. Selected right images from this sequence are as shown in Fig. 3.10. A pair of stereo cameras was mounted on a mobile robot. The camera height was 4.875 ft. and the cameras were tilted down towards the floor at an angle of 16.5°. The two cameras were parallel to each other, but were not carefully aligned (there was as much as a 6 - 10 pixel alignment error). The corners of the pieces of paper scattered on the floor are tracked as feature points. The obstacles in this sequence are two recycling bins and a box. Fig. 3.10(a) is the initial frame for the ground plane, as required for **EGP**. Fig. 3.10(b) to (e) are four consecutive frames with the obstacles as described earlier.

We use this sequence data (Fig. 3.10) to compare the performance of the three algorithms. Table 3.7 lists the single statistic value $\frac{\sigma_{min}(\boldsymbol{D})}{\sigma_{min}(\boldsymbol{Db})}$ for all five images of Fig. 3.10 for all the points (row 2) and for just the ground points (row 3) for the **KGP** algorithm. Table 3.8 is similar for the **UGP** algorithm. From these two tables, it can be seen that this statistic clearly separates the ground plane points from non-ground plane points. This is obvious because the statistics in the third row (ground points) are much larger than those in the second row (ground and non-ground points).

Table 3.9 shows the performance of **EGP** algorithm on the sequence shown in Fig. 3.10. The first column is the ground truth heights of the eight obstacle points selected in the images. The next eight columns are the estimated absolute and relative errors of the heights of the eight points with respect to their ground truth heights at the frames corresponding to Fig. 3.10(b), (c), (d), (e), respectively. Note that the errors do not decrease as the sensor gets closer to the obstacles. This implies that the errors at the frame shown in Fig. 3.10(b) are already at the limit, i.e. this amount of

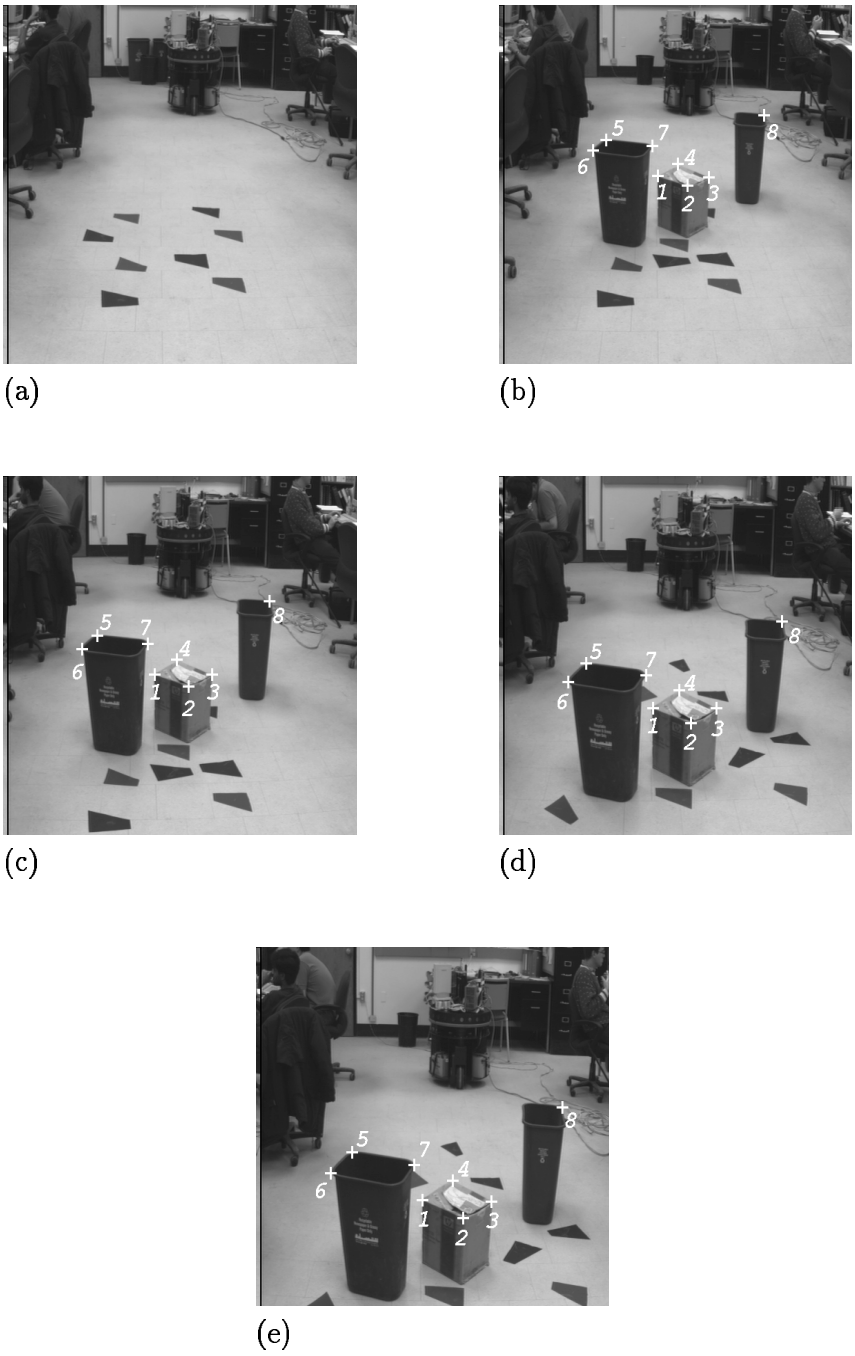

Figure 3.10. A sample of a stereo sequence of an indoor scene. The right images are not shown here. (a) is the ground plane scene. (b) - (e) are four consecutive frames with obstacles.

Table 3.7. $\sigma_{min}(\boldsymbol{D})/\sigma_{min}(\boldsymbol{Db})$ values for images in the robotics lab scene with **KGP**.

	(a)	(b)	(c)	(d)	(e)
All points		2.371	2.395	1.726	1.493
All ground points	11.351	10.128	9.642	11.958	10.233

Table 3.8. $\sigma_{min}(\boldsymbol{D})/\sigma_{min}(\boldsymbol{Db})$ values for images in the robotics lab scene with **UGP**.

	(a)	(b)	(c)	(d)	(e)
All points		2.199	2.911	2.118	1.646
All ground points	5.568	6.006	8.142	17.050	9.958

error comes from fixed error sources, e.g. mechanical measurement error in the camera height, and thus cannot be reduced by using more new frame information. Hence, there is no improvement when the robot is closer to the obstalces. The relative errors in this experiment are larger than those in Fig. 3.9, because no camera alignment procedure was used. However, even with this large alignment error (6 to 10 pixels), most of the relative errors are still within 5%.

3.6 Summary of Qualitative and Quantitative Obstacle Detection

The work presented in this chapter compares three different algorithms for obstacle detection using multiple images. These algorithms use different information about the environment. **KGP** assumes knowledge of the ground plane and camera parameters. **UGP** only assumes that the ground plane can be approximated by a plane (plane parameters

Table 3.9. Performance of **EGP** for images in the robotics lab scene.

Point Labels	Ground Truth	Absolute and Relative Errors							
		(b)		(c)		(d)		(e)	
1	0.958	-0.021	-2.19%	0.009	0.939%	0.035	3.65%	0.038	3.97%
2	0.969	0.032	3.30%	-0.010	-1.03%	-0.027	-2.79%	0.015	1.55%
3	0.969	-0.080	-8.26%	-0.080	-8.26%	-0.062	-6.40%	-0.045	-4.64%
4	0.979	-0.035	-3.58%	0.009	9.19%	-0.007	-7.15%	0.022	2.25%
5	1.677	0.122	7.27%	-0.032	-1.91%	0.275	16.4%	0.190	11.3%
6	1.677	0.005	0.298%	0.133	7.93%	0.185	11.0%	0.136	8.11%
7	1.677	-0.013	-0.775%	0.058	3.46%	0.073	4.35%	0.026	1.55%
8	1.656	-0.079	-4.77%	-0.095	-5.74%	0.028	1.69%	-0.118	-7.13%

unknown). **EGP** uses partially calibrated stereo cameras to compute obstacle height directly and Kalman filtering to estimate the ground plane. The simulation results predict that just on the basis of noise in the ground plane variation, if the ground plane really were known precisely, the first algorithm (**KGP**) would perform better than the second (**UGP**). However, the experiments on indoor scenes show that the performance of the two algorithms is comparable. One explanation is that errors in estimating the camera parameters and ground plane are significant compared to the other errors or that quantization errors dominate. In future experiments, we hope to be able to compare these algorithms on rougher terrain to determine if the *a priori* information would be useful in those cases.

In terms of robustness with respect to ground plane variation, the simulation shows that the **EGP** algorithm performs better than either of the other two. This algorithm does have many other advantages as well, since it is adaptive to changes in the ground plane and the Kalman filter updates can be computed quickly. However, this algorithm assumes that partially calibrated stereo cameras are available as resources, and that the height above the ground plane of one of them is known. In cases where partially calibrated stereo sensors are

not available, then algorithms like **KGP** might be sufficient, depending upon the expected operating scenario and whether or not a simple indication of the existence of an obstacle is sufficient. Note that although the **EGP** algorithm only requires partial calibration, it is still able to recover the heights of points above the ground plane.

CHAPTER 4

MODEL ACQUISITION AND EXTENSION: A HOMOGRAPHY MAPPING BASED APPROACH

4.1 Introduction

Model acquisition and extension is a necessary part of a complete robotic navigation system, since one would expect a mobile robot to build a model of the environment as it explores its "world". Techniques for model acquisition and extension can also be applied to other areas such as aerial image interpretation, object recognition, etc. In this chapter, we apply a homography mapping based approach to solve the problem of model acquisition and extension. Specifically, we address the problem of model acquisition and extension by reconstructing a 3D scene based on externally uncalibrated cameras, while assuming internal calibration is known. The problem is: given a correspondence of two sets of coplanar points from two unknown camera poses, reconstruct 3D points in Euclidean space. Thus, in the context of this dissertation, *model acquisition* means solving for the relative geometry between the two externally uncalibrated cameras, and/or the 3D plane that gives rise to the coplanar correspondences in the two images; *model extension* means reconstruction of any 3D point based on the solved relative geometry between the two cameras.

3D reconstruction is an important problem and still remains very difficult. Over the years, this problem has been related to structure from motion, stereo vision, pose determination, etc., and has been addressed by many researchers [66, 52, 68]. Recently, projective geometry has been used to perform 3D reconstruction (based on uncalibrated cameras) up to an unknown projective model [24, 44, 52, 83, 100, 91, 96, 69].

Faugeras [24] and Hartley *et al* [44, 52] are two of the earliest efforts to address the problem of 3D reconstruction based on two uncalibrated images. Faugeras showed that given five noncoplanar correspondences and epipoles, or given the fundamental matrix, 3D structures can be reconstructed up to a collineation of projective transforms. This result was then extended to the affine case, where he showed that given four noncoplanar points and epipoles, 3D structures can be recovered under an affine transform with three unknown parameters. Hartley *et al* [44, 52] independently arrived at the same conclusion by using matrix theory to linearly decompose the essential matrix. Previous research results [73, 114] showed that if the internal calibration of cameras is known, then it is possible to determine the relative motion (or geometry) between the two cameras and the relative locations of the 3D points corresponding to the matched points in the two views from the essential matrix, which needs at least eight correspondences. Hartley [44] then showed that this is also true even when the focal lengths of the two cameras are unknown, and Hartley *et al* [52] went on to show that if the internal calibration of cameras is completely unknown, it is *not* possible to recover the relative geometry and locations of 3D points unambiguously in a Euclidean space. In this case, the recovered 3D locations of the points and the camera geometry are subject to a 4×4 projective transform matrix, and absolute Euclidean coordinates of the 3D points can be computed only when a set of ground control points are known. Later Mohr *et al* [83] solved the same problem by making full use of the redundancy in multiple images to directly solve for a global least mean square solution. Shashua [100]

explored a new projective invariant, which he refered to as *projective depth*. Using this invariant, he showed that given four noncoplanar correspondences and epipoles of two views, 3D reconstruction can be achieved under either a projective transformation, or an affine transformation, depending on how the views were produced. As compared with previous work, the main contribution of this work (other than the exploration of the projective depth invariant) is that orthographic and perspective projections are treated in the same manner, and the computation does not need to recover the camera transformation first (i.e. structure without motion). Ponce *et al* [91] discussed several different cases under a projective transform and proposed algorithms to reconstruct 3D structures. Some of them assume weak calibration, i.e. known epipoles, while others do not. More recently, Shashua and Navab [102] presented a novel theory called *relative affine structure*. Based on this theory, they developed a unified method for recovering 3D structures. Again, at a minimum, two views, four noncoplanar points, and the epipoles are required. The 3D structure is obtained under an affine transform with three unknown parameters. Hartley [49] also extended his previous reconstruction method based on points to a new algorithm based on lines. Still, this reconstruction is under projective space, and it assumes at least three views. Sawhney [96] attacked the same problem slightly differently. Instead of aiming at point-based 3D structure reconstruction, he used planar motion parallax and image warping techniques to present a unified framework for intrinsic 3D shape reconstruction for three projection models: weak, para, and full perspective. A similar treatment but using a different approach was also independently done by Kumar and Anandan [69] using motion parallax. Most recently, Hartley [48, 51] proposed to use a *trifocal tensor* to reconstruct 3D scene up to a projectivity from three uncalibrated views. This is a linear algorithm, and is a unified approach in the sense that it can apply either to lines or to points (or to the combination of lines and points). In particular, he showed that this trifocal tensor is essentially identical to

the set of coefficients introduced by Shashua [101] to effect point transfer in the three view case. Another major contribution of this work is that Hartley showed that the minimun requirement for 3D projective reconstruction from 3 views with the same camera is either 7 point correspondences, or 13 line correspondences, or any combinations in between[1]. In the point case, the minimum number 7 is consistent with the results of Maybank *et al* [82] and Faugeras [25], except that in their work, a Euclidean reconstruction was possible because they explicitly recovered the internal camera parameters. However, they needed at least three motions, and assumed that the same camera was used to acquire all the views. In the line case, the minimum number 13 is consistent with Weng *et al* [123], except that their work assumed that the cameras were calibrated.

In summary, it has been proven that based on two uncalibrated views, it is impossible to recover the 3D scene in a Euclidean space. The best one can do is to recover the 3D scene up to an arbitrary projectivity, and to do so requires at least seven correspondences. If three views are available, and assuming the same camera is used for all three, then it is possible to get a scaled Euclidean 3D reconstruction [45, 47].

In this chapter, we present a method that recovers the 3D structures in a Euclidean space. This is an extension of recent work by Zhang and Hanson [129]. Like the previous work, at a minimum our method also needs two views and four correspondences. Unlike the previous work, which assumes four noncoplanar correspondences, we here assume four *coplanar* correspondences that are not collinear. Moreover, the 3D scene structures are recovered in Euclidean space up to two solutions with a uniform scaling factor, as opposed to a family of solutions in a projective space; this scaling parameter has an explicit physical meaning which is the distance of the first camera center from the 3D plane formed by the four points given in the correspondences. If this distance is known *a priori*, then a complete

[1]The constraint is [51]: #lines + 2#points $\geq$ 13, where # means "the number of".

3D Euclidean reconstruction can be obtained up to two solutions. Without any *a priori* knowledge, it is shown that these two solutions are indistinguishable. The other difference from most of the recent work in 3D reconstruction is that here we assume that the internal calibration is known, as opposed to assuming weak calibration, or completely unknown calibration. The basic idea is that we first find the homography mapping between the two cameras using the four given correspondences. The homography matrix is decomposed to obtain the relative pose between the two cameras. Finally, 3D structures are reconstructed based on the recovered relative pose. We will show in this chapter that this assumption is necessary for 3D reconstruction in Euclidean space.

Note that this proposed method is different from early work on 3D reconstruction based on the essential matrix [73, 114]. The differences are reflected in two ways. First, the homography matrix is not the same as the essential matrix. Second, using the essential matrix gives only one constraint for each correspondence. Thus the minimum number of correspondences required using the essential matrix is 8, if linear constraints are used [73, 114], or 7 if nonlinear constraints are used [58]. However, with the homography matrix, each correspondence produces two equations, which reduces the minimum number of required correspondences to 4.

Faugeras and Lustman [27] and Maybank [81] showed that a homographic transformation between two cameras can be decomposed, and in general, there are two real solutions. Weng *et al* [122] also arrived at the same conclusion. However, they only applied the decomposition of the homography matrix to planar 3D reconstruction. In this chapter, we propose a different way to decompose this matrix and perform a case by case analysis of different geometric situations. Finally, different analytical closed-form solutions are developed based on all the possible cases. This enables us to reconstruct any 3D scene, and owing to this

closed-form solution, the computation is very inexpensive and fast. It is also shown that this proposed algorithm is optimal.

This chapter is organized as follows. The next section briefly introduces the notion of a homography mapping between two cameras. Then we address the problem of how to form a homography matrix from the given coplanar correspondences. This is followed by the proposed method for decomposing the matrix (model acquisition), followed by a description of the algorithm for reconstructing the 3D scene based on this decomposition (model extension). Experimental results are then presented based on both simulation and real image data. Finally the optimality of this homography based 3D reconstruction algorithm is shown based on counting of number of degrees of freedom, and it is also shown that it should be possible to extend this algorithm to more than two view cases.

4.2 Homography Matrix Between Two Cameras

In this section, we briefly review the homography matrix and its related theory. Consider a planar 3D object under the general viewing configuration illustrated in Fig. 4.1. Note that the second camera could also be the first camera but moved to a new location.

Let us assume that the camera at the second position, O_2, has been first translated $\boldsymbol{t}$ from the position O_1, and then subjected to a rotation $\boldsymbol{R}$. Let us assume that both cameras are imaging the 3D plane π, which has a normal vector $\boldsymbol{n}$. Furthermore, assume everything is represented in the first camera's coordinate system. We use homogeneous coordinates to represent the 2D point $\boldsymbol{p}$ in the image plane for any 3D point $\boldsymbol{P} = (X, Y, Z)^T$. Its corresponding image point in the first camera retina is:

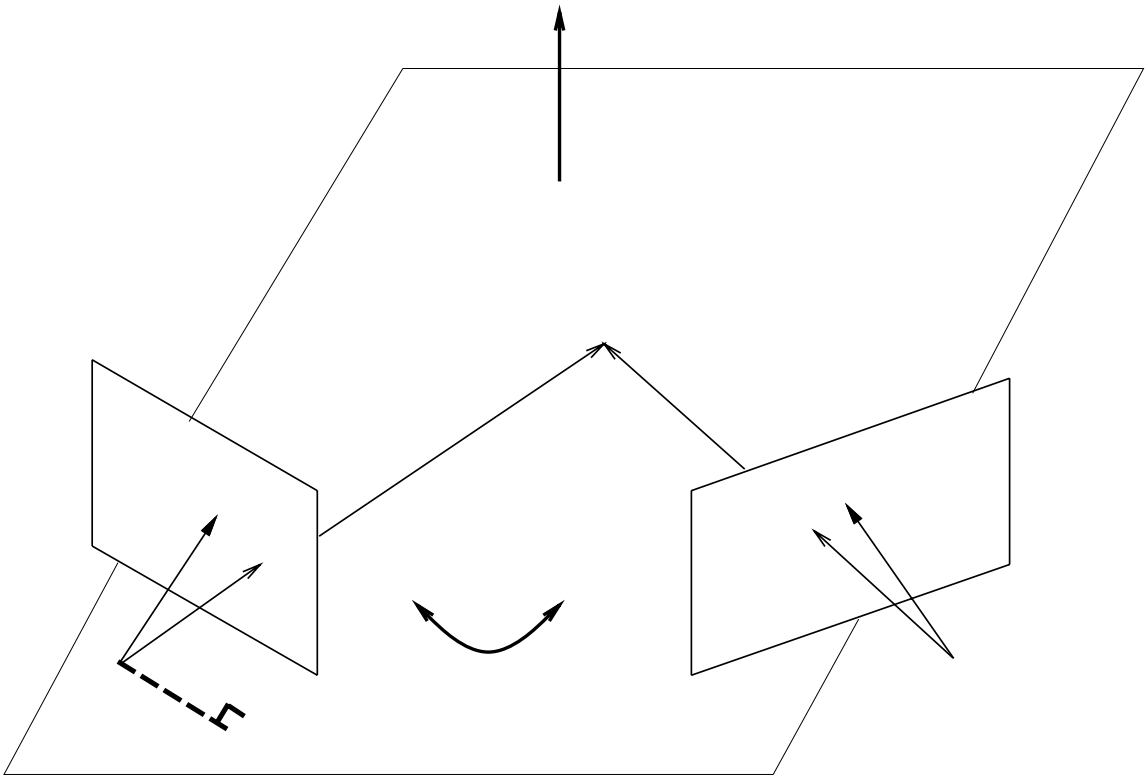

Figure 4.1. The geometry of a homography mapping.

$$\boldsymbol{p} \cong \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = w \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{y} \\ 1 \end{pmatrix} \tag{4.1}$$

where w is the standard scale factor. The above equations are under the constraints:

$$\boldsymbol{x} = \frac{X}{Z}, \quad \boldsymbol{y} = \frac{Y}{Z} \tag{4.2}$$

There is a similar relationship between the 3D point $\boldsymbol{P}$ and its projection point $\boldsymbol{p'}$ in the second camera retina. Now let H be the perpendicular distance from the center of projection of the first camera to the plane π. It can been shown [27] that if $\boldsymbol{P}$ is on plane π, i.e. if $\boldsymbol{P}$ satisfies:

$$\boldsymbol{P} \cdot \boldsymbol{n} = H \tag{4.3}$$

then there is a homographic mapping between the two corresponding projection points $\boldsymbol{p}$ and $\boldsymbol{p'}$ of this 3D point in the two camera images:

$$k\boldsymbol{p'} = \boldsymbol{A}\boldsymbol{p} \tag{4.4}$$

where $\boldsymbol{k}$ is a scaling factor accounting for the fact that the representations of $\boldsymbol{p}$ and $\boldsymbol{p'}$ are expressed in homogeneous coordinates. $\boldsymbol{A}$ is the homography mapping matrix, which is a 3×3 matrix and can be shown [27] to be[2]:

$$\boldsymbol{A} = \boldsymbol{R}(\boldsymbol{I} - \frac{\boldsymbol{t}\boldsymbol{n}^T}{H}) \tag{4.5}$$

where $\boldsymbol{I}$ is a 3×3 identity matrix.

[2]Note that since translation is performed first and then rotation, the mathematical form of $\boldsymbol{A}$ looks slightly different from that in [27]. A derivation of Eq. 4.5 is given in Appendix A.

In the following, the plane π that gives rise to the coplanar correspondences is referred to as the *reference plane*. Our goal is to solve for the motion (or relative pose) between the two cameras based on the homography matrix $\boldsymbol{A}$, and then reconstruct the 3D point $\boldsymbol{P}$.

4.3 Forming the Normalized Homography Matrix

Let $\boldsymbol{p}_i \Longleftrightarrow \boldsymbol{p}_i'$ be a correspondence in two arbitrarily externally uncalibrated cameras of an arbitrary 3D point $\boldsymbol{P}_i$ on some reference plane as shown in Fig. 4.1. Then based on Eq. 4.4, we have:

$$k_i \boldsymbol{p}_i' = \boldsymbol{A}\boldsymbol{p}_i \tag{4.6}$$

The scale factor can be eliminated by writing the matrix $\boldsymbol{A}$ (defined in Eq. 4.4) in the following normalized form:

$$\boldsymbol{A_0} = \begin{pmatrix} s_1 & s_2 & s_3 \\ s_4 & s_5 & s_6 \\ s_7 & s_8 & 1 \end{pmatrix} \tag{4.7}$$

$\boldsymbol{A_0}$ differs from $\boldsymbol{A}$ by a uniform scale factor; that is

$$\boldsymbol{A} = \lambda \boldsymbol{A_0} \tag{4.8}$$

Thus, by replacing $\boldsymbol{A}$ with $\boldsymbol{A_0}$, Eq. 4.6 is still valid. Using the homogeneous coordinate representation defined in Eq. 4.1, and eliminating the scale factor k_i, Eq. 4.6 becomes two equations:

$$x_i{}' = \frac{x_i s_1 + y_i s_2 + s_3}{x_i s_7 + y_i s_8 + 1} \tag{4.9}$$

$$y_i{}' = \frac{x_i s_4 + y_i s_5 + s_6}{x_i s_7 + y_i s_8 + 1} \tag{4.10}$$

Now if we are given n coplanar correspondences in the two camera images, the following linear system can be obtained:

$$\begin{pmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1 x_1{}' & -y_1 x_1{}' \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1 y_1{}' & -y_1 y_1{}' \\ \ldots & & & & & & & \\ x_n & y_n & 1 & 0 & 0 & 0 & -x_n x_n{}' & -y_n x_n{}' \\ 0 & 0 & 0 & x_n & y_n & 1 & -x_n y_n{}' & -y_n y_n{}' \end{pmatrix} \begin{pmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \\ s_8 \end{pmatrix} = \begin{pmatrix} x_1{}' \\ y_1{}' \\ \ldots \\ x_n{}' \\ y_n{}' \end{pmatrix} \tag{4.11}$$

Therefore, if we have at least four coplanar but non-collinear correspondences, the linear system will have a unique solution. If more than four coplanar correspondences are given, a least mean squares technique can be used to solve Eq. 4.11 to obtain the normalized homography matrix $\boldsymbol{A_0}$. The next section gives a method to solve for the relative pose between the two cameras based on this normalized homography matrix.

4.4 Model Acquisition: Closed-form Solutions to the Decomposition of $\boldsymbol{A_0}$

In the following derivation, closed-form solutions are obtained for recovering the transformations $\boldsymbol{R}$ and $\boldsymbol{t}$ and the normal of the ground plane $\boldsymbol{n}$ from $\boldsymbol{A_0}$.

We start with $\boldsymbol{A_0}$ as defined in Eq. 4.7. $\boldsymbol{A_0}$ and $\boldsymbol{A}$ (as defined in Eq. 4.5) are related by an unknown scale factor λ:

$$\lambda \boldsymbol{A_0} = \boldsymbol{A} = \boldsymbol{R}(\boldsymbol{I} - \frac{\boldsymbol{t}\boldsymbol{n}^T}{H}) \tag{4.12}$$

Define the following:

$$\boldsymbol{t_0} \stackrel{def}{=} -\frac{\boldsymbol{t}}{H} \tag{4.13}$$

$$k^2 \stackrel{def}{=} \boldsymbol{t_0} \cdot \boldsymbol{t_0}, \quad k > 0 \tag{4.14}$$

$$p \stackrel{def}{=} \boldsymbol{n} \cdot \boldsymbol{t_0} \tag{4.15}$$

Now we have:

$$\lambda \boldsymbol{A_0} = \boldsymbol{R}(\boldsymbol{I} + \boldsymbol{t_0}\boldsymbol{n}^T) \tag{4.16}$$

and by multiplying through by the transpose of Eq. 4.16, we obtain:

$$\lambda^2 {\boldsymbol{A_0}}^T \boldsymbol{A_0} = \boldsymbol{I} + \boldsymbol{n}{\boldsymbol{t_0}}^T + \boldsymbol{t_0}\boldsymbol{n}^T + k^2 \boldsymbol{n}\boldsymbol{n}^T \tag{4.17}$$

Now we will find the analytical form of the three eigenvalues and their corresponding eigenvectors. We first assume that $\boldsymbol{t_0}$ and $\boldsymbol{n}$ are not aligned. This assumption will be relaxed later on to discuss those special cases when these two vectors are aligned. By the definitions of eigenvalues and eigenvectors, it is easy to determine that the first eigenvector of $\lambda^2 {\boldsymbol{A_0}}^T \boldsymbol{A_0}$ is:

$$\boldsymbol{v_1} = \boldsymbol{t_0} \times \boldsymbol{n} \tag{4.18}$$

and the first eigenvalue is:

$$\eta_1 = 1 \tag{4.19}$$

Since $\lambda^2 \boldsymbol{A_0}^T \boldsymbol{A_0}$ is symmetric, as indicated in Eq. 4.17, the other two eigenvectors must be on the plane perpendicular to $\boldsymbol{v_1}$. It follows that the other two eigenvectors can be assumed to be of the form:

$$\boldsymbol{v_{2,3}} = a\boldsymbol{t_0} + b\boldsymbol{n} \tag{4.20}$$

Thus,

$$\begin{aligned} \lambda^2 \boldsymbol{A_0}^T \boldsymbol{A_0}(a\boldsymbol{t_0} + b\boldsymbol{n}) &= (a + ap + b)\boldsymbol{t_0} + (apk^2 + ak^2 + bk^2 + bp + b)\boldsymbol{n} \\ &\overset{def}{=} \eta_{2,3}(a\boldsymbol{t_0} + b\boldsymbol{n}) \end{aligned} \tag{4.21}$$

where $\eta_{2,3}$ are the two corresponding eigenvalues.

It follows that

$$a + ap + b = \eta_{2,3} a \tag{4.22}$$

and

$$apk^2 + ak^2 + bk^2 + bp + b = \eta_{2,3} b \tag{4.23}$$

By solving the above two equations, we have:

If $p \neq -1$,

$$\eta_2 = 1 + p + \frac{2k(p+1)}{-k + \sqrt{k^2 + 4(p+1)}} \tag{4.24}$$

$$\eta_3 = 1 + p + \frac{2k(p+1)}{-k - \sqrt{k^2 + 4(p+1)}} \tag{4.25}$$

and

$$\boldsymbol{v_2} = \frac{-k + \sqrt{k^2 + 4(p+1)}}{2k(p+1)}\boldsymbol{t_0} + \boldsymbol{n} \tag{4.26}$$

$$\boldsymbol{v_3} = \frac{-k - \sqrt{k^2 + 4(p+1)}}{2k(p+1)}\boldsymbol{t_0} + \boldsymbol{n} \tag{4.27}$$

If $p = -1$,

$$\eta_2 = k^2 \tag{4.28}$$

$$\eta_3 = 0 \tag{4.29}$$

and

$$\boldsymbol{v_2} = \frac{1}{k^2}\boldsymbol{t_0} + \boldsymbol{n} \tag{4.30}$$

$$\boldsymbol{v_3} = \boldsymbol{t_0} \tag{4.31}$$

It is easy to verify that $\boldsymbol{v_2} \cdot \boldsymbol{v_3} = 0$.

To simplify the mathematical expressions, let us introduce the following notation:

$$\gamma \stackrel{def}{=} \frac{-k + \sqrt{k^2 + 4(p+1)}}{2k(p+1)} \tag{4.32}$$

$$\theta \stackrel{def}{=} \frac{-k - \sqrt{k^2 + 4(p+1)}}{2k(p+1)} \tag{4.33}$$

Now we rewrite the other two eigenvectors and eigenvalues in the case of $p \neq -1$:

$$\boldsymbol{v_2} = \gamma \boldsymbol{t_0} + \boldsymbol{n} \tag{4.34}$$

$$\boldsymbol{v_3} = \theta \boldsymbol{t_0} + \boldsymbol{n} \tag{4.35}$$

$$\eta_2 = \frac{1}{\gamma} + p + 1 \tag{4.36}$$

$$\eta_3 = \frac{1}{\theta} + p + 1 \tag{4.37}$$

Based on the solutions of the obtained eigenvalues, we now examine the various cases for the three eigenvalues. By the definitions of k and p in Eq. 4.14 and Eq. 4.15, it is obvious that the following two constraints are always valid:

$$k \geq p \tag{4.38}$$

$$||k|| \geq ||p|| \tag{4.39}$$

We also know that $\lambda^2 \boldsymbol{A_0}^T \boldsymbol{A_0}$ is a real symmetric matrix, and thus it always has real eigenvalues. Therefore, a third constraint is:

$$k^2 + 4(p+1) \geq 0 \tag{4.40}$$

which means that $p \geq -\frac{k^2}{4} - 1$.

Based on the above three constraints, it can be shown that

$$\eta_2 \geq \eta_1 = 1$$

where equality holds iff $p = -2, k = 2$ or $p = -1, k = 1$; and

$$\eta_3 \leq \eta_1 = 1$$

and equality holds iff $p = k$.

Note that those conditions that make the above constraints become equalities violate our previous assumption that t_0 and n are not aligned. Therefore, in the general case that the two vectors are not aligned, we have a strict inequality[3]:

$$\eta_2 > \eta_1 = 1 > \eta_3$$

The following section gives an analytical closed-form solution for the decomposition of the $\boldsymbol{A_0}$ matrix in the general case. Various special cases when $\boldsymbol{t_0}$ and $\boldsymbol{n}$ are aligned are then discussed.

4.4.1 General Case: $\eta_2 > \eta_1 = 1 > \eta_3$

A singular value decomposition technique SVD [92] is used to decompose $\boldsymbol{A_0}$ into $\boldsymbol{t}$, $\boldsymbol{R}$, and $\boldsymbol{n}$. Assume the three singular values are:

$$\rho_2{}' > \rho_1{}' > \rho_3{}'$$

and their corresponding orthonormal eigenvectors are $\pm\boldsymbol{u_2}, \pm\boldsymbol{u_1}, \pm\boldsymbol{u_3}$.

Since

$$\lambda\rho_1{}' = \eta_1 = 1$$

the scale factor can be determined as:

$$\lambda = \frac{1}{\rho_1{}'} \tag{4.41}$$

Now we normalize the other two singular values:

[3]In the following discussion, we follow the convention that $\eta_2 \geq \eta_1 \geq \eta_3$ and similar subscript convention for the corresponding eigenvectors.

$$\rho_2 \stackrel{def}{=} \lambda {\rho_2}' \tag{4.42}$$

$$\rho_3 \stackrel{def}{=} \lambda {\rho_3}' \tag{4.43}$$

By recalling the relationship between the singular values and the eigenvalues [34], we have:

$$\eta_2 = {\rho_2}^2 \tag{4.44}$$

and

$$\eta_3 = {\rho_3}^2 \tag{4.45}$$

Solving these two equations, we obtain:

$$k = \rho_2 - \rho_3 \tag{4.46}$$

and

$$p = \rho_2 \rho_3 - 1 \tag{4.47}$$

Now define:

$$\begin{aligned} \mu &\stackrel{def}{=} ||\boldsymbol{v_2}|| \\ &= \sqrt{(\gamma {\boldsymbol{t_0}}^T + \boldsymbol{n}^T)(\gamma \boldsymbol{t_0} + \boldsymbol{n})} \\ &= \sqrt{\gamma^2 k^2 + 2\gamma p + 1} \end{aligned} \tag{4.48}$$

$$\begin{aligned} \delta &\stackrel{def}{=} ||\boldsymbol{v_3}|| \\ &= \sqrt{(\theta {\boldsymbol{t_0}}^T + \boldsymbol{n}^T)(\theta \boldsymbol{t_0} + \boldsymbol{n})} \\ &= \sqrt{\theta^2 k^2 + 2\theta p + 1} \end{aligned} \tag{4.49}$$

Thus,

$$\pm \boldsymbol{u_2} = \frac{\gamma \boldsymbol{t_0} + \boldsymbol{n}}{\mu} \tag{4.50}$$

$$\pm \boldsymbol{u_3} = \frac{\theta \boldsymbol{t_0} + \boldsymbol{n}}{\delta} \tag{4.51}$$

Solving the above four sets of two equations, we finally have four solutions:

$$\boldsymbol{t_0} = \pm \frac{\mu \boldsymbol{u_2} - \delta \boldsymbol{u_3}}{\gamma - \theta} \tag{4.52}$$

$$\boldsymbol{n} = \pm \frac{\gamma \delta \boldsymbol{u_3} - \theta \mu \boldsymbol{u_2}}{\gamma - \theta} \tag{4.53}$$

and

$$\boldsymbol{t_0} = \pm \frac{\mu \boldsymbol{u_2} + \delta \boldsymbol{u_3}}{\gamma - \theta} \tag{4.54}$$

$$\boldsymbol{n} = \mp \frac{\gamma \delta \boldsymbol{u_3} + \theta \mu \boldsymbol{u_2}}{\gamma - \theta} \tag{4.55}$$

Recalling that the solutions should be "real" in the sense that they should satisfy the constraint of $\boldsymbol{P} \cdot \boldsymbol{n} > 0$, two of the above four solutions are easily removed, i.e. we only keep those two solutions for which $\boldsymbol{t_0}$ and $\boldsymbol{n}$ take the same signs under the constraint that $\boldsymbol{P} \cdot \boldsymbol{n} > 0$. Therefore, we have obtained the two real solutions to this decomposition.

It can easily be verified that $\boldsymbol{n} \cdot \boldsymbol{n} = 1$.

Using these solutions, the rotation $\boldsymbol{R}$ (based on Eq. 4.5) can be obtained:

$$\boldsymbol{R} = \lambda \boldsymbol{A_0} (\boldsymbol{I} + \boldsymbol{t_0} \boldsymbol{n}^T)^{-1} \tag{4.56}$$

as well as the translation $\boldsymbol{t}$ (based on Eq. 4.13):

$$\boldsymbol{t} = -H\boldsymbol{t_0} \tag{4.57}$$

Up to this point, we have found the two closed-form analytical solutions to the decomposition of the matrix $\boldsymbol{A_0}$ for this case.

One question that remains is whether or not these two solutions can be differentiated so that the "true" solution can be uniquely identified. Without any *a priori* knowledge, the only recourse is to check the distribution of the two solutions in terms of the cheirality [46] constraint w.r.t. the reference plane, i.e. check whether or not the two solutions are on the same side of the plane.

The necessary and sufficient condition for the case that the two solutions lie on opposite sides of the reference plane, i.e. the camera center goes to the other side of the reference plane after motion, is:

$$\boldsymbol{t_0} \cdot \boldsymbol{n} < -1$$

Given Eqs. 4.52 to 4.55, for both solutions $i = 1, 2$, it can be shown that:

$$\boldsymbol{t_{0i}} \cdot \boldsymbol{n_i} = p$$

which again is consistent with the definition in Eq. 4.15. This result means that if $p < -1$, the solutions straddle the reference plane; otherwise, both solutions are on the same side of the reference plane. Consequently, the two solutions are intrisically indistinguishable. Therefore, this ambiguity of the two solutions cannot be resolved if there is no further *a priori* information available.

It is worth noting that if the condition $\boldsymbol{t_{0i}} \cdot \boldsymbol{n_i} < -1$ is valid, then this is the case that for both solutions, the cameras cross the reference plane after motion. This case is possible only if the reference plane is "transparent", i.e. all the features on this plane can be viewed

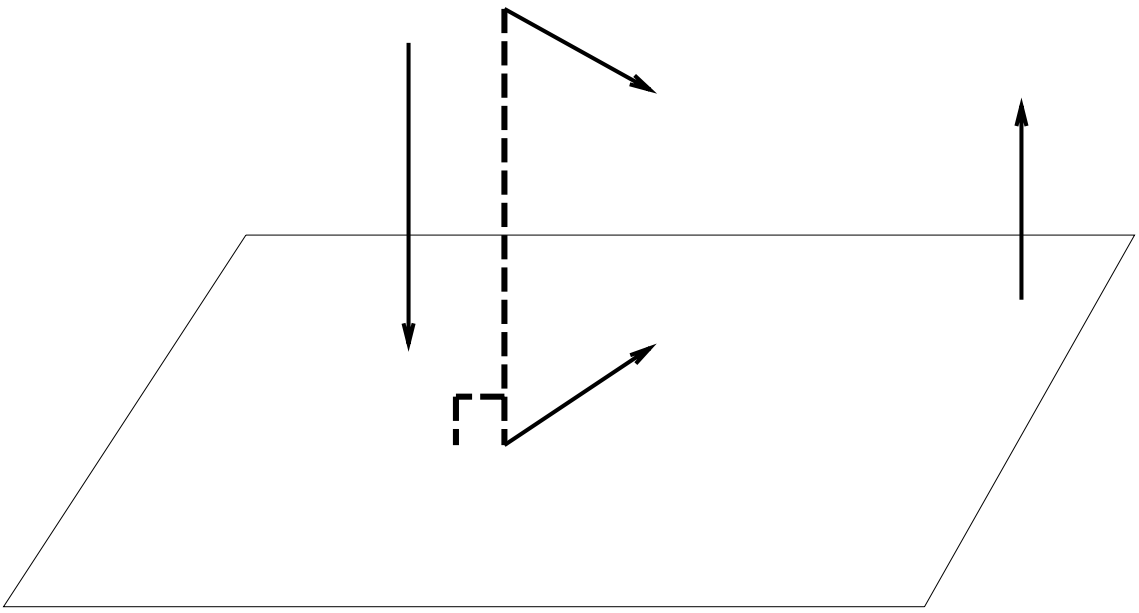

Figure 4.2. Geometric interpretation of case $k = -p = 1$.

from both sides of it. In this case the homography mapping is still valid, and thus, the recovered solutions are also valid.

4.4.2 Case $k = -p = 1$

This is a physically unrealizable case. To see this, refer to Fig. 4.2. The constraints $p = -1$ and $k = 1$ mean that the camera translates along $-\boldsymbol{n}$ a distance H; this puts the center of projection of the camera exactly on the reference plane. Regardless of its rotation, it will always see the reference plane as a line, which violates our assumption that the image correspondences should not be collinear.

4.4.3 Case $k = p$

The constraint of $k = p$ implies that $\boldsymbol{t_0} = k\boldsymbol{n}$. Substituting this into Eq. 4.17, we obtain:

$$\lambda^2 \boldsymbol{A_0}^T \boldsymbol{A_0} = \boldsymbol{I} + 2k\boldsymbol{n}\boldsymbol{n}^T + k^2\boldsymbol{n}\boldsymbol{n}^T \tag{4.58}$$

Obviously, any two arbitrary vectors perpendicular to $\boldsymbol{n}$ satisfy the eigenvector definition of the matrix $\lambda^2 \boldsymbol{A_0}^T \boldsymbol{A_0}$, and they have equal corresponding eigenvalues with value 1. Moreover, since $\lambda^2 \boldsymbol{A_0}^T \boldsymbol{A_0}$ is symmetric, the two eigenvectors should also be perpendicular to each other. Specifically, let us define $\boldsymbol{v_1}$ and $\boldsymbol{v_3}$ as the two eigenvectors, and η_1 and η_3 their corresponding eigenvalues. Thus, we have,

$$\eta_1 = \eta_3 = 1$$

$$\boldsymbol{v_1} \perp \boldsymbol{v_3}, \quad \boldsymbol{v_1} \perp \boldsymbol{n}, \quad \boldsymbol{v_3} \perp \boldsymbol{n}$$

Therefore, the third eigenvector must be $\boldsymbol{n}$, and its corresponding eigenvalue is $(k+1)^2$, i.e.

$$\eta_2 = (k+1)^2$$

$$\boldsymbol{v_2} = \boldsymbol{n}$$

Now let us denote the three singular values of $\boldsymbol{A}$ as:

$$\rho_2{}' > \rho_1{}' = \rho_3{}'$$

Thus, the normalization factor is:

$$\lambda = \frac{1}{\rho_1{}'} = \frac{1}{\rho_3{}'}$$

and the three singular values are normalized as:

$$\rho_i = \lambda \rho_i{}', \quad i = 1, 2, 3$$

Then, since

$$\eta_2 = (k+1)^2 = {\rho_2}^2 \tag{4.59}$$

k and p can be obtained:

$$k = p = \rho_2 - 1 \tag{4.60}$$

Finally, $\boldsymbol{n}$ can be solved for:

$$\boldsymbol{n} = \pm \boldsymbol{u_2} \tag{4.61}$$

Since $\boldsymbol{n}$ has to satisfy the constraint $\boldsymbol{P_i} \cdot \boldsymbol{n} > 0$, only one of the above two solutions is valid. Hence, only one solution for $\boldsymbol{n}$ is obtained. The second solution occurs when $k = -p > 2$ in the next section. Consequently, $\boldsymbol{t_0}$ is:

$$\boldsymbol{t_0} = k\boldsymbol{n} \tag{4.62}$$

and $\boldsymbol{t}$ and $\boldsymbol{R}$ can also be uniquely obtained from Eqs. 4.56 and 4.57.

The geometric explanation of this case is shown in Fig. 4.3. The camera center first translates distance kH along direction $\boldsymbol{n}$, and then rotates $\boldsymbol{R}$. This is the first solution for the case of $\eta_2 > \eta_1 = \eta_3 = 1$.

4.4.4 Case $k = -p > 2$

The constraint $k = -p$ implies that $\boldsymbol{t_0} = k\boldsymbol{n}$. Thus, Eq. 4.17 becomes:

$$\lambda^2 {\boldsymbol{A_0}}^T \boldsymbol{A_0} = \boldsymbol{I} - 2k\boldsymbol{n}\boldsymbol{n}^T + k^2\boldsymbol{n}\boldsymbol{n}^T \tag{4.63}$$

Similarly, the three eigenvelues are:

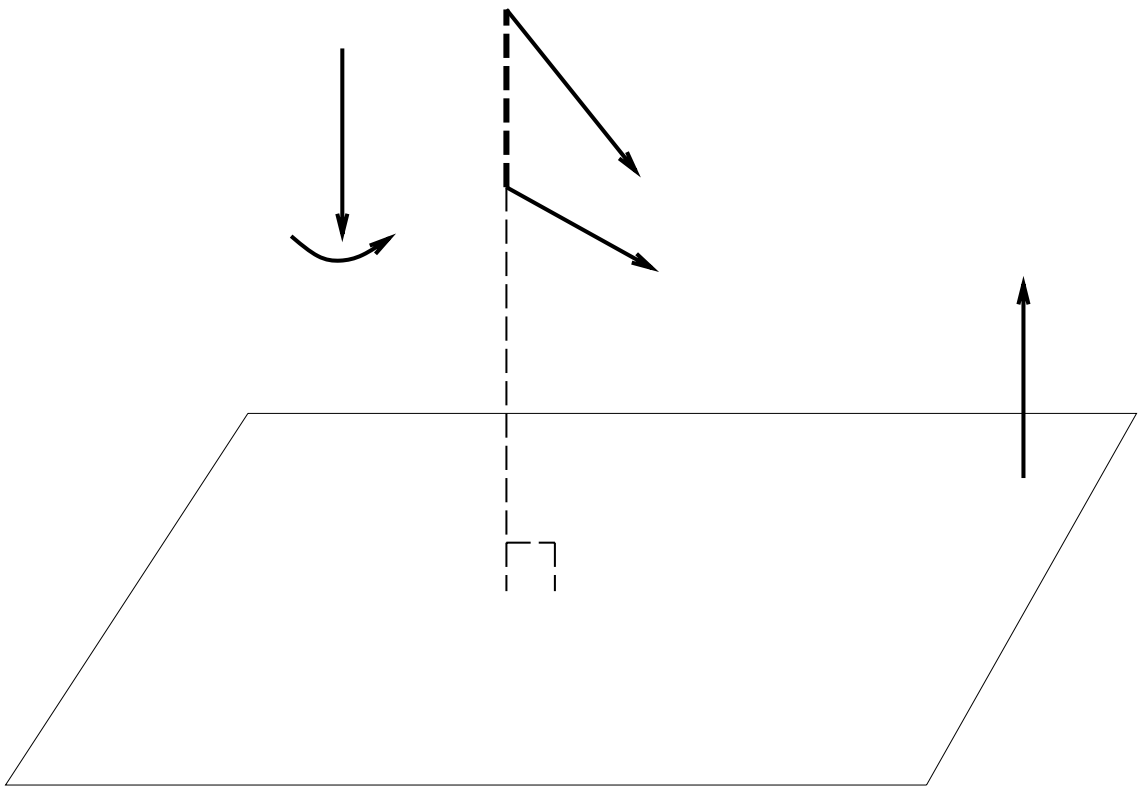

Figure 4.3. Geometric interpretation of case $k = p$.

$$\eta_2 = (k-1)^2 > \eta_1 = \eta_3 = 1$$

and the three eigenvectors are: $\boldsymbol{v_2} = \boldsymbol{n}$, with $\boldsymbol{v_1}$ and $\boldsymbol{v_3}$ two arbitrary vectors satisfying:

$$\boldsymbol{v_1} \perp \boldsymbol{n}, \quad \boldsymbol{v_3} \perp \boldsymbol{n}, \quad \boldsymbol{v_1} \perp \boldsymbol{v_3}$$

Thus, k and p are obtained as:

$$k = \rho_2 + 1 \tag{4.64}$$

$$p = -\rho_2 - 1 \tag{4.65}$$

Finally,

$$\boldsymbol{n} = \pm \boldsymbol{u_2} \tag{4.66}$$

subject to the constraint $\boldsymbol{P_i} \cdot \boldsymbol{n} > 0$, and

$$\boldsymbol{t_0} = -k\boldsymbol{n} \tag{4.67}$$

This unique solution obtained above, together with the unique solution derived for the case $k = p$, constitutes the two solutions for the case $\eta_2 > \eta_1 = \eta_3 = 1$. Fig. 4.4 (b) shows the geometric explanations of the case for $k = -p > 2$. The camera first translates along $-\boldsymbol{n}$ a distance kH, and then rotates $\boldsymbol{R}$. Since in this case $k > 2$, the new camera position must be on the other side of the reference plane.

4.4.5 Case $k = -p < 2$

This case is very similar to the case of $k = -p > 2$ except the three eigenvalues are:

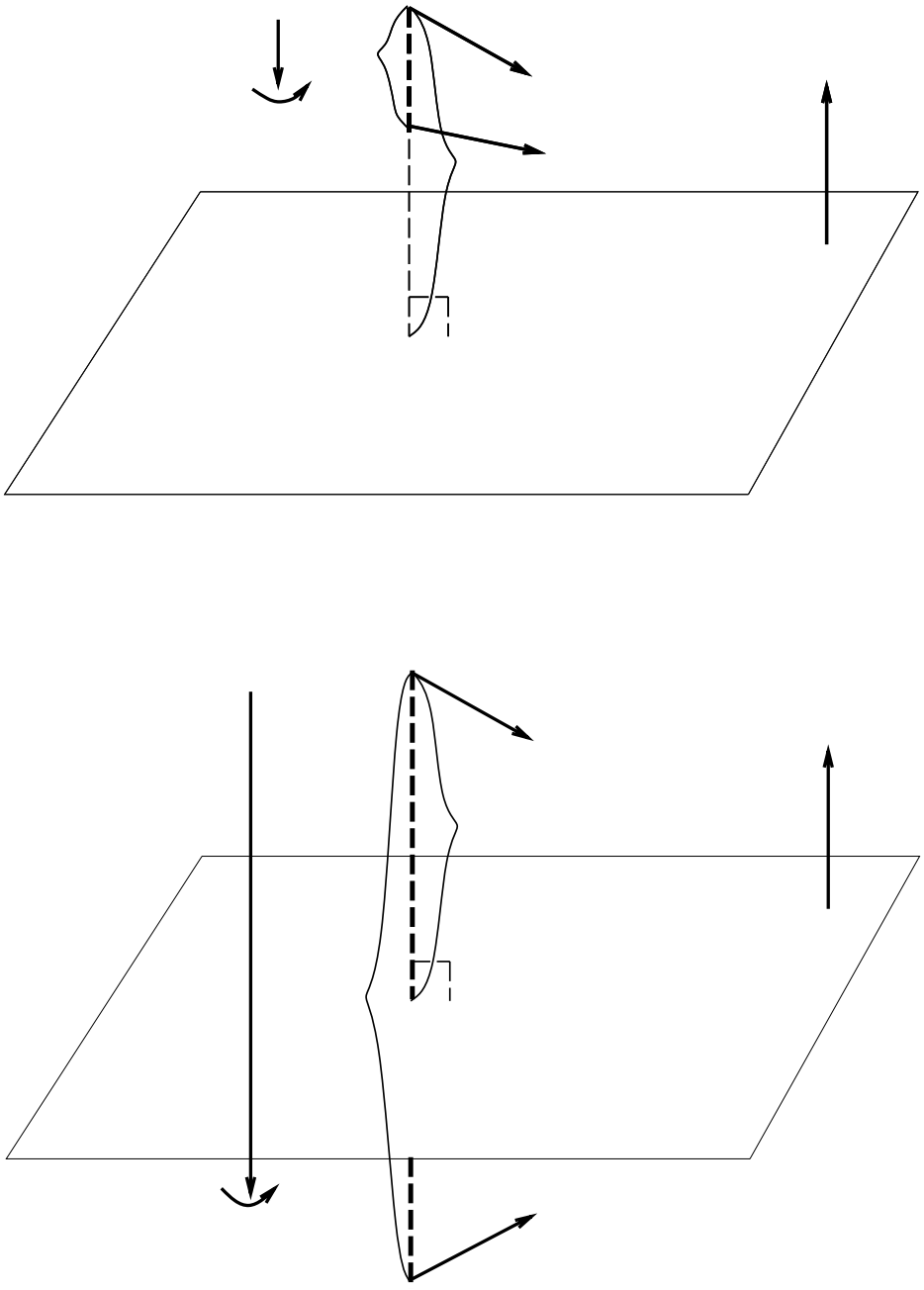

Figure 4.4. Geometric interpretation of case $k = -p$. (a) when $k < 1$ (b) when $k > 1$.

$$\eta_2 = \eta_1 = 1 > \eta_3 = (k-1)^2$$

and the three eigenvectors are: $\boldsymbol{v_3} = \boldsymbol{n}$ with $\boldsymbol{v_2}$ and $\boldsymbol{v}_1$ being two arbitrary vectors satisfying

$$\boldsymbol{v_2} \perp \boldsymbol{n}, \quad \boldsymbol{v_1} \perp \boldsymbol{n}, \quad \boldsymbol{v_2} \perp \boldsymbol{v_1}$$

Similarly, let the three singular values of $\boldsymbol{A}$ be:

$$\rho_2{}' = \rho_1{}' > \rho_3{}'$$

The normalization factor is:

$$\lambda = \frac{1}{\rho_2} = \frac{1}{\rho_1}$$

Thus, the singular values are normalized:

$$\rho_i = \lambda \rho_i{}'$$

Finally, we have

$$\eta_3 = (k-1)^2 = \rho_3{}^2$$

Hence,

$$k = \pm\rho_3 + 1 \tag{4.68}$$

and

$$p = \mp\rho_3 - 1 \tag{4.69}$$

depending on $k > 1$ or $k < 1$. Although $\boldsymbol{n}$ has a unique solution:

$$\boldsymbol{n} = \pm \boldsymbol{u_3}$$

subject to the constraint $\boldsymbol{P}_i \cdot \boldsymbol{n} > 0$, $\boldsymbol{t_0}$ has two solutions because k has two solutions:

$$\boldsymbol{t_0} = -k\boldsymbol{n}$$

Therefore, $\boldsymbol{t}$ and $\boldsymbol{R}$ also have two solutions. Fig. 4.4 shows the two possible scenarios for $k < 1$ and $k > 1$ for the case $k = -p < 2$. Again, the camera first translates along $-\boldsymbol{n}$ direction a distance of kH, and then rotates $\boldsymbol{R}$. If $k > 1$, the camera is on the other side of the reference plane; if $k < 1$, the camera is on the same side of the reference plane.

4.4.6 Case $k = -p = 2$

This constraint implies $\boldsymbol{t_0} = -2\boldsymbol{n}$. Thus, Eq. 4.17 becomes singular:

$$\lambda^2 \boldsymbol{A_0}^T \boldsymbol{A_0} = \boldsymbol{I} \tag{4.70}$$

Therefore, any three arbitrary vectors satisfying:

$$\boldsymbol{v}_i \perp \boldsymbol{v}_j, \quad i \neq j$$

are eigenvectors, and the corresponding eigenvalues are:

$$\eta_2 = \eta_1 = \eta_3 = 1$$

Thus, $\boldsymbol{n}$ can be any of the three eigenvectors. Consequently, $\boldsymbol{t}$ and $\boldsymbol{R}$ have infinitely many solutions.

The geometry of this case is very similar to Fig. 4.4 (b), except here $k = 2$, which means that the camera center after motion is at the mirrored position of the camera center before the motion w.r.t. the reference plane, and rotation may be arbitrary.

4.4.7 Summary of all the cases in the three eigenvalues

There are four possible orderings of the three eigenvalues. The general case is that all three eigenvalues are distinct, i.e. $\eta_2 > \eta_1 > \eta_3$. The solution space in this case is a general ellipsoid which has three different axes, corresponding to the three different eigenvalues, as shown in Fig. 4.5 (a). The solution space for the case $\eta_2 > \eta_1 = \eta_3 = 1$ is a symmetric, cylindrical ellipsoid, as shown in Fig. 4.5 (b). In other words, the intersection of the ellipsoid at any point perpendicular to the longer axis is a circle; in this case, the two eigenvectors corresponding to η_1 and η_3 can be arbitrary. Similarly, Fig. 4.5 (c) shows the case of $\eta_2 = \eta_1 = 1 > \eta_3$, which also is a symmetric, cylindrical ellipsoid, in which the shorter axis corresponds to η_3. Finally, the solution space of case $\eta_2 = \eta_1 = \eta_3 = 1$ is a sphere, as shown in Fig. 4.5 (d). The three eigenvectors can be arbitrary, and the number of solutions is infinite.

4.4.8 Model Acquisition Algorithm

Based on the above analysis, given $n \geq 4$ model points (i.e. coplanar correspondences), the model acquisition algorithm is as follows:

- Form the normalized homography matrix $\boldsymbol{A_0}$;
- Compute the three singular values of $\boldsymbol{A_0}$: $\rho_2' \geq \rho_1' \geq \rho_3'$;
- Compute the normalized scale factor: $\lambda = \frac{1}{\rho_1'}$;
- Normalize the three singular values: $\rho_i = \lambda \rho_i' \quad i = 1, 2, 3$;
- If $\rho_2 > \rho_1 = 1 > \rho_3$:

  –

$$\eta_2 = 1 + p + \frac{2k(p+1)}{-k + \sqrt{k^2 + 4(p+1)}}$$

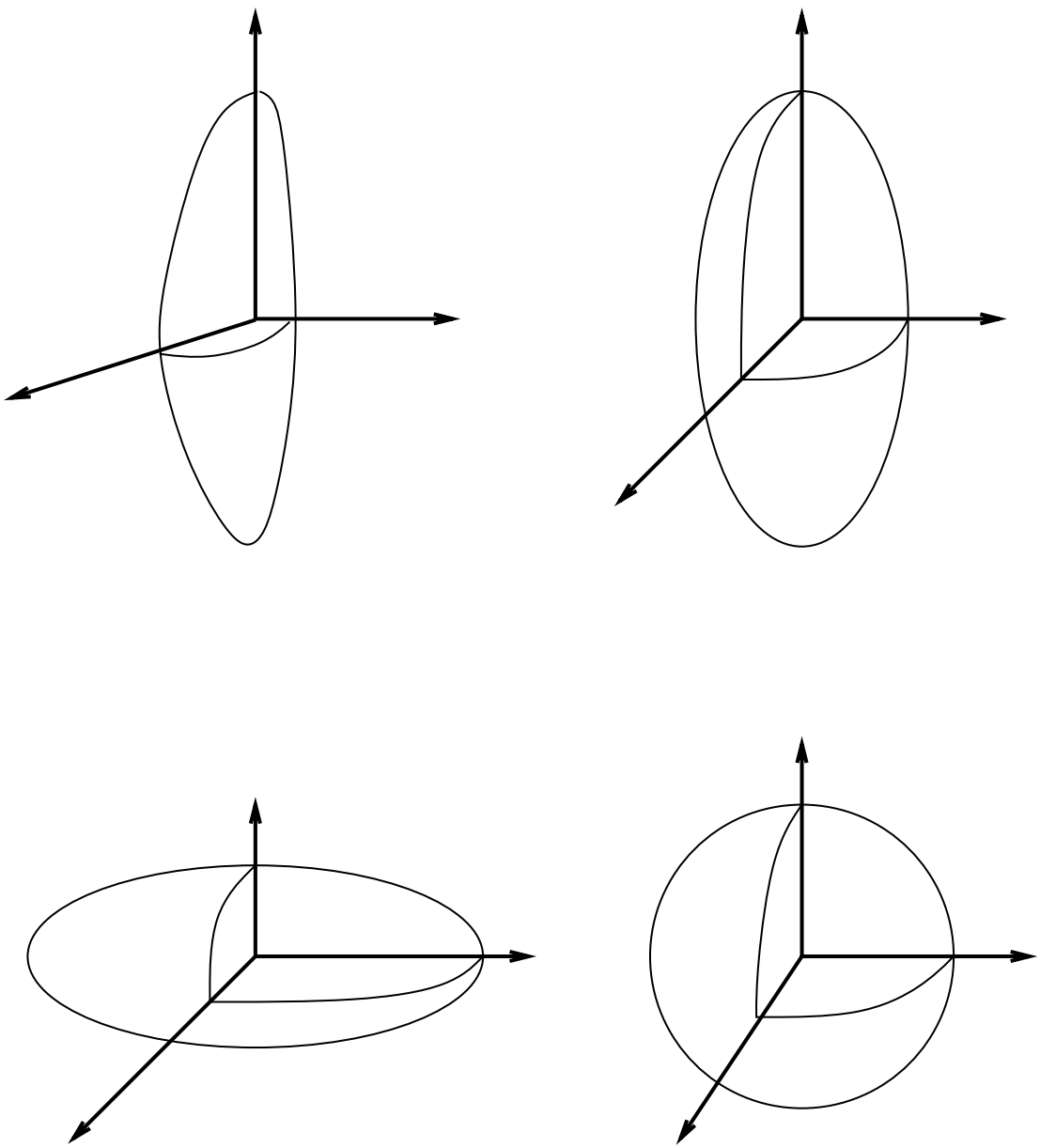

Figure 4.5. Geometric relationship among different eigenvalue cases.

–

$$\eta_3 = 1 + p + \frac{2k(p+1)}{-k - \sqrt{k^2 + 4(p+1)}}$$

–

$$k = \rho_2 - \rho_3$$

–

$$p = \rho_2 \rho_3 - 1$$

– Two solutions for $\boldsymbol{R}$, $\boldsymbol{t}$, and $\boldsymbol{n}$, based on constraint $\boldsymbol{P}_i \cdot \boldsymbol{n} > 0$

– The geometric illustration is shown in Fig. 4.1

- If $\rho_2 > \rho_1 = \rho_3 = 1$:

  – *1st Solution:*

    *

    $$\eta_2 = (k+1)^2 = (p+1)^2 = {\rho_2}^2$$

    *

    $$k = p = \rho_2 - 1$$

    *

    $$\boldsymbol{n} = \pm \boldsymbol{u_2}, \quad \boldsymbol{P}_i \cdot \boldsymbol{n} > 0$$

    *

    $$\boldsymbol{t_0} = k\boldsymbol{n}$$

  – *2nd Solution:*

    *

    $$\eta_2 = (k-1)^2 = (p+1)^2 = {\rho_2}^2, \quad k > 2$$

* $$k = \rho_2 + 1$$
* $$p = -\rho_2 - 1$$
* $$\boldsymbol{n} = \pm \boldsymbol{u_2}, \quad \boldsymbol{P_i} \cdot \boldsymbol{n} > 0$$
* $$\boldsymbol{t_0} = -k\boldsymbol{n}$$

  - The geometric interpretation of the two solutions is shown in Fig. 4.6

- If $\rho_2 = \rho_1 = 1 > \rho_3$:
  - $$\eta_3 = (k-1)^2 = {\rho_3}^2, \quad k < 2$$
  - $$k = \pm \rho_3 + 1$$
  - $$p = \mp \rho_3 - 1$$
  - $$\boldsymbol{n} = \pm \boldsymbol{u_3} \quad \boldsymbol{P_i} \cdot \boldsymbol{n} > 0$$
  - $$\boldsymbol{t_0} = -k\boldsymbol{n}$$

  - The geometric interpretation of the two solutions is shown in Fig. 4.7

- If $\rho_2 = \rho_1 = \rho_3$:

- The transformed camera center position is symmetric to the position before the transform (i.e. reflection w.r.t. the reference plane)
- Infinitely many solutions
- The geometric interpretation is shown in Fig. 4.8

4.5 Model Extension: 3D Reconstruction with Error Analysis

In the previous section, analytical closed-from solutions for recovering the relative geometry among the two cameras and the reference plane were obtained. Based on this relative geometry, the 3D coordinates of the point $\boldsymbol{P}$, expressed in the first camera's coordinate system, can be obtained up to a scale factor.

Assume that $\boldsymbol{P}$ is an arbitrary 3D point which may or may not be on the reference plane. Under homogeneous coordinates, we know:

$$\boldsymbol{P} \cong \boldsymbol{p} \tag{4.71}$$

and

$$\boldsymbol{P} - \boldsymbol{t} \cong \boldsymbol{R}^T \boldsymbol{p}' \tag{4.72}$$

Using our conventional notation, i.e. $\boldsymbol{P} = (X, Y, Z)^T, \boldsymbol{p} = (x, y, 1)^T, \boldsymbol{p}' = (x', y', 1)^T$, and letting $\boldsymbol{R} = (\boldsymbol{R_1}, \boldsymbol{R_2}, \boldsymbol{R_3})$, where $\boldsymbol{R_1}, \boldsymbol{R_2}, \boldsymbol{R_3}$ are the column vectors of the rotation matrix, we can solve for P from the above two equations by substituting $\boldsymbol{t}$ from Eq. 4.57:

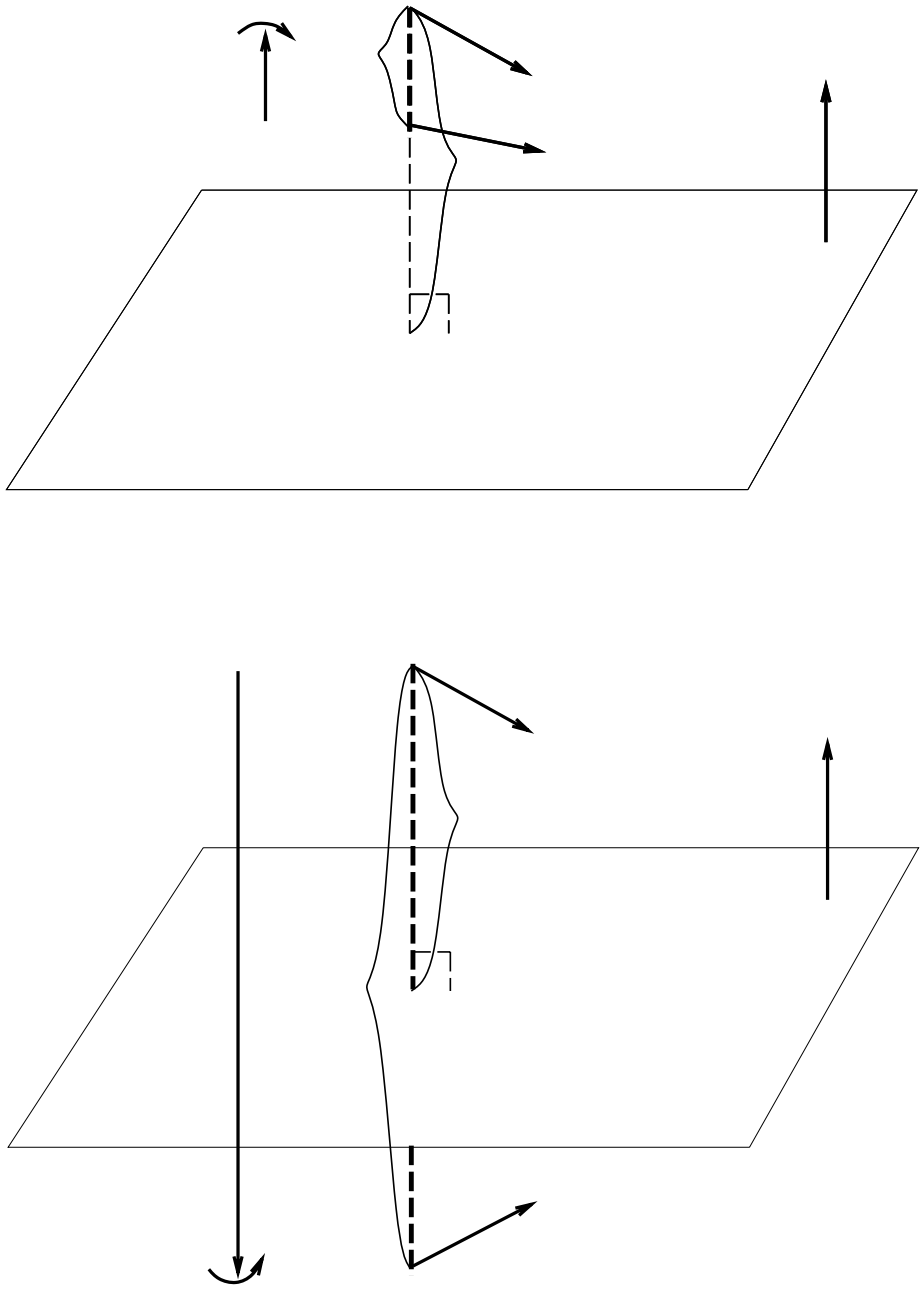

Figure 4.6. Geometric illustration for the two solutions for case $\eta_2 > \eta_1 = \eta_3 = 1$.

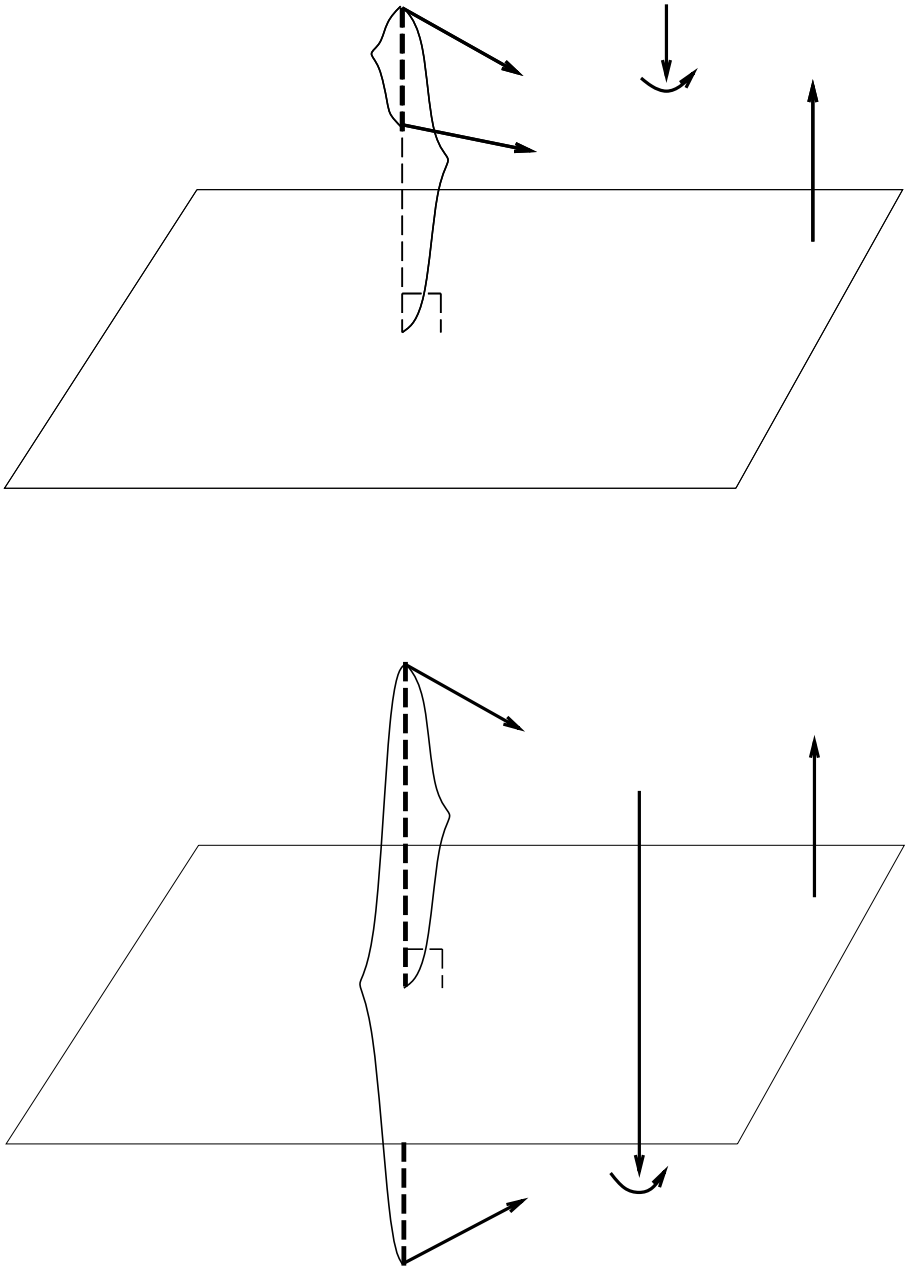

Figure 4.7. Geometric illustration for the two solutions for case $\rho_2 = \rho_1 = 1 > \rho_3$.

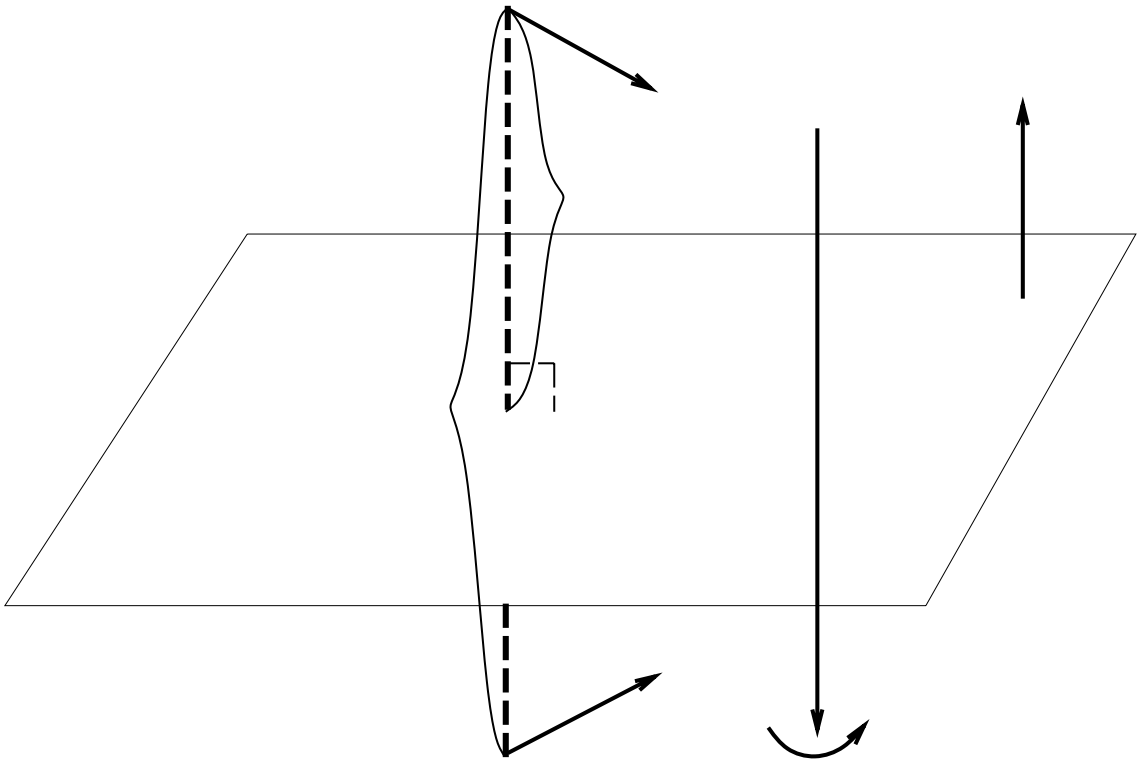

Figure 4.8. Geometric illustration for the solutions for case $\rho_2 = \rho_1 = \rho_3$.

$$X = \frac{t_{0Z}(\boldsymbol{R_1} \cdot \boldsymbol{p'}) - t_{0X}(\boldsymbol{R_3} \cdot \boldsymbol{p'})}{x(\boldsymbol{R_3} \cdot \boldsymbol{p'}) - (\boldsymbol{R_1} \cdot \boldsymbol{p'})} xH \tag{4.73}$$

$$Y = \frac{t_{0Z}(\boldsymbol{R_2} \cdot \boldsymbol{p'}) - t_{0Y}(\boldsymbol{R_3} \cdot \boldsymbol{p'})}{y(\boldsymbol{R_3} \cdot \boldsymbol{p'}) - (\boldsymbol{R_2} \cdot \boldsymbol{p'})} yH \tag{4.74}$$

$$Z = \frac{t_{0Z}(\boldsymbol{R_1} \cdot \boldsymbol{p'}) - t_{0X}(\boldsymbol{R_3} \cdot \boldsymbol{p'})}{x(\boldsymbol{R_3} \cdot \boldsymbol{p'}) - (\boldsymbol{R_1} \cdot \boldsymbol{p'})} H \tag{4.75}$$

Since $\boldsymbol{R}$ and $\boldsymbol{t_0}$ have two solutions in general, the final reconstruction also has two solutions. However, each reconstruction is subject to only one unknown scaling factor H, which is the distance of the first camera center from the reference plane. Therefore, we conclude that given at least four coplanar correspondences in two externally uncalibrated cameras, we can recover any 3D point in Euclidean space up to two solutions with only one uniform scale factor, which is the physical distance of the center of the first camera from the reference plane. If this distance is known, any 3D point can be completely recovered (up to two solutions). Two questions remain to be answered: (1) how to remove the ambiguity between the two solutions and (2) how to reconstruct the 3D scene in the presence of noise.

If there are only two views and there is no *a priori* information (such as known reference plane orientation, or known part of the relative geometry, e.g. known relative rotation or translation, etc.) available, it is impossible to disambiguate the two solutions. Perhaps the simplest way to resolve this ambiguity is to use a third view (see Section 4.7).

Now we investigate the problem of how to robustly implement this reconstruction technique in real world imaging situations, i.e. in the presence of noise and measurement error. There are several possible sources of error which can affect the reconstruction:

- the camera internal calibration may be subject to error;

- correspondences may be mismatched, i.e. the two corresponding points in the image domain may be the maps of two distinct 3D points; and
- image points may be subject to localization error.

Here we will only consider the third case (localization error), assuming that the calibration parameters and correspondence pairs are correct.

To overcome the effect of localization error, more than four coplanar correspondences are needed so that a least mean squares technique can be applied to obtain an optimal solution, assuming the localization error distribution is Gaussian or uniform. We adopt a fairly conventional technique for obtaining a robust solution in the presence of measurement error. A least median squares algorithm is employed in a global search for a subset of this n point set such that the least mean squares solution of this subset gives rise to the minimum value of the following error function:

$$\min_{subset} \sum_{i=1}^{n} [(x_i - x_i'')^2 + (y_i - y_i'')^2] \tag{4.76}$$

where (x_i'', y_i'') is the 2D point back-projected to the first image plane based on the reconstructed 3D point $\boldsymbol{P}_i$. Note that the error function is simply the square of the Euclidean distance of the 2D projection of a 3D point and the back projected point under the assumed solution.

In order to limit the combinatorial search when n is very large, we can use the following result from [29]:

Let q be the confidence probability that at least one of the trials produces the 'best' solution, and let r be the probability that a point is a "bad" point, i.e. when it is added to a subset, the best solution cannot be achieved. Then, the number of trials, m, that we have to make to achieve the confidence of q, is:

$$m = \frac{\log(1-q)}{\log(1-r^n)} \tag{4.77}$$

The investigation of the robustness of the performance of this algorithm is conducted using simulation. We randomly choose n points on a 3D plane, and another 5 points not on the plane. Then we map the $n+5$ points into our first camera retina to obtain their image plane coordinates. The camera plane is digitized to 500×500 resolution. The camera aspect ratio is set to be 1, and the focal length is set to be 1000 pixels. The camera is then translated $5, 6$, and -20 units along the X, Y, Z axes, respectively, and rotated by $10°, 5°, 15°$ about the X, Y, Z axes, respectively. Finally we map the $n+5$ 3D points to the second camera retina to obtain their corresponding coordinates.

Now we add different levels of Gaussian noise to the localizations of each of the $n+5$ points in the two image planes, and use those corrupted correspondences as input to the reconstruction algorithm to obtain the reconstructed coordinates. The reconstructed coordinates are then compared with the ground truth data (noise-free points), and the relative error in terms of percentage is calculated. The noise is assumed to be unbiased (with Gaussian mean as 0); the standard deviation of the noise is expressed in terms of pixels.

Fig. 4.9 shows the performance of this algorithm under different levels of noise w.r.t. the number of planar points, n, used to estimate the homography matrix with standard deviation of 1, 2, 3, and 5 pixels, respectively. Based on this simulation result, we conclude

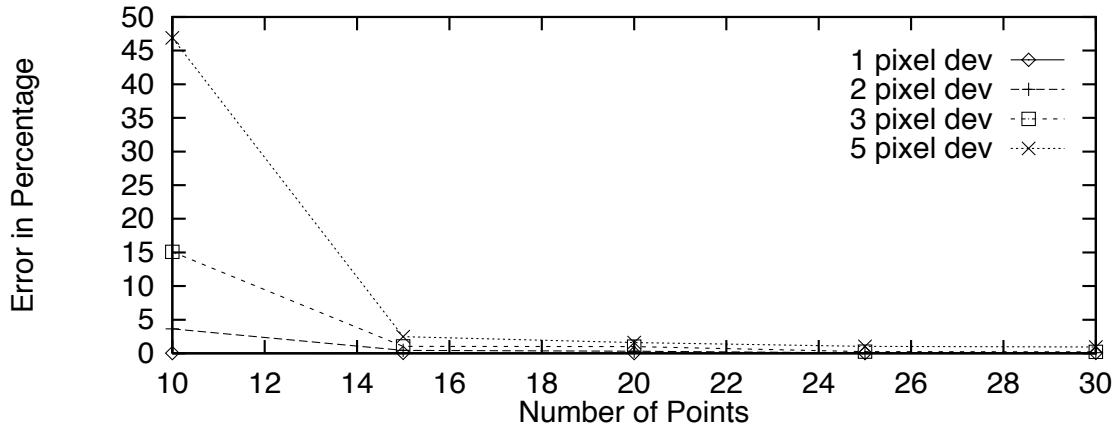

Figure 4.9. Error analysis results based on simulation.

that as long as the number of coplanar correspondences is greater than 15, the performance of the algorithm is quite good, with a reconstruction error of less than 5%. This provides an empirical threshold; in practice, we do not need more than 15 coplanar points to estimate the homography matrix in general, and the reconstruction is very stable even in the presence of Gaussian noise with up to 5 pixels standard deviation. Experimental results on real data agrees well with this conclusion (next Section).

4.6 Experimental Results with Real Images

The algorithm was tested using two real image sequences. Fig. 4.10 shows two images (frame 1 and frame 6) of a box sequence taken by a CCD camera. The camera is a SONY B/W AVC-D1, and has a FOV of 23.4° in the X direction and 22.4° in the Y direction. The image resolution is 256×242. In the first frame of this sequence, the camera was 650 mm distant from the top front corner of the box. The location of 30 points (marked $+$ and $\times$ in Fig. 4.10) in a world coordinate system was measured to an accuracy within 1 mm. The world coordinate system was chosen to be consistent with the natural axes of the

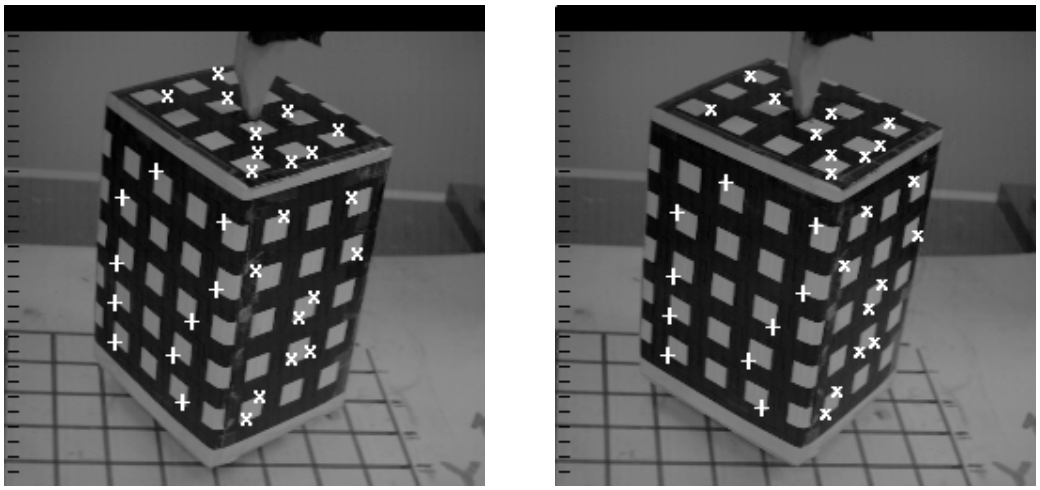

Figure 4.10. 1st frame (left) and the 6th frame (right) of the box sequence.

box. Table 4.1 shows the ground truth coordinates of the 30 points in the world coordinate system, as well as the ground truth coordinates converted to the camera 1 coordinate system. The box was rotated about its central vertical axis, with approximately 3.6° of rotation between consecutive frames, while the camera was kept stationary. The 30 points were tracked over 8 frames [68].

Table 4.2 shows the reconstruction result of the 30 points using frame 1 and 6 as input. In this experiment, we use the leftside face (with points marked as +) of the box (Fig. 4.10) as the reference plane. Fig. 4.11 shows the scenario of this experiment. The first ten points[4] in Table 4.2 on this leftside face were used to search for a best estimate of the normalized homography matrix $\boldsymbol{A_0}$, as specified by the minimization in equation 4.76. Based on this $\boldsymbol{A_0}$, the relative geometry between the two cameras was obtained. This was then used to reconstruct all of the 30 points in 3D Euclidean space in the first camera coordinate system; this data is shown in Table 4.2. On a DecStation 5000, the total time for a complete combinatorial search for the best solution was less than 1 second. With code optimization

[4]Only these ten points were available on the plane in this sequence.

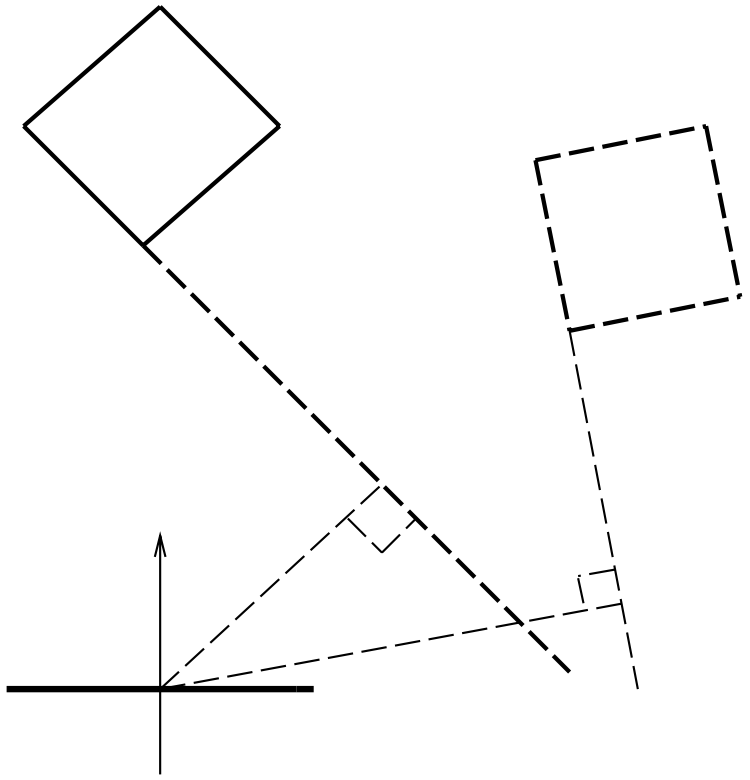

Figure 4.11. Top view of the Box Sequence experiment. The solid box shows the first (the true) solution while the dashed box shows the second solution. Note that this figure is illustrative only; it is not drawn in the actual scale of the experimental data.

and a more modern processor, we believe that the time to combinatorially search a modest set of points (say $n < 30$) could be made small enough that the algorithm could be useful for time critical applications such as navigation, without limiting the search by means of equation 4.77. As yet, however, 'real-time' applications have not been attempted.

The only ground truth available for this sequence was measured in terms of a world coordinate system (as listed in Table 4.1). In order to gauge the performance of the algorithm, the ground truth must be expressed in terms of the first camera's coordinate system. The conversion was accomplished using Kumar's pose algorithm [68] to compute the pose of the first camera in terms of the world coordinate system. The computed pose of the first frame is $\boldsymbol{t} = (-8.257, 74.647, 620.314)^T$, $\boldsymbol{R} = (0.400, -0.163, 0.872, -0.229)^T$. Also note here the rotation $\boldsymbol{R}$ is represented by a quaternion instead of a matrix. Note that this ground truth is actually a kind of "pseudo ground truth" because it is a combination of "pure" ground truth, which is based on actual measurements in a world coordinate system, and computed pose, which is not ground truth. Table 4.2 lists the reconstructed 3D coordinates for each point, the ratio values (the scaling factor) of each ground truth coordinate and its corresponding reconstructed coordinate for each point, and the relative deviations of these scaling factors as compared with the "pseudo ground truth". According to the theoretical results (see Eqs. 4.73 to 4.75), the ratio is the uniform scale factor of the reconstruction, and is the physical distance of the center of the first camera from the 3D reference plane. According to the "pseudo ground truth", this distance is 328.12 mm. As can be seen from Table 4.2, most of the values are very close to this "ground truth" value (within 5% error); the average of the absolute values of relative error is 3.79%, and the RMS average is 5.58%. It is worth noting that this algorithm gives rise to two solutions in general, and they cannot be disambiguated without any *a priori* knowledge. However, in this experiment, since we

Table 4.1. Ground truth coordinates in a world coordinate system and in the camera 1 coordinate system for the Box Sequence. Note that the ground truth coordinates in the camera 1 coordinate system were obtained by using Kumar's pose algorithm. The first ten points were used to estimate the homography matrix.

Point	World System			Camera 1 System		
Label	X	Y	Z	X	Y	Z
1	-16.25	-21.25	55.0	46.570	49.547	610.082
2	43.75	-21.25	55.0	8.964	21.494	572.682
3	-46.25	-51.25	55.0	68.403	38.315	644.682
4	43.75	-66.25	55.0	13.510	-16.395	596.531
5	-46.25	-96.25	55.0	72.949	-0.426	668.531
6	28.75	-96.25	55.0	25.942	-34.641	621.780
7	-46.25	-126.25	55.0	75.980	-24.833	684.431
8	13.75	-126.25	55.0	38.374	-52.887	647.030
9	-46.25	-156.25	55.0	79.010	-50.092	700.330
10	28.75	-156.25	55.0	32.003	-85.159	653.579
11	-60.0	0.0	-15.0	17.759	106.739	666.338
12	45.0	0.0	45.0	-1.692	41.493	566.390
13	-45.0	0.0	30.0	43.127	87.612	631.116
14	30.0	0.0	30.0	-3.880	52.545	584.365
15	30.0	0.0	-30.0	-50.238	68.696	618.862
16	45.0	0.0	15.0	-24.871	49.569	583.638
17	45.0	0.0	0.0	-36.461	53.607	592.263
18	-30.0	0.0	0.0	10.546	88.674	639.014
19	15.0	0.0	15.0	-6.068	63.596	602.339
20	0.0	0.0	-15.0	-19.846	78.685	628.938
21	55.25	-22.2	24.3	-21.867	23.581	583.668
22	55.25	-22.2	-20.7	-56.636	35.695	609.541
23	55.25	-52.2	39.3	-7.247	-5.716	590.943
24	55.25	-67.2	-35.7	-63.680	1.844	642.014
25	55.25	-97.2	9.3	-25.880	-35.529	632.040
26	55.25	-82.2	-5.7	-38.985	-18.861	632.715
27	55.25	-127.2	9.3	-22.850	-60.788	647.940
28	55.25	-127.2	-5.7	-34.439	-56.750	656.564
29	55.25	-157.2	39.3	3.360	-94.123	646.591
30	55.25	-142.2	24.3	-9.745	-77.455	647.265

Table 4.2. Reconstruction results. Note that the first ten points are the reference points.

Reconstructed 3D Coordinates in Camera 1 System			**Recovered Scale Factors for Each Coordinate**			**Relative Deviations of the Recovered Scale Factors**		
X	Y	Z	X	Y	Z	X	Y	Z
0.140	0.144	1.816	331.96	344.64	335.91	1.17%	5.03%	2.37%
0.028	0.063	1.707	318.90	339.04	335.50	-2.81%	3.33%	2.25%
0.202	0.112	1.912	338.62	341.34	337.23	3.20%	4.03%	2.77%
0.045	-0.050	1.771	302.33	326.25	336.87	-7.86%	-0.57%	2.67%
0.216	-0.001	1.979	337.88	363.73	337.85	2.97%	10.85%	2.96%
0.079	-0.105	1.844	328.77	330.08	337.18	0.20%	0.60%	2.76%
0.225	-0.073	2.019	338.11	339.01	339.03	3.04%	3.32%	3.32%
0.118	-0.155	1.910	324.76	342.02	338.83	-1.02%	4.24%	3.26%
0.233	-0.147	2.059	339.24	340.52	340.13	3.39%	3.78%	3.66%
0.097	-0.252	1.919	331.35	337.51	340.67	0.98%	2.86%	3.82%
0.055	0.334	1.981	324.06	319.90	336.31	-1.24%	-2.51%	2.50%
-0.006	0.125	1.688	283.51	331.21	335.57	-13.59%	0.94%	2.27%
0.124	0.267	1.886	349.14	327.92	334.63	6.41%	-0.06%	1.98%
-0.014	0.162	1.745	284.64	323.65	334.88	-13.25%	-1.36%	2.06%
-0.148	0.217	1.842	339.18	317.04	336.05	3.37%	-3.38%	2.42%
-0.073	0.153	1.733	339.42	323.40	336.84	3.44%	-1.44%	2.66%
-0.109	0.168	1.757	335.91	318.31	337.09	2.37%	-2.99%	2.73%
0.029	0.276	1.902	367.36	321.74	335.88	11.96%	-1.95%	2.37%
-0.019	0.197	1.795	312.68	323.43	335.59	-4.71%	-1.43%	2.27%
-0.056	0.244	1.865	354.51	322.28	337.28	8.04%	-1.78%	2.79%
-0.066	0.071	1.729	328.90	332.63	337.64	0.24%	1.37%	2.90%
-0.166	0.107	1.799	341.04	333.38	338.73	3.94%	1.60%	3.23%
-0.021	-0.016	1.757	341.67	365.64	336.36	4.13%	11.43%	2.51%
-0.187	0.004	1.884	339.91	435.63	340.72	3.59%	32.76%	3.84%
-0.077	-0.105	1.865	337.53	338.19	338.83	2.87%	3.07%	3.26%
-0.117	-0.057	1.859	334.17	328.29	340.44	1.84%	0.05%	3.75%
-0.069	-0.177	1.908	333.42	343.96	339.64	1.61%	4.83%	3.51%
-0.104	-0.172	1.923	332.23	329.54	341.50	1.25%	0.43%	4.08%
0.009	-0.273	1.897	356.62	344.91	340.81	8.69%	5.12%	3.87%
-0.032	-0.228	1.896	300.38	340.19	341.45	-8.46%	3.68%	4.06%

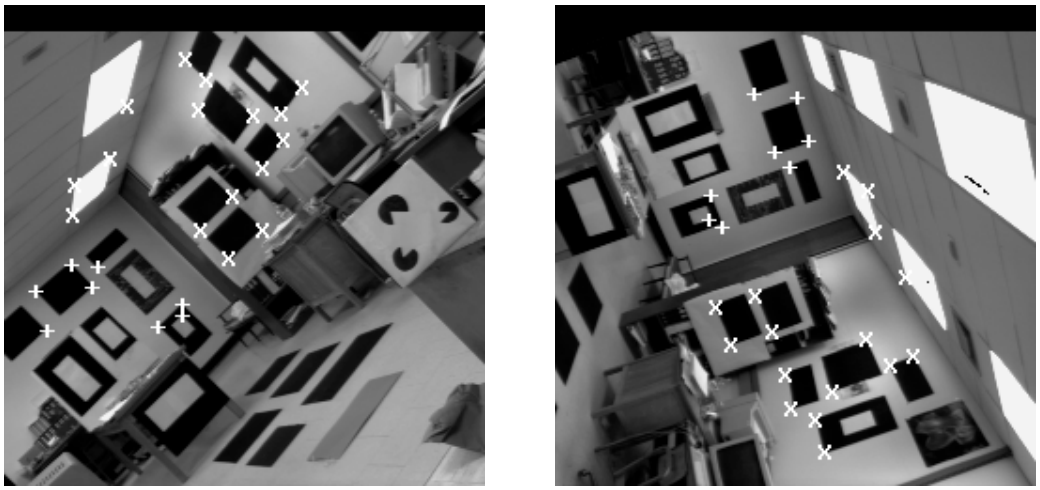

Figure 4.12. Frame 1 (left) and frame 30 (right) of the room sequence.

have the ground truth, it becomes straightforward to select the correct solution because only one of the two solutions can "confirm" to the ground truth.

Fig. 4.12 shows two frames of another sequence, taken by another SONY B/W AVC D-1 camera mounted on a PUMA2 robot arm in the Laboratory for Perceptual Robotics at UMass. The camera FOV is 41.675° by 39.529° and the image resolution is 256×242. The sequence was taken while the robot arm was rotating with a radius of approximately 2 ft. and a rotation of approximately 4° between frames; 30 frames were taken over a total angular displacement of 116° of the robot arm. Fig. 4.12 shows frame 1 and frame 30. 24 points marked in the images were tracked between the two frames. The eight points on the facing wall (marked with +) were used to find the $\boldsymbol{A_0}$ matrix. Then based on the decomposed relative pose between the two frames, 3D coordinates of all the 24 points were recovered. Again only the ground truth in terms of a world coordinate system is available, which is shown in Table 4.3. The world coordinate system was chosen to be consistent with the natural room axes, with the origin in the corner of the room facing the camera. We again used Kumar's pose algorithm [68] to transform the ground truth of the 24 points in

the world coordinate system into the first frame's camera coordinate system. The converted coordinates are also shown in Table 4.3. The computed pose of the first frame w.r.t. the world coordinate system is $\boldsymbol{t} = (0.048, -3.113, 32.572)^T$, $\boldsymbol{R} = (-0.195, 0.379, 0.902, 0.069)^T$, where the rotation $\boldsymbol{R}$ is represented as a quaternion. Thus, the "ground truth" of the 3D coordinates in terms of the first frame's camera system can be obtained. Note that again this is "pseudo ground truth" in the sense that it is the combination of real ground truth and computed information. Table 4.4 lists the reconstructed coordinates for each point by our algorithm, the recovered scaling factors for each coordinate and point, and the relative error for the final recovered scale factors as compared with the "pseudo ground truth" distance, which is 30.645 ft.. From Table 4.4, we can see that the performance of the algorithm for this sequence is worse than the box sequence with an average of the absolute values of the relative errors in the scale factor of 9.52% (the RMS error is 12.07%).

The performance of both experiments can be compared to the simulation results shown in Fig. 4.9 and discussed in Section 4.5. Fig. 4.9 implies that the number of coplanar points used to reconstruct the $\boldsymbol{A}_O$ matrix should be 15 or greater in order to achieve optimal performance from the algorithm, under the assumption that the geometry and statistical characterization of the localization error reflect the experimental conditions. The number of coplanar points used to reconstruct $\boldsymbol{A}_O$ in the two experiments on real data were 10 and 8, respectively, which implies that the results for the first experiment on the box sequence should be better than the results of the second experiment on the room sequence. This is supported by the experimental results, where the average absolute errors obtained were within 5% and 10% respectively. A second complicating factor in the experiments was the use of computed pose to establish the ground truth of the coplanar point coordinates in the first camera's frame of reference. It is unclear at this point how this affected the experimental results and the ground truth comparisons. Work is underway to further validate the

Table 4.3. Ground truth coordinates in a world coordinate system and in the camera 1 coordinate system for the Room sequence. Note that the first eight points were used to estimate the homography matrix.

Point	World System			Camera 1 System		
Label	X	Y	Z	X	Y	Z
1	-7.245	6.455	0.0	9.247	-3.341	29.494
2	-4.22	8.09	0.0	8.487	-0.204	30.680
3	-3.44	7.0	0.0	7.215	-0.457	31.020
4	-1.75	2.855	0.0	3.192	-2.258	31.020
5	-4.31	6.45	0.0	-0.264	-3.995	16.354
6	-6.335	8.125	0.0	-0.744	-3.601	13.501
7	-3.39	2.875	0.0	-1.660	-2.557	17.087
8	-2.22	2.45	0.0	-2.551	-1.744	14.341
9	0.0	5.37	13.89	-0.287	4.453	19.749
10	0.0	4.68	16.53	-1.567	4.689	17.351
11	0.0	4.12	14.07	-1.230	3.624	19.613
12	0.0	8.14	13.52	1.794	6.298	20.024
13	-3.35	9.01	7.025	6.487	2.933	24.588
14	-1.495	9.01	7.035	5.304	4.155	25.329
15	-3.34	9.01	3.13	7.645	1.876	28.153
16	-1.58	9.01	11.12	4.136	5.215	21.560
17	0.0	7.0	11.76	1.510	5.016	21.659
18	0.0	4.69	14.91	-1.076	4.254	18.832
19	0.0	4.17	11.95	-0.560	3.080	21.550
20	0.0	7.32	13.51	1.214	5.719	20.052
21	-1.30	4.15	11.28	0.453	2.029	21.638
22	-2.94	4.13	11.28	1.482	0.937	20.975
23	-1.38	2.77	11.28	-0.478	1.007	21.637
24	-3.02	2.75	11.28	0.551	-0.085	20.974

Table 4.4. Reconstruction results. Note that the first eight points were used as the reference plane points.

Reconstructed 3D Coordinates in Camera 1 System			**Recovered Scale Factors for Each Coordinate**			**Relative Deviations of the Recovered Scale Factors**		
X	Y	Z	X	Y	Z	X	Y	Z
-0.313	0.112	-0.997	-29.51	-29.73	-29.57	-3.67%	-2.99%	-3.51%
-0.292	0.006	-1.055	-29.07	-32.71	-29.07	-5.14%	6.74%	-5.13%
-0.250	0.016	-1.055	-28.84	-29.28	-28.94	-5.89%	-4.47%	-5.55%
-0.110	0.080	-1.106	-28.95	-28.33	-28.75	-5.52%	-7.55%	-6.19%
-0.252	0.046	-1.054	-29.24	-30.90	-29.11	-4.58%	0.83%	-5.00%
-0.336	0.053	-1.015	-29.37	-29.87	-29.38	-4.15%	-2.53%	-4.13%
-0.146	0.115	-1.072	-29.11	-29.01	-29.03	-5.02%	-5.34%	-5.25%
-0.109	0.098	-1.097	-29.32	-29.17	-28.82	-4.31%	-4.80%	-5.96%
0.012	-0.140	-0.782	-24.79	-31.91	-25.26	-19.11%	3.79%	-17.56%
0.064	-0.143	-0.708	-24.68	-32.76	-24.50	-19.48%	6.90%	-20.04%
0.050	-0.114	-0.781	-24.61	-31.77	-25.12	-19.70%	3.66%	-18.02%
-0.072	-0.207	-0.789	-24.80	-30.41	-25.38	-19.08%	-0.77%	-17.19%
-0.245	-0.102	-0.913	-26.47	-28.63	-26.95	-13.62%	-6.56%	-12.07%
-0.193	-0.159	-0.924	-27.55	-26.11	-27.42	-10.10%	-14.80%	-10.52%
-0.281	-0.058	-1.023	-27.21	-32.12	-27.52	-11.19%	4.80%	-10.18%
-0.159	-0.171	-0.851	-25.97	-30.52	-25.34	-15.25%	-0.40%	-17.31%
-0.060	-0.168	-0.832	-25.27	-29.84	-26.03	-17.55%	-2.62%	-15.05%
0.043	-0.133	-0.756	-25.02	-32.02	-24.91	-18.36%	4.50%	-18.72%
0.023	-0.100	-0.839	-24.71	-30.89	-25.69	-19.36%	0.80%	-16.18%
-0.049	-0.188	-0.788	-24.68	-30.49	-25.45	-19.47%	-0.50%	-16.93%
-0.014	-0.066	-0.831	-31.55	-30.68	-26.02	2.96%	0.11%	-15.08%
-0.055	-0.031	-0.799	-26.84	-30.55	-26.26	-12.43%	-0.31%	-14.30%
0.024	-0.033	-0.840	-20.21	-30.13	-25.76	-34.05%	-1.67%	-15.95%
-0.024	0.002	-0.799	-29.98	-35.45	-26.25	-2.16%	15.69%	-14.35%

robustness of the algorithm on real data sequences for which the correct "style" of ground truth is available.

4.7 Optimality and Extension to Completely Uncalibrated Views

4.7.1 Optimality

In this chapter, we have proposed an algorithm based on at least four coplanar correspondences to reconstruct the 3D scene in an Euclidean space. This algorithm assumes that the internal camera calibration is known.

As mentioned earlier, the current status of research shows that based on two *completely* uncalibrated views, it is impossible to recover 3D scene in a Euclidean space, no matter how many correspondences are given. Hence, the current algorithm cannot be extended to the completely uncalibrated two view case. However, it may be possible to extend this algorithm to the case where *some* of the internal camera parameters may be assumed unknown. The following property shows that it cannot.

Property *The algorithm for 3D reconstruction based on the homography matrix proposed in this chapter is optimal, both in terms of the number of required correspondences (i.e. 4), and in terms of the assumption that the camera internal calibration is known, i.e. no internal camera parameters can be assumed to be unknown.*

From Eq. 4.11, it is obvious that at least four correspondences are required to solve for the homography matrix. Hence, optimality in the required number of correspondences is straightforward. In order to see the optimality of the assumption of the internal calibration, Consider Eqs. 4.5 and 4.7. Clearly, from Eq. 4.7, the homography matrix $\boldsymbol{A}$ has eight degrees of freedom. On the other hand, from Eq. 4.5, $\boldsymbol{A}$ contains information about the camera

rotation $\boldsymbol{R}$, which has three degrees of freedom, the translation $\boldsymbol{t}$, which has three degrees of freedom, the unknown reference plane normal $\boldsymbol{n}$, which has two degrees of freedom, and a distance scale H, which counts one degree of freedom. Therefore, the homography matrix $\boldsymbol{A}$ contains information with nine degrees of freedom in total. However, this matrix can only be determined up to eight degrees of freedom no matter how many correspondences are given. This implies that the best any 3D reconstruction algorithm based on the homography matrix can do is to recover the 3D scene up to one unknown parameter (in this case a scale factor), which is what our algorithm does. The addition of unknown internal camera parameters would increase the total degrees of freedom beyond nine, and would lead to a solution with more than one unknown parameter. Hence, we complete the proof that the algorithm proposed in this chapter is optimal and that the assumption of internal calibration is necessary.

4.7.2 Unambiguous Reconstruction: More views

Faugeras and Maybank [82, 25] showed that based on three transformations with at least seven correspondences, and using the same camera, it is possible to recover the 3D scene in a scaled Euclidean space. Hartley [45, 47] went further to show that given at least three views with the same camera, it is possible to recover the 3D scene in an scaled Euclidean space, although in practical cases more views will be needed to achieve better performance. Those research efforts led us to ask if it is possible to use more than two views with the same but *completely* uncalibrated camera to reconstruct the 3D scene in a scaled Euclidean space. Based on counting the degrees of freedom, it appears that this may be possible.

Let us assume that we have n completely uncalibrated views with the same camera, and we have at least four coplanar correspondences in each view w.r.t. the same unknown reference plane. Based on the (at least) four correspondences, we can solve for the $n -$

1 independent uncalibrated homography matrices, $\boldsymbol{B}_{i,i+1}, i = 1, ..., n-1$, where $\boldsymbol{B}_{i,i+1}$ denotes the uncalibrated homography mapping between view i and view $i+1$. Here we use $\boldsymbol{B}$ to represent an uncalibrated homography matrix to differentiate it from a calibrated homography matrix $\boldsymbol{A}$. The relationship between the uncalibrated homography matrix and the corresponding calibrated homography matrix is:

$$\boldsymbol{B} = \boldsymbol{C}^{-1}\boldsymbol{A}\boldsymbol{C} \tag{4.78}$$

where $\boldsymbol{C}$ is the camera internal calibration matrix defined as:

$$\boldsymbol{C} = \begin{pmatrix} -k_x & k_x \cot(\theta) & x_0 \\ 0 & -k_y \csc(\theta) & y_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{4.79}$$

where k_x, k_y are the scale factors along X and Y coordinate axes in the camera image plane, respectively, x_0 and y_0 are the coordinates of the camera principal point, and θ is the angle between the two coordinate axes in the camera image plane.

Clearly, each $\boldsymbol{B}_{i,i+1}$ can be determined in eight independent degrees of freedom by solving the linear equations obtained from the coplanar correspondences. Thus, there are $8(n-1)$ degrees of freedom for the n views. On the other hand, each $\boldsymbol{B}_{i,i+1}$ contains information about the rotation $\boldsymbol{R}_{i,i+1}$, which has three degrees of freedom, the translation $\boldsymbol{t}_{i,i+1}$, which has another three degrees of freedom. That makes the total degrees of freedom equal to $6(n-1)$. In addition, the reference plane normal has two degrees of freedom, the internal calibration matrix $\boldsymbol{C}$ has five degrees of freedom, and the distance from the first camera center to the reference plane (H_1) counts as another degree of freedom, for a total of $6(n-1)+8$ degrees of freedom encoded in the $n-1$ $\boldsymbol{B}_{i,i+1}$s. In order to have a *complete* 3D reconstruction, the following constraint has to be satisfied:

$$6(n-1)+8 \leq 8(n-1)$$

which leads to $n \geq 5$. That means if we have at least five completely uncalibrated views with the same camera, it is possible to completely recover the 3D scene in Euclidean space. Moreover, as is shown in the previous sections, since with two views the proposed reconstruction technique developed here results in two solutions in general, reconstruction with more than two views can disambiguate the solutions and results in a unique solution.

A future research direction is how to realize this goal, i.e. to completely reconstruct the 3D scene in Euclidean space uniquely with at least five uncalibrated views of at least four correspondences using the same camera.

4.8 Summary of Model Acquisition and Extension

This chapter addresses the problem of 3D reconstruction in Euclidean space based on homography mapping between two views. Compared with previous work, this new method has the following distinct features:

- The 3D structure recovered by this method has two solutions, as opposed to a family of solutions (e.g. a projectivity), and up to one uniform scaling parameter, as opposed to several unknown scale factors. Moreover, this scale factor has a precise physical interpretation, which is the perpendicular distance of the first camera center from a fixed 3D plane (the reference plane). Thus, if this distance is known *a priori*, the 3D Euclidean structure can be completely recovered up to two solutions.

- This method requires the same minimum number of correspondences, i.e. four correspondences, along with the assumption of known calibration (previous work assumed

weak calibration, that is, known epipoles). The only difference is that previous methods based on the four point assumption require that the four points should be non-coplanar. This method assumes four coplanar correspondences that are not collinear. On the other hand, the assumption of known calibration is stronger than those made by the previous methods (weak calibration or completely unknown calibration). However, we have shown in this chapter that the known calibration assumption is necessary for homography mapping based 3D reconstruction, if only two views are given.

- Since this method is based on an analytical closed-form solution, it is very easy to implement, and very inexpensive to compute.

Another interesting feature of this method is that direct 3D reconstruction in Euclidean space is perhaps more intuitively appealing than that based on a projective or affine transform. While the method proposed here recovers more information than previous methods, it also requires more information as input. The trade-off is that it assumes that the internal calibration of the camera(s) is known. Also if only two views are available, there are two final solutions in general. If we have more than two views, the solution is unique. Both simulation and real data experiments show that with enough points (≥ 15), the reconstruction results are reasonably stable. It is also shown in this dissertation that this algorithm is optimal in terms of the minimum required number of correspondences and in terms of the assumption of the required internal calibration. An extension of this algorithm to at least five completely uncalibrated views using the same camera is under way.

CHAPTER 5

CONCLUSION

3D reconstruction in general is a difficult research area in computer vision. While a general solution to 3D reconstruction is still an open research problem, many approaches have been proposed under a wide spectrum of constraints on the camera geometry. In this dissertation, several new constraints on the camera geometry have been proposed and the resulting algorithms have been applied in a robotic navigation scenario with good results. The robotic navigation problems addressed in this dissertation includes automatic external camera calibration, visual servoing during navigation, obstacle detection, and 3D model acquisition and extension. In the following two sections, we will summarize the main contributions of this dissertation, and outline future research directions.

5.1 Main Contributions

The main contributions of this dissertation are summarized below:

- Useful geometric variables were explored in a special structured environment (a hallway). They were then used to visually servo a robot during navigation in the hallway. In particular, the problem of automatically calibrating the robot's pose in this structured environment was addressed. Although navigation features such as vanishing points and looming distances are not new, and their application to robotic navigation

and automatic calibration are not new, the unique aspect of the work presented here is the mechanism by which they are combined with position information (i.e. the directional axis and its lateral distance). This combined information was then used to solve the visual servoing problem in the structured environment and a closed-form solution was developed for dynamic visual servoing control. Close to real time (about 0.2 ft. per second) algorithms were developed and used to control a Denning mobile robot during navigation in a hallway environment.

- Conventional obstacle detection is usually based on complete 3D reconstruction. Based on the assumption that obstacles constitute a "rare event", we believe that complete 3D reconstruction at each processing cycle is not necessary. Instead, what is required is a qualitative answer (yes or no) w.r.t. the question of whether or not there are obstacles in the scene. If there are no obstacles, the navigation system does not have to incur the cost of complete 3D reconstruction. In this way, much computation can be saved, and a high speed can be achieved for the navigation system. In case there *are* obstacles, then their location and height must be determined (that is, some form of 3D reconstruction must be performed). Depending on the execution speeds of the 3D reconstruction algorithms, the vehicle/robot can be stopped or slowed in order to determine the necessary geometric information, perform path planning for avoidance, etc. Based on this motivation, three obstacle detection algorithms were developed in this dissertation. The first two are purely qualitative in the sense that they only return yes/no answers. The third one is more quantitative than the other two because it recovers height information for all the points in the scene. Three different constraints on camera geometry are employed. The first algorithm, **KGP**, assumes that the camera relative pose is known; the second one, **UGP**, is based on completely unknown camera relative pose; the third one, **EGP**, is based on

what we call *partial calibration*. The first two algorithms are based on linear system theory and provide qualitative results (yes/no). The third algorithm extends previous work in the literature on 3D reconstruction with uncalibrated cameras to partial 3D reconstruction with partial calibration. We have also used simulation to investigate the obstacle detectability problem for each of the three algorithms.

- Model acquisition and extension is another important research area in robotic navigation. Previous work shows that based on the fundamental matrix from two uncalibrated cameras, 3D reconstruction can be achieved under an unknown projectivity. In this dissertation, we show that based on four *coplanar* correspondences of two externally uncalibrated cameras, 3D reconstruction can be achieved in Euclidean space with only one unknown scale factor and up to two real solutions. We start by decomposing a homography matrix between two externally uncalibrated cameras into the relative pose on the basis of four coplanar points (not collinear) in analytically closed-form solutions, and then develop a direct Euclidean 3D reconstruction algorithm based on this decomposition technique that reconstructs 3D structures up to a uniform scale factor and two real solutions. This scale factor has an explicit physical meaning, which is the distance from the camera center to a 3D plane. Thus, if this distance is known *a priori*, then a 3D reconstruction can be completely recovered up to two solutions. This 3D reconstruction technique has been applied to model acquisition and extension (e.g. obstacle reconstruction) in robotic navigation scenario. We have also used simulation to determine the lower bound of the number of points needed for a stable reconstruction. At the practical level, since the solution is analytically closed-form, it is very easy to implement, and the computation is inexpensive. It is also shown that this algorithm is optimal in terms of the number of required correspondences and in terms of the assumption made on known internal calibration.

5.2 Future Research Direction

First, the visual servoing control algorithm developed in Chapter 2 can be improved and extended. The algorithm assumes that the environment is structured, i.e. the navigation path has locally parallel boundaries. Although the algorithm has only been tested in a university hallway environment, it can also be applied in many other environments such as structured roads. In the future, this algorithm will be applied to outdoor navigation scenarios. Currently this algorithm assumes that the ground plane is flat, and that the camera is parallel to the ground plane. This assumption may be relaxed to accommodate variations in the ground plane as well as tilts in the camera focal axis. In this case, it can be shown that the independence assumption between the orientation and the directional axis in the image domain may be violated. Hence, it may be very difficult to obtain closed-form solutions for the orientation and lateral distance. However, even without closed-form solutions, qualitative control of a mobile robot based on visual servoing may still be possible. An important and interesting issue is how to engineer the mechanical control of the robot to take advantage of the qualitative visual information under real-time constraints.

In this dissertation, we have addressed the problem of static obstacle detection. In many cases, obstacles may have independent motion, in which case the problem extends to one of independent motion detection and determining whether or not a potential collision situation exists.

There has been much work on independent motion detection in the literature [119, 61, 110, 109, 85], but comparatively little work on the problem of qualitative independent motion detection. Thompson *et al* [109] are among the few to address the problem; they used the rigidity constraint to qualitatively detect independent moving objects.

One of our proposals for solving this qualitative independent motion detection problem is to assume that the projection from 3D to 2D is affine, and to represent all the features

as oriented points. Under this assumption, Jacobs [60, 59] has shown that each 3D model corresponds to a pair of lines in two affine parameter spaces, respectively, and each 2D image corresponds to a pair of points in the two spaces, respectively. Moreover, if we have oriented points in 3D, they can be represented as affine slopes in the two parameter spaces. Based on this, we can show that given n oriented points, we will have an n dimensional affine-slope space that can be decomposed into $n-2$ subspaces, such that each subspace is a hyperboloid. Then, using linear system theory techniques similar to those developed in Chapter 3, each oriented point can be tested to determine if it has the same motion w.r.t. the robot by checking if it is on the surface of its hyperboloid in the affine-slope space. The question is how to carry out this test in the presence of noise. This is one of our future research directions.

As for homography based 3D reconstruction, in Chapter 4 we showed that if the internal camera calibration is known, then given four coplanar correspondences, 3D reconstruction can be achieved directly under Euclidean space up to only one parameter. However, in general, the reconstruction is subject to two solutions, and it has been shown that these two solutions are intrinsically indistinguishable. This algorithm is optimal in terms of the number of minimum required correspondences and in terms of the assumption made on the known internal calibration. The optimality of this algorithm tells us that in order to achieve 3D reconstruction in a scaled Euclidean space with two views, we have to assume that the internal camera calibration is known. A natural question is if it is possible to extend this result to more views but with unknown camera internal parameters. We have shown that in theory this is possible with at least five completely uncalibrated views using the same camera. As mentioned in Chapter 4, research in this direction is underway .

In Chapter 4, we also show by simulation that in order to have stable reconstruction results with noise corrupted data, at least 15 corresponding points are required. We would

also like to know if there is any theoretical relationship between this empirical threshold and the performance of the 3D reconstruction under this algorithm. As another future direction, we intend to more completely investigate the robustness of this method based on more extensive simulations and tests on real data for which ground truth information is more readily available. We also hope to develop a theoretical understanding of error propagation through to the reconstruction results and to determine the sensitivity of the algorithm to additional sources of error, such as imprecise calibration parameters. Moreover, we would like to extend our noise model to include the case where the 3D distance from one of the stereo cameras to the reference plane is also noise corrupted. This would allow the sensitivity of the algorithm w.r.t. the plane parameters to be established.

Finally, both in Chapter 3 and in Chapter 4, the performance of all the relevant algorithms are completely dependent on the solutions of linear systems. As can be seen in those chapters, all the linear systems are ill-conditioned, i.e. the element values in each linear system vary tremendously within the system. All the experimental results reported in those chapters are based on "direct" solutions of those linear systems, i.e. there is no preprocessing or "polishing" of the linear systems before solving the systems. Recently, Hartley [50] pointed out that ill-conditioned systems may be turned into better-conditioned systems by preprocessing them prior to their solution. This preprocessing involves a local transform in the image domain which translates and scales all the coordinates so that the element values in a linear system look much more uniform than they do before this transform. The solutions of this preprocessed linear system are then transformed back to the original scale. By applying this transform to ill-conditioned systems, Hartley reported that the solutions are much more stable and robust to the presence of noise than without this preprocessing. As another future research direction, we intend to incorporate this scheme into the algorithms developed here, and test their robustness.

These are just a few future directions based on the research conducted in this dissertation. As we said in the beginning of this dissertation, 3D reconstruction under varying constraints on camera geometry in general is still an open problem, and there is much more that needs to be pursued as future research work.

APPENDIX A

HOMOGRAPHY MATRIX

The review of the derivation of the homography matrix in this appendix is based on [27]. Refer to Fig. 4.1, where there are two arbitrarily posed cameras $\boldsymbol{O_1}$ and $\boldsymbol{O_2}$. The two camera poses are related by a translation $\boldsymbol{t}$ and a rotation $\boldsymbol{R}$. π is a 3D plane with normal vector $\boldsymbol{n}$. The distance from the focal center of camera $\boldsymbol{O_1}$ to the plane π is H. Let $\boldsymbol{P}$ be an arbitrary 3D point on the plane π, represented in the coordinate system of camera $\boldsymbol{O_1}$, and $\boldsymbol{P'}$ be the same 3D point represented in the coordinate system of camera $\boldsymbol{O_2}$. Let $\boldsymbol{p}$ and $\boldsymbol{p'}$ be the two corresponding vectors in the image planes of the two cameras, respectively. $\boldsymbol{P}$ and $\boldsymbol{P'}$ are related by a transform:

$$\begin{aligned} \boldsymbol{P'} &= \boldsymbol{R}(\boldsymbol{P}-\boldsymbol{t}) \\ &= \boldsymbol{R}(\boldsymbol{P}-\boldsymbol{t}\frac{H}{H}) \end{aligned} \tag{A.1}$$

Since the equation for the plane π is known:

$$\boldsymbol{P}\cdot\boldsymbol{n} = \boldsymbol{n}^T\boldsymbol{P} = H \tag{A.2}$$

Finally, we have,

$$\boldsymbol{P'} = \boldsymbol{R}(\boldsymbol{I}-\frac{\boldsymbol{t}\boldsymbol{n}^T}{H})\boldsymbol{P} \tag{A.3}$$

We call the matrix $\boldsymbol{R}(\boldsymbol{I}-\frac{\boldsymbol{t}\boldsymbol{n}^T}{H})$ a *homography matrix*, and denote it as $\boldsymbol{A}$,

$$\boldsymbol{A} = \boldsymbol{R}(\boldsymbol{I} - \frac{\boldsymbol{t}\boldsymbol{n}^T}{H}) \tag{A.4}$$

This matrix encodes the mapping relationships among the three planes represented by π and the two image planes. Since in a homogeneous representation,

$$\boldsymbol{P} \cong \boldsymbol{p} \tag{A.5}$$

$$\boldsymbol{P'} \cong \boldsymbol{p'} \tag{A.6}$$

then,

$$\boldsymbol{p'} \cong \boldsymbol{A}\boldsymbol{p} \tag{A.7}$$

There is a scale factor k under a projective transformation such that

$$\boldsymbol{p'} = k\boldsymbol{A}\boldsymbol{p} \tag{A.8}$$

Comparing Eq. A.3 and Eq. A.8, it is obvious that

$$k = \frac{Z'}{Z} \tag{A.9}$$

i.e. this scale factor is the ratio of the two depths of the same point represented in the two camera systems.

The homography matrix $\boldsymbol{A}$ captures the mapping relationship between two cameras w.r.t. a 3D plane π. Therefore, given each correspondence between the two camera images of a 3D point in plane π, Eq. A.8 gives rise to two equations. This is why, given four coplanar correspondences between two cameras, the normalized form of the homography matrix (see Chapter 4) can be determined by solving a linear system.

APPENDIX B

SVD AND ITS RELATIONSHIP TO EIGENVALUES OF SYMMETRIC MATRICES

This review of SVD and its relationship to eigenvalues of symmetric matrices is based on [34]. Let $\boldsymbol{A}$ be an $m \times n$ matrix ($m \geq n$). The singular value decomposition (SVD) of $\boldsymbol{A}$ is an $m \times n$ column-orthogonal matrix $\boldsymbol{U}$, an $n \times n$ row-orthogonal matrix $\boldsymbol{V}$, and an $n \times n$ diagonal matrix $\boldsymbol{\Lambda}$, such that

$$\boldsymbol{U}^T \boldsymbol{A} \boldsymbol{V} = \boldsymbol{\Lambda} \tag{B.1}$$

where

$$\boldsymbol{\Lambda} = \text{diag}\ (\sigma_1, \sigma_2, ..., \sigma_n) \tag{B.2}$$

satisfying

$$\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_n$$

The n real values $\sigma_1, \sigma_2, ..., \sigma_n$ are called *singular values* of matrix $\boldsymbol{A}$. It can also be shown that the matrix $\boldsymbol{V}$ is column-orthogonal.

Taking the transpose of Eq. B.1, we have

$$\boldsymbol{V}^T \boldsymbol{A}^T \boldsymbol{U} = \boldsymbol{\Lambda} \tag{B.3}$$

Multiplying Eq. B.1 by Eq. B.3, we obtain

$$\boldsymbol{V}^T (\boldsymbol{A}^T \boldsymbol{A}) \boldsymbol{V} = \boldsymbol{\Lambda}^2 \tag{B.4}$$

and

$$\boldsymbol{U}^T (\boldsymbol{A}\boldsymbol{A}^T) \boldsymbol{U} = \boldsymbol{\Lambda}^2 \tag{B.5}$$

where

$$\boldsymbol{\Lambda}^2 = \text{diag } ({\sigma_1}^2, {\sigma_2}^2, ..., {\sigma_n}^2) \tag{B.6}$$

From Eq. B.4, the n real values

$${\sigma_1}^2 \geq {\sigma_2}^2 \geq ... \geq {\sigma_n}^2$$

are the eigenvalues of the symmetric matrix $\boldsymbol{A}^T \boldsymbol{A}$, and the n columns of matrix $\boldsymbol{V}$ are the corresponding eigenvectors of this symmetric matrix.

Similarly, if we define matrices $\boldsymbol{W}$ and $\boldsymbol{\Sigma}$ as follows:

$$\boldsymbol{W} = [\boldsymbol{U} || \boldsymbol{0}] \tag{B.7}$$

where $\boldsymbol{0}$ is an $m \times (m-n)$ matrix with all elements set to 0, i.e. matrix $\boldsymbol{W}$ is an augmented matrix of $\boldsymbol{U}$ formed by adding $m-n$ column vectors whose elements are all 0. $\boldsymbol{W}$ is then an $m \times m$ square matrix. Similarly, define $\boldsymbol{\Sigma}$ as the augmented square matrix of $\boldsymbol{\Lambda}$

$$\boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Lambda} & \\ & \boldsymbol{0} \end{pmatrix} \tag{B.8}$$

where $\boldsymbol{0}$ is an $(m-n) \times (m-n)$ square matrix with all elements set to 0.

Thus, Eq. B.5 is equivalent to

$$\boldsymbol{W}^T(\boldsymbol{A}\boldsymbol{A}^T)\boldsymbol{W} = \boldsymbol{\Sigma}^2 \tag{B.9}$$

Eq. B.9 clearly indicates that the symmetric matrix $\boldsymbol{A}\boldsymbol{A}^T$ has m eigenvalues as

$$\sigma_1{}^2 \geq \sigma_2{}^2 \geq ... \geq \sigma_n{}^2 \geq 0 = 0...0$$

and has as its corresponding eigenvectors the m column vectors of matrix $\boldsymbol{W}$, including $m - n$ 0 vectors. Since the matrix $\boldsymbol{A}\boldsymbol{A}^T$ has rank at most n, it thus always has at least $m - n$ 0 eigenvalues and $m - n$ 0 eigen vectors.

BIBLIOGRAPHY

[1] Anandan, P. *Measuring Visual Motion from Image Sequences.* Ph.D. Thesis, COINS TR 87-21, 1987.

[2] Aubert, D., Kluge, K., and Thorpe, C. Autonomous navigation of structured city roads. In *Proc. SPIE Mobile Robots*, 1990.

[3] Badal, S., R., S., B., D., and A., H. A practical obstacle detection and avoidance system. In *Proc. 2nd IEEE Workshop on Applications of Computer Vision.* IEEE, 1994.

[4] Baker, R. Methods of stereophotomicrography. *Photographic Journal*, May 1936.

[5] Ballard, D., and Brown, C. *Computer Vision.* Prentice-Hall, Inc., Englewood Cliffs, NJ 07632, 1982.

[6] Banta, L., and Bubnick, T. An auto-calibration system for vision-servoed robots. In *Proc. the 4th Int. Conf. on Advanced Robotics*, Columbus, Ohio, June 1989. Springer-Verlag.

[7] Banta, L., and Bubnick, T. Visual servoing of a robot assembly task. In *Proc. the 4th Int. Conf. on Advanced Robotics*, Columbus, Ohio, June 1989. Springer-Verlag.

[8] Beni, G., and Hackwood, S. *Recent Advances in Robotics.* John Wiley & Sons, 1985.

[9] Binford, T. Visual perception by computer. In *Proc. IEEE Conference on Systems and Control*, December 1971.

[10] Born, G. Uber die nasenhohlen und den thranennasengang der amphibien. *Morphologisches Jahrbuch*, 2, 1876.

[11] Chenavier, F., and Crowley, J. Position estimation for a mobile robot using vision and odometry. In *Proc. IEEE International Conference on Robotics and Automation.* IEEE, 1992.

[12] Collins, R. *Model Acquisition Using Stochastic Projective Geometry.* Ph.D. Dissertation, CMPSCI TR 95-70, Unvi. of Mass., 1995.

[13] Collins, R., and Weiss, R. Vanishing point calculation as a statistical inference on the unit sphere. In *Proc. ICCV*, Osaka, Japan, Dec. 1990. IEEE.

[14] Courtney, J., Magee, M., and Aggarwal, J. Robot guidance using computer vision. *Pattern Recognition*, 1984.

[15] Craig, J. *Introduction to Robotics Mechanics and Control.* Addison-Wesley Publishing Company, 1986.

[16] Crisman, J., and Thorpe, C. *Vision and Navigation, The Carnegie Mellon Navlab*, chapter 2: Color Vision for Road Following. In Thorpe [111], 1990.

[17] D., M. *Vision.* W.H. Freeman and Company, San Francisco, 1982.

[18] Daily, M., Harris, J., and Reiser, K. Detecting obstacles in range imagery. In *Proc. IUW.* Morgan Haufmann, 1987.

[19] Daily, M., Harris, J., and Reiser, K. An operational perception system for cross-country navigation. In *Proc. IUW.* Morgan Haufmann, 1988.

[20] Dean, T., Basye, K., and Chekaluk, R. Coping with uncertainty in a control system for navigation and exploration. In *Proc. 8th National Conference on Artificial Intelligence.* AAAI-MIT, 1990.

[21] Dickmanns, E., and Grafe, V. Applications of dynamic monocular machine vision. *Machine Vision and Applications*, 1, 1988.

[22] Dickmanns, E., and Grafe, V. Dynamic monocular machine vision. *Machine Vision and Applications*, 1, 1988.

[23] Enkelmann, W. Obstacle detection by evaluation of optical flow fields. In *Proc. 1st ECCV*, 1990.

[24] Faugeras, O. What can be seen in three dimensions with an uncalibrated stereo rig? In *Proc. 2nd ECCV*, pages 563–578, Santa Margherita Ligure, Italy, May 1992. Springer-verlag.

[25] Faugeras, O., Luong, Q.-T., and Maybank, S. Camera self-calibration: Theory and experiments. In *ECCV.* Springer-Verlag, 1992.

[26] Faugeras, O., Luong, Q.-T., and Maybank, S. Camera self-calibration: Theory and experiments. In *Proc. 2nd ECCV*, pages 321–334, Santa Margherita Ligure, Italy, May 1992. Springer-Verlag.

[27] Faugeras, O., and Lustman, F. Motion and structure from motion in a piecewise planar environment. *International Journal of Pattern Recognition and Artificial Intelligence*, 2:485–508, 1988.

[28] Fennema, C., Hanson, A., Riseman, E., Beveridge, J., and Kumar, R. Model-directed mobile robot navigation. *Trans. Systems, Man, and Cybernetics*, 20(6):1352–1369, 1990.

[29] Fischler, M. A., and Bolles, R. C. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, June 1981.

[30] Foldyna, J. Use of stereoplotter std-2 in palaeontology for the morphological evaluation of fossil shells. *Photogrammetric Engineering*, 23, 1957.

[31] Fu, K., Gonzalez, R., and Lee, C. *Robotics: Control, Sensing, Vision, and Intelligence.* McGraw-Hill Inc., 1987.

[32] Fukui, I. Tv image processing to determine the position of a robot vehicle. *Pattern Recognition*, 1981.

[33] Gaunt, W., and Gaunt, P. *Three Dimensional Reconstruction in Biology.* University Park Press, 1978.

[34] Golub, G., and Loan, C. *Matrix Computations, 2nd Ed.* The Johns Hopkins University Press, 1989.

[35] Gopal, M. *Modern Control System Theory.* Wiley, 1993.

[36] Grandjean, P., and Matthies, L. Perception control for obstacle detection by a cross-country rover. In *Proc. ICRA*, pages 20–27, Atlanta, Georgia, May 1993. IEEE.

[37] Green, H. Technique of plastic reconstruction. *Nature*, 139, 1937.

[38] Gremban, K., Thorpe, C., and Kanade, T. Geometric camera calibration using systems of linear equations. In *Proc. Robotics and Automation*, pages 562–567. IEEE, 1988.

[39] Group, T. K. *Khoros Manual, Volume I, User's Guide.* University of New Mexico, 1991.

[40] Group, T. K. *Khoros Manual, Volume II, Programmer's Guide.* University of New Mexico, 1991.

[41] Group, T. K. *Khoros Manual, Volume III, References Manual.* University of New Mexico, 1991.

[42] Hallert, B. A symposium: Non-topographic photogrammetry: Introduction. *Photogrammetric Engineering*, 19, 1953.

[43] Hanson, A., Riseman, E., and Weems, C. Progress in computer vision at the university of massachusetts. In *IUW*, pages 39–47. Morgan Haufmann, April 1993.

[44] Hartley, R. Estimation of relative camera positions for uncalibrated cameras. In *ECCV*. Springer-Verlag, 1992.

[45] Hartley, R. Camera calibration using line correspondences. In *IUW*. Morgan Kaufmann Publishers, Inc., 1993.

[46] Hartley, R. Cheirality invariants. In *IUW*. Morgan Kaufmann Publishers, Inc., 1993.

[47] Hartley, R. An algorithm for self calibration from several views. In *CVPR*. IEEE, 1994.

[48] Hartley, R. Lines and points in three views — a unified approach. In *IUW*. Morgan Haufmann Publishers, Inc., 1994.

[49] Hartley, R. Projective reconstruction from line correspondences. In *CVPR*. IEEE, 1994.

[50] Hartley, R. In defence of the 8-point algorithm. In *ICCV*. IEEE, 1995.

[51] Hartley, R. A linear method for reconstruction from lines and points. In *ICCV*. IEEE, 1995.

[52] Hartley, R., Gupta, R., and Chang, T. Stereo from uncalibrated cameras. In *CVPR*. IEEE, 1992.

[53] Hartley, R., and Sturm, P. Triangulation. In *IUW*. Morgan Haufmann Publishers, Inc., 1994.

[54] Hayati, S. Robot arm geometric link calibration. In *Proc. 22nd Conf. on Decision and Control*, pages 1477–1483. IEEE, 1983.

[55] His, W. *Untersuchungen uber die erste Anlage des Wirbeltierleibes*. Leipzig: F.C.W. Vogel, 1868.

[56] Hong, J., Tan, X., Pinette, B., Weiss, R., and Riseman, E. Image-based navigation using 360 degree views. In *Proc. Image Understanding Workshop*, 1990.

[57] Horn, B. *Robot Vision*. The MIT Press, 1986.

[58] Huang, T., and Faugeras, O. Some properties of the e matrix in two-view motion estimation. *Trans. PAMI*, 11(12), 1989.

[59] Jacobs, D. *Recognizing 3-D Objects Using 2-D Images*. Ph.D. Dissertation, MIT, 1992.

[60] Jacobs, D. 2-d images of 3-d oriented points. In *Proc. CVPR*. IEEE, 1993.

[61] Jain, R. Segmentation of frame sequences obtained by a moving observer. *PAMI*, 6:624–629, Sept. 1984.

[62] Kabuka, M., and Arenas, A. Position verification of a mobile robot using standard pattern. *IEEE Journal of Robotics and Automation*, RA-3, 1987.

[63] Kahn, P., Kitchen, L., and Riseman, E. Real-time feature extraction: A fast line finder for vision-guided robot navigation. *Trans. PAMI*, 12(11):1098–1102, 1990.

[64] Kanatani, K., and Onodera, Y. Anatomy of camera calibration using vanishing points. *IEICE Trans. Infor. Sys.*, 74(10):3369–3378, 1991.

[65] King, F., Puskorius, G., Yuan, F., Meier, R., Jevabalan, V., and Feldkamp, L. Vision guided robots for automated assembly. In *Proc. Robotics and Automation*, pages 1611–1616. IEEE, 1988.

[66] Koenderink, J. J., and Doon, A. V. Affine structure from motion. *J. Opt. Soc. Am. A.*, 8:377–385, 1990.

[67] Krotkov, E. Mobile robot localization using a single image. In *Proc. Robotics and Automation*, Scottsdale, Arizona, May 1989. IEEE.

[68] Kumar, R. *Model Dependent Inference of 3D Information From a Sequence of 2D Images.* Ph.D. Dissertation, COINS TR 92-04, Univ. Mass., 1992.

[69] Kumar, R., and Anandan, P. Shape recovery from multiple views: A parallax based approach. *David Sarnoff Tech Report*, 1994.

[70] Kumar, R., and Hanson, A. Robust estimation of camera location and orientation from noisy data having outliers. In *Proc. Workshop on Interpretation of 3D Scenes*, Austin, TX, Nov. 1992.

[71] Lee, H.-J., and Deng, C.-T. Camera model determination using multiple frames. In *Proc. CVPR*, Lahaina, Hawaii, June 1991. IEEE.

[72] Leonard, J., and Durrant-Whyte, H. *Directed Sonar Sensing for Mobile Robot Navigation.* Kluwer Academic Publishers, 1992.

[73] Longuet-Higgins, H. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, 1981.

[74] Lowe, D. The viewpoint consistency constraint. *IJCV*, 1(1), 1987.

[75] Magee, M., and Aggarwal, J. Determining the position of a robot using a single calibration object. In *Proc. IEEE International Conference on Robotics.* IEEE, 1984.

[76] Mannen, H. Reconstruction of axonal trajectory of individual neurons in the spinal cord using golgi-stained serial sections. *Journal of Comparative Neurology*, 159, 1975.

[77] Marr, D., and Poggio, T. Coorperative computation of stereo disparity. *Science*, 194, 1976.

[78] Marr, D., and Poggio, T. A theory of human stereo vision. *AI Memo*, (451), 1977.

[79] Matthies, L. Toward stochastic modeling of obstacle detectability in passive stereo range imagery. In *Proc. CVPR.* IEEE, June 1992.

[80] Matthies, L., and Grandjean, P. Stochastic performance modeling and evaluation of obstacle detectability with imaging range sensors. In *Proc. CVPR.* IEEE, June 1993.

[81] Maybank, S. *Theory of Reconstruction from Image Motion.* Springer-Verlag, 1993.

[82] Maybank, S., and Faugeras, O. A theory of self-calibration of a moving camera. *IJCV*, 1992.

[83] Mohr, R., Veillon, F., and Quan, L. Relative 3d reconstruction using multiple uncalibrated images. In *CVPR.* IEEE, 1993.

[84] Moravec, H. Towards automatic visual obstacle avoidance. In *Proc. 5th IJCAI*, August 1977.

[85] Nelson, R. Qualitative detection of motion by a moving observer. In *CVPR*, pages 173–178. IEEE, 1991.

[86] Nelson, R., and Aloimonos, J. Obstacle avoidance using flow field divergence. *Trans. PAMI*, 11(10), 1989.

[87] Odhner, N. Eine neue graphische methode zur rekonstruktion von schnittserien in schrager stellung. *Anatomischer Anzeiger*, 39, 1911.

[88] Peter, K. Uber graphische rekonstruktion in schragansicht. *Zeitschrift fur wissenschaftliche Mikroskopie und fur miskroskopische Technik*, 39, 1922.

[89] Pinette, B. *Image-Based Navigation Through Large-Scale Environments.* Ph.D. Dissertation, CMPSCI TR 94-87, University of Massachusetts, 1994.

[90] Pomerleau, D. *Vision and Navigation, The Carnegie Mellon Navlab*, chapter 5: Neural Network Based Autonomous Navigation. In Thorpe [111], 1990.

[91] Ponce, J., Marimont, D. H., and Cass, T. A. Relative stereo and motion reconstruction. *Beckman Institute Technical Report*, 1993.

[92] Press, W., Teukolsky, S., Vetterling, W., and Flannery, B. *Numerical Recipes in C, The Art of Scientific Computing, 2nd Edition.* Cambridge University Press, 1992.

[93] Quam, L., and Hannah, M. Stanford automated photogrammetry research. *Stanford AI Lab*, 1974.

[94] Sack, W. Rapid wax plate modelling. *Anatomical Record*, 154, 1966.

[95] Sauthoff, N., and Von Goeler, S. Techniques for the reconstruction of two-dimensional images from projections. *Government Document, Princeton University, Plasma Physics Lab*, 1978.

[96] Sawhney, H. S. 3d geometry from planar parallax. In *CVPR*. IEEE, 1994.

[97] Schaeffer, K. Die reconstruction mittels zeichnung. *Zeitschrift fur wissenschaftliche Mikroskopie und fur mikroscopische Technik*, VII, 1890.

[98] Shapira, R. A technique for the reconstruction of a straight-edge, wire-frame object from two or more central projections. *CGIP*, 3(4), 1974.

[99] Shapira, R., and Freeman, H. Computer description of bodies bounded by quadratic surfaces from a set of important projections. *IEEE Trans. Computers*, 27(9), 1978.

[100] Shashua, A. Projective depth: A geometric invariant for 3d reconstruction from two perspective/orthographic views and for visual recognition. In *ICCV*. IEEE, 1993.

[101] Shashua, A. Trilinearity in visual recognition by alignment. In *ECCV*. Springer-Verlag, 1994.

[102] Shashua, A., and Navab, N. Relative affine structure: Theory and application to 3d reconstruction from perspective views. In *CVPR*. IEEE, 1994.

[103] Shiu, Y., and Ahmad, S. Calibration of wrist-mounted robotic sensors by solving homogeneous transform equations of the form ax-ab. *Trans. Robotics and Automation*, 5(1), 1989.

[104] Slama, C., Theurer, C., and Henriksen, S. *Manual of Photogrammetry, Fourth Edition.* American Society of Photogrammetry, 1980.

[105] Solder, U., and Graefe, V. Object detection in real time. *SPIE Mobile Robots V*, 1388, 1990.

[106] Storjohann, K., Zielke, T., Mallot, H., and W., V. S. Visual obstacle detection for automatically guided vehicles. In *Proc. ICRA*. IEEE, 1990.

[107] Strasser, H. Ueber die methoden der plastischen reconstruction. *Zeitschrift fur wissenschaftliche Mikroskopie und fur mikroskopische Technik*, 4, 1887.

[108] Streeter, G. The development of the cranial and spinal nerves in the occipital region of the human embryo. *American Journal of Anatomy*, 4, 1905.

[109] Thompson, W., Lechleider, P., and Stuck, E. Detecting moving objects using the rigidity constraint. *PAMI*, 15:162–166, 1993.

[110] Thompson, W., and Pong, T.-C. Detecting moving objects. *IJCV*, 4:39–57, 1990.

[111] Thorpe, C., editor. *Vision and Navigation, The Carnegie Mellon Navlab.* Kluwer, 1990.

[112] Thuringer, J. A suggestion for improvement in projection and drawing apparatuses. *Anatomical Record*, 19, 1920.

[113] Tsai, R. An efficient and accurate camera calibration technique. In *Proc. CVPR '86*, Miami Beach, June 1986. IEEE.

[114] Tsai, R., and Huang, T. Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces. *Trans. PAMI*, 6(1):13–27, 1984.

[115] Tsai, R., and Lens, R. Real time versatile robotics hand/eye calibration using 3d machine vision. In *Proc. Robotics and Automation*, pages 554–561. IEEE, 1988.

[116] Tsuji, S., Yagi, Y., and Asada, M. Dynamic scene analysis for a mobile robot in a man-made environment. In *Proc. Robotics and Automation*, St. Louis, Missouri, March 1985. IEEE.

[117] Tur, J. Sur l'application d'une methode graphique aux recerches embryologiques. *Bibliographie Anatomique*, 10, 1902.

[118] Turner, K. *Computer Perception of Curved Objects Using A Television Camera.* Ph.D. Dissertation, Univ. Edinburgh, 1974.

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

[120] Veitschegger, W., and Wu, C. Robot calibration and compensation. *Journal of Robotics and Automation*, 4(6), 1988.

[121] Waxman, A., LeMoigne, J., Davis, L., Srinivasan, B., Kushner, T., Liang, E., and Siddalingaiah, T. A visual navigation system for autonomous land vehicle. *IEEE Journal of Robotics and Automation*, RA-3, 1987.

[122] Weng, J., Ahuja, N., and Huang, T. Motion and structure from point correspondences with error estimation: Planar surfaces. *Trans. on Signal Processing*, 39(12), 1991.

[123] Weng, J., Huang, T., and Ahuja, N. Motion and structure from line correspondences: Closed-form solution, uniqueness and optimization. *Trans. PAMI*, 14(3), 1992.

[124] Wesley, M., and Markovsky, G. Fleshing out projections. *Compt. Sci. Dept., Research Report*, (RC8884), 1981.

[125] Whitney, D., Lozinski, C., and Rourke, J. Industrial robot calibration method and results. In *Proc. Int. Computers in Engineering Conf. and Exhibit*, pages 92–100, 1984.

[126] Williams, L., and Hanson, A. Translating optical flow into token matches and depth from looming. In *Proc. ICCV*, Clearwater, FL, Dec. 1988. IEEE.

[127] Wright, A. Three dimensional shape analysis of fine grained sediments. *Geological Abstracts*, 5, 1957.

[128] Young, G.-S., Hong, T.-H., Herman, M., and Yang, C. Obstacle detection for a vehicle using optical flow. In *SAE Intelligent Vehicle*, Detroit, MI, July 1992. IEEE.

[129] Zhang, Z., and Hanson, A. Scaled euclidean 3d reconstruction based on externally uncalibrated cameras. In *International Symposium on Computer Vision*. IEEE, 1995.

[130] Zhang, Z., Weiss, R., and Hanson, A. Automatic calibration for a robot navigation system. *CMPSCI TR92-70*, 1992.

[131] Zhang, Z., Weiss, R., and Hanson, A. Automatic calibration and visual servoing for a robot navigation system. In *Proc. IEEE International Conference on Robotics and Automation*. IEEE, 1993.

[132] Zhang, Z., Weiss, R., and Hanson, A. Automatic calibration and visual servoing for a robot navigation system. *CMPSCI TR93-14*, 1993.

[133] Zhang, Z., Weiss, R., and Hanson, A. Qualitative obstacle detection. In *Proc. CVPR*. IEEE, 1994.

[134] Zhang, Z., Weiss, R., and Riseman, E. Feature matching in 360° waveforms for robot navigation. In *Proc. IEEE International Conference on Computer Vision and Pattern Recognition*. IEEE, 1991.

[135] Zhang, Z., Weiss, R., and Riseman, E. Segment-based matching for visual navigation. *CMPSCI TR91-35*, 1991.