NOTE TO USERS

This reproduction is the best copy available.

UMI®

Université d'Ottawa · University of Ottawa

Université d'Ottawa · University of Ottawa

FACULTÉ DE ÉTUDES SUPÉRIEURES ET POSTDOCTORALES

FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

Hassan HAJJDIAB

AUTEUR DE LA THÈSE - AUTHOR OF THESIS

Ph.D (Computer Science)

GRADE - DEGREE

School of Information, Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT - FACULTY, SCHOOL, DEPARTMENT

TITRE DE LA THÈSE - TITLE OF THE THESIS

Mobile Robot Localization, Map Building and Obstacle Reconstruction

R. Laganière

DIRECTEUR DE LA THÈSE - THESIS SUPERVISOR

CO-DIRECTEUR DE LA THÈSE - THESIS CO-SUPERVISOR

EXAMINATEURS DE LA THÈSE - THESIS EXAMINERS

P. Bose

B. Boufama

P. Payeur

J. Zhao

J.-M. De Koninck, Ph.D

LE DOYEN DE LA FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTDOCTORALES

DEAN OF THE FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

VISION-BASED LOCALIZATION, MAP BUILDING AND OBSTACLE RECONSTRUCTION IN GROUND PLANE ENVIRONMENTS

By

Hassan Hajjdiab

SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
AT
UNIVERSITY OF OTTAWA
OTTAWA, ONTARIO
NOVEMBER 2004

© Copyright by Hassan Hajjdiab, 2004

Library and Archives Canada

Published Heritage Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et Archives Canada

Direction du Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 0-494-01705-8
Our file Notre référence
ISBN: 0-494-01705-8

NOTICE:
The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

AVIS:
L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

UNIVERSITY OF OTTAWA
SCHOOL OF INFORMATION TECHNOLOGY AND ENGINEERING

The undersigned hereby certify that they have read and recommend to the Faculty of Graduate Studies for acceptance a thesis entitled "**Vision-based localization, map building and obstacle reconstruction in ground plane environments**" by **Hassan Hajjdiab** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy**.

Dated: November 2004

External Examiner: _____
Boubaker Boufama

Research Supervisor: _____
Robert Laganière

Examing Committee: _____
Jit Bose

_____
Pierre Payeur

_____
Jiying Zhao

This work is dedicated to my parents and my dear wife for their patience and continuous support.

Table of Contents

List of Figures

Abstract

The work described in this thesis develops the theory of 3D obstacle reconstruction and map building problems in the context of a robot, or a team of robots, equipped with one camera mounted on board. The study is composed of many problems representing the different phases of actions taken by the robot.

This thesis first studies the problem of image matching for wide baseline images taken by moving robots. The ground plane is detected and the inter-image homography induced by the ground plane is calculated. A novel technique for ground plane matching is introduced using the overhead view transformation.

The thesis then studies the simultaneous localization and map building (SLAM) problem for a team of robots collaborating in the same work site. A vision-based technique is introduced in this thesis to solve the SLAM problem.

The third problem studied in this thesis is the 3D obstacle reconstruction of the obstacles lying on the ground surface. In this thesis a Geometric/Variational level set method is proposed to reconstruct the obstacles detected by the robots.

Acknowledgements

It is my pleasure to acknowledge the many people who offered their help and advice throughout the course of my work.

First of all, I would like to express my sincere gratitude to my supervisor Dr. Robert Laganière, for his knowledge, guidance and encouragement. It has been his kindness above all that has given me the many constructive opportunities I needed to complete this work.

I would like also to thank my committee members, Dr. Eric Dubois, Dr. Pierre Payeur, Dr. Jit Bose and Dr. Boubaker Boufama for their enthusiasm, support and inspiration.

Notation

Points

$\mathbf{X} = [X, Y, Z, 1]^T$	A point in the 3D homogeneous world plane
$\mathbf{x} = [x, y, 1]^T$	A point in the 2D homogeneous image coordinate system
$\bar{\mathbf{X}} = [X, Y, Z]^T$	A point in the 3D Euclidean world plane
$\bar{\mathbf{x}} = [x, y]^T$	A point in the 2D image coordinate system

Camera Parameters

f	Focal length
ξ	Aspect ratio
s	skew
$[x_0, y_0]$	Principal point
$\mathbf{K}$	The 3×3 Calibration matrix
$\mathbf{P}$	The 3×4 Projection matrix

Camera Geometry

$\mathbf{R}$	The 3×3 rotation matrix
$\mathbf{t}$	The 3×1 translation vector
$\mathbf{H}$	The 3×3 homography transformation between two images
$\mathbf{H}_O$	The 3×3 overhead view homography transformation
$\mathbf{E}$	The 3×3 essential matrix
$\mathbf{F}$	The 3×3 fundamental matrix

Matrix/Vector Operators

$\mathbf{a}.\mathbf{b}$	Scalar product of vectors $\mathbf{a}$ and $\mathbf{b}$
$\mathbf{a} \times \mathbf{b}$	Cross product of vectors $\mathbf{a}$ and $\mathbf{b}$
$[\mathbf{a}]_\times$	Anti-symmetric matrix of vector $\mathbf{a}$
$det(\mathbf{M})$	Determinant of matrix $\mathbf{M}$
$\mathbf{M}^{-1}$	Inverse of matrix $\mathbf{M}$
$\mathbf{M}^T$	Transpose of matrix $\mathbf{M}$

Chapter 1

Introduction

The development of autonomous mobile robots has gained remarkable attention in the last two decades. Such robots must be able to distinguish portions of the environment where they can freely move from other portions that are impossible to traverse. This implies that they must have the ability to identify the obstacles in the work area, and localize them with respect to their current position. These navigation systems operate generally on a planar surface on which the obstacles reside. In this thesis, the problem of obstacle detection and 3D reconstruction in the context of a robot simply equipped with a single camera is studied. The objective is to obtain a 3D representation of the environment and build a map of the environment inside which the autonomous vehicle would like to move.

In this work the problem of obstacle detection and localization is studied using a collection of sparse views of the scene. These views could have been obtained by having the robot patrolling along the periphery of the site or by a team of robots collaborating together to obtain a representation of the environment. The problem investigated in this thesis can be stated as follows:

Given a robot (or a team of robots) moving on a planar ground with a

single camera mounted on board.

Is it possible to localize the robot(s), build an environmental map along the robot path and perform a 3D reconstruction of the obstacles encountered assuming no ***a priori*** *knowledge of the environment (other than the fact that the work surface is planar)?*

To answer the above question, we propose a vision-based approach that depends only on visual information collected by a mobile robot. In order to achieve this goal, the robot must be able to perform image matching, camera motion estimation, obstacle detection and robot localization. In this study we will simply assume that we have an approximate knowledge of the camera orientation and of its height with respect to the ground plane. These orientation measurements do not have to be accurate as the algorithm presented here also includes a self-calibration phase.

1.1 Vision-based mobile robots

Mobile robot development has received considerable attention in the last few years. The mobile robot's ability to sense and perceive the work environment is a fundamental step towards developing autonomous robots. The robot may utilize vision to sense the environment or may use other active sensors like radio, acoustic, laser, magnetic or tactile. However, in the case of multiple robots moving in the same worksite, the active signals emitted may interfere with each other. Despite its computational complexity compared to other sensors, computer vision remains a powerful way to sense the environment. The visual information reported by the autonomous robots, especially in places where humans cannot reach, offers rich perception about the environment more than any other sensing signal. In addition, the low cost of the camera

system makes computer vision an efficient choice.

For a large number of applications the surface where the mobile robot systems are moving on can be modeled as a plane. Examples of systems that fall into this category are indoor mobile robots [88] and Automated Highway Systems (AHS) Vehicles [94]. In this thesis, an extensive study of a vision based multi-robot system exploring a previously unknown environment is presented. A team of robots moving on a planar surface and sharing a common workplace with a single camera mounted on each robot. The robots are able to sense the environment, share information about their mutual locations with respect to a commonly generated map and report visual information about the explored environment.

1.2 Contribution

This thesis contributes on various aspects of computer vision and image understanding. The main contributions of this thesis are as follows:

- **Ground plane image matching for sparse view images taken by an autonomous robot**:

  In this thesis an algorithm is proposed to match widely separated images of the work area of a moving robot. The matching process between the two images is performed using the overhead view transformation. The ground plane is then detected and the inter-image homography induced by the ground plane is calculated. This homography is used to estimate the camera motion between the two views. The information required in the algorithm is an estimate of the orientation with respect to the ground plane. The exact orientation is then calculated in a self calibration phase based on homography refinement.

- **Vision-based multi-robot Simultaneous Localization and Mapping (SLAM):**
  A vision-based approach for the multi-robot Simultaneous Localization and Mapping (SLAM) problem is presented. In this work, both the single-robot SLAM and the multi-robot SLAM are studied. In the single-robot SLAM, the robot incrementally estimates its current position with respect to its previous position using the calculated homography between the two views. The multi-robot SLAM is a generalization of the single-robot case, each robot in the team starts by building a local map of the environment. When an overlapping scene occurs between any two robots, a joint map between them is built. If such overlapping scenes occur between each pair of robots in the team, a joint map of the environment can be generated.

- **Ground plane obstacle reconstruction:**
  This thesis introduces a geometric/level set method to reconstruct the ground plane obstacles located in a robot environment. The robot equipped with a camera takes several images of the scene from which calibration information is obtained. Obstacles are then detected and their structures are reconstructed using level set surface evolution equations that minimize an energy functional characterizing properties of the 3D obstacle structure and its conformity to the acquired images. The method integrates a voxel coloring formulation and uses color invariants to compare image intensities from the available widely separated views. It also incorporates shape-from-silhouette concepts in the functional to minimize in order to account for possibly non-textured obstacle object surfaces.

1.3 Thesis outline

This thesis begins in Chapter 2 with a review of the fundamental concepts in computer vision. It first reviews the pin-hole perspective camera model and then provides a review of the epipolar geometry with derivation of both the essential matrix and the fundamental matrix. Then the plane-to-plane homography transformation between a plane in the world plane and its image in the image plane is presented. Followed by an illustration of the inter-image homography transformation induced by a plane viewed by two cameras. Finally, in this chapter a review about camera calibration from planar surfaces in the scene under observation is discussed.

Chapter 3 covers the overhead view mosaic building and image matching in a multi-robot environment. This chapter starts by introducing the overhead view transformation and shows how an overhead view mosaic can be built. Then our wide baseline matching algorithm is introduced. The algorithm is presented by first matching corners using the overhead views. The matching is based on Hilbert's color invariants. Then the calculation and refinements of the inter-image homography is presented. Finally, the complexity and statistical analysis of the our proposed algorithm is studied.

Chapter 4 starts with an overview of the robot localization and map building (SLAM) problem. The refined inter-image homography transformation calculated in Chapter 3 are used to estimate the camera motion. Then the robot localization algorithm is explained and two SLAM cases are studied. The single-robot SLAM is studied with an experimental example and then the multi-robot SLAM is also studied with an experimental example of two robots. For each case, map of the environment

and location graph are provided. Finally, a summary of the chapter is included.

Chapter 5 starts with a review of some 3D reconstruction algorithms. The structure from motion, shape from silhouette, voxel coloring and variational methods are introduced in the overview. Then a review of the level set methods with some experimental example are provided. Our Geometric/Variational algorithm is then discussed with detailed explanation of the level set formulation of the problem. Two examples are provided to verify the validity of the method.

Finally, Chapter 6 provides a summary and conclusion of the work introduced in this thesis and then discusses possible extensions of the work in the future.

1.4 Publications

Up to date, five publications have been published out of the work introduced in this thesis. The publications are as follows [27, 29, 30, 45, 28]:

Title : Wide baseline obstacle detection and localization

Authors : H. Hajjdiab, R. Elias and R. Laganière

Conference : in Proceedings of the IEEE Seventh International Symposium on Signal Processing and its Applications, ISSPA'03, (Paris, France,) vol i, pp. 21-24, July 2003.

Title : Vision-based robot localization

Authors : H. Hajjdiab and R. Laganière

Conference : in Proceedings of the 2^{nd} IEEE International Workshop on Haptic Audio Visual Environments and their Applications, HAVE'03, (Ottawa, Ontario, Canada) pp.19-24, September 2003.

Title : Vision-based Multi-Robot Simultaneous Localization and Mapping

Authors : H. Hajjdiab and R. Laganière

Conference : in Proceedings of the 1^{st} Canadian Conference on Computer and Robot Vision, CCCRV'04, (London, Ontario, Canada) May 2004.

Title : Shape-from-silhouettes, Photo-consistency, and Level Sets for the Reconstruction of Ground Plane Obstacles in a Robot Environment (submitted)

Authors : R. Laganière, A. Mitiche and H. Hajjdiab

Conference : British Machine Vision Conference, BMVC'04, (London, UK) September 2004.

Title : Obstacle Localization from Ground Plane Sparse Views (submitted)

Authors : H. Hajjdiab, R. Elias and R. Laganière

Journal : Robotics and Autonomous Systems

Chapter 2

The perspective camera

In this chapter the basic concepts that are used throughout the thesis are introduced. First the pin-hole perspective camera model is presented and the relation between a point in the 3D world and its projection in the image plane is illustrated. Then the epipolar geometry for two images of the same scene is explained. Finally the transformation induced by a known plane in a scene and the calibration of the cameras under such condition are discussed.

2.1 Introduction

2.1.1 Camera history

The pinhole box shown in Figure 2.1 is the simplest device that can perform the function of a camera. Theoretically, the tiny hole in the box allows only a single light ray to pass and thus produce an inverted image of the object. This principle has been utilized long before cameras came into use. Ancient Arabs discovered the principle of the pinhole camera by pinching a tiny hole in a dark tent and they could observe the inverted images of the outside illuminated objects. In the 1700's French artists used the same principle to trace the outlines of objects projected onto a wall opposite to

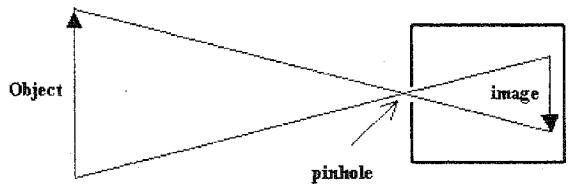


Figure 2.1: Schematic representation of a pinhole camera.

a pinhole. The principle of the camera as we know it today was materialized in 1839 when Louis Daguerre of France developed a photographic film that could capture a permanent record of images projected onto it. The combination of a pinhole box and the film became known as the camera [98].

2.1.2 Lens optics

Photographic plates or films are not sensitive enough to capture images through a pinhole in a short period of time. Allowing more light to pass through the pinhole to speed up the image capturing process by increasing the hole diameter results in blurred images. To avoid this effect, a lens is used to enlarge the hole and still retain focus. Lenses vary extensively in terms of quality and complexity. A simple converging lens is considered to have a negligible thickness and is represented by a single point at its center. The lens is symmetrical about its principal (or optical) axis. Rays from a long distance entering the lens parallel to the optical axis are deflected inward and intersect at a point on the optical axis known as the principal focus. The distance between the principal focus and the center of the lens is known as the focal length. The lens creates a focused inverted image at a distance v from the lens and obeys the following equation:

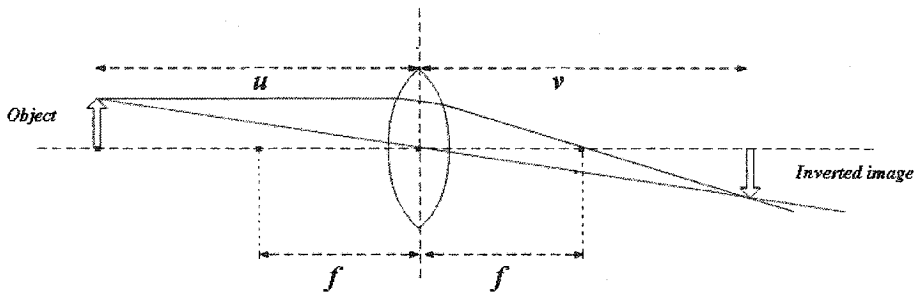


Figure 2.2: Convex lens optics

$$\frac{1}{u} + \frac{1}{v} = \frac{1}{f} \tag{2.1.1}$$

where

u: is the lens-object distance

v : is the lens-image distance

f: is the focal length

2.2 Perspective projection

The perspective projection can be modeled as shown in Figure 2.3. The point origin C is the center of projection of the 3D camera reference system. The image plane Π is parallel to the camera X-Y plane and displaced at a distance f (focal length) along the Z-axis from the origin. The principal point c is the orthogonal projection of C onto Π , and the projection line is the principal axis. A 3D point $\mathbf{X}$ of coordinates $[X, Y, Z, 1]^T$ in the world coordinate system is related to a point $\mathbf{x}$ of coordinates

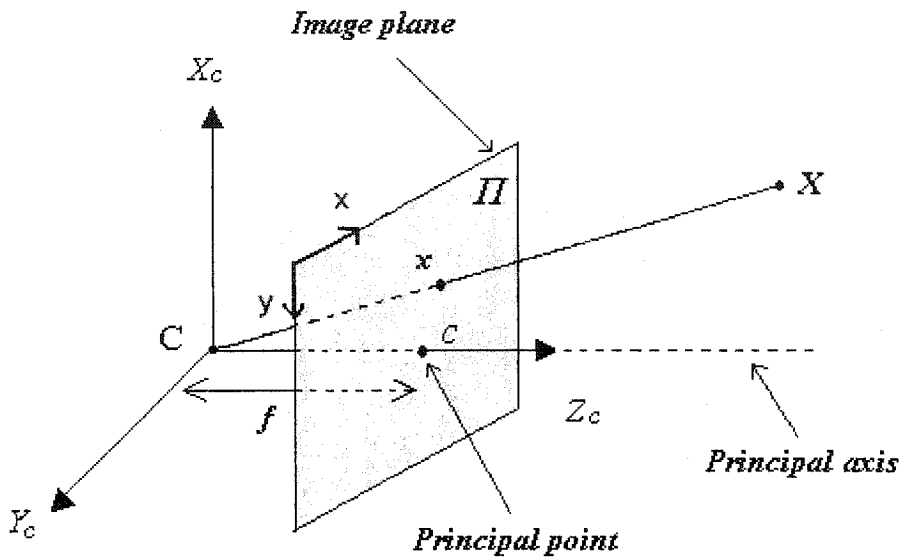


Figure 2.3: Standard perspective projection

$[x, y, 1]^T$ in the image coordinate system as follows:

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \mathbf{P} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.2.1}$$

where λ is an arbitrary scaling factor and $\mathbf{P}$ is a 3×4 matrix called the projection matrix [15].

The projection matrix models the projective relation between 3D points in the world coordinate system and the image . It can be decomposed as follows [15]:

$$\mathbf{P} = \mathbf{K}[\mathbf{R} \,|\, \mathbf{t}] \tag{2.2.2}$$

where $\mathbf{K}$ is a 3×3 matrix known as the camera calibration matrix and $\mathbf{R}$ is a 3×3 rotation matrix and $\mathbf{t}$ is the 3×1 translation vector. The matrix $[\mathbf{R} \,|\, \mathbf{t}]$ is known as

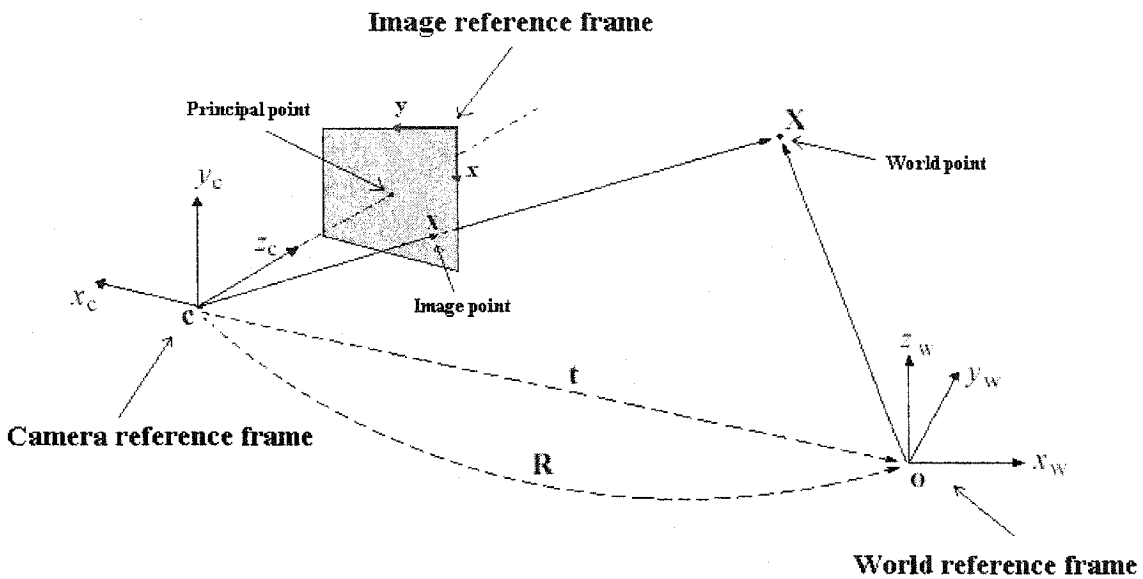


Figure 2.4: The extrinsic camera parameters $\mathbf{R}$ and $\mathbf{t}$ model the rigid-body transformation between the world reference frame centered at O and the camera reference frame centered at C.

the *extrinsic* camera parameters, it describes the rigid-body transformation between the camera and the world reference frame as shown in Figure 2.4.

The camera calibration matrix $\mathbf{K}$ models the relation between a point $\mathbf{x}$ in the image coordinate system and the direction of the ray defined by $\mathbf{x}$ and the camera center C. The matrix $\mathbf{K}$ has the following structure [32]:

$$\mathbf{K} = \begin{bmatrix} \xi f & s & x_0 \\ 0 & f & y_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.2.3}$$

where f is the focal length of the camera, ξ is the aspect ratio, s is the skew (non-orthogonality) between the axes, and (x_0, y_0) are the coordinates of the principal

point. Since f, ξ, s, x_0 and y_0 do not depend on the position or orientation of the camera, they are called the *intrinsic* parameters. If both the *intrinsic* and *extrinsic* parameters are known, then the camera is said to be fully calibrated.

2.3 Epipolar geometry

The projective geometry between two cameras is defined as the epipolar geometry [32]. It only depends on the relative pose and intrinsic parameters of the cameras. The epipolar geometry between two cameras of centers C_1 and C_2 is shown in Figure 2.5. The line joining C_1 and C_2 is called the *baseline*. The intersection of the baseline with the two image planes are the epipoles $\mathbf{e}_1$ and $\mathbf{e}_2$. The epipole $\mathbf{e}_1$ is the image of C_2 on the first camera and $\mathbf{e}_2$ is the image of C_1 on the second camera. The *epipolar plane* is defined as the plane containing C_1, C_2 and the 3D point $\mathbf{X}$. The epipolar line is the intersection of the epipolar plane with the two image planes. The epipolar line $\mathbf{l}_1$ in the first image is the image of the line through C_2, $\mathbf{x}_2$ and $\mathbf{X}$. Similarly, the epipolar line $\mathbf{l}_2$ in the second image is the image of the line C_1, $\mathbf{x}_1$ and $\mathbf{X}$. The correspondence of a point on one image must lie on the epipolar line on the other image, this is known as the *epipolar constraint.*

2.3.1 Essential and fundamental matrices

The essential matrix $\mathbf{E}$, introduced by Longuet-Higgins [49] , describes the geometric relationship between corresponding points of a pair of calibrated cameras with centers C_1 and C_2. Consider two cameras with the 3D world point $\mathbf{X}$ projected onto the image planes at $\bar{\mathbf{x}}_1$ and $\bar{\mathbf{x}}_2$ as shown in Figure 2.6. Assuming that the cameras are calibrated then $\bar{\mathbf{x}}_1$ and $\bar{\mathbf{x}}_2$ are represented with respect to their corresponding camera coordinate

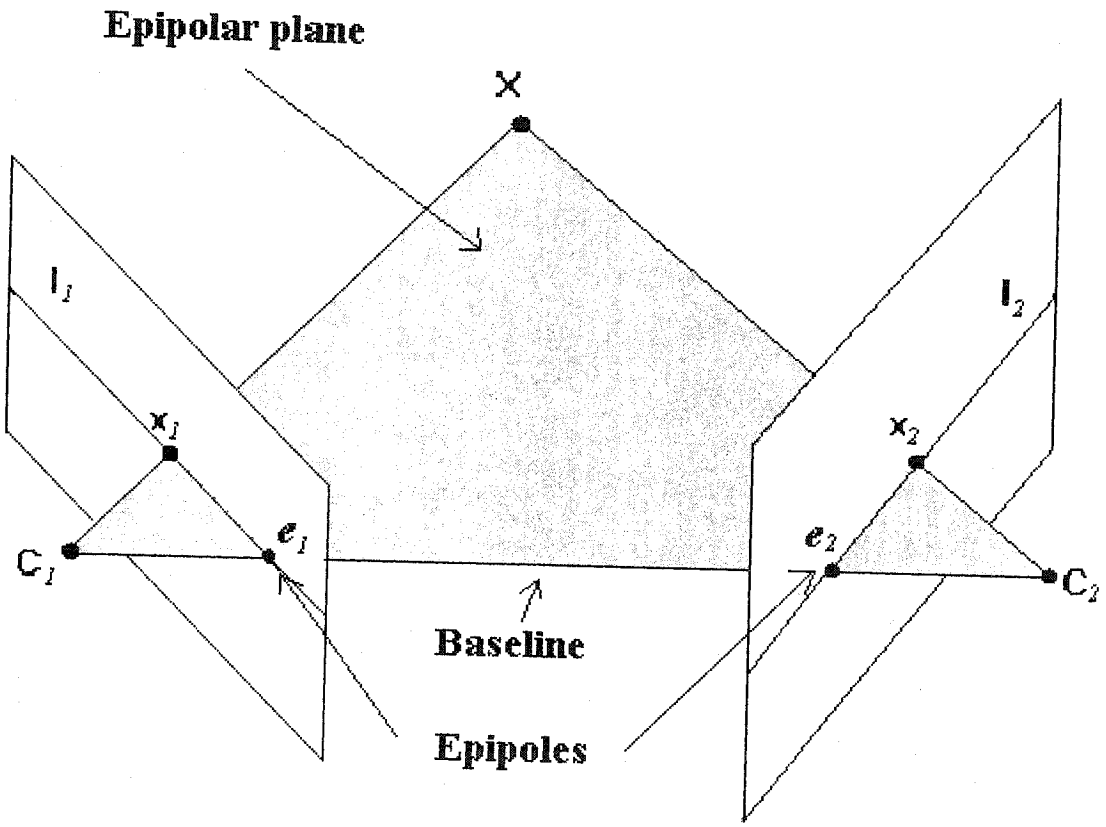


Figure 2.5: Epipolar geometry: The *baseline* is the line that joins the camera centers C_1 and C_2. The baseline intersects the image planes at the epipoles $\mathbf{e}_1$ and $\mathbf{e}_2$. The plane that contains the baseline is called the epipolar plane which intersects the two image planes at the lines $\mathbf{l}_1$ and $\mathbf{l}_2$.

system (*normalized coordinates*). The essential matrix can be derived by applying the epipolar constraint [105]. The vectors $\overrightarrow{C_1\bar{\mathbf{x}}_1}$, $\overrightarrow{C_2\bar{\mathbf{x}}_2}$ and $\overrightarrow{C_1C_2}$ are coplanar, this constraint can be expressed as follows:

$$\bar{\mathbf{x}}_2^T (\mathbf{t} \times \mathbf{R}\bar{\mathbf{x}}_1) = 0 \tag{2.3.1}$$

where $\mathbf{R}$ and $\mathbf{t}$ are the translation and rotation between the two camera coordinate systems. Equation(2.3.1) can be expressed as follows:

$$\bar{\mathbf{x}}_2^T ([\mathbf{t}]_\times \mathbf{R}\bar{\mathbf{x}}_1) = 0 \tag{2.3.2}$$

The *essential matrix* $\mathbf{E} = [\mathbf{t}]_\times \mathbf{R}$ and Equation (2.3.2) can be expressed as follows:

$$\bar{\mathbf{x}}_2^T \mathbf{E}\bar{\mathbf{x}}_1 = 0 \tag{2.3.3}$$

The matrix $\mathbf{E}$ only depends upon the rotation $\mathbf{R}$ and the translation $\mathbf{t}$ between the two cameras and is defined up to a scaling factor, thus $\mathbf{E}$ has five degrees of freedom.

The fundamental matrix $\mathbf{F}$ can be derived from Equation(2.3.3) by using the calibration matrices. Let $\mathbf{K}_1$ and $\mathbf{K}_2$ be the calibration matrices of cameras C_1 and C_2 respectively, the camera points are related to the image points by substituting $\bar{\mathbf{x}}_1 = \mathbf{K}_1^{-1}\mathbf{x}_1$ and $\bar{\mathbf{x}}_2 = \mathbf{K}_2^{-1}\mathbf{x}_2$ in Equation(2.3.3) as follows:

$$\mathbf{x}_2^T \mathbf{K}_2^{-T} \mathbf{E} \mathbf{K}_1^{-1} \mathbf{x}_1 = 0 \tag{2.3.4}$$

The fundamental matrix is defined to be:

$$\mathbf{x}_2^T \mathbf{F} \mathbf{x}_1 = 0 \tag{2.3.5}$$

where $\mathbf{F} = \mathbf{K}_2^{-T}\mathbf{E}\mathbf{K}_1^{-1}$.

Figure 2.7 illustrates the epipolar geometry for a scene. Figure 2.7(b) shows the epipolar lines for the points selected in Figure 2.7(a).

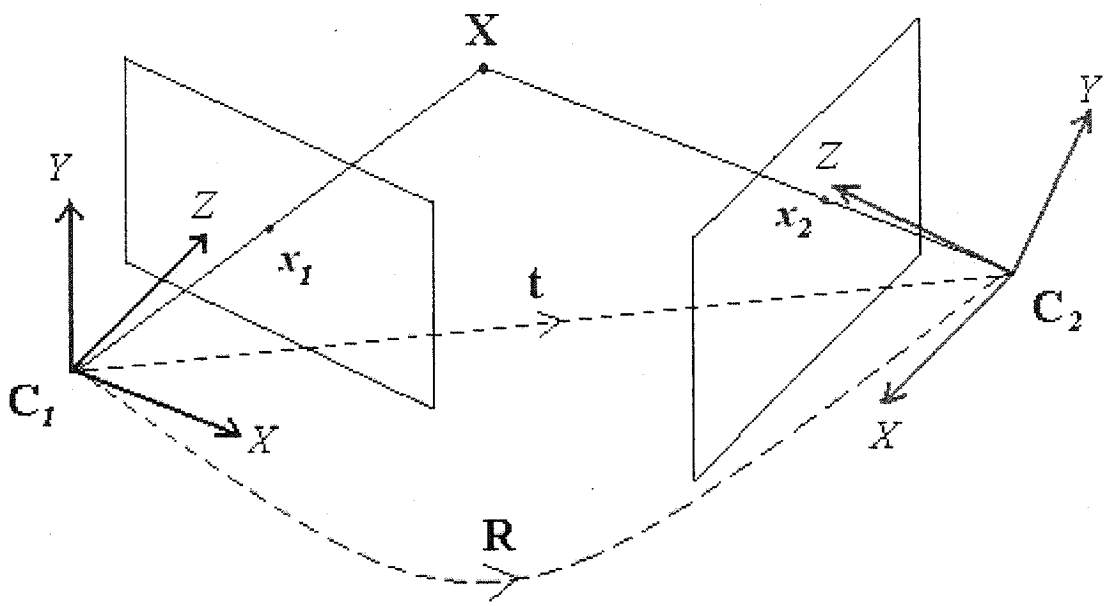
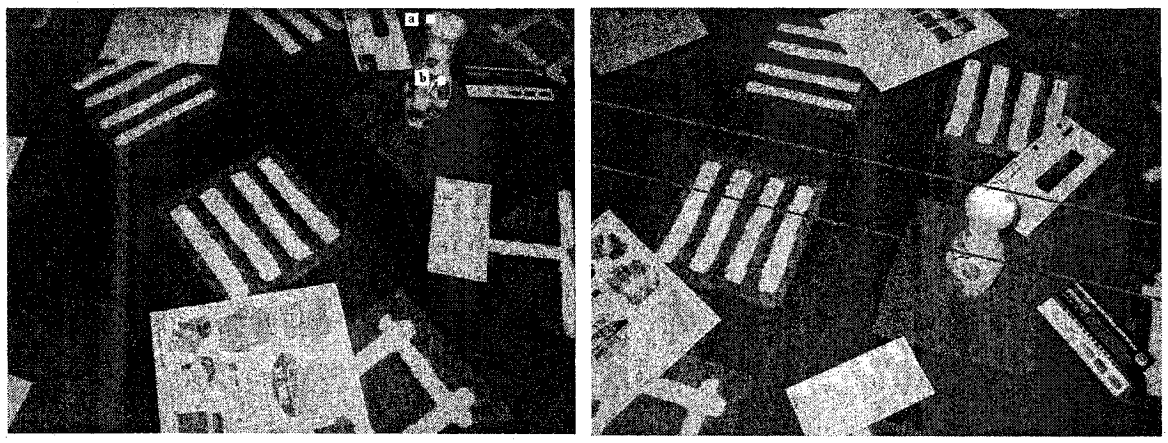


Figure 2.6: Epipolar geometry.

2.4 Planar homography

When the perspective camera images a world planar surface, a plane-to-plane projection occurs between the world plane and its image captured by the camera. The points on the world plane in the scene are mapped to points on the plane in the image by a *plane-to-plane homography*. This homography can be described by a 3×3 matrix $\mathbf{H}_w$. Figure 2.8 shows a plane-to-plane homography between a point $\mathbf{X}$ on the world X-Y plane Π and the image point $\mathbf{x}$ on the image plane π. The camera center is at point C and x-y and X-Y are the image plane and world plane coordinate systems respectively. The perspective projection relation between point $\mathbf{X} = [X, Y, 1]^T$ and $\mathbf{x} = [x, y, 1]^T$ is as follows [63]:



(a) (b)

Figure 2.7: (a) Two selected points on the first image (b) Corresponding epipolar lines.

$$\mathbf{x} = \mathbf{H}_w \mathbf{X} \tag{2.4.1}$$

The homography transformation $\mathbf{H}_w$ is invertible, the world points can be recovered from the image points as follows: $\mathbf{X} = \mathbf{H}_w^{-1}\mathbf{x}$. The matrix $\mathbf{H}_w$ can be computed if the camera is calibrated and the orientation of the world plane is known. Also, it can be calculated from at least four world-to-image corresponding points using SVD decomposition. Figure 2.9(a) shows the image of a world planar surface (the wall of the building). The points a, b, c and d are used to calculate the world-to-image homography. The calculated homography can be used to perform a *fronto-parallel*[1] transformation, the synthesized image is shown in Figure 2.9(b).

[1]parallel lines on the world plane are parallel on the image plane

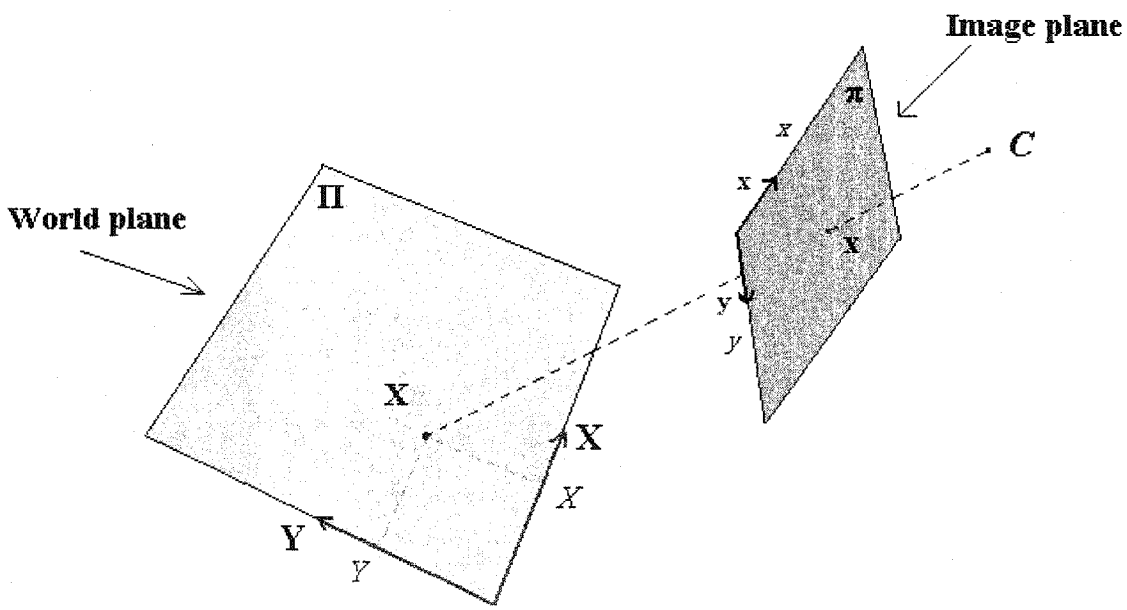
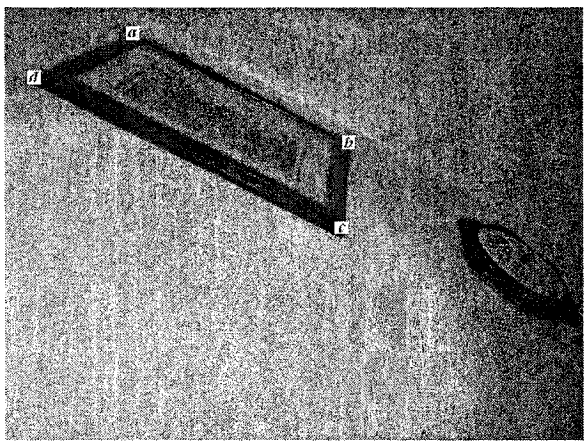
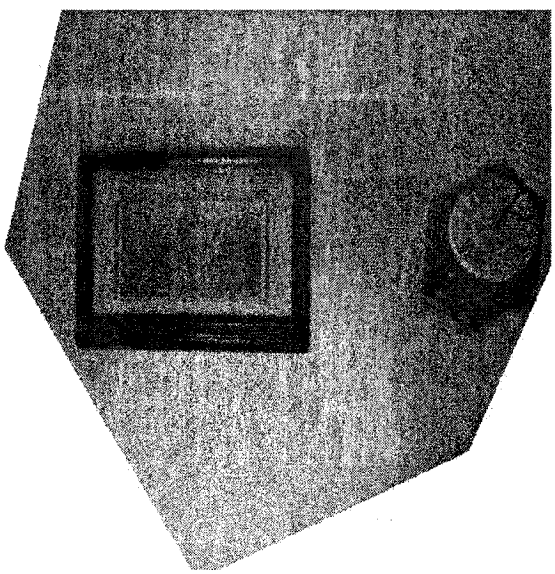


Figure 2.8: Plane-to-plane homography: a point $\mathbf{X}$ in the world `X-Y` plane $\mathbf{\Pi}$ is mapped to the image `x-y` plane π at a point $\mathbf{x}$.



(a)



(b)

Figure 2.9: *Fronto-parallel* transformation: (a) original image of a painting, the four corners are used to calculate the image-to-world homography transformation (b) the rectified image, parallel lines on the world plane are parallel in the image plane.

2.5 Inter-image homography

The inter-image homography is a transformation between two images induced by a plane in the scene. The relation between a point $\mathbf{x}_1$ in the first image and a point $\mathbf{x}_2$ in the second image is as follows:

$$\mathbf{x}_2 = \mathbf{H}\mathbf{x}_1 \tag{2.5.1}$$

Figure 2.10 (a) and (b) shows two images of a planar scene, the inter-image homography $\mathbf{H}$ can be calculated using at least four corresponding points. Figure 2.11(a) shows the first image warped using $\mathbf{H}$ onto the second image, Figure 2.11(b) shows the second image warped using $\mathbf{H}^{-1}$ onto the first image. Points off the ground (the cup) are not mapped correctly due to the motion parallax effect [23].

For a two-camera system C_i and C_j, Tsai and Huang [92] showed that the planar homography can be decomposed with C_i selected as a reference frame as follows:

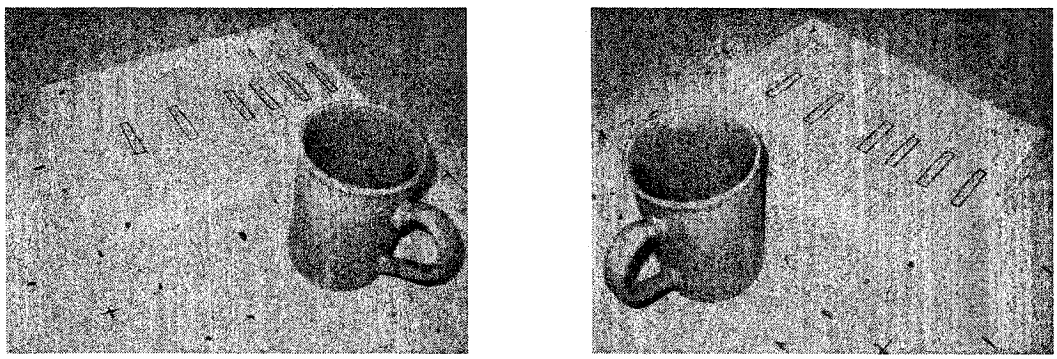
$$\mathbf{H}_{ij} = \mathbf{K}_j \left[\mathbf{R} - \frac{\mathbf{t}\mathbf{n}^T}{d}\right] \mathbf{K}_i^{-1} \tag{2.5.2}$$

where $\mathbf{K}_i$ and $\mathbf{K}_j$ are the matrices containing the intrinsic parameters of cameras C_i and C_j respectively, $\mathbf{R}$ is the rotation between the two cameras, $\mathbf{n}$ is the normal to the plane under consideration and $\mathbf{t}$ is the translation. Finally d is the distance from the camera to the plane.

2.6 Calibration from planes

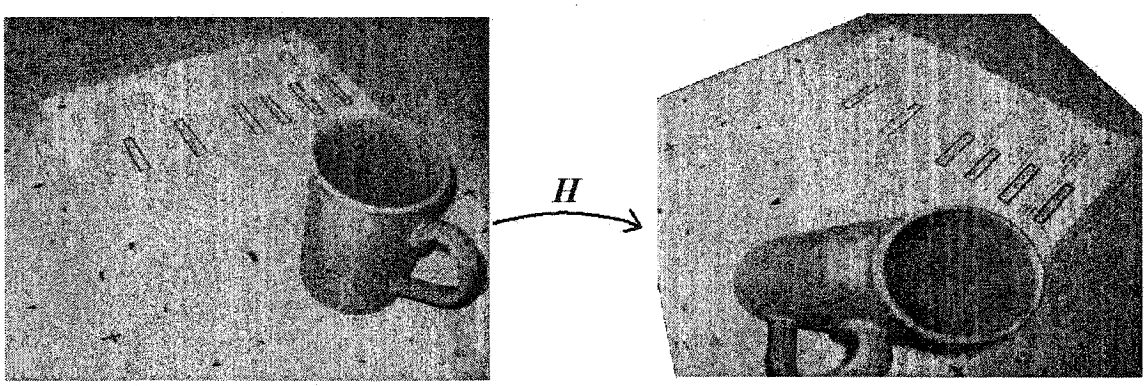
Many authors have proposed various approaches to estimate the camera motion using the homography matrix induced by a planar surface in the scene.

Xu et al. [100] introduced a linear algorithm to calculate the motion between two cameras. For two camera system C_1 and C_2 with unknown focal length f_1 and f_2

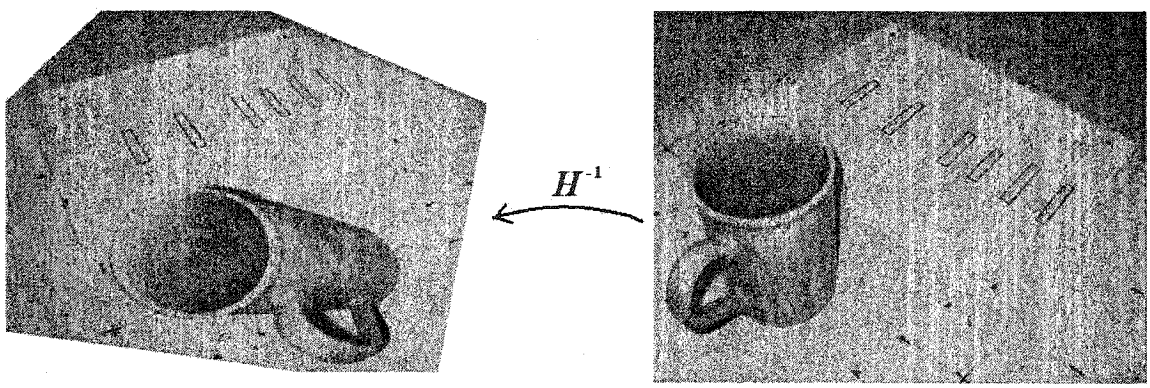


(a) (b)

Figure 2.10: Images of a cup: Two images of a cup with a planar surface. The planar surface induces a homography between the two images.



(a)



(b)

Figure 2.11: Image warp:(a) The image in Figure 2.10(a) warped onto Figure 2.10(b). (b) The image in Figure 2.10(b) warped onto Figure 2.10(a).

respectively, Equation(2.5.2) can be written as:

$$s\mathbf{H}_{12} = \mathbf{K}_2\mathbf{R}\mathbf{K}_1^{-1} - \mathbf{K}_2\mathbf{t}\frac{\mathbf{n}^T}{d}\mathbf{K}_1^{-1} \tag{2.6.1}$$

The term $\mathbf{K}_2\mathbf{t} = a\mathbf{e}_2$ where a is a scale such that $\|\mathbf{e}_2\| = 1$ and s is an unknown scalar factor. Provided that there are two planes in the scene with known homography matrices $\mathbf{H}_{12}^1$ and $\mathbf{H}_{12}^2$, the epipoles $\mathbf{e}_2$ and $\mathbf{e}_1$ are related as follows:

$$\mathbf{e}_2 = \mathbf{H}_{12}^1\mathbf{e}_1 \tag{2.6.2}$$

$$\mathbf{e}_2 = \mathbf{H}_{12}^2\mathbf{e}_1 \tag{2.6.3}$$

From Equations(2.6.2) and (2.6.3) the following equation can be written:

$$\gamma[\mathbf{H}_{12}^1]^{-1}\mathbf{e}_2 = [\mathbf{H}_{12}^2]^{-1}\mathbf{e}_2 \tag{2.6.4}$$

where γ is an unknown scalar. The epipole $\mathbf{e}_2$ can be obtained from Equation(2.6.4) as the generalized eigenvector associated with the third eigenvalues [37]. Substituting the value of $\mathbf{e}_2$ for the term $\mathbf{K}_2\mathbf{t}$ and doing some algebraic manipulation, Equation(2.6.1) can be written as :

$$s^2\mathbf{H}_{12}\mathbf{K}_1\mathbf{K}_1^T\mathbf{H}_{12}^T + m\mathbf{e}_2\mathbf{e}_2^T - s\left(\mathbf{H}_{12}\mathbf{K}_1\bar{\mathbf{n}}\mathbf{e}_2^T + \mathbf{e}_2\bar{\mathbf{n}}^T\mathbf{K}_1^T\mathbf{H}_{12}^T\right) = \mathbf{K}_2\mathbf{K}_2^T \tag{2.6.5}$$

where $\bar{\mathbf{n}} = a\mathbf{n}/d = [n_1, n_2, n_3]^T$ and $m = \bar{\mathbf{n}}^T\bar{\mathbf{n}}$.

Equation(2.6.5) is linear with respect to the following vector:

$$\mathbf{V} = \left[s^2f_2^2, s^2, f_1^2, m, sf_2n_1, sf_2n_2, sn_3\right]^T \tag{2.6.6}$$

The vector $\mathbf{V}$ is calculated using SVD and the estimated translation and rotation can be expressed as follows:

$$\mathbf{t} = a\mathbf{K}_2^{-1}\mathbf{e}_2 \tag{2.6.7}$$

$$\mathbf{R} = s\mathbf{K}_2^{-1}\mathbf{H}_{12}\mathbf{K}_1 - \mathbf{K}_2^{-1}\mathbf{e}_2\bar{\mathbf{n}}^T \tag{2.6.8}$$

Zhang [106] and Sturm-Maybank [82] independently introduced a technique for calibrating a camera using a planar pattern with known metric information in the scene (example checker pattern). The homography $\mathbf{H}$ between the pattern and the image plane can be expressed as follows:

$$\mathbf{H} = [\mathbf{h_1}\ \mathbf{h_2}\ \mathbf{h_3}] = \lambda\mathbf{K}[\mathbf{r_1}\ \mathbf{r_2}\ \mathbf{t}] \tag{2.6.9}$$

The vectors $\mathbf{r}_1$ and $\mathbf{r}_2$ are orthonormal, then $\mathbf{r}_1^T.\mathbf{r}_2 = 0$ and $\|\mathbf{r}_1\| = \|\mathbf{r}_2\|$. This can be expressed in terms of $\mathbf{h}_1$ and $\mathbf{h}_2$ as follows:

$$\mathbf{h}_1^T w_\infty \mathbf{h}_2 = 0 \tag{2.6.10}$$

$$\mathbf{h}_1^T w_\infty \mathbf{h}_1 = \mathbf{h}_2^T w_\infty \mathbf{h}_2 \tag{2.6.11}$$

where $w_\infty = \mathbf{K}^{-T}\mathbf{K}^{-1}$ is the image of the absolute conic. Three views of the plane are needed to solve for w_∞. The intrinsic matrix $\mathbf{K}$ can be obtained from w_∞ using Cholesky factorization [32].

After calculating $\mathbf{K}$, the rotation matrix can be obtained as follows:

$$\begin{aligned} \mathbf{r}_1 &= \lambda \mathbf{K}^{-1}\mathbf{h}_1 \\ \mathbf{r}_2 &= \lambda \mathbf{K}^{-1}\mathbf{h}_2 \\ \mathbf{r}_3 &= \mathbf{r}_1 \times \mathbf{r}_2 \\ \mathbf{t} &= \lambda \mathbf{K}^{-1}\mathbf{h}_3 \end{aligned} \tag{2.6.12}$$

where $\lambda = \frac{1}{\|\mathbf{K}^{-1}\mathbf{h}_1\|}$.

Matsunaga and Kanatani [58] presented a calibration algorithm by placing a distinguishable planar pattern in the scene with known metric information (example checker board). First an XYZ world coordinate system is set with the XY plane parallel to the planar pattern at a distance d. A hypothetical camera of focal length f_0 is set at the origin O of the world coordinate system. The image plane of the hypothetical camera is set parallel to the XY. The points on the planar surface are of known coordinates and a homography can be calculated between the points on the planar surface and their image on the hypothetical camera as follows:

$$\mathbf{x}_i = \mathbf{H}_{wi}\mathbf{X}_w \tag{2.6.13}$$

where $\mathbf{x}_i = [x_i, y_i, f_0]$ is the point on the hypothetical camera and $\mathbf{X}_w = [X_i, Y_i, d]$ is the point on the planar surface and $\mathbf{H}_{wi}$ is a homography that maps a point on the planar pattern to the image plane of the hypothetical camera. The motion of the real camera can be obtained by performing a rotation $\mathbf{R}$ and a translation $\mathbf{t}$ and then changing the focal length to f. In this case the homography matrix can be described

as follows:

$$\mathbf{H}_{wi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \frac{f_0}{f} \end{bmatrix} \mathbf{R}^T \left[\mathrm{I} - \frac{\mathbf{t}\mathbf{n}^T}{d} \right] \tag{2.6.14}$$

where $\mathbf{n} = [0, 0, 1]^T$ is normal to the planar surface. The homography matrix is first calculated by a technique presented in [40]. An initial estimate of the camera motion is then estimated by decomposing the homography matrix in Equation (2.6.14) into f/f_0, $\mathbf{t}/d$ and $\mathbf{R}$ by an analytical procedure described in [57]. The camera motion is finally calculated by using bundle-adjustment approach.

Vigueras et al. [93] introduced a model selection algorithm to calculate the camera motion among consecutive images for an *Augmented Reality* application. The algorithm is based on the knowledge of the equations of a piecewise planar surfaces in the scene provided by the user. In the preprocessing phase, the projection matrix of the first camera is estimated by using a poster in the scene. After the preprocessing phase, feature points for each plane are extracted and matched from frame to frame and the homography transformations are calculated. Then, for each plane, the projection matrix of each frame is computed using the constraint induced by the homography transformations. The camera motion is estimated using the projection matrix corresponding to the best model.

Triggs [90] developed a self-calibration algorithm from planar views without knowing their metric structure. The algorithm is based on the constraint induced by the absolute quadratic and the plane to image homography. Since no metric information of the scene plane is known, a view of the scene is chosen as a reference. The inter-image homography transformations are first calculated with respect to this view

and then used in the calibration process. The main idea of the algorithm is to map the circular points from image to image using the inter-image homography matrices, then the calibration matrix is calculated. The calibration equation that describes such constraints for m views of a planar scene is as follows:

$$(\mathbf{H}_{i1}\mathbf{x}_{\pm})^T w_{\infty}^{-1}(\mathbf{H}_{i1}\mathbf{x}_{\pm}) = 0,\ i = 1, \ldots, m \tag{2.6.15}$$

where $\mathbf{H}_{i1}$ is the inter-image homography between view i and the reference view (say view 1). The two points $\mathbf{x}_{\pm}$ are the images of the circular points. The circular points are the intersection of the absolute conic with the plane.
The orientation of the views relative to the reference view is recovered from the SVD of the inter-image homography matrices $\mathbf{H}_{i1}$.

Malis and Cipolla [54] introduced an algorithm for self-calibration of cameras with different focal length observing a planar scene. The inter-image homography induced by the planar scene is used in the algorithm and the metric information of the scene are not needed. The inter-image homography transformations introduce a constraint that is used in the algorithm. If the camera matrices $\mathbf{K}_i$ and $\mathbf{K}_j$ are known, then the Euclidean homography $\mathbf{HE}_{ij}$ can be recovered from $\mathbf{H}_{ij}$ in Equation(2.5.2) to be:

$$\mathbf{HE}_{ij} = \mathbf{R} - \frac{\mathbf{t}\mathbf{n}_j^T}{d} \tag{2.6.16}$$

where $\mathbf{n}_j$ is expressed with respect to camera C_j. From Equation (2.6.16) a constraint can be satisfied $\forall k > 0$ as follows:

$$\begin{aligned} [\mathbf{n}_i]_{\times}^k \mathbf{HE}_{ji}^T &= \mathbf{HE}_{ij}[\mathbf{n}_j]_{\times}^k \ , \ \forall k > 0 \\ [\mathbf{n}_i]_{\times} &= \mathbf{HE}_{ij}[n_j]_{\times}\mathbf{HE}_{ij}^T \\ \mathbf{n}_i &= \mathbf{Q}_{ij}\mathbf{n}_j \end{aligned} \tag{2.6.17}$$

where $\mathbf{n}_i$ is expressed with respect to camera C_i and $\mathbf{Q}_{ij} = det(\mathbf{HE}_{ij})\mathbf{HE}_{ij}^{-T}$.
The constraints in Equation (2.6.17) are extended to multiple view geometry and the homography matrices are decomposed to recover the motion. The details of the decomposition algorithm are explained in [53].

2.7 Summary

This chapter presents an overview of some concepts and theories that are essential for the rest of the thesis. In this chapter, the pin-hole camera model is first introduced then the epipolar geometry is explained followed by the derivation of the essential matrix $\mathbf{E}$ and the fundamental matrix $\mathbf{F}$. The plane-to-plane and the inter-image homography transformations are then presented and finally this chapter provides an overview of calibration from planar surfaces in the scene.

In the pin-hole camera, the relation between a point in the 3D world and its image on the image plane can be modeled by a 3×4 matrix $\mathbf{P}$ called the projection matrix. Matrix $\mathbf{P}$ can be decomposed into three components, the intrinsic parameters of the camera, the rotation and the translation with respect to the world plane.

The epipolar geometry induces a constraint for the corresponding points between images. A point on one image must have its corresponding point on the other image lie on a line called the epipolar line. This relation is represented by the essential matrix $\mathbf{E}$ in case of calibrated cameras and by the fundamental matrix $\mathbf{F}$ in case of un-calibrated cameras.

The plane-to-plane homography represents a transformation between a plane in the world coordinate system with its image on the image plane. When two views are

taken for a scene with a planar surface, this surface induces a homography transformation between the two views called the inter-image homography. Both the plane-to-plane and the inter-image homography matrices are represented by a 3×3 matrix and can be calculated by at least four points.

Knowledge of planes in the scene can be instrumental in estimating the camera intrinsic and extrinsic parameters. When the metric information of the plane is known, it is possible to estimated the motion of the camera with respect to the plane by analyzing the constraint induced by the plane-to-plane homography and decomposing it to recover the motion parameters. When no metric information is known about the plane, the inter-image homography is used instead. It is then decomposed to recover the motion parameters with respect to one selected view.

Chapter 3

Ground plane matching

In this chapter an algorithm to match sparse view images taken by a mobile robot is introduced. The robot is equipped with a single camera and moves on a planar surface in an unknown environment. The algorithm proposed here proceeds by first matching feature points on widely separated images of the work area using an overhead view transformation. The inter-image homography transformation induced by the ground plane thus obtained is used to estimate the camera motion parameters.

3.1 Ground plane detection

Ground plane detection is an essential problem for robots moving on a planar surface. In order to navigate, the robot must be able to detect portions of the environment that are possible to traverse. This implies that it must have the ability to identify the ground plane in the work area. The environment on which these navigation systems operate can generally be modeled as a planar surface on which the obstacles reside.

In the literature most techniques are based on detecting feature points from the observed scene and then impose some constraints to classify the feature points to either lie on the ground plane or above/below the ground plane (the obstacles).

The conventional solution to the ground plane detection problem is to use a stereovision system. Based on the constant geometric constraints that exist between two cameras, points on both images are matched in order to obtain a disparity map from which depth information can be computed. Beside the fact that this approach requires more complex hardware, it also has the drawback of imposing the execution of some calibration phase. While the system is in operation, validations must be regularly performed to ensure that the calibration information has not been ruined by an unfortunate displacement of the stereo rig. Registration between the different views is also a challenging problem.

In a calibrated stereo system, image points of the ground plane can be transferred from one image to another. This can be done by virtue of the planar homography relation that exists between two views of a plane. The computed disparity vectors can therefore be used to obtain a residual disparity giving information about the amount of deviation of a point with respect to the ground plane [107]. This is the motion parallax effect that has been exploited by many authors in obstacle detection. Lourakis et al. [50], for example, introduced an algorithm to detect obstacles along the path of an autonomous robot using two images. Based on a normalized cross-correlation approach, such as [103], the planar homography between two views is estimated from detected corner features. The matched corners that do not map to this ground plane homography are said to belongs to an obstacle. This is the strategy that has been used by Chow and Chung [10] and Ferrari et al. [17] in their real-time obstacle detection system. A similar idea has been used by Williamson and Thrope [97] but where they used a set of homographies to perform calibration, matching and obstacle detection. Jenkin and Jepson [36] used the disparity field

to build an ownership probability model that computes the probability that a given disparity belong to the floor or to an obstacle. To avoid the matching process, Xie [99] proposed to simply use edge primitives and from them to segment the projected image difference in order to locate regions containing residual disparity. Hatari and Maki [33] simplified the ground plane relation by using an affine representation through the introduction of a pseudo-perspective model. The resulting simpler transformation is then estimated from the vanishing point of two parallel lines.

When only one camera is used, information about camera motion relative to its environment and about the 3D structure of the imaged scene can be obtained by estimating the optical flow field of the image sequence resulting from the robot motions. Carlsson and Eklundh [6] presented a model-based prediction approach to detect the ground plane surface. The image transformation for motion relative to the planar surface is first computed using a recorded sequence of images. This transformation is then used to predict the motion of each pixel in a temporal image sequence. The image point that is correctly predicted is considered to belong to the planar ground and the ones that are erroneously predicted are considered obstacle points. A similar approach has been used by Enkelmann [14]. First velocity information from robot sensors are integrated to calculate a model optical flow vector field for motion parallel to the ground surface. Then the model flow is compared to the flow from a temporal image sequence. The image points that deviate from the model flow are considered obstacle points. Fornland [20] presented a technique based on spatio-temporal derivatives. The camera motion parameters are related to the first order spatio-temporal derivatives of an image sequence. The ground plane induces constraint on the statistical distribution (Gaussian) of the orthogonal signed distance of the corresponding

image points. Points that view obstacle points have different statistics and are detected as outliers using the RANSAC [18] method. Willersinn and Enkelmann [96] also used spatio-temporal to estimate the optical flow vectors. These vectors are then clustered into groups of similar flow vectors. Kalman filter is used to track the flow vectors over consecutive frames and the obstacle detection problem is reduced to state estimation problem. Santos-Victor and Sandini [73] proposed an algorithm to detect obstacles by projecting the optical flow vectors onto a horizontal plane parallel to the ground. This simplifies the resulting flow pattern due to the perspective distortion. The points that belong to the ground surface have equal vectors, thus the points that belong to the obstacles can be detected easily.

In this chapter, an algorithm for detecting the ground plane and the camera motion estimation is presented in the context of a robot equipped with a single camera. A collection of sparse views of the scene with no restriction on the baseline width is assumed to be available. These views could have been obtained by having the robot patrolling along the periphery of the site or by a team of robots collaborating together to obtain a representation of the environment. This is a very challenging situation since it requires the registration of images taken from very different points of view. But this approach offers the advantage of considerably limiting the amount of data to be processed and/or to be transmitted. Since many real-world robotic applications have to cope with limited bandwidth issues, this asset can be key. In order to proceed to the motion estimation, an approximate knowledge of the camera orientation with respect to the ground plane is assumed. This measurement does not have to be accurate as the algorithm presented here includes a self-calibration phase.

3.2 Overhead view transformation

The overhead view of a plane can be obtained by applying the appropriate transformation. Many authors have used overhead views in different applications such as augmented reality and video surveillance. Bradshaw et al. [3], use known ground plane features, such as lines, to transform planar motion into an overhead view and then recover the trajectory of moving objects on the plane. Intille and Bobick [35] use overhead view to track football players on the coordinate system of the field rather than the image coordinates. The overhead view is calculated using known measurement of the lines on the football field. Lee et al. [47] use multiple cameras to obtain an overhead view without depending on known land marks or manual registration. Instead, they detect objects moving simultaneously in cameras with overlapping views. The object centroids are used as potential point correspondences to recover the homographies between camera pairs. Laganière [44] uses overhead view to build a bird's eye view of a worksite. The overhead view is calculated assuming known intrinsic camera parameters and known orientation with respect to the observed plane. In our work, the overhead views are used to match the two images and get the homography. The matching is done without depending on known patterns, moving objects on the plane or manual registration, instead the algorithm is based on features that have to be matched between views. However when a plane is observed by a tilted camera the features lying on it are subject to important perspective deformations. In the context of sparse views, this makes the features difficult to match using the usual correlation schemes. Applying the overhead view transformation will undo the perspective deformation which will make the apparent features differing only by a similarity transformation.

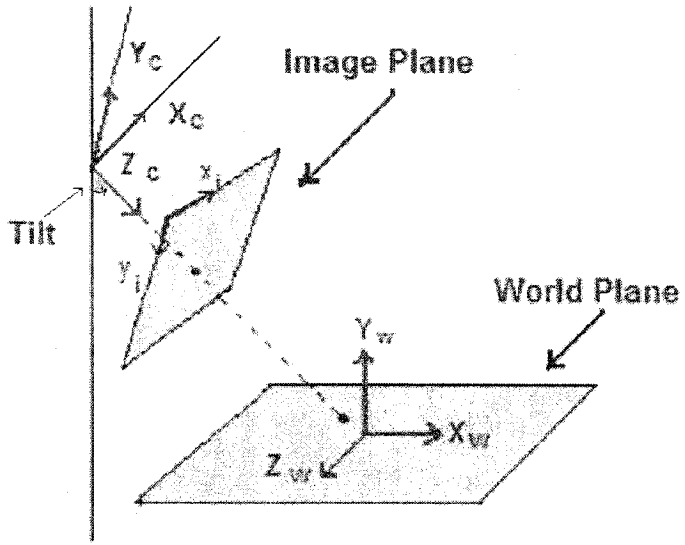


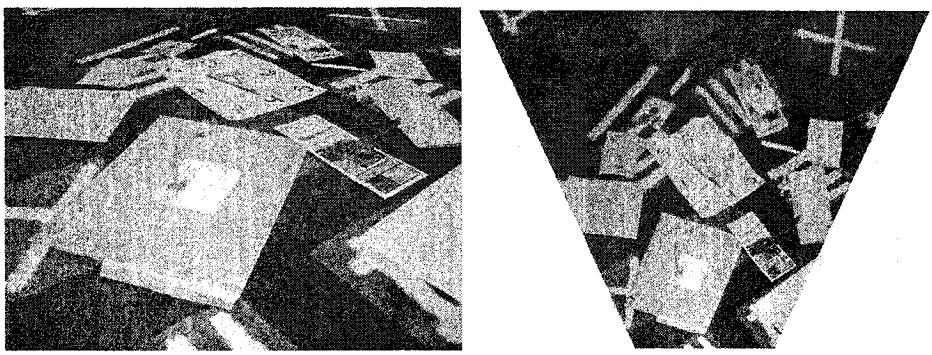
Figure 3.1: Geometry of the system.

Figure 3.1 shows the structure of the coordinate systems of the world, camera and image coordinate systems. The ground plane surface is at $\mathtt{Y} = 0$, the optical axis of the camera is aligned with the `Z` axis and the camera is rotated around the `X` axis by an angle α. The camera height from the world plane is h.

The projective relation between the world plane and the corresponding overhead image point can be represented as follows:

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \mathbf{H}_O \begin{bmatrix} X_W \\ Z_W \\ 1 \end{bmatrix} \tag{3.2.1}$$

When the geometry of the system is as shown in Figure 3.1, the 3×3 homography



(a) (b)

Figure 3.2: Overhead view: (a) The image of a planar surface taken with tilt = 33^o, (b) The corresponding generated overhead view.

matrix $\mathbf{H}_O$ can be described as follows [44]:

$$\mathbf{H}_O = \begin{bmatrix} f & x_0\cos(\alpha) & x_0 h\cos(\alpha) \\ 0 & f\sin(\alpha) + y_0\cos(\alpha) & y_0 h\cos(\alpha) - fh\sin(\alpha) \\ 0 & \cos(\alpha) & h\cos(\alpha) \end{bmatrix} \tag{3.2.2}$$

where f represents the focal length, x_0 and y_0 are coordinates of the principal point, α is the tilt angle and h is the height of the camera. This transformation is inversible such that the overhead view can be generated from a perspective or vice versa. Figure 3.2(a) shows an image taken with a tilt = 33^o, the corresponding overhead view image is shown in Figure 3.2(b).

3.2.1 Overhead view mosaic

In this section the composition of overhead view mosaics is discussed. The results of the algorithm presented here are used throughout the following chapters for map building and 3D obstacle reconstruction. The method is based on estimating the homography transformation of the set of images with respect to a certain plane in the scene (for example the ground plane). The overhead view mosaic is finally built

by rectifying the images using an overhead transformation with respect to a reference image. One of the images (say image 0) is selected as a reference frame. The homographic transformation between the reference view 0 and a view i is:

$$\mathbf{x}_i = \mathbf{H}_{0i}\mathbf{x}_0 \tag{3.2.3}$$

For known camera parameters, the image to plane homography $\mathbf{H}_O$ between the reference view (view 0) and the ground plane can be calculated as described in Equation(3.2.2). The two matrices $\mathbf{H}_{0i}$ and $\mathbf{H}_O$ can be used to calculate the homography transformation between a view i and the composted overhead view as follows:

$$\mathbf{H}_{iO} = \mathbf{H}_O^{-1}\mathbf{H}_{0i}^{-1} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \tag{3.2.4}$$

Armed with all these homographic transformations, it is possible to produce an overhead mosaic representing the environment under study. When combining the information coming from different images, two strategies can be considered. The first one consists in determining the value of each pixel in the mosaic by selecting the pixel from the source image that has the best resolution [44]. For a world point $[X, Y, 1]^T$ the mosaic pixel is selected from the source image that best samples the segment between points $[X, Y, 1]^T$ and $[X + \Delta_X, Y + \Delta_Y, 1]^T$ where Δ_X and Δ_Y are small increments in the `X` and `Y` directions respectively. The relation between a point $[X, Y, 1]^T$ in world plane and a point $[x, y, 1]^T$ in the image plane is:

$$X = h_X(x, y) = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}} \tag{3.2.5}$$

$$Y = h_Y(x, y) = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}} \tag{3.2.6}$$

The image resolution is determined by a measure called *instantaneous sampling rate.* For a world point $[X, Y, 1]^T$ the sampling rate at the X direction is the number of source images between points $[X, Y, 1]^T$ and $[X + \Delta_X, Y, 1]^T$ divided by the distance between them. The instantaneous horizontal sampling rate is as follows:

$$s_X = \sqrt{\left(\frac{\delta h_X(x,y)}{\delta x}\right)^2 + \left(\frac{\delta h_X(x,y)}{\delta y}\right)^2} \tag{3.2.7}$$

Similarly the instantaneous vertical sampling rate is:

$$s_Y = \sqrt{\left(\frac{\delta h_Y(x,y)}{\delta x}\right)^2 + \left(\frac{\delta h_Y(x,y)}{\delta y}\right)^2} \tag{3.2.8}$$

The instantaneous sampling rate is defined as follows:

$$s = s_X s_Y \tag{3.2.9}$$

The mosaic pixel at point (x, y) is selected from the source image that have the highest sampling density s.

Figure 3.3 shows a set of images of a ground plane scene; the reference image is Figure 3.3(a). Figure 3.4(a) shows the mosaic obtained for this set of images. As it can be seen, when obstacles are visible in the source image, these ones interfere with the ground plane representation. In order to cope with this problem, we have to discard the source pixels that contains intensities that belong to the obstacles. To do so, we propose the following algorithm:

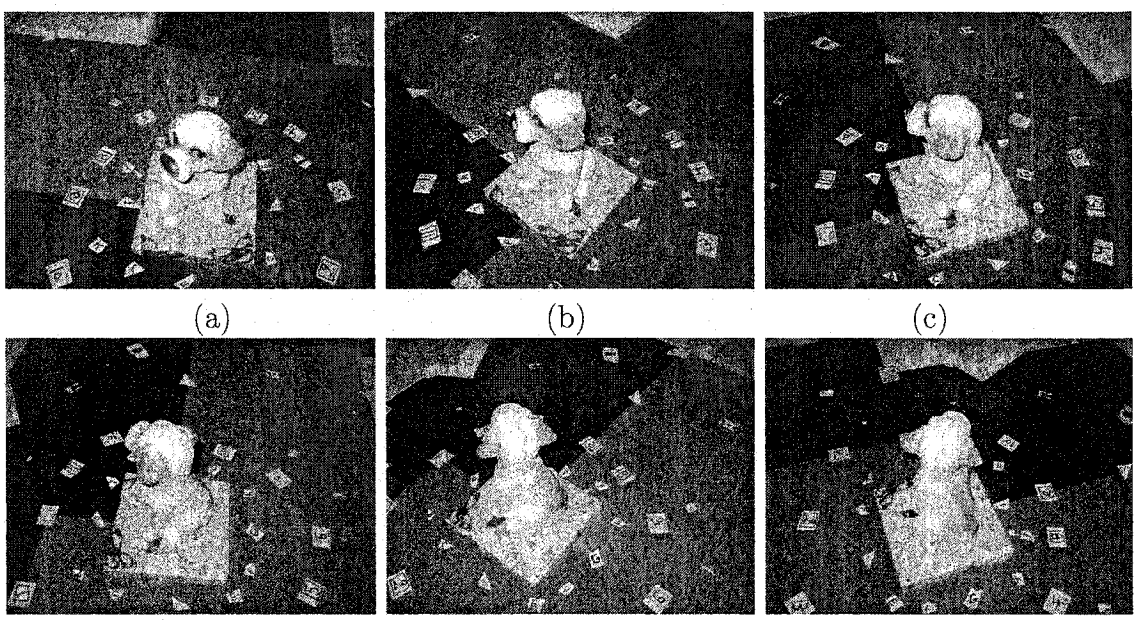
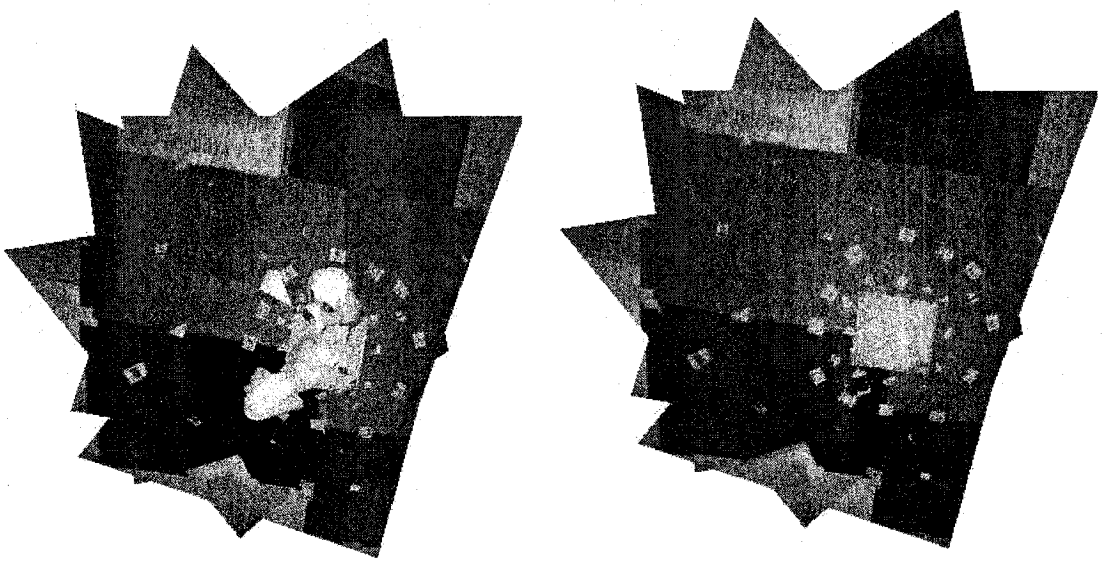


Figure 3.3: Set of images of a ground plane scene.

- **Step 1**: For each point on the ground plane apply the appropriate transformation to obtain the corresponding image point in each view.
- **Step 2**: Compute the mean *RGB* value of the pixels and the pixel that deviates the most from this mean value is discarded.
- **Step 3**: Repeat steps 1 and 2 until half of the points are discarded.
- **Step 4**: The mean *RGB* value of the remaining points is then used in the mosaic composition.

The result obtained using this algorithm is shown in Figure 3.4(b). An appropriate representation is thus obtained that could be used as a ground plane model in the chapters to follow. But before, we show in the next section how it is possible to obtain the inter-image transformations $\mathbf{H}_{ij}$ necessary to build such overhead view mosaic. This is done by matching the different available views, however taken from widely



(a) (b)

Figure 3.4: Mosaic of the set of images in Figure 3.3 : (a) selecting points with best resolution. (b) discarding points that belong to the obstacle.

separated viewpoints.

3.3 Matching two widely separated views

In order to match two views of a scene, feature points must be detected. To this end, we used the Harris corners detector [31]. The detected corners are here the feature of interest and they are mapped on the overhead views on which matching will be performed (see Figure 3.5). Working in the overhead view space has the virtue of eliminating the perspective distortion that deforms the objects in each view. Moreover, to match points in these overhead views, only a rotationally invariant measure is required. Assuming that the intensity variation of images due to the

changes of observation positions is not significant, the Gaussian convolution product is such an invariant measure.

The matching algorithm is presented in the next sections and the two images in Figures 3.5(a) and (b) are used in the process. Both images are taken with the same camera and the orientation angle with the ground is set manually to 45^o. The total number of corners in the first image is $N_1 = 227$, and in the second image is $N_2 = 220$.

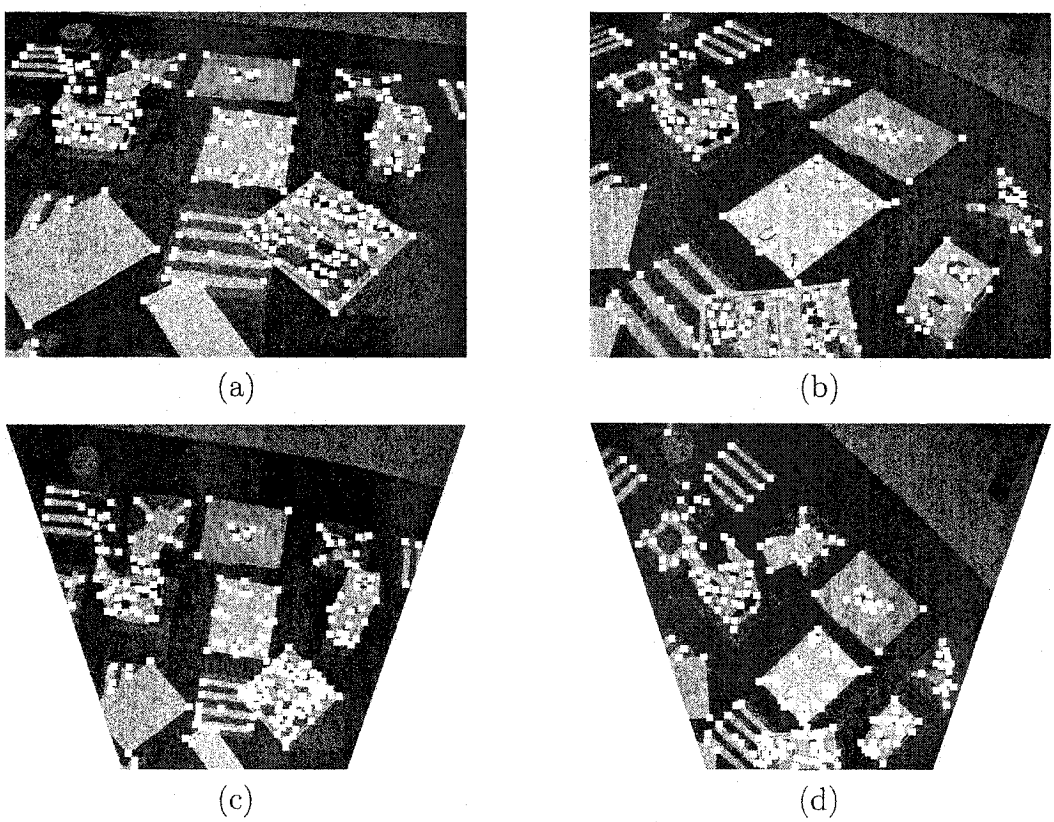


Figure 3.5: Two views of a *carpet* scene taken with camera tilt of 45^o. (a) The first image of the scene has 227 corners. (b) The second image has 220 corners. (c) and (d) The overhead view transformations on which feature points have been mapped.

3.3.1 Matching with color invariants

The Gaussian convolution is a low-pass filter that transforms an image $\mathbf{I}$ as follows:

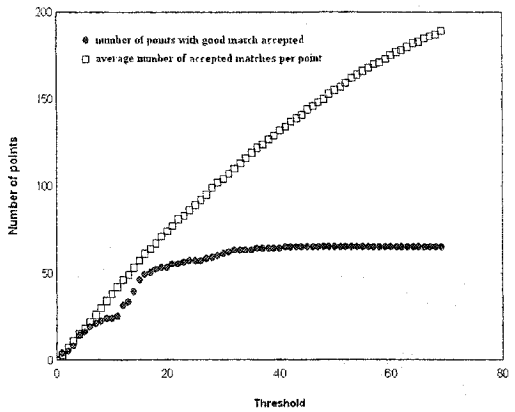
$$\mathbf{I}_\sigma(x,y) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{x^2+y^2}{2\sigma^2}} * \mathbf{I}(x,y) \tag{3.3.1}$$

The corners could then be matched with corners in the second image based on their Gaussian value. Obviously, such a simple matching criterion will lead to several false matches. Improvements can be made in the matching phase by using the color information contained in the images we have at our disposal. One well known color invariant measure is based on Hilbert's color invariants, these having the advantage of requiring only first order derivatives. These derivatives are computed using Gaussian filters. Montesinos et al. [61] proposes to use an invariant vector of 8 such components. In our case, we just use the Gaussian filtered color intensities, $R_\sigma(x,y), G_\sigma(x,y), B_\sigma(x,y)$, and their corresponding gradient magnitudes, $|\nabla_\sigma R(x,y)|, |\nabla_\sigma G(x,y)|, |\nabla_\sigma B(x,y)|$.

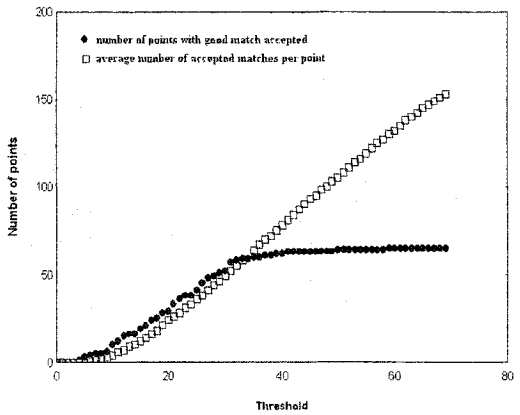
In order to compute the similarity between two such vectors, the components must be resized since color intensities and gradient magnitudes have different ranges of values. Moreover, since the stability of the gradient over viewpoint variation is not very good, we normalize these values using a sigmoidal function, i.e.:

$$s(t) = \frac{255}{1 + e^{-\mu(t-t_o)}} \tag{3.3.2}$$

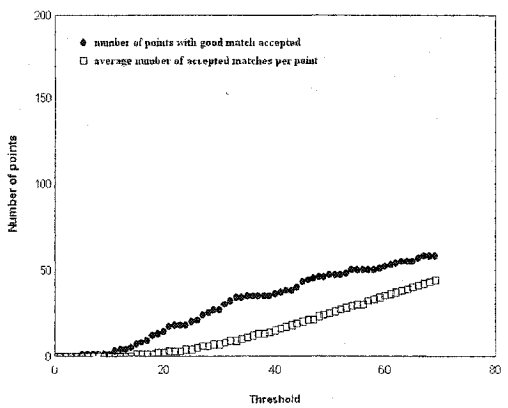
This non-linear normalization leads to a more qualitative characterization where pixels are classified (in a fuzzy way) as point with low or high color gradient magnitude. The value t_o is the threshold that defines the point of transition between low and high magnitudes and μ controls the fuzzyness of the classification (here we used



(a)



(b)



(c)

Figure 3.6: Corner matching (a) using Gaussian of gray-level image $\mathbf{I}_\sigma$ (b) using $[R_\sigma, G_\sigma, B_\sigma]$ (c) using $[R_\sigma, G_\sigma, B_\sigma, s(|\nabla_\sigma R|), s(|\nabla_\sigma G|), s(|\nabla_\sigma B|)]$.

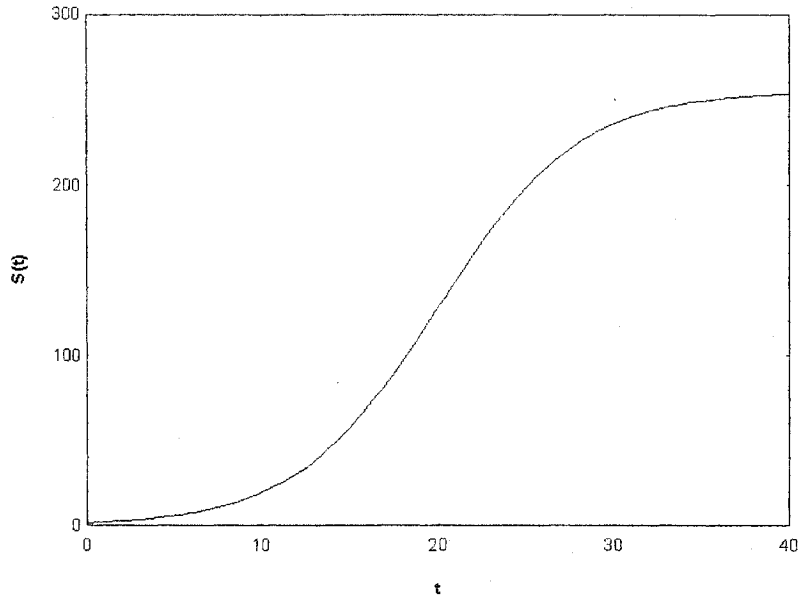


Figure 3.7: The sigmoidal function with $t_0 = 20$ and $\mu = 0.26$.

$t_0 = 20$ and $\mu = 0.26$). The gradient magnitude thus transformed corresponds to the measure of *edgeness* as defined in [38]. This leads us to the following normalized invariant vector:

$$\mathbf{I}_\nu(x,y) = [R_\sigma, G_\sigma, B_\sigma, s(|\nabla_\sigma R|), s(|\nabla_\sigma G|), s(|\nabla_\sigma B|)]^T \tag{3.3.3}$$

The similarity between two putative match points, p and p', is then measured by computing the Euclidean distance, $d_\nu(p,p')$, between their respective invariant vectors $\mathbf{I}_\nu(p)$ and $\mathbf{I}'_\nu(p')$. However the effectiveness of this 6 dimensional invariant vector in the context of feature point matching needs to be validated. In particular, before proposing a matching scheme based on invariants we wanted to justify its use versus the use of the much simpler gray-level Gaussian. To this end we performed the following experiments: First we selected a certain number of typical image pairs for

which the inter image homography was known. With this homography, it was possible to determine for each feature point where its corresponding point is located in the other image, and thus validate the ability of a given invariant vector in identifying a good match.

The conventional matching strategy consists in comparing the invariant vector of a given point in one image with the ones of all points in the other view. All pairs for which the distance is below a certain threshold are accepted as potential matches. An effective invariant vector will be the one that results in few candidate matches for the given point among which the real corresponding point is included. To this end we selected a test set of 70 corners in the first image with known correspondences in the second image. Each point of the test set in the first image is compared with all the points (the 220 corners) in the second image for different threshold values. For each selected threshold in the matching process, we record two values as follows:

N_{av} : The average number of candidate matches obtained for each point of our test set

N_{match}: The number of points in our test set in which the real match was part of the candidate set

In our case, an efficient invariant vector is the one that minimizes the value of N_{av} and maximizes the value of N_{match}. The test is performed using our three invariant measures: The gray Gaussian, the color Gaussian and the vector in Equation (3.3.3). The performance of each vector is shown in Figure 3.6, the light graphs correspond to N_{av} and the solid graphs correspond to N_{match}.

Figure 3.6(a) shows the performance of the matching algorithm when only the gray

Gaussian is used. The performance is further improved by using the color Gaussian $[R_\sigma, G_\sigma, B_\sigma]$ as shown in Figure 3.6(b). The color gradients are introduced and the color vector in Equation (3.3.3) is used in the matching process. Adding the color gradients outperform matching based on Gaussians only, the performance is shown in Figure 3.6(c).

3.3.2 Identifying the isometric transformation

While the transformation between the two original images is a general homography, the transformation between the two composed overhead views is an isometric transformation [32]; it is composed of a rotation and a translation. The transformation has three degrees of freedom, two for translation and one for rotation; it can therefore be computed from two point correspondences. This isometric transformation $\mathbf{H}_{OiOj}$ can be described as follows:

$$\mathbf{H}_{OiOj} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & T_x \\ \sin(\theta) & \cos(\theta) & T_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3.3.4}$$

An invariant in this transformation is the Euclidean distance between two points. Indeed, the line length between the first and the second overhead views is preserved. To make use of this invariant, we proceed as follows. Following a RANSAC-like scheme, we randomly select two matches (as obtained in Section 3.3.1) and the length of the line segments that join the two selected points in each image is compared. If the difference in length is sufficiently low, then the points are considered to be good candidate matches. The number of candidate matches can be further reduced by considering the features introduced by intensity profiles of the candidate segments. Tell and Carlsson [86] introduced an algorithm that compares the affinely invariant

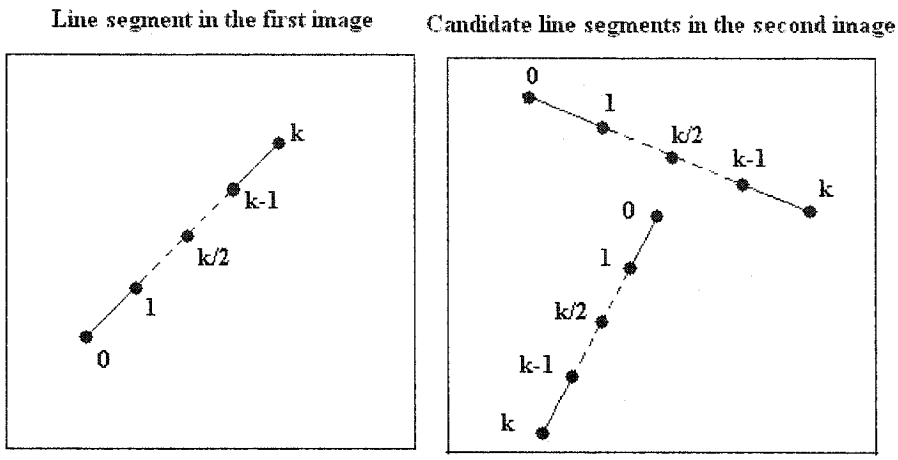


Figure 3.8: The lines in the first and second images are divided into $k+1$ segments.

Fourier features from intensity profiles extracted from the image data between randomly selected pairs of image interest points. We use a similar approach; however, in our algorithm each line segment is divided into $k+1$ points as shown in Figure 3.8.

Then the cross correlation between the line segments in the first and second images is calculated as follows:

$$l_c = \frac{\sum_{i=0}^{k}[(A_i - \bar{A})(B_i - \bar{B})]}{\sqrt{[\sum_{i=0}^{k}(A_i - \bar{A})^2][\sum_{i=0}^{k}(B_i - \bar{B})^2]}} \tag{3.3.5}$$

where A_i and B_i are the pixel intensity of the points in the first image and second image segments respectively. The value of the correlation coefficient l_c can range from -1 to $+1$, with $+1$ indicating perfect match. If the correlation coefficient l_c exceeds a certain threshold then the two segments, and therefore their corresponding points, are assumed to match. Finally, a best-fit approach is used to identify the best transformation.

As an example, Figure 3.9(a) shows two corner points p and q randomly selected

from the first image such that the color Euclidean distance $d_v(p,q) > 2 * MinD_{color}$ ($MinD_{color}$ is a constant parameter set to 50). A corner point p'_i in the second image is a candidate match for corner p if $d_v(p,p'_i) < MinD_{color}$ and similarly a corner point q'_j in the second image is a candidate match for point q if $d_v(q,q'_j) < MinD_{color}$. The total number of matches for corners p and q in the second image is 18 and 29 respectively. The matches for corner p are represented by the " + " sign and those that match corner q are represented by a small square as shown in Figure 3.9(b). The total number of candidate line segments in the second image such that $|\overline{pq} - \overline{p'_i, q'_j}| < MinD_{Euclidean}$ is 120 segments, $MinD_{Euclidean}$ is a parameter set to $0.1 \times \overline{pq}$.

The segment $\overline{pq}$ is cross-correlated with the set of candidate segments in the second image as described in Equation(3.3.5). The number of line segments with cross-correlation value $l_c > Minl_c$ is 11 out of 120, $Minl_c$ is a constant set to 0.7.

The first line segment $\overline{pq}$ and each of the candidate line segments in the second image identify an isometric transformation between the first and second overhead views. In this case, 11 candidate isometric transformation exist between the two overhead views. These candidate transformations are calculated as described in Equation(3.3.4) and a best-fit approach is applied to select the best transformation. The best-fit method is applied by mapping the corner points in the first overhead view to the second overhead view and the transformation that maximizes the number of matches is considered the best transformation. In this example, the maximum number of corner match after mapping corners in Figure 3.9(a) to Figure 3.9(b) using a 4×4 window size is 47. The dotted segment in Figure 3.9 represents the correct

match. The isometric transformation associated with this match is calculated to be:

$$\mathbf{H}_{OiOj} = \begin{bmatrix} 0.82 & -0.57 & 187.58 \\ 0.57 & 0.82 & -58.38 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.3.6}$$

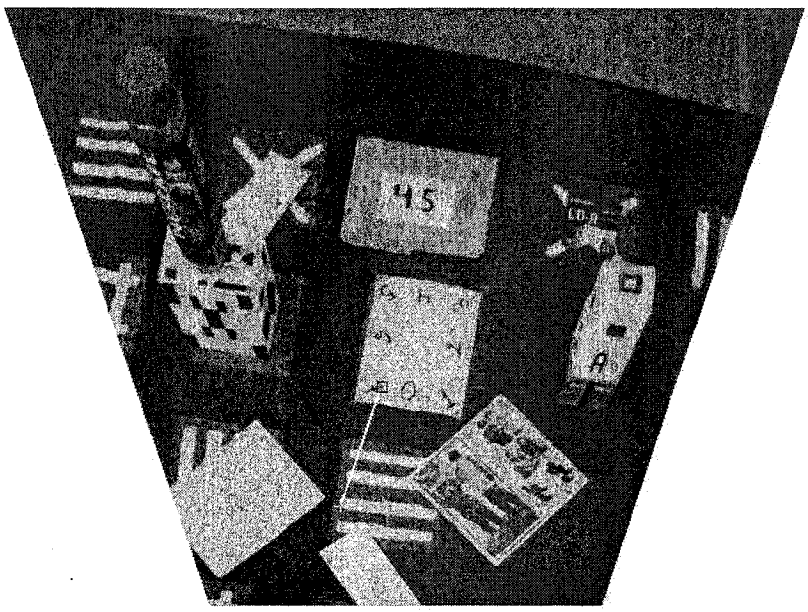
3.4 The inter-image homography

The above matching phase allows to obtain the geometric relation between two overhead views. Using the system geometry, we have also been able to compute the transformation between an image and its corresponding overhead view. We now would like to chain all these transformations in order to obtain the transformation that relates two perspective views of the work surface. That is to say that the transformation between two perspective views of a plane is also a homography and is given by:

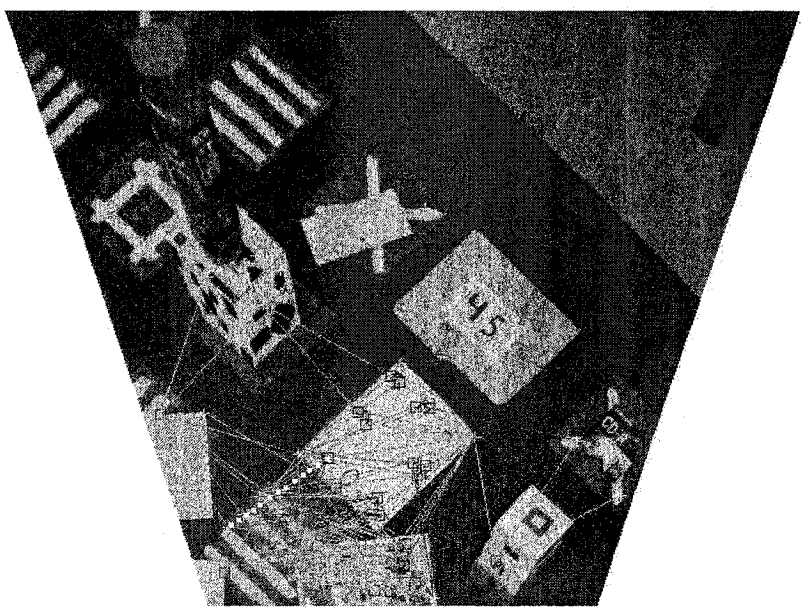
$$\mathbf{H}_{ij} = \mathbf{H}_{Oi}^{-1}\mathbf{H}_{OiOj}\mathbf{H}_{Oj} \tag{3.4.1}$$

where $\mathbf{H}_{Oi}$ and $\mathbf{H}_{Oj}$ are the two overhead homography transformations and $\mathbf{H}_{OiOj}$ is the overhead isometric transformation between the two overhead views. The matrices $\mathbf{H}_{Oi}$ and $\mathbf{H}_{Oj}$ are defined in Equation(3.2.2) depend on the value of α that is an estimate of the true value. However, in order to accurately estimate the value of α and obtain reliable camera position from $\mathbf{H}_{ij}$, this one needs to be refined. This can be done by transferring, using $\mathbf{H}_{ij}$, feature points in one view to the other and searching in a window of size $W \times W$ for the closest detected features (see Figure 3.10). We also impose that the $\mathbf{I}_\nu$ difference between the two matched corners has to be below certain threshold.

The matched corners are used to refine the homography $\mathbf{H}_{ij}$. At least 4 correspondences are needed to rectify the homography. The relation between the points



(a)



(b)

Figure 3.9: Isometric transformation (a) Two randomly selected corners p and q. (b) The set of candidate matches for segment $\overline{pq}$, the dotted line is the match found by the algorithm.

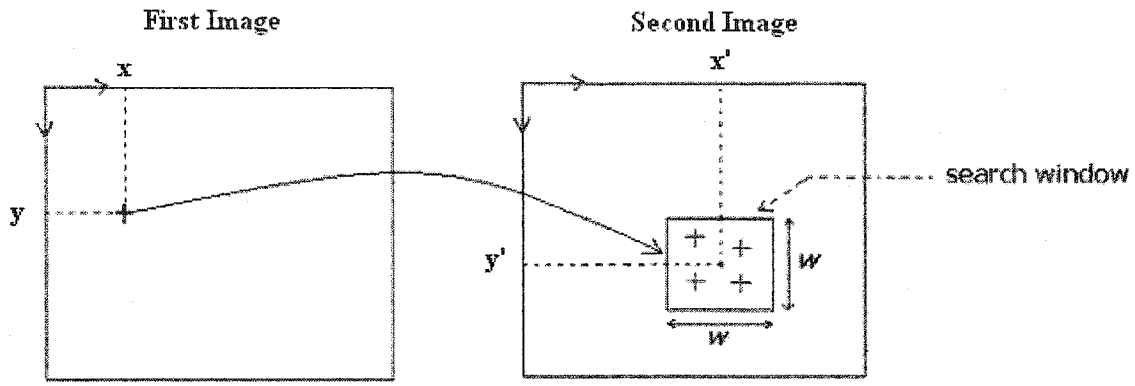


Figure 3.10: The search window for corner matching.

in the two image planes can be described as follows:

$$\mathbf{x}'_n = \mathbf{H}_{ij}\mathbf{x}_n \tag{3.4.2}$$

here we use the notation $\mathbf{x}_n = [x_n, y_n, 1]^T$ to denote a point in image i and its correspondence in image j is denoted by $\mathbf{x}'_n = [x'_n, y'_n, 1]^T$.

A homogenous solution may be obtained using SVD as follows. Writing the $\mathbf{H}_{ij}$ matrix in vector form as [32]:

$$\mathbf{h} = [h_{11} \quad h_{12} \quad h_{13} \quad h_{21} \quad h_{22} \quad h_{23} \quad h_{31} \quad h_{32} \quad h_{33}]^T \tag{3.4.3}$$

For $n \geq 4$ point correspondences, the solution becomes $\mathbf{Ah} = 0$, with $\mathbf{A}$ is a $2n \times 9$ matrix. Each computation match provides two rows of the $\mathbf{A}$ matrix, for n computation match the $\mathbf{A}$ matrix is as follows :

$$\mathbf{A} = \begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x'_1 & -y_1x'_1 & -x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y'_1 & -y_1y'_1 & -y'_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & 1 & 0 & 0 & 0 & -x_nx'_n & -y_nx'_n & -x'_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -x_ny'_n & -y_ny'_n & -y'_n \end{bmatrix} \tag{3.4.4}$$

The vector $\mathbf{h}$ that minimizes the algebraic residuals $\|\mathbf{Ah}\|$, subject to $\|\mathbf{h}\| = 1$, is given by the eigenvector of the least eigenvalue of $\mathbf{A}^T\mathbf{A}$. This eigenvector can be obtained directly from the SVD of $\mathbf{A}$.

The homography transformation between the two views in Figures 3.5(a) and (b) is calculated as defined in Equation (3.4.1) to be:

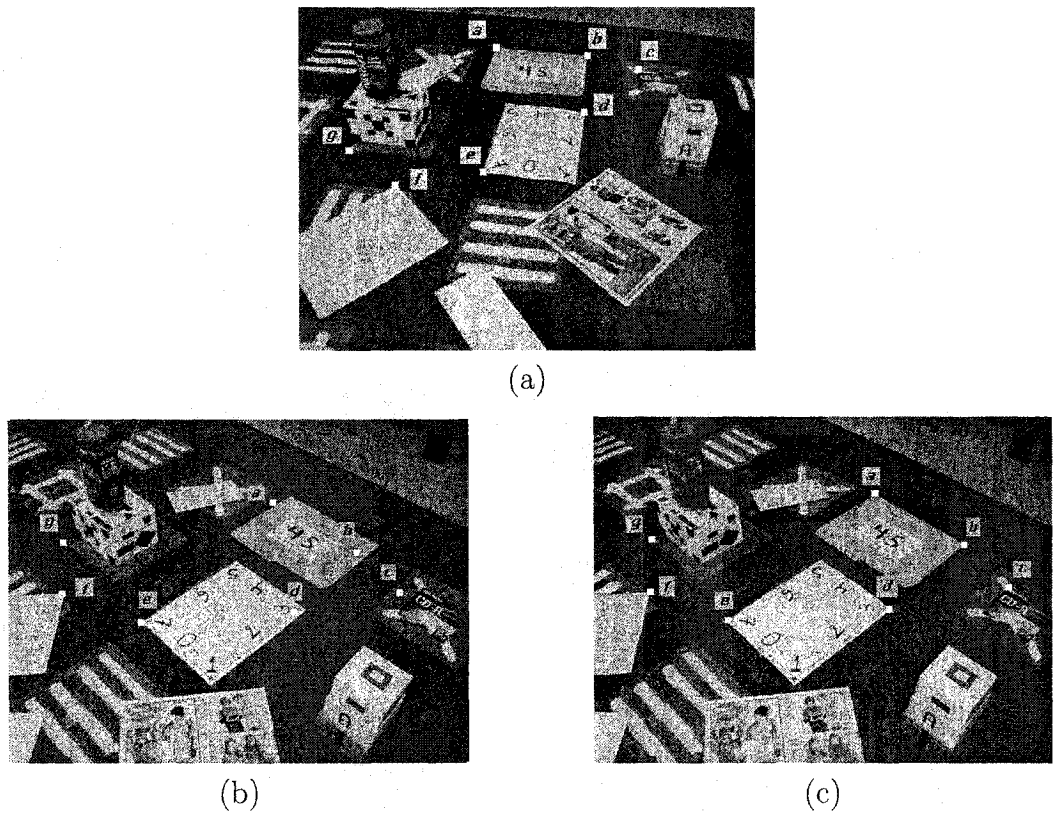
$$\mathbf{H}_{initial} = \begin{bmatrix} 0.53 & -0.80 & 210.12 \\ 0.24 & 0.86 & -21.66 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.4.5}$$

Figure 3.11(a) shows some selected points on the first image, Figure 3.11(b) shows the transferred points as obtained using the homography obtained in Equation (3.4.5). The homography was improved using the refinement step described in this section. The value of the refined homography is calculated to be:

$$\mathbf{H}_{refined} = \begin{bmatrix} 0.59 & -0.88 & 213.66 \\ 0.27 & 0.94 & -39.85 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.4.6}$$

The result of applying the refined homography can be shown in Figure 3.11(c). The improvement can be observed, particularly by comparing the location of points a, b and c in Figures 3.11(b) and (c).

The refined homography is used to estimate α and the motion parameters ($\mathbf{R}$ and $\mathbf{t}$) between the two views, this is explained in Chapter 4.



(a) (b) (c)

Figure 3.11: Homography refinement: (a) Corners selected in the first image(b) Correspondences using the original homography (c) Correspondences using the refined homography.

3.5 Algorithm analysis

Throughout this section we use the notation $d_v(p,q)$ to represent the color Euclidean distance between color vectors of points p and q, the color vector being defined in Equation(3.3.3). The notation $d\perp(p,q)$ represents the Euclidean distance between corner points p and q. The cost of computing $d_v(p,q)$ and $d\perp(p,q)$ is defined by the constants C_c and $C_\perp$ respectively. The matching algorithm can be summarized into the following steps:

- Step 1: Select two random corners (p,q) from the first overhead view image such that $d_v(p,q) > 2 * MinD_{color}$.
- Step 2: Find two sets of corners P_m and Q_n in the second overhead view matching corners p and q respectively. The two sets are defined as follows:
  $P_m = \{p_i^{'}/d_v(p,p_i^{'}) < MinD_{color}, i \in (0...n_p)\}$
  $Q_n = \{q_j^{'}/d_v(q,q_j^{'}) < MinD_{color}, j \in (0...n_q)\}$
- Step 3: Find the candidate lines in the second overhead view image such that $|d\perp(p,q) - d\perp(p_i^{'},q_j^{'})| < MinD_{Euclidean}$ where $p_i^{'} \in P_m$ and $q_j^{'} \in Q_n$.
- Step 4: Find the normalized cross correlation between the line segment $\overline{pq}$ and the candidate lines in the second image. If $l_c > Minl_c$ then $(p,p^{'})$ and $(q,q^{'})$ are added to the set of candidate matches.
- Step 5: Calculate the isometric homographies $\mathbf{H}_{O_iO_j}$ using the candidate matches from Step 4. Keep the homography with the highest best fit.

3.5.1 Complexity analysis

Let N_i and N_j be the number of corners in the overhead views i and j respectively and n_p and n_q are the number of corners that match corners p and q in the second view. First the two sets P_m and Q_n are two disjoint sets ($P_m \cap Q_n = \emptyset$), they form a bipartite graph where each edge between the two sets represents a candidate segment in the second overhead view whose Euclidean length equals that of the segment $\overline{pq}$ in

the first overhead view.

In Steps 1 and 2 the complexity is obviously of $O(C_c N_i)$ and $O(C_c N_j)$ respectively, in Step 3 the worst case is when each element $p_i^{'} \in P_m$ is compared with every element $q_i^{'} \in Q_n$ such that $d\perp(p,q) = d\perp(p_i^{'}, q_j^{'})$, the complexity is $O(C_\perp n_p n_q)$. In Step 4 the complexity is $O(C_l k)$ where k is the number of candidate lines as defined in Step 3 and the constant C_l represents the cost of line cross-correlation as defined in Equation (3.3.5). The number of pairs of corners in sets P_m and Q_n that are at distance $d\perp(p,q)$ from each other is $O(N_j^{4/3})$[84], thus $k \approx O(N_j^{4/3})$.

In Step 5 we assume that the number of candidate lines is $O(k)$, then the complexity is $O(C_s N_i k)$ where the constant C_s is the cost of calculating the isometric transformation in Equation (3.3.4). The total cost of the algorithm C_{Total} is the sum of all steps:

$$C_{Total} = O(C_c N_i) + O(C_c N_j) + O(C_\perp N_j^2) + O(C_l k) + O(C_s N_i k) \tag{3.5.1}$$

Since k is $O(N_j^{4/3})$, the complexity of the algorithm is dominated by Step 5 and the total cost is:

$$C_{Total} \approx O(C_s N_i k) \approx O(N_i N_j^{4/3}) \tag{3.5.2}$$

In practice the value of k depends on the length of the segment $\overline{pq}$ and the minimum distance between corners (d_{cor}). Figure 3.12 shows the search region for a corner $p_i^{'} \in P_m$ within a distance of $R \pm d_R$ from $q_i^{'}$. The value of R is the length of the first image segment $d\perp(p,q)$. The maximum number of matches for the corner point $p_i^{'}$ is achieved by triangulating the search region with corners from set Q_n. The triangles are equilateral where each side equals d_{cor}, the area of each triangle is $\frac{\sqrt{3}}{2} d_{cor}^2$, the area

of the search region is $4\pi R d_R$. The total number of triangles in the search region is:

$$n_{\triangle} = \frac{8\pi R d_R}{\sqrt{3} d_{cor}^2} \tag{3.5.3}$$

Let $d_R = \delta R$ where $\delta < 1$, the number of triangles in Equation(3.5.3) is expressed as follows:

$$n_{\triangle} = \frac{14.5\delta R^2}{d_{cor}^2} \tag{3.5.4}$$

The triangulated region represents a planar graph $G(V, E)$ with vertex set V and edge set E. The number of vertices v, edges e and faces f are related by the classical *Euler's formula* [2] as follows:

$$v - e + f = 2 \tag{3.5.5}$$

In our case each triangle has at least two edges in common with the neighboring triangles, then each vertex in the graph has degree ≥ 3. By applying this property to Equation(3.5.5), it is simple to prove that:

$$v \leq 2f - 4 \tag{3.5.6}$$

The number of faces equals $n_{\triangle}$ plus the two faces inside and outside of the triangulated region, then $f = n_{\triangle} + 2$. The number of vertices of graph G can be obtained by substituting f in Equation (3.5.6) to obtain:

$$v \leq 2n_{\triangle} = \frac{29\delta R^2}{d_{cor}^2} \tag{3.5.7}$$

Finally, the value of v in Equation (3.5.7) can be substituted for the value of k in Equation (3.5.2). The total complexity of the algorithm is:

$$C_{Total} \leq O\left(\frac{29 C_s \delta N_i R^2}{d_{cor}^2}\right) \approx O\left(\frac{\delta N_i R^2}{d_{cor}^2}\right) \tag{3.5.8}$$

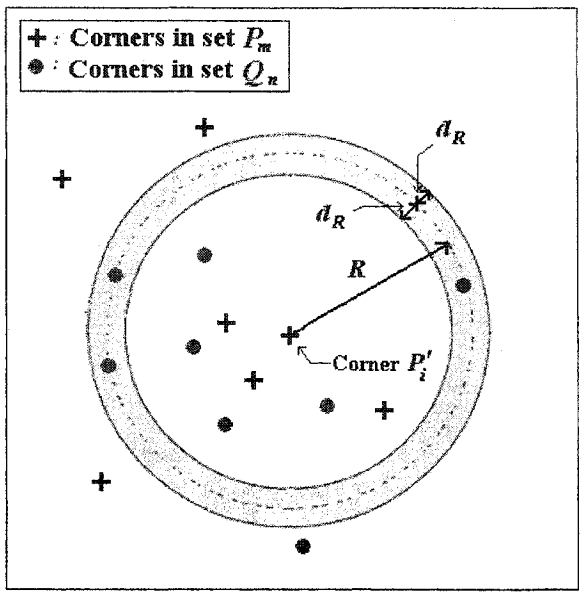


Figure 3.12: Search area: The search area for corner p_i'.

Equation(3.5.8) shows that the complexity of the algorithm is directly proportional on the length R of the segment in the first image. This means that as R increase, the number of candidate matches in the second image also increases. To verify this relation, an empirical experiment has been applied on the two images in Figure 3.5. The two images in Figure 3.5 (a) and (b) has $N_1 = 227$ and $N_2 = 220$ corners respectively, the corners are detected with a minimum distance between corners $d_{cor} >$ 5 pixels.

Two corners p and q are selected in the first image with different length $R = \overline{pq}$, the number of matching corners in the second image for corners p and q are n_p and n_q respectively. The number of line segments $n_{segments}$ in the second image increases as the length of the segment $R = \overline{pq}$ increases as shown in Table 3.1.

$\overline{pq}$ in pixels	n_p	n_q	$n_{segments}$	$l_c > 0.7$
12	116	32	19	2
23	57	86	62	7
30	117	53	109	12
40	112	58	151	11
55	48	116	196	8
68	112	32	263	9
75	116	48	393	14
80	81	64	350	23
134	116	50	1193	113
172	104	66	1741	205

Table 3.1: Empirical results: **Column** 1 contains the length of the segment in pixels for two selected corners p and q in the first image. **Columns** 2 and 3 shows the number of matched corners in the second image for points p and q respectively. **Column** 4 shows the number of line segments in the second image to be matched with the segment $\overline{pq}$. **Column** 5 shows the number of segments in the second image with cross-correlation $l_c > 0.7$.

3.5.2 Statistical analysis

The algorithm assumes that the two corners randomly selected lies on the ground plane and their matched corners in the second images also lie on the ground plane. Let a be the probability of selecting a corner point on the ground plane of the first image such that its match in the second image also lies on the ground plane. Let $p = a^2$ be the probability of selecting two points on the ground floor, and $q = 1 - p$ be the probability of selecting at most one point on the ground floor. Let k be a random variable to represent the number of trials to select two corners on the ground floor. The mean of k is defined as follows:

$$\begin{aligned} E(k) &= \sum_{i=0}^{\infty} ipq^{i-1} \qquad (3.5.9) \\ &= p + 2pq + 3pq^2 + 4pq^3 + \ldots + ipq^{i-1} + \ldots \\ &= p(1 + 2q + 3q^2 + 4q^3 + \ldots + iq^{i-1} + \ldots) \end{aligned}$$

but for $q < 1$:

$$\frac{1}{(1-q)^2} = 1 + 2q + 3q^2 + 4q^3 + \ldots + iq^{i-1} + \ldots$$

Then

$$E(k) = \frac{p}{(1-q)^2} = \frac{1}{p} \tag{3.5.10}$$

In our algorithm we assume that approximately 50% of the corners are on the ground plane, then $a = 0.5$ and $E(k) = 4$. This means that on the average we need 4 trials to select two corners on the ground plane. However to be confident with certain probability z that at least one out of the k trial is correct:

$(1-p)^k = (1-z)$

The number of trials required is:

$$k = \frac{log(1-z)}{log(1-p)} \tag{3.5.11}$$

To be 90% confident that at least one selection is error-free with $a = 0.5$, the number of trials needed is $k = 10.41$. In our implementation we use $k = 16$ which is equivalent to 99% confidence.

In Figure 3.5, the number of corners on the ground plane is 165 and on the obstacles is 62 for the first image. For the second image there are 63 corners on the obstacles and 157 on the ground plane. Some of the ground plane corners in the first image do not have correspondences on the ground plane of the second image. This is due to the fact that some corners are occluded in the second image by the obstacles above the ground, or because some corners are not visible from the second image or because corresponding corners are not detected in the second image. The number of corners lying on the ground plane of the first image with corresponding corners also

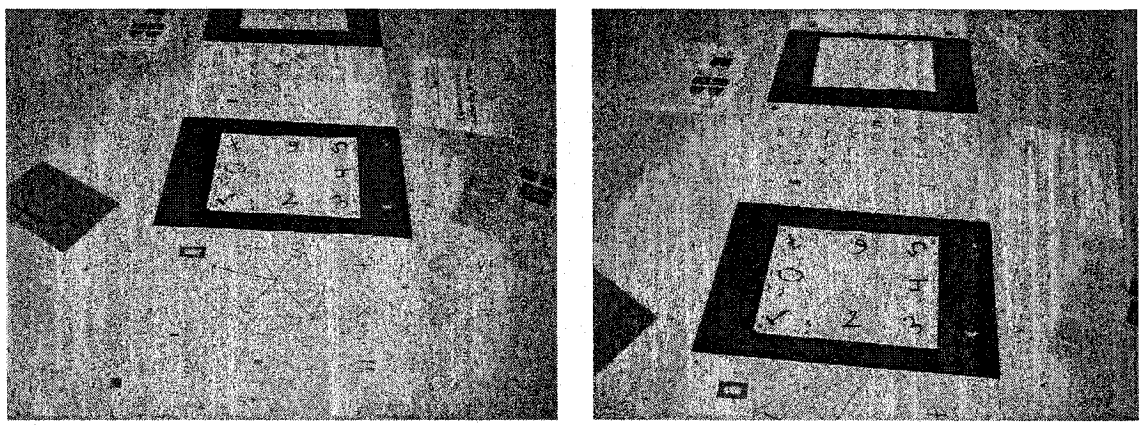
lying on the ground plane in the second image is calculated to be 110. The probability of selecting one corner point on the ground plane of the first image with corresponding corner lying on the ground plane in the second image is as indicated in this section $a = \frac{110}{227} = 0.484$. The probability of selecting two points on the ground plane is $p = a^2 = 0.234$. The average number of trials needed is as expressed in Equation (3.5.10) is $E(k) = \frac{1}{p} = 4.25$. In this example it took an average of 5 iterations to get the correct solutions which is close to the theoretical number of 4.25 iterations.

3.6 More experiments

Two additional experiments are performed to demonstrate the validity of the algorithm. Figure 3.13 shows two images of a scene with the tilt set manually to 45^o. The algorithm is applied on these two images and the homography transformation is calculated. Figure 3.14 shows some visual comparison of the results obtained. Figure 3.15 shows another set of images where the algorithm applied. The two images in Figure 3.15 (a) and (b) are taken with a camera set manually to tilt of 22^o and 33^o respectively. Figure 3.16 shows some visual information about the results obtained. Table 3.2 shows a summary of the results obtained after applying the algorithm on the images presented in this chapter. In the next chapter this table is used to estimate the camera motion between each set of images.

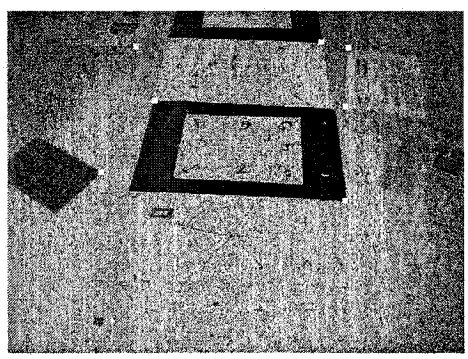
3.7 Summary

In this chapter I present a novel technique for matching sparse views of a work surface. My main contribution is by integrating different wide baseline matching algorithm in

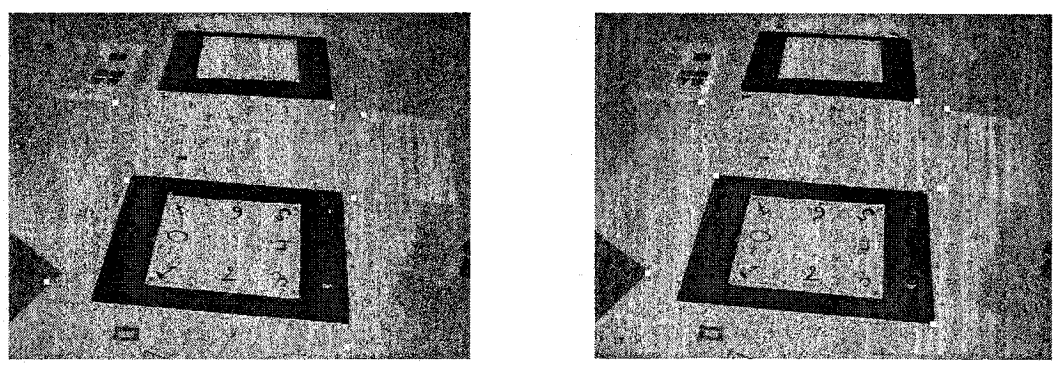


(a) (b)

Figure 3.13: Floor images (a)and (b) images for a floor taken at tilt of 45°.

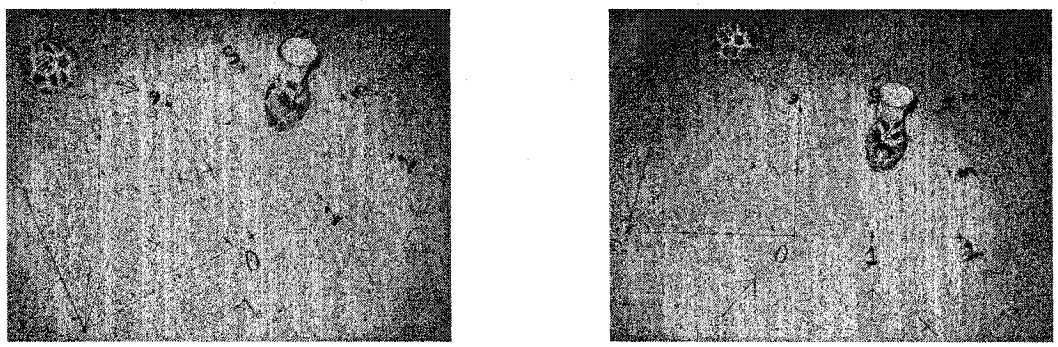


(a)



(b) (c)

Figure 3.14: Homography refinement: (a) Corners selected in the first image(b) Correspondences using the original homography (c) Correspondences using the refined homography.



(a) (b)

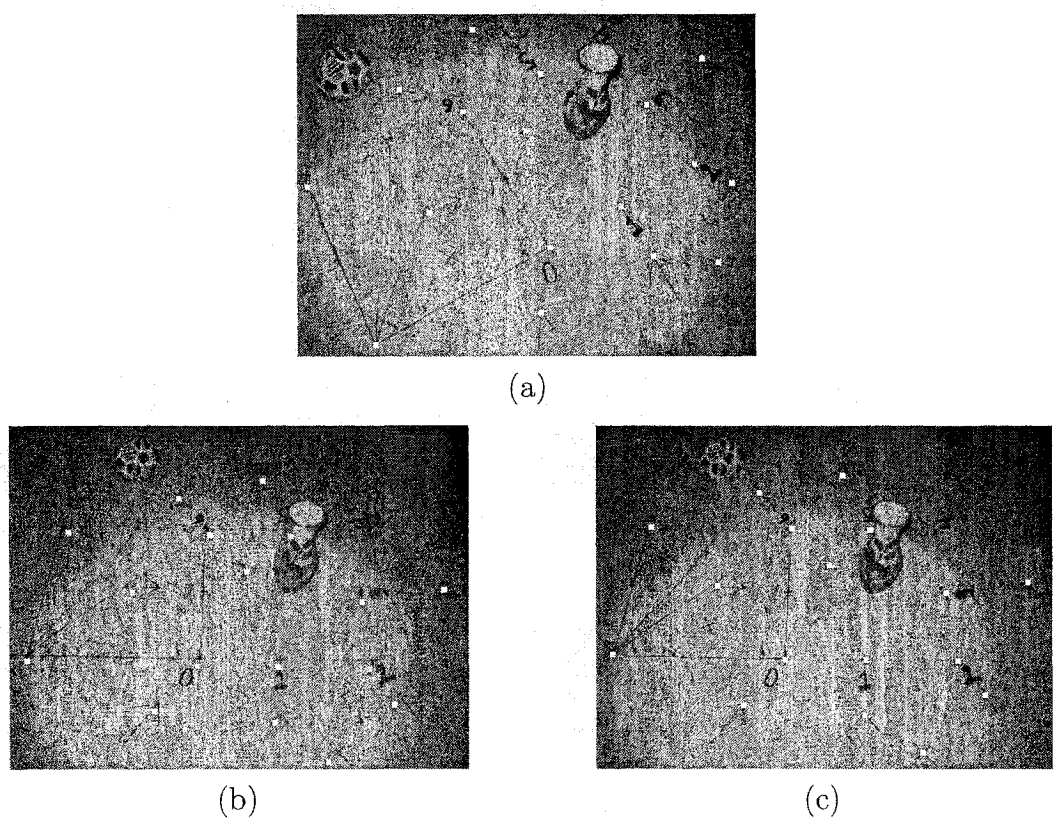
Figure 3.15: Vase images: (a) tilt = 22^o and (b) tilt = 33^o.

Scene	$H_{initial}$	$H_{refined}$
carpet	$\begin{bmatrix} 0.53 & -0.80 & 210.12 \\ 0.24 & 0.86 & -21.66 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 0.59 & -0.88 & 213.66 \\ 0.27 & 0.94 & -39.85 \\ 0 & 0 & 1 \end{bmatrix}$
floor	$\begin{bmatrix} 1.07 & -0.23 & -37.25 \\ 0.03 & 1.23 & 54.15 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1.10 & -0.15 & -41.62 \\ 0.01 & 1.25 & 56.36 \\ 0 & 0 & 1 \end{bmatrix}$
vase	$\begin{bmatrix} 0.56 & -0.51 & 198.09 \\ 0.27 & 0.63 & -8.72 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 0.57 & -0.48 & 186.32 \\ 0.25 & 0.61 & -12.40 \\ 0 & 0 & 1 \end{bmatrix}$

Table 3.2: Summary for calculated homography transformation

order to match images collected by a mobile robot moving on a planar ground with a tilted camera.

The idea of my proposed technique is to use the overhead view images in the matching process. The procedure first starts by detecting corners on the original perspective images and the corners are then mapped to the overhead view images. Corner matching starts on the overhead view images in order to eliminate the projective deformation between the two views. The Hilbert's color invariants are used in the matching process. The similarity transformation is calculated between the two



(a) (b) (c)

Figure 3.16: Homography refinement: (a) Corners selected in the first image(b) Correspondences using the original homography (c) Correspondences using the refined homography.

overhead views by matching two corners in the first overhead view with two corners in the second overhead view, the number of candidate matches is reduced by calculating the cross-correlation between the line segment defined by the two randomly selected corners in the first overhead view with the line segments produced by the putatively matched corners in the second view. The homography between the two overhead views is calculated and the inter-image homography between the two original views is estimated. The inter-image homography is further refined in order to get more reliable camera position.

Chapter 4

Vision-Based multi-robot Simultaneous Localization and Mapping

In this chapter a vision-based approach for the multi-robot Simultaneous Localization and Mapping (SLAM) problem is presented. We study the case of a team of robots equipped with a single camera on each and collaborating in the same worksite. The only information used to solve this problem is the visual data coming from cameras mounted on the robots. We propose to calculate the location of the robots by using a collection of sparse views of the planar surface on which these robots are moving. The camera motions are estimated using inter-image homographies computed from the matching of overhead transformed views (Chapter 3). Results of maps generated from the estimated robot locations are presented.

4.1 Introduction

The ability of a mobile robot to localize itself in its work site is a key prerequisite towards developing autonomous robots. This problem has gained considerable attention in the robotics community in the recent few years. Many authors have proposed various approaches to estimate robot location with some a priori information about the environment. Krotkov [41] assumes that the robot has a map of its working environment. The map marks the positions of vertically oriented objects in the scene such as doors, desks, etc. The robot is then localized by establishing a correspondence between the landmark directions and points in the map.

Sim and Dudek [78] use learned landmarks to perform position estimation of a mobile robot. The extrema of the density of the edge distribution in each image is selected as landmarks. The landmarks are detected from a preliminary traversal of the environment and then saved in a database. A feature vector that relates the landmarks and the camera pose is then created. The mobile robot position is estimated by matching the landmarks extracted from the image with the landmarks saved in the database. Assuming a linear variation in the landmark characteristics, an estimation of the camera pose is based on linear interpolation of the landmark feature vectors.

Dudek and Zhang [12] use multi-layer neural networks to estimate the pose of the robot. The neural network is trained in the preliminary phase using a set of training images at known locations and orientations. The perceptual structure (like edge distribution and edge orientation) detected in the training images are associated with the known position of the robot and then expressed as a collection of features. These features are used by an artificial neural network to construct a nonlinear interpolator.

When the robot takes a new image, the pose of the robot is estimated by interpolation.

Werman et al. [95] introduce a robot localization method based on image measurements that are invariant to the intrinsic parameters of the camera. The measurements depend only on the extrinsic parameters .i.e the rotation and translation ($\mathbf{R}$ and $\mathbf{t}$) of the camera with respect to some world coordinate system. The algorithm is based on detecting landmarks (feature points) with known Euclidean coordinates with respect to a reference frame. If a sufficient number of features are detected by the camera (at least five), then the robot pose can be determined.

Thrun et al. [88] developed a tour-guide robot to guide people in an indoor environment (like a museum). The robot localizes itself using two maps generated in a learning phase. The first map, the occupancy map [87], is learned using laser scans and odometry data. The second map is the mosaic of the ceiling using camera images and odometry data. At each new location, the robot collects the sensor data (camera images, odmetry readings and laser scans). The localization is based on incorporating the sensor data and the learned maps using a Markov localization approach [39, 80].

Grabowski and Kosla [25] introduced a self-localization algorithm for a team of small robots. Each robot is equipped with a beacon transceiver that allows position measurement by coordinating transmission and reception of beacon signals. Each robot in turn transmits a series of beacon signals, the remaining robots detect and correlate the time between signals and calculate their position with respect to the transmitting robot.

The robot is said to be truly autonomous [11], if it has the ability to start at an unknown location in an unknown environment and then simultaneously build a map of the environment and localize itself in the map. Thus the robot has to solve the

simultaneous localization and mapping (SLAM) problem. The information used are sequences of relative observations captured by the mobile robot. Many approaches has been proposed to solve the SLAM problem.

Se et al. [74] proposed an approach for a robot equipped with a trinocular stereo system [65] and an odometer. The algorithm starts by detecting feature points [52] in the three images. Then these feature points are matched using the epipolar and disparity constraints that exist between the three cameras. Assuming known camera intrinsic parameters, the 3D position of each matched feature point is estimated. As the robot moves, the odometry readings are used to provide a rough estimate of the motion. This estimate can be employed in matching feature points among consecutive frames. The 3D position of each newly detected feature point is estimated and the motion of the robot is localized using a least-squares minimization scheme [51]. Finally, a map is built for the 3D positions of the detected feature points and the location graph of the robot is computed.

Other SLAM methods are based on evaluating some probabilistic models of the robot motion and sensed data from the environment. It is assumed that the robot can sense landmarks relative to its local coordinate frame. The landmarks may be naturally occurring in the environment like trees or artificially added like steel poles.

Smith and Cheeseman [81] use extended Kalman filters (EKF) to estimate the posterior distribution over the robot pose. The problem they solved can be stated as follows: Given a measurement of the environment $\mathbf{z}^t = [z_1, z_2, ..., z_t]$ and a set of control inputs $\mathbf{u}^t = [u_1, u_2, ..., u_t]$, determine the robot pose $\mathbf{s}^t$ and the location of all the k landmarks $\mathbf{m} = [m_1, m_2, ..., m_k]$. In probabilistic terms, this can be expressed as the posterior:

$$p(\mathbf{s}^t, \mathbf{m}/\mathbf{z}^t, \mathbf{u}^t) \tag{4.1.1}$$

where the superscript ^{t} refers to a set of variables at time t.

The extended kalman filters (EKFs) are relatively slow when estimating maps with very large number of landmarks. Murphy [64] and Montemerlo et al. [60] proposed more capable approaches. They proposed algorithms to solve the SLAM problem by integrating particle filter and Kalman filter representation to solve the posterior function in Equation (4.1.1).

When a team of robots are sharing the same worksite, the SLAM problem becomes more challenging. The robots have to build a joint map of the environment and be able to localize their positions in the joint map in order to coordinate the navigation and minimize the overlap in information.

Thrun et al. [89] presented a multi-robot SLAM algorithm based on likelihood maximization to find the maps that are maximally consistent with the sensor data. The sensor data used are laser scanners to sense the environment and odometers to estimate the robot motion. The exact initial pose of all robots relative to each other is assumed to be known. The localization process starts as follows: Each robot collects a sequence of its own odometry and sensor measurements (like laser scans). Using these measurements, each robot incrementally constructs a maximum likelihood estimate for its position and a maximum likelihood estimate for the location of surrounding objects and a posterior function to determine its location in the map. To build a joint map, the posterior estimation component is used. The relative location of robots is unknown; however, each robot in the team starts within a map of a specific robot called the team leader. Using the posterior estimation, each robot localizes itself in

the team leader's map and thus a unified map for the team is built.

Simmons et al. [79] introduced similar approach but with known approximate initial pose of the robots (within 1 meter distance and 20^o orientation). Each robot in the team incrementally constructs the likelihood and posterior estimates as in the approach proposed by Thrun et al. [89]. However, to build a joint map, each robot in the team sends its sensed data (laser range scans and odometer readings) to the team leader. Since the initial approximate pose of each robot is known, the team leader localizes the robots relative to each other.

When the initial pose of the robots is unknown, an important question is raised:

> *If two robots in the team are sensing similar data in the environment, are they actually sensing the same part of the environment or are they sensing different parts that look alike?*

Liu and Thrun [48] presented an approach to solve the multi-robot SLAM problem assuming unknown initial positions and ambiguous landmarks. Each robot in the team builds a local map similar to ones discussed in [89] and [79]. The joint map is built by fusing the maps acquired by the robots into one map. First, the landmarks between different local maps are matched. Correspondences found in this matching process provide an estimate of the rotation and translation between local maps. Using this estimate, a global map of the environment may be built by estimating a posterior similar to the one defined in Equation(4.1.1) evaluated over all robot poses and all landmarks from all available data. Liu and Thrun applied the algorithm using a single vehicle equipped with laser range finder and an odometer in an outdoor environment. Features in the map are the stems of trees detected by the laser range finder. The multi-robot case is demonstrated by splitting the data acquired by the vehicle into 8

disjoint sequences and then the proposed multi-robot SLAM algorithm is applied.

In this chapter, we propose to solve the multi-robot SLAM problem by using a collection of sparse views of the scene. The originality of our approach is that it uses purely vision sensors (single camera on each robot) and that no special landmarks are assumed to exist in the environment. Moreover, no assumptions are made on the initial relative pose of the robots.

The robot location estimated from sparse views making the estimation much more accurate. In contrast, methods based on small incremental displacements are more affected by noise and inaccuracies and are thus more subject to error accumulation. In our proposed algorithm, each robot in the team starts at an arbitrary unknown location and incrementally builds a local map of the environment with the ability to localize itself in the map. When an overlap occurs between any two robots, a joint map can be built between them and the two robots are able to localize themselves in the joint map for all their previous as well as future locations without the need for a new overlap. Under this approach a joint map of the team can be built if each robot has at least one overlap with any other robot in the team.

In our approach we simply assume that each robot is equipped with a single camera and the robots are operating on a planar surface. We also assume that the height of the camera with respect to the ground plane as well as its orientation are known. Note that the tilt angle does not have to be accurate since this one is simply used as an initial approximation and will be re-estimated by the algorithm. However, a good accuracy in the height measure could be required since robot localization will be specified in terms of height units.

4.2 Robot localization

4.2.1 Estimating the camera motion

In Chapter 3, the computed inter-image homography matrices have been used to generate an overhead view mosaic representing the ground plane model. In this section, these matrices are used to obtain the pose of each camera. The inter-image homography transformation for two cameras C_i and C_j viewing a planar scene can be expressed as defined in Equation(2.5.2) to be:

$$\mathbf{H}_{ij} = \mathbf{K}_j \left[\mathbf{R} - \frac{\mathbf{t}\mathbf{n}^T}{d} \right] \mathbf{K}_i^{-1} \tag{4.2.1}$$

where $\mathbf{K}_i$ and $\mathbf{K}_j$ are the matrices containing the intrinsic parameters of cameras C_i and C_j respectively, $\mathbf{R}$ is the rotation between the two cameras, $\mathbf{n}$ is the normal to the plane under consideration and $\mathbf{t}$ is the translation. Finally d is the distance from the camera to the ground. The optical axis of the two cameras are along the Z-axis as described in Figure 3.1, then the rotation matrix between the two cameras can be expressed as follows [1]:

$$\mathbf{R} = \begin{bmatrix} c_\theta & -s_\theta c_{\alpha_i} & -s_\theta s_{\alpha_i} \\ c_{\alpha_j} s_\theta & c_\theta c_{\alpha_j} c_{\alpha_i} + s_{\alpha_j} s_{\alpha_i} & c_{\alpha_j} s_{\alpha_i} c_\theta - c_{\alpha_i} s_{\alpha_j} \\ s_{\alpha_j} s_\theta & s_{\alpha_j} c_\theta c_{\alpha_i} - c_{\alpha_j} s_{\alpha_i} & c_\theta s_{\alpha_j} s_{\alpha_i} + c_{\alpha_j} c_{\alpha_i} \end{bmatrix} \tag{4.2.2}$$

where α_i and α_j are the angle of cameras C_i and C_j with respect to the ground plane and θ is the angle between the two cameras.

The normal to the ground plane $\mathbf{n}$ can be expressed with respect to camera C_i as

follows:

$$\mathbf{n} = \begin{bmatrix} 0 \\ -\sin(\alpha_i) \\ \cos(\alpha_i) \end{bmatrix} \tag{4.2.3}$$

The matrix $\frac{\mathbf{t}\mathbf{n}^T}{d}$ is therefore defined to be:

$$\frac{\mathbf{t}\mathbf{n}^T}{d} = \begin{bmatrix} 0 & \frac{-t_x \sin(\alpha_i)}{d} & \frac{t_x \cos(\alpha_i)}{d} \\ 0 & \frac{-t_y \sin(\alpha_i)}{d} & \frac{t_y \cos(\alpha_i)}{d} \\ 0 & \frac{-t_z \sin(\alpha_i)}{d} & \frac{t_z \cos(\alpha_i)}{d} \end{bmatrix} \tag{4.2.4}$$

The value $\mathbf{R} - \mathbf{t}\mathbf{n}^T$, with d arbitrarily set to 1 is given by :

$$\mathbf{R} - \mathbf{t}\mathbf{n}^T = \begin{bmatrix} c_\theta & -s_\theta c_{\alpha_i} + t_{x_i} s_{\alpha_i} & -s_\theta s_{\alpha_i} - t_{x_i} c_{\alpha_i} \\ c_{\alpha_j} s_\theta & c_\theta c_{\alpha_j} c_{\alpha_i} + s_{\alpha_j} s_{\alpha_i} + t_{y_i} s_{\alpha_i} & c_{\alpha_j} s_{\alpha_i} c_\theta - c_{\alpha_i} s_{\alpha_j} - t_{y_i} c_{\alpha_i} \\ s_{\alpha_j} s_\theta & s_{\alpha_j} c_\theta c_{\alpha_i} - c_{\alpha_j} s_{\alpha_i} + t_{z_i} s_{\alpha_i} & c_\theta s_{\alpha_j} s_{\alpha_i} + c_{\alpha_j} c_{\alpha_i} - t_{z_i} c_{\alpha_i} \end{bmatrix} \tag{4.2.5}$$

with $c_\theta = \cos\theta$ and $s_\theta = \sin\theta$. If the homography $\mathbf{H}_{ij}$ between the two images is known then:

$$\mathbf{R} - \mathbf{t}\mathbf{n}^T = \mathbf{K}_j^{-1}\mathbf{H}_{ij}\mathbf{K}_i = \begin{bmatrix} a_i & b_i & c_i \\ d_i & e_i & f_i \\ g_i & h_i & i_i \end{bmatrix} \tag{4.2.6}$$

The tilt α_j of camera C_j can be calculated as:

$$\alpha_j = \tan^{-1}\left(\frac{g_i}{d_i}\right) \tag{4.2.7}$$

The angle between the two cameras is given by

$$\theta = \cos^{-1}(a_i) \tag{4.2.8}$$

Similarly, the translation vector can be obtained by combining Equations (4.2.5) and (4.2.6). A double solution is obtained as follows:

$$\mathbf{t}_1 = \begin{bmatrix} \frac{b_i + \sin(\theta)\cos(\alpha_i)}{\sin(\alpha_i)} \\ \frac{e_i - \cos(\theta)\cos(\alpha_j)\cos(\alpha_i) - \sin(\alpha_j)\sin(\alpha_i)}{\sin(\alpha_i)} \\ \frac{h_i - \cos(\theta)\cos(\alpha_i)\sin(\alpha_j) + \cos(\alpha_j)\sin(\alpha_i)}{\sin(\alpha_i)} \end{bmatrix} \tag{4.2.9}$$

$$\mathbf{t}_2 = \begin{bmatrix} \frac{-c_i - \sin(\theta)\sin(\alpha_i)}{\cos(\alpha_i)} \\ \frac{-f_i - \cos(\theta)\sin(\alpha_i)\cos(\alpha_j) - \cos(\alpha_i)\sin(\alpha_j)}{\cos(\alpha_i)} \\ \frac{-i_i + \cos(\theta)\sin(\alpha_j)\sin(\alpha_i) + \cos(\alpha_j)\cos(\alpha_i)}{\cos(\alpha_i)} \end{bmatrix} \tag{4.2.10}$$

The translation vector $\mathbf{t}$ can be estimated by the average of $\mathbf{t}_1$ and $\mathbf{t}_2$:

$$\mathbf{t} = \frac{\mathbf{t}_1 + \mathbf{t}_2}{2} \tag{4.2.11}$$

To calculate α_i, consider the inverse of Equation (4.2.5) as follows:

$$[\mathbf{R} - \mathbf{t}\mathbf{n}^T]^{-1} = [\mathbf{K}_j^{-1}\mathbf{H}_{ij}\mathbf{K}_i]^{-1} = \begin{bmatrix} a_j & b_j & c_j \\ d_j & e_j & f_j \\ g_j & h_j & i_j \end{bmatrix} \tag{4.2.12}$$

From Equation (4.2.12), the tilt α_i of camera C_i can be calculated as:

$$\alpha_i = \tan^{-1}\left(\frac{g_j}{d_j}\right) \tag{4.2.13}$$

The motion parameters can be also estimated using SVD decomposition as proposed by Triggs [90] as follows:

Let the coordinates of camera C_i be the reference frame, the projection matrices of cameras C_i and C_j are respectively:

$$\mathbf{P}_i = [\mathbf{I} \mid \mathbf{0}] \tag{4.2.14}$$

$$\mathbf{P}_j = \mathbf{R}\,[\mathbf{I} \mid -\mathbf{t}] \tag{4.2.15}$$

where $\mathbf{R}$ is the rotation matrix, $\mathbf{t}$ is translation vector and $\mathbf{I}$ is the 3×3 identity matrix.

The inter-image homography matrix $\mathbf{H}_{ij}$ can be decomposed as follows:

$$\mathbf{H}_{ij} = \mathbf{R}\left[\mathbf{I} - \frac{\mathbf{t}'\mathbf{n}^T}{d}\right] \tag{4.2.16}$$

where $\mathbf{t} = -\mathbf{R}\mathbf{t}'$.

For a point $\mathbf{x}$ located on the plane, $\frac{\mathbf{n}^T\mathbf{x}}{d} = 1$ and Equation (4.2.16) can be expressed as follows:

$$\mathbf{H}_{ij} = \mathbf{R}\left[\mathbf{I} - \mathbf{t}'\right] = \mathbf{R}\mathbf{H}^*_{ij} \sim \mathbf{P}_j \tag{4.2.17}$$

The motion can be estimated from the SVD's $\mathbf{H}_{ij}$ and $\mathbf{H}^*_{ij}$.

$$\begin{aligned} \mathbf{H}_{ij} &= \mathbf{U}\mathbf{S}\mathbf{V} \\ \mathbf{H}^*_{ij} &= \mathbf{U}_\mathbf{1}\mathbf{S}\mathbf{V} \end{aligned} \tag{4.2.18}$$

where $\mathbf{U}$ and $\mathbf{V}$ are 3×3 rotation matrices denoted by the columns $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3]$ and $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3]$, $\mathbf{S} = diag(s_s, s_2, s_3)$ is a diagonal matrix and $\mathbf{U} = \mathbf{R}\mathbf{U}_1$.

The motion between the two cameras $(\mathbf{R}, \mathbf{t}')$ can be estimated as follows:

$$\mathbf{t}' = \frac{\beta_1}{s_1}\mathbf{v}_1 + \frac{\beta_2}{s_3}\mathbf{v}_3 \tag{4.2.19}$$

$$\mathbf{R} = \mathbf{U}\mathbf{U}_1^T \tag{4.2.20}$$

where $\beta_1 = \pm\sqrt{1 - s_3^2}$ and $\beta_2 = \pm\sqrt{s_1^2 - 1}$

In general, the SVD approach gives two distinct solutions. This indetermination can be resolved if additional information about the scene is known. In our case, the normal vector to the ground plane is known (Equation (4.2.3)) and is used to eliminate the ambiguity.

In the following section an experiments is performed to estimate the motion parameters between two cameras. The internal camera parameters are assumed to be known.

Experiment : motion estimation

In this section the details of calculating the motion between two camera is provided. The carpet example in Figures 3.5(a) and (b) is used here to calculate $\mathbf{R}$ and $\mathbf{t}$. The refined homography is defined in Equation (3.4.6) and is rewritten again for convenience:

$$\mathbf{H}_{refined} = \begin{bmatrix} 0.599 & -0.883 & 213.665 \\ 0.270 & 0.946 & -39.851 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.2.21}$$

First assuming known $\mathbf{K}_i$ and $\mathbf{K}_j$, The value of $\mathbf{R} - \mathbf{t}\mathbf{n}_i^T$ is calculated as described in Equation (4.2.6) to be as follows:

$$\mathbf{R} - \mathbf{t}\mathbf{n}^T = \begin{bmatrix} 0.801 & 0.970 & -0.114 \\ -0.429 & 0.931 & -0.103 \\ -0.439 & -0.129 & 0.874 \end{bmatrix} \tag{4.2.22}$$

It follows that:

$$[\mathbf{R} - \mathbf{t}\mathbf{n}^T]^{-1} = \begin{bmatrix} 0.817 & -0.85 & 0.006 \\ 0.429 & 0.678 & 0.136 \\ 0.474 & -0.326 & 1.202 \end{bmatrix} \tag{4.2.23}$$

From Equation (4.2.8) the angle between the two images is $\theta = cos^{-1}(0.801) = 36.7^o$ and the tilt angle of cameras C_i and C_j are calculated as defined in Equations(4.2.7) and (4.2.13) to be: $\alpha_j = tan^{-1}(\frac{-0.439}{-0.429}) = 45.6^o$ and $\alpha_i = tan^{-1}(\frac{0.474}{0.429}) = 47.8^o$.

The normal to the ground plane $\mathbf{n}$, the rotation matrix $\mathbf{R}$ and the translation vector $\mathbf{t}$ are calculated as defined in Equations (4.2.3), (4.2.2) and (4.2.11) to be:

$$\mathbf{n} = \begin{bmatrix} 0 \\ -0.714 \\ 0.699 \end{bmatrix} \quad \mathbf{R} = \begin{bmatrix} 0.799 & 0.419 & 0.429 \\ -0.419 & 0.902 & -0.1 \\ -0.429 & -0.1 & 0.897 \end{bmatrix} \quad \mathbf{t} = \begin{bmatrix} -0.76 \\ -0.04 \\ -0.015 \end{bmatrix} \tag{4.2.24}$$

The SVD approach is applied to the matrix in Equation (4.2.21) to compare the results. The following two solutions were obtained:

Solution 1:

$$\mathbf{n} = \begin{bmatrix} -0.039 \\ -0.697 \\ 0.715 \end{bmatrix} \quad \mathbf{R} = \begin{bmatrix} 0.787 & 0.413 & 0.457 \\ -0.43 & 0.899 & -0.07 \\ -0.44 & -0.141 & 0.886 \end{bmatrix} \quad \mathbf{t} = \begin{bmatrix} -0.798 \\ -0.045 \\ -0.016 \end{bmatrix} \tag{4.2.25}$$

Solution 2:

$$\mathbf{n} = \begin{bmatrix} 0.709 \\ 0.683 \\ 0.17 \end{bmatrix} \quad \mathbf{R} = \begin{bmatrix} 0.613 & 0.772 & -0.164 \\ -0.787 & 0.585 & -0.189 \\ -0.05 & 0.245 & 0.968 \end{bmatrix} \quad \mathbf{t} = \begin{bmatrix} -0.289 \\ -0.505 \\ 0.548 \end{bmatrix} \tag{4.2.26}$$

The solution is the one with normal vector that satisfies the normal to the plane $\mathbf{n}$ as defined in Equation (4.2.3). In this case **Solution** 1 is selected.

Although the two methods give similar results, in our work the SVD solution is used for one main reason. The geometric solution in Equation (4.2.24) is based on a strict camera configuration as shown in Figure 3.1. Any deviation from this configuration may require to be compensated for using an extra bundle adjustment step. The SVD solution has no restrictions on the geometry of the system. This makes the SVD solution more practical in our multi-robot environment.

The SVD algorithm is used to calculate the camera motion for the images in Figures 3.5, 3.13 and 3.15 based on both the initial and refined homography transformations shown in Table 3.2. The results are shown in Table 4.1.

In the next section, the robot localization algorithm is presented. The algorithm is

applied on Table 4.1 and the localization results for the initial and refined homography transformations are compared.

Scene	using $\mathbf{H}_{initial}$	using $\mathbf{H}_{refined}$
carpet	$\mathbf{R} = \begin{bmatrix} 0.773 & 0.429 & 0.465 \\ -0.419 & 0.898 & -0.132 \\ -0.475 & -0.093 & 0.875 \end{bmatrix}$ $\mathbf{t} = [-0.731\ 0.04\ -0.025]^T$ $\mathbf{n} = [0.002\ -0.613\ 0.79]^T$ $\theta = 39.32^o \quad \alpha_j = 48.55^o$	$\mathbf{R} = \begin{bmatrix} 0.787 & 0.413 & 0.457 \\ -0.43 & 0.899 & -0.07 \\ -0.44 & -0.141 & 0.886 \end{bmatrix}$ $\mathbf{t} = [-0.798\ -0.045\ -0.016]^T$ $\mathbf{n} = [-0.039\ -0.697\ 0.715]^T$ $\theta = 38.02^o \quad \alpha_j = 45.61^o$
floor	$\mathbf{R} = \begin{bmatrix} 0.998 & 0.034 & 0.04 \\ -0.033 & 0.999 & -0.009 \\ -0.04 & 0.008 & 0.999 \end{bmatrix}$ $\mathbf{t} = [-0.119\ -0.253\ -0.288]^T$ $\mathbf{n} = [0.073\ -0.699\ 0.711]^T$ $\theta = 3.032^o \quad \alpha_j = 50.36^o$	$\mathbf{R} = \begin{bmatrix} 0.999 & 0.025 & 0.028 \\ -0.026 & 0.999 & 0.016 \\ -0.028 & -0.016 & 0.999 \end{bmatrix}$ $\mathbf{t} = [-0.082\ -0.256\ -0.209]^T$ $\mathbf{n} = [-0.013\ -0.661\ 0.75]^T$ $\theta = 2.14^o \quad \alpha_j = 47.09^o$
vase	$\mathbf{R} = \begin{bmatrix} 0.841 & 0.489 & 0.226 \\ -0.445 & 0.867 & -0.219 \\ -0.304 & 0.083 & 0.948 \end{bmatrix}$ $\mathbf{t} = [-0.3\ 0.206\ 0.147]^T$ $\mathbf{n} = [0.037\ -0.421\ 0.906]^T$ $\theta = 32.66^o \quad \alpha_j = 34.31^o$	$\mathbf{R} = \begin{bmatrix} 0.852 & 0.474 & 0.221 \\ -0.424 & 0.873 & -0.237 \\ -0.306 & 0.108 & 0.945 \end{bmatrix}$ $\mathbf{t} = [-0.289\ 0.239\ 0.15]^T$ $\mathbf{n} = [-0.01\ -0.377\ 0.926]^T$ $\theta = 31.55^o \quad \alpha_j = 35.81^o$

Table 4.1: Camera motion using the initial and refined homography transformations.

scene	h	d_{cam}	$\Gamma_{initial}$	$\Gamma_{refined}$	$\Delta_{d_{initial}}$	$\Delta_{d_{refined}}$
carpet	78 *cm*	61.5 *cm*	$[0.73, -3.66, 141.4]$	$[0.79, -3.18^o, 138.76^o]$	4.56 cm	0.588 *cm*
floor	80 *cm*	27 *cm*	$[0.40, -72.76, 104.2]$	$[0.33, -75.99^o, 101.76^o]$	5.01 cm	0.008 *cm*
vase	78 *cm*	32 *cm*	$[0.39, 39.76, 97.59]$	$[0.40, 44.09^o, 102.06^o]$	1.58 cm	0.8 *cm*

Table 4.2: The calculated robot position from the estimated homographies

4.2.2 Locating the robots

The localization problem is formalized as shown in Figure 4.1. This can be parameterized by the triplet $\Gamma = [\rho, \phi_1, \phi_2]$. Where ρ is the Euclidean distance between the two robots, ϕ_1 is the angle of *Robot*2 with respect to *Robot*1 and ϕ_2 is the angle of *Robot*1 with respect to *Robot*2.

The robot locations with respect to one another may be expressed by projecting the two 3D camera coordinate systems on *Robot*1 and *Robot*2 2D coordinate systems. The location of *Robot*2 with respect to *Robot*1 is at (x_1, y_1) defined as follows:

$$x_1 = -t_{x_1} \tag{4.2.27}$$

$$y_1 = \sqrt{t_{y_1}^2 + t_{z_1}^2}\sin(\alpha_1 + \beta_1) \tag{4.2.28}$$

$$\rho_1 = \sqrt{x_1^2 + y_1^2} \tag{4.2.29}$$

where $\beta_1 = \tan^{-1}\frac{t_{y_1}}{t_{z_1}}$

Similarly, the location of *Robot*1 with respect to *Robot*2 is at (x_2, y_2) defined as follows:

$$x_2 = -t_{x_2} \tag{4.2.30}$$

$$y_2 = \sqrt{t_{y_2}^2 + t_{z_2}^2}\sin(\alpha_2 + \beta_2) \tag{4.2.31}$$

$$\rho_2 = \sqrt{x_2^2 + y_2^2} \tag{4.2.32}$$

where $\beta_2 = \tan^{-1}\frac{t_{y_2}}{t_{z_2}}$

Finally, the location vector Γ is defined as follows:

$$\Gamma = [\rho, \phi_1, \phi_2] = \left[\frac{\rho_1 + \rho_2}{2}, tan^{-1}\left(\frac{y_1}{x_1}\right), tan^{-1}\left(\frac{y_2}{x_2}\right)\right] \tag{4.2.33}$$

The value of ρ in Equation (4.2.33) is calculated in term of camera height units. For

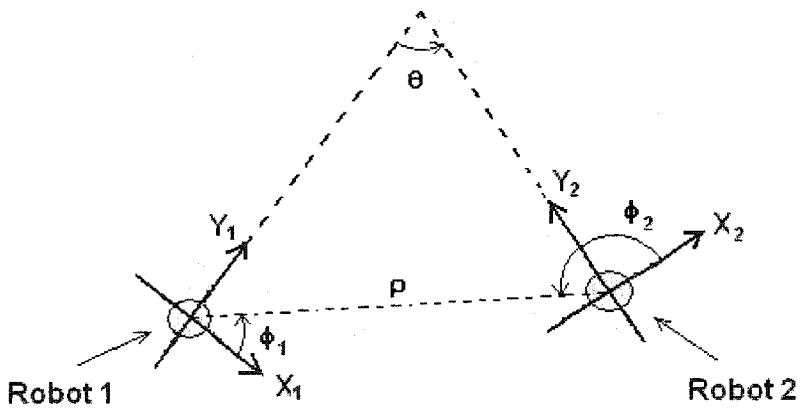


Figure 4.1: The Robot angles.

a known camera height h in cm, the value of Γ can be expressed as follows:

$$\Gamma = [\rho h \text{ cm}, \phi_1, \phi_2] \tag{4.2.34}$$

The information in Table 4.1 is used to localize the robots in the *carpet*, *floor* and *vase* scenes. The algorithm is applied using the motion estimated for the initial and refined cases as calculated in Table 4.1 columns 2 and 3. The localization

results are shown in Table 4.2. The camera height h and the measured distance between the cameras for each scene are shown in columns 2 and 3 respectively. The location vectors for the initial and refined cases are calculated as described in this section and the results are shown in columns 4 and 5 respectively. To compare the improvements introduced by the refined homography, the measured camera distance d_{cam} and the calculated distance ρh are compared. Column 6 shows the absolute difference between the measured (d_{cam}) and the calculated (ρh) camera distance for the initial homography case. Column 7 shows the absolute difference for the refined homography case.

4.3 SLAM experiments

In this section two experimental examples are provided for a robot traversing a work site. The objective is that the robot is capable of calculating its pose with respect to its previous position.

4.3.1 Single-robot SLAM

Figure 4.2(a) shows a set of images collected by a robot moving on a planar surface, the tilt with the ground is set manually to 33^o and the height of the cameras is measured to be $h = 82\,\text{cm}$. First, the matching algorithm discussed in Chapter 3 is applied and the inter-image homographies are calculated between consecutive robot locations. The homography transformation calculated are: $\mathbf{H}_{01}$, $\mathbf{H}_{12}$, $\mathbf{H}_{23}$, $\mathbf{H}_{34}$, $\mathbf{H}_{45}$, $\mathbf{H}_{56}$ and $\mathbf{H}_{67}$. These homographies are used to incrementally build a location graph. At each new location, the robot pose is calculated with respect to its previous position

as described in Section 4.2.2. The resulting location graph is shown in Figure 4.3. To estimate the accumulated error in estimating the pose between consecutive locations, the pose between the initial location (Robot0) and last location (Robot 7) is estimated by concatenating the calculated inter-image homography transformations as follows:

$$\mathbf{H}_{07} = \mathbf{H}_{67}\mathbf{H}_{56}\mathbf{H}_{45}\mathbf{H}_{34}\mathbf{H}_{23}\mathbf{H}_{12}\mathbf{H}_{01} \tag{4.3.1}$$

Any error in estimating any of the matrices in the right hand side of Equation(4.3.1) will influence the estimation of $\mathbf{H}_{07}$. In Figure 4.3 the value of r_1 represents the measured distance between Robot 7 and Robot 0, the estimated distance using $\mathbf{H}_{07}$ is represented by r_2.

The error in estimating the distance between Robot 0 and Robot7 can be evaluated by $d_r = |r_1 - r_2|$ as shown in Figure 4.3. The value of d_r is $11.5cm$ for a total displacement error of 2.95%.

The global overhead view map of the environment can be built by combining the overhead transformation and the inter-image homographies as described in Chapter 3. The result is shown in Figure 4.4(a). Due to the parallax effect [42] points above the ground plane are not mapped correctly (see the 'vase' on Figure 4.4(a)). To build a map that shows only the ground points, we use the procedure described in Section 3.2.1. This procedure may produce some blurring effect due to the pixel averaging step. The map obtained is shown in Figure 4.4(b).

In the next section, our single-robot SLAM algorithm is generalized to multiple-robot case.

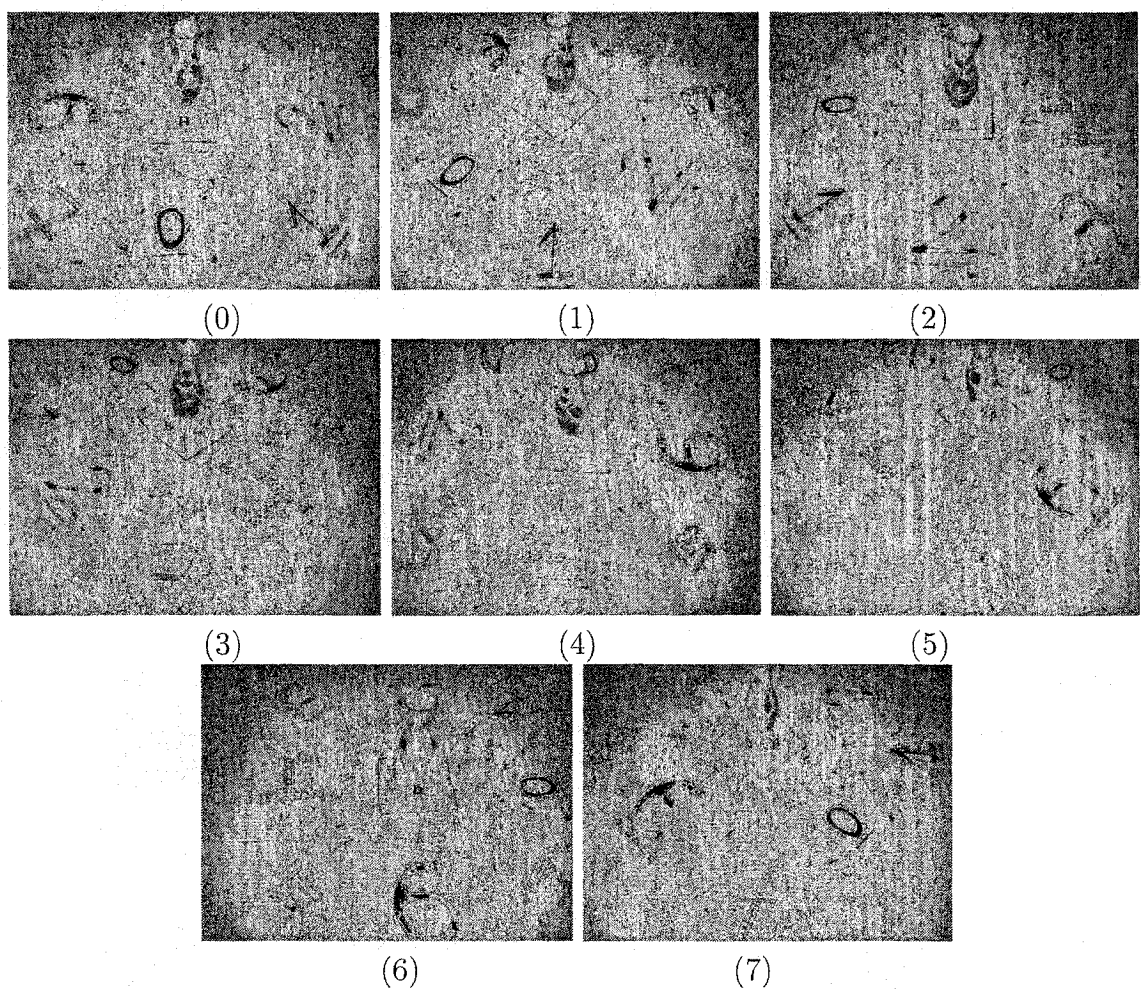


Figure 4.2: Single-robot SLAM: Image set collected by the robot.

4.3.2 Multi-robot SLAM

In this section we provide an example of two mobile robots *RobotA* and *RobotB* moving in the same work site. The camera tilt on robots *RobotA* and *RobotB* are set manually to 33^o and 45^o respectively, the height of the camera from the ground plane is $55cm$. Figure 4.5 shows the images captured by *RobotA* and Figure 4.6 shows the images captured by *RobotB*. The two robots start at arbitrary unknown locations

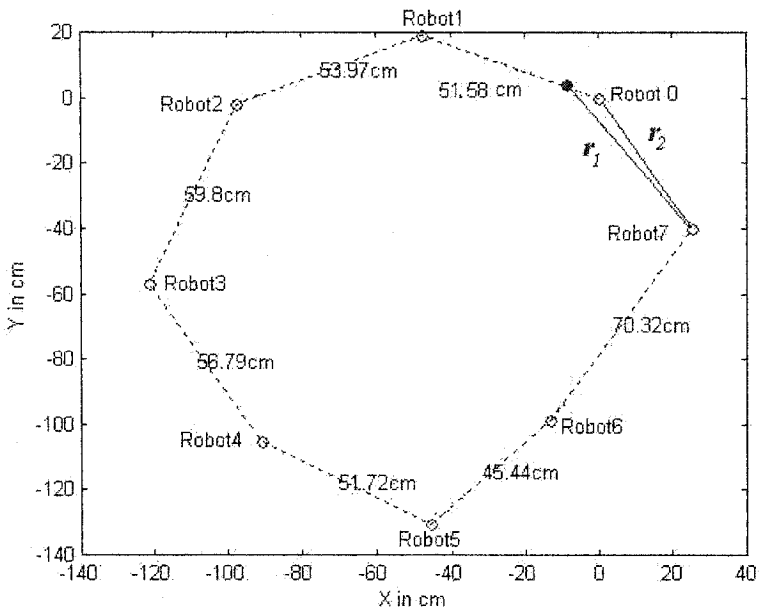
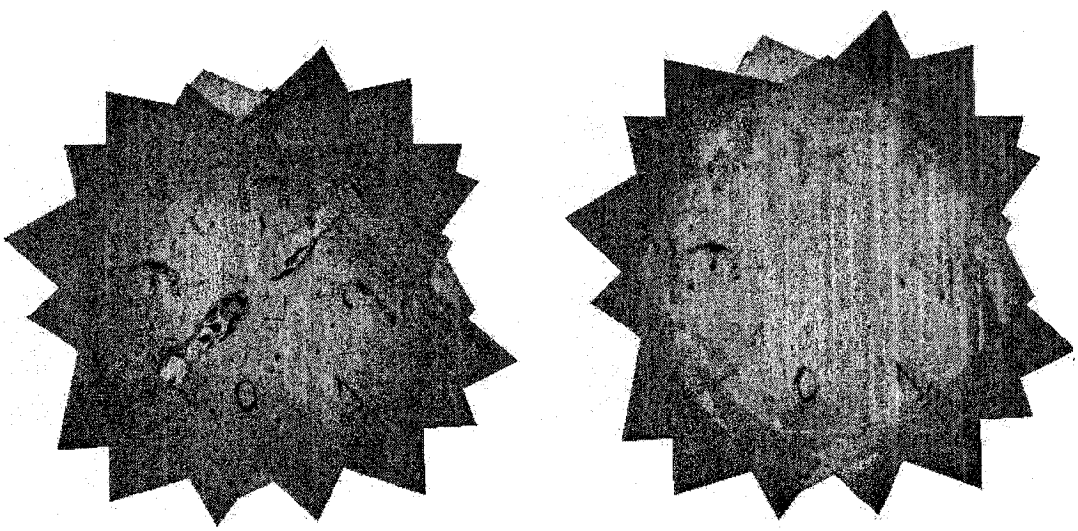


Figure 4.3: Single-robot SLAM: Location graph.

(location $A0$ for $RobotA$ and location $B0$ for $RobotB$). Each of the robots starts building a local location graph and a local map of the environment as discussed in the single-robot SLAM case, the local locations graphs are shown in Figure 4.7(a) and (b). When an overlap occurs between the two robots, a joint location graph and environment map can be build by calculating the relative pose of the two robots. The approach used to determine the overlap between two robots is presented in the next section. At location $A7$ for $RobotA$ and location $B8$ for $RobotB$ the two robots are viewing the same scene, this overlap can be verified by comparing image $A7$ in Figure 4.5 and image $B8$ in Figure 4.6. The relative pose of the two robots is estimated by the inter-image homography between the two images and a joint location graph can be built. Figure 4.7(c) shows the joint location graph which relates all the robot locations with $A7$ selected as the reference frame. The joint map of the environment is shown in Figure 4.8.



(a) (b)

Figure 4.4: Single-robot SLAM (a) generated map of the site (b) map with obstacles discarded.

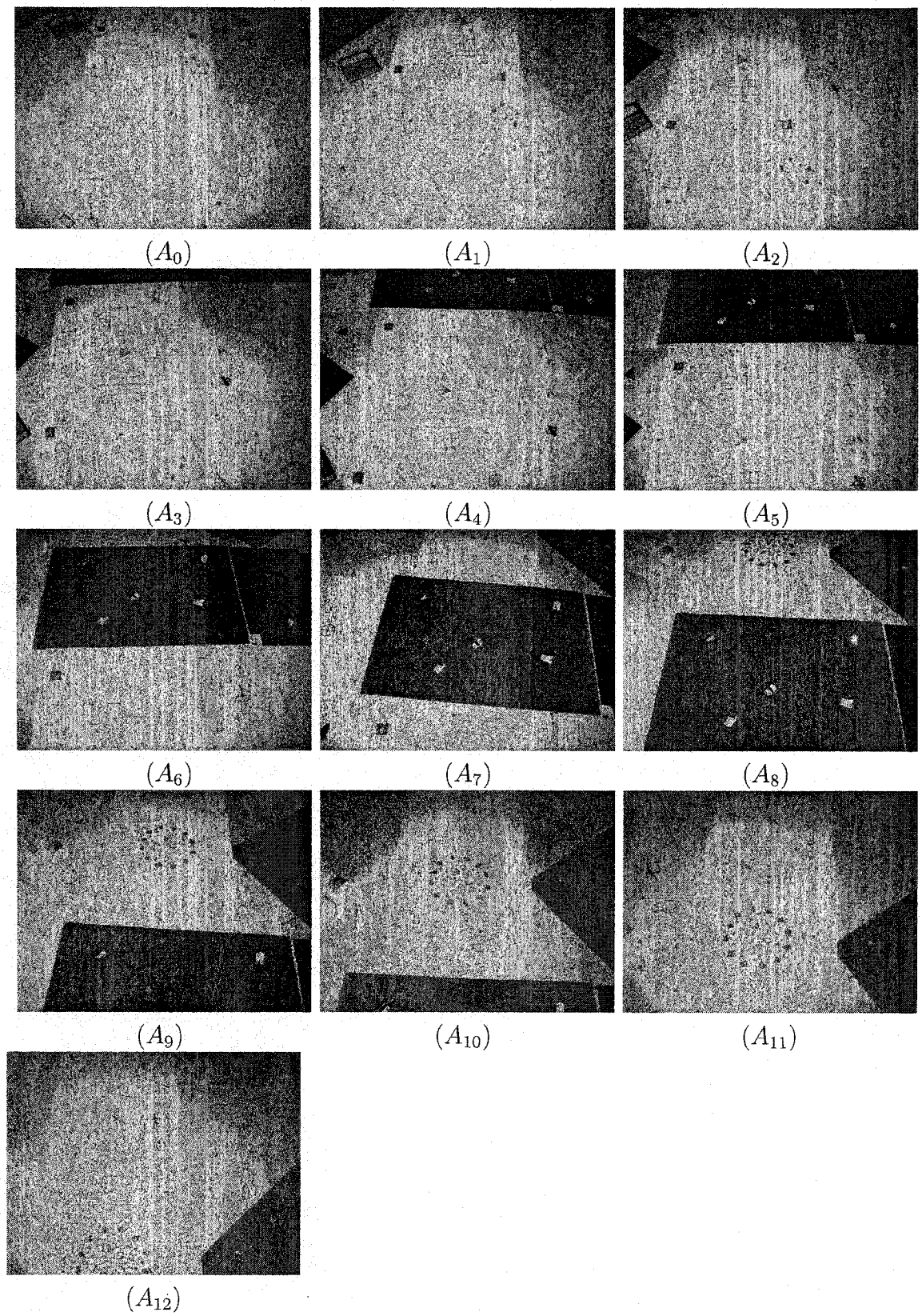


Figure 4.5: Image set collected by the *Robot A*.

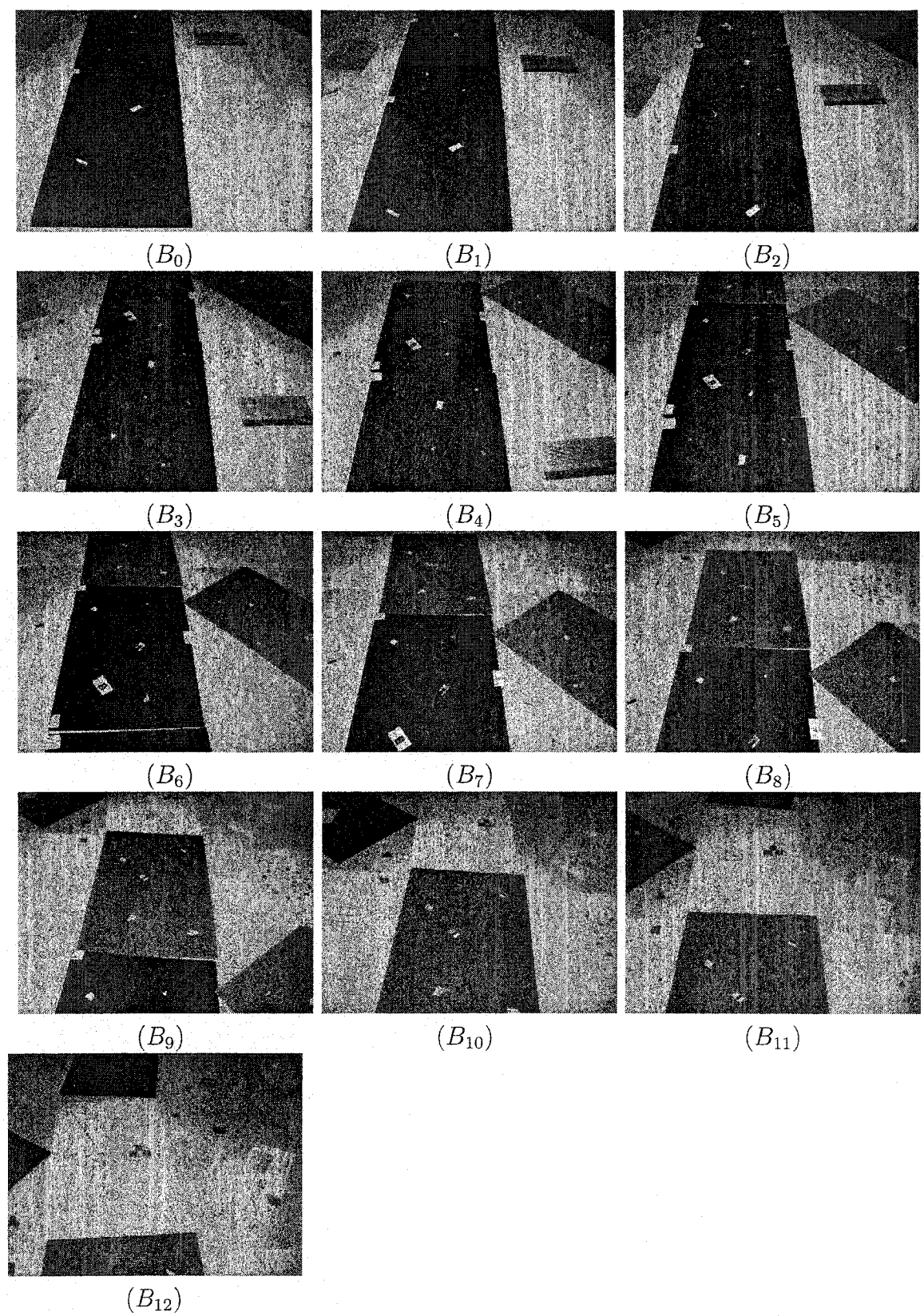


Figure 4.6: Image set collected by the *Robot B*.

4.3.3 Overlapping view detection

In order to determine if two cameras are viewing the same scene, the images have to be compared. One approach is to calculate the inter-image homography as discussed in Chapter 3 and then determine the validity of this transformation. The validity may be checked by mapping the corner points from one image to the other. If there exist a large number of matches then this transformation is considered to be a valid transformation and accepted, otherwise it is rejected.

For two robots, this approach requires calculating the inter-image homography for every possible combination of images. For N images captured by each robot, a complexity of $O(N^2)$ of inter-image homography calculations is required. To improve the efficiency, the homography between any two images is only calculated if there is a potential for match. To do so, we propose a method based on color histogram distance.

The color histograms are commonly used to compare images based on their overall appearance [69]. Because they are computationally efficient, they have been used in many applications like image retrieval [4, 19, 66, 70] and object identification [83]. The histogram $\mathbf{h}$ of an image is a vector consisting of n bins as follows:

$$\mathbf{h} = [h_1, \ldots, h_n] \tag{4.3.2}$$

Each bin $h_i \in \mathbf{h}$, $1 \leq i \leq n$, contains the number of pixels of color i in the image. Two images $\mathbf{I}$ and $\mathbf{I}'$ of histograms $\mathbf{h}$ and $\mathbf{g}$ can be compared based on the inter-bin distance $d(\mathbf{h}, \mathbf{g})$ between their histograms. Images $\mathbf{I}$ and $\mathbf{I}'$ are considered similar if $d(\mathbf{h}, \mathbf{g})$ is minimized. The histogram is trivially computed compared to the inter-image homography. We propose an algorithm to use histogram distance to determine

the most likely overlap and then use the inter-image homography transformation to validate such an overlap. The algorithm is stated as follows:

Let $\mathbf{A} = \{\mathbf{I}_1, \ldots, \mathbf{I}_{N_A}\}$ and $\mathbf{B} = \{\mathbf{I}'_1, \ldots, \mathbf{I}'_{N_B}\}$ be the sets of images collected by *RobotA* and *RobotB* respectively. With set $\mathbf{A}$ contains N_A images and set $\mathbf{B}$ contains N_B images. The problem to solve is to find, using color histogram distance, an image $\mathbf{I}_a \in \mathbf{A}$ and an image $\mathbf{I}'_b \in \mathbf{B}$ such that $\mathbf{I}_a$ and $\mathbf{I}'_b$ are most likely the overlap images if such an overlap exists between the two robots.

Let $Hist(\mathbf{I})$[1] denote the histogram of the overhead view of image $\mathbf{I}$ and C_{Hist} and C_{Homog} denote the cost of computing the histogram distance d and the inter-image homography between two images. The algorithm we propose is summarized as follows:

Step 1: **For** $k = 1$ to N_A
  For $l = 1$ to N_B
    calculate $d(Hist(\mathbf{I}_k), Hist(\mathbf{I}'_l))$
    If d is minimized **then**
      $\mathbf{I}_a = \mathbf{I}_k$ and $\mathbf{I}'_b = \mathbf{I}'_l$
    End
  End
End

Step 2: Calculate the inter-image homography $\mathbf{H}_{ab}$ between images $\mathbf{I}_a$ and $\mathbf{I}'_b$

Step 3: Accept/reject $\mathbf{H}_{ab}$ based on the validation test.

In Step 1, the histogram distances between all the images acquired by the two robots are calculated. The outcome is the two images ($\mathbf{I}_a$ and $\mathbf{I}'_b$) whose histogram distance is the global minimum. In Step 2, the inter-image homography ($\mathbf{H}_{ab}$) is calculated for the two images. Finally, in Step 3, $\mathbf{H}_{ab}$ is validated, if there exist a high number

[1]throughout this chapter all the histograms are calculated using the overhead view images

of matches between corners in image $\mathbf{I}_a$ and corners in image $\mathbf{I}'_b$ then the homography $\mathbf{H}_{ab}$ is accepted.

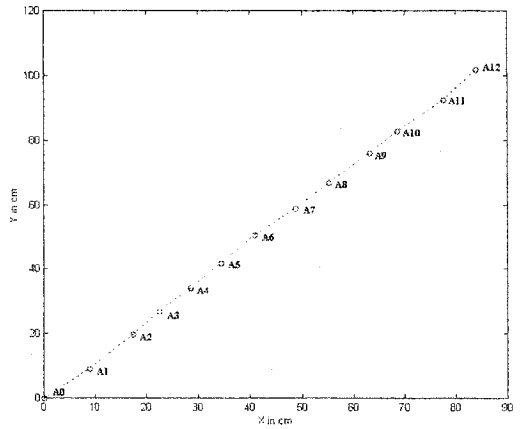
The algorithm requires $O(N_A N_B) \times C_{Hist} + O(1) \times C_{Homog}$. When a new image is collected by one of the robots, this image may produce a minimization on d. Consequently, a computation of the inter-image homography is required. For $N \approx N_A \approx N_B$, the complexity of the algorithm is $O(N^2) \times C_{Hist} + O(N) \times C_{Homog}$.

Images $A0$ and $A7$ in Figure 4.5 and all the images in Figure 4.6 are used to demonstrate our color histogram comparison approach. The images are represented in RGB color space. The total number of bins used is $n = 64$ formed by using the most 2 significant bits of each color channel [102]. First, the histograms of $A0$ and $A7$ are calculated as shown in Figure 4.9(a) and(b) and the histograms of the images in Figure 4.6 are shown in Figure 4.10.

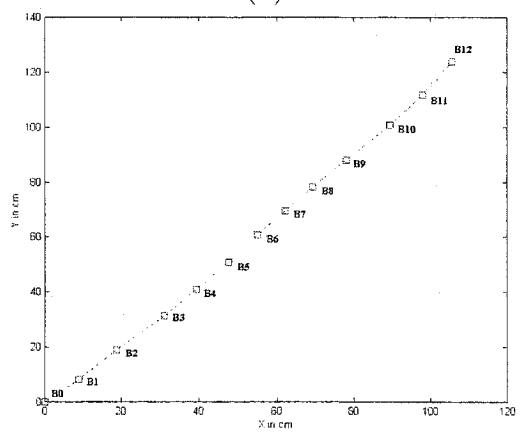
The inter-bin distance between two histograms $\mathbf{h}$ and $\mathbf{g}$ may be expressed in the following form:

$$d(\mathbf{h}, \mathbf{g}) = \frac{\sum_{i=1}^{n} f(h_i, g_i)}{W} \tag{4.3.3}$$

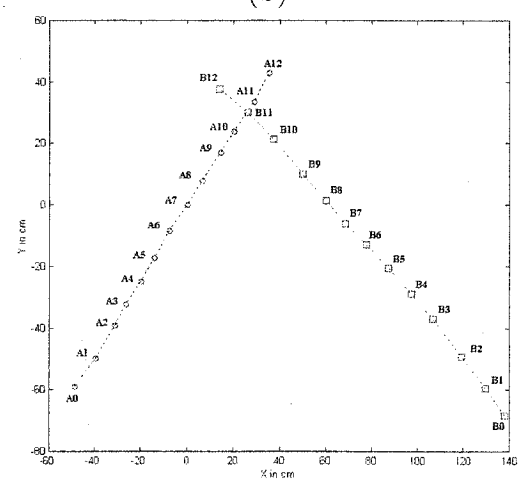
Table 4.3 shows some commonly used distance functions for inter-bin distance calculation [24, 69].



(a)



(b)



(c)

Figure 4.7: Multi-Robot Localization: (a) location graph generated by *RobotA* (b) location graph generated by *RobotB* (c) the joint location graph

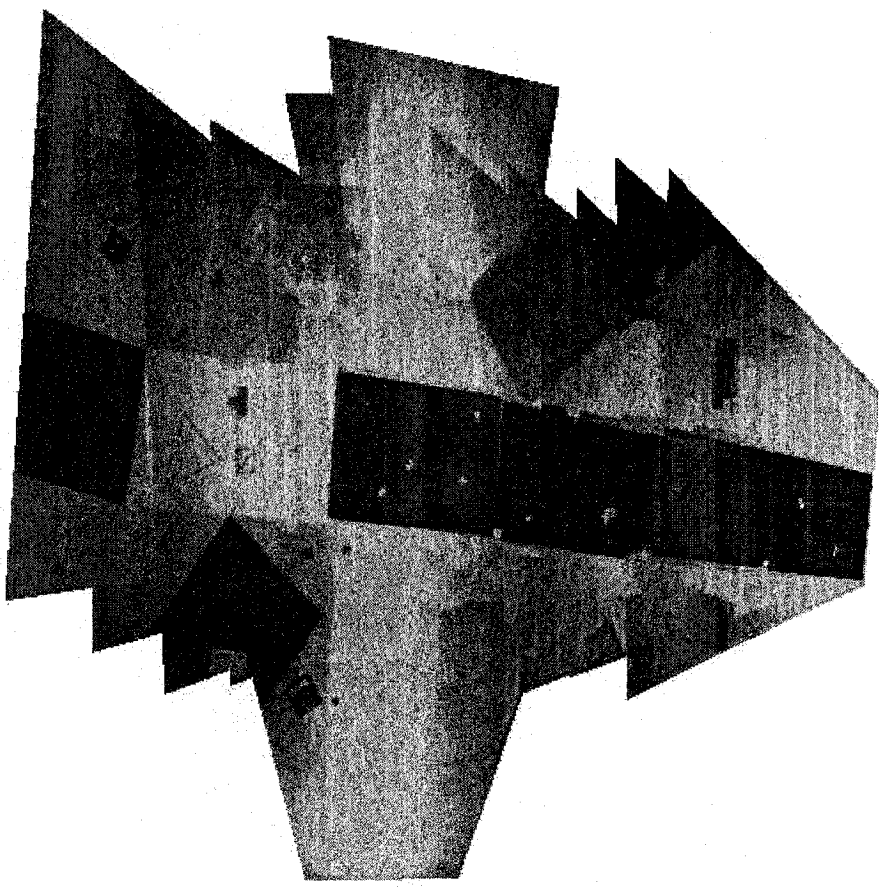


Figure 4.8: The map generated from images collected by *Robot A* and *Robot B*.

Distance type	$f(h_i, g_i)$	W	
Absolute difference	$\lvert h_i - g_i \rvert$	1	
Sum of squared differences	$(h_i - g_i)^2$	1	
Intersection	$g_i - \min(h_i, g_i)$	$\sum_{i=1}^{n} g_i$	
χ^2 distance	$\frac{(h_i - m_i)^2}{m_i}$	1	where $m_i = \frac{h_i - g_i}{2}$

Table 4.3: Some commonly used histogram distance functions.

The inter-bin distances defined in the table are used to calculate the distances between $A0$ and $B0$ to $B12$ and similarly between $A7$ and $B0$ to $B12$. The results are shown in Figure 4.11. Figure 4.11 shows that the graph of $A7$ is minimized compared to the graph of $A0$ in all distance types and the minimum distance is at image $B8$ in Figures 4.11(a), (b) and (c) and at $B8$ and $B10$ in Figures 4.11(d). This shows that there is most likely an overlap between $A7$ and $B8$ or between $A7$ and $B9$, by looking at Figures 4.5 and 4.6, it is simple to verify that there is indeed an overlap between $A7$ and $B8$ and $B9$. Since the graph of $A7$ is minimized in all distance functions at $B8$, the inter-image homography is calculated between images $A7$ and $B8$.

One important issue to note about histogram distances is that two images of two different scenes might have similar color histograms [69]. In this case choosing only the global minimum of the histogram distance might not correspond to a correct overlapping scene. To avoid this, we propose two conditions to select images for overlapping detection. The histogram distance must be below a certain threshold and must be a local minimum. Based on this, the algorithm is updated to be:

For $k = 1$ to N_A

Step 1: **For** $l = 1$ to N_B

calculate $d(Hist(\mathbf{I}_k), Hist(\mathbf{I}'_l))$

If d is minimized **then**

$\mathbf{I}_a = \mathbf{I}_k$ and $\mathbf{I}'_b = \mathbf{I}'_l$

End

End

Step 2: **IF** $d <$ Threshold

- Calculate the inter-image homography $\mathbf{H}_{ab}$ between images $\mathbf{I}_a$ and $\mathbf{I}'_b$
- Accept/reject $\mathbf{H}_{ab}$ based on the validation test.

End

End

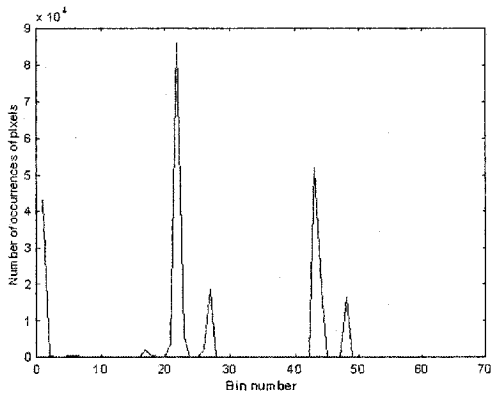
The outcome of Step 1 in the algorithm is the local minimum of the histogram distance between one image of *RobotA* and all the images of *RobotB*. In Step 2, if the histogram distance is below certain threshold, the inter-image homography is calculated and checked for validity. As indicated by the outer loop, steps 1 and 2 are repeated for all images of *RobotA*.

In this algorithm, the worst case scenario is when a new image collected by one of the robots produces a local minimum below a certain threshold with all the images collected by the other robot. Consequently, an $O(N)$ computation of the inter-image homography is required. The complexity of the algorithm is $O(N^2) \times C_{Hist} + O(N^2) \times C_{Homog}$. However, in practice this scenario is unlikely to happen.

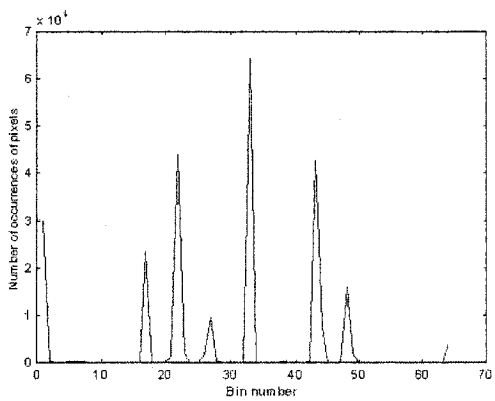
In our algorithm any one of the histogram distances presented in Table 4.3 can be used. To demonstrate the validity of the algorithm, consider the intersection distances between all images collected by *Robot A* and *Robot B* as shown in Figure 4.12. In particular consider the graphs for images A4, A5, A7, A9 and A10. The global

minimum in our case is at point a which indicates that there is potential overlapping between images $A7$ and $B8$ which can be verified to be correct. For a threshold of 0.2, the local minimums of images A4, A5, A9 and A10 are at points b, d, c and e respectively. The overlapping indicated by these local minimums can be verified to be correct by looking at images in Figures 4.5 and 4.6.

The local minimum for each graph in Figure 4.12 represents a valid overlapping scenes except for the cases of A0 and A1 since there is no overlap with any image collected by Robot B.



(A_0)



(B_0)

Figure 4.9: Histogram of images A_0 and A_7 using the overhead views

4.4 Summary

In this chapter I have presented a technique to solve the multi-robot SLAM problem. My main contribution is that I use only vision-based sensors to experimentally solve the SLAM problem without using other types of sensors like odometers, laser scan, ultrasound or others. Moreover, I assume no a priori known environment map, scene landmarks, initial robot pose or preliminary training phase.

In my proposed method, the different robot locations are computed by finding the transformations that relate together the captured images of the scene. The camera motions are estimated from the computed inter-image homographies and each robot is localized with respect to a local map. To build a global map, the inter-image homographies have to be calculated between different robots in the team. To do so, I presented an overlapping view detection technique. The technique is based on color histogram distances between images collected by the team of robots. When two images from two robots have similar color histograms, a potential overlapping between the two robots is assumed. The inter-image homography is calculated to verify if this overlapping is valid. If such an overlapping exists, then it is possible to build a common map of the environment inside which the robots are evolving.

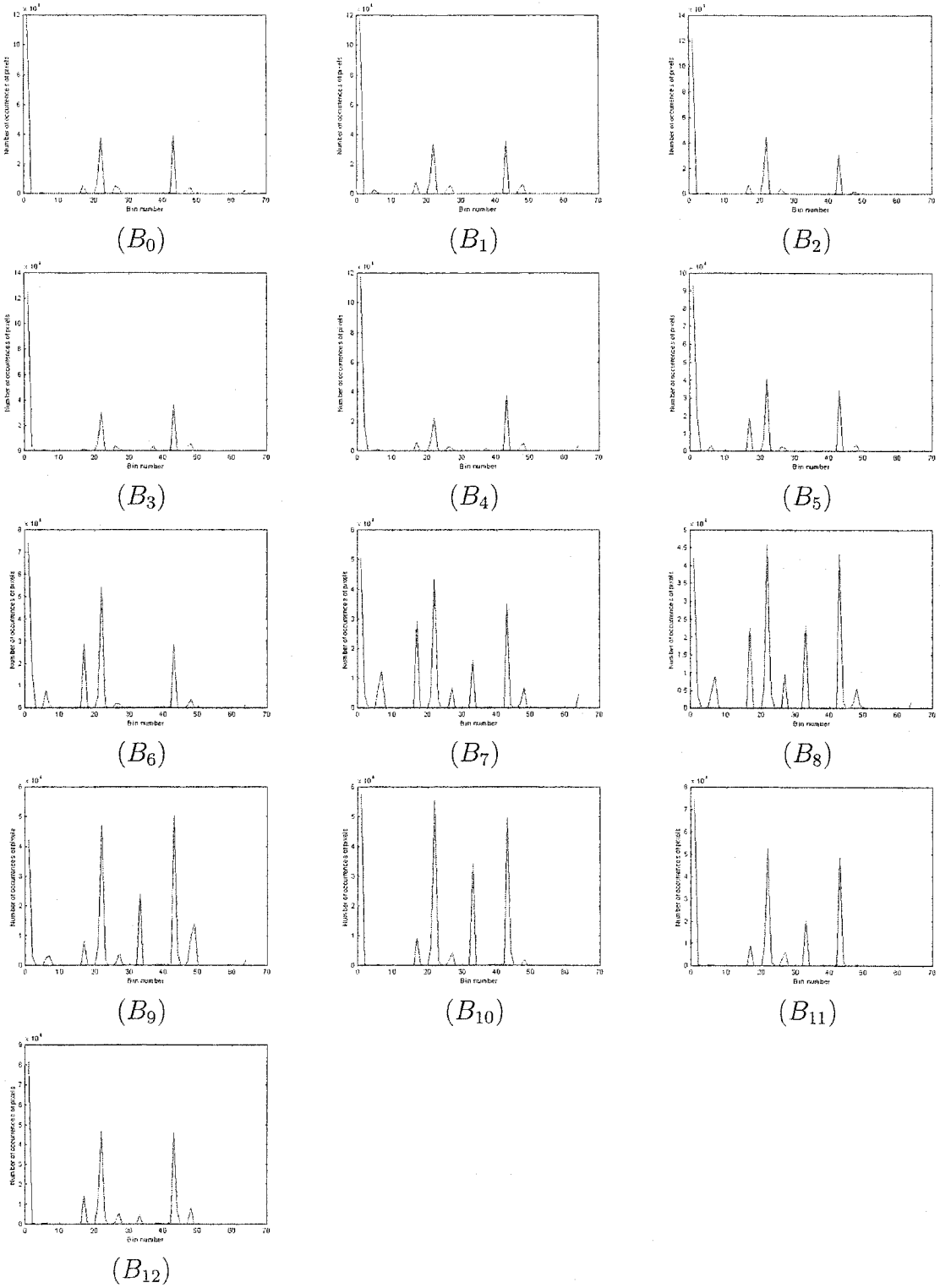


Figure 4.10: Histogram of images collected by the *Robot B* using the overhead views.

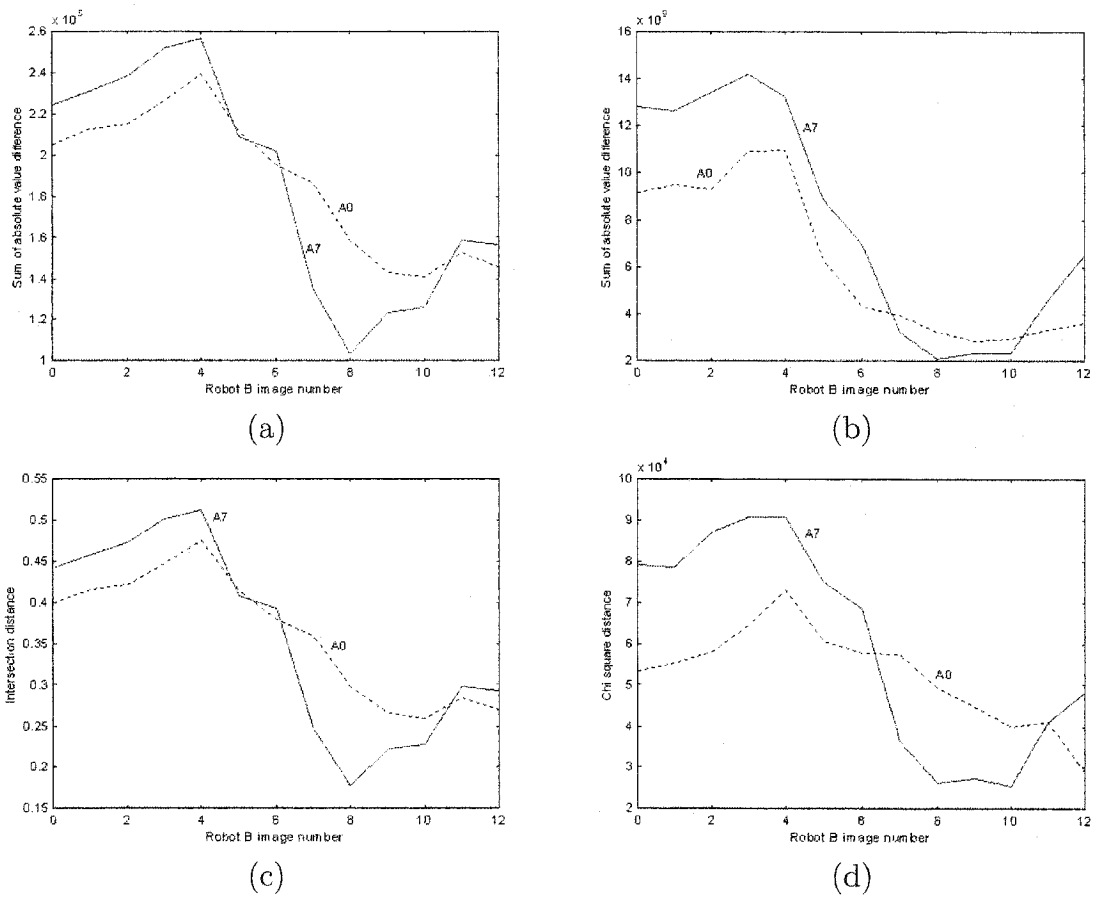
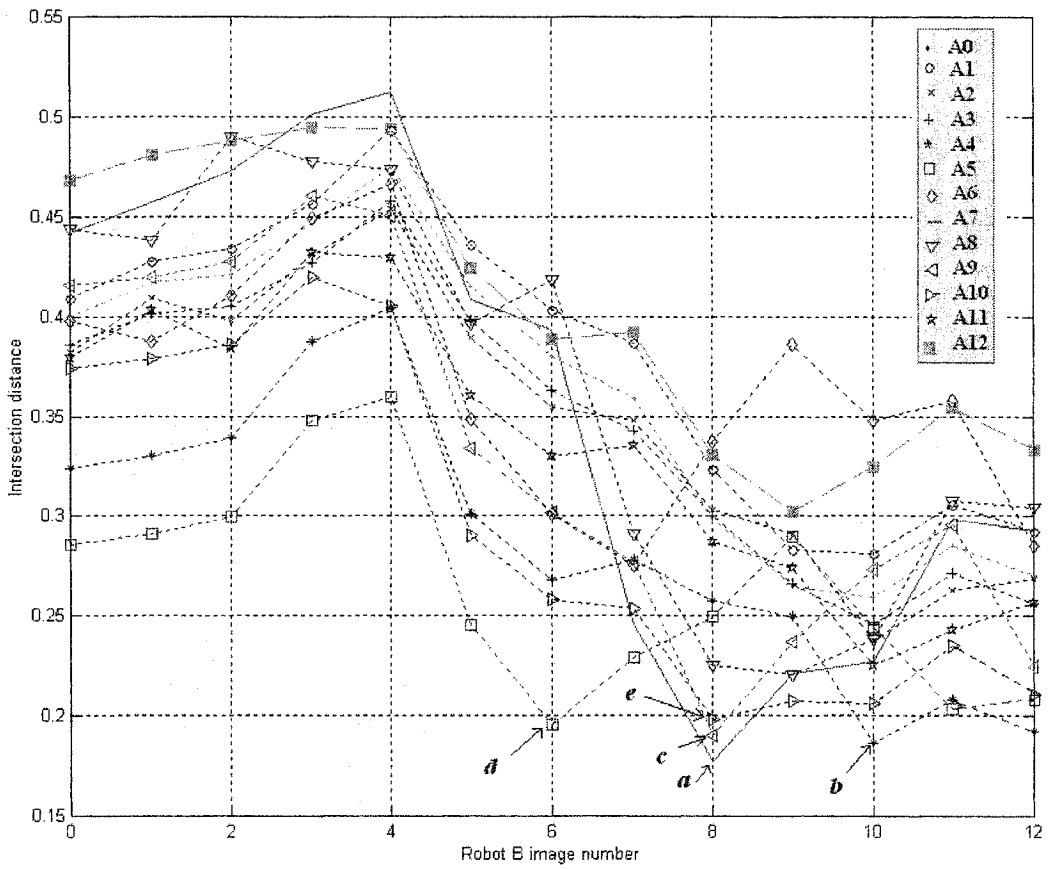


Figure 4.11: Histogram distance between images $A0$ and $A7$ and the images collected *Robot B*. (a) Absolute difference. (b) Sum of squared differences. (c) Intersection distance. (d) χ^2 distance



(c)

Figure 4.12: Histogram distance between images $A0$ to $A12$ and $B0$ to $B12$ using Intersection distance.

Chapter 5

Ground plane 3D obstacle reconstruction

This chapter introduces a geometric/level set method to reconstruct the ground plane obstacles for robots moving on a planar ground. The method integrates a voxel coloring formulation and shape-from-silhouette concepts in the energy functional to minimize. The level set surface evolution technique is used to minimize the energy functional and thus recover the 3D structure of the obstacle.

5.1 Introduction

In Chapters 3 and 4 algorithms have been presented to detect the ground plane, estimate the camera motion, localize robots and detect obstacles using a collection of sparse views of a scene as captured by a robotic vehicle equipped with a single camera. In this chapter a further step is taken to understand the 3D structure of the explored robot environment. To do so, a 3D reconstruction method is introduced. Note that,

in the application proposed here, the reconstruction of the scene is meant to be done **offline**. The robot would simply use the obstacle localization information to safely collect data about the scene of interest. This can be achieved by using the computed homography that warps one image with respect to the other one. Subtracting the warped image from the other one and applying a threshold to the absolute difference gives a crude segmentation that identifies regions where an obstacle is visible [50]. The thus collected images would be transmitted to a base station that would be in charge of computing the 3D scene model.

5.2 Overview of 3D scene reconstruction

In this section an overview of different 3D object reconstruction algorithms is presented. The methods are structure from motion, shape-from-silhouette, voxel-coloring, evidence grids and variational methods.

5.2.1 Structure from motion

The structure from motion approach (SFM) refers to the estimation of the relative camera pose and the 3D reconstruction of a scene from at least two images taken from different view points.

Since the seminal work of Longuet-Higgins [49] many SFM algorithms have been introduced [92, 15, 31, 50]. The main idea of the SFM algorithms is to minimize an objective function which represent the epipolar geometry constraint and recover the motion between views. The Structure From Motion method starts by detecting some

image features like corners [31] or edges [5] in the two images. Then the *correspondence problem* [55] is solved by matching features that correspond to the projection of the same scene structure. The matching is usually based on cross-correlating image intensities [104] between views. The set of matched image features are used to compute an initial estimate of the epipolar geometry between the views. The constraint induced on the views by the initial estimate of the epipolar geometry is used in a guided feature matching step to refine the set of matched features. The epipolar geometry is calculated from the refined set of matched features and finally the motion between the views and the camera matrices are estimated. The 3D structure of the corresponding features can be obtained by using the projection matrices and applying triangulation [91] from at least two views of the scene as shown in Figure 5.1.

This approach can be applied on scenes that contain distinct features such as corners, lines or edges. However, for scenes with featureless objects other method like shape from silhouette are more appropriate.

5.2.2 Shape from silhouette

The shape from silhouette approach is applied efficiently on smooth textureless objects. The silhouette (outline) is the projection of points on the surface of the object at which the visual ray from the camera center is orthogonal to the surface normal. In general, the silhouettes of different views of a smooth object represent the projection of different curves in the object's 3D space. Thus silhouettes are view dependent in contrast to features, like corners [31], that are view independent. View dependence leads to the development of algorithms that do not assume rigidity of the scene unlike the SFM algorithms.

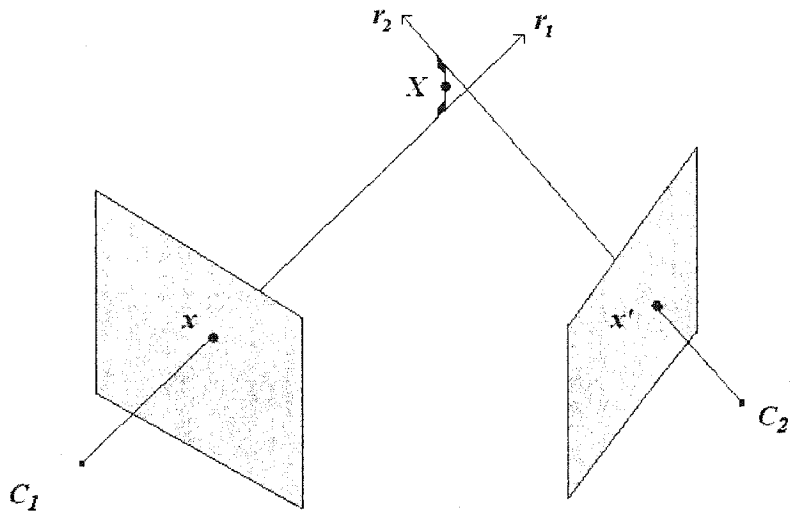


Figure 5.1: Reconstruction by Triangulation: two corresponding image points $\mathbf{x}$ and $\mathbf{x}'$ define two rays $r_1 = \vec{C_1\mathbf{x}}$ and $r_2 = \vec{C_2\mathbf{x}'}$. The equation of the rays r_1 and r_2 are known and the intersection can be computed to recover the 3D world point $\mathbf{X}$, in practice the rays will not intersect, thus the 3D point $\mathbf{X}$ can be estimated as the minimum distance between the two rays.

With the *shape-from-silhouette* strategy [8, 108], each view is segmented into silhouette and background points. The union of all visual rays emanating from all views and intersecting the image silhouette defines a generalized cone within which the 3D object must lie. If a sufficient number of views is used, the *visual hull* of the observed 3D structure is obtained [46]. However, as indicated by Laurentini [46], concavities in the object cannot be removed and the *visual hull* is an approximation of the true 3D structure of the reconstructed object. Figure 5.2 shows the reconstruction of a 2D elliptic object using four silhouettes. The generalized cones associated with the four images result in reconstruction represented by the dark area. By adding a sufficient number of images, the elliptic object will be reconstructed, however the concavity (the hatched area) cannot be reconstructed regardless of the number of images used.

The visual hull of a 2D scene equals the convex hull; however, for 3D scenes the visual hull is included in the convex hull.

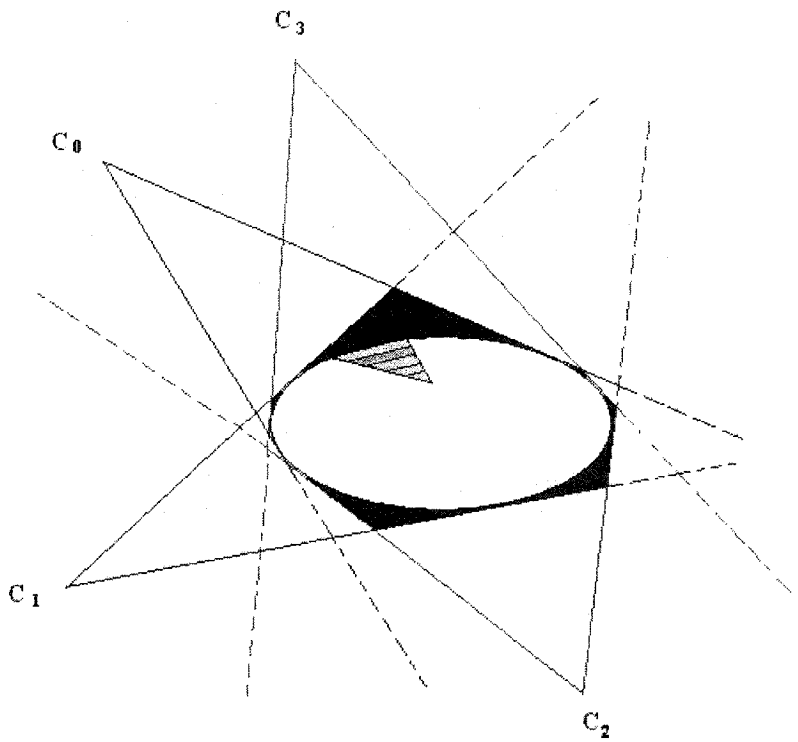
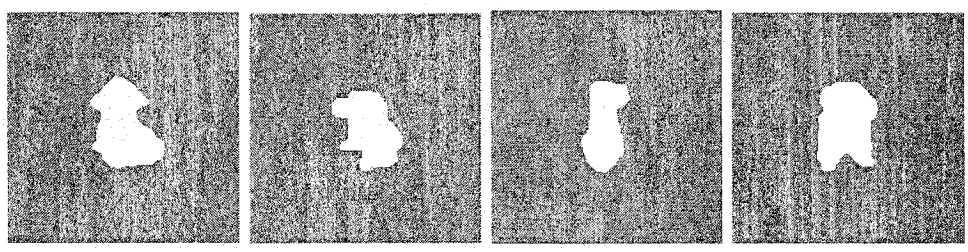


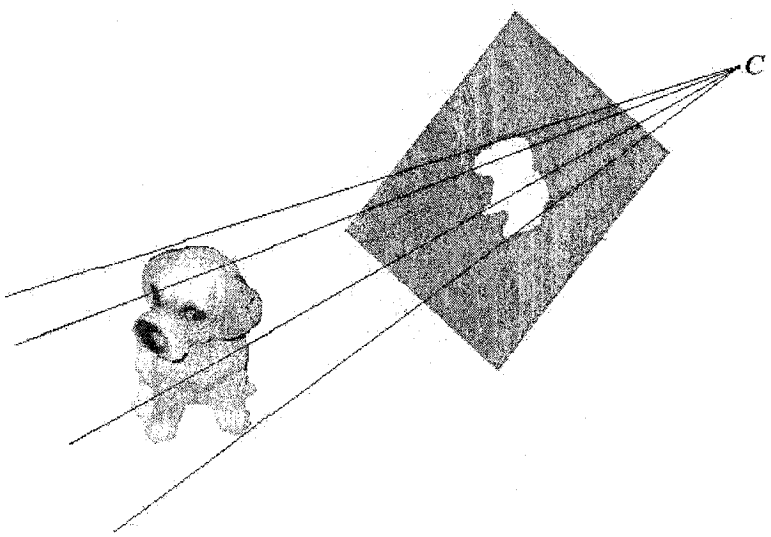
Figure 5.2: Reconstruction using silhouettes: The reconstruction of the 2D elliptic object using the four silhouettes is indicated by the dark area. Adding sufficient images, the elliptic object will be reconstructed. However, the concavity (hatched area) cannot be reconstructed regardless of the number of images used.

The silhouette image is represented as a binary image, with pixels classified as silhouette points or background points. Figure 5.3(a) shows some image silhouettes, Figure 5.3(b) shows the conic volume. To model the 3D structure of objects in the scene, the scene is first discretized into voxels in 3D space and then the 3D structure of the scene is modeled by the voxel occupancy description corresponding to the intersection of the conic volumes. The voxels are modeled using octree representation

[9, 71, 85]. The 3D scene is initially represented by a coarse volume of voxels and then each voxel is projected into all the input images and then tested if it belongs to the silhouettes. The voxel is processed based on three cases. First if the voxel belongs to all silhouettes, then the voxel is marked as opaque (i.e belongs to the object visual hull). If it does not intersect any silhouette, then the voxel is marked as transparent (i.e does not belong to the object visual hull). Finally if the voxel belongs to at least one silhouette, the voxel is subdivided into octants and the algorithm is applied recursively on the resulting sub-voxels.



(a)



(b)

Figure 5.3: Shape from Silhouettes: (a) Image silhouettes (b) the conic volume.

5.2.3 Voxel coloring

While the shape-from-silhouette methods use only the binary images of the outline of the objects, the voxel-coloring methods make use of the image pixel colors. The photometric information captured by the set of input images can be used to recover the 3D structure of the scene. The *photo-consistency* [75] can be used to distinguish points on a surface of an object from other points on the scene. A point of a scene is said to be photo-consistent if the image pixel intensity, from each image in which this point is visible, equals the image irradiance of that point. Figure 5.4 shows two 3D points p_1 and p_2 visible from two cameras C_1 and C_2. Point p_1 is located on the surface of an object, its projections on the two cameras at a_1 and a_2 have consistent pixel colors, thus point p_1 is said to be photo-consistent with the two images. Point p_2 lies above the ground plane, its projections on the two cameras at b_1 and b_2 have inconsistent pixel colors and p_2 is not photo-consistent.

The photometric information contained in each view can be used in the 3D reconstruction process by using the *voxel-coloring* strategy introduced by Seitz and Dyer [75]. The algorithm begins by dividing the scene space into small volumetric elements (the voxels) as shown in Figure 5.5. The scene is then visited and each initially opaque voxel is projected onto the input images and examined to determine if it belongs to an object surface. This is done by measuring the *photo-consistency* [75] constraint between the set of input images. The voxels that are found to be inconsistent are made transparent. The algorithm terminates when all the remaining opaque voxels are photo-consistent. These voxels form a model that describes the 3D structure of the scene. The visibility of each voxel must be determined before performing color consistency. For a given voxel, the transparency of all the voxels that occlude it must

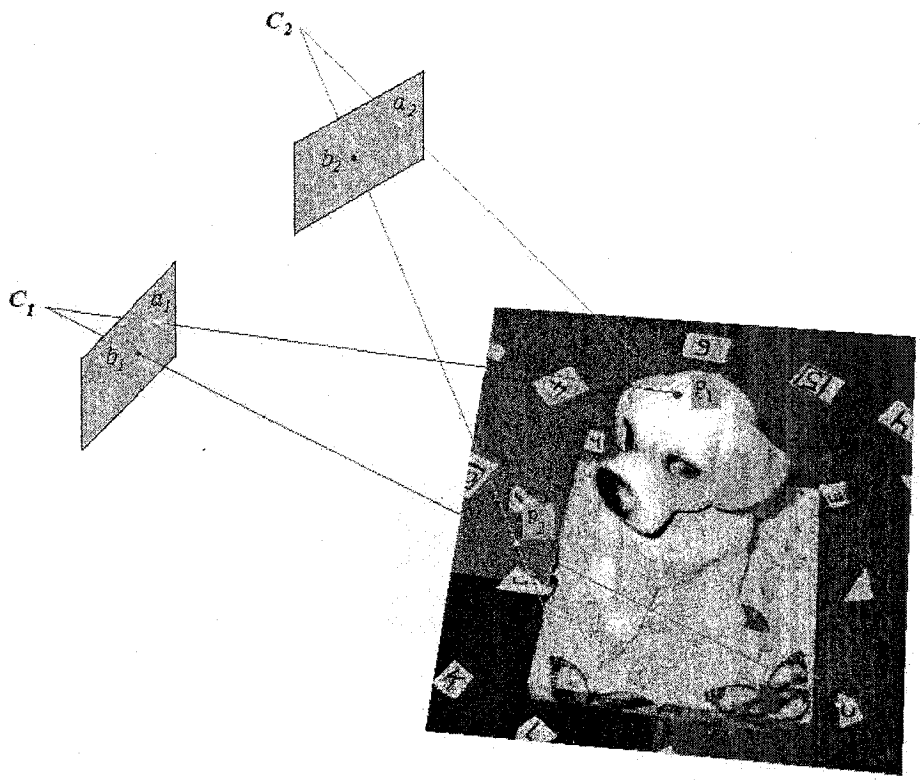


Figure 5.4: Color consistency can be used to distinguish points on a scene. Two 3D points p_1 and p_2 are visible from two cameras C_1 and C_2. Point p_1 is located on the surface of the object, the two cameras see consistent colors on image points a_1 and a_2. Point p_2 is located above the ground and the two cameras see inconsistent colors at points b_1 and b_2.

be determined first. If a given voxel is not occluded by an opaque voxel, then this given voxel is considered to be visible. Seitz and Dyer [75] introduced the *ordinal visibility* constraint on the camera locations to simplify the voxel visibility computation. The voxels are visited in a near to far order relative to all the cameras. This can be achieved by placing the cameras on one side of the scene and then visit the voxels in planes at increasing distances from the cameras. For arbitrary camera placement, the voxels are visited in planes at increasing distances from the convex hull of the cameras.

Prock and Dyer [72] introduced a method to enhance the performance of the voxel-coloring approach [75]. The approach is based on a coarse-to-fine voxels strategy. Coarse-resolution voxels are used to represent the interior of large objects and the free space areas in the scene and fine voxels are used to represent objects surfaces. First, the scene is represented using coarse voxels and the voxel-coloring approach is applied. The resulting opaque voxels are then subdivided into eight smaller ones and another pass of voxel-coloring is executed. The algorithm is repeated until a certain defined resolution is achieved.

Kutulakos and Seitz [43] introduced an approach called *space carving* to reconstruct scenes using arbitrary camera placement. The approach is similar to the voxel-coloring approach, planes of voxels are scanned and evaluated for color consistency. The *ordinal visibility* of the scanned plane of voxels is maintained by sweeping a plane through the cameras and the images that are passed by the plane are used in the computation. To ensure photo-consistency of voxels with all input views, multiple sweeps are needed. Six passes on each voxel is used by sweeping a plane through the positive and negative directions of the 3D voxel coordinate system. As the algorithm runs, the voxels that are found to be inconsistent are made transparent (i.e are *carved*). When the algorithm terminates, the remaining set of opaque voxels form a model of the scene which is a superset of any other photo consistent model. This model is called the *photo hull.*

Both the SFM and the voxel-coloring methods require the projection matrices for all cameras to be known. However, for the voxel-coloring approach solving the correspondence problem for each 3D voxel is not required; instead each voxel is projected on all the images and the photo-consistency is calculated. On the other hand, solving

the correspondence problem for the SFM is inevitable and it is challenging especially for smooth objects with constant radiance.

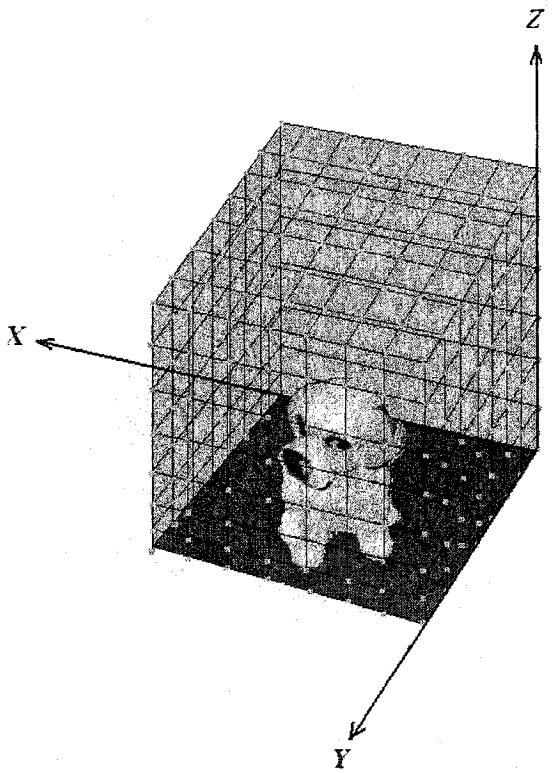


Figure 5.5: Voxel coloring: The scene is divided into voxels.

5.2.4 Evidence grids

The evidence grids approach, proposed by Moravec [62, 56], is used to represent regions of robot's surrounding using stereoscopic images. It integrates concepts from both voxel coloring and Structure From Motion (SFM) methods.

The 3D space is modeled as a three-dimensional array of cells (voxels) called the *evidence grid.* Each grid cell accumulates evidence about its occupancy. The cells

are initialized to zero to indicate no evidence about the occupancy. After the reconstruction process, the grid cells are updated and the positive cells indicate occupied regions and the negative cells indicate free space. The approach [62] starts by collecting images using a calibrated stereo camera. First, features are detected in the two images and then correlated to obtain a set of corresponding features [55]. The 3D structure of each corresponding feature is calculated using triangulation [91] as discussed in the SFM approach (see Section 5.2.1). The cells that lie along the rays joining the calculated 3D point and the two cameras are assigned negative values (i.e free space) and a positive value (i.e occupied) on the 3D point. The final 3D structure of the scene is described by the occupied cells in the grid. To obtain a photo realistic model of the environment similar to the model obtained using the voxel coloring approach (see Section 5.2.3), the occupied cells are colored by back-projecting the original images.

5.2.5 Variational methods

Traditional 3D reconstruction algorithms for the stereo problem is to solve the *correspondence* problem first, then solve the *reconstruction* problem for each corresponding pair of image points using triangulation as discussed in Section 5.2.1. In their pioneer work, Faugeras and Keriven [16, 67] introduced a variational approach without separating the two problems. The method is based on 3D reconstruction in the context of a multi-view stereo problem, i.e. when several cameras of known perspective projection matrices are observing the same objects. Their work proposed functionals for matching to control the evolution of the surface. The functionals depends on cross-correlation of image intensities across pairs of views. To demonstrate the

idea of the algorithm, consider the point $\mathbf{M}$ on Figure 5.6. This point belongs to a surface S to be reconstructed and is projected onto two images $\mathbf{I}_1$ and $\mathbf{I}_2$ on points $\mathbf{m}_1$ and $\mathbf{m}_2$ respectively. The object S is represented in the neighborhood of point $\mathbf{M}$ by its tangent plane. This tangent plane induces a homography transformation $\mathbf{H}$ between points $\mathbf{m}_1$ and $\mathbf{m}_2$. Thus, a rectangular window centered at point $\mathbf{m}_1$ in image $\mathbf{I}_1$ is mapped to a window, not necessarily rectangular, centered at $\mathbf{m}_2$ in image $\mathbf{I}_2$. The cross-correlation between the two windows can be used to measure the similarity between image points $\mathbf{m}_1$ and $\mathbf{m}_2$.

Notice that the homography $\mathbf{H}$ is a function of the normal $\mathbf{n}$ and the point $\mathbf{M}$. Based on this, an error criterion can be stated as follows:
Given a point $\mathbf{S}$ on S with normal vector $\mathbf{n}$ and projected at $\mathbf{m}_1$ in $\mathbf{I}_1$, the correlation between images $\mathbf{I}_1$ and $\mathbf{I}_2$ at point $\mathbf{m}_1$ is defined to be:

$$\phi(\mathbf{S}, \mathbf{n}, \mathbf{m}_1) = \frac{< \mathbf{I}_1, \mathbf{I}_2 >}{|\mathbf{I}_1|.|\mathbf{I}_2|} \tag{5.2.1}$$

where $< \mathbf{I}_1, \mathbf{I}_2 >$ denotes correlation between $\mathbf{I}_1$ and $\mathbf{I}_2$. An error criterion for any point on the surface S can be written based on Equation (5.2.1) as follows:

$$C(\mathbf{S}, \mathbf{n}) = \int_S \phi(\mathbf{S}, \mathbf{n}) d\sigma \tag{5.2.2}$$

The integration over S is carried out with respect to the area element $d\sigma$.
Finally, the Euler-Lagrange equation of the variational problem in Equation (5.2.2) is written and the problem is numerically solved using the level sets methods [77]. An overview of the level sets methods is introduced in Section 5.3.

Yezzi and Soatto [101] introduced a 3D reconstruction algorithm from a collection of images where the scene under observation is made of smooth surfaces with constant

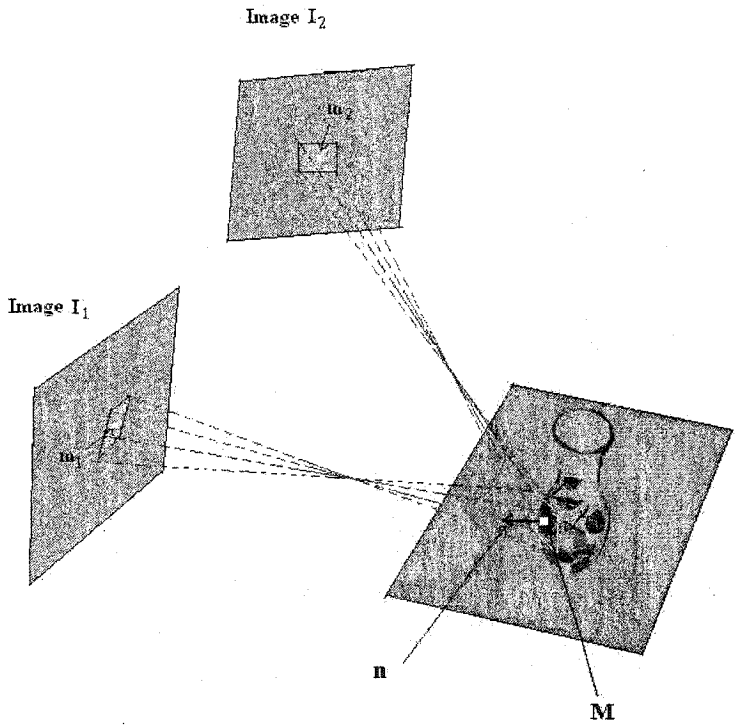


Figure 5.6: The square window on the first image is mapped to a window, not necessarily square, in the second image using the homography induced by the tangent plane to the surface S at point $\mathbf{M}$.

radiance. The algorithm starts with a rough estimate of the relative camera pose then the surface shape and motion parameters are simultaneously estimated. The method is based on a cost functional that minimizes the discrepancy between the projection of the 3D model onto the images and the measured projections captured by the collection of images.

The scene background and the object surface are assumed to have different radiance functions $\mathbf{h}$ and $\mathbf{f}$ respectively. This is an important assumption for the segmentation process during the surface reconstruction and pose estimation. The cost

functional of the algorithm is a function of radiance functions $\mathbf{f}$ and $\mathbf{h}$, surface S and camera pose g. For constant radiance functions $\mathbf{f}$ and $\mathbf{h}$ the cost function can be described as follows:

$$C(\mathbf{f}, \mathbf{h}, S, g) = C_{data}(\mathbf{f}, \mathbf{h}, S, g) + C_{geom}(S) \tag{5.2.3}$$

where C_{data} describes the discrepancy between measured images and the images predicted by the model and C_{geom} describes the smoothness of the surface. The Euler-Lagrange equations are derived for Equation (5.2.3) and then implemented using level set methods (see Section 5.3).

Sekkati and Mitiche [76] introduced a variational approach for recovering relative depth and 3D motion from a temporal sequence of images. The relation between the velocity of a point $\mathbf{P} = [X, Y, Z]^T$ in the 3D Euclidean space and the velocity of its image $\mathbf{p} = [x, y, f]^T$ in the image plane can be described by the basic equation of motion as follows [34, 91]:

$$\begin{cases} v_x &= \frac{t_z x - t_x f}{Z} - w_y f + w_z y + \frac{w_x xy}{f} - \frac{w_y x^2}{f} \\ v_y &= \frac{t_z y - t_y f}{Z} + w_x f - w_z x - \frac{w_y xy}{f} + \frac{w_x y^2}{f} \end{cases} \tag{5.2.4}$$

where $\mathbf{t} = [t_x, t_y, t_z]^T$ is the translational component of motion and $w = [w_x, w_y, w_z]^T$ is the angular velocity. Under the assumption that the brightness from a point on a surface in the scene does not change during motion, the following gradient equation is satisfied:

$$E_t + \dot{x}E_x + \dot{y}E_y = 0 \tag{5.2.5}$$

where E_x, E_y and E_t are the spatio-temporal derivatives of the image brightness E. A rigid body motion constraint equation can be obtained based on Equations (5.2.4) and (5.2.5) as follows:

$$E_t + \mathbf{s}.\frac{\mathbf{t}}{Z} + \mathbf{q}.w = 0 \tag{5.2.6}$$

where $\mathbf{s}$ and $\mathbf{q}$ are both functions of x, y, f, E_x and E_y. The 3D motion of the image sequence can be recovered by minimizing the following cost functional:

$$C(\mathbf{t}, w, Z) = C_{motion} + C_{reg} \tag{5.2.7}$$

where C_{motion} represents the conformity to the rigid body motion in Equation(5.2.6) and C_{reg} is a regularization term to preserve the boundaries of the 3D interpretation.

Finally, as in [16, 67] and [101], the Euler-Lagrange equations for Equation (5.2.7) are derived and implemented using level set methods.

5.3 Level set methods

The level set methods introduced by Osher and Sethian[77, 68, 67] has been applied successfully to solve many problems in mechanics, computer vision and other areas. The main idea of level set methods is to analyze and compute the subsequent motion of an interface Γ bounding an open region Ω. The motion of Γ is driven by a velocity field $\mathbf{v}$ that may depend on position, time, geometry of the interface or external factors. At each time step the interface $\Gamma(t)$ can be explicitly defined as the zero level of a smooth function $\phi(\mathbf{x}, t)$, where $\mathbf{x} \in \mathbb{R}^n$ and t represents time. Function $\phi(\mathbf{x}, t)$ has the following properties:

- $\phi(\mathbf{x}, t) < 0 \ \mathbf{x} \in \Omega$
- $\phi(\mathbf{x}, t) > 0 \ \mathbf{x} \in \overline{\Omega}$
- $\phi(\mathbf{x}, t) = 0 \ \mathbf{x} \in \partial\Omega = \Gamma(t)$

The implicit function $\phi(\mathbf{x}, t)$ evolves by convecting the levels of ϕ with a velocity field $\mathbf{v}$, the convection equation is as follows:

$$\frac{\partial \phi}{\partial t} + \mathbf{v}.\nabla\phi = 0 \tag{5.3.1}$$

When the velocity field $\mathbf{v}$ has component in the normal direction only, then $\mathbf{v} = v_n\mathbf{n}$ can be substituted in Equation(5.3.1) as follows:

$$\frac{\partial \phi}{\partial t} + v_n\mathbf{n}.\nabla\phi = 0 \tag{5.3.2}$$

where v_n is the component of the velocity in the normal direction and $\mathbf{n}$ is the unit normal to the interface defined to be $\mathbf{n} = \frac{\nabla\phi}{|\nabla\phi|}$. By substituting $\mathbf{n}$ in Equation(5.3.2), the convection equation becomes:

$$\frac{\partial \phi}{\partial t} + v_n \mid \nabla\phi \mid = 0 \tag{5.3.3}$$

The domain where ϕ is defined is divided into a discrete set of data points called a *grid*. In 2D space it is convenient to divide the space into an $m \times n$ uniform Cartesian grid as shown in Figure 5.7. The Cartesian grid is defined as follows:

$\{(x_i, y_j)/1 \leq i \leq m, 1 \leq j \leq n\}$

The subintervals $\Delta x = x_{i+1} - x_i$ along the x-direction and $\Delta y = y_{j+1} - y_j$ along the y-direction are of equal sizes. Once the domain has been discretized, the values of the implicit function can be stored at each point in the domain. The value of the function ϕ at grid point (x_i, y_j) is denoted by ϕ_{ij}.

Many practical problems can be modeled as a minimization to some energy functional. The variational formulation of the functional can be expressed as shown in Equation (5.3.1). The evolution of the interface is driven by some velocity function

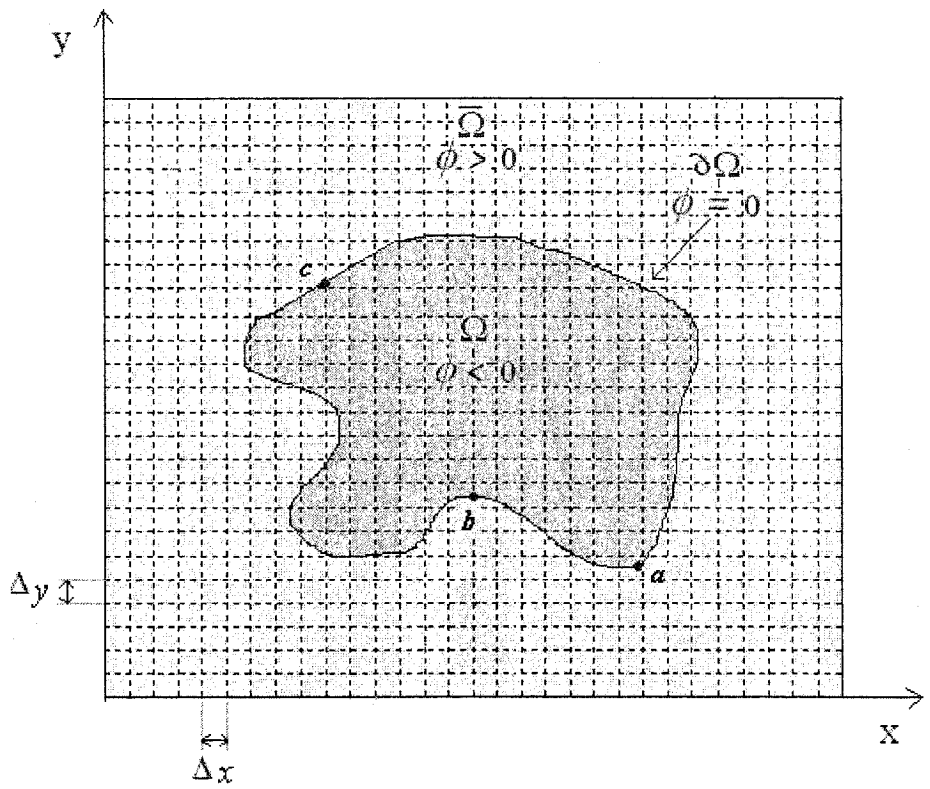


Figure 5.7: Implicit interface representation, $\phi > 0$ outside, $\phi < 0$ inside and $\phi = 0$ on the interface.

$\mathbf{v}$, when the surface is far away from the solution, the speed is high and the zero level set is set to move through free space towards the solution. When the zero level set is close to the solution, the speed slows down and at the solution the speed is zero.

The evolution of the interface as well as the numerical solution of Equation(5.3.1) depend on the representation of $\mathbf{v}$. In the following sections three types of velocities are discussed in $2D$ space as follows:

- Curvature dependent motion
- Motion in the normal direction

- Externally generated motion

5.3.1 Curvature dependent velocity

When the interface motion is driven by a velocity field that depends on the mean curvature of the interface, the velocity can be modeled by a components in the normal direction only. The motion can be characterized by $v_n = -b\kappa$, the convection equation in Equation (5.3.3) becomes:

$$\frac{\partial \phi}{\partial t} - b\kappa \mid \nabla\phi \mid = 0 \tag{5.3.4}$$

where κ is the mean curvature and is defined to be:

$$\kappa = \text{div}\left(\frac{\nabla\phi}{\mid \nabla\phi \mid}\right) = \frac{\phi_{xx}\phi_y^2 - 2\phi_x\phi_y\phi_{xy} + \phi_{yy}\phi_x^2}{\left(\phi_x^2 + \phi_y^2\right)^{3/2}} \tag{5.3.5}$$

Convex regions in the interface, example point a in Figure 5.7, have $\kappa > 0$. Concave regions, example point b in Figure 5.7, have $\kappa < 0$ and $\kappa = 0$ for planar regions as in point c in Figure 5.7. The sign of the constant b in Equation(5.3.4) determines the directions of motion. When $b > 0$ the motion is in the direction of concavity, and when $b < 0$ the motion is in the direction of convexity. The term $b \mid \nabla\phi \mid$ is a *parabolic*[1] term, and the domain of information depends on all spatial directions. The numerical solution can be discretized using central differencing as follows:

$$\phi_{ij}^{n+1} = \phi_{ij}^n - \Delta t \left[b\kappa_{ij}^n \sqrt{(D_{ij}^{0x})^2 + (D_{ij}^{0y})^2} \right] \tag{5.3.6}$$

[1]since it depends on the second derivative of ϕ

where D_{ij}^{0x} and D_{ij}^{0y} are the second-order central difference as follows:

$$\begin{aligned} D_{ij}^{0x} &= \frac{\phi_{i+1,j}-\phi_{i-1,j}}{2\Delta x} \\ D_{ij}^{0y} &= \frac{\phi_{i,j+1}-\phi_{i,j-1}}{2\Delta y} \end{aligned} \tag{5.3.7}$$

Figure 5.8 shows the evolution of a star shaped interface driven by the speed of its mean curvature. The points on the convex regions move inward and points on the concave regions move outward. As proved by Grayson [26], the interface will finally evolve to a point and then disappear.

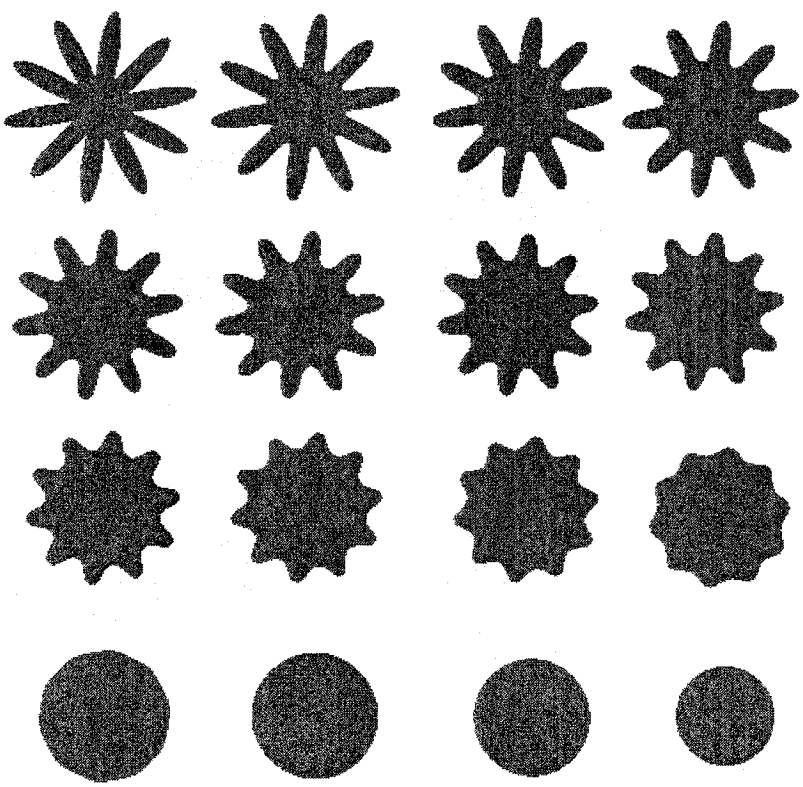


Figure 5.8: Development of a star interface with curvature dependent velocity. Points on the convex regions move inward and points on the concave regions move outward.

5.3.2 Motion in the normal direction

When the interface is moving under a constant velocity field in the normal direction, the value of v_n can be represented with $v_n = a$ and Equation(5.3.3) becomes:

$$\frac{\partial \phi}{\partial t} + a \mid \nabla \phi \mid = 0 \tag{5.3.8}$$

Where the sign of the constant a determines the direction of the motion. When $a > 0$ the interface moves in the direction of the normal, when $a < 0$ the motion is in the direction opposite to the normal. The term $a \mid \nabla \phi \mid$ is *hyperbolic*[2], the numerical solution can be approximated using the entropy satisfying scheme as follows:

$$\phi_{ij}^{n+1} = \phi_{ij}^{n} - \Delta t \left[\max(a,0) D^{+} + \min(a,0) D^{-} \right] \tag{5.3.9}$$

Operators D^+ and D^- are:

$$\begin{aligned} D^{+} &= \left[\max(D_{ij}^{-x},0)^2 + \min(D_{ij}^{+x},0)^2 + \max(D_{ij}^{-y},0)^2 + \min(D_{ij}^{+y},0)^2 \right]^{\frac{1}{2}} \\ D^{-} &= \left[\max(D_{ij}^{-x},0)^2 + \min(D_{ij}^{+x},0)^2 + \max(D_{ij}^{-y},0)^2 + \min(D_{ij}^{+y},0)^2 \right]^{\frac{1}{2}} \end{aligned} \tag{5.3.10}$$

where D_{ij}^{-x} and D_{ij}^{-y} are the first-order backward difference and D_{ij}^{+x} and D_{ij}^{+y} are the first-order forward difference defined as follows:

$$\begin{aligned} D_{ij}^{-x} &= \frac{\phi_{i,j} - \phi_{i-1,j}}{\Delta x} \\ D_{ij}^{+x} &= \frac{\phi_{i+1,j} - \phi_{i,j}}{\Delta x} \\ D_{ij}^{-y} &= \frac{\phi_{i,j} - \phi_{i,j-1}}{\Delta y} \\ D_{ij}^{+y} &= \frac{\phi_{i,j+1} - \phi_{i,j}}{\Delta y} \end{aligned} \tag{5.3.11}$$

Figure 5.9 shows the evolution of a star interface driven by an outward constant velocity.

[2]depends on at most the first derivatives of ϕ

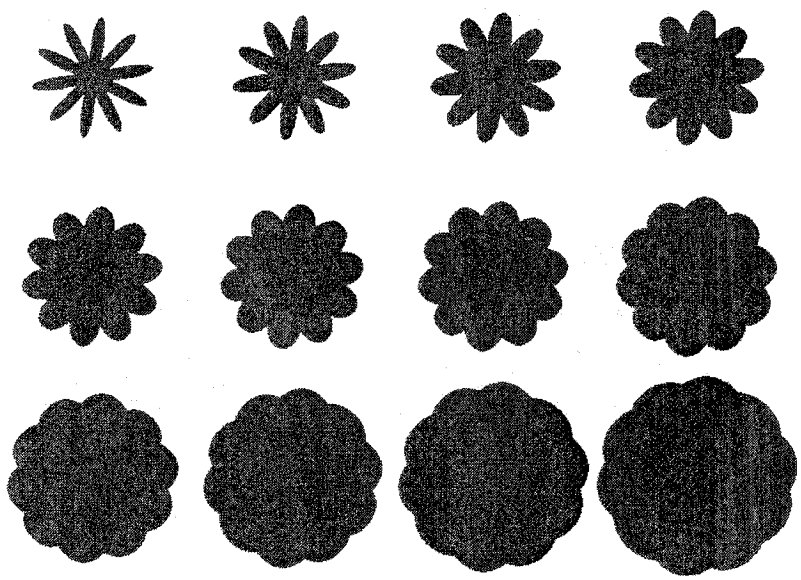


Figure 5.9: Development of a star interface with normal speed in the outward direction.

5.3.3 Motion in externally generated velocity

In case of an externally generated velocity field $\mathbf{v} = (u, v)$, the convection equation is as defined in Equation(5.3.1):

$$\frac{\partial \phi}{\partial t} + \mathbf{v}.\nabla\phi = 0 \tag{5.3.12}$$

The numerical solution can be approximated using the method of characteristics (the upwinding scheme) as follows:

$$\phi_{ij}^{n+1} = \phi_{ij}^{n} - \Delta t \left[\max(u_{ij}^{n}, 0)D_{ij}^{-x} + \min(u_{ij}^{n}, 0)D_{ij}^{+x} + \max(v_{ij}^{n}, 0)D_{ij}^{-y} + \min(v_{ij}^{n}, 0)D_{ij}^{+y}\right] \tag{5.3.13}$$

where D_{ij}^{-x}, D_{ij}^{+x}, D_{ij}^{-y} and D_{ij}^{+y} are defined in Equation(5.3.11). Figure 5.10 shows a polygon interface with rotational velocity in the clockwise direction.

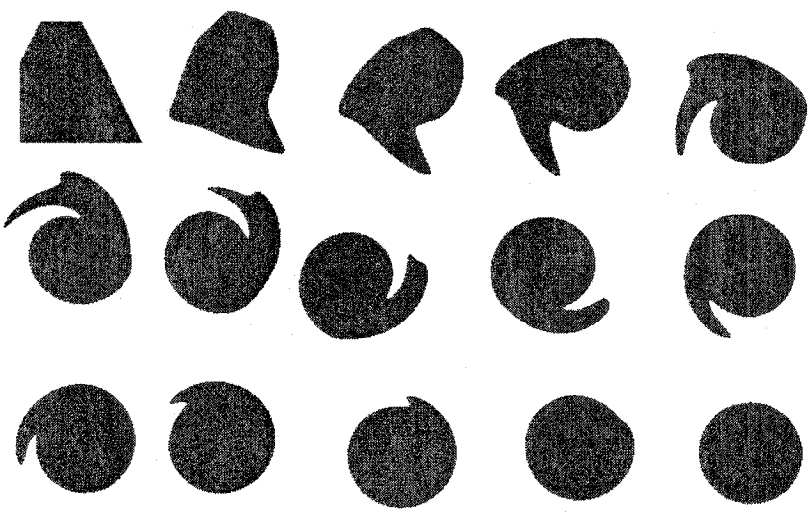


Figure 5.10: Development of a polygon interface with rotational velocity in the clockwise direction.

5.4 Geometric/variational method

In this section a variational level set formulation is used for ground plane obstacle reconstruction. The idea is to define a deformable surface model in 3D space that will converge to a smooth boundary corresponding to the geodesic contour having the minimal weighted area [7]. We propose an energy functional containing three terms: a term of shape-from-silhouettes consistency to characterize the ground plane objects structure and to account for possibly non-textured object surfaces; a term of photo-consistency to measure the conformity of the objects surface visual information to the acquired images, and finally, a term of regularization to bias the solution toward smooth object surfaces. The functional is minimized following the associated Euler-Lagrange surface evolution descent equations, implemented via level set PDEs for topology invariant surface evolution and numerical stability.

Before proceeding into discussing the method, we provide a comparison between our level set formulation and the formulations discussed in Section (5.2.5). The level set formulations in Section(5.2.5) are based on the following assumptions about the camera settings and the reconstructed scenes:

I. In Sekkati and Mitiche [76], the study was concerned with temporal image sequences and short-term motion, rather than with wide baseline images as in our case.

II. In Yezzi and Soatto [101], the 3D reconstruction of a single object and estimation of camera poses were sought under the following two assumptions:

a. the object surface is smooth and projects onto images as piecewise smooth irradiance segments with brightness discontinuities corresponding

to occlusion boundaries

b. the background and the object surface support two significantly distinct radiance functions

III. In Faugeras and Keriven [16, 67], the method, based on perspective invariant intensity correlation, required:

a. prior accurate knowledge of camera positions

b. pre-segmented images in terms of object and background

The methods in both (**II**) and (**III**) assume a model of brightness variations and a non-intervening background. These assumptions cannot be retained in our case because we are dealing with real-world scenes with objects that may not have sufficient brightness variations to inform on shape, and with backgrounds that cannot be easily abstracted out of the problem. The lack of sufficient object brightness variations is handled in our formulation by the shape-from-silhouette consistency component in the energy functional, and this is a major difference in our formalism. The inclusion of this shape-from- silhouette in the functional is important because of the assumptions it relaxes and that must be relaxed with images of real scenes. Also, in our method, the background is not abstracted out from of the reconstruction process, by assuming, for instance, a known or uniform-brightness background. Abstraction of the background would avoid the problem where it is most delicate to solve: object boundaries. Finally, we explicitly account for objects on a ground plane in the geometry.

In the next section we provide an overview about the geometry and the information we know about the scene before presenting our 3D reconstruction algorithm.

5.5 Scene geometry

Before discussing the reconstruction algorithm, it is important to note the information we know about the geometry of a scene given a set of N images collected by a mobile robot. The scene can be modeled as shown in Figure 5.11.

From Chapter 3, the inter-image homography and the ground plane mosaic are calculated. From Chapter 4, the camera motion is estimated. The intrinsic camera matrix $\mathbf{K}$ is assumed to be known, thus the projection matrices can be trivially computed with respect to a view selected as a reference as follows:

$$\begin{aligned} \mathbf{P}_0 &= \mathbf{K}[\mathbf{I} \mid \mathbf{0}] \\ \mathbf{P}_i &= \mathbf{K}[\mathbf{R} \mid \mathbf{t}] \end{aligned} \tag{5.5.1}$$

where $\mathbf{P}_0$ is the projection matrix of the reference camera C_0 and $\mathbf{P}_i$ is the projection matrix of camera C_i, $i = 1, ..., N$.

The normal $\mathbf{n}$ to the ground plane is also calculated in Chapter 4, the equation of the ground plane with respect to camera C_0 is given by:

$$\mathbf{n}.\mathbf{X} - d = 0 \tag{5.5.2}$$

Throughout this chapter, we assume that the inter-image homography transformations, the projection matrices, and the ground plane equation are all calculated as described in this section. Based on this, we proceed to solve the reconstruction problem.

The first step in the reconstruction process after obtaining the inter-image homography transformations and the projection matrices is to obtain a model of the ground plane over which the robot is operating.

By combining the computed in-between image homographies with the overhead transformations, it is possible to build the global overhead view mosaic of the ground

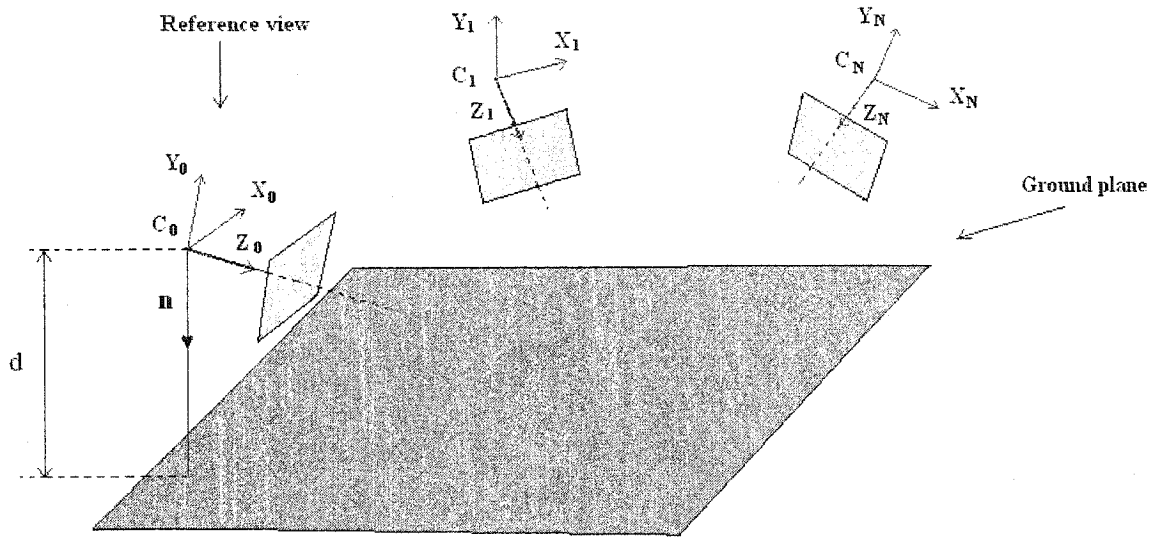


Figure 5.11: The geometry of the reconstruction scene.

plane. However, when assembling the different transformed views, only the image point showing the ground plane must be used, that is, images of the obstacles must be discarded. To do so, the algorithm in Section 3.2.1 is applied on the images in Figure 5.12. This figure shows a set of images with two textured obstacles on a textured heterogeneous ground. The mosaic obtained after discarding obstacle points is shown in Figure 5.13. This type of mosaic images is important for the visual hull construction as explained in the next section.

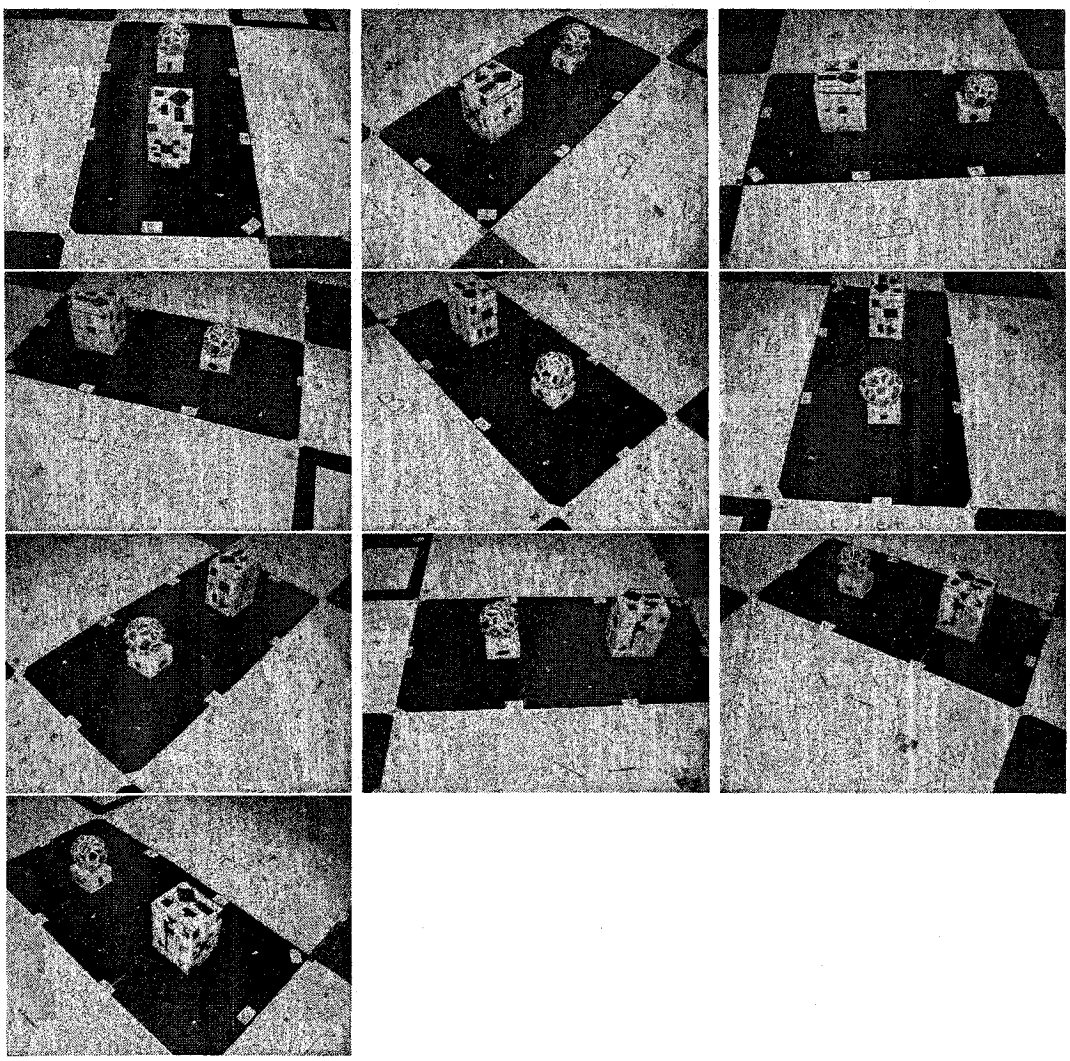


Figure 5.12: Different views of a ground plane with two textured obstacles.

5.6 Reconstruction of ground plane obstacles structure

Our goal here is to write a variational formulation that will allow us to proceed with the reconstruction of the ground plane obstacles. To achieve this goal, we have at our

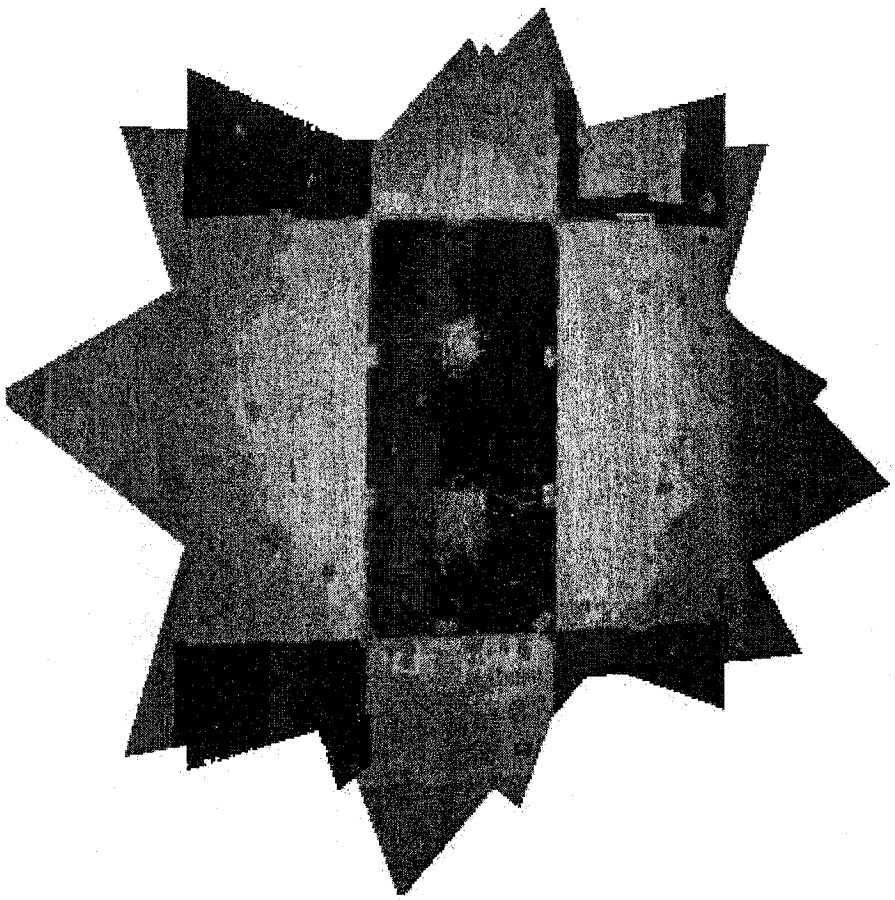


Figure 5.13: The overhead view mosaic of Figure 5.12 with obstacle points discarded.

disposal the different views of the scene, the corresponding projection matrices and a ground plane model in the form of a mosaic image.

5.6.1 Characterization

In this section the 3D reconstruction problem is characterized by the silhouettes (outlines) of the obstacles with respect to the ground plane and by the photometric information on the surface of the obstacles.

However, in the present case, our knowledge of the obstacle silhouettes is only approximative. We will therefore formulate the problem by trying to estimate the probability that a given 3D point belongs to the visual hull of an obstacle. Looking at a 3D point from one of the cameras, we know that if the background plane can be seen, then this means that the corresponding visual ray does not intersect the obstacle silhouette. Therefore, point $\mathbf{X}$ is necessarily outside any of the obstacles' visual hull, $VH(O)$. This can be verified by comparing the color of the pixel to which $\mathbf{X}$ is projected (given by $\mathbf{x} = \mathbf{PX}$) to the color of the ground plane of the point of intersection between the visual ray and the ground plane:

$$\lambda_i(\mathbf{X}) = \|\mathbf{I}_i^{\nu}(\mathbf{P}_i\mathbf{X}) - \mathbf{B}_i^{\nu}(\mathbf{H}_{iO}\mathbf{P}_i\mathbf{X})\| \tag{5.6.1}$$

where $\mathbf{I}_i^{\nu}$ and $\mathbf{B}_i^{\nu}$ are the RGB color vectors for the projection of a 3D point $\mathbf{X}$ on the image plane and the ground plane respectively. The approach can be demonstrated by looking at Figure 5.14. The value of $\lambda_i(\mathbf{X})$ is expected to be low for points outside the visual cone and high otherwise. Assuming that visual measurements are subject to Gaussian noise, the probability of a given point to be along a silhouette visual ray can be given by:

$$P(\mathbf{X} \in VH(O) \mid \mathbf{I}_i) \propto 1 - e^{-\frac{\lambda_i^2(\mathbf{X})}{\beta^2}} \tag{5.6.2}$$

A plot of Equation (5.6.2) is shown in Figure 5.15(a).

Since the visual hull is given through the intersection of all silhouette rays and assuming independence between views, the probability that a point $\mathbf{X}$ is within the visual hull can be expressed as:

$$P(\mathbf{X} \in VH(O)) \propto \frac{1}{1 + e^{-\alpha\varpi\mathbf{X}}} \prod_i \left(1 - e^{-\frac{\lambda_i^2(\mathbf{X})}{\beta^2}}\right) \tag{5.6.3}$$

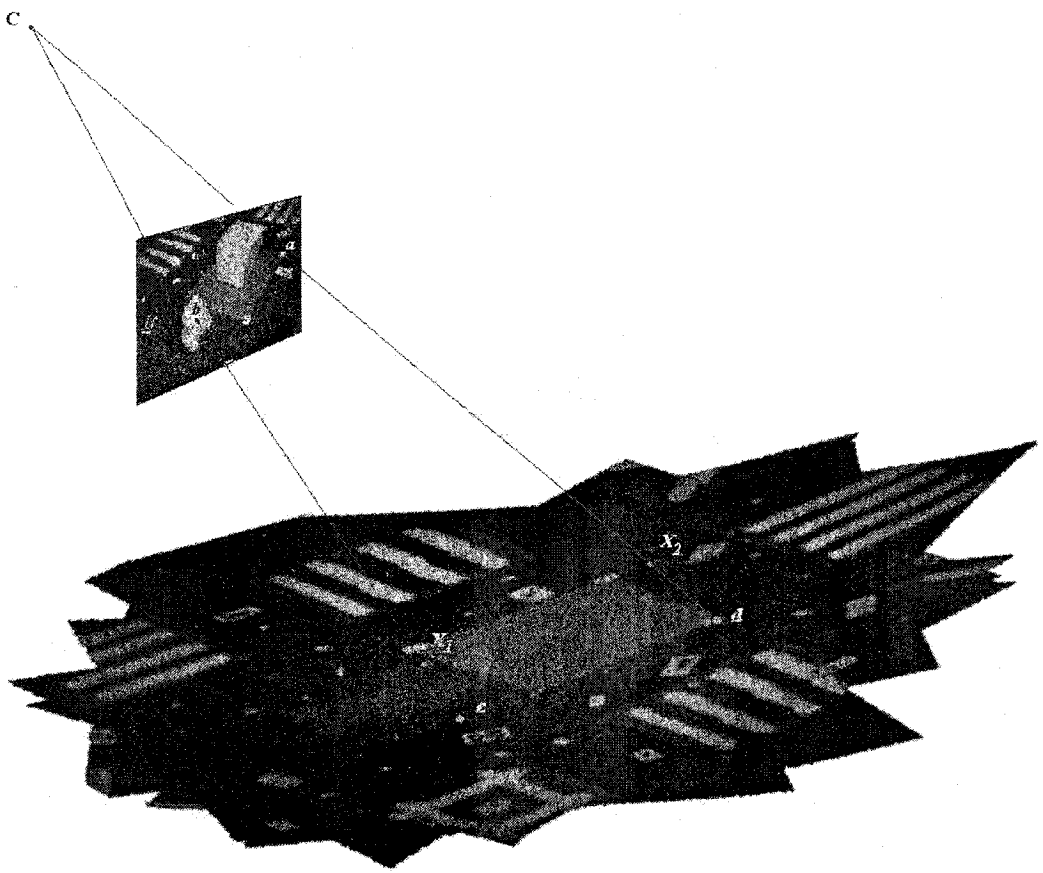


Figure 5.14: A model of the ground plane for the set of images of Figure 5.18. The ground plane point d can be seen from image point a, then the ray through the 3D point $\mathbf{X}_2$ is outside the obstacle. On the other hand, ground point c cannot be seen from image point b, then the ray through the 3D point $\mathbf{X}_1$ intersects the obstacle.

The extra leading factor in the right-hand side of the expression above reflects our knowledge of the scene structure. Indeed, from the calibration step, we have at our disposal the equation of the plane (Equation (5.5.2)) over which the obstacles are lying. Consequently, for any point located under the plane (i.e., for which $\varpi X < 0$), the probability in Equation (5.6.3) is brought back to zero.

The shape-from-silhouette approach, as represented by Equation (5.6.3), is a very powerful formulation of the reconstruction problem. However, to further improve the

3D reconstruction of the observed objects, the photometric information contained in each view can be used by adopting the *voxel-coloring* strategy. The photo-consistency of a given 3D point can be measured by computing the standard deviation of the pixel colors to which that 3D point projects [72]. A more robust approach would consist in considering instead the median deviation from the pixel average color value; this would avoid having a very dissimilar pixel to excessively contribute to the consistency measure. Here, we use a continuously derivable function similar to the one proposed by [22] and given by $-(1+|t|)^{-1}$. It is a concave function that have been introduced to implicitly address the outlier problems. This is important in our application since, because of occlusion, it is not expected to have all cameras seeing all obstacle points. Our goal is simply to maximize the consensus concerning the color of an obstacle point as seen by the different views. The conformity of a given point $\mathbf{X}$ to the observation is then measured by:

$$\Gamma(\mathbf{X}) = \sum_i \frac{-1}{1+\gamma\|\mathbf{I}_i^\nu(\mathbf{P}_i\mathbf{X}) - \overline{\mathbf{I}^\nu}(\mathbf{X})\|} \tag{5.6.4}$$

where $\mathbf{I}_i^\nu$ is the *RGB* vector of the project of point $\mathbf{X}$ on image i and $\overline{\mathbf{I}^\nu}$ is the average *RGB* vector of the projection of point $\mathbf{X}$ on all the views. A plot of $\Gamma(\mathbf{X})$ is shown in Figure 5.15(b). This function is expected to be low for points located on an obstacle surface.

The two Equations(5.6.3) and (5.6.4) can now be combined to obtain an energy functional representing our characterization of the ground plane obstacle reconstruction problem.

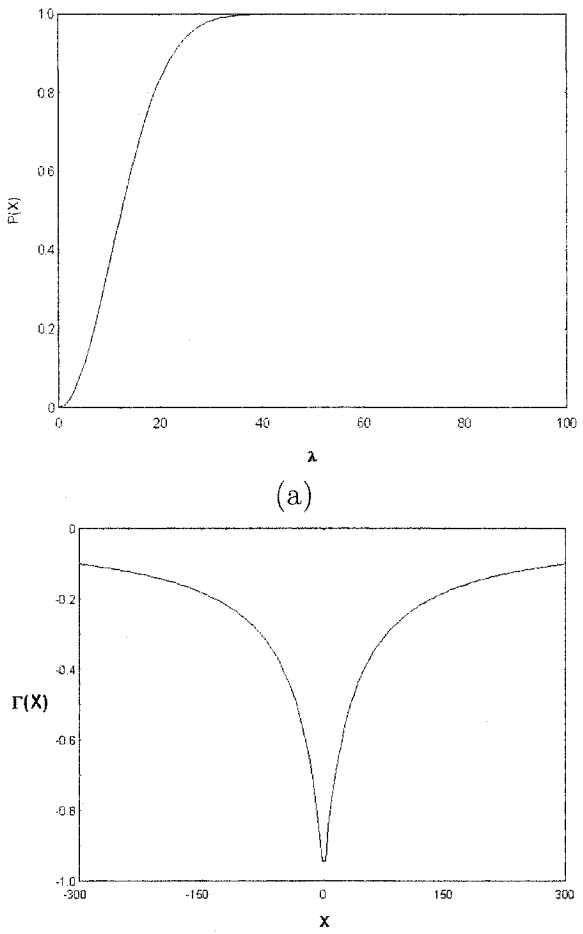


Figure 5.15: (a) graph of $P(\mathbf{X})$ for $\beta = 0.066$ (b) graph of $\Gamma(\mathbf{X})$ for $\gamma = 0.05$.

5.6.2 Functional, Euler-Lagrange equations, and level set representation

An energy functional to minimize can now be written. This functional will contain terms which follow from our characterization of ground plane obstacles reconstruction (Equations (5.6.3) and (5.6.4)) and a regularization term to obtain smooth obstacle

surfaces. Let S be a closed surface and R_S the region enclosed by S. Let

$$f(\mathbf{X}) = \frac{-1}{1+e^{-\alpha\varpi\mathbf{X}}} \prod_i \left(1 - e^{-\frac{\lambda_i^2(\mathbf{X})}{\beta^2}}\right) \tag{5.6.5}$$

The energy functional to minimize over all closed surfaces S is:

$$E(S) = \int_{R_S} f(\mathbf{X})\, dV + a \int_S \Gamma(\mathbf{X})\, dS + b \int_S dS \tag{5.6.6}$$

where a and b are positive constant weight coefficients.

The first integral on the right hand side of Equation(5.6.6) is a volume integral, it can be transformed into a surface integral over S using Gauss' divergence theorem [13] as follows:

$$\int_{R_S} f\, dV = \int_{R_S} \operatorname{div} \mathbf{F}\, dV = \int_S \mathbf{F}.\mathbf{n}\, dS \tag{5.6.7}$$

where $\mathbf{F}$ is a vector field and $\mathbf{n}$ is the outward unit normal to S.

Let $g(\mathbf{X}) = a\Gamma(\mathbf{X}) + b$, Equation (5.6.6) becomes:

$$E(S) = E_1(S) + E_2(S) \tag{5.6.8}$$

where $\quad E_1(S) = \int_S \mathbf{F}.\mathbf{n}\, dS \quad$ and $\quad E_2(S) = \int_S g(\mathbf{X})\, dS$

Generic derivations of the Euler-Lagrange equations corresponding to each integral in Equation (5.6.8) (surface integrals of scalar functions) can be evaluated as follows:

Let ϕ be a parametrization of S such that $\phi : (r,s) \in D \longmapsto \mathbf{X}(r,s) \in S$

The surface integrals of $E_1(S)$ and $E_2(S)$ can be expressed as follows:

$$E_1(S) = \int_S \mathbf{F}.\mathbf{n}\, dS = \iint_D \mathbf{F}.\mathbf{n}\, \|\mathbf{N}\|\, dr\, ds \tag{5.6.9}$$

$$E_2(S) = \int_S g(\mathbf{X})\, dS = \iint_D g(\mathbf{X}(r,s))\, \|\mathbf{N}\|\, dr\, ds \tag{5.6.10}$$

where $\mathbf{N}$ is normal to the surface S defined to be $\mathbf{N} = \mathbf{X}_r \times \mathbf{X}_s$ and the unit normal $\mathbf{n} = \mathbf{N}/|\mathbf{N}|$. The partial derivatives of $\mathbf{X}$ with respect to r and s are indicated by $\mathbf{X}_r$ and $\mathbf{X}_s$ respectively.

Let L_1 and L_2 designate the integrands of E_1 and E_2 respectively, the Euler-Lagrange equations [21] associated with E_1 and E_2 are:

$$\frac{\partial E_1}{\partial \mathbf{X}} = \frac{\partial L_1}{\partial \mathbf{X}} - \frac{\partial}{\partial r}\frac{\partial L_1}{\partial \mathbf{X}_r} - \frac{\partial}{\partial s}\frac{\partial L_1}{\partial \mathbf{X}_s} \tag{5.6.11}$$

$$\frac{\partial E_2}{\partial \mathbf{X}} = \frac{\partial L_2}{\partial \mathbf{X}} - \frac{\partial}{\partial r}\frac{\partial L_2}{\partial \mathbf{X}_r} - \frac{\partial}{\partial s}\frac{\partial L_2}{\partial \mathbf{X}_s} \tag{5.6.12}$$

Mitiche et al. [59] derived a similar functional for tracking moving objects in a sequence of images. Based on their analysis, the Euler-Lagrange equations in Equations (5.6.11) and (5.6.12) can be expressed as follows:

$$\frac{\partial E_1}{\partial \mathbf{X}} = f\mathbf{N} \tag{5.6.13}$$

$$\frac{\partial E_2}{\partial \mathbf{X}} = (\nabla g.\mathbf{n} - 2g\kappa)\mathbf{N} \tag{5.6.14}$$

Then, the Euler-Lagrange descent equations to minimize Equation(5.6.6) is:

$$\frac{d\phi}{dt} = -(f + \nabla g \cdot \mathbf{n} - 2g\kappa)\mathbf{n} \tag{5.6.15}$$

where κ is the mean curvature function. The right-hand side of Equation (5.6.15) is independent of surface parametrization, as it should be, because we are interested in the image S of ϕ and not in ϕ itself.

Execution of the descent equation by explicit representation of S as a set of points does not accommodate changes in the topology of S. An alternative execution is via level sets where S is represented implicitly as the zero level set of a one-parameter family of functions u:

$$(\forall \tau) \quad u \circ \phi(\tau) = u(x(\tau), y(\tau), z(\tau), \tau) = 0 \tag{5.6.16}$$

where x, y, z are the spatial variables. Differentiation of Equation (5.6.16) with respect to τ yields the level set equation that drives the evolution of u:

$$\nabla u \cdot \frac{d\phi}{d\tau} + \frac{\partial u}{\partial \tau} = 0 \tag{5.6.17}$$

Referring to (5.6.15) and taking u to be positive inside S_R and negative outside ($u = 0$ on S) so that $\mathbf{n} = -\nabla u / \|\nabla u\|$, the evolution equation of u is, in our case:

$$\frac{\partial u}{\partial \tau} = (\nabla g \cdot \frac{\nabla u}{\|\nabla u\|} + 2g\kappa - f)\|\nabla u\| \tag{5.6.18}$$

Mean curvature is expressed in terms of u as [77]:

$$\kappa = \text{div}\left(\frac{\nabla u}{\|\nabla u\|}\right) \tag{5.6.19}$$

By construction, S can be recovered at any instant as the 0-level surface of function u regardless of the changes in topology during its evolution.

5.7 Experimental results

Figure 5.17 shows the obtained 3D structure of the obstacles. This one has been obtained using the 10 available views of the scene (Figure 5.12) and the ground plane model shown in Figure 5.13. Figure 5.16 shows the evolution of the initial cubic shape during the execution of the level set algorithmic minimization. Figures 5.16(a) and (b) show the reconstructed obstacles after 400 and 1000 iterations respectively. The finally reconstructed obstacles after 1400 iterations are shown in Figure 5.17. In the voxel coloring methods (see Section 5.2.3), the ordinal visibility constraint has to be maintained in the reconstruction process. The voxels are classified (in a binary way) as either opaque when the voxel belongs to the object surface or transparent if it does

not belong to the object surface. In our case the ordinal constraint is relaxed and the voxels are visited in arbitrary order. This is due to the fact that each voxel is classified, using Equation(5.6.4), in a fuzzy way as a point with low or high value. Figure 5.15(b) shows a plot of the function $\Gamma(\mathbf{X})$ for an empirically selected value of $\gamma = 0.05$. If the voxel $\mathbf{X}$ is located on the obstacle surface, then $\mathbf{I}_i^{\nu}(\mathbf{P}_i\mathbf{X}) \simeq \overline{\Gamma^{\nu}}(\mathbf{X})$ and voxel $\mathbf{X}$ is assigned a low value, otherwise the voxel $\mathbf{X}$ is assigned a relatively high value. This assigned value for a voxel $\mathbf{X}$, relatively low on the obstacle surface, is significant in our formalism.

To demonstrate the validity of the method, we applied the algorithm on scenes with both textured and non-textured obstacles. Figure 5.18 shows 12 images of a scene with a textured obstacle (the box) and a non-textured obstacle (the frog).

These types of scenes : nonuniform-brightness background combined with both textured and non-textured objects are challenging to most 3D reconstruction algorithms. Figures 5.19(a) and (b) show the functions $f(\mathbf{X})$ and $g(\mathbf{X})$ for a layer parallel to the ground plane. The function $f(\mathbf{X})$ is high inside the obstacles and low outside as shown in Figure 5.19(a). On the other hand, the function $g(\mathbf{X})$ is ideally low on the surface of the obstacles and high otherwise. However, due to the textured background some low values may be detected outside the surface of the obstacles as shown in Figure 5.19(b). Combining both $f(\mathbf{X})$ and $g(\mathbf{X})$ provides a good estimate of the obstacle surface. Figure 5.20 shows the evolution of the layer after starting with a square interface. The interface changes its topology naturally without user intervention. The interface brakes and finally evolve to represent the structure of the objects as shown in Figure 5.20(p). The function $f(\mathbf{X})$ plays an important role in this evolution. Consider the interface in Figure 5.20(m), two parts of the interface

are evolving towards the obstacle surface and one part splits away. Since the value of $f(\mathbf{X})$ is zero out side the obstacles, the evolving curve that split outside the obstacles will finally disappear. The curves that evolve towards the obstacles will not cross the obstacle surface due to the high value of $f(\mathbf{X})$ inside the obstacle. The minimal energy of these curves is at the surface of the obstacles. Some views of the finally reconstructed obstacles after 1500 iterations are shown in Figure 5.21.

5.8 Test on an outdoor natural sequence

The capability of our 3D reconstruction method is further demonstrated by considering outdoor scenes. As an example, consider the images in Figure 5.22. This sequence of images represent a textured obstacle (the stone) lying on a textured ground plane.

This scene is similar to the two indoor scenes studied in Figures 5.12 and 5.18, the only difference is that Figure 5.22 represents an outdoor scene with naturally occurring textures (like tree leaves). The algorithm is applied in the same manner as in the previous two indoor scenes. First, the image matching algorithm presented in Chapter 3 is applied and the homography induced by the planar ground is calculated for all the images with respect to one selected reference image. These inter-image homography transformations are used to draw the ground plane mosaic of the environment as shown in Figure 5.23 and to calculate the motion of the camera as described in Chapter 4. Finally, the projection matrices for all the images are calculated as presented in Equation 5.5.1.

Our proposed 3D reconstruction algorithm is applied. Starting with a cubic interface, the evolution of the ground plane obstacle reconstruction can be shown in Figure 5.24 after 250, 500, 750 and 1000 iterations. The finally reconstructed stone

after 1500 iteration is shown in Figure 5.25.

This shows that our algorithm can be applied to natural outdoor scenes. However, outdoor scenes impose some challenges on our method. This due to the fact that many factor (natural or human made) may affect the scene under study. The illumination changes is one important challenge. The illumination of a scene may vary due to many factors. For example, when a large dynamic equipments (like cars) are present in the workspace, these equipments might create shady areas and thus the collected images may suffer from intensity variations. Also a sudden change of sunlight, rain, snow or other natural conditions may result in illumination changes. Other important challenge is the dynamics of the scene. The location of some objects in the scene may vary while the image collection is underway. For example, some tree leaves on the ground plane may freely move in the scene from one position to another. Thus the rigidity of the scene is violated and our image matching step is particularly challenged.

5.9 Numerical implementation

The numerical implementation is based on the level set methods proposed by Osher and Sethian [68, 77]. Equation(5.6.18) can be expressed as follows:

$$\frac{\partial u}{\partial \tau} = s_1 \|\nabla u\| + s_2 \|\nabla u\| \tag{5.9.1}$$

where $s_1 = \nabla g \cdot \frac{\nabla u}{\|\nabla u\|} - f$ and $s_2 = 2g\kappa$. The coefficients a and b are set empirically to both equal 1. The first step in the implementation is to discretize the object scene to 3D grid points (voxels). This is achieved by our knowledge of the ground plane equation with respect to the selected reference camera as defined in Equation(5.5.2). The 3D grid points are divided into layers parallel to the ground plane. Each layer

has 200×200 grid points for a total of 100 layers. The surface evolution represented in Equation (5.9.1) is evaluated one layer at a time. The final output of our level set method is a dense cloud of points representing the surface of the obstacles located above the ground plane.

The term $s_1\|\nabla u\|$ may be approximated using the entropy satisfying scheme (as in Section 5.3.2) while the term $s_2\|\nabla u\|$ may be approximated using the central difference (as in Section 5.3.1). The discretization of u is as follows:

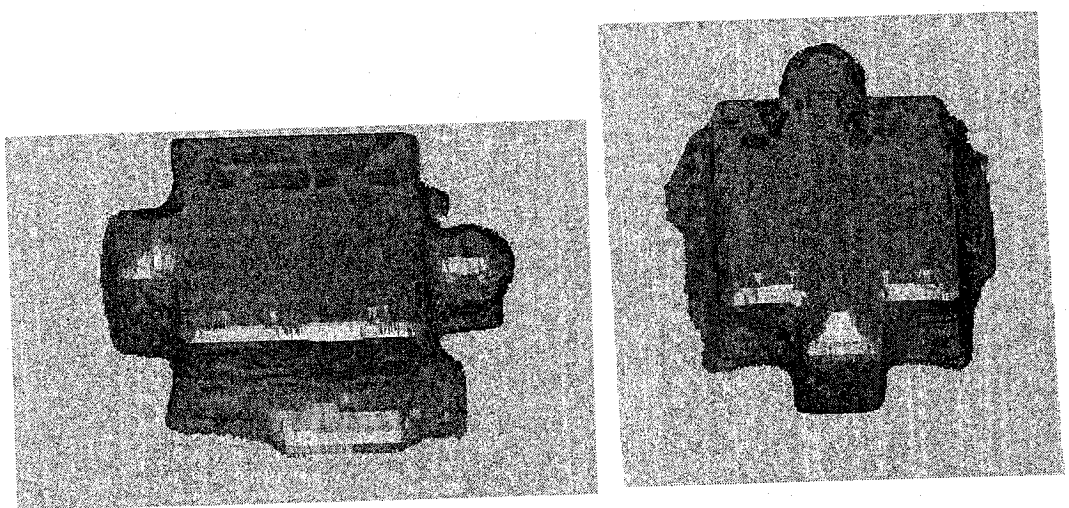
$$u_{i,j}^{n+1} = u_{i,j}^{n} - \Delta\tau \left\{ \left[\max(s_1^n, 0)D^+ + \min(s_1^n, 0)D^-\right] + \left[s_2^n \sqrt{(D_{ij}^{0x})^2 + (D_{ij}^{0y})^2}\right] \right\} \tag{5.9.2}$$

The operators D^+ and D^- are defined in Equation (5.3.10). Operators D_{ij}^{0x} and D_{ij}^{0y} are the central difference approximation calculated as defined in Equation(5.3.7).

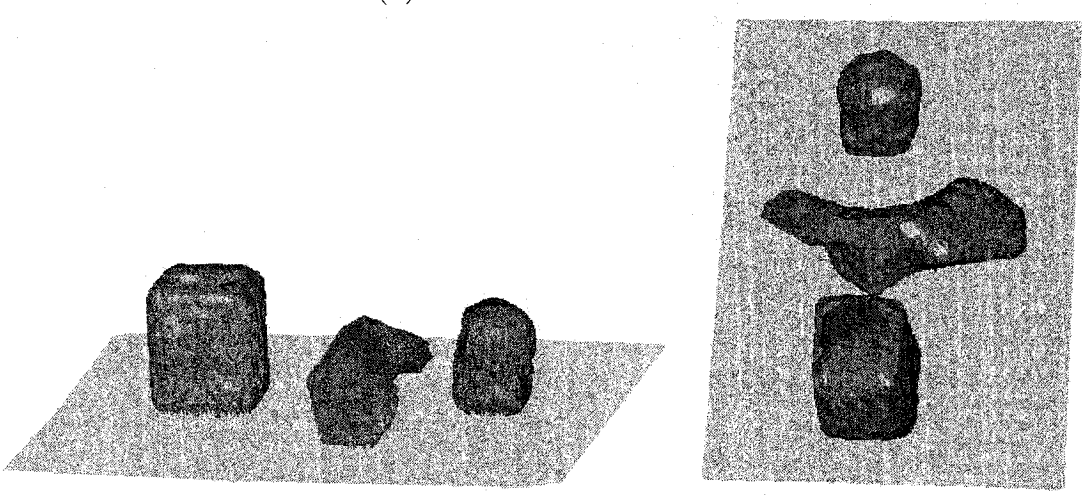
5.10 Summary

In this Chapter I propose a novel method to reconstruct ground plane obstacles by using a collection of sparse views of the scene. A robot equipped with a camera takes several images of the scene from which calibration information is obtained. A ground plane model of the scene is built and obstacles are thus detected. The main contribution in this chapter is the level set formulation for obstacle reconstruction. The level set surface evolution equations minimize an energy functional characterizing properties of the 3D obstacle structure. The conformity of the sought surface to the acquired images is verified through a voxel coloring formulation. The method also incorporates shape-from-silhouette concepts in the functional to minimize in order

to account for possibly non-textured obstacle object surfaces. This extends the 3D reconstruction problem statement to an arbitrary number of real-world objects which may lack the brightness variations necessary to infer a shape-from-brightness process.

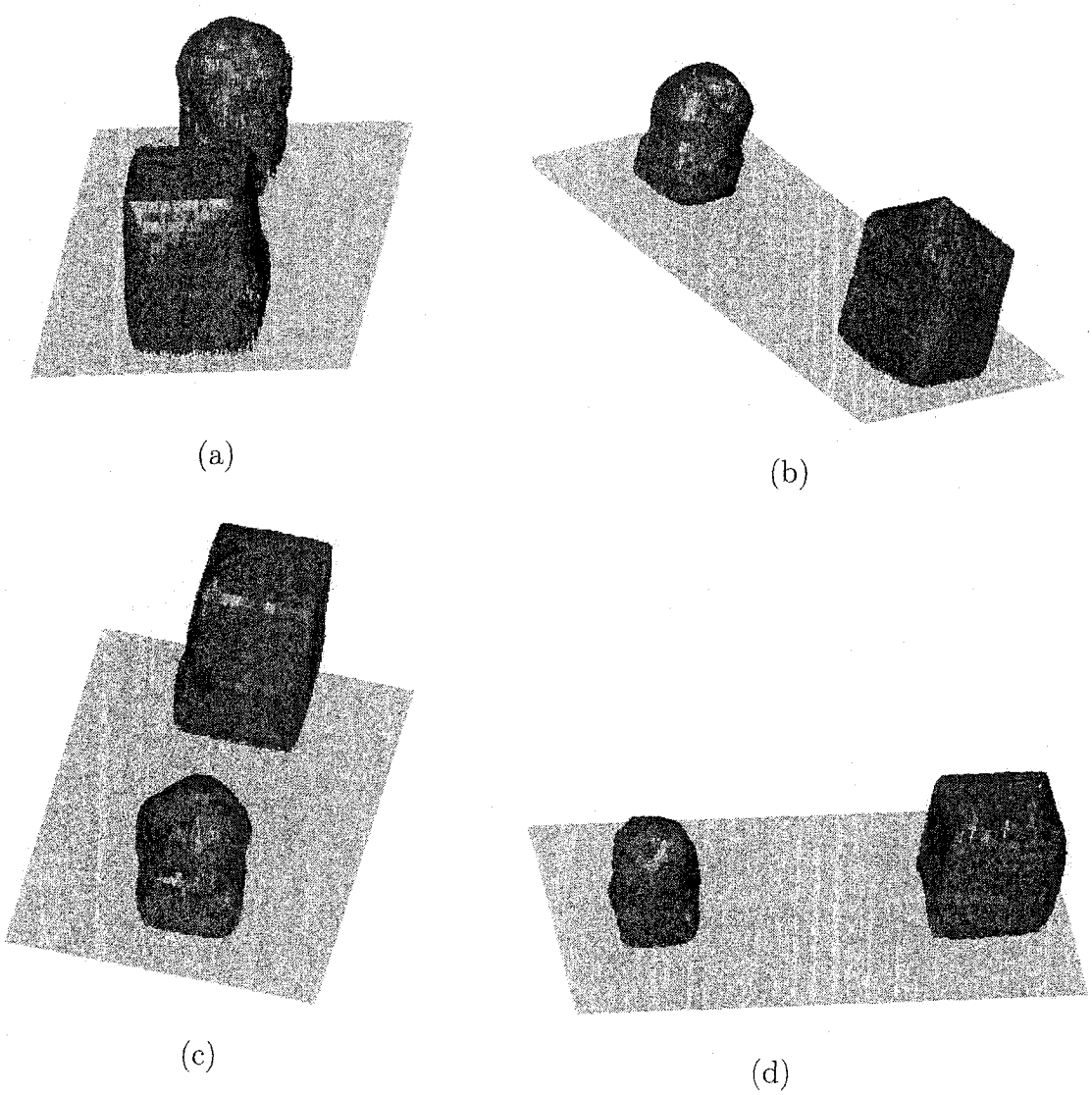


(a) after 400 iterations



(b) after 1000 iterations

Figure 5.16: Evolution of the reconstructed obstacles.



(a) (b)

(c) (d)

Figure 5.17: Different views of the reconstructed obstacles after 1400 iterations .

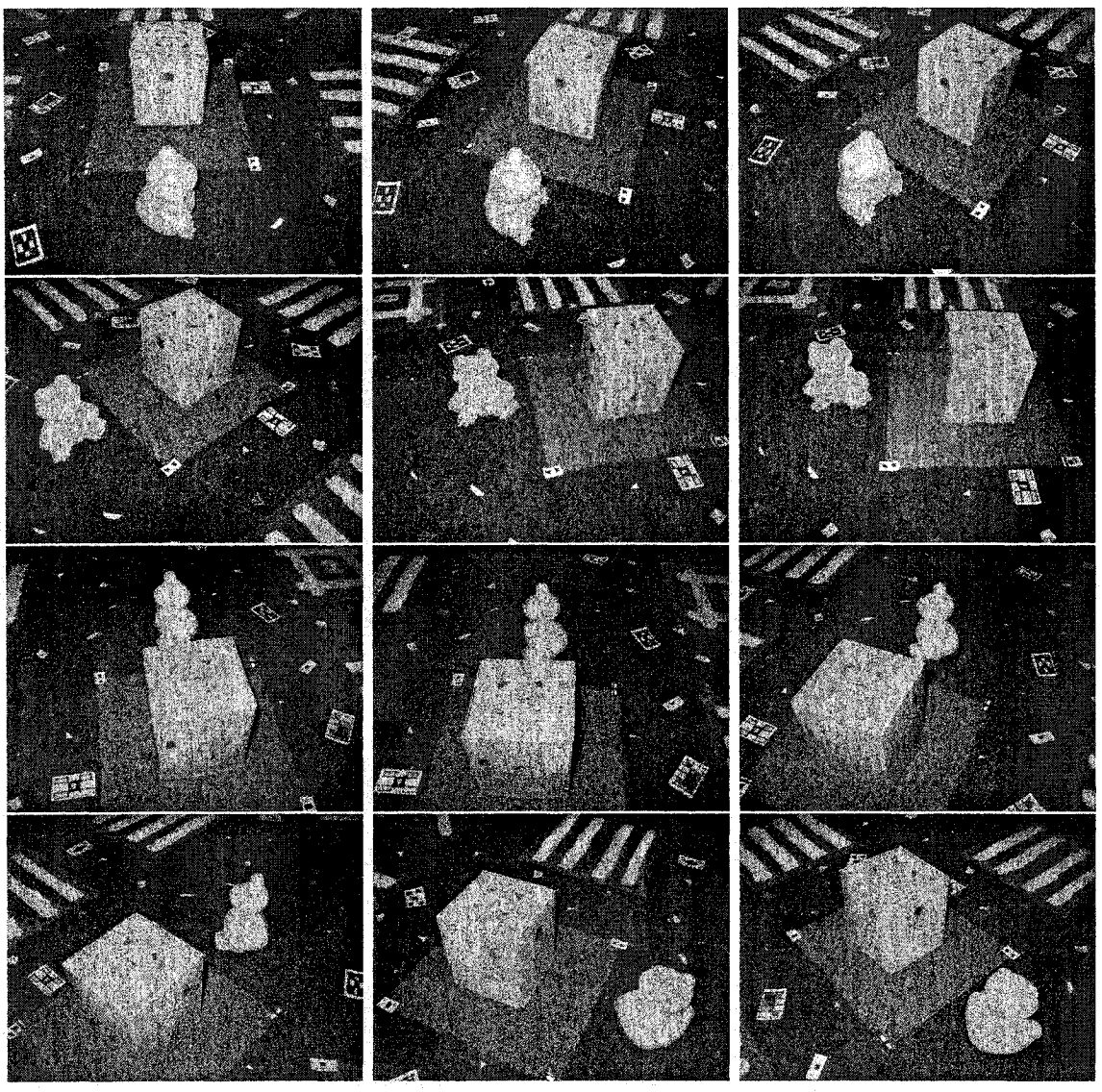
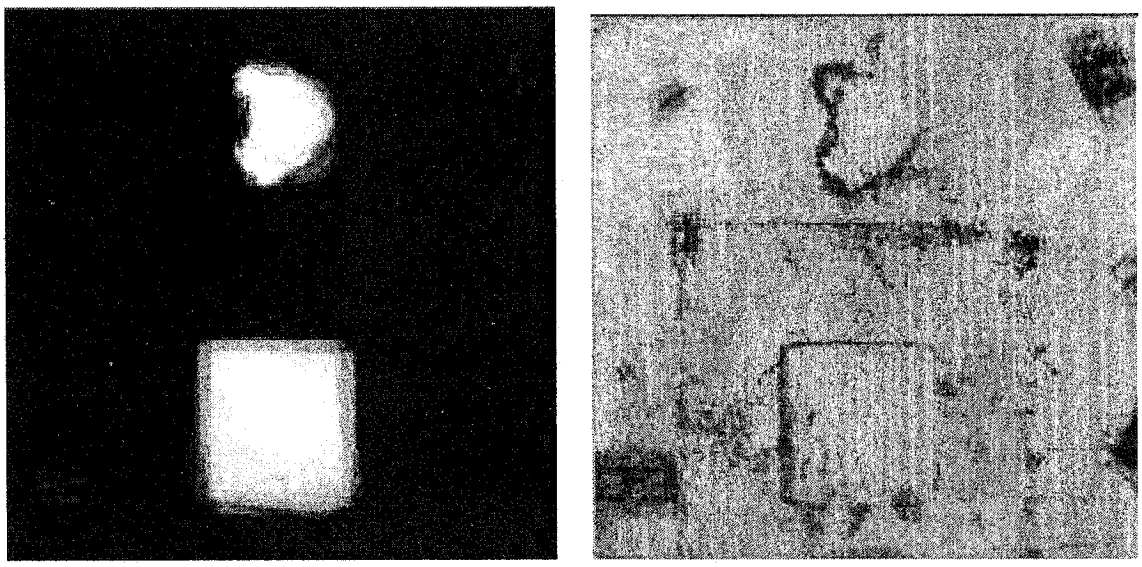


Figure 5.18: Different views of a ground plane scene with two objects on a heterogeneous background . The first object (the frog) is textureless and the second one (the box) is textured. These types of scenes are challenging for most 3D reconstruction algorithms



(a) (b)

Figure 5.19: Functions $f(\mathbf{X})$ and $g(\mathbf{X})$ for a layer parallel to the ground plane (a) function $f(\mathbf{X})$ (b) function $g(\mathbf{X})$. Dark areas represent low values and bright areas represent high values.

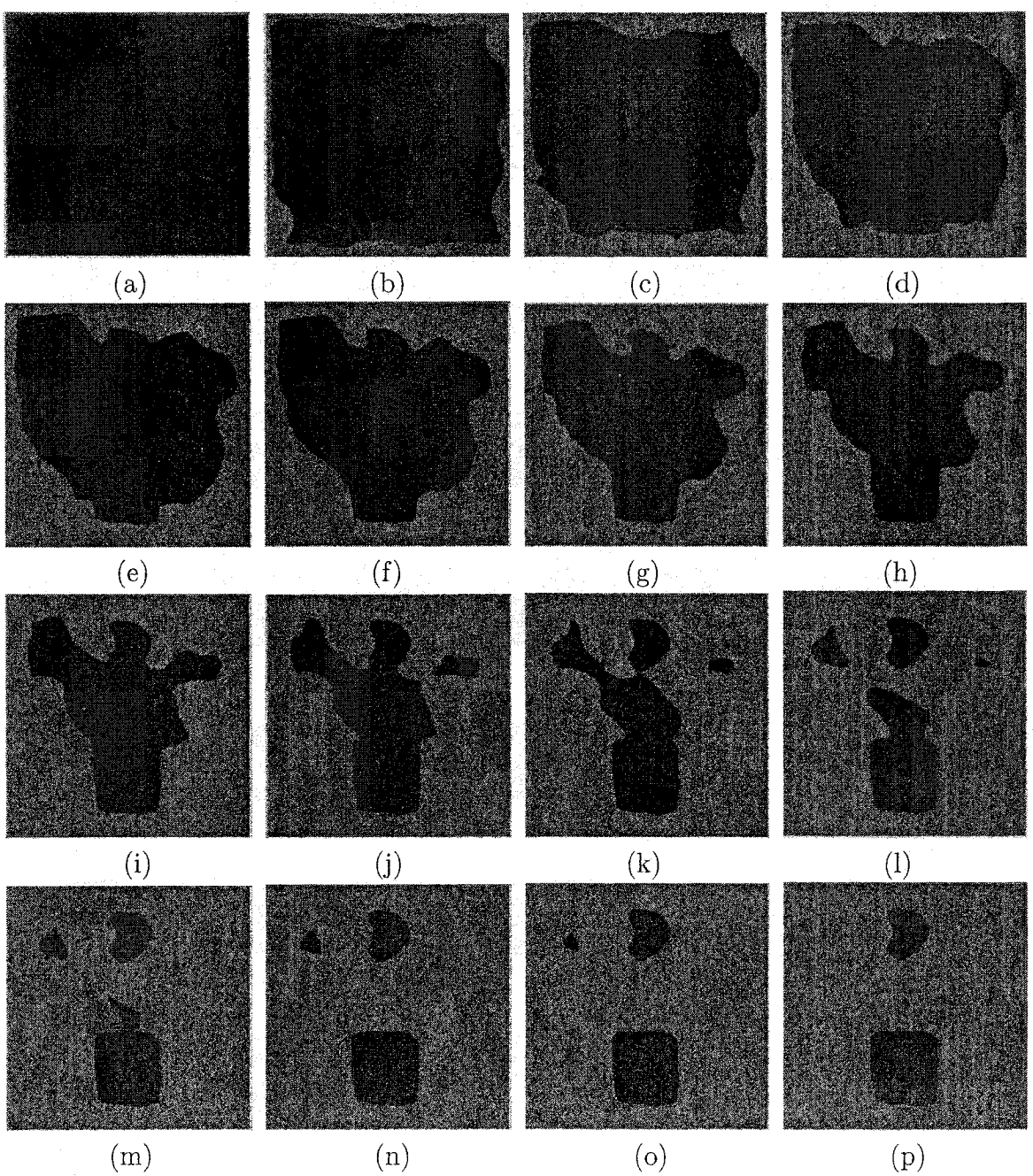


Figure 5.20: Evolution of a layer starting with a square interface. The interface splits and the motion is towards the surface of the two obstacles. The part of the interface that splits outside the obstacles will finally disappear.

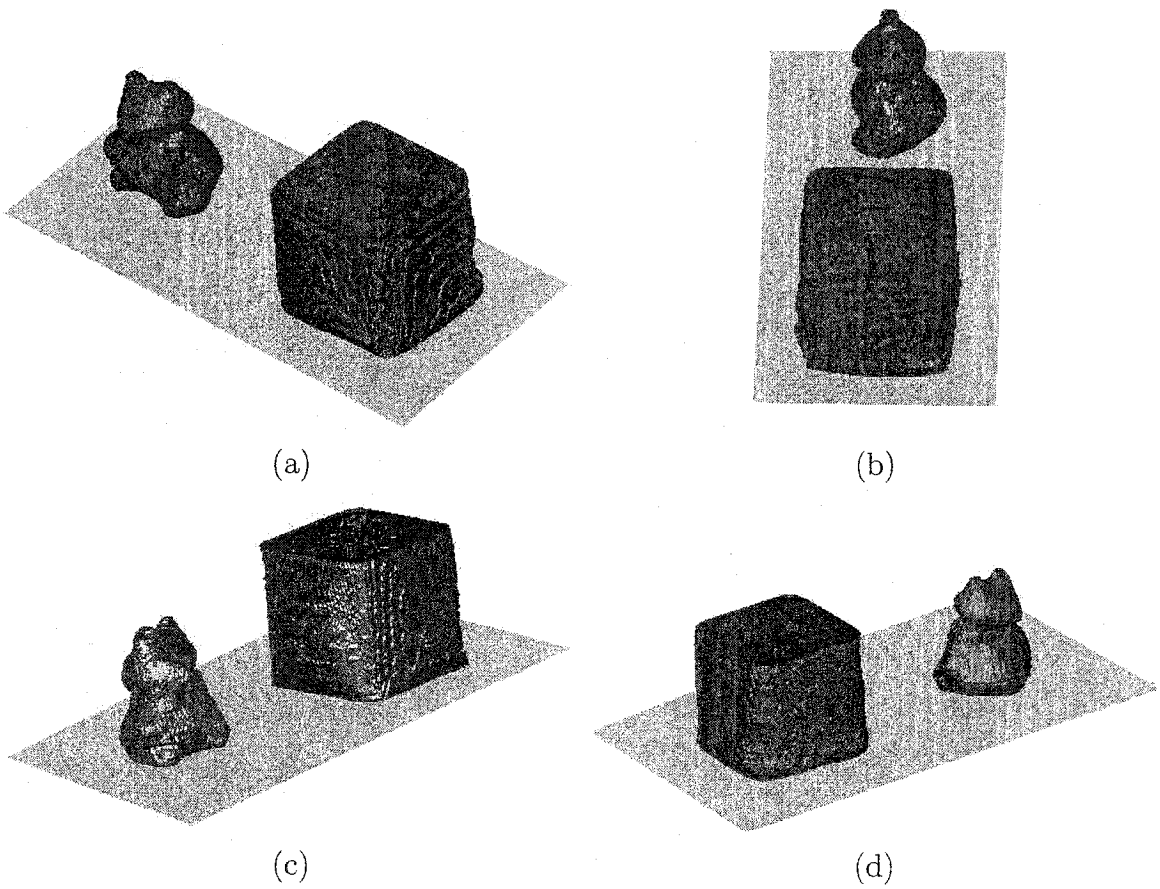


Figure 5.21: Different views of the reconstructed obstacles after 1500 iterations.

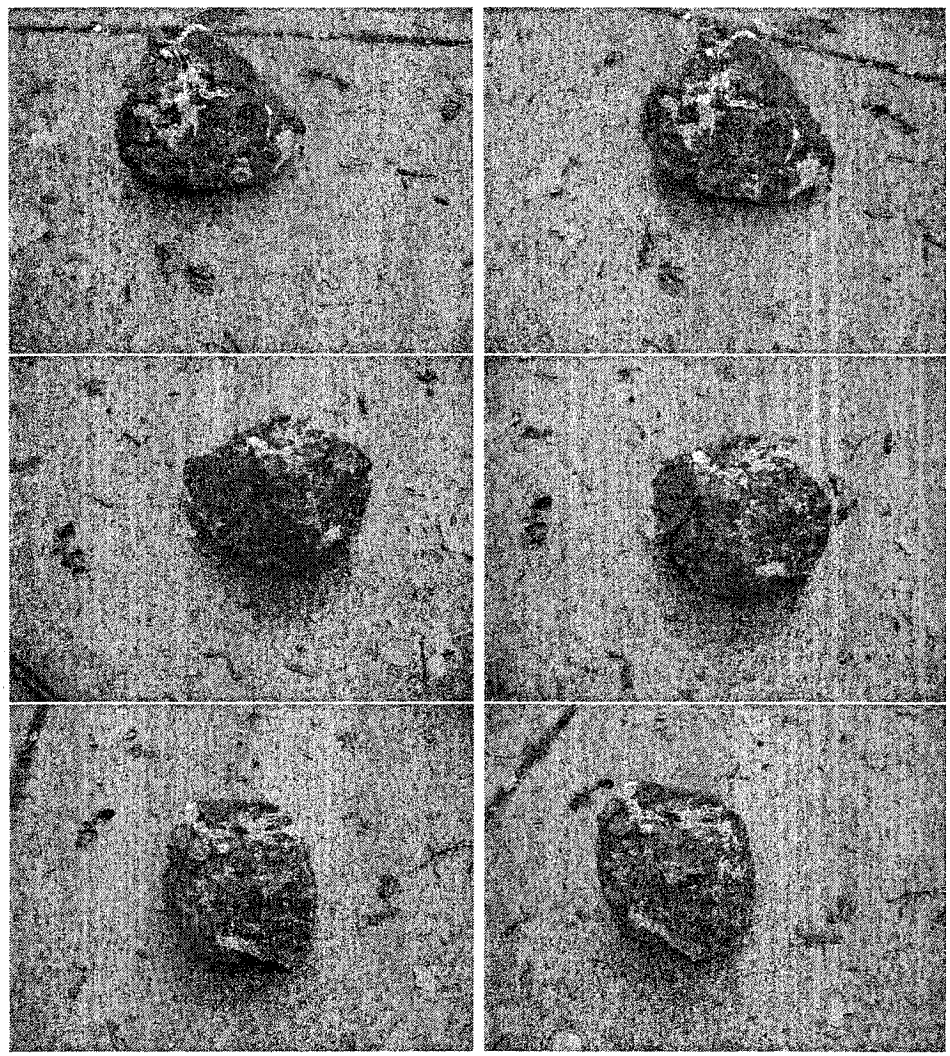


Figure 5.22: Different views of an obstacle (the stone) in an outdoor scene. This figure shows 6 out of the 15 images used in the reconstruction.

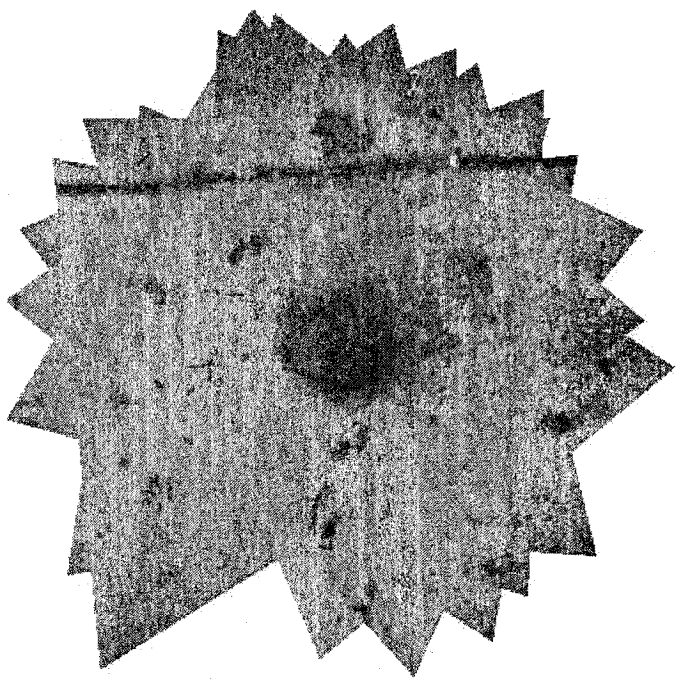
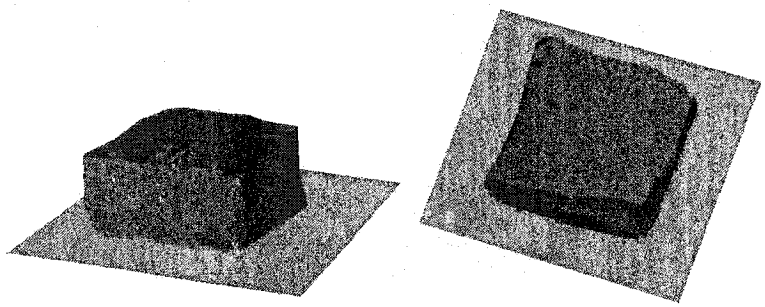
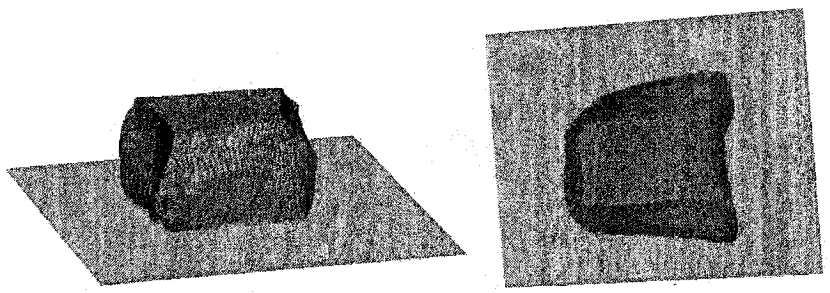


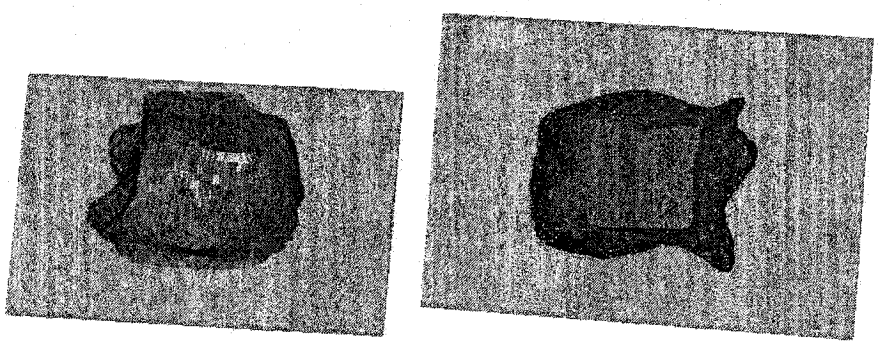
Figure 5.23: The overhead view mosaic of Figure 5.22 with obstacle points discarded.



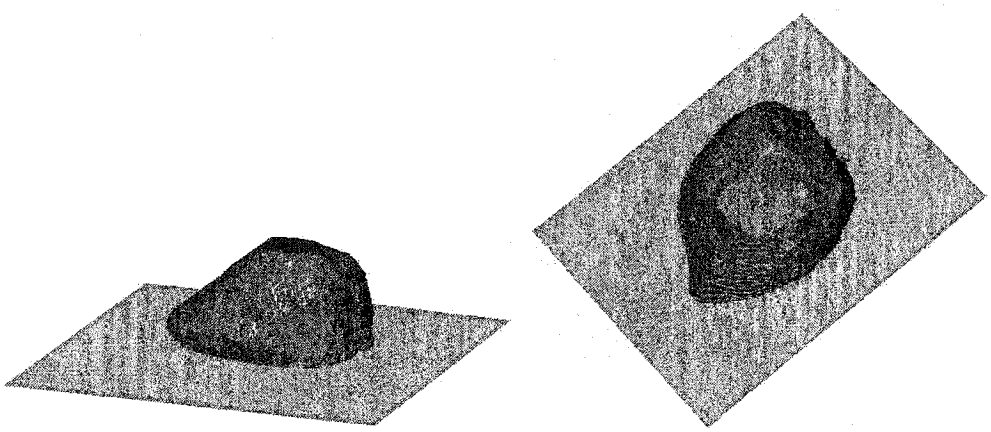
(a) after 250 iterations



(b) after 500 iterations



(b) after 750 iterations



(b) after 1000 iterations

Figure 5.24: Evolution of the reconstructed stone starting with a cubic interface.

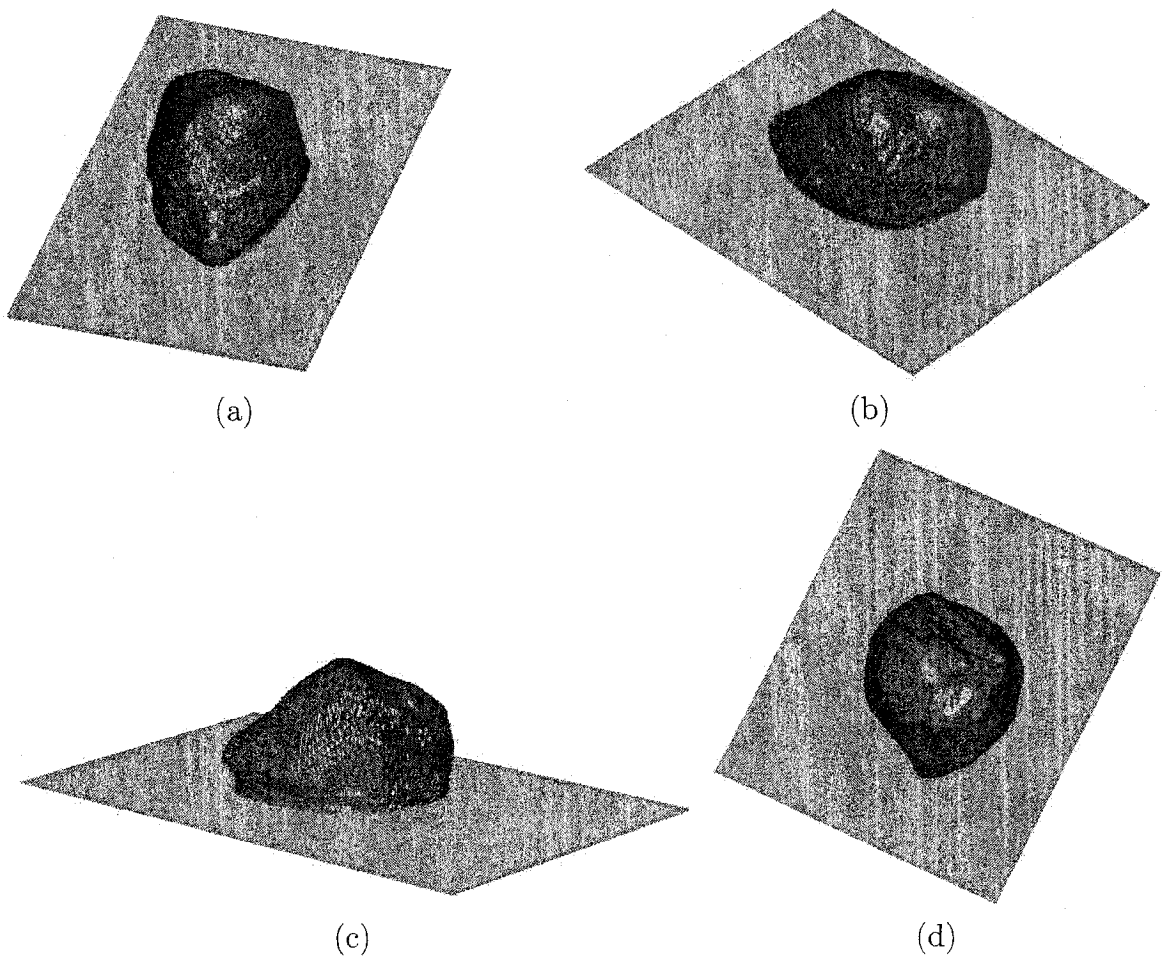


Figure 5.25: Different views of the reconstructed stone after 1500 iterations.

Chapter 6

Conclusion and future work

This chapter provides a summary and conclusion for the work introduced in this thesis.

6.1 Summary

In this thesis there computer vision problems are extensively studied and analyzed. The problems are wide baseline image matching, Simultaneous Localization and Mapping (SLAM) and 3D obstacle reconstruction. These problems are studied under the context of mobile robots sharing the same workspace. Each robot is equipped with a tilted camera and is assumed to be moving on a flat workspace on which lie obstacles. An approximate knowledge of the tilt angle with respect to the ground plane is assumed to be known. The objective is to detect and reconstruct obstacles and build a joint map of the environment.

The theoretical derivations and the techniques and algorithms introduced in this thesis contributes to the computer vision literature as summarized below:

1. In Chapter 3 I integrate different wide baseline matching techniques to solve the ground plane matching problem in the context of sparse views taken by a tilted camera. The proposed algorithm is a feature based method that depends on detecting and matching features among images collected by the mobile robot. The overall objective of the matching process is to match corners that belong to the ground plane and then calculate the inter-image homography transformation induced by this plane. This makes the algorithm capable of matching images of a scene with a featured ground plane. However, matching images of a scene with featureless ground plane remains challenging for this algorithm.
   In my proposed method, I use the overhead views in the matching process instead of the original images to compensate for the projective distortion caused by the tilted camera. The matching algorithm starts by detecting corner points on the original images and then map them on the overhead view images. Two pairs of corners between the first and second overhead views are matched based on their Hilbert's color invariants and the cross-correlation between the line segments that join the pair of corners in each image. When a matched pair is detected, the isometric transformation between the two overhead views is calculated. This transformation and the overhead view transformation are used to calculate the inter-image homography between the two original views. The inter-image homography is then refined and the motion between the cameras is estimated.

2. In Chapter 4 I experimentally show that the SLAM problem can be solved using a vision-based approach relying on ground plane matching. The inter-image homography induced by the ground plane calculated in Chapter 3 is used

to calculated the camera motion between robot locations and to build a ground plane mosaic of the work environment. In my single-robot SLAM method, the robot calculates its current location with respect to the previous one and update the map by building a mosaic of the set of images collected. The multi-robot SLAM is a generalization of the single-robot case, each robot starts to build its own local map and when an overlapping scene occurs between the two robots in the team a joint map between them can be built. To detect overlapping scenes, I present an algorithm based on histogram distances.

3. In Chapter 5 I propose a new formulation for the ground plane obstacle reconstruction that combines shape-from-silhouette and voxel coloring approaches. This problem is solved using level set formalism that minimizes an energy functional. The energy is minimized at the surface of the obstacle. My proposed algorithm is capable of reconstructing obstacles that lie below the camera. This is due to the fact that the shape-from-silhouette part is formalized with the ground plane as the background, thus the visual rays from the cameras need to intersect the ground plane during the obstacle reconstruction process. The information about the scene under reconstruction is based on the algorithms proposed in Chapters 3 and 4. The ground plane model of the environment, the equation of the ground plane and the camera motion between different robot locations are used as a pre-request before starting the proposed 3D reconstruction algorithm.

4. In this thesis I present different experimentation to validate the different algorithms used in the thesis. The matching algorithm in Chapter 3 is tested and

studied using three sets of images and it is then studied empirically to verify the complexity and the statistical analysis of my proposed algorithm. To solve the SLAM problem many histogram distance methods are studied to detect overlapping views between robots. And finally, the obstacle reconstruction method is tested on both indoor and outdoor scenes and on textured and non-textured obstacles.

6.2 Future work

The work proposed in this thesis provides a practical solution for image matching, robot localization and 3D reconstruction in a multi-robot environment. Improvement can be made in the following areas:

- In Chapter 3 the cameras system setup has a configuration as described in Figure 3.1. An improvement can be made on the system by studying the case where the camera has an arbitrary orientation with respect to the ground plane. Another improvement to investigate is the case of matching two images taken with different cameras or with different illuminations.

- In Chapter 4 an improvement may be added on view overlapping detection between robots. A good improvement is to find an algorithm where the overlapping view can be always detected in $O(1) \times C_{Homog}$ using histogram distances or other methods.

- Finally, in Chapter 5 the results presented of the 3D reconstructed objects shows the efficiency of our algorithm. A possible work to investigate is to study

the possibility of applying the algorithm on other applications in the field of computer vision and robotics.

Bibliography

[1] S.L. Altmann, *Rotations, quaternions, and double groups*, Oxford University Press, Oxford, 1986.

[2] B. Bollobas, *Graphy theory, an introductory course*, Springer-Verlag, New York, 1979.

[3] K. Bradshaw, I. Reid, and D. Murray, *The active recovery of 3d motion trajectories and their use in prediction*, IEEE Trans. on Pattern Analysis and Machine Intelligence **19** (1997), no. 3, 219–233.

[4] M.G. Brown, J.T. Foote, G.J.F Jones, K.S. Jones, and S.J. Young, *Automatic content-based retrieval of broadcast news*, ACM Multimedia conference, 1995.

[5] J. Canny, *A computational approach to edge detection*, IEEE Trans. on Pattern Analysis and Machine Intelligence **8** (1986), no. 6, 679–698.

[6] S. Carlsson and J-O Eklundh, *Object detection using model based prediction*, European conf. on Computer Vision, 1990, pp. 134–138.

[7] V. Casselles and R. Kimmel, *Minimal surfaces based object segmentation*, IEEE Trans. on Pattern Analysis and Machine Intelligence **19** (1997), no. 4, 394–398.

[8] C.H. Chian and J.K. Aggarwal, *Model reconstruction and shape recognition form occluded contours*, IEEE Trans. on Pattern Analysis and Machine Intelligence **11** (1989), 372–389.

[9] C.H. Chien and J.K. Aggarwal, *Volume / surface octrees for the representation of three-dimensional objects*, Computer Vision, Graphics, and Image Processing **36** (1986), no. 1, 100–113.

[10] Y.H. Chow and R. Chung, *Obstacle avoidance of legged robot witout 3d reconstruction of the surroundings*, Proc. of the IEEE Conf. on Robotics and Automation (2000), 2316–2321.

[11] G. Dissanayake, P. Newman, H.F. Durrant-Whyte, S. Clark, and M. Csobra, *A solution to the simultaneous localization and map building (slam) problem*, IEEE Trans. on Robotics and Automation **17** (2001), no. 3, 229–241.

[12] G. Dudek and C. Zhang, *Vision-based robot localization without explicit object models*, in Proc. IEEE Int. Conf. on Robotics and Automation (1996).

[13] C.H. Edwards and D.E. Penney, *Calculus and analytic geometry*, 2rd ed., Prentice-Hill, 1986.

[14] W. Enkelmann, *Obstacle detection by evaluation of optical flow fields from image sequences*, Image and Machine Computing **9** (1991), no. 3, 160–168.

[15] O. Faugeras, *Three-dimensional computer vision, a geometric viewpoint*, MIT Press, Cambridge, MA, 1996.

[16] O. Faugeras and R. Keriven, *Variational principles, surface evolution, PDE's, level set methods and the stereo problem*, IEEE Trans. Image Processing **7** (1998), no. 3, 336–344.

[17] F. Ferrari, E. Grosso, G. Sandini, and M. Magassi, *A stereo vision system for real time obstacle avoidance in unknown environment*, IEEE Int. Workshop on Intelligent Robots and Systems (1990), 703–708.

[18] M. Fischler and R. Bolles, *Random sample consensus: a paradigm for model fitting with application to image analysis and automated cartography*, Commun. Assoc. Comp. **24** (1981), 381–395.

[19] M. Flicker, *Query by image and video content: The qbic system*, IEEE computer, 9, vol. 28, September 1995, pp. 23–32.

[20] P. Fornland, *Direct obstacle detection and motion from spatio-temporal derivatives*, CAIP (Prague,Czech Republic), Sept 1995.

[21] A.R. Forsyth, *Calculus of variations*, NEW York:Dover, 1960.

[22] D. Geman and G. Reynolds, *Constrained restoration and the recovery of discontinuities*, IEEE Trans. on Pattern Analysis and Machine Intelligence **14** (1992), no. 3, 367–383.

[23] J.J. Gi, *The perception of the visual world*, Houghton-Mifflin, 1950.

[24] S. Ginson, R. Harvy, and J.A. Bangham, *Multi-dimensional histogram comparison via scale trees*, In Proceedings of the International Conference on Image Processing, vol. 2, Oct 2001, pp. 709–712.

[25] R. Grabowski and P. Khosla, *Localization techniques for a team of small robots*, in Proc. of the 2001 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems **3** (2001), 1067–1072.

[26] M. Grayson, *The heat equation shrinks embedded plane curves to round points*, Journal of Differential Geometry **26** (1987), no. 285.

[27] H. Hajjdiab, R. Elias, and R. Laganière, *Wide baseline obstacle detection and localization*, in Proceedings of the IEEE Seventh International Symposium on Signal Processing and its Applications (Paris-France), vol. 1, July 2003, pp. 21–24.

[28] ———, *Obstacle localization from ground plane sparse views (submitted)*, Robotics and Autonomous Systems (2004).

[29] H. Hajjdiab and R. Laganière, *Vision-based robot localization*, in Proceedings of the 2^{nd} IEEE International Workshop on Haptic Audio Visual Environments and their Applications (Ottawa,Ontario,Canada), September 2003, pp. 19–24.

[30] ———, *Vision-based multi-robot simultaneous localization and mapping*, in Proceedings of the 1^{st} Canadian Conference on Computer and Robot Vision, CCCRV'04 (London,Ontario,Canada), May 2004.

[31] C. Harris and M. Stephens, *A combined corner and edge detector*, Alvey vision Conf (1988), 147–151.

[32] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*, Cambridge University Press, 2000.

[33] H. Hatari and A. Maki, *Stereo without depth search and matric calibration*, IEEE conf. on Computer Vision and Pattern Recognition (2000), 177–184.

[34] B. Horn and B. Schunk, *Determining optical flow*, Artificial Intelligence **17** (1981), 185–203.

[35] S. Intille and A. Bobick, *Closed-world tracking*, Int. Conf. on Computer Vision, 1995, pp. 672–678.

[36] M. Jenkin and A. Jepson, *Detecting floor anomalies*, British Machine Vision Conference (York, U.K.), 1994, pp. 731–740.

[37] B. Johansson, *View synthesis and 3d reconstruction of piecewise planar scenes using intersection lines between the planes*, 7th International conference on Computer Vision (Corfu, Greece), 1999, pp. 54–59.

[38] J.Weng, N.Ahuja, and T.Huang, *Two-view matching*, Int. Conf. on Computer Vision, 1988, pp. 65–73.

[39] L.P. Kaelbling, A.R. Cassandra, and J.A. Kurien, *Acting under uncertainty: Discrete bayesian models for mobile-robot navigation*, IROS (1996).

[40] K. Kanatani and N. Ohta, *Accuracy bounds and optimal computation of homography for image mosaicing applications*, 7th Int. Conf. Comput. Vision (Kerkya, Greece), September 1999, pp. 73–78.

[41] E. Krotkov, *Mobile robot localization using single image*, in Proc. 1989 IEEE Int. Conf. on Robotics and Automation (1989), 978–983.

[42] R. Kumar, P. Anandan, and K. Hanna, *Direct recovery of shape from multiple views: a parallax based approach*, In Proc 12th Int. Conf. on Pattern Recognition **555** (1994), 111–222.

[43] K. N. Kutulakos and S.M Seitz, *A theory of shape by space carving*, International Journal of Computer Vision **38** (2000), no. 3, 199–218.

[44] R. Laganière, *Composing a bird's eye view mosaic*, Vision Interface (2000), 382–386.

[45] R. Laganière, A. Mitiche, and H. Hajjdiab, *Shape-from-silhouettes, photo-consistency, and level sets for the reconstruction of ground plane obstacles in a robot environment (submitted)*, British Machine Vision Conference (London, UK), September 2004.

[46] A. Laurentini, *The visual hull concept for silhouette-based image understanding*, IEEE Trans. on Pattern Analysis and Machine Intelligence **17** (1997), no. 2, 150–162.

[47] L. Lee, R. Romano, and G. Stein, *Monitoring activities from multiple video streams: Establishing a common coordinate frame*, IEEE Trans. on Pattern Analysis and Machine Intelligence **22** (2000), no. 8, 758–767.

[48] Y. Liu and S. Thrun, *Gaussian multi-robot slam*, submitted to NIPS (2003).

[49] H.C. Longuet-Higgins, *A computer algorithm for reconstructing a scene from projections*, Nature **293** (1981), 133–135.

[50] M.I.A Lourakis and S.C. Orphanoudakis, *Visual detection of obstacles assuming a locally planar ground*, In Proc. 3rd Asian Conf. on Computer Vision **2** (1998), 527–534.

[51] G. Lowe, *Fitting parameterized three-dimensional models to images*, IEEE Trans. Pattern Analysis and Mach. Intell, PAMI, vol. 13, May 1991, pp. 441–450.

[52] ———, *Object recognition from local scale invariant features*, Proceedings of the 7^{th} International conference on Computer Vision (Kerkyra, Greece), September 1999, pp. 1150–1157.

[53] E. Mallis and R. Cipolla, *Multi-view constraints between collineations: application to self-calibration from unknown planar structure*, European Conference on Computer Vision, June 2000.

[54] ———, *Self-calibration of zooming cameras observing an unknown planar structure*, In Proc. 15th International Conference on Pattern Recognition, vol. 1, 2000, pp. 85–88.

[55] D. Marr and T. A. Poggio, *Cooperative computation of stereo disparity*, Science **194** (1976), no. 4262, 283–287.

[56] M.C. Martin and H.P. Moravec, *Robot evidence grids*, Tech. Report 06, Carnegie Mellon University, Robotics Institute, Pittsburgh, PA, USA, 96.

[57] C. Matsunaga and K. Kanatani, *Calibration of a moving camera using planar pattern*, 5th Symposium on Sensing via Image Information(SII'99) (Yokohama, Japan), June 1999.

[58] ———, *Calibration of a moving camera using a planar pattern: optimal computation, reliability evaluation and stabilization by model selection*, Proc. 6th Euro. Conf. Computer Vision, Dublin, Ireland **2** (2000), 595–609.

[59] A. Mitiche, R. Feghali, and A. Mansouri, *Motion tracking as spatio-temporal motion boundary detection*, Robotics and Autonomous Systems **43** (2003), 39–50.

[60] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, *Fastslam2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges*, IJCAI (03).

[61] P. Montesinos, V. Gouet, R. Deriche, and D. Pel, *Matching color uncalibrated images using differential invariants*, Image and Vision Computing **18** (2000), no. 9, 659–672.

[62] H.P. Moravec, *Robot spatial perception by stereoscopic vision and 3d evidence grids*, Tech. Report 34, Carnegie Mellon University, Robotics Institute, Pittsburgh, PA, USA, 96.

[63] J.L. Mundy and A. Zisserman, *Geometric invariance in computer vision*, MIT Press, 1992.

[64] K. Murphy, *Bayesian map learning in dynamic environments*, NIPS (99).

[65] D. Murray and J. Little, *Using real-time stereo vision for mobile robot navigation*, Proceedings of the IEEE Workshop on Perception for Mobile Agents (Santa Barbara, CA), June 1998.

[66] V. Olga and M. Stonebraker, *Chabot: Retrieval from a relational database of images*, IEEE computer, 9, vol. 28, September 1995, pp. 40–48.

[67] S. Osher and N. Paragios, *Geometric level set methods in imaging, vision, and graphics*, Springer, 2003.

[68] S. Osher and J. Sethian, *Fronts propagation with curvature-dependant speed: algorithms based on hamilton-jacobi equations*, Journal of Comp. Physics **79** (1988), 12–49.

[69] G. Pass, R. Zabih, and J. Miller, *Comparing images using color coherence vectors*, In Proceedings of ACM Multimedia 96 (Boston, Ma, USA), 1996, pp. 65–73.

[70] A. Pentland, R. Picard, and S. Sclaroff, *Photobook: Content-based manipulation of image database*, International Journal of Computer Vision **18** (1996), no. 3, 233–254.

[71] M. Potmesil, *Generating octree models of 3d objects from their silhouettes in a sequence of images*, Computer Vision, Graphics, and Image Processing **40** (1987), no. 1, 1–29.

[72] A.C. Prock and C.R. Dyer, *Towards real-time voxel coloring*, Proc. Image Understanding Workshop, 1998, pp. 315–321.

[73] J. Santos-Victor and G. Sandini, *Uncalibrated obstacle detection using normal flow*, Machine vision and applications (1996), 130–137.

[74] S. Se, D. Lowe, and J. Little, *Vision-based mobile robot localization and mapping using scale-invariant features*, Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) (Seoul, Korea), May 2001, pp. 2051–2058.

[75] S.M. Seitz and C.R. Dyer, *Photorealistic scene reconstruction by voxel coloring*, IEEE conf. on Computer Vision and Pattern Recognition, 1997, pp. 1067–1073.

[76] H. Sekkati and A. Mitiche, *Dense 3d reconstruction of image sequence: a variational approach using anisotropic diffusion*, Proceedings of the 12^{th} International Conference on Image Analysis and Processing, (ICIAP'03), 2003.

[77] J.A. Sethian, *Level set methods and fast marching methods*, Cambridge University Press, 1996.

[78] R. Sim and G. Dudek, *Mobile robot localization from learned landmarks*, Proc. IEEE/RSJ Conf. on Intelligent Robots and Systems (IROS) (Victoria, BC), Oct. 1998.

[79] R. Simmons, D. Apfelbaum, W. Burgard, M. Fox, D. Moors, S. Thrun, and H. Younes, *Coordination for multi-robot exploration and mapping*, In Proceedings of the AAAI National Conference on Artificial Intelligence (Austin, TX), 2000.

[80] R. Simmons and S. Koenig, *Probabilistic robot navigation in partially observable environments*, IJCAI (1995).

[81] R.C. Smith and P. Cheeseman, *On the representation and estimation of spatial uncertainty*, Int. J. Robotics Research **5** (1986), no. 4.

[82] P.F. Sturm and S.J. Maybank, *On plane-based camera calibration: A general algorithm, singularities, applications*, In Proc. Computer Vision and Pattern Recognition, vol. 1, 1999, pp. 432–437.

[83] M. Swain and D. Ballard, *Color indexing*, International Journal of Computer Vision **7** (1991), no. 1, 11–32.

[84] L.A. Szèkely, *Crossing numbers and hard erdös problems in discrete geometry*, Combinatorics, Probability and Computing **6** (1997), 353–358.

[85] R. Szeliski, *Rapid octree construction from image sequences*, Computer Vision, Graphics, and Image Processing: Image Understanding **58** (1993), no. 1, 23–32.

[86] D. Tell and S. Carlsson, *Wide baseline point matching using affine invariants computed from intensity profiles*, European conf. on Computer Vision, 2000, pp. 814–828.

[87] S. Thrun, *Learning metric-topological maps for indoor mobile robot navigation*, Artificial Intelligence, 1, vol. 99, 1998.

[88] S. Thrun, M. Bennewitz, W. Burgard, A. Cremers, F. Dellaert, D. Fox, D. Hahnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz, *Minerva: A second-generation museum tour-guide robot*, International Conference on Robotics and Automation (Detroit, Michigan), May 1999, pp. 1999–2005.

[89] S. Thrun, W. Burgard, and D. Fox, *A real-time algorithm for mobile robot mapping with application to multi-robot and 3d mapping*, in Proceedings of the IEEE International Conference on Robotics and Automation, 2000.

[90] B. Triggs, *Autocalibration from planar scenes*, In Proc. European Conference on Computer Vision, 1998, pp. 89–105.

[91] E. Trucco and A. Verri, *Introductory techniques for 3d computer vision*, Prentice-Hall, New Jersey, 1998.

[92] R. Y. Tsai, T.S. Huang, and W.L. Zhu, *Estimating three-dimensional motion parameters of a rigid planar patch*, IEEE Acoustic Speech Signal Processing **30** (1982), no. 4, 525–534.

[93] J.F. Vigueras, M.O. Berger, and G. Simon, *Iterative multi-planar camera calibration: improving stability using model selection*, Vision, Video and Graphics (2003), 1–8.

[94] W.Enkelmann, V.Gengenbach, W.Kroger, S.Rossle, and W.Tolle, *Obstacle detection by real-time optical flow evaluation*, Intelligent Vehicles 1994 Symposium (1994), 97–102.

[95] M. Werman, S. Banerjee, S.D. Roy, and M. Qiu, *Robot localization using uncalibrated camera invariants*, TBD (1999), 353–359.

[96] D. Willersinn and W. Enkelmann, *Robust obstacle detection and tracking*, Proc. of IEEE Conf. on Intelligent Transportation Systems (1997), 325–329.

[97] T. Williamson and C. Thorpe, *A specialized multibaseline stereo technique for obstacle detection*, IEEE conf. on Computer Vision and Pattern Recognition, 1998, pp. 238–244.

[98] P.R. Wolf and B.A Dewitt, *Elements of photogrammetry with applicatioins in gis*, 3rd ed., McGraw-Hill, 2000.

[99] M. Xie, *Ground plane obstacle detection from stereo pair of images without matching*, Proc. 2nd Asian Conf. on Computer Vision **2** (1995), 280–284.

[100] G. Xu, T. Terai, and H. Shum, *A linear algorithm for camera self-calibration, motion and structure recovery for multi-planar scenes from two perspective images*, in Proc. of the 2000 IEEE/RSJ Int. Conf. on Computer Vision and Pattern Recognition **2** (2000), 474–479.

[101] A. Yezzi and S. Soatto, *Structure from motion for scenes without features*, IEEE conf. on Computer Vision and Pattern Recognition, 2003, pp. 554–559.

[102] H. Zhang, A. Kankanhalli, and S.W. Smoliar, *Automatic partitioning of full-motion video*, Multimedia Systems, vol. 1, 1993, pp. 10–28.

[103] Z. Zhang, *A new and efficient iterative approach to image matching*, proc. of Int. Conference on Pattern Recognition (Jerusalem, Israel), 1994, pp. 563–565.

[104] ———, *A new and efficient iterative approach to image matching*, ICPR (Jerusalem, Israel), 1994.

[105] ———, *Determining the epipolar geometry and its uncertainty: a review*, Tech. Report 2927, INRIA, July 1996.

[106] ———, *Flexible camera calibration by viewing a plane from unknown orientations*, In Proc. 7th International Conference on Computer Vision, vol. 1, 1999, pp. 666–673.

[107] Z. Zhang, R. Weiss, and A.R. Hanson, *Qualitative obstacle detection*, IEEE conf. on Computer Vision and Pattern Recognition, 1994, pp. 554–559.

[108] J.Y. Zheng, *Acquiring 3d models from sequences of contours*, IEEE Trans. on Pattern Analysis and Machine Intelligence **16** (1994), no. 2, 163–178.