Vision-Based 3D Motion Tracking in Natural Environments

by

Parvaneh Saeedi

M.Sc., University of British Columbia, Vancouver, Canada, 1998
B.Sc., Iran University of Science and Technology, Tehran, Iran, 1991

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
Doctor of Philosophy

in

THE FACULTY OF GRADUATE STUDIES

(Department of Electrical and Computer Engineering)

We accept this thesis as conforming
to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA
March 2004
© Parvaneh Saeedi, 2004
Abstract

Remotely controlled mobile robots have been a subject of interest for many years. They have a wide range of applications in science and in industries such as aerospace, marine, forestry, construction and mining. A key requirement of such control is the full and precise knowledge of the location and motion of the mobile robot at each moment of time.

This thesis presents a vision-based location tracking system suitable for autonomous vehicle navigation and guidance in unknown environments. The system includes a trinocular vision head that can be mounted anywhere on a navigating robot. Consecutive sets of triplet 2D images are used to determine the 3D location and the 3D motion parameters of the robot at each frame. By selecting only the most informative points of each image, using a feature detection algorithm, faster performance is achieved. The use of 3 cameras improves the accuracy and the robustness of the system. By tracking the 3D location of world features within a multi-stage tracking approach, the location of the observer camera is estimated and tracked over time. The motion with 6 DoF is found via a least squares minimization method. A Kalman filter implementation is used to optimize the 3D representation of scene features in order to improve the accuracy of the overall system.

The system introduces several novel contributions for vision-based trajectory tracking. Firstly, it presents a new binary corner detector that can automatically detect the most informative and reliable scene feature points. This feature detector performs 60% faster than the most common method, the Harris corner detector, used in vision-based tracking applications. Secondly, it compensates for an inherent source of inaccuracy in the camera geometry. By identifying and matching features in raw images, and then unwarping matched
data, accurate 3D world feature reconstruction is achieved. Thirdly, through a two-stage search and tracking design, similar 3D world feature points are identified and tracked over time. This design, by itself, improves the accuracy of the estimated motions at each time by up to 10%. Finally, it improves the accuracy of the 3D trajectory tracking system with 6 degrees of freedom in natural environments.

Results of the application of our method to the trajectory tracking problem in natural and unknown environments are reported to validate our approach, hypotheses and methods. The cumulative translational and rotational errors of the system are less than 1% for the studied examples.
Contents

Abstract
Table of Contents
List of Tables
List of Figures
Acknowledgments

1 Introduction
  1.1 Motivation ................................................. 4
  1.2 Major Contributions ....................................... 6
  1.3 Thesis Outline ............................................ 8

2 Visual Trajectory Tracking
  2.1 Landmark-Based Methods ................................ 12
     2.1.1 Navigation using predesigned landmarks ............... 14
     2.1.2 Navigation using natural landmarks .................. 15
  2.2 Natural Feature-Based Methods ......................... 17
     2.2.1 2D estimation with single cameras .................. 19
     2.2.2 3D estimation using single cameras ................ 19
     2.2.3 3D estimation using multiple cameras ............... 20
  2.3 Thesis Objective .......................................... 22
CONTENTS

3 Feature Detection

3.1 Features ........................................................................................................... 25
3.2 Corner Detection Background ........................................................................ 26
3.3 Harris Corner Detector ..................................................................................... 27
3.4 SUSAN Corner Detector .................................................................................... 29
3.5 Binary Corner Detector ...................................................................................... 32
3.6 Evaluation Metric: Repeatability Rate ............................................................... 36
3.7 Repeatability Results ........................................................................................ 37
  3.7.1 Camera noise ............................................................................................... 37
  3.7.2 Rotation ........................................................................................................ 38
  3.7.3 Scale ........................................................................................................... 39
  3.7.4 Illumination .................................................................................................. 40
    3.7.4.1 Uniform change ..................................................................................... 40
    3.7.4.2 Directional change in lighting ................................................................. 42
3.8 Discussion .......................................................................................................... 43
3.9 Computational Time .......................................................................................... 43
3.10 Chapter Summary ............................................................................................. 44

4 3D World Reconstruction

4.1 Camera Model .................................................................................................... 46
  4.1.1 Camera calibration ....................................................................................... 47
  4.1.2 Image unwarping ......................................................................................... 50
4.2 Stereo Vision Geometry ..................................................................................... 51
  4.2.1 Epipolar constraint ..................................................................................... 52
    4.2.1.1 Rectification procedure .................................................................... 53
  4.2.2 Stereo correspondence matching process .................................................... 54
    4.2.2.1 Multiple baseline .............................................................................. 55
    4.2.2.2 Matching through correlation with validation ..................................... 57
  4.2.3 Stereo correspondence matching rules ......................................................... 58
CONTENTS

4.2.4 Depth construction inaccuracy sources ........................................... 59
4.2.5 Depth construction with higher accuracy ........................................... 64
4.3 Chapter Summary ................................................................. 64

5 Tracking and Motion Estimation .................................................. 66
  5.1 Similarity Measurement .................................................. 67
  5.2 Feature Tracking .................................................. 68
  5.3 Motion Estimation .................................................. 71
    5.3.1 Least-squares minimization .................................. 72
    5.3.2 Setting up equations .................................. 74
    5.3.3 Implementation consideration .................................. 75
    5.3.4 Motion estimation results .................................. 76
    5.3.5 Feature flow results .................................. 76
  5.4 Dynamic Features and Estimation Reliability .................................. 78
  5.5 Feature Update .................................................. 78
  5.6 Feature Retirement .................................................. 80
  5.7 Chapter Summary .................................................. 80

6 Trajectory Error Modeling .................................................. 82
  6.1 Global 3D Feature Creation .................................. 83
  6.2 Camera Position Uncertainty .................................. 84
    6.2.1 Prediction .................................. 85
    6.2.2 Measurement .................................. 86
    6.2.3 Update .................................. 87
  6.3 Feature Position Uncertainty .................................. 88
  6.4 Feature Update .................................................. 89
    6.4.1 Feature covariance update .................................. 90
      6.4.1.1 Roll transformation .................................. 91
      6.4.1.2 Pitch transformation .................................. 94
      6.4.1.3 Yaw transformation .................................. 95
CONTENTS

6.4.1.4 Translational transformation ........................................ 96
6.4.2 Feature position update .................................................. 96
6.5 Experimental Results ...................................................... 97
6.6 Chapter Summary .......................................................... 99

7 Implementation Results .................................................... 101
7.1 Camera System: Digiclops™ .............................................. 101
7.2 Trajectory Estimation ...................................................... 102
  7.2.1 Experiment 1: ............................................................. 103
  7.2.2 Experiment 2: ............................................................. 105
  7.2.3 Experiment 3: ............................................................. 107
  7.2.4 Experiment 4: ............................................................. 109
7.3 Trinocular and Binocular Stereo Comparison ......................... 115
7.4 Trajectory Estimation Refinement by Kalman Filtering .......... 117
7.5 Computation Time .......................................................... 120
7.6 Chapter Summary .......................................................... 128

8 Conclusions and Future Work .......................................... 129
8.1 Major Contributions ...................................................... 129
  8.1.1 Fast feature detection .............................................. 131
  8.1.2 3D feature reconstruction with minimal positional error .... 131
  8.1.3 Multi-stage feature tracking ........................................ 132
8.2 Future Research .......................................................... 132
  8.2.1 Combined wide and narrow views ................................ 132
  8.2.2 Orthogonal views .................................................... 133
  8.2.3 Hierarchical structure ............................................... 133
  8.2.4 Sub-image processing ............................................... 134
  8.2.5 Cumulative error reduction ........................................ 135
  8.2.6 Specialized hardware ................................................ 136
CONTENTS

Bibliography .......................................................... 137

Appendix ................................................................. 149

A Location Determination Problem by Triangulation .......... 150
  A.1 Triangulation Problem ............................................ 150
  A.2 The 3D Location Determination Problem ...................... 151
List of Tables

3.1 Comparison of the time complexity of the three methods.
4.1 8.0 cm motion estimation using unwarped and warped images.
5.1 The partial derivatives table.
5.2 Iteration results along with the error residual, in pixels, for one motion estimation.
5.3 Feature numbers and comparison of the matching accuracy of the tracking process at different stages of the system for two individual experiments.
5.4 Comparison of the number of dynamic features versus estimation accuracy.
7.1 Localization error for the experiment 7.2.1 (using completely calibrated images).
7.2 System drift for Experiment 7.2.2 (using unwarped images).
7.3 3D drift in the estimated camera trajectory for a 6 meter long translation.
7.4 3D drift in the estimated camera trajectory for a 3 meter long path.
7.5 Estimated 3D drift for the camera motion with 360° rotation on a circular path with a diameter of 60cm.
7.6 Comparison of the estimated cumulative error for trinocular and binocular stereos.
7.7 Comparison of the reduction of 3D drift for a 6 meter long path using Kalman filter.
7.8 CPU time spent on various functions of trajectory tracking system.
## List of Figures

1.1 Visual motion tracking system overview. ........................................ 7

2.1 Landmark based triangulation. ....................................................... 13
2.2 (a). Top views of the landmark and projected images. (b). Estimation geometry. 15

3.1 The SUSAN corner detector. ......................................................... 30
3.2 SUSAN corner detector uses the local area information to identify corners. . 30
3.3 The block diagram of Binary Corner Detector. ................................ 33
3.4 An image and its binary sign of Laplacian image. ............................ 33
3.5 Conditions in (3.15) and (3.18) do not (by themselves) provide enough stability. 35
3.6 Illustration of different cases for a corner. ..................................... 36
3.7 Repeatability results for camera noise for neighborhood of 1.5 (left) and 0.5 (right) pixels. ............................................................. 38
3.8 Repeatability rate under rotation for neighborhood of 1.5 (left) and 0.5 (right) pixels. ............................................................. 38
3.9 Images of a scene under different rotations. ..................................... 39
3.10 Repeatability rate for small rotations a neighborhood of 1.5 (left) and 0.5 (right) pixels. ............................................................. 39
3.11 Repeatability rate for different scales in neighborhood of 1.5 (left) and 0.5 (right) pixels. ............................................................. 40
3.12 Images of a scene using different scales. ....................................... 40
3.13 Images of a scene with linear illumination change. .......................... 41
3.14 Repeatability rate under uniform illumination change for neighborhood radioes of 1.5 (left) and 0.5 (right) pixels. ........................................... 41
3.15 Images of a scene under changing illumination. ........................................... 42
3.16 Repeatability rate for changing illumination for neighborhood of 1.5 (left) and 0.5 (right) pixels. ........................................... 42

4.1 An ideal pinhole camera produces a perspective projection model of the world. 47
4.2 Camera geometry with perspective projection and lens distortion. ................. 48
4.3 a- A warped image. b- The corresponding cut unwarped image. ................. 50
4.4 Depth reconstruction using stereo geometry. ........................................... 51
4.5 Epipolar constraint. ........................................... 53
4.6 Rectification of two images. ........................................... 54
4.7 Camera positions for multiple-baseline stereo. ........................................... 55
4.8 Consistent vs inconsistent matches can be identified by validity check. ........ 58
4.9 Radial lens distortion removal can change the shape of objects of smaller size.
   a) A warped (raw) image. b) The corresponding unwarped image. ................. 60
4.10 The raw image (1). The raw image after the calibration (2). The calibrated image after the interpolation(3). The final cut unwarped image(4). ................. 61
4.11 Warped (a-1, b-1, c-1) and unwarped (a-2, b-2, c-2) images of a plane at three different positions of the image. ........................................... 62
4.12 Some important features are eliminated by the unwarping process. ................. 63

5.1 Graphical overview of the validity check in feature tracking process. The valid match candidates are shown with the solid line. ........................................... 68
5.2 Validity checking reduces the number of false match correspondences. In this figure feature points are shown in white dots. Matched features in the first frame with no validity check are shown in circles (a). Match features in the second frame are shown with arrows that connect their previous positions into their current positions (b). Matched features in the first frame with validity check are shown in circles (c). Match features with validity check in the second frame are shown with arrows that connect their previous positions into their current positions (d). .................................................. 69

5.3 The tracking is performed in two steps. .................................................. 72

5.4 Dynamic features can distract the estimated motion. ........................................ 79

6.1 Camera position Kalman filtering model .................................................. 88

6.2 Positional uncertainties associated with features in the image plane ...................... 97

6.3 a- Projected relative positional variances of features with distances up to 3.5 meters from the camera on the XZ plane. b- Top view of positional uncertainties for all features .................................................. 98

6.4 Original and updated positional uncertainties of world features ................................ 99

7.1 Digiclops stereo camera system .................................................. 102

7.2 The indoor scene from camera point of view .................................................. 103

7.3 The graphical presentation of the 3D camera motion while moving from point A to point B, described in Experiment 7.2.1, using Visual Toolkit 4.0. In this figure the position is shown with a red sphere and the orientation with a yellow cone .................................................. 104

7.4 3D position and orientation of the mobile camera in experiment 7.2.2 using VTK 4.0 .................................................. 105

7.5 The outdoor scene for experiment 7.2.3 .................................................. 107

7.6 Graphical 3D representation of the motion using VTK 4.0 ........................................ 110

7.7 The graphic representation of the traced path in Experiment 7.2.3 using VTK 4.0 ............. 111
### LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.8</td>
<td>a-The outdoor scene used for the rotational motion. b-An up close view of the same scene from robot’s point of view.</td>
</tr>
<tr>
<td>7.9</td>
<td>A number of images from the sequence that is used in Experiment 7.2.4.</td>
</tr>
<tr>
<td>7.10</td>
<td>Graphical presentation of the 3D estimated camera motion on a circular path with a yaw rotation of 360° using VTK 4.0.</td>
</tr>
<tr>
<td>7.11</td>
<td>A closer view of the estimated circular path with a radius of 30cm that is traversed by the camera system in Experiment 7.2.4.</td>
</tr>
<tr>
<td>7.12</td>
<td>Selected number of images viewed while traveling on a path. The motion of an object is highlighted using the yellow circle.</td>
</tr>
<tr>
<td>7.13</td>
<td>Estimated traveled distance along x[cm] for a 1m closed loop.</td>
</tr>
<tr>
<td>7.14</td>
<td>Estimated camera distances relative to a starting point for a 6 meter long path with Kalman filter.</td>
</tr>
<tr>
<td>7.15</td>
<td>Estimated camera orientations for a 6 meter long path with Kalman filter on.</td>
</tr>
<tr>
<td>7.16</td>
<td>Estimated camera trajectories relative to a starting point for a 6 meter long path without Kalman filter.</td>
</tr>
<tr>
<td>7.17</td>
<td>Estimated camera orientation for a 6 meter long path with no Kalman filtering.</td>
</tr>
<tr>
<td>7.18</td>
<td>System’s modular flowchart for the time analysis.</td>
</tr>
<tr>
<td>7.19</td>
<td>Comparison of the running time versus search window dimensions for the tracking module.</td>
</tr>
<tr>
<td>8.1</td>
<td>A second set of camera overcomes the coupling effect between translation and rotation.</td>
</tr>
<tr>
<td>8.2</td>
<td>Processing small image patches can reduce the processing time.</td>
</tr>
<tr>
<td>A.1</td>
<td>The intersection of the two circles using two pairs of viewed landmarks provides the observer location.</td>
</tr>
<tr>
<td>A.2</td>
<td>The locations of three landmarks along three image projectors must match the distances between the same points.</td>
</tr>
</tbody>
</table>
Acknowledgments

My Graduate studies at UBC can be best described by the words of John Lennon: “Life is what happens to you while you’re busy making other plans.” Starting with the idea of getting a Ph.D., I formed ties with people that have and will influence my life forever.

My thanks must go to my supervisor, professor Peter Lawrence, for his patience and guidance over all these years. He has been very supportive and a continuous source of encouragement during my PhD. My thanks also go to my co-supervisor, professor David Lowe, for introducing me to computer vision. I would have floundered long ago were it not for his genuine interest in my work and his willingness to answer my constant questions. I would also like to thank the committee members, university and external examiners for their invaluable time and advice in the improvement of this thesis. Special thanks also go to Don Murray and Simon Bachman for their technical help through my entire work. I would like to thank Danny French for his priceless help with outdoor experiments. Finally, I would like to thank the ECE office administration members, IT staff, and workshop technicians.

I would like to express my deep gratitude to my parents and my sister for their love and trust, and for encouraging me to pursue my interests. My sincerest thanks go to my good friends for listening to my thoughts during the best and worst periods of my graduate studies. A special thanks extend to the IRIS Network of Centers of Excellence for their financial support.

Parvaneh Saeedi

*The University of British Columbia*

*March 2004*
Chapter 1

Introduction

Vision, as one of our most important senses, provides us with a vast amount of information. Fundamentally, our vision system helps us to identify structures in an environment and to locate ourselves within it. The inherent difficulty in visually determining observer and object locations in an environment originates from the fact that optical projection is a destructive transformation. This is mainly because the projected images on the retina (camera's image plane) are purely 2D and contain very little 3D information. Construction of 3D information from 2D images, therefore, is an ill-posed problem necessitating additional assumptions and constraints. The extraordinary proficiency of the human visual system is a sign that it is possible to solve the inverse projection problem for machine vision.

Computer vision, the inverse process of image formation, attempts to construct meaningful and explicit descriptions of the world from images. The principal aim of computer vision is to take an image or sequence of images from a scene and process them in such a way as to infer specific information about the scene. In this sense, vision is a very challenging problem because it is full of uncertainties. Computer vision is becoming increasingly important as an essential technology for a broad range of applications, such as industrial automation and inspection [11,37], medical diagnosis [43,68], human-computer interaction [20,102], virtual reality [19,72], and autonomous navigation [83,87,101].

One of the principal subjects in computer vision is the visual tracking problem. Visual
tracking relates to the problem of following objects, markers or paths for many purposes, such as the behavioral study of specific species in science [26, 109], or automation in industrial applications [73]. In this group of tracking applications the aim is to constantly have a target in the field of view, or to move toward a landmark where specific aspects of it can be observed. Often a sequence of landmark targets serve as a route that the robot can traverse for this group of applications. Here, estimating the motion vector, however, is not the goal but the tracking task itself.

Another group of applications where visual tracking is used comprises trajectory and motion estimation for mobile robots from image sequences, acquired with an on board camera system. Usually, the objective of this group is localization for the purpose of autonomous navigation. Autonomous navigation is required for a wide range of applications, spanning from scientific research and industrial automation to entertainment and military purposes [6, 7, 50, 79, 81]. The localization task here is performed first by detection and identification of static scene structures and objects. The positions of these structures are either estimated or provided as system inputs. The relative or absolute position of the observer system is then estimated, either by measuring the relative changes in the position of these structures from one frame to another, or by a geometric approximation model.

Different assumptions and constraints made during visual tracking provide the bases for different approaches. Some of assumptions that have been used are these:

- The scene is constructed mostly from static objects, and if it does undergo any changes, appropriate adjustments are made to accommodate new changes.

- The degrees-of-freedom of the motion are often limited to a plane.

- Multiple sensors may be required.

- Previous methods have often assumed that the environment is known in advance, or at least some predefined landmarks with specific characteristics at known positions in the scene are identified.
• When known landmarks are used, a number of them must be observable at each time during the system operation.

Each one of these constraints and hypotheses could reduce the applicability domain and the generality of the associated tracking system. While some of these assumptions might be valid for indoor and small to medium size environments, they are not practical for natural terrains or unknown environments.

Perhaps the most restrictive of all these assumptions is the assumption of prior knowledge of the environment or objects within it. Existing methods in trajectory and motion tracking can be partitioned, depending on the amount of prior information about the environment and the model that is attempted to capture. In methods based on prior information, a topological map of the environment is facilitated by placing landmarks within it and including the expected visibility region of each. The positions of these landmarks are either provided to the system or are found during an initialization process. Most commonly, these landmarks are artificial and are designed with the intention of achieving fast, accurate and unique recognition. The position of the observer robot at each time is found by positional triangulation of the observed landmarks. The most important requirement for systems based on landmarks is the observation of two or more landmarks at each time.

In the absence of any prior information, vision based systems either recover the full geometric structure of the scene or attempt to reconstruct a partial representation of the scene through the use of salient points or other naturally occurring features of the scene. The 3D location of features or scene structures are computed through the use of stereo cameras, laser range finders, multiple cameras with known geometry, or monocular cameras. Generally, the focus of these techniques is on the exact localization and recovery of the relative pose of the robot, with respect to its environment. This is achieved by finding the optimal transformation that can project instances of identical features from one instant of time to another. Ideally, such systems allow arbitrary camera motion and only require most of the scenes to be static.
1.1 Motivation

Existing operational trajectory tracking systems are mostly based on fusing a combination of sensors, such as laser range finders, sonar sensors, GPS, prior maps, artificial landmarks and wheel encoders. Each of these types of sensor modalities has several shortcomings that make it insufficient by itself. For example, not only can laser range finders be expensive and slow, but they are also very sensitive to outdoor elements such as dust, rain and snow. GPS systems have the problem of the obstruction of line-of-sight to satellites, and they may suffer from active jamming by other radio frequency sources. Sonar sensors suffer from specular reflections on smooth surfaces. Employment of artificial landmarks for large scale outdoor environments is also often hard if not impossible and the maintenance costs can be high. Moreover, acquiring prior information about an unexplored or dangerous environment is not possible. Dead reckoning sensors may generate a great deal of uncertainty as they are mostly designed for smooth and man-made surfaces, and therefore, their readings for rough and uneven outdoor surfaces are unreliable. For instance, simultaneous localization and mapping (SLAM) has been the core of many successful indoor robot systems. SLAM allows map generation and localization within large consistent environments. Trajectory and tracking systems based on SLAM generally assume that the unknown environment is static and contains only rigid, non-moving objects. Se [87] represents a SLAM-based system using a Scale Invariant Feature Transform (SIFT) [61] for indoor environments using odometers and a stereo camera system. Cobzas introduced a system using 3D vertical lines that are obtained by laser range finders [25]. SLAM-based systems use multiple readings of several kinds of sensors such as odometers, laser range finders, ultrasonic sensors and camera systems.

As presented earlier, existing vision-based motion and trajectory tracking systems often impose a number of constraints that confine them to specific applications or limit their performance. As a result, most of the existing systems generally do not perform well enough in outdoor applications, largely because they are not designed robustly enough to deal with the irregularities of natural environments. Therefore, robust, autonomous, long-range navigation for natural terrains or unknown environments, based on machine vision, remains an
unsolved problem in robotics.

This research is therefore motivated to address some of the limitations of vision-based trajectory and motion tracking problems. The key deficiencies in many different solutions to visual trajectory and motion tracking solutions can be identified as the following:

- **Accuracy**: cumulative error makes system output unreliable, as often their drift increases over time.

- **Robustness**: different environmental and imaging conditions can influence the reliability and the robustness of the estimated solution.

- **Scene dependence**: dependence on a unique environment makes these systems inflexible and less useful.

- **Cost**: the use of expensive and sophisticated sensors is not affordable or practical for smaller applications.

- **Real-time performance**: real-time performance adds to the systems' reliability in dealing with abrupt changes or unexpected events.

This thesis presents a completely vision-based trajectory estimation system for outdoor and unknown environments. At each time frame, the system provides the 3D trajectory of the navigating robot. The system includes an inexpensive trinocular set of stereo cameras that can be mounted anywhere on the robot. It employs existing scene information and requires no prior map, nor any modification to be made in the scene. The system focuses on the improvement in accuracy by removing any noise introduced by sensory devices or other processes in the system. Special attention is also paid to the problems of robustness and reliability, in different environmental and imaging conditions. The main assumptions here are that the scene provides enough features for matching and that most of the scene objects are static. Moreover, it is assumed that the velocity of the robot is limited in such a way that there is some overlap between each two consecutive frames. The system is mainly
1.2 Major Contributions

The major contributions of this thesis can be summarized as follows:

1. A novel binary corner detector for choosing the most informative scene features in a fast manner.

2. A new approach for 3D world reconstruction that not only minimizes the lens radial distortion, but also removes unwanted added distortion that is introduced by the lens distortion removal algorithm.

3. A multi-step scheme for the motion estimation process, in which a rough motion estimation facilitates a more precise tracking implementation, resulting in the elimination of more outliers at earlier stages.

4. Improvement of the robustness and accuracy of the general trajectory tracking problem for outdoors, under different environmental conditions with no prior information about the scene. The translational and rotational accuracy is improved.
1.2 Major Contributions

Figure 1.1: Visual motion tracking system overview.

yielding a value of 1% with paths as long as 6 meters and with rotations up to 360 degrees.
1.3 Thesis Outline

This thesis is concerned with the design and implementation of a vision-based system for tracking the 3D trajectory and motion of a robot in an unknown environment using an on-board camera system. It consists of eight chapters as follows.

Chapter 2: Visual Trajectory Tracking. In this chapter the background of visual motion and trajectory tracking problems is explained. Different approaches to the tracking problem, along with examples from each are represented later. The performances of these systems are compared based on their sensory devices, accuracy, applicability and cost. The approach selected in this thesis is also justified in this chapter.

Chapter 3: Feature Detection. An overview of some common feature detectors is presented in this chapter. The basis of a novel binary corner detector, that is developed for this work, is explained later. The performance of the suggested method is then compared with the most common feature detector, the Harris corner detector, and the original inspiration for this work, the SUSAN corner detector.

Chapter 4: 3D World Reconstruction. This chapter explains the conventional method for reconstructing the 3D world, using the stereo algorithm first. We take a slightly different approach to the 3D world reconstruction problem in which the positional uncertainty resulting from the lens distortion removal process is minimized. Through this approach, every single pixel of the acquired images is processed; traditionally, some of this information is discarded during the lens distortion removal process, and therefore a wider scene is processed.

Chapter 5: Feature Tracking and Motion Estimation. A multiple stage approach for tracking world features is represented in this chapter. By taking this direction, improvements are gained in the accuracy of the overall motion estimation by means of more accurate match correspondences and a lower number of outliers. The motion estimation process involving the calculation of image Jacobians as well as the least squares fit solution to the problem, are also explained in this chapter.

Chapter 6: Trajectory Error Modeling. This chapter starts with a discussion of
the inherent positional uncertainty associated with each feature point. It continues with the study of a Kalman filter scheme for each feature that is used for combining positional information and uncertainties over time. Derivation of the formulation for propagating the existing uncertainty in the estimated motion to feature positions concludes this chapter.

**Chapter 7: Experimental Results.** A study of the implemented system performance, accuracy and cost, is represented through several examples for indoor and outdoor environments in this chapter. A discussion of some practical considerations as well as time issues for implementing the system are also represented in this chapter.

**Chapter 8: Conclusions and Future Work.** The contributions of this thesis, along with suggestions for future research, are summarized in this chapter.
Chapter 2

Visual Trajectory Tracking

In visual motion and trajectory tracking, the relative motion between objects in a scene and the camera is determined through the apparent motion of objects in a sequence of images. This motion may be characterized or measured through observing the apparent change in brightness patterns in the images or motion of a discrete set of scene structures. Visual trajectory tracking approaches can be divided into two different approaches:

- Motion-Based Methods
- Feature-Based Methods

Motion-Based Methods

Motion-based methods detect motion through optical flow tracking and motion-energy estimation.

- **Optical Flow-Based Methods**: Optical flow tracking operates by extracting the velocity field, assuming that image intensity is a continuous function of time. Methods based on optical flow are fast and can be used for motion detection by any unconstrained platform; however, they cannot be used where the camera motion is more
than a few pixels between successive frames, thus limiting the pan/tilt angles to small changes [92].

- **Motion-Energy-Based Methods**: Motion-energy tracking is based on calculating the temporal derivatives of images, which are then thresholded to eliminate noise. The resulting image is segmented into regions of motion and inactivity. Methods based on temporal derivatives are simple and suitable for parallel computing architectures, but are subject to noise, leading to imprecise values. In addition, pixel motion is detected but not quantified [70].

**Feature-Based Methods**

Feature-based methods recognize an object or objects (landmarks or scene structures) and extract the position in successive frames. The 2D change in the object in the image can be due to either a change in its position in the scene, or to a new aspect of the object. Therefore, by tracking an object over a sequence of images and measuring the similarities, the motion of the navigation system can be retrieved. While the performance of these approaches can be limited by the efficiency of the recognition method, object type, and positional accuracies, they have the advantage of 3D performance capability in non-stationary environments with higher accuracy than motion-based methods. Work on the feature-based camera localization problem can be divided into two general domains:

- **Landmark-Based Methods**: Landmark-based methods detect distinguishable known structures (landmarks) in the scene and through the geometric interpretation of the known position of these structures, obtain the position of the camera. Tracking with these methods might be aided by an *a priori* map of the environment.

- **Natural Feature-Based Methods**: Natural feature-based approaches attempt to track projection of preliminary and more general features of a scene in a sequence of images. By tracking and finding relative changes in the position of these features in images, it is possible to find the trajectory and motion of the navigation system.
As described in the previous chapter, precise measurement of the motion and trajectory of a mobile navigation system is the main goal of this thesis. Therefore, feature-based approaches seem to be more suitable for this application. A brief study of the previous work in vision-based navigation systems is presented in the following sections.

## 2.1 Landmark-Based Methods

A landmark is defined as any detectable structure, marking or object with a known position in a physical environment that can be recognized in an image. A landmark can be a simple vertical line, a specially designed mark, such as a cross or a pattern of concentric circles, or even a known object in the scene, such as a building or an indoor ceiling lamp. Landmarks can be categorized into two types:

- **Predesigned landmarks.** These artificial landmarks enrich the environment with simple and distinguishable geometric patterns.
- **Natural landmarks.** If the workspace is known in advance, it might be possible to manually select objects to constitute a set of known landmarks.

A system based on landmarks usually requires a training phase in which the landmarks and their corresponding information, such as world position and physical attributes, are extracted to construct a database. Motion tracking proceeds by detecting landmarks, followed by the camera position estimation based on triangulation. Triangulation for robot localization is based on traditional methods in cartography and navigation, that use angles measured between lines of sight to known landmarks in the environment. There has been extensive research on these methods in the domains of cartography, photogrammetry and computational geometry. Anderson et al. [2] categorized the triangulation methods into three groups, based on the number of landmarks:

I. Triangulation based on one landmark, views one landmark ($LM$) from two viewpoints; see Figure 2.1(a). The observer position ($P_2$) is determined using the initial location
(P₁), or the traveled distance (b). The main assumption here is that the observer's displacement occurs parallel to the image plane.

II. Triangulation using two landmarks from two viewpoints is similar to using a single landmark with the same assumption and measurements. The additional landmark, however, constrains the position estimation process, thus vastly reducing the noise sensitivity of the process; see Figure 2.1(b).

III. Triangulation via three landmarks from the same viewpoint relies on the world positions (LM₁, LM₂, LM₃), and the angular separations (ϕ₁, ϕ₂) of the landmarks, to recover the observer's position as demonstrated in Figure 2.1(c). More details about 2D and 3D localizations using triangulation can be found in Appendix A.

Several parameters influence the result of the triangulation procedure, including the camera's focal length, the world position of the landmarks and the odometeric readings, and/or the initial position estimation. A study of these methods in [2] shows that their efficiency and robustness is proportional to the number of viewed landmarks. Sutherland and Thompson [98] and later Betke and Gurvits [13], studied triangulation methods to show the effect of landmark configuration on the size and shape of the area of triangulation uncertainty. In this work, landmark-based navigation methods are classified according to the type of landmark. A study of several examples in this class is presented next.

Figure 2.1: Landmark based triangulation.
2.1 Landmark-Based Methods

2.1.1 Navigation using predesigned landmarks

In these types of systems, a predesigned landmark is placed at different but known locations in the environment. Reliable position estimation with this type of approach depends on bounding the uncertainty existing in initialization, recognition and identification of such markers. Simple geometric models such as circles, ellipses or polygons often are used as artificial landmarks because they can be detected easily in real-time using edge detection algorithms. Most of the previous works have adopted planar designs when creating landmarks [12,21,33,56,93,112]; however, better accuracy may be obtained by using 3D artificial landmarks. For instance Bao et al. [5] describe a system for estimating the position and orientation of a vehicle using 3D artificial landmarks. They show a significant improvement of orientation accuracy can be achieved by replacing 2D landmarks with 3D ones. Several systems are presented for indoor navigation using 3D geometric landmarks [53,99]. They emphasize landmark issues such as high information content, generality under different viewing conditions, and fast extraction.

Generally, in these types of systems, circular patterns are the more popular among the other geometric shapes because their projection in the image plane can be approximated by an ellipse, and they do not usually get confused with the majority of patterns frequently seen in the scene. Also, circular patterns are more robust regarding noise and occlusion than polygonal patterns during the matching process. For instance, one algorithm employs 3D artificial landmarks, introduced by Lin and Tummala [58], in which the landmark is a dark circular disk on a white background, with a secondary, smaller, white, concentric disk (see Figure 2.2(a)).

The relative position and heading of the robot can be estimated using corresponding parameters of the ellipses observed in the image, as illustrated in Figure 2.2(b). The major disadvantage to these approaches is that they require substantial engineering of the environment, and therefore, they are not suitable for changing or unexplored environments. Moreover, these methods are mostly limited to planar estimates. The choice of the landmark type might also add some limitations to the tracking system. Sometimes, the cost of the cre-
2.1 Landmark-Based Methods

Figure 2.2: (a). Top views of the landmark and projected images. (b). Estimation geometry.

ation and maintenance of beacons might not be practical, and sometimes, finding one-to-one correspondence between objects in the workspace and their projections in the image plane is difficult in the presence of noise. Also, as the detection of each landmark in an image requires an approximation for the perspective projection of that landmark, the accuracy of such systems can be influenced by a poor approximation.

2.1.2 Navigation using natural landmarks

Much of the recent work in landmark-based navigation attempts to avoid the requirement for artificial landmarks or domain-specific features. This is mainly due to the restrictions that artificial landmarks generally place on the types of environment that can be explored. Not only do they restrict camera motion and pose, but also their calibration in the environment is usually an onerous task. This has given rise to methods that are not based on predesigned landmarks; rather, they automatically extract naturally-occurring landmarks via a local distinctiveness criterion from the environment during a learning phase. Such systems usually require a priori map of the environment that is either provided or created during a learning phase.

Over the past few years, many vision-based localization systems using landmarks have been developed. They mainly differ in the features they employ. The distinctiveness criterion
for feature selection vary based upon the application and the navigation environment. Many of these methods focus on using structured objects, such as doors, windows and straight lines. For instance, Munoz and Gonzalez [69] develop a system based on early work by Krotkov [55] that employs vertical line models. The system fuses a single camera's information, a two dimensional map including landmark positions and attributes (vertically oriented parts of fixed objects such as doors, desks and wall junctions) and odometer readings. Another work is explained in [9] where lines and edges are extracted from reference images in a learning phase. For each reference image a set of geometric models is created using extracted features. Then, the rough position of the robot is estimated by applying geometric transformations that fit the extracted data to the reference image models. In another work, Trahanias et al. [103] develops a system to find 2D camera motion by employing natural landmarks of the walls. During the learning phase distinct features on the walls are chosen using statistical measurements and placed into a navigation map. Sim and Dudek [94] introduce a method in which landmark candidates are selected, based upon the extrema of the edge distribution density, to construct a representation of the environment in the form of a database. The position estimation is achieved by extracting image landmark candidates and matching them with the tracked landmarks in the database, and then, by linear position interpolation of the match correspondence candidates. In another approach [54], vertical lines are extracted from camera images and are combined with data obtained from ultrasound sensors to estimate the position and orientation of the robot.

As demonstrated by these examples, such approaches have had success in finding 2D motion tracking for indoor environments, where structured landmarks are more present and where the readings of an odometer sensor are more reliable. Such systems, however, become less successful in outdoor applications. This is because odometer readings in environments with rough and uneven surfaces are less accurate. Moreover, outdoor scenes usually include less structured objects and characteristics, and the shapes of these objects can easily vary under different or changing environmental or imaging conditions. One system that attempts to achieve natural target tracking for an outdoor environment is proposed by Murrieta-Cid et al. [71]. In their system, principal regions of the scene are extracted by color segmentation
and are characterized by their color and texture using statistical information. These principal regions compose a database which includes different environmental classes such as grass, trees, soil, sky and rocks. The tracking task is carried out by comparing image segments, extracted from image sequences, with database models [44].

The major advantage of these landmark based approaches is that they have a bounded cumulative error. Moreover, their simplicity and the fact that they do not require any 3D reconstruction of scenes have made them very popular. However, these systems require some knowledge of the geometric model of the environment, either built into the system in advance, a prior map, or acquired using sensory information during movement, the learning phase, or sometimes a combination of both. This requirement seriously limits the approach’s capability in unknown environments. Moreover, the motion of the robot is limited along one or two axes and the orientation is confined to a field in which the landmark(s) is visible. These restrictions make such approaches inappropriate for environments where motion involves more degrees of freedom. Furthermore, the underlying assumption is that the robot can always reliably detect a landmark in the sensor data. This is a fundamental assumption, critical for the reliability of the whole system. One of the important aspects of all of these solutions is that sensor measurements are not always accurate, and therefore, it is more reasonable to seek a solution that minimizes the uncertainty of the position estimate.

### 2.2 Natural Feature-Based Methods

Natural feature-based tracking methods take advantage of the fact that as the camera moves in an environment, changes are usually related to each other in the pattern of image intensities in short time intervals. This correlation shows that patterns move in an image stream as the camera moves in the environment [90]. If \( I(x, y, t) \) represents the image intensity function, for a simple translational motion of the camera system, formally the following equality holds:

\[
I(x, y, t) = I(x + \xi, y + \eta, t + \tau)
\] (2.1)
2.2 Natural Feature-Based Methods

Therefore, the local displacement between pixels in subsequent 2D images corresponds to relative 3D motion between corresponding points in the scene. Through the measurement of such displacements, the 3D location of these features, as well as the motion of the camera, can be determined. One important issue involved with finding these displacements is that a single pixel-sized scene element, unless it has a very distinctive brightness with respect to each of its neighbors, is the fact that they can be confused with adjacent scene elements. A solution to this problem is the use of preliminary features of the scene with maximum local stability under projective transformations. Therefore, the objective of natural feature-based tracking methods becomes to derive the motion of features in a scene through the analysis of the motion of corresponding features in the sequence of images. Common local structural features extracted from images are points, lines, statistical features, surfaces and geometric patterns. The fact that these features are usually not scene-dependent makes them very popular for tracking applications in unknown environments.

The type of feature is highly dependent on the working environment that the system is designed for. For instance Wilson et al. [110] uses object corners, centroids and areas of holes in an object as local features. The centroid and diameter of circles are used by Harrell et al. [38] for a fruit tracking robot for harvesting. Rives and Borrelly [78] employ edge features to track pipes with an underwater robot. The road-following vehicle of Dickmanns et al. [29] is also based on edge tracking. In another application, edge contours are tracked in real-time [111] for manipulation of free-floating objects by a space robot. Vanishing points (the intersection of two nearly parallel lines) and line orientations are used by Zhang et al. [116] for robot navigation. Chaumette et al. [22] and Espiau et al. [30] derive variations of a tracking method for points, circles and lines.

The main advantage of using local features is that they correspond to specific physical elements of the observed objects, and once correctly located and matched, provide very accurate information concerning the relative position between camera and scene. Also, in systems based on landmarks or models, it is possible that no landmark is visible, so the motion estimation cannot be accurate for some percentage of the time, while estimations
Natural Feature-Based Methods

Based on scene features are potentially less likely to fail due to the large number of features that can be available from any point of view. The accuracy of these methods, however, is highly dependent on the accuracy of the features. Even a small amount of positional uncertainty can eventually result in a significant trajectory drift.

The previous work in natural feature-based tracking can be categorized based on the estimation's degree of freedom and the number of employed cameras.

2.2.1 2D estimation with single cameras

Due to limited processing and sensory technologies, early work in the natural feature-based motion tracking area is limited to planar motions. Sethi and Jain [88] describe a method for finding 2D trajectories of feature points in monocular images by employing an iterative optimization scheme based on constraints on the direction and magnitude of the motion. Wang and Clarke [108] implement a 2D general motion estimator using Fourier descriptors of object contours. Soroushi [96] implements a real-time, natural feature-based trajectory tracking system for planar motion of a down-looking monocular camera in an unknown environment, using a coarse-fine registration algorithm. Betke et al. [14] develops a real-time vision system that analyzes color videos of a highway to recognize and track road boundaries and vehicles. The boundaries of the road and the outline of the vehicle are found using edges, color, and saturation contents of video images. Etoh and Shirai [31] address a system that segments input images into fragments, characterized by their color, pixel position and intensity gradients. The planar motion parameters are estimated precisely by tracking clusters of fragments over image sequences.

2.2.2 3D estimation using single cameras

Later attempts are directed toward 3D tracking using monocular images. For instance, Basri et al. [10] presents a feature-based navigation system that moves a 3D robot arm to a desired location in space. The destination information is provided by a single image taken
2.2 Natural Feature-Based Methods

at the goal position. Through the matching of a minimum of eight corner correspondences between two consecutive frames, the rotation matrix between the two images is found first [8]. Employing a multi step approach, the camera rotates and then translates until it reaches the goal position. Irani et al. proposes a 3D motion estimation approach for static scenes in [46]. They use the particular behavior of points on planar surfaces to register patches on consecutive frames first, and to derive 3D translation and rotation vectors of the camera motion later. Harris introduces a monocular vision system that uses the visual motion of image corners to create a 3D geometric model of the viewed scene [39]. In his work he finds 3D locations of features through an egomotion process. Successive frames of a moving camera are then processed and the motion of the camera, as well as 3D position of features in the consecutive frames, are estimated by an optimization algorithm. Leung et al. [57] proposes a constrained 3D motion estimation method using lines over monocular images. In his method lines are identified from a set of edge correspondences and their mathematical equations are computed. The lines are tracked over three consecutive frames, and the motion of the camera is estimated using an optimization scheme on a triplet set of correspondences.

2.2.3 3D estimation using multiple cameras

The main problems with the systems described above are poor overall motion estimation, limited motion, small range tolerance, and long term cumulative error. Several attempts are made to increase motion range, accuracy and robustness through the employment of images from multiple views. Through orthogonal or slanted views, 3D position estimation and alignment are achieved with higher accuracy. Joshi and Sanderson address an application of feature-based visual tracking for three dimensional filament alignment, 5 degrees of freedom (DoF), using two orthogonal cameras [47]. Argyros and Fredrik describe a vision system [3] that allows the robot to move in the middle of the free space by exploiting a forward-looking camera and two side-looking slanted cameras for the peripheral visual fields. Zhang [113] implements an algorithm to estimate the 3D motion of line segments in space using two perspective images taken by two calibrated cameras. With the assumption that the matched
line segments contain the projection of a common part of the corresponding line segment in space, he can estimate the relative motion between the two cameras. The main problem with these techniques however is the existing ambiguity with respect to the absolute value of the distance between the camera and the scene [1].

Later research in natural feature-based 3D motion and trajectory estimation addresses stereo vision analysis. The use of stereoscopy uniquely determines depth and hence absolute values for motion parameters. Wettergreen et al. [109] embark the development of autonomous underwater vehicles for performing automatic sub-sea searches by tracking dynamic or fixed targets. The system uses area-based features and is equipped with a stereo vision system that was designed specifically for tracking targets and extracting their position, orientation and velocity. Thorpe et al. [100] successfully develops a real-time autonomous driving system, Navlab, to navigate on highways with the aid of a map, a laser range finder and a stereo camera system. The visual tracking module employs a variety of perception methods and techniques, selected depending upon road and lighting conditions. Zhang and Faugeras [115] propose an approach using 3D line segments obtained from stereo images. Tracking is performed using a prediction, matching and update loop and the motion was estimated using an extended Kalman filtering scheme. Shieh et al. [91] introduces a method to find the relative motion of an object with respect to a moving camera using stereo and image brightness information. The main assumption here is that the motion of the object is limited to a very small range. Although they show that their approach improves accuracy, the reported error is still very large. Mark et al. [106] represents a vehicle ego motion estimation system using stereo and corner features based on the Harris corner detector [40]. The major problem with their work is reported to be the accuracy. Moreover, the results are represented only for simulated and synthetic data.

Pedersini et al. [75] describes a method for egomotion estimation using 3D straight and curved contours. The motion is first approximated through the optimization of a 3D transformation that maps the 3D straight line correspondences, and it is refined by adding 3D curved contour correspondences to the optimization process. Demirdjian and Darrel [27]
address a system using correspondences in disparity images. They show that working in a projective disparity space where the error associated with the image points are precisely known can help to reduce the disturbing noise effect on the final estimated motion. Another stereo-based 3D motion estimation is addressed by Goncalves and Araujo in [35]. An estimation of the velocity along the direction perpendicular to the image plane is used to recover complete motion parameters. The analysis of the results shows that the suggested method is very path dependent, and generally suffers from an average error of between 10% and 30%.

The main advantage of natural feature-based systems are reduced complexity, as only a limited set of features are processed at each time, and applicability to more general environments. However, as represented, central problems for these methods are the accuracy and robustness of the estimated motion. The performance of these systems is highly dependent on an accurate feature correspondence establishment/tracking [59, 105, 117], and the resolution of the optical system. These become serious issues when estimating and tracking trajectory over a long sequence of images. Therefore, while these type of systems can perform reliable real-time 3D object tracking, they may drift in long term trajectory tracking.

## 2.3 Thesis Objective

The design presented in this thesis is an exploration of the relevant issues in creating a real-time on-board motion tracking system, for a natural environment using an active camera. The system is designed to be as general as possible, but there are some necessary assumptions to be made, as some uncertainties and exceptions do exist. The following is the list of assumptions that we have undertaken in our approach:

1. Much of the current vision research is concerned with stationary objects, since non-stationary objects are very difficult to handle without prior knowledge of their behavior. Thus, we assume that our scene includes mostly stationary objects. If there are a few moving objects, the system is able to rely on static object information, while
Thesis Objective

information from moving objects can be discarded as statistical outliers.

- The camera characteristics are known. In particular, focal length and the base line separation of the stereo cameras should be provided.

- The motion of the robot is assumed to be limited in acceleration. This allows the match searching techniques to work on a small and predictable range of possible matches.

- There is no specific knowledge about the unstructured scene which is to be processed. The working environment is assumed not to be a uniform scene and includes a number of objects and textures. There is also the assumption that at most only 40% of the scene objects are moving.

The ultimate goal of this thesis is to define a solution to cover the following objectives:

- Accuracy, simplicity and cost. The motion accuracy is the most important consideration of this problem. Most of the current, accurate systems are expensive, and therefore, are not applicable to any small application. In this work, a solution that is reliable, simple, and inexpensive enough to be used for every purpose is sought.

- Continuous sensing. The system should be able to track its own motion in a continuous manner. Some systems have a stop-sense-and-move mode, which is contrary to the application of this system.

- Real-time performance. The system should be able to provide the location and orientation of the robot at each moment in time. This is an important requirement since making the necessary decision based on abrupt and unexpected changes in natural environments is a critical issue in the safety of such autonomous operations.

- Vision-based only. System input is only provided through vision based sensors, that is, cameras. The use of wheel-based encoders, range finders or GPS sensors are not considered practical. This is mainly because the working environment includes severely uneven surfaces, outdoor elements such as fog, dust, snow and rain and foliage.
Chapter 3

Feature Detection

Many vision based tracking systems estimate camera motion by measuring relative changes in the projection of identical scene features in different image frames. Although, globally all points in the scene convey some information about the motion, locally not all the pixels of the image carry valuable motion information. For example, edges, occlusions or areas of uniform local intensity, can convey, at best, only partial information about the motion. An important aspect of a motion tracking system is its real time performance. Processing all the pixels of an image, from which only a small number carry information about the camera’s motion, may not be possible with the real-time requirement for such systems. Therefore, special attention is paid to selecting regions with higher information content. There have been many different criteria for choosing such regions, mostly based on areas with high second-order derivatives or high frequency content. In spite of the fact that these methods deliver traceable features, there are several characteristics that make some of them more reliable than others. This chapter first presents an overview of the previous work in feature detection. A novel feature detector using binary images is then introduced. Comparison of the efficiency and performance of the proposed method with existing ones is then presented.
3.1 Features

Marr [66] describes scene features as tokens that represent attributes of the image, which corresponds to physical events in the scene, e.g., a discontinuity in the surface reflectance or depth. In a simpler definition, features are defined by meaningful tokens of a scene that can be identified and tracked over a sequence of image frames. Several feature attributes, such as good localization, robustness with respect to noise and illumination changes, and efficiency of the detection algorithm can lead to a more accurate or faster estimation. Deciding on the type of feature is critical and depends greatly on the type of input sensors, environment, and the application that these features are used for. Common features that are generally used include the following [17]:

- Raw pixel values, i.e. the intensities. They retain the most information possible, but the extraction of that information is expensive. Moreover raw pixel values are sensitive to noise.

- Edges, surfaces and contours. These features usually correspond to real 3D structures in the scene, and they provide connectivity information [97]. For example, edges are less sensitive to noise and working with them is fast. However they are subject to inconsistencies, such as dropout from image to image, especially when extracted from real images. Also, they are often not viewable as complete entities, for example one or both end points may be obscured by other structures or be outside the field of view. Edges also suffer from the aperture problem. This means that for a moving edge, only the component of the motion perpendicular to the edge can be found accurately while the other component may belong to a set of possible motions.

- Salient features, such as corners, line intersections, points of locally maximum curvature on contour lines and texture patches. These types of features have the advantage of being discrete, reliable, meaningful and more accurate for finding the position.

- Statistical features, such as moment invariance, energy, entropy and color histograms.
3.2 Corner Detection Background

These features are measurements over a region of an image, representing the evaluation of the region. The main problem with this type of feature is that they are difficult to use in a matching process.

- Higher level features, such as structural and synthetic features. In this type of feature, relations and other higher level information is used. They are usually unique and fitted to specific applications or specific environments.

Choosing simple features within the scene increases the reliability of the solution for motion tracking and enables the system to find answers to problems most of the time, unless the scene is very uniform. In the search for a feature type that suits our application, a natural, unstructured environment with varying lighting conditions, it was decided to work with corners, because they are discrete and partially invariant to scale and rotational changes.

3.2 Corner Detection Background

Two broad groups of corner detectors are identified by Deriche and Giraudon [28].

- The first group extracts corners from edges represented as chain codes, by searching for points with maximum curvature or with some polygonal approximation, and searching for intersection points.

- The second group detects corners by computing a measure that indicates the presence of an interest point directly from the signal; for example, the grey-level values or brightness gradient.

Several detectors for identifying and localizing interest points have been developed and many of them are reviewed in [86]. Kitchen and Rosenfeld [52] find points in the image where the product of the gradient magnitudes and edge contour curvatures are maximum. Wang and Brady [107] define corners as points with maximum gradient on the direction perpendicular to the edge curvature. Moravec [67] presents a method based on computing
the intensity autocorrelation on four directions over a small window. Features with the local minimum auto-correlations are declared as corners. Harris and Stephens [40] improve Moravec’s method by computing auto-correlation for the squared, first order image derivatives. Smith and Brady [95] introduce a significantly different low level corner detector by measuring the similarity of the related parts of the image to each individual pixel. They associate each pixel with its local image regions with similar brightness. Distinctive features are detected by minimizing these image regions.

Schmid et al. [86] show that Harris and Stephens' method give the best results based on several criteria, such as repeatability with respect to scale, illumination, viewpoint change and noise. Robustness of a feature detector can strongly affect the performance of a system based on such features. Originally, for this work, the Harris corner detector was used. Since Harris method involves several smoothings, for the purpose of noise suppression, and computing first order image derivatives, it becomes computationally expensive. Also, by smoothing the intensity image and its derivatives, the higher spatial frequencies are attenuated, and therefore, the position of the corner appears with an offset, usually a fraction of a pixel, from its real position. These two effects limit performance of the system and to some extent restrict further attempts at quality improvement. One of the methods that promises a very good positional accuracy is the Smith and Brady [95] corner detector. As it will be explained later, this method is computationally very expensive and highly noise sensitive. The principles of these two methods as well as a novel method that is developed for this research are described in the following sections.

3.3 Harris Corner Detector

Harris and Stephens corner detector [40] was developed for enhancing the Moravec interest operator. The problem of detecting corners can be analyzed in terms of the curvature properties of local image brightness autocorrelation function, where curvature information can be represented by the Hessian matrix. The autocorrelation function is useful for characterizing
3.3 Harris Corner Detector

how brightness values change in the neighborhood of a location. At a corner or an isolated brightness peak, all shifts result in large changes in the autocorrelation function at that location. One version of the brightness spatial change function for a small shift \((x, y)\) can be expressed by

\[
E(x, y) = \sum_{u,v} W_{u,v} |I_{x+u,y+v} - I_{u,v}|^2
\]  

(3.1)

Here \(W_{u,v}\) represents a smooth circular window and is defined by

\[
W_{u,v} = e^{-\frac{u^2+v^2}{2}}
\]  

(3.2)

A first order approximation of \(E\) is given by

\[
E(x, y) = Ax^2 + 2Cxy + By^2
\]  

(3.3)

where \(A, B\) and \(C\) are approximations of the second order directional derivatives of the Gaussian smoothed brightness image.

\[
A = X^2 \otimes W \quad , \quad B = Y^2 \otimes W \quad , \quad C = XY \otimes W
\]  

(3.4)

\(E\) can be rewritten

\[
E(x, y) = [x, y] M \begin{bmatrix} x \\ y \end{bmatrix}
\]

where

\[
M = \begin{bmatrix} A & C \\ C & B \end{bmatrix}
\]  

(3.5)

where \(M\) is an approximation of the Hessian matrix of \(E\) and is closely related to the image's local autocorrelation function. The principal curvatures of the image brightness autocorrelation function at a point can be approximated by the eigenvalues of the approximated two by two Hessian matrix, \(M\), defined at that point. A corner is announced if principal curvatures are approximately equal and sufficiently large. The determinant of the approximated Hessian matrix is proportional to the product of the principal curvature. The Harris and Stephen's corner detector is given by the following operator where a large value of \(R\) corresponds with
the presence of corners [40].

\[ R(\alpha, \beta) = \text{Det}(M) - KT_r(M)^2 \]  

\[
\begin{cases}
T_r(M) = \alpha + \beta = A + B \\
\text{Det}(M) = \alpha\beta = AB - C^2
\end{cases}
\]

\[ X = I \otimes [-1, 0, 1] \approx \frac{\partial I}{\partial x} \\
Y = I \otimes \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} \approx \frac{\partial I}{\partial y} \]  

Here \( I \) is the image brightness. A positive value for scalar \( K \) can be used to ensure that the Gaussian curvature approximation \( \text{det}(M) \) is valid by suppressing the detector response for image regions with very high contrast, as signaled by a large mean curvature. Zhang et al. specify a value of 0.04 for \( K \) to discriminate against high contrast pixel step edges [114].

### 3.4 SUSAN Corner Detector

A low level corner detector recently introduced by Smith [95] may provide a solution to the position uncertainty and noise problems. The SUSAN corner detector is based on measuring the similarity of the related parts of the image to each individual pixel by means of non-linear filtering; each pixel is associated with local image regions with similar brightness. Distinctive features are detected by minimizing these regions. Since the method is not based on a derivative, it is less sensitive to existing white noise in the image; therefore, initial noise reduction is not necessary.

The principle of this corner detector is illustrated in Figures 3.1(a) and 3.1(b), which show a circular mask (with a central pixel, the nucleus) at 5 different positions of a dark rectangle on a white background. An area of the mask that has similar brightness to that of
3.4 SUSAN Corner Detector

Figure 3.1: The SUSAN corner detector.

the nucleus is called the USAN\(^1\) and is the basis for the SUSAN\(^2\) corner detector. The local area contains much information about the structure of the image. From its size, centroid and second moments, one and two dimensional features can be detected. Figure 3.1(b) shows the area of USAN, where each point in the input image is used as the nucleus of a small circular mask associated with the USAN at that point. It can be seen that this area decreases as an edge is approached, and it decreases even further near a corner, giving rise to a local minima at the exact position of the image corner. Figure 3.2 shows some examples where a corner or a non-corner situation is identified. Therefore, a circular mask, for an isotropic response, is placed at each point in the image. The brightness of each pixel within the mask is compared with that of the nucleus.

\[
c_1(\vec{r}, \vec{r}_0) = e^{-\left(\frac{I(\vec{r}) - I(\vec{r}_0)}{t}\right)^6}
\]  

(3.9)

Here, \(\vec{r}_0\) is the position of the nucleus in the two dimensional image, \(\vec{r}\) is the position

\(^1\)An acronym standing for Unvalue Segment Assimilating Nucleus.
\(^2\)An acronym standing for Smallest Unvalue Segment Assimilating Nucleus.
of any point within the mask, \( I(\vec{r}) \) is the brightness of any pixel. Parameter \( t \) defines the minimum contrast of detected features, as well as the maximum amount of ignored noise. A running total of \( n \) is then generated from the outputs of \( c_1 \).

\[
n(\vec{r}_0) = \sum_{\vec{r}} c_1(\vec{r}, \vec{r}_0)
\]

(3.10)

Here, \( n \) represents the area of SUSAN. For a point to be considered a corner, \( n \) must be less than half of its maximum possible value, \( g \), which determines corner quality. The dependence of the correct threshold on the data and the capability of setting it without human intervention are two factors having a large effect on the success of the algorithm.

Occasionally, SUSAN gives false positives with real data, where blurring of boundaries between regions occurs or where there is a thin line with a brightness approximately halfway between the two surrounding regions. This problem is eliminated by computing USAN’s center of gravity \( \vec{r} \).

\[
\vec{r}(\vec{r}_0) = \frac{\sum_{\vec{r}} c_1(\vec{r}, \vec{r}_0)}{\sum_{\vec{r}} c_1(\vec{r}, \vec{r}_0)}
\]

(3.11)

Clearly, an USAN corresponding to a proper corner has a center of gravity that is not near the nucleus, while false positives can be rejected by their short distance from the nucleus.

The real-time performance for our 3D motion tracking system requires a fast corner detector with a high positional accuracy. After initial inspections of both methods, it was concluded that both methods are computationally expensive. This can be explained by the fact that in Harris corner detector for each input image, three image derivatives are computed. These images are later smoothed by Gaussian filters. Computing these images as well as the corner response \( R_i \) (Equation 3.6) for every pixel of an image increases the execution time for this method. Computationally, SUSAN corner detector was more promising as it cuts down the number of corner candidates at each inspection phase. However, measurement
3.5 Binary Corner Detector

We developed a novel corner detector called Binary Corner Detector [82]. The basic idea behind the SUSAN corner detector is the main inspiration for this algorithm. The main emphasis of this method is on exploiting binary images and substituting arithmetic operations with logicals. Figure 3.3 represents the block diagram of the binary corner detector.

To generate a binary image that contains good low-level information content, first, the Laplacian is computed at each point of the intensity image. Horn [42] approximates the image Laplacian value by

$$\frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} \approx (I_{i-1,j} + I_{i,j-1} + I_{i+1,j} + I_{i,j+1} - 4I_{i,j})$$  \hspace{1cm} (3.12)

$I_{i,j}$ stands for the image intensity value at row $i$ and column $j$. Such an approximation for 2D Laplacian is separable and is implemented efficiently by logical operations. The binary image is then generated by the invariance of the sign of the Laplacian value at each point, (1) in Figure 3.3. Figure 3.4 shows a typical image and the corresponding binary image.

At this point a circular mask, $w$, is placed on each point of the binary image. The binary
3.5 Binary Corner Detector

(1) Create a binary image based on the sign of Laplacian

(2) Count the number of similar pixels to the centroid within a circular mask, $n(p_o)$

(3) Is $n(p_o)$ smaller than half the window? Yes

(4) $R(p_o) = n(p_o)$
Compute the center of gravity $G(p_o)$

(5) Is $G(p_o)$ close to the mask center? No

(6) Compute the intensity variation along the vector of center of gravity and centroid $|I(p_o) - I(p)|$

(7) Is it large enough? No

(8) Announce a corner

Figure 3.3: The block diagram of Binary Corner Detector.

Figure 3.4: An image and its binary sign of Laplacian image.
value of each point inside the mask is compared with that of the central point, $L(p_0)$. 

\[ C(p_0, p) = \begin{cases} 
1 & \text{if } L(p) = L(p_0), \\
0 & \text{if } L(p) \neq L(p_0). 
\end{cases} \]  (3.13)

$L(p)$ represents the binary image value at location $p(x, y)$. Now a total running sum $n$ is generated from the output of $C(p_0, p)$, (2) in Figure 3.3.

\[ n(p_0) = \sum_{w} C(p_0, p) \]  (3.14)

$n$ represents the area of the mask where the sign of Laplacian of the image is the same as that of the central point. For each pixel to be considered a potential corner, the value of $n$ must be smaller than at least half the size of the mask $w$ in pixels, (3) in Figure 3.3. This value is shown by $t$ in the corner response equation (3.15).

\[ R(p_0) = \begin{cases} 
n(p_0) & \text{if } n(p_0) < t, \\
0 & \text{otherwise.} 
\end{cases} \]  (3.15)

At this point, for each candidate with $R(p_0) > 0$, a center of connectivity $G(p_0)$ is computed, (4) in Figure 3.3. In order to be persistent with the notation in SUSAN corner detector [95] the center of connectivity is referred as the center of gravity.

\[ G(p_0) = \sqrt{g(x_0)^2 + g(y_0)^2} \]  (3.16)

where

\[ g(x_0) = \frac{\sum_{w}(x_0 - x)}{n(p_0)} \quad , \quad g(y_0) = \frac{\sum_{w}(y_0 - y)}{n(p_0)} \]  (3.17)

The center of gravity $G$ provides the corner direction, as well as a condition to eliminate
points with random distributions. Random distributed binary patches tend to have a center of gravity fairly close to the center of the patch. Therefore, all points with close centers of gravity are filtered out of the remaining process, (5) in Figure 3.3.

\[
G(p_0) > |r_g|
\]

The two conditions in (3.15) and (3.18) do not (by themselves) provide enough stability. For instance, Figure 3.5 shows an example where conditions 3.15 and 3.18 are both satisfied, however the geometrical distribution of pixels do not represent a corner.

![Figure 3.5](image.png)

Figure 3.5: Conditions in (3.15) and (3.18) do not (by themselves) provide enough stability.

Therefore, a third inspection is performed by computing the directional derivative along the corner direction for remaining candidates, (6) in Figure 3.3. Once again points with small directional intensity variations are eliminated, (7) in Figure 3.3. This condition is shown by

\[
|I(p_0) - I(p)| > I_t
\]

\(I_t\) represents the brightness variation threshold. Figure 3.6.a to 3.6.e illustrates the principles of the binary corner detector through different cases.

Figure 3.6.a shows a circular mask on an ideal corner. The response \(R(p_0)\) for this corner is smaller than half the mask size and the center of gravity \(g\) is located a fair distance from the center of the mask \(p_0\). Furthermore, the intensity variation of the mask center along \(g p_0\) is large. Figure 3.6.b and 3.6.c demonstrate cases where the mask centers are located either on an edge or in an area of uniformity. In such cases, \(n(p_0)\) fails in equation (3.15). Figure 3.6.d
represents an example where pixels on the mask have random distribution. Although the initial verification in equation (3.15) may be satisfied, the condition on a distant center of gravity in (3.18) fails. Finally, Figure 3.6.e shows an instance where the first two tests are fully fulfilled, but the intensity variation along vector \( \overrightarrow{gP_0} \) fails to provide a sufficiently large value, condition (3.19).

### 3.6 Evaluation Metric: Repeatability Rate

Schmid et al. [85] introduce an evaluation metric to verify the performance of several feature detectors. This measurement is defined as the percentage of the total detected corners that are repeated in the two images, and is called the repeatability rate. A scene point \( P \) with projection \( p_i \) in image \( I_i \) is repeated, if the corresponding point \( p_j \) is detected in image \( I_j \). In computing this rate, points that are seen in one image, but due to the existing transformation, \( H_{ij} \), between the two frames are not visible in the other one, are not counted. Also, since the corresponding point to \( p_i, p_j \), is usually not detected at the exact predicted position \( p_j \), but rather in the neighborhood of \( \epsilon \), the repeatability rate is defined as a function of the a neighborhood size by

\[
r_i(\epsilon) = \frac{|D(\epsilon)|}{n_i}
\]  

(3.20)
where

\[ D(\varepsilon) = \{(p_i, p_j) | \text{dist}(p_i, H_{ij}p_j) < \varepsilon\} \]  \hspace{1cm} (3.21)

\( n_i \) represents the number of the corners in image \( I_i \) that can potentially be observed in image \( I_j \).

### 3.7 Repeatability Results

In this section, the performance of the algorithm under different imaging conditions is presented. A reliable computation of repeatability in this part involves the careful registration of image frames. For this purpose, first the registration parameters between each image pair are roughly estimated by solving the registration equations for manually found correspondences. Second, through a recursive, least squares minimization, the registration parameters are refined until an error residual, smaller than half a pixel, is attained. Since the repeatability rate depends on the neighborhood \( \varepsilon \), in Equation 3.20, for each case, repeatability results are presented for two neighborhoods of 1.5 (left) and 0.5 (right) pixels.

#### 3.7.1 Camera noise

The camera noise effect is studied by processing images of a static scene, captured under identical conditions, but at different moments. The results are shown in Figure 3.7. In this experiment Harris delivers the highest robustness, followed closely by the binary method. The SUSAN method, however, does not retain a good rate, mainly due to the elimination of Gaussian smoothing. Even though binarization of SUSAN corner detector is an approximation and should lead to worse results the third condition (Equation 3.19) more than compensates and provides better repeatability rates.
3.7 Repeatability Results

3.7.2 Rotation

The behavior of the method under rotation is studied by processing a set of images that are acquired while the camera rotates near its optical axis. Due to the limitation of the rotation mechanism, the transformations between frames are not purely rotational around $y$, and include translation as well. Figure 3.8 shows the repeatability results for neighborhoods 1.5 (left) and 0.5 (right) pixels. This experiment covers rotations between $0^\circ$ to $180^\circ$ with steps of $10^\circ$. As the rotation angle increases the repeatability of the Harris and binary method becomes closer. Since in many real-time applications the rotation between the two consecutive frames is small, another experiment is performed to test the methods for smaller rotations covering a range of $0^\circ$ to $10^\circ$, as shown in Figure 3.10.

Figure 3.7: Repeatability results for camera noise for neighborhood of 1.5 (left) and 0.5 (right) pixels.

Figure 3.8: Repeatability rate under rotation for neighborhood of 1.5 (left) and 0.5 (right) pixels.
3.7 Repeatability Results

Figure 3.9: Images of a scene under different rotations.

Figure 3.10: Repeatability rate for small rotations a neighborhood of 1.5 (left) and 0.5 (right) pixels.

3.7.3 Scale

The scale change is accomplished by moving the camera in a perpendicular direction to the image plane. The scale change over 1.8 is not tested due to the high sensitivity and poor performance of all the methods.

Figure 3.11 represents the repeatability rate for the set of images shown in Figure 3.12. The results show that all methods perform better in a larger neighborhood, ($\epsilon = 1.5$). This can be explained by the fact that at different scales, identical features of the scene project onto areas with different resolutions. Therefore, although corners are considered invariant
3.7 Repeatability Results

Figure 3.11: Repeatability rate for different scales in neighborhood of 1.5 (left) and 0.5 (right) pixels.

to scale change, their positional precision and their repeatability are highly scale dependent.

Figure 3.12: Images of a scene using different scales.

3.7.4 Illumination

In this part, the sensitivity of the proposed method to image intensity variation is studied.

3.7.4.1 Uniform change

Uniform illumination change is simulated in this part, due to existing limitations on
3.7 Repeatability Results

changing our camera's aperture. For this purpose, first a set of images from a static scene with an identical point of view at different moments is acquired. Then, the average grey level (intensity) of each image is manually changed with steps of 10%, as shown Figure 3.14. Figure 3.13 displays the computed repeatabilities. These graphs reveal that as the relative grey level with respect to the reference image changes, the number of repeating features decreases with these methods.

![Figure 3.13: Images of a scene with linear illumination change.](image)

![Figure 3.14: Repeatability rate under uniform illumination change for neighborhood radiiuses of 1.5 (left) and 0.5 (right) pixels.](image)
3.7 Repeatability Results

3.7.4.2 Directional change in lighting

One of the important features of a good corner detector is its robustness in more complex lighting conditions. Therefore, the method’s stability is evaluated under directional illumination variations in which the light source illuminates the scene from different directions. Figure 3.15 represents some of the processed image taken under such conditions. The light source is moved with a step of 10° and covers a range of 0° to 90°. As the light source moves from a direction perpendicular to the camera plane (0°), the left image in Figure 3.15 to its side (at 90°), the right image in Figure 3.15, shadow effect becomes stronger. Such shading effects cause object boundaries to move toward the light source. This noticeably effects the repeatability by a large amount, especially in small neighborhoods. Figure 3.16 shows the results of this experiment.

![Figure 3.15: Images of a scene under changing illumination.](image)

![Figure 3.16: Repeatability rate for changing illumination for neighborhood of 1.5 (left) and 0.5 (right) pixels.](image)

It is important to note that in real situations there are interactions between different
imaging conditions. Here however the effect of each imaging condition was studied individually. Such interaction may affect presented results.

3.8 Discussion

As presented in the previous section, on average the repeatability of the binary method is about 20% less than in Harris' and 15% more than in SUSAN's. This might seem to be a problem at first, but as discussed here, for many vision based applications, including our trajectory tracking system, it may not affect reliability. There are two important aspects that make the binary corner detector still suitable:

- If the acquired images are processed faster, the transformation between the two consecutive frames becomes smaller. A smaller difference between the two frames can increase the repeatability rate. For instance, if binary corner detector performs 60% faster than Harris corner detector, for a motion of 10 degrees per second the repeatability rate of the binary corner detector becomes about 6% more than the reported results in Figure 3.8.

- One of the questions that was confronted in our previous work [80] is how many corners are needed for a reliable motion estimation. It is observed that motion parameters can be determined from a minimum of 30 to 40 corners. However, a number between 50 and 100 guarantees reliable motion estimation. For a typical outdoor image, Harris detects about 400 corners. A loss of 20% with the binary method results in 300 corners, still plenty to providing sufficient information for a reliable motion estimation.

3.9 Computational Time

The computational complexity of all the methods is studied by comparing their running time under the same conditions. In this study, images of 320×240 pixels and average the
running time for one thousand executions on a 1.14 GHz AMD Athlon(tm) Processor® are used. The results of this comparison are presented in Table 3.1. As shown in this table,

Table 3.1: Comparison of the time complexity of the three methods.

<table>
<thead>
<tr>
<th>Method</th>
<th>Execution time (msec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary</td>
<td>23.293</td>
</tr>
<tr>
<td>Harris</td>
<td>37.674</td>
</tr>
<tr>
<td>SUSAN</td>
<td>168.452</td>
</tr>
</tbody>
</table>

the SUSAN algorithm has the slowest performance. Details of the SUSAN corner detector suggest a simple computational complexity. This is because most of the computations and evaluations are performed on a considerably small selection of pixels than the total pixels of the image. However, the initial inspections for nominating such candidates include a considerable number of operations, causing the SUSAN algorithm to have the slowest performance. The Harris method performs significantly faster than the SUSAN, however 60% of its time is spent on the Gaussian smoothing of each image and its corresponding squared first order derivatives. This is in spite of the fact that all the 2D Gaussian filters are approximated by two 1Ds and a large number of the arithmetic operations are approximated by logical operations through a sigma of 0.8.

The binary corner detector performs 1.64 times faster than the Harris, and 7.23 times faster than the SUSAN. Moreover, about 70% of all the employed operations can be substituted by bitwise operations using specific hardwares such as FPGA (Field Programmable Gate Array) for even faster implementation.

3.10 Chapter Summary

In this chapter a novel corner detector for selecting informative pixels of an image is described. The proposed method is evaluated by computing the repeatability rate and comparing it to those of the Harris and SUSAN corner detectors. All the experiments are performed on
images of outdoor scenes, including natural geometric shapes. The faster performance of this method may allow for a better real-time performance of feature based vision systems. On average, the repeatability rate for our method is about 20% less than Harris's. As mentioned earlier, the estimated motion can be computed from a small subset of the image points, usually about 10% to 5% of the image size. A decrease of 20% in that number still provides more than enough points and delivers the same motion parameters. Also, as the system performs faster, the changes between consecutive frames lessen that potentially can cause a higher repeatability rate.
Chapter 4

3D World Reconstruction

Systems with no prior information about a scene usually need to determine the 3D positions of features in a scene. This is a difficult problem because each projected 2D image only records a point's relative direction from the camera, not its distance. Recovering distance or depth is known as the "inverse projection problem". This chapter describes the problem of optical projection as performed by our camera system, 3D world position reconstruction for feature points, and considerations for increasing the system accuracy.

4.1 Camera Model

A camera can be simply modeled using the classic pinhole model, its optical center $C$ and its image plane $\rho$, as in Figure 4.1. This leads to equations for calculating where on an image plane a point in space will appear. These equations are based on perspective projections. The projective transformations that project a world point $P(x, y, z)$ to its image point $p(x_u, y_u)$ are

$$x_u = f_x \frac{x}{z}$$

$$y_u = f_y \frac{y}{z}$$

(4.1)
4.1 Camera Model

Figure 4.1: An ideal pinhole camera produces a perspective projection model of the world.

Here, $f_x$ and $f_y$ represent the horizontal and vertical focal lengths of the camera. Since a camera exhibits non-ideal behavior, precise measurements from an image that are necessary in the 3D reconstruction process require a more sophisticated camera model than the ideal model. In other words, a simple perspective projection is not enough for locating world points in the image.

4.1.1 Camera calibration

Camera calibration is the process of finding internal and external camera system parameters that affect the imaging process. A camera model consists of extrinsic and intrinsic parameters. Extrinsic parameters are those that are not a function of the construction of the camera but its position in the environment. Intrinsic parameters, on the other hand, are physical parameters related to the construction of the camera and CCD itself. The camera model transforms real world coordinates into image coordinates and vice versa. Figure 4.2 shows the geometry of the camera calibration problem. There are four steps to transforming from a 3D world coordinate to a camera coordinate:

I. Transforming from the object world coordinates to the camera 3D coordinate system.

$(x_w, y_w, z_w)$ is the coordinate of the object point $P$ in a known 3D world coordinate.
4.1 Camera Model

system, and \((x, y, z)\) is the same point in a camera coordinate system. This rigid body transformation can be presented by rotation and translation operations:

\[
\begin{bmatrix}
x \\
y \\
z
\end{bmatrix} = \begin{bmatrix}
x_w \\
y_w \\
z_w
\end{bmatrix} + T
\]

where \(R\) is a \(3 \times 3\) and \(T\) is a \(3 \times 1\) matrices (camera's extrinsic parameters).

II. Transformation from 3D camera coordinate \((x, y, z)\) to ideal (undistorted) image coordinate \((x_u, y_u)\), perspective projection in Equation 4.1.

III. Transformation from the ideal image plane to the distorted image plane, radial lens distortion:

\[
\begin{align*}
x_d + D_x &= x_u \\
y_d + D_y &= y_u
\end{align*}
\]

\((x_d, y_d)\) represents the distorted image coordinates and \((D_x, D_y)\) represent their distor-
4.1 Camera Model

\[ D_x = x_d(k_1 r^2 + k_2 r^4 + \cdots) \]
\[ D_y = y_d(k_1 r^2 + k_2 r^4 + \cdots) \]  
(4.4)

\[ r = \sqrt{x_d^2 + y_d^2} \]

\( D_x \) and \( D_y \) can be approximated by their first terms:

\[ D_x \approx x_d(k_1 r^2) \]
\[ D_y \approx y_d(k_1 r^2) \]  
(4.5)

From these equations, the distortion can be seen to increase when moving from the center of the image to its edge.

IV. Transforming from distorted image plane coordinates to the frame buffer coordinates, \((x_f, y_f)\) row and column offset of a buffer in the computer memory.

\[ x_f = S_x (d_x N_{cx})^{-1} x_d + C_x \]
\[ y_f = (d_y)^{-1} y_d + C_y \]  
(4.6)

Where:

\((x_f, y_f)\) : row and column of the image pixel in computer frame memory.

\((C_x, C_y)\) : row and column of the center of computer frame memory.

\(d_x\) : Center to center distance between adjacent CCD sensor elements in \(x\) direction.

\(d_y\) : Center to center distance between adjacent CCD sensor elements in \(y\) direction.

\(N_{cx}\) : Number of sensor elements in the \(x\) direction.

\(N_{fx}\) : Number of pixels in a line as sampled by the computer.

\(S_x\) : The uncertainty image scale factor to be calibrated. This additional uncertainty factor is introduced due to a variety of factors such as slight hardware timing mismatch between image acquisition hardware and camera scanning hardware.
4.1 Camera Model

The intrinsic parameters to be found are focal lengths \((f_x, f_y)\), computer image centers \((C_x, C_y)\), lens distortion coefficient \((k_1)\), and the uncertainty factor \(S_x\). Details of the method for estimating intrinsic parameters can be found in [41,104]. Once cameras intrinsic parameters are found, the positional relationship between the undistorted and distorted images are established. The intrinsic parameters for our camera system are estimated by the manufacturer of the unit, Point Grey Research [45].

4.1.2 Image unwarping

Once the camera intrinsic parameters are found, they can be used to transform distorted image coordinates to their ideal and undistorted coordinates. This procedure is usually called the “unwarping” process and is performed using a lookup table that transfers each pixel on the distorted image onto its location on the corresponding undistorted location. Figure 4.3-a shows an image, acquired by our camera system, that has a 104° field of view. In this image the distortion effect is more noticeable on the curved bookshelves. Figure 4.3-b shows the same image after removal of the lens distortion. It can be clearly seen that the curved shelves on the original image are now straightened.

Figure 4.3: a- A warped image. b- The corresponding cut unwarped image.

As explained in more detail in the next section, reconstruction of the world through the
stereo process is based on the underlying assumption of an ideal pinhole camera model. Therefore, camera calibration and image unwarping are central issues for stereo-based vision systems. In fact, each time a set of images are acquired they must be corrected for the radial lens distortion, if they are to be used in the stereo vision system.

4.2 Stereo Vision Geometry

Stereo geometry is based on the fact that two spatially separated cameras (or eyes) view a scene from slightly different angles. The resulting shift in the projected images encodes the depth of objects in the scene. The stereo geometry is shown in Figure 4.4.

\[ \triangle OLP \sim PC_2 S \Rightarrow \frac{z-f}{k} = \frac{f}{d_1} \Rightarrow k = \frac{d_1(z-f)}{f} \quad (4.7) \]

\[ \triangle OLP \sim QC_1 T \Rightarrow \frac{z-f}{k + d_1 + B - d_2} = \frac{f}{d_2} \quad (4.8) \]

Here \( z \) is the distance of the object point \( O \) from the image planes, \( C_1 \) and \( C_2 \) are the camera centers, \( \rho \) is the image plane, \( f \) and \( B \) stand for the focal length and the displacement between the stereo cameras, and \( d_1 \) and \( d_2 \) represent points image displacement from the origin of the...
first and second image centers. Substituting \( k \) from 4.7 into 4.8 concludes the main stereo formulation:

\[
z = \frac{f \cdot B}{d_2 - d_1}
\]  \( (4.9) \)

Equation 4.9 shows that finding the disparity difference of \((d_2 - d_1)\), the positional change for one point in two images, is sufficient for constructing the point’s depth, \( z \). Therefore, with stereo processing one of the important factors that affects reliability of depth reconstruction is the baseline selection.

One of the assumptions that is made implicitly when describing the stereo geometry is that the two stereo images are coplanar. In reality, this assumption may not be valid and therefore, extra effort is required to make the two images coplanar. This assumption is called the “epipolar constraint” and the process involved in creating such a condition is known as the “rectification procedure”. Details of these processes are described next.

\subsection*{4.2.1 Epipolar constraint}

Given a known imaging geometry, the epipolar constraint in stereo images defines a line in one image along which a match can be found. Let’s consider a stereo vision system with two pinhole camera models with optical centers \( C_1 \) and \( C_2 \) (see Figure 4.5).

A World point \( P \) projects to \( P_r \) and \( P_l \), respectively. \( E_1 \) and \( E_2 \) are the epipoles through which all epipolar lines pass and \( DE_1 \) and \( DE_2 \) are the epipolar lines on which \( P_r \) and \( P_l \) are bound to lie. \( P_1 \), \( P_2 \) and \( P_3 \) are points in the scene that may project into the point \( P_r \) in the right image, and points \( P_{11} \), \( P_{12} \) and \( P_{13} \) are the projections of points \( P_1 \), \( P_2 \) and \( P_3 \) on the left image. The epipolar constraint forces the points \( P_{11} \), \( P_{12} \) and \( P_{13} \) to lie along a straight line. The implication of the epipolar constraint is that the search for a matching point in one image needs to be made only along one line in the other image.
4.2 Stereo Vision Geometry

4.2.1.1 Rectification procedure

As seen in the epipolar constraint, with several images and a point in one of them, the corresponding points in the other images are bound to lie on epipolar lines. These epipolar lines would be parallel if and only if all the image planes are parallel. In general, epipolar lines produce pencil lines going through an epipolar center. The epipolar center in each image is the image of the other camera center in this image plane. If the image planes are coplanar and lie parallel to the vector $C_1C_2$, defined by optical centers, then the epipolar centers are projected to infinity and the epipolar lines form a pencil of parallel lines.

Ayache and Hansen [4] show that it is always possible to apply a linear transformation to each image to obtain “conjugated” horizontal epipolar lines, e.g. the 2 lines $E_2D$ and $E_1D$. Rectification is one of the important processes in stereo vision and it allows potential matches between two or more images to satisfy simpler relations, further allowing for simpler and more efficient matching algorithms. Positional corrections for the rectification process are also performed by Point Grey Research [45]. Such corrections are added to the displacement parameters $D_x$ and $D_y$ in Equations 4.5. Therefore once unwarping lookup tables are created,
4.2 Stereo Vision Geometry

they correct for the rectification distortion as well.

As discussed, for finding a match, a search along the epipolar line is sufficient, and since the epipolar lines are along an axis after rectification, a search in one column or row is sufficient. A reliable 3D reconstruction of a physical point in space depends heavily on the correct identification of its match correspondences on the stereo images.

4.2.2 Stereo correspondence matching process

As discussed in Section 4.2 the essence of the stereo algorithm is in finding two corresponding points in stereo images such that the two points are the projections of an identical physical point. Equation (4.9) also indicated that the depth is proportional to the baseline length, $B$; accordingly the estimated distance is more accurate with two more distant cameras. Therefore, in stereo processing one of the important factors that affect the reliability of the depth reconstruction is the baseline selection. A short baseline means that the estimated distance is less precise due to a narrow triangulation, but it also means that the two images have more similarities and consequently a higher correspondence rate; whereas, a longer baseline increases the accuracy by creating a larger range of disparity. This nevertheless results in a more expensive search during the matching process, as well as an increase in the
number of false correspondences. Thus, there is a tradeoff between the accuracy and cost.

4.2.2.1 Multiple baseline

Okutomi and Kanade [48,74], introduce a stereo matching algorithm by using multiple stereo image pairs with different baselines. These multiple baselines are generated by displacement of one the cameras as illustrated in Figure 4.7. The image intensity functions \( f_0(x) \) and \( f_i(x) \)

\[
\begin{align*}
P_1 & \quad P_2 & \quad P_3 & \quad \ldots & \quad P_{n-1} \\
\bigcirc & \quad B_1 & \quad \bigcirc & \quad \ldots & \quad \bigcirc \\
\bigcirc & \quad B_2 & \quad \bigcirc & \quad \ldots & \quad \bigcirc \\
\bigcirc & \quad B_{n-1} & \quad \bigcirc & \quad \ldots & \quad \bigcirc
\end{align*}
\]

Figure 4.7: Camera positions for multiple-baseline stereo.

near the matching positions can be modeled as,

\[
f_0(x) = f(x) + n_0(x), \quad \text{and} \quad f_i(x) = f(x - d_{r(i)}) + n_i(x),
\]

(4.10)

where \( d_{r(i)} \) represents the real disparity for point \( x \) in image pair \( P_0 \) and \( P_i \). With the assumption of constant distance near \( Z \) and independent Gaussian white noise such that:

\[
n_0(x), n_i(x) \sim N(0, \sigma^2)
\]

(4.11)

the SSD function, \( e_{d(i)} \), over a window \( W \) and at pixel position \( x \) of the image \( f_0(x) \) for the candidate disparity \( d_{r(i)} \) is defined as

\[
e_{d(i)}(x, d_{r(i)}) = \sum_{j \in W} (f_0(x + j) - f_i(x + d_{r(i)} + j))^2
\]

(4.12)

By introducing:

\[
\zeta = \frac{1}{Z}, \quad d_{r(i)} = B_i F \zeta, \quad d(i) = B_i F \zeta
\]

(4.13)
the SSD with respect to inverse distance can be compute from:

\[ e_{\zeta(t)}(x, \zeta) = \sum_{j \in W} (f_0(x + j) - f_1(x + B_i F \zeta + j))^2 \] (4.14)

where \( \zeta_r \) and \( \zeta \) are the real and candidate inverse distance. The expected value of function \( e_{\zeta(t)}(x, \zeta) \) then is computed from:

\[ E[e_{\zeta(t)}(x, \zeta)] = \sum_{j \in W} (f(x + j) - f(x + B_i F(\zeta - \zeta_r) + j))^2 + 2N_w \sigma_n^2 \] (4.15)

where \( N_w \) is the number of the points within the window \( W \). An integration of each \( e_{\zeta(t)} \), creates an evaluation function \( e_{\zeta(12...n)}(x, \zeta) \), called SSSD, with an expected value as follows:

\[ e_{\zeta(12...n)}(x, \zeta) = \sum_{i=1}^{n} e_{\zeta(t)}(x, \zeta) \] (4.16)

\[ E[e_{\zeta(12...n)}(x, \zeta)] = \sum_{i=1}^{n} \sum_{j \in W} (f(x + j) - f(x + B_i F(\zeta - \zeta_r) + j))^2 + 2nN_w \sigma_n^2 \] (4.17)

This new function has two interesting characteristics that make it beneficial in depth construction and motivated its use in this work.

1. Elimination of the ambiguity

If intensity pattern \( f(x) \) has the same pattern around \( x \) and \( x + a \), then:

\[ E[e_{\zeta(t)}(x, \zeta_r)] = E[e_{\zeta(t)}(x, \zeta + \frac{a}{B_i F})] = 2nN_w \sigma_n^2 \] (4.18)

Equation (4.18) shows that the expected SSD value is minimum for the correct, \( \zeta_r \), as well as for false matches, \( \zeta_f = \zeta_r + \frac{a}{B_i F} \). However, an important point to be observed is that the position of this false minimum is dependent upon the baseline \( B_i \); whereas the minimum for the correct match is not. Therefore, having more stereo camera pairs and combining their information, can clarify the inherent ambiguity existing in finding correspondence in the periodic patterns.
2. Precision

This method not only resolves the ambiguity for periodic and non-periodic patterns, but it also improves the precision of the estimated distance. If \( \sigma_n^2 \) represents the variance of the white Gaussian image noise then the variance of the an estimate \( \hat{\zeta}_{r(i)} \) is computed by:

\[
Var(\hat{\zeta}_{r(i)}) = \frac{2\sigma_n^2}{B_i^2 F^2 \sum_{j \in W} (f'(x + j))^2}
\]  
(4.19)

The variance of the estimated inverse distance \( \hat{\zeta}_{r(12...n)} \) that minimizes the SSSD in inverse-distance is then defined by:

\[
Var(\hat{\zeta}_{r(12...n)}) = \frac{2\sigma_n^2}{\left(\sum_{i=1}^{n} B_i^2 F^2 \sum_{j \in W} (f'(x + j))^2\right)} = \left(\sum_{i=1}^{n} Var(\hat{\zeta}_{r(i)})\right)^{-1}
\]  
(4.20)

Equation (4.20) demonstrates that using the SSSD in inverse-distance with multiple baseline pairs, improves the accuracy of the estimated depth.

4.2.2.2 Matching through correlation with validation

In this method, by Fua [32], the similarity scores for every feature point in the image is computed by measuring the NSSD (Normalized Sum of Squared Differences) for all the match candidates along the epipolar line. The best match is then selected based on the similarity metric value. In this approach, the probability of a mismatch decreases as the size of the correlation window and the texture density of the area inside the window increase. Using a large window is not a practical solution as it slows down the system. In order to improve the reliability, a validation is performed in which two images play similar roles.

Figure 4.8 represents a graphical presentation of the validity check for the stereo matching process. Given a point \( P_1 \) in \( I_1 \), let \( P_2 \) be the point of \( I_2 \) located on the epipolar line corresponding to \( P_1 \) such that the windows centered on \( P_1 \) and \( P_2 \) yield to an optimal
correlation measure. The match is valid if and only if $P_1$ is also the point that maximizes the scores when correlating the window centered on $P_2$ with windows that shift along the epipolar line of $I_1$ corresponding to $P_2$. This method was also used in this work in matching features that belong to areas that were common between either the reference and horizontal, or the reference and vertical images.

4.2.3 Stereo correspondence matching rules

Our camera system, Digiclops [45], includes 3 stereo cameras that are vertically and horizontally aligned. The displacement between the reference camera and the horizontal and the vertical cameras is 10 centimeters. Due to the orthogonal arrangement of the stereo cameras and their identical baseline values, an improvement of depth accuracy by incorporating longer baselines is not really possible. However, to fully take advantage of the existing features in the three stereo images, the following constraints are employed in the stereo matching process:

- **Feature stability constraint I**: For each feature in the reference image that is located in the common regions amongst the three stereo images, there should be two correspondences, otherwise the feature gets rejected due to the instability condition. 3D locations of the features that pass this constraint are estimated by the multiple baseline method [74].
4.2 Stereo Vision Geometry

- **Feature stability constraint II**: Features located on the areas common to only the reference and horizontal or to the reference and vertical images are reconstructed if they pass the validity check by Fua's method [32].

- **Disparity constraint**: The disparities of each feature from the vertical and horizontal images to the reference image have to be positive, similar (with maximum difference of 1 pixel), and smaller than 90 pixels. This constraint allows the construction of the points as close as 12.5 cm from the camera for our system configuration.

- **Epipolar constraint**: The vertical disparity between the matched features in the horizontal and reference images must be within 1 pixel. The same rule applies to the horizontal disparity for matches between the vertical and reference match correspondences.

- **Match uniqueness constraint**: If a feature has more than one match candidate that satisfies all the above conditions, it is considered ambiguous and gets omitted from the rest of the process.

The similarities between each feature and its corresponding candidates are measured by employing the Normalized Sum of Squared Differences metric [32]. After matching the features, a subset of features from the reference image is retained, and for each one, its 3D location with respect to the current camera coordinate system is obtained using equation 4.9.

### 4.2.4 Depth construction inaccuracy sources

While the previous constraints decrease the chance of false matches to construct a more reliable depth value, the accuracy of the 3D reconstruction is still limited to the accuracy of the unwarped images.

Although the unwarping process is a necessary step in the stereo process, it is noticeable that for a camera with a wide field of view, the unwarped image includes some blurriness
that increases when moving from the center toward the sides of the image. This effect is more noticeable and disturbing when the scene is constructed of objects with finer details, Figure 4.9.

![Figure 4.9: Radial lens distortion removal can change the shape of objects of smaller size. a) A warped (raw) image. b) The corresponding unwarped image.](image)

Lens distortion elimination through the unwarping process is the procedure of projecting distorted image pixels onto their undistorted positions. As explained in Section 4.1.2, the unwarping process is performed by using intrinsic parameters of the camera. During this process the following occurs, as shown in Figure 4.10 1 to 4.10 4:

I. The image coordinates of each pixel, integer values, are transformed into the corresponding undistorted image coordinates, floating point, Figure 4.10 2.

II. Since not all the pixels in the unwarped image can be defined from the warped image an interpolation scheme is used to reconstruct the image values at an integer, equally spaced, mesh grid, Figure 4.10 2 and 4.10 3.

III. The resultant image is cut to the size of the original raw image, Figure 4.10 4.

For our camera system, 28.8% of the unwarped image pixels have no correspondences in the warped image, and therefore, are merely guessed at by the interpolation of the neighboring pixels. This creates considerable distortion of the shape of smaller objects located near
4.2 Stereo Vision Geometry

Figure 4.10: The raw image (1). The raw image after the calibration (2). The calibrated image after the interpolation (3). The final cut unwarped image (4).

the sides. It also increases the inaccuracy of the 3D world reconstruction, which influences the overall accuracy of the motion tracking system.

The effect of this distortion on the estimated 3D motion is examined by measuring a similar motion using identical features, located at different regions of the image plane. Figures 4.11 a-2, b-2, and c-2 demonstrate the raw images of a planar subject with small details at three different locations of a scene. Figures 4.11 a-1, b-1, and c-1 show corresponding unwarped images. The subject plane is placed at a distance of 20cm from the camera plane. For each case, a motion of 8cm in the direction perpendicular to the image plane is created. The 3D motion is then estimated using two approaches:
• In the first approach, feature points are found in the warped raw images first. Second, using calibration equations in Section 4.1.1 the exact location of the feature points in the unwarped images are estimated. These values are used in finding disparity values and hence in estimating the 3D locations of the feature points.

• In the second approach, feature points are found in unwarped images and are projected into the world using the stereo algorithm that uses their interpolated, unwarped positions.

The motion in each case is then estimated using the displacement between these world points in two consecutive frames. Table 4.1 represents the results of this experiment.

As illustrated in Table 4.1, using warped images instead of unwarped ones may improve trajectory estimation results. Elimination of the unwarping process can potentially reduce the execution time of the system. When using unwarped images, at each frame capture three new images are acquired that must be unwarped.

Another undesirable effect of the unwarping process is the elimination of some raw image
Table 4.1: 8.0 cm motion estimation using unwarped and warped images.

<table>
<thead>
<tr>
<th>Image Type</th>
<th>a</th>
<th>b</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td>Warped (-1)</td>
<td>7.98</td>
<td>7.97</td>
<td>7.99</td>
</tr>
<tr>
<td>Unwarped (-2)</td>
<td>7.18</td>
<td>8.01</td>
<td>7.98</td>
</tr>
</tbody>
</table>

regions after this process. Generally, the behavior of image pixels located at relatively far distances from the camera with image projections close to the center, under small rotations are very similar to lateral translations. As these points move further from the image center, their rotational behaviors become more distinguishable from their translational behaviors. Therefore, by cutting the edges of the unwarped image, a number of features that may carry crucial information about the nature of the motion get eliminated. This may be responsible for another source of inaccuracy in the estimated motion. Figure 4.12 illustrates an example

![Figure 4.12: Some important features are eliminated by the unwarping process.](image)

where a number of far away features are eliminated after the size adjustment.
4.2.5 Depth construction with higher accuracy

For all of the above reasons, it is chosen to employ a combination of raw and unwarped images. This means that features are detected only on raw images with sharper detail. The image coordinate of each feature is then unwarped using unwarping lookup tables. For constructing the 3D positions of the features, the unwarped locations are used. However, when measuring the similarity of features, raw locations with warped image content are used. Performing a partial unwarping procedure for a small percentage of each image also improves the running time of the system.

The 3D reconstruction of the feature points can be summarized as having the following steps:

1. A projection lookup table implementation using intrinsic camera parameters for raw image projection onto the unwarped image and vice versa.
2. Detection of features in raw images.
3. Disparity measurement in raw images using the projection lookup table.
4. 3D reconstruction of image features using the constraints in 4.2.2.

Another gained advantage on employing the above steps in 3D reconstruction process is the elimination of the sub-pixel estimator when computing the disparity values; This is due to the fact that the exact position of each corner is now found (in the undistorted images) with an accuracy of 0.01 pixel.

4.3 Chapter Summary

In this chapter, the camera calibration process as well as the stereo algorithm for the inverse perspective projection problem are explained. It is explained that camera calibration is an essential process in the 3D reconstruction problem. However, the existing approximation
performed in the reconstruction of calibrated images can degrade the accuracy of the overall system. To remove this undesirable effect, the reconstruction of the calibrated images is eliminated. Instead, a calibration map for each one of the cameras is constructed that defines the relationship between the integer position of uncalibrated pixels with corresponding floating point locations on the calibrated image. Through this process, three main advantages are gained. First, the elimination of the resampling process improves the running time, as previously, for each time three images needed to be unwarped. Second, a sharper image is processed that improves the number of detected features. For wide angle lenses, such as the one we use, the blurring effect on the sides is so disruptive that it degrades the feature detection process in those areas. Third, this process increases the accuracy in found disparities. Therefore, a more reliable world reconstruction is achieved, resulting in more accurate motion estimation.
Chapter 5

Tracking and Motion Estimation

The measurement of local displacement between the 2D projection of similar features on consecutive image frames is the basis for measuring the 3D camera motion in the world coordinate system. Therefore, world and image features must be tracked from one frame (at time=$t$) to the next frame (at time=$t + \Delta t$).

In order to take advantage of all the information acquired by the camera while navigating in the environment, a database is created. This database includes information about all the features seen since the first frame. For each feature, the 3D location in the reference coordinate system and the number of times it has been observed are recorded. Each database entry also holds a $3 \times 3$ covariance matrix that represents the uncertainty associated with the 3D location of each feature. The initial camera frame is used as the reference coordinate system, and all the features are represented in relation to this frame. After the world features are reconstructed using the first frame, the reference world features, as well as the robot's starting position, are initialized. By processing the next frame, in which a new set of world features are created, corresponding to the current robot position, new entries are created in the database. This database is updated as the robot navigates in the environment.

Since corner features are simply points there should be a matching process to find corresponding features in subsequent frames. The accuracy of the match correspondence finding
5.1 Similarity Measurement

In the feature matching process, a feature is selected in one image frame and a similar feature is sought in the following image frames based upon different similarity measurement criteria. There are many different similarity measurement techniques in the literature. Suitability of each one of these methods depends highly on the application that they are used for [17, 51, 65].

To measure the similarity of a feature to a set of correspondence candidates in this work, a normalized cross-correlation metric [32] is employed. Each feature and its candidates are first projected onto their corresponding image planes, $I_1(x, y)$ and $I_2(x, y)$. The cross-correlation function, Equation 5.1, is then estimated for each pair.

$$C(I_1, I_2) = \frac{\sum_{x=-\frac{M}{2}}^{\frac{M}{2}} \sum_{y=-\frac{M}{2}}^{\frac{M}{2}} ((I_1(x, y) - \bar{I}_1) - (I_2(x, y) - \bar{I}_2))^2}{\sqrt{\sum_{x=-\frac{M}{2}}^{\frac{M}{2}} \sum_{y=-\frac{M}{2}}^{\frac{M}{2}} (I_1(x, y) - \bar{I}_1)^2 \sum_{x=-\frac{M}{2}}^{\frac{M}{2}} \sum_{y=-\frac{M}{2}}^{\frac{M}{2}} (I_2(x, y) - \bar{I}_2)^2}}$$

(5.1)

Here, $\bar{I}_1$ and $\bar{I}_2$ are average gray levels over image patches of $I_1$ and $I_2$ with dimensions of $M \times M$. After evaluation of the similarity metric for all pairs, the best match with the highest similarity is selected.

The highest similarity as estimated by the cross-correlation measurement does not, by itself, provide enough assurance for a true match. Since the patch sizes are fairly small, there may be cases where a feature (at time=$t$) and its match correspondence (at time=$t+\Delta t$) do not correspond to an identical feature in the space. In order to eliminate such falsely matched pairs, a validity check is performed. In this check, after finding the best match for a feature, the roles of the match and the feature are exchanged. Once again, all the candidates for the match are found on the previous frame (at time=$t$). The similarity metric is evaluated for
all candidate pairs and the most similar pair is chosen. If this pair is exactly the same as the one found before, then the pair is announced as a true match correspondence. Otherwise, the corner under inspection gets eliminated from the rest of the process. Figure 5.1 represents a graphical overview of the match correspondence validity check.

Figure 5.1: Graphical overview of the validity check in feature tracking process. The valid match candidates are shown with the solid line.

Comparison of the validity check of the number of correct match correspondences for two consecutive outdoor images is shown in Figure 5.2. Figure 5.2 a and b, show the match correspondence without the validity check. Figure 5.2 c, and d display the results of the matching process for the same images in the presence of a validity check. Clearly, the number of false matches are reduced after the validity check.

5.2 Feature Tracking

The objective of the feature matching process is to find and to match the correspondences of a feature in the 3D world on two consecutive image planes (the current and the previous frames) of the reference camera . At all times, a copy of the previous image frame is maintained. Therefore, database feature points in the reference coordinate system are transformed to the
Figure 5.2: Validity checking reduces the number of false match correspondences. In this figure feature points are shown in white dots. Matched features in the first frame with no validity check are shown in circles (a). Match features in the second frame are shown with arrows that connect their previous positions into their current positions (b). Matched features in the first frame with validity check are shown in circles (c). Match features with validity check in the second frame are shown with arrows that connect their previous positions into their current positions (d).

last found robot (camera) position. They are then projected, if their projection fall inside the image plane, onto the previous image plane. With two sets of feature points, one in the previous image frame and one in the current image frame, the goal becomes to establish a one by one correspondence between the members of both sets. The matching and tracking
5.2 Feature Tracking

process is performed using a two-stage scheme.

I. Position of each feature in the previous frame is used to create a search boundary for corresponding match candidates in the current frame. For this purpose it is assumed that the motion of features from previous frame to the current frame do not have image projection displacements more than 70 \( (w) \) pixels in all four directions. This means that if a feature is located in row \( n \) and column \( m \) of the previous image frame, its match candidates are sought in a window that expands horizontally from \( n-w \) to \( n+w \) and vertically from \( m-w \) to \( m+w \) on the current frame. The value of \( w \) is highly dependent on the lens type, the average distance of the camera from the scene, and the speed of the mobile robot. For the current system, a search window of 141x141 seemed to be appropriate. Therefore, all the feature points in the current image frame that fall inside these boundaries are chosen as match candidates. If a feature does not have any correspondences, it cannot be used at this stage and therefore is ignored until the end of first stage of the tracking.

The normalized cross-correlation with validity check, as explained in Section 5.1, is then evaluated over windows of 13x13 pixels to select the most similar candidate. Using the established match correspondences between the two frames, the motion of the camera is estimated. Due to the large search window, and therefore, a large number of match candidates, some of these match correspondences may be chosen falsely. This influences the accuracy of the motion process. In order to eliminate inaccuracy due to faulty matches, the estimated motion is used as an initial guess for the amount and direction of the motion to facilitate a more precise motion estimation in the next stage.

II. Then, using the found motion vector and the previous robot location, all the database features are transformed into the current camera coordinate system. Regardless of the motion type or the distance of the features from the coordinate center, features with a persistent 3D location end up on a very close neighborhood to their real matches on the current image plane. Using a small search window of 4x4, the normalized
5.3 Motion Estimation

Given a set of corresponding features between a pair of consecutive images, motion estimation becomes the problem of optimizing a 3D transformation that projects the world corners, from the previous image coordinate system, onto the next image. Although the 3D construction of 2D features is a nonlinear transformation, the problem of motion estimation is still well behaved. This reasonable behavior is because any 3D motion includes rotations and translations [63].

- Rotations are functions of the sine and cosine of the rotation angles. With the assumption of small rotational changes between each two frames, sine and cosine can be linearly related to the angle.

- Translation toward or away from the camera introduces a perspective distortion as a function of the inverse of the distance from the camera.

- Translation parallel to the image plane is almost linear.
Therefore, the problem of 3D motion estimation is a promising candidate for the application of Newton's minimization method, which is based on the assumption of local linearity.

### 5.3.1 Least-squares minimization

The objective of the Newton method is to find the camera motion that brings each projected 3D feature into best alignment with its observed matching feature. Rather than solving this directly for the camera motion with 6 DoF, Newton's method iteratively estimates a vector of corrections $x$, that if subtracted from the current estimate, resulting in the new
5.3 Motion Estimation

If $P^{(i)}$ is the vector of parameters for iteration $i$, then

$$P^{(i+1)} = P^{(i)} - x$$  \hspace{1cm} (5.2)

Given a vector of error measurements between the projection of 3D world features on two consecutive image frames, $e$, a vector $x$ is computed that eliminates (minimizes) this error [63].

$$Jx = e$$  \hspace{1cm} (5.3)

The effect of each correction vector element, $x_j$, on error measurement $e_i$ is shown in Equation (5.4), where

$$J_{ij} = \frac{\partial e_i}{\partial x_j} \begin{cases} i = 1\ldots6, \\ j = 1\ldots2n \end{cases}$$  \hspace{1cm} (5.4)

Here $e_i$ is the error vector between the predicted location of the object and the actual position of the match found in image coordinates. $n$ represents the number of matched features. Since Equation (5.3) is usually over-determined, $x$ is estimated to minimize the error residual [34].

$$\min \|Jx - e\|^2$$

$$x = [J^T J]^{-1} J^T e$$  \hspace{1cm} (5.6)

Therefore, in each iteration of Newton's method, Equation (5.6) is solved for $x$ using $LU$ decomposition [77].

The most computationally expensive aspect of implementing Newton's method is calculating the partial derivatives. The partial derivatives with respect to the translation parameters can be most effectively calculated by first reparameterizing the projection equations to express translations in terms of each camera coordinate system [62].
5.3 Motion Estimation

5.3.2 Setting up equations

If the motion parameter vector is \((D_x, D_y, D_z, \phi_x, \phi_y, \phi_z)\), then the new location of a rotated point \((x, y, z)\) in the image is this:

\[
(u, v) = \left( \frac{f(x + D_x)}{z + D_z}, \frac{f(y + D_y)}{z + D_z} \right)
\]  

(5.7)

\(D_x, D_y\) and \(D_z\) represent the incremental translations, while \(\phi_x, \phi_y\) and \(\phi_z\) are rotational increments around the \(x, y\) and \(z\) axes. The partial derivatives in rows \(2n\) and \(2n + 1\) of the Jacobian matrix, Equation 5.3, that corresponds to the \(n\)'th matched feature, can be calculated by:

\[
J_{(2n, 1:6)} = \begin{cases}
\frac{\partial u}{\partial D_x} = \frac{f}{z + D_z} , \quad \frac{\partial u}{\partial \phi_x} = \frac{f}{z + D_z} \frac{\partial x}{\partial \phi_x} - \frac{f(x + D_z)}{(z + D_z)^2} \frac{\partial z}{\partial \phi_x} \\
\frac{\partial u}{\partial D_y} = 0 , \quad \frac{\partial u}{\partial \phi_y} = \frac{f}{z + D_z} \frac{\partial x}{\partial \phi_y} - \frac{f(x + D_z)}{(z + D_z)^2} \frac{\partial z}{\partial \phi_y} \\
\frac{\partial u}{\partial D_z} = \frac{f(x + D_z)}{(z + D_z)^2} , \quad \frac{\partial u}{\partial \phi_z} = \frac{f}{z + D_z} \frac{\partial x}{\partial \phi_z} - \frac{f(x + D_z)}{(z + D_z)^2} \frac{\partial z}{\partial \phi_z}
\end{cases}
\]  

(5.8)

\[
J_{(2n+1, 1:6)} = \begin{cases}
\frac{\partial v}{\partial D_x} = \frac{f}{z + D_z} , \quad \frac{\partial v}{\partial \phi_x} = \frac{f}{z + D_z} \frac{\partial y}{\partial \phi_x} - \frac{f(y + D_y)}{(z + D_z)^2} \frac{\partial z}{\partial \phi_x} \\
\frac{\partial v}{\partial D_y} = 0 , \quad \frac{\partial v}{\partial \phi_y} = \frac{f}{z + D_z} \frac{\partial y}{\partial \phi_y} - \frac{f(y + D_y)}{(z + D_z)^2} \frac{\partial z}{\partial \phi_y} \\
\frac{\partial v}{\partial D_z} = \frac{f(y + D_y)}{(z + D_z)^2} , \quad \frac{\partial v}{\partial \phi_z} = \frac{f}{z + D_z} \frac{\partial y}{\partial \phi_z} - \frac{f(y + D_y)}{(z + D_z)^2} \frac{\partial z}{\partial \phi_z}
\end{cases}
\]  

(5.9)

The partial derivative of \(x, y\) and \(z\) with respect to counterclockwise rotation parameters \(\phi\) (in radians) can be found in Table 5.1. This table shows that computing the Jacobian elements in Equations 5.8 and 5.9 is a fast task and quite promising for a real-time system performance. Jacobian components with respect to rotation in Equations 5.8 and 5.9 are
Table 5.1: The partial derivatives table.

<table>
<thead>
<tr>
<th>( \partial/\partial )</th>
<th>( x )</th>
<th>( y )</th>
<th>( z )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \phi_x )</td>
<td>0</td>
<td>(-z)</td>
<td>( y )</td>
</tr>
<tr>
<td>( \phi_y )</td>
<td>( z )</td>
<td>0</td>
<td>(-x)</td>
</tr>
<tr>
<td>( \phi_z )</td>
<td>(-y)</td>
<td>( x )</td>
<td>0</td>
</tr>
</tbody>
</table>

then simplified into:

\[
\begin{align*}
\frac{\partial u}{\partial \phi_x} &= -fy(x + D_z) \frac{1}{(z + D_z)^2}, & \frac{\partial v}{\partial \phi_x} &= -\frac{fz}{z + D_z} - \frac{fy(y + D_y)}{(z + D_z)^2} \\
\frac{\partial u}{\partial \phi_y} &= \frac{fz}{z + D_z} + \frac{fx(x + D_z)}{(z + D_z)^2}, & \frac{\partial v}{\partial \phi_y} &= \frac{fx(y + D_y)}{(z + D_z)^2} \\
\frac{\partial u}{\partial \phi_z} &= -fy \frac{1}{z + D_z}, & \frac{\partial v}{\partial \phi_z} &= \frac{fx}{z + D_z}
\end{align*}
\]

(5.10)

After setting up Equation 5.6, it is solved iteratively until a stable solution is obtained.

5.3.3 Implementation consideration

Converging to a true solution requires a high quality feature matching process. Although, through several tests, it is attempted to prevent outliers from reaching this stage, chances are that some outliers exist among our matched features. In order to minimize the effect of faulty matches on the final estimated motion, the following considerations are taken into account during implementation:

- The estimated motion is allowed to converge to a more stable state by running the first three consecutive iterations.

- At the end of each iteration the residual error for each matched pair in both coordinate directions, \( E_u \) and \( E_v \), are computed.

- From the fourth iteration, the motion is refined by elimination of outliers. For a feature
to be considered an outlier, it must have a large residual error, \( \sqrt{E_u^2 + E_v^2} \). Removing all outliers, where the estimation is not finalized yet, may affect the correct solution dramatically. For this reason, at each time only 10% of the features with the highest residual error will be discarded as outliers.

- The minimization is repeated for up to 10 iterations if changes in the variance of the error residual vector is more than 10%. During this process the estimation moves gradually toward the best solution.

- The minimization process stops if the number of inliers drops to 40 or less matches. This is due the fact the least squares minimization results, for six unknowns, when the number of match correspondences between two frames are less than 40 may not be very stable and accurate. Moreover, a minimum number of 40 match correspondences prevents matrix \( J^TJ \) from being ill conditioned.

### 5.3.4 Motion estimation results

The results of an entire motion estimation cycle, for a distance of about 5cm in the outdoor environment, is presented in Table 5.2.

As shown in this table, the error is reduced in a consistent manner and the final error residual is less than a pixel. Generally the error residual is only a fraction of a pixel.

### 5.3.5 Feature flow results

As it has been explained in Section 5.3.1, when solving the linear equation system with six unknowns, at least three valid corresponding points should be found in two sets of images. However, for accurate and reliable results, it is better to have an over-constrained problem. Moreover, in match validation required for the stereo process, and for tracking identical features in different frames, some of the extracted features might be discarded. Therefore, having more features results in more reliable results. Table 5.3 presents the number of
5.3 Motion Estimation

Table 5.2: Iteration results along with the error residual, in pixels, for one motion estimation.

<table>
<thead>
<tr>
<th>Iteration</th>
<th>No of matches</th>
<th>( D_x ) Cm</th>
<th>( D_y ) Cm</th>
<th>( D_z ) Cm</th>
<th>( \phi_x ) Degree</th>
<th>( \phi_y ) Degree</th>
<th>( \phi_z ) Degree</th>
<th>( \sqrt{E_u^2 + E_v^2} ) Pixel</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>188</td>
<td>-6.063</td>
<td>-0.059</td>
<td>-0.233</td>
<td>-0.047</td>
<td>0.127</td>
<td>-1.041</td>
<td>19.7947</td>
</tr>
<tr>
<td>2</td>
<td>188</td>
<td>-5.976</td>
<td>0.012</td>
<td>-0.207</td>
<td>0.118</td>
<td>0.099</td>
<td>-1.014</td>
<td>10.5260</td>
</tr>
<tr>
<td>3</td>
<td>188</td>
<td>-5.975</td>
<td>0.012</td>
<td>-0.207</td>
<td>0.117</td>
<td>0.096</td>
<td>-1.015</td>
<td>10.48685</td>
</tr>
<tr>
<td>4</td>
<td>170</td>
<td>-6.413</td>
<td>0.063</td>
<td>-0.493</td>
<td>-0.030</td>
<td>0.570</td>
<td>-0.345</td>
<td>4.4073</td>
</tr>
<tr>
<td>5</td>
<td>153</td>
<td>-6.401</td>
<td>0.096</td>
<td>-0.414</td>
<td>0.018</td>
<td>0.530</td>
<td>-0.207</td>
<td>2.4057</td>
</tr>
<tr>
<td>6</td>
<td>138</td>
<td>-6.469</td>
<td>0.112</td>
<td>-0.454</td>
<td>-0.030</td>
<td>0.541</td>
<td>-0.121</td>
<td>1.5502</td>
</tr>
<tr>
<td>7</td>
<td>125</td>
<td>-6.530</td>
<td>0.141</td>
<td>-0.424</td>
<td>0.049</td>
<td>0.585</td>
<td>-0.052</td>
<td>1.1942</td>
</tr>
<tr>
<td>8</td>
<td>113</td>
<td>-6.681</td>
<td>0.101</td>
<td>-0.427</td>
<td>-0.025</td>
<td>0.614</td>
<td>-0.061</td>
<td>1.2864</td>
</tr>
<tr>
<td>9</td>
<td>102</td>
<td>-6.521</td>
<td>0.145</td>
<td>-0.468</td>
<td>-0.014</td>
<td>0.588</td>
<td>-0.075</td>
<td>0.7966</td>
</tr>
<tr>
<td>10</td>
<td>92</td>
<td>-6.514</td>
<td>0.137</td>
<td>-0.458</td>
<td>-0.013</td>
<td>0.563</td>
<td>-0.078</td>
<td>0.6333</td>
</tr>
</tbody>
</table>

Features at different stages of the system for two individual experiments.

Table 5.3: Feature numbers and comparison of the matching accuracy of the tracking process at different stages of the system for two individual experiments.

<table>
<thead>
<tr>
<th>No</th>
<th>Variable definition</th>
<th>Static camera</th>
<th>Moving camera</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>No. of features in the reference image</td>
<td>2199</td>
<td>2127</td>
</tr>
<tr>
<td>2</td>
<td>No. of world features after stereo validation</td>
<td>1054</td>
<td>963</td>
</tr>
<tr>
<td>3</td>
<td>No. of tracked features in two consecutive frames</td>
<td>476</td>
<td>160</td>
</tr>
<tr>
<td>4</td>
<td>Features' column variance in the first iteration ((E_u))</td>
<td>0.0949</td>
<td>107.22</td>
</tr>
<tr>
<td>5</td>
<td>Features' row variance in the first iteration ((E_v))</td>
<td>0.1180</td>
<td>41.635</td>
</tr>
<tr>
<td>6</td>
<td>Features' column variance in the last iteration</td>
<td>0.0946</td>
<td>0.19811</td>
</tr>
<tr>
<td>7</td>
<td>Features' row variance in the last iteration</td>
<td>0.1176</td>
<td>0.0756</td>
</tr>
</tbody>
</table>

Here, two cases are studied. In the first case, the camera is static and in the second, it is mobile (columns 3 and 4). From this table, about 50% of features fail the validation test or have no correspondences in the stereo reconstruction process row 2 of the Table 5.3. Variables
5.4 Dynamic Features and Estimation Reliability

The main task of the motion estimation is to find the 3D motion vector that projects two set of observations onto each other with minimal error.

One of the main assumptions in this work was that most of scene objects are static. In that case the optimum results are gained by eliminating outliers that are detected through their large positional error residual. These outliers originate from either falsely matched features or dynamic scene features. If outliers are from different dynamic objects with random motions they would not affect the estimated results as much as when they are clustered and they all have the same motion. Figure 5.4 shows a static plane, with a dynamic part (highlighted with the bright square in Figure 5.4 a). The motion of the static plane is estimated while the dynamic part is moved. A motion vector of zero, in the last column of Table 5.4, is expected if the system is not overwhelmed by outlier features.

Table 5.4 shows results of this experiment: From these results when the number of dynamic features is more than 35% to 40% of the employed scene features, rows b and c, the accuracy of the estimated motion is questionable.

5.5 Feature Update

After the motion parameters are found, the database information must be updated. This is performed based on the prediction and observation of each feature and the robot's motion.
5.5 Feature Update

The position and uncertainty matrix for features that are expected to be seen and have corresponding matches are updated (details in Chapters 6). Their count increases by 1.

- Positions and covariance matrices for features that are not expected to be seen are also updated; however, their counts stay unchanged.

- Features that are expected to be seen but have no unique matches are updated. The uncertainty for these features is increased by a constant rate of 10% and their count decreases by 1.

- New features, with no correspondence in the reference world, are initialized in the database and their count is set to 1.
5.6 Feature Retirement

After updating global feature points, an investigation is carried out to eliminate those features that have not been seen over some distance. For this purpose, feature points with a count value of −5, indicating that they have not been observed for at least 5 consecutive frames, which for our system corresponds to a distance of 50cm, are eliminated. This condition removes some of the remaining unstable features that falsely pass the stability and disparity conditions in the stereo matching process in spite of their poor conditions. It also decreases the size of the global database and prevents the system from wasting the processing time on such features in later operations.

5.7 Chapter Summary

In this chapter, the feature tracking process as well as motion estimation, are explained. Identification of similar features is performed through a normalized sum of squares differences
with a validity check. The tracking is refined by a novel multi-step scheme that increases the correct match correspondences by up to 30%. The motion is estimated using Newton's least squares minimization. In this estimation, the dynamic scene features are eliminated as outliers, forcing the motion vector to a more reliable solution. For an even more accurate 3D representation, the global location of the features are updated at the end of each frame. Also, features that have not been seen for a while, and will probably not be seen again are eliminated.
Chapter 6

Trajectory Error Modeling

There are several error sources that can affect the accuracy of motion estimation results. Image noise, the quantization associated with images and the locations of found image features can all introduce inaccuracy to the location of the world features, as well as into the overall estimated robot trajectory. Knowing how reliable the 3D feature locations are is key to incorporating the uncertainty factor (position covariance) for each feature that later facilitates the elimination of less reliable features.

The noise associated with an image is considered to be white and Gaussian [18]. The disparity measurement using such an image inherits this noise. Since the 3D estimation of each feature in space is a linear function of the inverse disparity, a Kalman filter estimator seems to be a promising model for reducing the system error associated with the existing noise [36]. Therefore, a Kalman filtering scheme is incorporated into the system, that uses the many measurements of a feature over time and smooths out the effects of noise in the feature positions, as well as in the robot’s trajectory. In this scheme, for each feature an individual Kalman filter is generated. Each Kalman filter includes a $3 \times 1$ mean position vector and a $3 \times 3$ covariance matrix that represent the mean position and the positional uncertainty associated with that feature in space. A Kalman filter is also created for the robot mounted camera, that includes position and the uncertainty associated with it.
6.1 Global 3D Feature Creation

At each frame, the 3D location of the features generated through the stereo algorithm are computed relative to the current position of the camera system. Many of these features get repeated in future frames, and as the robot moves around, these features may come into or go out of view. In order to combine several observations of an identical feature that is seen over two or more frames, it is necessary to represent these features in a reference coordinate system. For this purpose, a global reference coordinate system is created, which is the same as the first camera coordinate system. The processes that are involved in maintaining and updating world features include the following:

I. Initializing world features and the camera pose in a global reference frame. This global reference frame is defined by the initial pose of the camera stereo unit and at the position of reference camera, Figure 7.1.

II. Acquiring the next set of images which become the “current frame” and constructing the next set of observed features in the current camera frame. Also, estimating the camera motion from the previous frame and updating the camera pose accordingly.

III. Using the current robot position to transfer the observed features in the current frame into the global coordinate system.

IV. Matching and combining the measurements of identical scene features.

V. Adding entries to the global set for features that are observed for the first time.

VI. Keeping global features seen before which might be seen in the near future.

VII. Retiring features that have not been seen for a while and may not be seen again.

Each time a feature from the current frame matches an existing global feature, the covariance matrix of the feature in the current frame is transferred into the global coordinate system, where it combines with the uncertainty matrix of the matching feature.
6.2 Camera Position Uncertainty

The robot's position is updated using the estimated motion vector found by the least-squares minimization. For this purpose, a simple Kalman filter model is employed. The only assumption that is made for this model is that the robot moves with a constant velocity. Following equations represent the Kalman filter model for this application:

\[ x_{k+1} = F x_k + \xi_k \]  \hspace{1cm} (6.1)
\[ z_k = H x_k + \eta_k \]  \hspace{1cm} (6.2)

where \( x_k \) represents the state variable at frame \( k \),

\[ x_k = [x, y, z, \phi_x, \phi_y, \phi_z, \dot{x}, \dot{y}, \dot{z}, \dot{\phi}_x, \dot{\phi}_y, \dot{\phi}_z]^T \]  \hspace{1cm} (6.3)

and \( F \) is a constant \( 12 \times 12 \) matrix and is defined by:

\[
F = \begin{bmatrix}
1 & T & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & T & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & T & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & T & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & T & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & T & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\end{bmatrix} \]  \hspace{1cm} (6.4)

In matrix \( F \), \( T \) is the sampling rate and is set to 1. \( \xi_k \) and \( \eta_k \) are respectively (unknown) system and observation noises. \( H \) is a \( 6 \times 12 \) matrix and is defined by \( \begin{bmatrix} I & N \end{bmatrix} \), where \( I \) is a \( 6 \times 6 \) identity and \( N \) is a \( 6 \times 6 \) zero matrices.
6.2 Camera Position Uncertainty

6.2.1 Prediction

Using the standard Kalman filter notation [23], the state prediction is made by

$$x(k + 1|k) = Fx(k|k)$$  \hspace{1cm} (6.5)

If $P(k|k)$ represents the process covariance, the process covariance prediction is

$$P(k + 1|k) = FP(k|k)F^T + Q(k)$$  \hspace{1cm} (6.6)

$Q$ shows the noise associated with the process covariance and is defined by

$$Q(k) = \begin{bmatrix}
0.05 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.05 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.05 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.03 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.03 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.03 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.03 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.03 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
\end{bmatrix}$$  \hspace{1cm} (6.7)

Matrix $Q$ is a constant matrix and is found experimentally. In this matrix, the associated uncertainties with the rotational components of the state variable are defined to be smaller than those of the translational parameters. This is mainly due to the fact that the accuracy of the estimated rotational parameters of the motion is higher. These values however are defined to be larger than the estimated uncertainties associated with the measurements as shown in Equation 6.9. Such larger uncertainties emphasize the fact that the measurement values are more reliable under normal circumstances. However, if for any reason, the least-squared minimization for estimating the motion parameters fails, then the covariance matrix
of the measurements in Equation 6.9 is changed to an identity matrix, forcing the system to give larger weight to the predicted values.

### 6.2.2 Measurement

The measurement prediction is computed as

$$z(k+1|k) = Hx(k+1|k)$$  \hspace{1cm} (6.8)

The new position of the robot is obtained by updating its previous position using estimated motion parameters by the least squares minimization from Equation 5.6. This is the new position measurement for the robot or $x_{LS}$. The covariance $R_{LS}$ for the measurement is obtained by computing the inverse of $J^TJ$ [64] in Section 5.3.1.

Matrix 6.9 represents a typical $R_{LS}$ that is computed by our system during one of the tracking process.

$$R_{LS} = \begin{bmatrix}
0.000001 & 0.000000 & 0.000000 & 0.000001 & -0.000002 & -0.000001 \\
0.000000 & 0.000000 & -0.000000 & 0.000001 & -0.000001 & -0.000000 \\
0.000000 & -0.000000 & 0.000001 & 0.000001 & -0.000001 & -0.000001 \\
0.000001 & 0.000001 & 0.000001 & 0.000005 & -0.000004 & -0.000004 \\
-0.000002 & -0.000001 & -0.000001 & -0.000004 & 0.000006 & 0.000005 \\
-0.000001 & -0.000000 & -0.000001 & -0.000004 & 0.000005 & 0.000007
\end{bmatrix}$$  \hspace{1cm} (6.9)

If for any reason, a feasible result for the measurement vector $x_{LS}$ is not found by the least-squared minimization procedure, $R_{LS}$ is set to a 6 x 6 identity matrix. A $R_{LS}$ with larger components, comparing to $Q$, causes the system to give the prediction values a higher weight than the unknown measurements that are set to zero.
6.2 Camera Position Uncertainty

6.2.3 Update

The filter gain is defined by

\[ W(k + 1) = P(k + 1|k)H^TS^{-1}(k + 1) \]  

(6.10)

The state update is then computed from

\[ x(k + 1|k + 1) = x(k + 1|k) + W(k + 1)[x_{LS} - z(k + 1|k)] \]  

(6.11)

The covariance update is then

\[ P(k + 1|k + 1) = P(k + 1|k) - W(k + 1)HP(k + 1|k) \]  

(6.12)

In summary the Kalman filtering formulation can be presented by the following set of relationships:

\[ P(0|0) = Var(x_0) \]
\[ P(k + 1|k) = FP(k|k)F^T + Q(k) \]
\[ W(k + 1) = P(k + 1|k)H^T[H^TP(k + 1|k)H^T + R_{LS}]^{-1} \]
\[ P(k + 1|k + 1) = P(k + 1|k) - W(k + 1)HP(k + 1|k) \]
\[ x(k + 1|k) = Fx(k|k) \]
\[ x(k + 1|k + 1) = x(k + 1|k) + W(k + 1)(x_{LS} - z(k + 1|k)) \]
\[ k = 1, 2, \ldots \]

Figure 6.1 also represents a graphical presentation of the Kalman filtering model that is used for the linear motion of the camera.
6.3 Feature Position Uncertainty

Uncertainties in the image coordinates and disparity values of the features from the stereo algorithm propagate to uncertainty in the features' 3D positions. A feature point at \((u, v)\) on the image plane, with a disparity value of \(d\), has the world corresponding location \((x, y, z)\) that is found by the inverse perspective projection:

\[
\begin{align*}
x &= \frac{(C_x - u)b}{d} \\
y &= \frac{(v - C_y)b}{d} \\
z &= \frac{fb}{d}
\end{align*}
\]

Where \((C_x, C_y)\), \(b\) and \(f\) represent the image center, stereo camera separation and camera focal length. A first-order error propagation model [15] provides the positional uncertainties associated with each feature.

\[
\begin{align*}
\sigma^2_x &= \frac{b^2 \sigma^2_{C_x}}{d^2} + \frac{b^2 (C_x - u)^2 \sigma^2_d}{d^4} \\
\sigma^2_y &= \frac{b^2 \sigma^2_{C_y}}{d^2} + \frac{b^2 (v - C_y)^2 \sigma^2_d}{d^4} \\
\sigma^2_z &= \frac{f^2 b^2 \sigma^2_d}{d^4}
\end{align*}
\]

Where, \(\sigma^2_x\), \(\sigma^2_y\), \(\sigma^2_z\), \(\sigma^2_{C_x}\), \(\sigma^2_{C_y}\) and \(\sigma^2_d\) are the variances of \(x\), \(y\), \(z\), \(C_x\), \(C_y\) and \(d\), respectively. Based on the results given in Section 5.3.3, where the mean of error in the
least-squares minimization is less than one pixel, assumptions are made that \( \sigma^2_{c_x} = 0.5 \), \( \sigma^2_{c_y} = 0.5 \) and \( \sigma^2_d = 1 \).

Knowing the intrinsic parameters of the camera system, we compute the variances for each feature's 3D position, in the current camera frame coordinate, according to the above error propagation formula in the stereo process.

### 6.4 Feature Update

Each time a feature is observed, a new set of measurements is obtained for that feature in the space. Therefore, at the end of each frame and after estimating the motion, world features found in the current frame are used to update the existing global feature set. This requires that these features be transformed into the global coordinate system first. Next, the positional mean and covariance of each feature are combined with corresponding matches in the global set. The 3D uncertainty of a feature in the current frame is computed as described by Equations in 6.15. However, when this feature is transformed into the global coordinate system the uncertainty of the motion estimation and robot position propagates to the feature's 3D position uncertainty in the global frame. Therefore, before combining the measurements, the 3D positional uncertainties of the feature are updated first.

From the least-squares minimization procedure, the current robot pose, as well as its covariance \( ([J^T J]^{-1}) \), are obtained [64] (Section 5.3.1). The current position of the features can be transferred into the reference frame by

\[
P_{\text{new}} = (R_Y (R_X (R_Z (P_{\text{obs}})))) + T
\]

(6.16)

where \( P_{\text{obs}} \) and \( P_{\text{new}} \) are the observed position in the current frame and the corresponding transformed position in the reference frame. \( T \) is the translational transformation and \( R_Z \), \( R_X \) and \( R_Y \) are the rotational transformations (roll, pitch and yaw) required around each
6.4 Feature Update

one of the three axes defined by

\[
R_z = \begin{bmatrix}
\cos(\phi_z) & -\sin(\phi_z) & 0 \\
\sin(\phi_z) & \cos(\phi_z) & 0 \\
0 & 0 & 1
\end{bmatrix}
\] (6.17)

\[
R_x = \begin{bmatrix}
0 & 0 & 1 \\
0 & \cos(\phi_x) & -\sin(\phi_x) \\
0 & \sin(\phi_x) & \cos(\phi_x)
\end{bmatrix}
\] (6.18)

\[
R_y = \begin{bmatrix}
\cos(\phi_y) & 0 & \sin(\phi_y) \\
0 & 1 & 0 \\
-\sin(\phi_y) & 0 & \cos(\phi_y)
\end{bmatrix}
\] (6.19)

6.4.1 Feature covariance update

The goal is to obtain the covariance of the features in the reference coordinate system, given the diagonal uncertainty matrix for each observed feature in the current frame consisting of \(\sigma_x^2\), \(\sigma_y^2\) and \(\sigma_z^2\) (Equation set 6.15). For this purpose, the new covariance matrix \((\Sigma_{new})\) of a feature is combined with the previous covariance matrix of the matched feature in the global set \((\Sigma_{KF})\) to obtain the new covariance matrix \((\Sigma_{KF'})\). Given

\[
X' = PX
\] (6.20)

where \(P\) is a 3 x 3 transformation matrix and \(X\) and \(X'\) are the old and new position vectors, 3 x 1. If there are errors associated with both \(P\) and \(X\), \(\Lambda_P\) (9 x 9 covariance for \(P\)) and, \(\Lambda_X\) (3 x 3 covariance for \(X\)), the 3 x 3 covariance of the resulting vector \(X'\) is computed
In Equation 6.21, the first matrix is a $3 \times 12$, the second is a $12 \times 12$ and the third, which is the transpose of the first matrix, is a $12 \times 3$ matrix. With the assumption that at each time the three rotation angles are small and therefore independent, the transformation proceeds, in order, for rotations $R_Z$ (roll), $R_X$ (pitch), $R_Y$ (yaw) first, and then the translation $T$. Variances of $\sigma_{\phi_x}^2$, $\sigma_{\phi_y}^2$ and $\sigma_{\phi_z}^2$ are already found during the last motion estimation. Required transformations for each stage and how the positional uncertainties propagate are explained next.

**6.4.1.1 Roll transformation**

With the assumption that noises associated with the rotational angles $\phi_x$, $\phi_y$ and $\phi_z$ are Gaussian and of zero mean, the $9 \times 9$ covariance matrix for the roll transformation is computed by [24]:

$$\Lambda_{X'} = \begin{bmatrix} X^T & 0 & 0 \\ 0 & X^T & 0 \\ 0 & 0 & X^T \end{bmatrix} P \begin{bmatrix} \Lambda_P & 0 \\ 0 & \Lambda_X \\ 0 & 0 & X \end{bmatrix} \begin{bmatrix} X & 0 & 0 \\ 0 & X & 0 \\ 0 & 0 & X \end{bmatrix}^T$$ (6.21)
6.4 Feature Update

by

\[
\begin{bmatrix}
A & 0 & 0 & 0 & A & 0 & 0 & 0 & 0 \\
0 & B & 0 & -B & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -B & 0 & B & 0 & 0 & 0 & 0 & 0 \\
A & 0 & 0 & 0 & A & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}
\]

(6.22)

where \( A \) and \( B \) are defined by \( \text{Variance}(\cos \phi_z) \) and \( \text{Variance}(\sin \phi_z) \). By definition

\[
\text{Variance}(\cos \phi_z) = E(\cos^2 \phi_z) - E^2(\cos \phi_z)
\]

(6.23)

\[
\text{Variance}(\sin \phi_z) = E(\sin^2 \phi_z) - E^2(\sin \phi_z) = E(\sin^2 \phi_z)
\]

(6.24)

The expected value of \( \cos^2 \phi_z \) is computed [76], with the assumption that \( \phi_z \) has a Gaussian distribution, by

\[
E(\cos^2 \phi_z) = \frac{1}{\sqrt{2\pi}\sigma_{\phi_z}} \int_{-\infty}^{+\infty} e^{-\frac{z^2}{2\sigma_{\phi_z}^2}} \frac{1 + \cos 2\phi_z}{2} d\phi_z
\]

(6.25)

\[
= \frac{1}{2} + \frac{1}{2} e^{-2\sigma_{\phi_z}^2}
\]

also

\[
E(\cos \phi_z) = e^{-\sigma_{\phi_z}^2}
\]

(6.26)
therefore

\[ A = \frac{1}{2}(1 + e^{-2\sigma^2_x} - 2e^{-2\sigma^2_z}) \]  
\( (6.27) \)

In the same manner \( B \) is computed by

\[ B = \frac{1}{2}(1 - e^{-2\sigma^2_x}) \]  
\( (6.28) \)

Using Equations 6.21 and 6.17, the covariance matrix after the roll, \( \phi_z \), rotation is computed from

\[
\Lambda_{\phi_z} = \begin{bmatrix}
Ax^2 + By^2 + \sigma_x^2\cos^2\phi_z & (A - B)xy & \sigma_x^2\cos\phi_z \\
-2\sigma_{xy}\cos\phi_z\sin\phi_z & +\sigma_y^2(\cos^2\phi_z - \sin^2\phi_z) & -\sigma_{yz}\sin\phi_z \\
+\sigma_y^2\sin^2\phi_z & +(\sigma_z^2 - \sigma_y^2)\sin\phi_z\cos\phi_z & \\
(A - B)xy & Ay^2 + Bx^2 + \sigma_z^2\sin^2\phi_z & \sigma_z^2\sin\phi_z \\
+\sigma_{xy}^2(\cos^2\phi_z - \sin^2\phi_z) & +2\sigma_{xy}\sin\phi_z\cos\phi_z & +\sigma_{yz}\cos\phi_z \\
+(\sigma_z^2 - \sigma_y^2)\sin\phi_z\cos\phi_z & +\sigma_y^2\cos^2\phi_z & \\
\sigma_x^2\cos\phi_z & \sigma_x^2\sin\phi_z \\
-\sigma_x^2\sin\phi_z & +\sigma_{yz}\cos\phi_z & \sigma_y^2 \\
\end{bmatrix} \]  
\( (6.29) \)

Here, \((x, y, z)\) is the 3D location of the feature in the current frame. Since this is the first time the transformation is carried out, \( \sigma_{xy}^2 = \sigma_{xz}^2 = \sigma_{yz}^2 = 0 \) as the initial covariance matrix of the observation is a diagonal matrix. Applying the roll transformation to the initial position of the feature provides the transformed position of the feature. This new position is used for the next stage. The uncertainty associated with this position is the recent covariance matrix of \( \Lambda_{\phi_z} \).
6.4.1.2 Pitch transformation

The $9 \times 9$ covariance matrix for the pitch rotation, $\phi_z$, is computed in a similar way, as shown in Equation 6.22. Once again, substituting the pitch rotational transform in Equation 6.21 and the covariance matrix in 6.29, the new covariance matrix can be defined by

$$
\Lambda_{\phi_z \phi_z} = \begin{bmatrix}
\sigma_z^2 & \sigma_{xz}^2 \cos \phi_z & \sigma_{yz}^2 \sin \phi_z \\
-\sigma_{xz}^2 \sin \phi_z & \sigma_{xx}^2 + D z^2 & (C - D) y z \\
-2\sigma_{xz}^2 \cos \phi_z \sin \phi_z & +\sigma_{yz}^2 \cos \phi_z + \sigma_{zz}^2 \phi_z + (\sigma_{yz}^2 - \sigma_{zz}^2) \cos \phi_z \sin \phi_z \\
-2\sigma_{yz}^2 \cos \phi_z \sin \phi_z & +\sigma_{yz}^2 (\cos^2 \phi_z - \sin^2 \phi_z) + 2\sigma_{zz}^2 \cos \phi_z \sin \phi_z & +\sigma_{zz}^2 \cos \phi_z - \sin^2 \phi_z
\end{bmatrix}
$$

(6.30)

where $C$ and $D$ are defined by

$$
C = \frac{1}{2} (1 + e^{-2\phi_z^2} - 2 e^{-\phi_z^2})
$$

(6.31)

$$
D = \frac{1}{2} (1 - e^{-2\phi_z^2})
$$

(6.32)

In this formula, $\sigma_{xz}^2$ is found from the last motion estimation. $(x, y, z)$ is the transformed 3D location of the feature after the roll transformation and $\sigma_z^2, \sigma_y^2, \sigma_x^2, \sigma_{xy}^2, \sigma_{xz}^2$ and $\sigma_{yz}^2$ are from the covariance matrix $\Lambda_{\phi_x}$, Equation 6.29. Applying the pitch transform provides the transformed position of the feature, which is used in the next stage together with this new feature covariance.
6.4.1.3 Yaw transformation

The $9 \times 9$ covariance matrix after the yaw rotation is computed by

\[
\Lambda_{\phi_x, \phi_y} = \begin{bmatrix}
Ex^2 + Fz^2 & \sigma_{xy}^2 \cos \phi_y & (E - F)xz \\
\sigma_{xy}^2 \cos \phi_y + \sigma_z^2 \sin \phi_y & \sigma_{zy}^2 \sin \phi_y + (\sigma_z^2 - \sigma_x^2) \sin \phi \cos \phi_y & +2\sigma_{xz}^2 \sin \phi_y \cos \phi_y \\
+2\sigma_{zy}^2 \sin \phi_y \cos \phi_y & +\sigma_{xz}^2 (\cos^2 \phi_y - \sin^2 \phi_y) & +\sigma_{zx}^2 \sin \phi_y \cos \phi_y \\
(E - F)xz & \sigma_{zy}^2 \sin \phi_y & Ez^2 + Fx^2 \\
+\sigma_{zy}^2 \sin \phi_y & -\sigma_{zy}^2 \cos \phi_y & +\sigma_{zx}^2 \sin^2 \phi_y + \sigma_{zx}^2 \cos^2 \phi_y \\
+\sigma_{zx}^2 (\cos^2 \phi_y - \sin^2 \phi_y) & -2\sigma_{zx}^2 \sin \phi_y \cos \phi_y & 
\end{bmatrix}
\] (6.33)

$E$ and $F$ are defined by:

\[
E = \frac{1}{2} (1 + e^{-2\sigma_{\phi_y}^2} - 2e^{-\sigma_{\phi_y}^2}) \quad (6.34)
\]

\[
F = \frac{1}{2} (1 - e^{-2\sigma_{\phi_y}^2}) \quad (6.35)
\]

and $\sigma_{\phi_y}^2$ is the variance of $\phi_y$ estimated earlier in the motion estimation process. $(x, y, z)$ is the transformed 3D location of the feature after the pitch transformation and $\sigma_x^2, \sigma_y^2, \sigma_z^2, \sigma_{xy}^2, \sigma_{xz}^2$ and $\sigma_{yz}^2$ are from the covariance matrix $\Lambda_{\phi_x, \phi_y}$, Equation 6.29. Applying the yaw transformation gives the transformed position of the feature, which is used along with the new covariance in the next stage.
6.4 Feature Update

6.4.1.4 Translational transformation

The final transformed covariance for each feature in the global coordinate system is computed from:

\[
\Sigma_{\text{new}} = \Lambda_{\phi_x \phi_y \phi_z} + \begin{bmatrix}
\sigma_x^2 & 0 & 0 \\
0 & \sigma_y^2 & 0 \\
0 & 0 & \sigma_z^2
\end{bmatrix}
\]  \hspace{1cm} (6.36)

Here \(\sigma_x^2\), \(\sigma_y^2\) and \(\sigma_z^2\) are the translational uncertainties associated with the estimated motion and found by the first three diagonal components of matrix \([J^TJ]^{-1}\) computed in Section 5.3.1.

6.4.2 Feature position update

To update the 3D position of a feature [89], the transformed covariance matrix, \(\Sigma_{\text{new}}\), is combined with the existing covariance of the matching global feature, \(\Sigma_{KF}\), to obtain the new covariance matrix, \(\Sigma'_{KF}\):

\[
\Sigma'_{KF} = (\Sigma_{KF}^{-1} + \Sigma_{\text{new}}^{-1})^{-1}
\]  \hspace{1cm} (6.37)

The new global position of the feature, \(S'_{KF}\), is then found using the covariances, the transformed position (using Equation 6.16) and the previous position.

\[
S'_{KF} = \Sigma'_{KF}(\Sigma_{KF}^{-1}S_{KF} + \Sigma_{\text{new}}^{-1}r_{\text{new}})
\]  \hspace{1cm} (6.38)

Here, \(S_{KF}\) represents the global position of a feature in the database and \(r_{\text{new}}\) represents the new position of the same feature in the global coordinate system.
Figure 6.2: Positional uncertainties associated with features in the image plane.

Figure 6.2 shows the projection of estimated uncertainties associated with world features on the image plane. In this figure, the uncertainties associated with closer objects are very small and therefore appear as bright dots. As expected farther features, for instance features around windows on the upper right corner of the scene, have larger projected uncertainties. Some of the closer features also have large positional uncertainties associated with them. These large positional uncertainties imply wrong depth estimation for those features.

To get a closer look at 3D scene features and their positional uncertainties Figure 6.3 is generated. It demonstrates two top views of world features and their associated uncertainties. Figure 6.3 a represents a look at the features located on close distances, up to 3.5 meters, from the image plane. Figure 6.3 b represents a broader view of the uncertainty space for
the entire existing features. As clearly displayed, associated positional uncertainties with features grow in dimensions as these features move away (in depth) from the camera plane and as they move to the sides (from the camera center).

Figure 6.3: a- Projected relative positional variances of features with distances up to 3.5 meters from the camera on the XZ plane. b- Top view of positional uncertainties for all features.
Figure 6.4 shows the updated positional uncertainties of world features and their corresponding initial positional uncertainties. In this figure, dotted ellipsoids show the original and the solid ellipsoids show the updated uncertainties.

Results of the Kalman filters incorporated into the trajectory tracking system are presented in Chapter 7.

6.6 Chapter Summary

In this chapter the error modeling is described. The main idea is to combine the measurements of an identical feature viewed several times to reduce the error originating from existing noise in the measurement of such feature. For this purpose, for each feature a 3D ellipsoid is created that represents the 3D positional uncertainty associated with that feature. Such ellipsoids are updated in each frame and depending on the visibility of features, and location of the features with respect to camera they might expand or shrink. World features with large
uncertainties are eliminated as they might not be very reliable. A Kalman filtering model is created for the camera (robot) motion. With the assumption of constant translational and rotational velocities, the camera location on the next frame is predicted and refined using the camera position measurement gained through the least-squares minimization process.
Chapter 7

Implementation Results

This chapter contains the experimental results obtained from implementing the solution strategies put forth in previous chapters. These results provide trajectory estimations for several indoor and outdoor experiments. They also discuss system processing time and accuracy as well as some related issues.

7.1 Camera System: Digiclops™

The Digiclops stereo vision camera system, Figure 7.1, is designed and built by Point Grey Research [45]. It provides real-time digital image capture for different applications. It includes three monochrome cameras and a software system with the IEEE-1394 interface. These three cameras are rigidly positioned so that each adjacent pair is horizontally or vertically aligned.

The Digiclops system is equipped with a standard C/C++ interface library and is capable of acquiring and displaying continuous real-time digital video from the camera. It is also capable of real-time depth and 3D image construction with a speed of 30 to 4.3Hz depending on the image size. In this work, the Digiclops is used only for the purpose of image acquisition. The main reason for this decision is due to the fact that not all the image pixels carry...
significant information about the camera motion. Therefore, reconstruction of the 3D map of the entire viewed scene could waste the time that is required for other tasks of the system. In this work, the intrinsic camera parameters are also provided by Point Grey Research. The camera system captures gray scale images of $320 \times 240$ pixels. In order to reduce the ambiguity between the yaw rotation and sideways movements, a set of wide angle lenses with a $104^\circ$ field of view, VL-2020 by Universe Kogaku America, is used. These lenses incorporate information from the sides of the images that behave differently under translational and rotational movements.

### 7.2 Trajectory Estimation

The performance of the system is evaluated based on its cumulative trajectory error or positional drift. For this purpose a series of experiments are performed in indoor and outdoor environments. These experiments are either performed for known paths or closed paths. For a known path the location of the robot is compared either with a destination point or some known points along the path. The location of the destination or known points are measured ahead of time by a human operator. For closed path experiments, the robot starts from an initial point with an initial pose. After moving around, it returns to the starting point. To ensure returning to the exact initial pose, an extra set of images are acquired at the starting position, right before the robot starts its motion. This set is used as the last set and since a second image will differ from the first image by noise and is thus more appropriate to use.
than the first image again. For perfect tracking, the expected cumulative positional error over the whole trajectory should be zero and therefore anything else represents the system's drift.

The first two presented experiments, 1 and 2, show the performance of the system using unwarped images. Experiments 3 and 4, however, show system's output using warped images.

The error was observed to be increasing with distance and consequently the use of percent error - being the error divided by the total traversed distance seemed reasonable as then the percent error is roughly constant.

7.2.1 Experiment 1:

This experiment is intended to investigate the localization accuracy. It represents the performance of the system at an earlier stage of this thesis when unwarped images were processed for the trajectory estimation purpose. The camera is moved along a route from point A to the known point B, Figure 7.2. A number frames (32) are processed along this route, white line segment in this figure.

Figure 7.2: The indoor scene from camera point of view.

Figure 7.3 represents the estimated 3D path of the system. The gradual motion of
the camera system is displayed using a graphical interface that is written in Visual C++ and Visual Toolkit (VTK) 4.0. In this figure the estimated location and orientation of the camera along the path are shown with red spheres and 3D yellow cones.

Figure 7.3: The graphical presentation of the 3D camera motion while moving from point A to point B, described in Experiment 7.2.1, using Visual Toolkit 4.0. In this figure the position is shown with a red sphere and the orientation with a yellow cone.

As shown, the camera rotates as well as translates. The coordinates of point B are measured manually in a coordinate system with origin A. Table 7.1 provides the estimated location and compares it with the actual location of point B. The path in this experiment is a straight line and it includes translation as well as rotation. The motion range includes an overall translation of 158.9cm along Z and an overall yaw rotation of 45°. From these results, the drift of the system is about 4.16 centimeters with an orientational drift of 4.20 degrees.
Table 7.1: Localization error for the experiment 7.2.1 (using completely calibrated images).

<table>
<thead>
<tr>
<th>Motion range</th>
<th>Estimation error</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D_x (\text{cm}), D_y (\text{cm}), D_z (\text{cm})$</td>
<td>$E_{D_x} (\text{cm}), E_{D_y} (\text{cm}), E_{D_z} (\text{cm})$</td>
</tr>
<tr>
<td>$\phi_x (^\circ), \phi_y (^\circ), \phi_z (^\circ)$</td>
<td>$\phi_x (^\circ), \phi_y (^\circ), \phi_z (^\circ)$</td>
</tr>
<tr>
<td>0.00, 0.00, -158.90</td>
<td>0.4, -0.86, 4.05</td>
</tr>
<tr>
<td>0.00, 45.0, 0.00</td>
<td>-1.25, -3.97, 0.59</td>
</tr>
</tbody>
</table>

7.2.2 Experiment 2:

The next experiment is designed to measure the resulting drift using a closed path. For this purpose, the camera starts moving from point A on an arbitrary route and then returns to its starting point. Along this path 71 consecutive frames are processed. The images for this experiment were fully calibrated. The subject environment is an indoor scene with object on distances that range between 0.5 meter to about 8 meters from the camera image plane. Figure 7.4 represents the graphical presentation of the estimated 3D positions and orientations of the camera along this path by the system using VTK 4.0.

Figure 7.4: 3D position and orientation of the mobile camera in experiment 7.2.2 using VTK 4.0.
7.2 Trajectory Estimation

Table 7.2 represents the existing drift in this case. In this table Motion range represents the absolute amount of the motion along each coordinate variable. The motion range along the z axis is 2.94 m, along the x axis is 1.25 m with a yaw rotation of 10°. To show the improvement resulting from Kalman filtering, the results are presented once without the Kalman filter and once with it. It can be seen that the system with the Kalman filter has about 8 cm of positional drift.

<table>
<thead>
<tr>
<th>Kalman Filter</th>
<th>Motion range $D_z$,(cm), $D_y$,(cm), $D_z$,(cm)</th>
<th>Estimation error $E_{D_z}$,(cm), $E_{D_y}$,(cm), $E_{D_z}$,(cm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Off</td>
<td>125, 0, 294, $0^\circ$, $10^\circ$, $0^\circ$</td>
<td>13.68, 8.06, 4.04 $E_{\phi_z}$, $E_{\phi_y}$, $E_{\phi_z}$</td>
</tr>
<tr>
<td>On</td>
<td>125, 0, 294, $0^\circ$, $10^\circ$, $0^\circ$</td>
<td>$-1.77$, 7.57, $-0.45$ $E_{\phi_z}$, $E_{\phi_y}$, $E_{\phi_z}$</td>
</tr>
</tbody>
</table>

The large amount of cumulative error could be explained by the fact that most of the scene features were located in fairly far distances from the camera plane (about 8 meters far). Only a small number of scene features, 10% of the entire features, were located on distances within 1 meter from the camera plane. Generally for an accurate trajectory estimation, there should be a good balance between the close and far features in the scene. This is due to the fact that close features affect the precision improvement of the found motion as they have more accurate 3D reconstructed locations. Far features, however, help in the separation of motion components from each other. Also, they are more prone to sudden positional changes. For instance if the camera undergoes a large motion, while closer features may go under dramatic shape changes or simply fall out of the viewable scene, features at farther distances still remain viewable. This characteristic of far features assists the robustness of the system in dealing with such condition.

Therefore, a tilted camera can create images that include objects on the floor as well as
surrounding areas. In the case of this example however, a tilted view would not be advisable as the floor includes no visible textures.

7.2.3 Experiment 3:

In this experiment the robot moves along an outdoor path and the system performs using warped images. The scene was a natural environment including trees, leaves and building structures that were located in distances between 0.1 to 20 meters from the camera image plane. For this environment, most of the scene features belong to leaves with a small amount of dynamic motion due to the breeze. Other possible scene components such as sidewalks and building walls, in which there are only a few features, are problematic for feature-based vision systems. Figure 7.5 shows the scene in this experiment. This environment allowed us to test the stability of the system in an outdoor environment with more non-static elements in the image. The traversed path was 6 meters long and along the path 172 warped images were captured and processed. In this figure the traversed path is highlighted with a red line.

Figure 7.5: The outdoor scene for experiment 7.2.3.
and the orientation of the camera is shown using a white arrow. The robot starts the forward motion from point A to point B. At point B the backward motion begins until point A is reached. Figure 7.6 26-a, 7.6 96-a and 7.6 151-a represent a number of processed warped images by the system during this experiment. Although the scene includes some structures from the building, but most of the features, over 90%, belong to the unstructured objects of the scene. The estimated 3D path, starting from the first frame till each displayed image frame, is shown in Figures 7.6 26-b, 7.6 96-b and 7.6 151-b. The estimated trajectory at each frame is shown with the red sphere and the orientation is displayed with the yellow cone. The center of the reference camera is considered as the center of the motion.

Figure 7.7 shows a closer look at the overall estimated trajectory along the entire experiment. Table 7.3 displays the cumulative trajectory error in this experiment. From this table the translation error is about 2.651cm which in this case is only 0.4% of the overall translation. To show the repeatability of the same results, another experiment is performed in which a 3 meter path is traveled. Table 7.4 displays the the error associated with this experiment. The error associated with this case is about 0.54%.

<table>
<thead>
<tr>
<th>Cumulative error</th>
<th>$E_{Dx}$ (Cm)</th>
<th>$E_{Dy}$ (Cm)</th>
<th>$E_{Dz}$ (Cm)</th>
<th>$E_{\phi x}$ (Degree)</th>
<th>$E_{\phi y}$ (Degree)</th>
<th>$E_{\phi z}$ (Degree)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Experiment 7.2.3</td>
<td>-1.93000</td>
<td>1.74472</td>
<td>0.52874</td>
<td>-0.06628</td>
<td>0.00783</td>
<td>-1.00911</td>
</tr>
</tbody>
</table>

Table 7.4: 3D drift in the estimated camera trajectory for a 3 meter long path.

<table>
<thead>
<tr>
<th>Cumulative error</th>
<th>$E_{Dx}$ (Cm)</th>
<th>$E_{Dy}$ (Cm)</th>
<th>$E_{Dz}$ (Cm)</th>
<th>$E_{\phi x}$ (Degree)</th>
<th>$E_{\phi y}$ (Degree)</th>
<th>$E_{\phi z}$ (Degree)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 meters long path</td>
<td>-1.3830</td>
<td>-0.4921</td>
<td>0.7340</td>
<td>-0.1578</td>
<td>0.0231</td>
<td>0.5980</td>
</tr>
</tbody>
</table>
7.2 Trajectory Estimation

7.2.4 Experiment 4:

In this experiment the robot moves on a closed circular path including a full 360° yaw rotation. The orientation of the camera is toward the ground. Figure 7.8 a represents the overview of the environment in this scenario and Figure 7.8 b shows the scene from robot point of view.

During this experiment, 101 warped images are captured and processed. Some of these images are shown in Figure 7.9. To make it easier for the viewer to sense the motion between frames, along each row of this figure, bright circles highlight identical features of the scene. Once again the presented results in this experiment are based on processing scene warped images. The estimated trajectory from the first frame until corresponding frames in Figure 7.9, are shown in Figure 7.10.
Figure 7.6: Graphical 3D representation of the motion using VTK 4.0.
7.2 Trajectory Estimation

Figure 7.7: The graphic representation of the traced path in Experiment 7.2.3 using VTK 4.0.

Figure 7.8: a-The outdoor scene used for the rotational motion. b-An up close view of the same scene from robot's point of view.
Figure 7.9: A number of images from the sequence that is used in Experiment 7.2.4.
Figure 7.10: Graphical presentation of the 3D estimated camera motion on a circular path with a yaw rotation of 360° using VTK 4.0.
Figure 7.11 represents the overall estimated trajectory from a closer distance. The cumulative error in this case is represented in Table 7.5.

![Visualization Toolkit - Win32OpenGL #1](image)

Figure 7.11: A closer view of the estimated circular path with a radius of 30cm that is traversed by the camera system in Experiment 7.2.4.

From this table the overall rotational error in this Experiment is about 3.341° or 0.9% and the translational error is 1.22cm or about 0.6% of the overall translation.
7.3 Trinocular and Binocular Stereo Comparison

Table 7.5: Estimated 3D drift for the camera motion with 360° rotation on a circular path with a diameter of 60cm.

<table>
<thead>
<tr>
<th>Cumulative error</th>
<th>$E_{D_x}$ (Cm)</th>
<th>$E_{D_y}$ (Cm)</th>
<th>$E_{D_z}$ (Degree)</th>
<th>$E_{\phi_x}$ (Degree)</th>
<th>$E_{\phi_y}$ (Degree)</th>
<th>$E_{\phi_z}$ (Degree)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Experiment 7.2.4</td>
<td>1.030093</td>
<td>-0.252289</td>
<td>0.603087</td>
<td>-1.07127</td>
<td>-2.59989</td>
<td>1.18193</td>
</tr>
</tbody>
</table>

7.3 Trinocular and Binocular Stereo Comparison

Establishing accurate match correspondences in a stereo system is a key issue in the accuracy and precision of the 3D reconstruction and trajectory tracking problems. The physical arrangement of the cameras in stereo vision has an important role in the correspondence matching problem. As shown in Equation 4.9, the accuracy of the depth reconstruction has a direct relationship with the baseline and it can be improved by choosing a wider separation between the stereo lenses. On the other hand a narrower baseline facilitates a faster search scheme when establishing the correspondences in the stereo image pair. The use of more than one stereo camera was originally introduced to compensate for the trade-off between the accuracy and ease of the match correspondences [49].

As shown in Figure 7.1 the stereo baselines are almost identical in length. Therefore the improvement of the accuracy by means of multi-baseline stereo matching is not expected. However, the non-collinear arrangement of the lenses adds a multiple view of the scene that could improve the robustness and therefore the long term system accuracy. This is mainly because:

- Generally a broader scope of the scene is viewable by the three images increasing the number of the features.

- Moreover, the third image is used for a consistency check, eliminating a number of unstable features that are due to shadows and light effects.

- Often close objects to the camera plane have a large shape change. Sometimes a feature,
7.3 Trinocular and Binocular Stereo Comparison

depending on its position in the scene, is recognizable only in one pair of stereo images. Switching between the stereo pairs when necessary, prevents elimination of the feature points that even though are not identifiable in both stereo sets, but are stable enough to be used in the system.

- The perpendicular arrangement of the two stereo systems improves the accuracy by creating a condition in which the recognition of the periodic patterns in the horizontal and vertical directions, such as horizontal and vertical lines in man made objects and environments, is less ambiguous.

The above improvement however could potentially cause a slow down in the system as each time there is one extra image to be processed.

To assess the effectiveness of trinocular stereo versus binocular, an experiment is conveyed in which a closed path is traversed. Once again the cumulative error is used as a measure of system performance. Figure 7.3 represents a number of warped images, among a set of 32, that are taken on a path of approximately one meter in length.

Table 7.6 shows the resultant error vector in both cases. In this table $E_T$ and $E_O$ represent

<table>
<thead>
<tr>
<th>Cumulative error</th>
<th>$E_{D_x}$</th>
<th>$E_{D_y}$</th>
<th>$E_{D_z}$</th>
<th>$E_{\phi_x}$</th>
<th>$E_{\phi_y}$</th>
<th>$E_{\phi_z}$</th>
<th>$E_T$</th>
<th>$E_O$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trinocular</td>
<td>-1.138</td>
<td>-0.029</td>
<td>-0.233</td>
<td>0.178</td>
<td>0.005</td>
<td>0.189</td>
<td>1.162</td>
<td>0.259</td>
</tr>
<tr>
<td>Binocular</td>
<td>-1.770</td>
<td>-0.402</td>
<td>0.580</td>
<td>-0.055</td>
<td>0.135</td>
<td>-0.031</td>
<td>1.906</td>
<td>0.149</td>
</tr>
</tbody>
</table>

the overall translational and rotational errors. These results clearly show the similarity of estimated motions based on the number of used cameras. Figure 7.3 displays the traveled distance along the $x$ direction for this experiment.

Although the trinocular provides a slightly smaller error vector, the estimation based on the binocular stereo is still quite accurate. Considering that the cost and the complexity of
7.4 Trajectory Estimation Refinement by Kalman Filtering

Comparison of the estimated trajectory with and without a Kalman filtering scheme is represented through the comparison of the cumulative error in 3D trajectory parameters. This comparison is studied for the case in Experiment 7.2.3, in which the traversed path

Figure 7.12: Selected number of images viewed while traveling on a path. The motion of an object is highlighted using the yellow circle.

a binocular system is less than a trinocular stereo system, the binocular stereo might be a better solution for some applications.
7.4 Trajectory Estimation Refinement by Kalman Filtering

Figure 7.13: Estimated traveled distance along x[cm] for a 1m closed loop.

is 6 meters long. Table 7.7 represents results of this comparison. In this table $E_T$ and $E_O$ represent overall translational and rotational drifts. The overall translational error with a

Table 7.7: Comparison of the reduction of 3D drift for a 6 meter long path using Kalman filter.

<table>
<thead>
<tr>
<th>Cumulative error</th>
<th>$E_{D_x}$ (Cm)</th>
<th>$E_{D_y}$ (Cm)</th>
<th>$E_{D_z}$ (Cm)</th>
<th>$E_{\phi_x}$ (Degree)</th>
<th>$E_{\phi_y}$ (Degree)</th>
<th>$E_{\phi_z}$ (Degree)</th>
<th>$E_T$ (Cm)</th>
<th>$E_O$ (Degree)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kalman filter on</td>
<td>-1.930</td>
<td>1.745</td>
<td>0.529</td>
<td>-0.066</td>
<td>0.008</td>
<td>-1.009</td>
<td>2.655</td>
<td>1.011</td>
</tr>
<tr>
<td>Kalman filter off</td>
<td>-4.092</td>
<td>-0.138</td>
<td>4.805</td>
<td>-0.277</td>
<td>-0.262</td>
<td>-0.104</td>
<td>6.313</td>
<td>0.395</td>
</tr>
</tbody>
</table>

Kalman filter is considerably less than that when Kalman filtering scheme is turned off. The rotational error with the Kalman filtering is slightly more. However, both these rotational drifts are very small, 1.011 and 0.395 degrees, and they can easily be due to the noise in the estimation process.

Figures 7.14 represents the estimated trajectories in the presence of Kalman filtering scheme. As shown in Figure 7.14's top sub-figure, the robot moves along X for 3 meters in the forward direction and then it returns to its starting point. The overall translational
error for this experiment is about 2.65 centimeters. Figure 7.15 represents the estimated orientation at each frame for this experiment. The cumulative orientation error for this experiment is about 1°. The exact amount of cumulative errors are shown on the first row of Table 7.7.

![Graphs showing estimated camera distances relative to a starting point for a 6 meter long path with Kalman filter.]

Figure 7.14: Estimated camera distances relative to a starting point for a 6 meter long path with Kalman filter.

Figures 7.16 and 7.17 show results for the same experiment when the Kalman filtering process is turned off. As represented in the second row of Table 7.7, the positional error is increased to 6.31 centimeters. The cumulative orientational error in this case is smaller than 1°.
7.5 Computation Time

The presented system is implemented in Microsoft Visual C++ 6.0 language, on a 1.14 GHz AMD Athlon™ processor under Microsoft Windows® operating system. The camera system captures gray scale images of 320×240 pixels at 8Hz rate. The most effort has been done to optimize the code and modulate the system in order to obtain fast subsystems with less communicational cost and required memory.

The most severe drawback of the system is its high computational requirement. Especially, when finding and evaluating match correspondences. Computation is mostly spent on performing correlation between image patches. In the older version of the system when

Figure 7.15: Estimated camera orientations for a 6 meter long path with Kalman filter on.
unwarped images were used, the system had a speed of 2.8Hz. However, when the system was changed to improve the accuracy by means of eliminating the unwarping process, and it was moved to outdoor environments the running time of the system increased to 8.4 seconds per frame. This can be explained by several sources.

- Perhaps the most important reason for this problem is the considerably larger number of features in outdoors comparing to indoor environments. If \( n \) represents the number of corners, the stereo matching and construction stages have complexities of \( O(n^2) \). Tracking \( n \) 3D features from one frame to another frame has also a complexity of \( O(n^2) \). This is due to the fact that both tracking and stereo processes are heavily involved

Figure 7.16: Estimated camera trajectories relative to a starting point for a 6 meter long path without Kalman filter.
7.5 Computation Time

in the use of the normalized cross-correlation function for the purpose of measuring similarities. It must be pointed out that the number of features in Experiment 7.2.3 was 4 times higher than of the number of features in Experiments 7.2.1 and 7.2.2. An increase in the number of features by a factor of 4 increases can potentially cause the running time of the tracking and stereo tasks alone by a minimum factor of 16. Therefore, the running time of the tracking task is expected to increase from 0.21 second to 3.36 seconds and the stereo procedure from 0.057 second to 0.912 seconds. Since tracking and stereo routines include several added functions that are required for the partial image unwarping process, in practice the running times of these two procedures are increased to 5.1 and 1.35 seconds (as shown in Table 7.8). Therefore,
a dramatic increase in the running time of the system seems to be expected when moving to outdoor environments with higher number of features. As shown in Table 7.8 about 80% of the time in Experiment 7.2.3 is spent on the tracking and the similarity evaluation of matched features.

- Another reason for the slower performance is the distance of features from camera. For indoor environments, Experiments 7.2.1 and 7.2.2, the objects were located at distances ranging from 0.5 to 8 meters. This range is increased to 0.1 to 20 meters, with 60% of the features in distances of 0.125 to 0.4 meter from the camera in outdoor environments in Experiments 7.2.3 and 7.2.4. For a same amount of motion, closer features undergo larger positional changes and therefore for outdoor cases the search window had to be increased by 4 times (141 x 141 versus 71 x 71) [84]. Such larger search boundaries cause a higher number of match candidates and consequently increase the running time.

- Finally, when using warped images, features are detected and stored based on their position in the warped images. However, they must be sorted based on their unwarped image locations right before the stereo algorithm. This sorting scheme is a necessary process as in stereo process, the search is performed along epipolar lines. Also when transferring world features from world coordinates to project them onto the previous image plane, they must, once again, be sorted based on their warped image locations. This is also a necessary process since the tracking is performed based on the features’ warped image locations and contents. These sorting processes, although not much, increase the running time as well.

Figure 7.18 shows implemented software modules. In this figure, the first column of modules (1.1, 1.2, 1.3, and 1.4) are performed only for the first frame. The rest of the modules are repeatedly performed for later frames.

Table 7.8 is created to represent the cost associated with each module for the typical outdoor scene in Experiment 7.2.3. The running time of the system is highly dependent
7.5 Computation Time

on the scene, number of features, and imaging conditions. In this table at the end of each module name there is a number which shows the corresponding block number in Figure 7.18. The displayed times are the result of 100 runs for each module.
Figure 7.18: System's modular flowchart for the time analysis.
Table 7.8: CPU time spent on various functions of trajectory tracking system.

<table>
<thead>
<tr>
<th>Function name</th>
<th>Spent time (ms/call)</th>
<th>No. of calls</th>
<th>Total time (ms)</th>
<th>% of total time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read input images (1)</td>
<td>0.95353</td>
<td>3</td>
<td>2.85606</td>
<td>0.0339</td>
</tr>
<tr>
<td>Feature detection (2)</td>
<td>31.13</td>
<td>3</td>
<td>93.39</td>
<td>1.1094</td>
</tr>
<tr>
<td>Sort features (3)</td>
<td>3.16</td>
<td>1</td>
<td>3.16</td>
<td>0.0375</td>
</tr>
<tr>
<td>Stereo algorithm (4)</td>
<td>1350.25</td>
<td>1</td>
<td>1350.25</td>
<td>16.0403</td>
</tr>
<tr>
<td>Transfer world features (5)</td>
<td>0.8336</td>
<td>1</td>
<td>0.8336</td>
<td>0.0099</td>
</tr>
<tr>
<td>Project features onto the image (6)</td>
<td>0.4202</td>
<td>1</td>
<td>0.4202</td>
<td>0.0050</td>
</tr>
<tr>
<td>Compute mean and square values (7)</td>
<td>218.5476</td>
<td>2</td>
<td>437.0952</td>
<td>5.1925</td>
</tr>
<tr>
<td>Track features (8)</td>
<td>5109.3</td>
<td>1</td>
<td>5109.3</td>
<td>60.6959</td>
</tr>
<tr>
<td>Find the motion (9)</td>
<td>8.132</td>
<td>1</td>
<td>8.132</td>
<td>0.1198</td>
</tr>
<tr>
<td>Transform world features using the motion (10)</td>
<td>0.82037</td>
<td>1</td>
<td>0.82037</td>
<td>0.0966</td>
</tr>
<tr>
<td>Sort features using their unwarped locations (11)</td>
<td>0.10125</td>
<td>1</td>
<td>0.10125</td>
<td>0.0097</td>
</tr>
<tr>
<td>Project features after transformation on the image (12)</td>
<td>0.78925</td>
<td>1</td>
<td>0.78925</td>
<td>0.0012</td>
</tr>
<tr>
<td>Find correspondences (small neighborhood) (13)</td>
<td>5.1716</td>
<td>1</td>
<td>5.1716</td>
<td>0.0614</td>
</tr>
<tr>
<td>Estimate the motion (14)</td>
<td>6.16</td>
<td>1</td>
<td>6.16</td>
<td>0.0732</td>
</tr>
<tr>
<td>Update robot’s current location (15)</td>
<td>0.3411</td>
<td>1</td>
<td>0.3411</td>
<td>0.0041</td>
</tr>
<tr>
<td>Transfer from current to the global (16)</td>
<td>5.51</td>
<td>1</td>
<td>5.51</td>
<td>0.0655</td>
</tr>
<tr>
<td>Combine features (17)</td>
<td>91.43</td>
<td>1</td>
<td>91.43</td>
<td>1.0861</td>
</tr>
<tr>
<td>Update database and retire if is possible (18)</td>
<td>0.1</td>
<td>1</td>
<td>0.1</td>
<td>0.0012</td>
</tr>
<tr>
<td>Miscellaneous</td>
<td>1302</td>
<td>1</td>
<td>1302</td>
<td>15.4671</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td></td>
<td>8417.9</td>
<td>100</td>
</tr>
</tbody>
</table>
7.5 Computation Time

From this table, it is concluded that the most time consuming modules are the stereo and tracking routines. The main core of these two modules is the correlation (NSSD) function. The tracking process, block (8), requires over 60% of the processing time. In this part of the system, since there is no knowledge about the motion, a fairly large search neighborhood was required. Since the number of match candidates increases by the search window dimensions, it is interesting to see the effect of the search boundary on the running time. Figure 7.19 represents the running time for our system based on the search dimension in block 8 Figure 7.18.

![Graph showing the running time versus search window dimensions for the tracking module.](image)

Figure 7.19: Comparison of the running time versus search window dimensions for the tracking module.

As expected when the search size decreases a faster performance achieved. Another time consuming process, block 7, calculates the average and square intensity values of the compared candidate patches used in the correlation process. Obviously, by using a faster processor, the running time of the system can be lowered.

To conclude this discussion, it is important to see the trade off between the system processing rate with the motion rate and search window dimensions in the tracking process. A smaller motion, between two consecutive frames, results in smaller displacements of image
features in two corresponding image frames. In such condition, corresponding features can be found by searching over smaller regions. As shown in Figure 7.19, smaller windows speed up the system processing rate. Therefore, through a slower moving robot a faster performance can be achieved.

7.6 Chapter Summary

In this chapter the implementation results are presented. The improvement of the system accuracy by employing a combined use of images before and after lens distortion removal was shown through several indoor and outdoor experiments. The comparison of the trinocular and binocular stereo is also presented. It is concluded that the use of binocular system can still provide a good accuracy. The effectiveness of the Kalman filter over a long range of motion is presented. The cost of different modules is also discussed.
Chapter 8

Conclusions and Future Work

This chapter summarizes the main contributions of this thesis. It also discusses the scope for future work and extensions to this thesis for further improvement.

8.1 Major Contributions

This thesis has presented important contributions for the successful development of a general purpose 3D trajectory tracking problem for unknown indoor and outdoor environments. It requires no modifications to be made to the scene and it is not dependent on any prior information about the scene. The system operates using a small fraction of the scene features that can potentially carry maximum information about the camera motion. There are no constraints imposed on the camera motion or the scene features other than the rigidity of most of the features. The system reduces the time that is generally spent for the lens distortion correction process by removing it only for a number of selected feature points (at most only 2% of the entire image pixels). This process minimizes the positional projection error of image features, corresponding to identical scene features, at different time instances. This improved positional accuracy allows one to obtain the camera’s position and motion parameters with a higher accuracy. Moreover the system implements an active vision-based
8.1 Major Contributions

3D trajectory tracking system that is affordable and can be easily fitted to satisfy different requirements. The cumulative positional error of the system is less than 1% in both position and angle.

Four main objectives were sought in this thesis.

- **3D estimation**: This requirement is fully satisfied and at each estimation six motion parameters are estimated.

- **Scene independence**: This condition for indoors and outdoors with the exception of the following pathological cases are achieved:
  
  I. The environment must be textured and the details of the textured surfaces must be large enough to be identified individually in scene images.
  
  II. No significant surface specularity such as objects with polished mirror-like surfaces.
  
  III. Periodic scene features (picket fence effect).
  
  IV. Significant effects of occlusions.

- **High accuracy**: The system performs with an accuracy of 1% in both rotational and translational movements. Although we do not have the % error for a system with a similar configuration as ours, our system appears to be more accurate than some of the prior similar works (e.g. Harris [103], who we estimated, achieved a 3.2% error).

- **Real-time performance**: The real-time performance of the system is not achieved as there is a trade off between the performance accuracy and speed. Improvement of the accuracy was given a higher priority in this work. Suggestions for future improvement of the speed are represented in Section 8.2.

The major contributions of this thesis can be summarized as described below:
8.1 Major Contributions

8.1.1 Fast feature detection

A novel fast feature detection algorithm named the Binary Corner Detector (BCD) is developed. The performance quality of this algorithm is compared with the most commonly used method (Harris corner detector) as well as the method inspiring it (SUSAN corner detector). The result of this comparison indicates that it performs considerably faster than both methods while maintaining a high stability under different environmental and imaging conditions. The faster performance is gained by substituting arithmetic operations with logical ones. Since the main assumption for the whole system has been that temporal changes between consecutive frames are not large, a faster feature detector leads to less temporal changes between the consecutive frames and therefore resulting in a higher accuracy in the overall system.

8.1.2 3D feature reconstruction with minimal positional error

Reconstruction of world 3D features using the pinhole camera model and stereo algorithm is an essential part of the trajectory tracking approach that was undertaken in this thesis. Due to imperfect lenses, the acquired images include some distortions that are generally corrected through the calibration process. Not only is the image calibration at each frame for three images a time consuming process but it could add positional shifts (error) to image pixels. This process degrades 3D reconstruction results and increases the cumulative error in the overall trajectory tracking process. To remove this undesired effect, a calibration map for each one of the cameras is constructed that defines the relationship between the integer position of the uncalibrated pixels with the corresponding floating point location on the calibrated image. This process allows one to work with sharper images. It also provides a more accurate disparity estimation up to several pixels as well as a faster processing time by eliminating the calibration process for three individual images.
8.1.3 Multi-stage feature tracking

Tracking identical 3D world features at different times is a key issue for accurate 3D trajectory estimation. Correct identification of such features however, depends on several factors such as search boundaries, similarity measurement window size, and a robot's motion range. Expanding search boundaries and the window size for similarity evaluation can improve the accuracy by adding more correct matches. They can however slow down the performance, leading to a larger motion between two consecutive frames. A larger motion introduces more inaccuracy into the system. In order to improve the accuracy, a two-stage tracking scheme is introduced in which the match correspondences are first found using a large search window and a smaller similarity measurement window. Through this set of correspondences a rough estimation of the motion is obtained. These motion parameters are used in the second stage to find and track identical features with higher accuracy. The process increases the tracking accuracy by up to 30%.

8.2 Future Research

There are several aspects of the trajectory tracking system that can be improved or extended to make the system more useful and reliable. Some example are outlined below:

8.2.1 Combined wide and narrow views

The type of the lenses in the stereo set has an important role on the overall accuracy as well as the real-time performance of the system. A wider field of view can improve the speed since its coarser image resolution leads to a smaller search window which can recover motions with larger ranges. A narrower field of view on the other hand can provide more precise location information. This however will decrease the speed as wider search windows are required. As presented in Section 7.3 the orthogonal arrangement of the cameras with equal baselines does not affect the accuracy of the system significantly. For this reason, it
seems more reasonable to substitute the trinocular stereo set with two individual binocular stereo systems with two different lens types. The motion can be estimated first through the wider set with the lower resolution with a smaller search window depending on the width of the view. This result can be used as an initial estimate of the motion and can be refined with a much higher accuracy through the narrower lens set in a timely manner.

### 8.2.2 Orthogonal views

One of the general existing problems for our system is the increase of the translational error with rotational movements. This mainly originates from the fact that points at different locations in the environment show different behaviors under varying motions. This can be explained in Figure 8.1. For a rotational motion of $\alpha$ the behavior of point A is the same as moving toward left with amount of $\Delta x$. Adding a second set of stereo cameras with an image plane perpendicular to the first pair can prevent this. With a pure translation toward left, point A still remains fixed in the image of the second camera set, while with a rotation $\alpha$ toward left it moves toward left on the second image plane. As suggested in Section 8.2.1, the use of two sets of individual stereo systems with different lenses can improve the accuracy. Arranging the two sets in an orthogonal way, as explained, can prevent the coupling effect between the translation and the rotation.

### 8.2.3 Hierarchical structure

The most time consuming process of the system was the establishment of feature correspondences for consecutive frames. This deficiency becomes more costly as moving from indoor scenes to outdoor scenes increases the number of features dramatically. One possible solution to this problem is the establishment of a hierarchical approach in feature extraction and tracking as well as in motion estimation. For this purpose, input images are resampled to create an image resolution pyramid. At each time, features are detected on the coarser level and using them a rough motion estimation is obtained. The estimated motion along
with a finer image from a lower level of the pyramid will be employed to refine the motion parameters. This method reduces the computation by the 4'th power of the reduction in the resolution. For instance reduction by a factor 2 reduces the image sizes to 1/4 of the original sizes. Since there is a dual role in treating the image features this process will speed up the computation by a factor of 16.

8.2.4 Sub-image processing

Another possible way for improving the processing rate is to select and process only selected patches of each image instead of the entire image. In order to obtain the most possible information from these image patches, they should be chosen from specific areas. An example of such areas are shown in Figure 8.2.

Patches from areas close to image edges help to separate the rotation from the sideway translation, while a patch from the center provides features with higher 3D positional accuracies and therefore results in more accurate motion estimation. It is also possible to choose these image patches based on the number of their features. However these patches must be a carefully distributed to avoid false information provided by the unexpected motion of an unknown object in a certain part of the scene.
8.2 Future Research

8.2.5 Cumulative error reduction

An inherent characteristic of the system is the estimation error which is accumulated over time and distance. Although estimated transformation between each two frames is reliable, over a long period of time the system must be able to correct this error and reset itself. Integrating other sensors such as odometers can help to find the motion between frames more accurately and therefore, the system would perform more accurate estimation while there is a fast motion. Another possibility could be the creation of a map of the environment and retaining specific frames (for example every 10'th frame) and their corresponding location in a database. As the system moves around if it becomes close to a place that has been traversed before, the saved frames and their location can be used to decrease the cumulative error.
8.2.6 Specialized hardware

Employment of specific hardware (e.g. FPGA's) that allows the system to perform bitwise parallel operations on the generated binary images can improve the speed of the system. Bitwise operations can increase the speed of the feature detection sub-system. They can also be used to substitute the expensive NSSD on the intensity images with a binary cross-correlator as described by [109].
Bibliography


Appendix A

Location Determination Problem by Triangulation

If a set of known beacons are observed from an unknown location, then the position of the observer can be computed from the relative directions of the observed beacons. This is known as the triangulation problem. In other word, for visual localization problem the camera’s optical center can be estimated by examining the relative direction of a set of seen landmarks. The 2D and the 3D location determination problems are defined differently and are therefore examined separately in here.

A.1 Triangulation Problem

In $\mathbb{R}^2$ the observed angle $\alpha_{1,2}$ between two landmarks $p_1$ and $p_2$, with unit direction vectors $v_1$ and $v_2$, respectively is given by

$$\sin \alpha_{1,2} = |v_1 \times v_2|$$  \hspace{1cm} (A.1)

The law of Cosine states that the $\alpha_{1,2}$ defines a circle passing through $p_1$, $p_2$ and the optical center $o$ of the image, as shown in Figure A.1.
The 3D problem has the same analogy as the 2D problem, except that now the observed angle \( \alpha_{1,2} \) between landmarks \( p_1 \) and \( p_2 \) defines a surface which \( o \) must lie on. In principle,
The 3D Location Determination Problem

The intersection of three surfaces, derived from the three different pairs of landmarks, will uniquely determine the location of \( o \). However, this approach requires solving a complex 4th order polynomial. A simple approach was described by Fischler and Bolles [16], called the location determination problem. This method is non-linear and examines three landmarks, with fourth used to resolve ambiguity. Faugeras et al. [60] also described a linear solution to the location determination problem that used six landmarks.

![Diagram of three landmarks and projectors](image)

Figure A.2: The locations of three landmarks along three image projectors must match the distances between the same points.

If three landmarks \( p_1, p_2 \) and \( p_3 \) are projected onto an image \( o \), with direction vectors \( v_1, v_2 \) and \( v_3 \), then the position of each feature point along its respective projector must match the distance between the recovered point in the solution, as shown in Figure A.2 that:

\[
\begin{align*}
  p_1 &= o + \Omega_1 v_1 \\
  p_2 &= o + \Omega_2 v_2 \\
  p_3 &= o + \Omega_3 v_3
\end{align*}
\]  

(A.3)

with the constraint that

\[
\begin{align*}
  |p_1 - p_2| &= d_{1,2} \\
  |p_1 - p_3| &= d_{1,3} \\
  |p_2 - p_3| &= d_{2,3}
\end{align*}
\]  

(A.4)
Here $d$ is the distance between the two landmarks in the solution. Equation A.4 is quadratic with three unknowns, $\Omega_1$, $\Omega_2$ and $\Omega_3$, and has up to eight solutions [16]. However, for every positive solution there is a geometrically isomorphic negative solution behind the camera, so there are at most four actual solutions to consider. To resolve the ambiguity, the procedure is repeated substituting a fourth point $p_4$ in place of $p_3$ to identify the unique solution.

Equation A.3 gives the relative distance $\Omega_i$ of each feature point $p_i$ from the focal point $o$. The actual position of $o$ with respect to the feature points is found by solving:

\begin{align*}
|o - p_1| &= \Omega_1 \\
|o - p_2| &= \Omega_2 \\
|o - p_3| &= \Omega_3
\end{align*}

which has two solutions, one on each side of the triangle with vertices $p_1$, $p_2$ and $p_3$. The correct position of $o$ is determined by examining the triple scalar product $(v_1 \times v_2).v_3$. 