Die approbierte Originalversion dieser Dissertation ist an der Hauptbibliothek der Technischen Universität Wien aufgestellt (http://www.ub.tuwien.ac.at).

The approved original version of this thesis is available at the main library of the Vienna University of Technology (http://www.ub.tuwien.ac.at/englweb/).

DISSERTATION

Real-Time Mobile Robot Self-localization

A Stereo Vision Based Approach

Submitted at the Faculty of Electrical Engineering and Information Technology, Vienna University of Technology in partial fulfillment of the requirements for the degree of Doctor of Technical Sciences

under supervision of

Prof. Dr. Dietmar Dietrich
Institute of Computer Technology
Vienna University of Technology
1040 Vienna

and

Prof. Dr. Robert Sablatnig
Pattern Recognition and Image Processing Group
Institute of Computer Aided Automation
Vienna University of Technology
1040 Vienna

by

Abdul Bais
Matr. Nr. 0357233
Elisenstraße 1
1230 Vienna

June 2007

Kurzfassung

Das Ziel dieser Arbeit ist die Echtzeit-Selbstlokalisation von kleinen, autonomen, mobilen Robotern in bekannten, aber hochdynamischen Umgebungen basierend auf Bildverarbeitung. Dabei wird eine erste Schätzung der Position kontinuierlich aktualisiert. Für den Algorithmus zur Lokalisierung ist es nicht notwendig, zusätzliche künstliche Markierungen oder spezielle Strukturen in der Umgebung anzubringen. Weiters ist es nicht notwendig, dass sich Orientierungspunkte direkt am Boden oder in Bodennähe befinden. Der Algorithmus erlaubt dem Roboter, seine Ausgangsposition zu bestimmen und seine aktuelle Position während der Bewegung ständig zu aktualisieren.

Eine Schätzung der Position in einer gleichbleibenden Umgebung erfolgt normalerweise von der Roboterposition aus über Entfernungs-oder Winkelmessungen zwischen bekannten Orientierungspunkten. Eine andere Möglichkeit ist der Vergleich einer lokal aus Sensordaten erstellten Landkarte mit einer vorab gespeicherten, globalen Karte der Umgebung. Im Gegensatz zu anderen Lokalisationsalgorithmen werden in dieser Arbeit aus der Stereobildverarbeitung gewonnene Tiefeninformationen verwendet. Erkannte Orientierungspunkte werden mit Trilaterationsverfahren in eine unvollständige 3D-Karte der unmittelbaren Umgebung transformiert, die dann mit dem globalen Modell verglichen wird. Für die Modellerstellung werden Längeninformationen verwendet, da man bei dieser Methode weniger Orientierungspunkte als bei der winkelbasierten benötigt. Das Stereobildverarbeitungssystem ist an einen drehbaren Kopf des Roboters montiert.

In dieser Arbeit wird Gradient Based Hough Transform (GBHT) als Basis für die Merkmalsextraktion verwendet. GBHT wurde gewählt, da es die stärkste Gruppierung von kolinearen Pixel mit ähnlicher Kantenausrichtung erlaubt. Eine kontinuierliche, vollständige Bestimmung der eigenen Position wäre sehr rechenintensiv und nur bei genügend sichtbaren Orientierungspunkten möglich. Daher wird die einmal bestimmte Position durch die robotereigenen Sensorwerte aktualisiert. Diese Methode ist viel schneller und ausreichend genau, da sich die Summenfehler innerhalb kurzer Zeitintervalle unterdrücken lassen. Zum Vereinigen der verschiedenen Sensorwerte wird der Extended-Kalman-Filter verwendet. Eine aus den Sensordaten gewonnene grobe Schätzung der Position wird als Eingangsgröße bei der Merkmalsextraktion und dem Vergleich mit ein Weltkarte verwendet und erhöht die Genauigkeit. Bedeutende Leistungsverbesserungen sind mit einer neuen hybriden Methode erzielt worden, die die globale Positionsschätzung mit der Spurhaltung kombiniert. Simulationsergebnisse der Kartenerstellung, Merkmalsextraktion, Berechnung der Tiefeninformation, kombinierten Informationsverarbeitung mehrerer Sensorwerte und ein Test der Programmstruktur sind der Arbeit beigefügt. Die Ergebnisse dieser Arbeit können verwendet werden, um die Anzahl der notwendigen Orientierungspunkte für einen Roboter mit Bildverarbeitung zu minimieren.

Abstract

The main focus of this thesis is vision based real time self-localization of tiny autonomous mobile robots in a known but highly dynamic environment. The problem covers tracking the position with an initial estimate to global self-localization. The localization algorithm is not dependent on the presence of artificial landmarks or special structures in the environment nor does it require that features should lie on or close to the ground plane. The algorithm enables the robot to find its initial position and to verify its location during every movement. Global position estimation in unmodified environments normally involves measuring distance to or angles between distinct features (natural landmarks) from the robot position or matching a local map constructed from sensor readings to a global map of the environment. On contrary to other localization algorithms stereo vision based depth computation is used for self-localization. A localization framework is in progress that uses trilateration based techniques whenever distinct landmark features are extracted. The trilateration based method is complemented by a sparse 3D map of the local environment constructed based on sensor data and matching it with the environment model. The stereo vision system is mounted on a pivoted head as an aid in feature exploration. Distance measurements are used as they require fewer landmarks compared to methods using angle measurements.
Visual features are extracted using Gradient Based Hough Transform (GBHT), which provides the strongest groupings of collinear pixels having roughly the same edge orientation. Global self-localization is computationally slow and sometimes impossible if enough features are not available. Therefore, once the robot position is computed it is tracked with local sensors. This is fast and reasonably accurate as the accumulating error is suppressed after short intervals. Extended Kalman filter is used to fuse information from multiple heterogeneous sensors. Keeping a rough estimate of the robot position helps in features extraction and matching with the global map. Significant performance improvements have been achieved with a new hybrid method that combines the global position estimation with tracking. Simulation results for the robot environment modeling, feature extraction, depth computation, information fusion, and initial test of the framework have been reported. As such marked minimization of landmarks for vision based self-localization of robots has been achieved.

Preface

I would like to convey my gratitude towards my supervisors Prof. Dietmar Dietrich and Prof. Robert Sablatnig for their extensive cooperation and support from the beginning to the end of my research period.
I am thankful to Prof. Jason Gu (Robotics Research Laboratory, Dalhousie University, Canada) for his contribution in this work during my one year stay at Dalhousie University. I also thank members of his group Pifu Zhang, Weimin Shen, Luis Bustamante, Love Karla, Karthick Srinivasan, Kevin Walker, for their discussions, support and motivating company.
I extend special thanks to Stefan Mahlknecht and Gregor Novak for their constant support, taking notice of the smallest details and making extra effort for solution of even minor problems arising during the course of my research. I thank Tobias Deutsch who provided data for the comparison study. I thank all members of Tinyphoon project including Stefan Krywult, Roland Oberhammer, Daniel Steinmair, Daniel Weber, Martin Hausner, Markus Bader, and members of the Institute of Computer Technology, for their good company which made me feel comfortable while working with them. My thanks go to Dietmar Bruckner and Christoph Bacher for their guidance during the starting months in Austria.
I would like to thank all others who are not mentioned here but were involved or associated with this work. My special thanks go to Aaliya Rehman Khan for extensive reviewing and correcting my work. Her constant moral support kept me motivated throughout my research period. I gratefully acknowledge the role of Higher Education Commission of Pakistan towards the completion of my PhD, which provided the monetary support during the whole research period.
And I feel immensely grateful towards my parents, who always stood behind me, motivating and pushing me to strive for more, especially my mother whose unwavering confidence and prayers kept me on the right path of hard work and dedication to achieve my aims and to reach thus far. I humbly dedicate this work to my mother.
The thesis is organized in the following way:
Chapter 1 introduces the topic, gives motivation and the problem statement. It also states the requirements for self-localization and presents the proposed methodology. Related work is given in Chapter 2 focusing mainly on indoor robots and the localization problem in broad sense. Details of vision based feature extraction and range computation are presented in Chapter 3. Related papers to this chapter are [NBM04, BSN05]. Features extracted in Chapter 3 are used for global self-localization in Chapter 4. Two methods are discussed: the first one requires single distinct landmark and absolute orientation of the robot, whereas, the second one requires two distinct landmarks with ordering constraint. This chapter also discusses propagation of uncertainty in landmark based position estimation. Related publications to Chapter 4 are [BS06, BSG06, BSGM06, BSKN07].
The global self-localization is impossible if enough features are not available. Therefore, it is required to track the robot position once it is estimated. During tracking, the uncertainty grows, which is suppressed using observation of the robot environment. This is the subject of the Chapter 5. The last part of Chapter 5 presents a comparison of position estimation using extended Kalman filter and particle filter. The ideas explored in Chapter 5 are also discussed in [BSGK07, BDN07]. Performance evaluation of all algorithms is given in Chapter 6, while Chapter 7 concludes the work and gives directions for future research in this particular field.

Contents

1 Introduction **1**
1.1 Motivation . . . 1
1.2 Problem Statement . . . 3
1.2.1 The Robot Environment . . . 3
1.2.2 Constraints About the Robot . . . 3
1.2.3 Requirements for Self-localization . . . 4
1.3 Proposed Methodology . . . 4

2 Related Work **8**
2.1 Relative Position Estimation . . . 8
2.1.1 Odometry . . . 8
2.1.2 Inertial Navigation . . . 10
2.2 Landmark-Based Methods . . . 11
2.2.1 Natural Landmarks for Localization . . . 11
2.2.2 Artificial Landmarks for Localization . . . 12
2.2.3 Active Beacons . . . 13
2.3 Map-Based Positioning . . . 13
2.3.1 Map Representation . . . 14
2.3.2 Map Building . . . 16
2.3.3 Map Matching . . . 16
2.4 Probabilistic Methods . . . 17
2.4.1 Markov Localization . . . 20
2.4.2 Monte-Carlo Localization . . . 21
2.4.3 Kalman Filter Based Techniques . . . 24
2.5 Hardware for Robot Self-Localization . . . 25
2.6 Miscellaneous Methods of Localization . . . 27
2.6.1 Collaborative Multi-Robot Localization . . . 27
2.6.2 Global Vision . . . 28
2.6.3 Mosaic-Based Localization . . . 29
2.6.4 Position Estimation Based on Camera Calibration Techniques . . . 29
2.7 Discussion . . . 29

3 Vision Based Feature Extraction **31**
3.1 Related Work . . . 31
3.2 The Stereo Vision System . . . 33
3.3 Detecting Line Features . . . 36
3.3.1 Line Detection with Hough Transform . . . 36
3.3.2 Peak Detection . . . 37

3.3.3 Peak Verification and Completing Line Segment Description 39
3.4 Detecting Field Markings 42
3.5 Detecting Corners and Junctions 43
3.6 Detecting Color Transitions 46
3.7 Discussion 46

4 Feature Based Global Self-localization 47
4.1 Related Work 47
4.2 Landmark Based Position Estimation 49
4.2.1 Position Estimation using Range Measurements 49
4.2.2 Position Estimation using Angle Measurements 50
4.2.3 Adding Absolute Orientation of the Robot 51
4.2.4 Using a Virtual Landmark 52
4.2.5 Position Uncertainty Due to Erroneous Measurements 54
4.2.6 Error in Landmark Identification and Matching 54
4.2.7 Sensitivity to Landmark Configuration 54
4.2.8 Position Estimation with More Than Three Landmarks 56
4.3 Position Estimation Using Vision Based Range 58
4.3.1 Calculating Position Using Two Landmarks 58
4.3.2 Position Estimation Using Single Landmark 59
4.4 Uncertainty Analysis 61
4.4.1 Robot Location Uncertainty Using Single Landmark 61
4.4.2 Location Uncertainty Using Two Landmarks 63
4.5 Discussion 65

5 Position Tracking and Information Fusion 66
5.1 Extended Kalman Filter 67
5.1.1 State Transition Model 67
5.1.2 Observation Model 68
5.1.3 Kidnapped Robot Problem 72
5.1.4 Representing Uniform Position Belief 75
5.2 Robot Observation 76
5.3 Particle Filter Based Position Estimation 78

6 Results and Evaluation 80
6.1 Feature Extraction Results 80
6.2 Statistical Results for Vision Based Range Estimation 84
6.3 Feature Based Global Localization Using Two Landmarks 85
6.3.1 Motion without Rotation 86
6.3.2 Single Point Rotation 89
6.3.3 Motion and Rotation 91
6.3.4 Linear Motion with Restricted Rotation 93
6.3.5 Linear Motion with Rotation 94
6.4 Location Estimation Using One Landmark 96
6.4.1 Motion without Rotation 98
6.4.2 Single Point Rotation 100
6.4.3 Motion and Rotation 101
6.4.4 Linear Motion with Restricted Rotation 104

6.4.5 Linear Motion with Rotation 108
6.5 Position Tracking and Information Fusion 108

7 Discussion and Future Work 125
7.1 Discussion 125
7.2 Future Work 126

A Notations and Coordinate Systems 128
A.1 Notations 128
A.2 Coordinate Systems 129

B Odometry Model 131
B.1 Geometric Construction of a Differential Drive Robot 131
B.2 Control Vector Uncertainty 133
B.3 Compounding of Transformation 134
B.4 Special Cases 136
B.4.1 Motion Along a Straight Line 136
B.4.2 Rotation Around Center of Mass 137

Abbreviations

CA	Condensation Algorithm
CCD	Charge Coupled Device
CRHT	Connective Randomized Hough Transform
DGPS	Differential Global Positioning System
ECRHT	Extended Connective Randomized Hough Transform
EKF	Extended Kalman Filter
GBHT	Gradient Based Hough Transform
GHT	Generalized Hough Transform
GPS	Global Positioning System
HT	Hough Transform
KF	Kalman Filter
LRF	Laser Range Finder
MCL	Monte Carlo Localization
ML	Markov Localization
OHT	Optical Hough Transform
PF	Particle Filter
PHT	Probabilistic Hough Transform
RHT	Randomized Hough Transform
RWHT	Range Weighted Hough Transform
SIR	Sampling Importance Resampling
SIS	Sequential Importance Sampling
SLAM	Simultaneous Localization and Map Building
UKF	Unscented Kalman Filter

Chapter 1

Introduction

The primary task of this work is to address the self-localization problem of an autonomous mobile robot.

1.1 Motivation

To illustrate the importance of localization in robot navigation, let us take the example of a man somewhere in a large office building with a labyrinth of corridors and passages leading to various departments[1]. He knows the address to the room which he wishes to visit. In order to reach that office, the logical order of sequences would be that he must in the first instance, determine where he is himself located at that moment. Once he knows his own location, then he can look for signs, like an information notice board, indicating which department lies in which direction. He can then follow that lead and turn corners and passages until he reaches the right department. He must then look for more landmarks like the number of the rooms on his left and right. He can judge then whether those numbers are in ascending order or in descending order. From their order he can then draw inference whether he has to move up the corridor or down it. In this manner the man must continue to move on until he reaches the desired office.

Thus, it is clear from the above example that the first step the man has to do in order to reach his final destination is to find his own position. To find his position he can look around and compare his observation with the information represented in the building map. An example of observation could be a room number or a name plate. A single observation may not be enough since many locations in the environment may look a like. The problem of ambiguities is an inherent part of the problem of localization. In order to distinguish between the different possible locations, one has to keep these locations in mind and collect additional information within the same locality. Once the person knows where he is, he must continue to keep track of his position as he moves on. The task of tracking can be efficiently achieved by continuously comparing the observations with the places expected in the building map. However, as soon as the person loses his position, he has to re-localize himself, which often requires similar steps to those performed on arrival in the office building.

[1]A similar example is given in [FBTC98]

From the above given example it is obvious that estimation of one's position in the given environment is essential for using a map. Autonomous indoor mobile robots face the same problems as the person in the above example when performing their tasks. Hence, effective navigation is essential for successful mobile robot applications and in order to carry out navigation plans, the prerequisite is mobile robot's localization [FBTC98]. Therefore, localization is one of the fundamental problems in mobile robotics [Cox91]. Similarly, in an application where multiple robots are working on a common global task, knowledge of the position of individual robots turns out to be a very basic requirement for its execution. A successful global strategy for navigating can only be devised if the robot knows its own position and those of other robots with whom it has to coordinate. Our robots have been designed to play soccer and during the game they need to keep track of the ball, avoid collisions with other players (also robots) and to successfully make goal hits. Tasks such as playing the ball and hitting the goal assume that all objects are in the same coordinate system.

For an autonomous robot it is essential to gather as much information about its environment as possible to act intelligently. Therefore, they need to determine depth information of objects they want to interact with. This can be done with a wide range of sensors. A sonar sensor can also calculate depth but the resulting depth-information is noisy due to interference of the outgoing signal with the reflected signal. The most accurate sensor to compute a depth map of the environment is a Laser Range Finder (LRF). However, this sensor is too large to encapsulate in our robot and also takes a long time to generate a complete three dimensional map of the environment. A stereo vision system can also be used to obtain a three dimensional map and can be built small enough to fit the dimensions needed. A detailed discussion on these sensors is given in the next chapter.

Stereo vision tasks try to imitate the capability to see objects, to estimate their three-dimensional position in the world, and to identify the objects for grasping them. The tasks of perceiving depth, identifying objects and grasping are performed by humans in real time for everyday purposes, without being aware of this complex process. One example of a complex vision task involved in solving a particular problem is the action and reaction taken if humans are playing soccer. Here scene understanding is essential, one has to know position of the ball, of other players and of the goal in order to perform the best. This is a typical application where range is more important than recognition, all objects on the playground are known and limited. The two main tasks are localization (both self-localization and localization of other moving objects) and continuous range estimation for all moving objects. Such kind of applications are an ideal testbed for computer vision tasks involved with range estimation rather than object recognition.

If a single camera observes a real world object, the three-dimensional information is projected into a two-dimensional image plane. In this case the information of depth is lost during the projection process. In order to reconstruct the three-dimensional information of a scene, computer vision tries to imitate the human visual system to use two different views of the same scene. This technique is known under the term *stereo vision* or *binocular vision* [BF82]. In a static environment it is possible to obtain range information from one camera if its position is changing and the geometric relations between two or more images are known. On a mobile robot without a static environment a stereo vision system with two or more cameras has to be used to obtain stereo information. If the images are taken simultaneously, a static environment is guaranteed [Ent05]. The goal of a stereo analysis method is the calculation of depth information due to geometric relations. Therefore, a stereo vision system is used on moving platforms playing soccer.

1.2 Problem Statement

Localization entails positioning the robot relative to its environment during navigational sessions and can be defined as computing and updating three parameters (x, y, θ), which, respectively, define the position and orientation of the robot within the environmental frame [AHC05]. This thesis addresses this problem keeping in view the minimum requirements and constraints listed in this section.

1.2.1 The Robot Environment

The robot environment is supposed to be known but highly dynamic. It could be described with visual landmarks i.e. lines, corners, junctions, line intersections and color transitions [BSN05]. A simple test environment model is shown in Fig. 1.1(a). Objects in the robot environment are few and known. The height of the surrounding walls is less than the robot's height and can be removed in future. Objects in the form of isosceles triangles are placed in the four corners formed by the surrounding wall so that the ball is not trapped in the corner. This makes these corners blunt. A sample image by a hypothetical onboard camera is shown in Fig. 1.1(b). Field markings and the blue goal can be seen in the image. Further details about dimensions of the field and model representation in robot memory can be found in [Deu07].

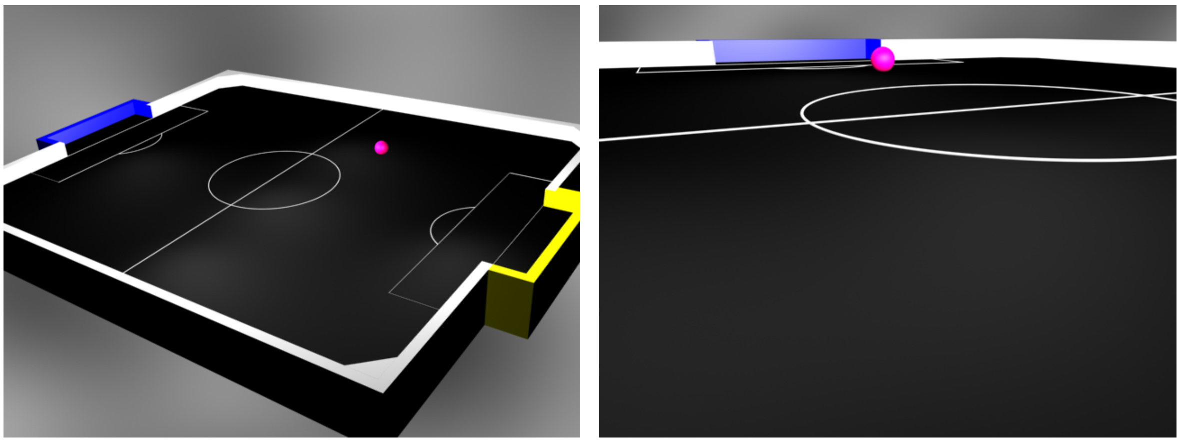

(a) (b)

Figure 1.1: The robot environment

1.2.2 Constraints About the Robot

In our application the robots has to be within a pre-defined size limit. Since they need to work autonomously, all processing is done by the on-board processors. The robot is battery driven and thus the power consumption should be less so that it can work for extended duration of time. It should be able to communicate with its team-mates in order solve a global strategy of operation. The robot is supposed to perform multiple tasks and it should be fitted with multi-purpose sensors.

1.2.3 Requirements for Self-localization

There are certain requirements which the localization algorithm should fulfil. They are listed below:

- The localization algorithm should not enforce the constraint that all features are lying on or close to the ground plain.
- The environment is not necessarily surrounded by rectangular walls.
- The environment can not be modified with artificial landmarks or by placing active beacons as an aid in navigation.
- Globally distinct landmarks are few and they are frequently occluded by other moving objects in the environment.
- The environment is highly dynamic. For example in the test application there are 8 high speed robots and an even higher speed ball. Tinyphoon can reach a speed of 3.5 m/s with an acceleration of 5 m/s^2 [Deu07].
- The localization algorithm should be able to address the kidnapped robot problem [EM92, TFBD01] as well as the bootstrap problem [Ren93]. In kidnapped robot problem, the robot is displaced without information, while the bootstrap problem is a special case of the kidnapped robot problem where the robot is aware that it has to do global localization [FBDT99a].

It is assumed that there exist features that are globally distinct. However, such features are supposed to be scarce and often occluded by other objects in the environment. The robot motion is on a flat surface where the robot pose has three degrees of freedom (x, y, and θ). (x, y) indicate position of the robot and θ is its orientation. The dynamic nature of of the target application environment and requirement of onboard processing warrants that algorithms should be developed/selected so that they require less computation time.

1.3 Proposed Methodology

A simple solution for robot position estimation would be to start at a known location and track the robot position locally using methods such as odometry or inertial navigation [COB01]. These methods have proven to be efficient and provide good short term position estimates but suffer from unbounded error growth due to integration of minute measurements to obtain the final estimate [Bor98]. The failure of local methods to track the robot position over extended periods of time and the requirement of an initial estimate makes global position estimation a necessity. In global position estimation robots use external[2] sensors to sense their environment and calculate position [AR98].

The process of global position estimation is often simplified by engineering the robot environment with active beacons or other artificial landmarks such as bar code reflectors and

[2]External sensors are gathering information from the robot's environment. Examples of such sensors are the stereo vision, laser range finder and sonar.

visual patterns [SN04]. Methods that don't require modification of the environment are less accurate and demand significantly more computational power [BEF96]. Additionally minimum number of features required to determine a unique global position are normally not available through the entire state space [BS06, BSGM06]. This leads to techniques where robot (with a known initial position) tracks its position and the accumulating position errors are mitigated by gathering information from the robot environment and applying corrections to the position estimate [BSGK07]. However, it is not always possible or at least desirable to provide the robot with a start point. Hence, the robot must be able to estimate its position from the very beginning or when/if it loses track of its position during navigation.

It is proposed to use range information to distinctive environment features. Feature extraction and range estimation is done with a binocular stereo vision system. To extend the robot field of view the cameras are mounted on a pivoted head. This aids in feature exploration which are assumed to be scarce. Unlike methods based on angle measurements, this method requires only two distinct landmarks [BS06]. Distinct landmark features are sparse in our application domain, which makes simultaneous acquisition of two or more landmarks difficult. Therefore, another method is proposed that requires only one distinct landmark [BSG06]. The method, however, requires independent estimation of the robot absolute orientation. To enable the robot to estimate its absolute orientation it is equipped with a compass in addition to odometry and the stereo vision system [NM05, NCB$^+$06].

However, instantaneous acquisition of distinct landmarks at all times through the entire state space is not possible as landmarks are few in number and are frequently occluded [BS06, BSG06, BSGM06]. Furthermore, global position estimation using only external sensors is time consuming [BEF96]. Similarly, only local sensors[3] are not enough for position estimation as they require knowledge of initial position and the errors accumulate unbounded [BDN07]. Therefore, it is required to use a combination of local and external sensors and fuse the information obtained from them. The growing position error is suppressed by acquiring features from the robot environment whenever possible.

Features found in the application environment are line based and color transitions. Lines are determined by a large number of pixels which makes it possible to locate them accurately and are natural in the sense that a number of lines can be found in structured environments [DYOC04]. The primary target is to extract the global structure of the line segments, in the presence of heavy occlusions. Line segments are extracted using Gradient Based Hough Transform (GBHT) which provide the strongest groupings of collinear pixels having roughly the same edge orientation [BSN05]. These groups are further processed to compute length and end points of line segments, which together with the length and direction of the normal completes the description of the line segments [AA94]. This is followed by classification of line segments into different groups based on width, color and the presence of parallel edges of opposite gradient. Corners, junctions and line intersections are determined by semantic interpretation of detected line segments. The motivation behind using GBHT and the semantic interpretation of line segments to extract corners, line intersections and junction can be found in Chapter 3.

The information obtained from both the internal and external sensors is uncertain and incomplete. These uncertainties are represented by probability models. The robot's local sensors are calibrated for systematic errors, whereas, for the non-systematic error, it is assumed that it can be represented by zero mean Gaussian distribution [BSGK07]. Similarly, the robot

[3]Examples of internal or local sensors are digital encoders and gyroscopes.

observations are also represented by a Gaussian distribution [BSKN07]. Information from internal and external sensors is fused using Extended Kalman Filter (EKF). Alternative approaches are discussed in Chapter 2. Maintaining a rough estimate of the robot position helps in features extraction and matching with the global map of the robot environment [CKKK94]. With an estimate of its current pose, the robot will search for features within a local area instead of searching over the whole image [TRM+05].

The proposed system is shown in Fig. 1.2. In stereo vision it is preferable that the two image planes are coplanar and parallel to their baseline (line between the two centers of projection) [Bra96]. Given the case that two stereo images are coplanar and parallel to their baseline they are *rectified* [HS93]. In rectified images the epipolar lines are the scan lines in both images. One can produce rectified images either by placing two cameras exactly parallel or compute virtual parallel images planes in order to have rectified images. Features extracted from intensity images are used for correspondence analysis and depth computation. Three dimensional position of features along with their descriptions is fed into the localization module. The localization module is fusing this information with the one from local sensors and map of the environment to deduce its own position and position of other moving objects. The dead reckoning information consist of the output of the wheel encoders, gyroscopes and the compass. However, this work is using only information from the digital encoders and the compass. Properties of the landmarks such as their global location and type etc can be obtained from the environment model. The localization module has access to one of the intensity image. This is used for extracting information such as color or re-confirming the presence or absence of certain features.

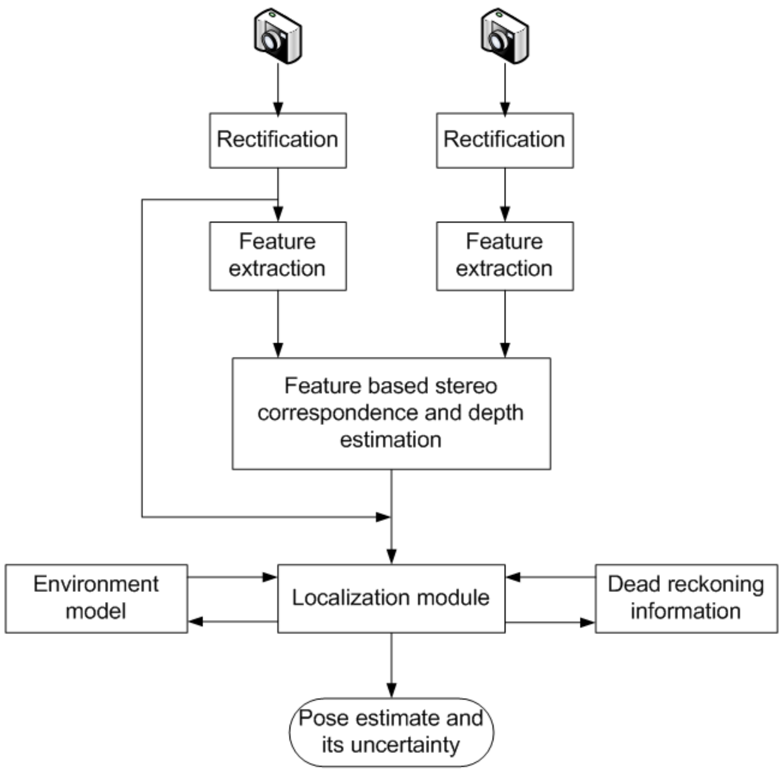

Rectification
Rectification
Feature extraction
Feature extraction
Feature based stereo correspondence and depth estimation
Environment model
Localization module
Dead reckoning information
Pose estimate and its uncertainty

Figure 1.2: The proposed system

As stated earlier the robot location is estimated using landmark based global localization methods which also provide uncertainty. The globally estimated position and its uncertainty is combined with the local information using a new hybrid method. This approach results in significant performance improvements. Further discussion on the method and experimental results are provided in Chapter 5 and Chapter 6. The use of stereo vision system enables range computation to distinct landmarks and the use of the range based methods. With this framework reduction in minimum number of landmarks required for vision based global self-localization is achieved.

Chapter 2

Related Work

The position estimation problem is not new. Leonard et al. [LDW91] put this problem as: "The problem of position determination has been of considerable interest over the last 4000 years. Today, navigation is a well-known quantitative science, used routinely in maritime and aviation applications. Given this, the question must be asked as to why robust and reliable autonomous mobile robot navigation remains such a difficult problem. In our view, the reason for this is clear, it is not the navigation process per se that is a problem, it is the reliable acquisition or extraction of information navigation beacons, from sensor information, and the automatic correlation or correspondence of these with some navigation map that makes the autonomous navigation problem so difficult".

Methods addressing the localization problem are reviewed in this chapter. Different alternatives dealing with sub-problems are discussed in the related chapters. The presentation of the chapter is primarily based on categorization of localization systems by Borenstein et al. [BEF96]. However, the review is not limited to methods that explicitly fall in those categories.

2.1 Relative Position Estimation

In relative position estimation, the robot is given initial estimate of its position, which is tracked as the robot moves in its environment. Methods in this category are based on integration of minute motion information for position update and can be divided into two groups based on the type of measurements [BEF96]. This section provides further details on the two groups of relative position estimation.

2.1.1 Odometry

Odometry is based on the fundamental assumption that wheel revolutions can be translated into linear displacement relative to the floor. Odometry provides good short-term accuracy, is inexpensive, and allows high sampling rates [BF96, Bor98]. However, errors in odometry limit the validity of the fundamental assumption. These errors can be put into two categories: systematic errors and non-systematic errors [BEF96].

Systematic errors could be due to the fact that different wheels of the robot have different diameters and the actual wheel diameter is different from the nominal one. The separation of the wheels, called wheelbase, may also be different from its nominal value. Misalignment of the wheels and finite encoder resolution also account for systematic errors in robot odometry.

Systematic errors accumulate constantly, whereas, non-systematic errors appear unexpectedly, and may cause large position errors. These errors appear due to a variety of reasons such as traveling on uneven floors and moving over unexpected objects on the floor. Furthermore, wheel-slippage due to slippery floors, over acceleration, fast turning, non-point wheel contact with the floor and interaction with external bodies in the robot environment could cause large errors in the position estimate.

Fig. 2.1 illustrates the difference between systematic and non-systematic odometry errors. The robot starts on the right side and travels towards the left. The start and end of the true path are marked as 'A' and 'B' respectively. The true path of the robot is shown in red color, whereas, the estimated path is shown in blue color. The ellipses represent the uncertainty in the robot position. Fig. 2.1(a) simulates the effect when one of the wheel's diameter is larger than its nominal value, which causes it travel more distance than the other. The result is that the robot is following a circular path instead of moving on a straight line.

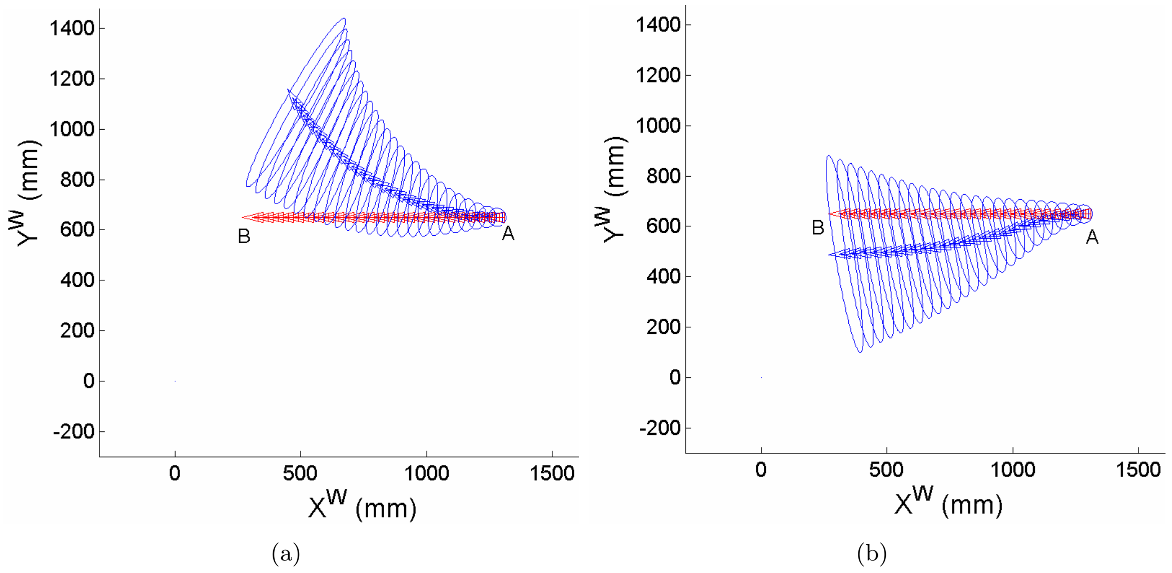


Figure 2.1: Simulated run representing errors in robot odometry (a) systematic error: simulating effect incorrect wheel diameter (b) non-systematic error: simulating robot motion over an uneven floor. The start and end points are marked 'A' and 'B', respectively. The true path of the robot is shown in red color, whereas, the estimated path is shown in blue color. The ellipses represent the uncertainty in the robot position

The effect of moving over a slightly uneven floor is simulated in Fig. 2.1(b). The irregular shape of the resultant path of the robot is due to the random error in odometry. The growing ellipses indicate that the robot position uncertainty is increasing as the error at every time step is accumulating with the existing error in position. There is no feedback in the process and the error accumulation continues without bounds.

The robot odometry can be calibrated for systematic errors using methods such as the University of Michigan Benchmark (UMBmark) [BF96], whereas, the non-systematic errors are modeled probabilistically [Wan88]. Chong and Kleeman [CK97] report a detailed analysis of uncertainty in the odometry of a differential drive robot. To take into account the error difference for different length of travel, they assume that uncertainty in distance covered by each wheel is proportional to its absolute value. The curve traversed by the robot between the two time intervals is divided into infinite number of small steps, where the size of each step tends to zero. They assume that the radius of curvature for all the intermediate steps remains constant. This way the uncertainty in final position of the robot stays the same irrespective of the number of steps in which the distance is covered. It is observed that if the radius of curvature is assumed to be constant then this division is not required. The problem of traveling the same distance with different time steps resulting in different uncertainty, is solved by making the uncertainty proportional to the distance covered and if one starts at the low level of wheel counts and distance covered. Other methods to reduce odometry error are reported in [Bor98, TTA94, KO94]. The only thing that can be achieved is to reduce the rate of divergence.

Even though errors are accumulated and grow without bounds, odometry is used in almost all mobile robots [BEF96]. The reason for the widespread use of odometry is that better and more reliable position estimation can be achieved by fusing odometry data with absolute position measurements [Cox91, CC92]. Similarly, many model based approaches to position estimation such as [GSO92, CC92] assume that the robot can maintain its position well enough to allow the robot to look for landmarks in a limited area to achieve short processing time and to improve matching correctness [Cox91].

2.1.2 Inertial Navigation

An alternative method for relative position estimation is inertial navigation, which involves continuous sensing of angular rate information and accelerations in each of the three directional axes. The angular rate information and acceleration can be obtained by direct measurements using gyroscopes and accelerometers, respectively. The robot orientation can be estimated by integrating the angular rate information, whereas, the velocity rate measurements need to be integrated twice to obtain position. Sensors used for inertial navigation have error sources that are independent from the external sensors. Furthermore, these sensors are non radiating and non-jammable [BEF96].

Inertial sensor data drift with time, which is the most important contributor to navigation errors, and is dependent on the device technology [BDW93]. Because of the need to integrate rate data, even very small errors can cause an unbounded growth in the error of integrated measurements [BEF96]. This makes inertial sensors unsuitable for accurate positioning over an extended period of time.

Accelerometers are also sensitive to uneven grounds, as any disturbance from the perfectly horizontal position will cause the sensor to detect the gravitational acceleration. Barshan and Durrant-Whyte aim at overcoming this problem by using a tilt sensor [BDW93]. The information provided by the tilt sensor is used to cancel the effect of the gravitational acceleration. However, the information from the tilt sensor can be used only when the robot is stationary, since tilt sensors are sensitive to acceleration [BDW93].

Barshall and Durrant-Whyte [BDW93] evaluate performance of inertial navigation systems for robot location estimation. They report a five to six fold reduction in the angular measurement error by using an error model for the gyroscope within an EKF [May79] framework. However, even with the usage of the error model, a drift rate of 1 to 3 °/min could still be unacceptable for most mobile robot applications [BEF96]. Using tilt-compensated accelerometer a position drift rate of 1-8 cm/sec, depending on the frequency of acceleration changes, was achieved.

To reduce odometry errors and improve the overall accuracy of position estimation, Komoriya and Oyama [KO94] report fusion of information from a fiber optic gyroscope with odometry information. Since in odometry based methods change in robot orientation is computed based on the distance covered by different wheels, they propose that error in robot orientation can be reduced by directly measuring the angular velocity with a fiber-optic gyroscope. The information from two different sensors is fused through a Kalman Filter (KF).

The EKF uses the encoder and the gyroscope exclusively and does not correct their errors mutually but only attempts to reduce errors based on the independent models [PCCL96]. An alternative approach for reducing systematic odometry errors with the help of a gyroscope is proposed by Chung et al. [COB01] and Park et al. [PHL98]. They use the difference between the angle from the differential encoder and the gyroscope in an indirect KF [PCCL96] to compensate the encoder error and the gyroscope error mutually. However, as stated earlier, even small errors grow unbounded and it is necessary to reset them using absolute measurements from the robot environment [BDW93].

2.2 Landmark-Based Methods

Landmarks are distinct features that a robot can recognize from its observations. Landmarks can be geometric shapes such as lines and circles, and can be tagged with additional information such as adding a bar code. Normally, they have fixed position in the robot environment, with respect to which the robot estimates its position. The types of features used for localization play an important role in the success or failure of a method. Features used for localization can broadly be divided into two groups; natural and artificial. Artificial landmarks are specially designed and placed in the environment as an aid in robot navigation [BEF96].

Two different techniques are used for landmark based positioning, which differ in the type of input data. These techniques are called triangulation and trilateration. Trilateration is the determination of a robot's position based on distance measurements to known landmarks. Whereas, in triangulation, bearing to different landmarks in the environment is measured. The remainder of this section discusses state of the art position estimation methods using natural and artificial landmarks. Additionally, the use of active beacons is discussed for position estimation.

2.2.1 Natural Landmarks for Localization

In landmark based navigation methods it is most desirable to base the algorithm on natural landmarks. Data from a variety of sensors can be processed to extract natural landmarks.

Sugihara [Sug88] presented one of the pioneering studies in robot self-localization using angle measurements from single camera images. This algorithm tries to find the location of the robot inside a room using vertical edges extracted from an image taken by an onboard camera whose optical axis is kept parallel to the ground. He considers two different classes of the problem. In the first one all vertical edges are identical, whereas, in the second class, the vertical edges are distinguishable from each other but the exact directions in which they are seen are not given, only the order of their appearance is given. The algorithm runs in time $O(n^3 \lg(n))$, where n is number of identical landmark points. Sugihara assumed that there is no error in angle measurements. This work is furthered by Krotkov [Kro89], who also studies the resulting uncertainty in robot location when angle measurements are erroneous.

In addition to the absolute position estimation, vision based extraction of natural landmarks have been used to track position of the robot. For example, Panzieri et al. [PPSU01] report experimental results for a robot navigating in a corridor scene. Ceiling lights are extracted from vision data as natural landmarks for reducing the accumulating error in the robot position. They use a topological representation of the environment. Similarly, Howard and Kitchen [HK99] use walls and doorways as landmarks. They maintain a probability distribution across all possible robot positions and track them over time using Baye's rule. Dulimarta and Jain [DJ97] extract door number plates and ceiling lights in camera images for position estimation. Their robot is also equipped with ultrasonic sensor ring (24 sensors) and an infra red proximity sensor. The ultrasonic sensor enhances depth measurements. Chenavier and Crowley [CC92] apply a triangulation based method to track the robot position using natural landmarks extracted from single camera images of a known environment. An EKF is implemented to combine data from the natural landmarks and odometry. Libuda and Kraiss [LK04] and Braunegg [Bra93] extract natural environment features using a stereo vision system for robot navigation. The former uses high level interpretation of the scene to extract walls, door and floor, whereas the latter detects vertical straight lines which are aggregated into features i.e. doorways, windows, bookcases etc. to identify locations.

As with vision based methods, natural landmarks have been widely used for position estimation using range sensors. Commonly used landmarks extracted from laser range data are line segments [GS96, GWN01, LFW96, GSO92], points and lines [JC01], corners [ZRJ03]. Lingemann et al. [LNHS05] report a fast method of pose tracking and absolute localization using laser scan matching. Natural features are extracted and matched between scans. Larsson et al. [LFW96] use Range Weighted Hough Transform (RWHT) [FLW95] to extract lines from range data. The association between the map and extracted features is done in Baye's framework. The range of the detected line features is combined with dead-reckoning information with the help of a KF. Similarly, naturally occurring geometric beacons in known environment are extracted from ultrasonic range data [MAG$^+$02]. This system is using a ring of 24 ultrasonic sensors for sensing the environment.

2.2.2 Artificial Landmarks for Localization

Natural landmarks are desirable but detection and matching characteristic features from sensor data remains the main problem in navigation that uses naturally occurring landmarks. Therefore, the environment is often engineered by placing artificial landmarks. Artificial landmarks make detection process easier [AH93]. They are designed for optimal contrast and have known size and shape [BEF96]. The selection of features is important since it will determine the complexity in feature description, detection, and matching. Different types of features to be detected in camera images have been reported in literature.

Briggs et al. [BSB$^+$00] use self-similar intensity patterns as landmarks. A binary barcode is added to distinguish different landmarks. Nested square patterns are used in [CKP95], whereas Li, So and Tso attach a deep green colored square with a white colored square at the center and two black colored squares at two corners of a wall [LST02]. They use a vision system to locate the landmark. After the landmark is detected a LRF is used to calculate its distance from the robot position. Similarly, a landmark composed of two vertically neighboring color patches is used in [JKLK02]. Kabuka and Arenas [KA87] localize their robot with respect to landmarks that consist of two segments; a relative displacement pattern and an identification code and Borenstein uses a landmark consisting of black rectangles with white dots [Bor87].

Feng et al. [FFK92] use a circular landmark and apply the Optical Hough Transform (OHT) to extract the parameters of the ellipse on image plane in real time [SS86]. Lazanas and Latombe [LL92] use sheets of paper with a unique checkerboard pattern placed at the ceiling as artificial landmarks.

2.2.3 Active Beacons

To achieve a highly reliable solution and further simplify the problem of detection and matching, an active beacon system is designed and deployed in the target environment [SN04]. Active beacons provide accurate positioning information with minimal processing, which result in high sampling rates and yields high reliability. However, it also incurs high cost in installation and maintenance [BEF96].

Multiple type of transmitters could be used i.e. laser pointer [HTMA03], ultrasonic [Kle92] or ultrasonic and radio frequency [SZO$^+$03] etc. In [HTMA03], the emitter is a laser pointer that acts as a beacon. The receiver intercepts this signal and calculates its position and orientation. Kleeman [Kle92] uses ultrasonic beacons mounted at different locations in the environment. Information from these beacons in fused with information from dead-reckoning sensors using an iterative EKF. The Global Positioning System (GPS) can be considered as such a system. A good discussion on placement of beacons to calculate the robot position using triangulation based methods can be found in [SS00].

Beacons, could also be retro-reflective markers, which can be easily detected by a mobile robot based on their reflection of energy back to the robot. Given known positions for the optical retro-reflectors, a mobile robot can identify its position whenever it has three such beacons in sight simultaneously [SN04].

As noted by Borenstein et al. [BEF96], the main problem with the active beacons is the need of powerful transmitters to ensure omni-directional transmission over large distances. However, if such powerful beacons are not feasible, they are focused within a cone-shaped propagation pattern. This results in beacons not visible in many areas, where position can not be calculated as at least three beacons must be visible for triangulation.

2.3 Map-Based Positioning

Map-based positioning is a technique in which the robot creates a map of its local environment using its sensors. This local map is then matched with the global map of the environment.

If a match is found the robot position and orientation can be estimated. This method uses the naturally occurring structure of the environments to derive position information without modifying the environment. This method can be used to generate an updated map of the environment which may allow a robot to learn a new environment and to improve positioning accuracy through exploration [BEF96].

Map-based positioning requires that there be enough stationary, easily distinguishable features that can be used for matching, the sensor map be accurate enough to be useful. Additionally, it requires a significant amount of sensing and processing power [SN04].

The sub-problems in map-based positioning are the map representation and matching. If the robot is not given a map of its environment it has to build one using its sensors. This is termed as map building and is the dual of localization. A detailed presentation about map-based positioning can be found in [BEF96, SN04]. The discussion in this section is partially adapted from these sources.

2.3.1 Map Representation

The problem of representing the environment is the dual of the problem of representation of the robot's possible position or positions. Siegwart and Nourbakhsh [SN04] note that:

1. The precision of the map must appropriately match the precision with which the robot needs to achieve its goals.
2. The precision of the map and the type of features represented must match the precision and data type returned by the robot's sensors.
3. The complexity of the map representation has direct impact on the computational complexity of reasoning about mapping, localization, and navigation.

The two common representations are geometric maps and topological maps [BEF96]. The former represents objects according to their absolute geometric relationships, whereas, the latter approach is based on recording the geometric relationships between the observed features with respect to an arbitrary coordinate system.

2.3.1.1 Geometric Representation

Geometric maps can be continuous or discrete.

Continuous representation

In continuous maps, features are represented by their exact position. In this approach one normally assumes that the representation specifies all environmental objects in the map. Any area in the map that is devoid of objects has no objects in the corresponding portion of the environment. Hence, the total storage is proportional to the density of objects in the environment [LL92].

For example a robot with a range sensor could extract lines from dense range scans. An appropriate continuous mapping approach is to populate the map with a set of infinite lines. The continuous nature of the map guarantees that lines can be positioned at arbitrary positions in the plane and at arbitrary angles.

The CS Freiburg's self-localization relies on scans of angular distance readings obtained by a LRF mounted on the robot [GTN99, GHH$^+$99, GS96]. This is the most successful method in the Robocup domain. Two contributing factors for its success were that the LRF provides rather precise and reliable measurements of the distance to the walls surrounding the field and that the scans are dense [UNMK03].

Iocchi & Nardi [IN00] uses field lines as markers for self-localization. The lines are extracted from the range data using Hough Transform (HT) [Hou62, DH72]. They explicitly extract symbolic field markings by searching for local maxima in the Hough domain. Their original algorithm [IN00] considers lines detected locally and use odometry to remove ambiguities. In a similar approach, Hans Utz et al. [UNMK03] use lines and the center circle of the soccer field for matching. The key advantage of a continuous map representation is the high accuracy and expressiveness with respect to the environmental configuration as well as the robot position within that environment.

Discrete representation

Due to the high cost of continuous representation, more simplifications may be desired. A form of simplification is abstraction, which is a general decomposition and selection of environmental features. The decomposition may be geometric or topological. The geometric decomposition is further divided into exact cell decomposition and fixed cell decomposition.

Exact cell decomposition is a standard lossless form of decomposition. This form of map representation tessellates the space into areas of free space and stores each of such area as a single node [SN04, p.204]. This method is useful only in situation where the only requirement is the robots ability to traverse from each area of free space to the adjacent areas and the exact position of the robot within each area of free space does not matter. This method becomes infeasible if information about free-space and obstacles of particular types is unknown or expensive to collect.

An alternative is fixed cell decomposition, in which the world is represented as a grid and it is possible for narrow passageways to be lost. There is another approach called adaptive cell decomposition, where the cell size is variable and not fixed.

One popular version of fixed cell decomposition is called occupancy grid representation [HE85]. In occupancy grid, the environment is represented by a discrete grid, where each cell is either filled or empty. This method is of particular value when a robot is equipped with range-based sensors because the range values of each sensor, combined with the absolute position of the robot, can be used directly to update the filled or empty value of each cell.

The size of the map in robot memory grows with the size of the environment and the approach is not compatible with the close-world assumption as memory must be set aside for every cell in the matrix. The occupancy grid (or any other fixed decomposition method) imposes a geometric grid on the world regardless of the environment details. For these reasons, an alternative, called topological decomposition, has been the subject of some exploration in mobile robotics. This approach is discussed in the next subsection.

2.3.1.2 Topological Representation

If geometry is not a salient feature of the environment, direct measurements of geometric qualities can be avoided. In topological map representations concentration is instead focused on characteristics of the environment that are relevant to the robot for localization. The resulting presentation takes the form of a graph where nodes represent the observed features and edges represent the relationships between features. Topological maps can be built and maintained without any estimates for the position of the robot, which makes errors in this representation independent of any errors in the estimates for the robot position [BEF96].

Topological representation can be effectively used if the environment contains important non geometric features. These features have no ranging relevance but are useful for localization. Fennema et al. [FHR$^+$90] outlined a system for robot navigation in a partially modeled environment. An internal model of the robot environment and models of the robot actions enable the robot to predict its actions. The environment is represented as a graph of connected nodes called locale.

2.3.2 Map Building

The robot has to build a map of its own environment if it is not provided with one. The map building problem may be stated as "Given the robot's position and a set of measurements, what are the sensors seeing?" [Ren93].

The position estimation strategies that use map-based positioning rely on the robot's ability to sense the environment and to build a representation of it [TA93]. The construction and maintenance of the robot map will then be to build an abstract description of the most recent sensor data, match and determine their correspondence with the current contents of the composite local model, modify the components of the composite local model and reinforce the confidences to reflect the results of matching [Cro89].

The robot must explore its environment to map uncovered areas with the assumption that the robot begins its exploration without having any knowledge of the environment. Then, a certain motion strategy is followed which aims at maximizing the amount of covered area in the least amount of time.

2.3.3 Map Matching

One of the important and challenging aspects of map-based navigation is map matching, which is establishing the correspondence between a current local map and the stored global map [BEF96]. In the computer vision community, work on map matching is often focused on the general problem of matching an image of arbitrary position and orientation relative to a model [TA93].

In order to match the current sensory data to the stored environment model reliably, several features must be used simultaneously which decreases the likelihood of a mismatch but increases the processing. The search to determine the correct correspondence between features extracted from the sensor data and model features is usually constrained in some form [Cox91]. A good estimate of the current position from odometry reduces the space for feature search. Chenavier and Crowley [CC92] maintain that a realistic model for the odometry and its associated uncertainty is the basis for the proper functioning of their system. The landmark detection as well as the updated position calculation rely on odometric estimates.

Schiele and Crowley [SC94] discussed different matching techniques for matching two occupancy grids. The first grid is the local grid that is centered on the robot and models its vicinity using the most recent sensor readings. The second grid is a global model of the environment furnished either by learning or by some form of computer-aided design tool.

Rencken [Ren93, Ren94] has made substantial contributions toward solving the bootstrap problem resulting from the uncertainty in position and environment. This problem exists when a robot must move around in an unknown environment, with uncertainty in its odometry-derived position.

Bauer and Rencken [RR95] develop path planning methods that assist a robot in feature-based navigation. The novel aspect of this method is a behavior that steers the robot in such a way that the observed features stay longer in view and can thus serve as a navigation reference for a longer time.

Fennema et al. [FHR+90] use a vision sensor to correct the model's predictions about current location or to progress towards some goal. The images are matched to the map by first matching the two-dimensional projection of landmarks to lines extracted from the image. The best fit minimizes the difference between the model and the lines in the data. Once the correspondence between model and two-dimensional image is found, the relation of the robot to the world coordinate is estimated. This relation is expressed as the rotation and translation that will match the robot and world coordinate systems.

Talluri and Aggarwal [TA90, TA91] reported their extensive work on model-based positioning. They use three-dimensional building models as a world model and a tree search is used to establish a set of consistent correspondences.

Kak et al. [KALAC90] use their robot's encoders to estimate its position and heading. A visual sensor in conjunction with a known model of the building is used to derive a more accurate estimate of the robot's position and pose. The approximate position from the encoders is utilized to generate, from the known model, an estimated visual scene that would be seen. The scene is then matched against the actual scene viewed by the camera. Once the matches are established between the features of the two images, the position of the robot can be estimated with a reduced uncertainty. The generation of an estimated visual scene is necessary as the sensory data and the stored model are in different formats [TA93].

2.4 Probabilistic Methods

Due to noisy measurements and incomplete sensor and environment models, position estimation is not deterministic and hence probabilistic methods are used. During the past decade there has been a growing interest in probabilistic methods used for mobile robots and a number of approaches have been reported [Fox98, FBT98, FBDT99a, NPB95, BFHS96, FBTC98, JK01, ACSS03, KE02, FG03]. In probabilistic localization the task is to estimate probability $p(\mathbf{p}_k|\mathbf{d}_{0\dots k})$, where $\mathbf{p}_k$ is a random vector representing state of the robot at time k and $\mathbf{d}_{0\dots k}$ represent noisy data that the robot has received starting at time 0 to the current time k. Usually, for robot localization, both relative and absolute measurements are used. The relative information is obtained by using the robot encoders, whereas, the absolute information is obtained by using the robot's perceptual sensors such as cameras, sonar, LRF etc. The relative information is also called *motion command* or *action* and is denoted by $\mathbf{u}$. For instance $\mathbf{u}_{k-1}$ is action executed at $k-1$ which will change the robot pose from $\mathbf{p}_{k-1}$ to $\mathbf{p}_k$. The absolute measurements also known as *observations* are denoted by $\mathbf{z}$ [Fox98]. Using this notation, $\mathbf{z}_k$ represents information retrieved from the robot environment using the robot external sensors.

Using the two types of data the robot belief to be at location $\mathbf{p}_k$ at time k can be written as [FBT98]:

$$p(\mathbf{p}_k|\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k}) \tag{2.1}$$

Using Bayes theorem (2.1) can be written as follows:

$$p(\mathbf{p}_k|\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k}) = \frac{p(\mathbf{z}_k|\mathbf{p}_k,\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k-1})p(\mathbf{p}_k|\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k-1})}{p(\mathbf{z}_k|\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k-1})} \tag{2.2}$$

The denominator in (2.2) is independent of $\mathbf{p}_k$ and can be replaced by a constant as given by (2.3). This is a normalizer which makes sure that $\int p(\mathbf{p}_k|\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k})d\mathbf{p}_k = 1$ [TFBD01]. After this replacement (2.2) can be written as given by (2.4).

$$\eta = \frac{1}{p(\mathbf{z}_k|\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k-1})} \tag{2.3}$$

$$p(\mathbf{p}_k|\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k}) = \eta p(\mathbf{z}_k|\mathbf{p}_k,\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k-1})p(\mathbf{p}_k|\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k-1}) \tag{2.4}$$

Using total probability theorem $p(\mathbf{p}_k|\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k-1})$ can be written as follows:

$$\begin{aligned} & p(\mathbf{p}_k|\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k-1}) && (2.5)\\ = & \int p(\mathbf{p}_k|\mathbf{p}_{k-1},\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k-1})p(\mathbf{p}_{k-1}|\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k-1})d\mathbf{p}_{k-1} && (2.6)\end{aligned}$$

Substitution of (2.6) in (2.4) results in:

$$\begin{aligned} & p(\mathbf{p}_k|\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k}) = \\ & \eta p(\mathbf{z}_k|\mathbf{p}_k,\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k-1}) \int p(\mathbf{p}_k|\mathbf{p}_{k-1},\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k-1})p(\mathbf{p}_{k-1}|\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k-1})d\mathbf{p}_{k-1}\end{aligned} \tag{2.7}$$

If this belief is denoted as $Bel(\mathbf{p}_k)$, then (2.7) can be written as follows:

$$\begin{aligned} & Bel(\mathbf{p}_k) = \\ & \eta p(\mathbf{z}_k|\mathbf{p}_k,\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k-1}) \int p(\mathbf{p}_k|\mathbf{p}_{k-1},\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k-1})Bel(\mathbf{p}_{k-1})d\mathbf{p}_{k-1}\end{aligned} \tag{2.8}$$

Equation (2.8) recursively estimates the robot belief. To be able to use this equation it is required to specify three probability distributions [TFBD01]. $p(\mathbf{p}_k|\mathbf{p}_{k-1},\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k-1})$, known as *motion model*, models the probability of the robot to be at state $\mathbf{p}_k$ given the current state $\mathbf{p}_{k-1}$, actions $\mathbf{u}_{0\ldots k-1}$ and observations $\mathbf{z}_{1\ldots k-1}$. The distribution $p(\mathbf{z}_k|\mathbf{p}_k,\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k-1})$, known as *observation model*, describes the probability of observing $\mathbf{z}_k$ conditioned on the current state, actions $\mathbf{u}_{0\ldots k-1}$ and observations $\mathbf{z}_{1\ldots k-1}$. Finally, it is required to know the belief $Bel(\mathbf{p}_0)$. If there is no information of the initial state distribution, $Bel(\mathbf{p}_0)$ will have a uniform distribution over the entire state space. The complexity of (2.8) depends on the number of independent variables which grow linearly with the passage of time.

The state vector $\mathbf{p}_k$ is a continuous random vector. The number of components (degrees of freedom of the robot pose) of the state vector depend on the type of robot motion. For motion over a flat terrain this vector has three components, whereas, for a 3D motion the state vector will have six components: three for position and three for rotation. Representation of arbitrary probability distribution over the entire state space could be a difficult task.

Another issue is the representation of motion and observation models. Making models that adequately describe real actuator and sensor systems could be a difficult task. This problem becomes more pronounced when measurements are high-dimensional, for instance when camera images are used. One way to deal with high modeling complexity is to let the robot learn the action and sensor models [Thr98].

The remainder of this section discusses different probabilistic algorithms and the methods they use to deal with the complexity issues mainly the representational complexity. These methods can be broadly divided into two groups. The first group of methods uses a discretization of the state space and is based on Partially Observable Markov Decision Processes (POMDP) [Fox98]. The second group may also be regarded as analogous to the (POMDP), with the key difference that the state variables are continuous [Wik07b].

For the methods described here, it is supposed the environment satisfies the Markov property, which means that it has to have two properties. These properties are the conditional independence of actions and conditional independence of observations [Fox98]. The independence of actions means that all states, actions and observations prior to time $k-1$ do not affect the state at time k, this means that the state $\mathbf{p}_k$ at time k only depends on the previous state $\mathbf{p}_{k-1}$ and the last action $\mathbf{u}_{k-1}$ performed by the robot. This can be stated mathematically as:

$$p(\mathbf{p}_k|\mathbf{p}_{1\ldots k-1},\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k-1}) = p(\mathbf{p}_k|\mathbf{p}_{k-1},\mathbf{u}_{k-1}) \tag{2.9}$$

which states that at time k, the state can be predicted with the help of motion models, if state and motion commands at time $k-1$ are known.

Similarly, the independence of observation means that the observation $\mathbf{z}_k$ at time k depends only on the state of the world at time k. Once $\mathbf{p}_k$ is known, all other measurements, prior states, and actions have no influence on observation $\mathbf{z}_k$. This property of the environment can be stated as follows [Fox98]:

$$p(\mathbf{z}_k|\mathbf{p}_{1\ldots k},\mathbf{u}_{0\ldots k-1},\mathbf{z}_{1\ldots k-1}) = p(\mathbf{z}_k|\mathbf{p}_k) \tag{2.10}$$

which makes it explicit that at a given time robot observation only depends on the state at that time and does not depend on observation, action or robot position at any other time or by another sensor. Substitution of (2.9) and (2.10) in (2.8) results in the following expression for position belief:

$$\begin{aligned} &Bel(\mathbf{p}_k) = \\ &\eta p(\mathbf{z}_k|\mathbf{p}_k)\int p(\mathbf{p}_k|\mathbf{p}_{k-1},\mathbf{u}_{k-1})Bel(\mathbf{p}_{k-1})d\mathbf{p}_{k-1} \end{aligned} \tag{2.11}$$

2.4.1 Markov Localization

Markov Localization (ML) is a special case of state estimation within the framework of POMDP. POMDP consist of a finite set of states, finite set of observation and an initial state distribution. Each state has a finite set of actions that can be executed on that state. Furthermore, it consist of a transition and observation functions. The transition function states the probability of state change from one state to another under certain action, whereas, the observation function describes observation probability from a state. POMDP also consists of a reward for executing certain action from a state [KS98].

When applying POMDP to a localization problem, the agent is a mobile robot and state of the world is the position of the robot within its environment. This state cannot be observed directly but can only be estimated based on uncertain sensory information, which makes state estimation based on incoming sensor data an important component of POMDP.

As stated earlier, within the framework of POMDP the states are finite. Therefore, the state space has to be discretized. After discretization the state $\mathbf{p}_k$ will be defined only at discrete locations within the state space and the integrals in (2.8) and (2.11) become summations and the belief over this space can be computed and stored explicitly [Neg03]. Using the discrete state space the belief that the robot has to be at location l can be written as follows:

$$p(\mathbf{p}_k = l|\mathbf{u}_{0\dots k-1}, \mathbf{z}_{1\dots k}) = \frac{p(\mathbf{z}_k|\mathbf{p}_k = l, \mathbf{u}_{0\dots k-1}, \mathbf{z}_{1\dots k-1})p(\mathbf{p}_k = l|\mathbf{u}_{0\dots k-1}, \mathbf{z}_{1\dots k-1})}{p(\mathbf{z}_k|\mathbf{u}_{0\dots k-1}, \mathbf{z}_{1\dots k-1})} \tag{2.12}$$

where

$$Bel(\mathbf{p}_k = l) = p(\mathbf{p}_k = l|\mathbf{u}_{0\dots k-1}, \mathbf{z}_{1\dots k}) \tag{2.13}$$

and the normalizer

$$\eta = \frac{1}{p(\mathbf{z}_k|\mathbf{u}_{0\dots k-1}, \mathbf{z}_{1\dots k-1})} \tag{2.14}$$

Application of Markov property results in:

$$p(\mathbf{z}_k|\mathbf{p}_k = l, \mathbf{u}_{0\dots k-1}, \mathbf{z}_{1\dots k-1}) = p(\mathbf{z}_k|\mathbf{p}_k = l) \tag{2.15}$$

$$p(\mathbf{p}_k = l|\mathbf{p}_{k-1} = l', \mathbf{u}_{0\dots k-1}, \mathbf{z}_{1\dots k-1}) = p(\mathbf{p}_k = l|\mathbf{p}_{k-1} = l', \mathbf{u}_{k-1}) \tag{2.16}$$

Drawing parallels with (2.6) $p(\mathbf{p}_k = l|\mathbf{u}_{0\dots k-1}, \mathbf{z}_{1\dots k-1})$ can be written as follows:

$$\begin{aligned} & p(\mathbf{p}_k = l|\mathbf{u}_{0\dots k-1}, \mathbf{z}_{1\dots k-1}) \\ = & \sum_{l'} p(\mathbf{p}_k = l|\mathbf{p}_{k-1} = l', \mathbf{u}_{0\dots k-1}, \mathbf{z}_{1\dots k-1})p(\mathbf{p}_{k-1} = l'|\mathbf{u}_{0\dots k-1}, \mathbf{z}_{1\dots k-1}) \\ = & \sum_{l'} p(\mathbf{p}_k = l|\mathbf{p}_{k-1} = l', \mathbf{u}_{k-1})Bel(\mathbf{p}_{k-1} = l') \end{aligned} \tag{2.17}$$

Equation (2.18) can be obtained by substituting (2.15), (2.16) and (2.17) into (2.12) as follows:

$$Bel(\mathbf{p}_k = l) = \eta p(\mathbf{z}_k|\mathbf{p}_k = l) \sum_{l'} p(\mathbf{p}_k = l|\mathbf{p}_{k-1} = l', \mathbf{u}_{k-1})Bel(\mathbf{p}_{k-1} = l') \tag{2.18}$$

The normalizer η can also be written in terms of motion and observation models as given by the following expression [Fox98]:

$$\begin{aligned}\frac{1}{\eta} &= \sum_{l} p(\mathbf{z}_k|\mathbf{p}_k = l, \mathbf{u}_{1...k}, \mathbf{z}_{1...k-1})p(\mathbf{p}_k = l|\mathbf{u}_{0...k-1}, \mathbf{z}_{1...k-1}) \\ &= \sum_{l} p(\mathbf{z}_k|\mathbf{p}_k = l)\sum_{l''} p(\mathbf{p}_k = l|\mathbf{p}_{k-1} = l'', \mathbf{u}_{k-1})Bel(\mathbf{p}_{k-1} = l'') \qquad (2.19)\end{aligned}$$

The existing approaches to ML can broadly be divided based on the type of discretization of the state space i.e. coarse or fine. In coarse discretization the state space is represented in a topological graph like structure with limited actions and observations. For instance Cassandra et al. [CKK96] use a topological representation of the state space where in each state the robot is able to observe if there is a doorway, wall, open area, or it is undetermined in each of the three nominal directions namely front, left and right. Five abstract actions are defined for the robot. These actions are turn-left, turn-right, move-forward, no-op and declare goal. Similarly, Nourbachsh et al. [NPB95] use ML for landmark based corridor navigation and the state space is organized according to the topological structure of the environment. Each node in the topological map corresponds to distinctive places in the hallways. Typical observations of the robot are detection of an open door, close door or an open hallway. Other approaches of ML using topological representation of the environment can be found in [SK95, KS98, HK96].

Another famous approach to ML is the grid based representation of the state space in order to deal with multi-modal and non-Gaussian densities at a fine resolution [FBDT99a, BFHS96, Fox98, FBT98, Thr98]. This method represents the state space by a fine grained grid with fixed spatial and angular resolution. An example representation of the state with three degrees of freedom is shown in Fig. 2.2. In this example, the state space size is $150cm \times 150cm \times 360\,^{\circ}$ and the resolution is $30cm \times 30cm \times 60\,^{\circ}$. Four of the cells in the grid are shown dark. The state here represent position in the xy-plane and orientation of the robot. For a resolution of $2cm \times 2cm \times 5\,^{\circ}$ and state space of size $5m \times 10m \times 360\,^{\circ}$, 9000000 locations are required to be updated whenever an action command or an observation is available.

The grid-based representation is more fine-grained as compared to topological approaches and more accurate position estimates are possible. However, this comes at high computational costs, which increase with finer resolutions. Another problem is the a priori commitment to the size of state space. Additionally ML is based on the assumption that the position of the robot is the only state in the world. Thus, this method is likely to fail in highly dynamic environments and special extensions are required to filter the damaging effect of sensor data corrupted by external dynamics [FBTC98].

2.4.2 Monte-Carlo Localization

ML based on topological representation of the state-space is too coarse to be useful for certain applications, whereas, the precision of the grid based implementation is dependent on the cell size. For larger environments the precision has to suffer or the calculation time is getting problematic. To overcome this problem, a new method of discretization the state space is to represent probability density by a set of weighted samples that are randomly drawn from it [TFBD01]. Each sample in the set is a location and the weight associated with it indicates

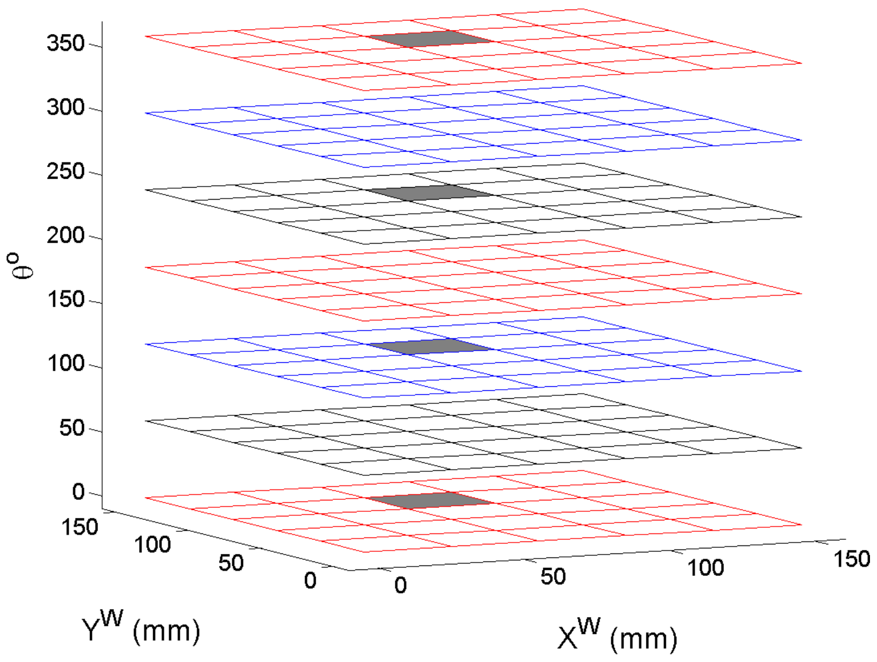


Figure 2.2: Grid formation in ML

its importance. It has been shown that the density can be approximately reconstructed from the samples [DFBT99, DBFT99].

This method is known as Particle Filter (PF) [FTBD00, KFM04, GDCT04, Fox03, WP03], Monte Carlo Localization (MCL) [FBDT99b, FBKT99, WBB05, MZPI04, TFBD01, DFBT99] or Condensation Algorithm (CA) [JAWA00, DBFT99]. The dense grid representation is replaced by a lower number of particles. By adjusting the number of particles taken from the belief distribution, the balance between accuracy and computational costs can be achieved.

Restating the basic idea of representing the belief by a set of N weighted samples $Bel(\mathbf{p}_k)$ can be written as follows [TFBD01]:

$$Bel\left(\mathbf{p}_k\right) \approx \left\{\mathbf{p}_k^{(i)}, w^{(i)}\right\}_{i=1\ldots N} \tag{2.20}$$

where $\mathbf{p}_k^{(i)}$ is a sample of the state vector $\mathbf{p}_k$. The non-negative weight $w^{(i)}$ represents importance of the sample. The weights are such that $\sum_{i=1\ldots N} w^{(i)} = 1$. This way the complete set of samples define a discrete probability function that approximates the continuous belief $Bel\left(\mathbf{p}_k\right)$. Higher the number of samples, more closer is the approximation.

The sample set is initialized according to initial knowledge of the state space. For instance in global localization the robot has a uniform belief over the state space and hence the samples are drawn from a uniform distribution, where each sample's weight is $1/N$. On the other hand, if the initial position is known the samples can be selected from Gaussian distribution centered at the true position.

Following the algorithm outlined in [TFBD01] and using notations introduced in this section, the recursive update may be summarized as follows:

1. Sample $\mathbf{p}_{k-1}^{(i)} \sim Bel\left(\mathbf{p}_{k-1}\right)$ from the (weighted) sample set representing $Bel\left(\mathbf{p}_{k-1}\right)$
2. Draw sample $\mathbf{p}_{k}^{(i)}$ from distribution $p\left(\mathbf{p}_k|\mathbf{p}_{k-1}^{(i)}, \mathbf{u}_{k-1}\right)$. This step accounts for motion update of each sample. The sample pair $< \mathbf{p}_{k}^{(i)}, \mathbf{p}_{k-1}^{(i)} >$ is distributed according to $p(\mathbf{p}_k|\mathbf{p}_{k-1}, \mathbf{u}_{k-1})Bel(\mathbf{p}_{k-1})$, which is called proposal distribution.
3. The mismatch between the proposal distribution and the correct distribution specified by (2.11) is accounted for by the following (non-normalized) new importance factor, which is assigned to ith sample.

$$w^{(i)} = p(\mathbf{z}_k|\mathbf{p}_k^{(i)})$$

Now the importance of this particle is in accordance with the latest observation. The above process is repeated N times, after which the weights are normalized by dividing the weight of each sample by the total sum.

The selection of the new set of particle as discussed above is called Sampling Importance Resampling (SIR). At each step particles are drawn from the actual set which are used as basis for the next step. The particles are drawn in accordance with their weight i.e. it is more likely to pick important particles for the next iteration. This way, areas with a high probability are gaining more weight by replicating their particles, whereas areas with a low probability are thinning out. This may introduce a source of degeneration that results from lack of appropriate particles close to the true state [Deu07, BDN07].

To avoid degeneration some particles are replaced by uniformly distributed new particles when required. This way empty areas are regularly refilled. Usually, a fixed number of particles are re-injected each round or when/if the quality of the estimation drops below a certain threshold [Deu07]. Time required for position estimation is proportional to the number of particles. More particles are required to resolve a global uncertainty, whereas, an already known position can be tracked with less number of particles. Reduction in number of particles can be achieved by dynamically adopting the number of particles [Fox03] or using techniques such as Rao-Blackwellization [GDCT04]. PFs for position estimation are the state of the art. In the following paragraphs some of the approaches reported during the past years are discussed.

Fichtner and Grossmann [FG03] report a sensor model and its use in a Monte Carlo framework for camera based pose estimation. Röfer and Jüngel [RJ04] extract edges corresponding to prominent lines in the environment and fuse them with motion input using Monte Carlo methods. Similarly, samples of the PF are weighted using the properties of the Fourier transform of omnidirectional images according to the similarity among images [MZPI04].

Kraetzschmar and Enderle [KE02] evaluate the MCL methods in an environment with sporadic features. They use angle and distance information of different features in the environment using single camera images. In this approach, goal posts and field corners are being used as landmark features, and distances to walls as distance features. The method becomes computationally expensive for getting high accuracy and is also memory intensive. As this method is used in the RoboCup environment, Hans Utz et al. [UNMK03], argue that a new approach is now necessary due to changes to the field geometry.

Lenser and Veloso [LV00] apply Monte Carlo method for localization of Aibo robots in four legged RoboCup league. They argue that if dead-reckoning position error is high and all particles are concentrated around a wrong position then continuing with these particles will not improve the localization accuracy and hence the robot should be declared lost.

2.4.3 Kalman Filter Based Techniques

KF is a recursive state estimator, which can deal with incomplete and noisy data [WB95, May79]. In its original form it can be applied to the localization problem when the motion and observation models are approximately linear with Gaussian error models. The initial belief of the robot state is also Gaussian, hence, the density remains Gaussian at all times. The beauty of Gaussian distribution is that only mean and covariance are sufficient for its complete description. Therefore, in KF based position estimation it is not required to store probabilities for every location, only parameters (mean and covariance) of the probability distribution are stored [GBFK98, GS96, SC94, AVT$^+$00, CKS$^+$00, Neg03].

A non-linear version of KF has to be used if the observation or state transition model (motion model) is non-linear. The EKF and the Unscented Kalman Filter (UKF) [JU97] are the famous non-linear versions of KF [Neg03]. The EKF does not require the observation and the state transition models to be linear but may be differentiable functions. The EKF linearize all the nonlinear models so that the traditional linear KF equations can be applied. EKF propagate only mean through non-linear functions, whereas, the UKF uses a deterministic sampling technique to pick a set of sample points around the mean and propagate them through the non-linear functions. This results in a filter that more accurately captures the true mean and covariance [Wik07b].

KF is effective in fusing noisy information from multiple sensors and has been used extensively for this purpose in robot navigation problems [Cro95, New05]. Leonard and Durrant-Whyte [LDW91] formulated the localization algorithm as a tracking problem. They extract naturally occurring geometric beacons from sonar data as landmarks and match them to a known navigation map to maintain an estimate of the robot location. Matthies and Shafer [MS87] apply an EKF for motion estimation. A stereo vision system is used to estimate translation and rotation between two 3D sets of points. Similarly, Kriegman et al. [KTB89] use a stereo vision system for robot navigation in indoor environments. They use vertical edges for correspondence between points in the left and right images. The robot position is tracked based on information gathered from features between successive frames as the robot moves from one point to another. The main drawback of such an approach is that without referencing to features in the global map, the uncertainty increases with the passage of time.

Iocchi and Nardi [LN02] use the HT [DH72] to extract line segments from range data. These line segments are then matched with a reference map of the environment. The matched features are used to correct the odometry error using an EKF. The features used are not distinctive in the global space and the method cannot be used to recover a lost robot. Barshan and Durrant-Whyte [BDW93] generate error models for inertial sensors (solid state gyroscope and an accelerometer) and include them in an EKF for estimating the position and orientation of a moving robot. Rencken [Ren94] attacks the bootstrap problem of navigation in unknown environment. He uses an EKF to fuse odometry and sonar data. Weckesser and Dillmann [WD96] apply an EKF for fusing information from laser range and intensity images for position estimation of mobile robots. The intensity image is obtained from a digital camera.

Vandorpe et al. [VXBA96] extract natural landmarks from 2D laser range data and fuse them with information from an inertial navigation system to deduce position of an autonomous mobile robot. Lee et al. [LSL$^+$03] aim at reducing uncertainty in the robot position by observing a moving object in image streams captured by a Charge Coupled Device (CCD) camera attached on top of a mobile robot. The trajectory of the object is assumed to be known. Information obtained from object observation is fused with the roughly known position of the robot with the help of an EKF.

Ohno et al. [OTY04] apply a Kalman smoother to fuse data from odometry, LRF and Differential Global Positioning System (DGPS) to localize a robot in an outdoor environment. A system for RoboCup robots computes depth using calibrated single camera images [MPS01]. Feature points are clustered and interpolated through a linear regression algorithm in order to minimize the average square error in different clusters. The uniquely colored regions in the robot environment are used to help in correspondence analysis. Information from features and odometry is combined using KF.

The primary disadvantage of using Gaussian for modeling robot belief is due to its unimodal nature, since only single hypothesis about position can be represented. This makes KF unsuitable for global localization in recurrent environments. Therefore, the Gaussian representation of the state space can only be done when the robot knows its initial position or it can identify features that are unique globally [BSGK07]. To solve the problem of global localization in repetitive environments, some researchers generate and track multiple hypothesis to constitute a framework for global KF [JK01, ACSS03, CL94, RB00b]. Multi-hypothesis KF represent beliefs using mixtures of Gaussians, thereby enabling them to pursue multiple, distinct hypotheses, each of which is represented by a separate Gaussian [TFBD01]. With the passage of time correct hypothesis collects more evidence and the wrong ones disappear.

2.5 Hardware for Robot Self-Localization

Range sensors (Laser scanning and sonar) are the sensors of choice for localizing mobile robots in indoor environment (rooms and corridors) [GS96, GWN01, ZRJ03, JC01, LFW96, LNHS05, GSO92, MAG$^+$02, MAES99]. Sonar is fast and cheap but crude, whereas, laser scanning is accurate but slow. These sensors are active, bulky and power consuming. These systems provide data that can be interpreted with much less computational effort but they require the environment to consist of reflecting bodies (i.e. rectangular walls) to reflect the emitted energy back to the robots. In a situation where no sufficient reflecting objects are available or the environment cannot be modified, they are of limited use. For example walls have already been removed from the middle-size league of the RoboCup [UNMK03]. Being limited in range, they cannot be expected to provide accurate localization in large open areas [HKYR97].

Approaches using single frontal cameras in conjunction with odometric sensors are widely used for self-localization. These methods are either based on calculating range and bearing based on known shape and size of landmarks or enforce special constrains on environment features [SSB03, JCBJ02, BKHS00, CRK99]. Choi et al. [CRK99] present a localization system for mobile indoor robot (mobot) using a monocular vision system. They use calibrated camera to acquire depth of features. The mobot's actual position is computed by referencing door corners over the floor edge, and heading by direction of the floor edge lines. Herrero-Pérez et al. detect features such as goal posts and corners made by the field lines. These features are treated as landmarks in a technique that uses fuzzy logic to account for errors and imprecision in visual recognition [HPMBS05a].

Single camera images provide a wealth of information about the environment but not about range, which is an important factor in position estimation. On the other hand range sensor can provide this missing information but lack in providing the information that could be extracted from camera images. This situation has lead many researcher to fuse information from range and intensity sensors. Nickerson et al. [NJW$^+$98] report a novel system called LaserEye. In situations where the robot has rectangular walls surrounding, a scan matching based technique is used for localization. Whereas, in situations when the robot is operating in large open areas the vision system is used to identify natural landmarks, distance to which is calculated using a LRF. In the latter case distance measurements between two distinct points are used for robot localization. Similarly, a laser range sensor is used to estimate depth of an artificial landmark once it is detected in camera images [LST02]. Arras and Tomatis [AT99] use LRF and monocular vision for indoor localization.

Similarly, Clerentin et al. [CDPB00] report a perception system for robot self-localization that is composed of an omni-directional vision system and a panoramic range finder. The Omni-directional vision system is used to measure the angle between different vertical objects. However, to calculate depth of these vertical edges, the vision system is associated with range finding sensor. The angle and depth measurements augment the position estimation process. Additional approaches that use a combination of LRF with single frontal cameras and omni-directional cameras can be found in [AT99, ATJS01, ATJS01, WFJ$^+$01, WFJP02]. Whereas, fusion of vision and ultrasonic sensors is reported in [Wic98, RHD$^+$02, OKK98].

There are several approaches using omni-directional cameras [MZPI04, SB05, JIPB03, BAK04]. A camera looking at a mirror of special shape provides the robot with visual information in all directions simultaneously [ML01]. The major advantage of these approaches is that the robot has a panoramic view of its environment and consequently can acquire more landmark features. For example, Marques and Lima [ML01] detect field lines using the HT [DH72] and correlate them with the field model to estimate the robot position. The omni-directional camera assembled on their robot is shown in Fig. 2.3. Wolf and Pinz [WP03] are using the same panoramic imaging integrated in a Monte Carlo framework for self-localization. This method can only track the robot position and cannot localize the robot from scratch.

Ji et al. [JIPB03] report a triangulation based method for localization of soccer robots. Landmarks are extracted with the help of an omni-directional vision system. Wolf and Pinz integrate features extracted by an omni-vision system in a Monte Carlo framework for self-localization [WP03]. Similarly, a combination of robust illumination insensitive eigenspace approach and sensor fusion with odometry data is used as a solution for the self-localization [SB05]. In [TRM$^+$05] odometry is used to calculate the expected position of landmarks and then a local search algorithm finds their exact position. Whereas, Motomura et al. localize their robots using dead-reckoning and angle measurements between two landmarks [MMH04].

Because of the high cost, noise, requirement of sufficient reflecting surfaces, and modification requirement of the environment there has been a decreasing importance of using active sensors in the soccer robots [Nov03]. Although there are approaches to calculate range from single camera images [BKHS00, CRK99], it is erroneous and cannot be calculated all the

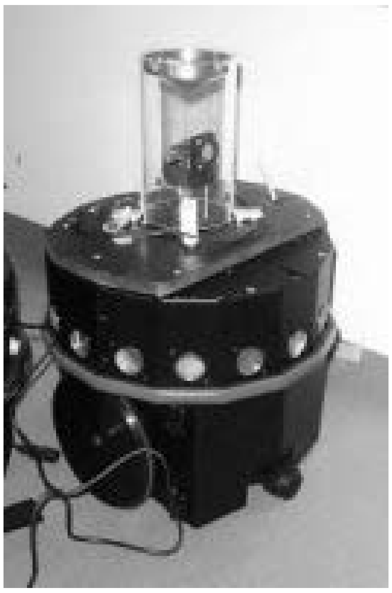


Figure 2.3: Mobile robot with omni-directional vision system [ML01]

time [NJW+98]. The disadvantage of omni-vision systems is the low resolution of the images and requirement to fit the mirror at a specific height so that the robot can "see" its surroundings. Another disadvantage is the high cost of the mirror. Keeping these problems in mind stereo vision seems to be the only sensor for the purpose of localization in open (wall free) environments.

For feature detection and range estimation a pivoted stereo vision system is used. This approach enables the robot to measure the distance to landmarks and to use bi/trilateration approach to calculate robot position. The pivoted camera head enables the robot to have a 360 °view of its environment. The robot could acquire additional information from the environment by rotating its head. Together with the odometric sensors and the world model it should approximate a soccer player such that its own position and its distances to other objects should be computed automatically and that the robot is able to interact with the scene.

2.6 Miscellaneous Methods of Localization

Methods that that do not fit in any of the categories discussed before but are related to this work are presented in this section.

2.6.1 Collaborative Multi-Robot Localization

When teams of robots localize themselves in the same environment, they may synchronize each others belief whenever one robot detects another. This helps robots to localize themselves faster and maintain higher accuracy [FBKT00, FBKT99]. Additionally, high cost sensors may be distributed across multiple robot platforms. Hanek and Schmitt [HS00] report experimental results of using the collaborative multi-robot localization applied to localize robots in the Robocup scenario. Other examples for the Robocup are reported in [RB02, SHB$^+$02]. Similarly, two robots equipped with a stereo vision system are able to localize with respect to each other in order to have a common reference frame and can work on a common task. These robots explore their environment autonomously and build occupancy grid maps of it [JML99].

2.6.2 Global Vision

Yet another approach is the so-called global vision that refers to the use of cameras placed at fixed locations in a workspace to extend the local sensing available on board each robot [KL92, KL93]. In global vision methods, characteristic points forming a pattern on the mobile robot are identified and localized from a single view. A probabilistic method is used to select the most probable matching according to geometric characteristics of those percepts.

Robots in the RoboCup small size league (F180) [Rob07b] are marked with special markers [BWN04, DBBD03]. These markers are tracked and localized in images taken by a global camera as shown in Fig. 2.4 [BV03, TNS03]. Similarly, the FIRA Micro Robot World Cup Soccer Tournament (MiroSot) robots (see Fig. 2.5) are marked with special color patterns so that they can be localized with the help of images taken by a global camera [FIR07b]. The camera setup for FIRA MiroSot is shown in Fig. 2.6 [FIR07a].

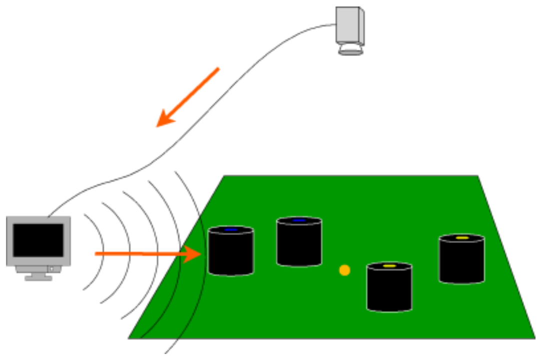


Figure 2.4: Camera setup for RoboCup small size league (F180) [Rob07a]

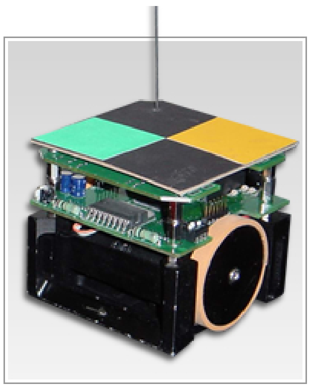


Figure 2.5: FIRA MiroSot robot color markings [FIR07b]

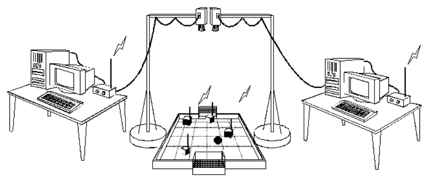


Figure 2.6: Camera setup for FIRA MiroSot [FIR07a]

2.6.3 Mosaic-Based Localization

The success of mosaic based localization is primarily due to recent advances of high speed processors, fast cameras, and large storage media [SN04]. A system, reported in [Kel00], uses a CCD camera [Wik07a] which is fitted to the robot and is pointed towards the floor. The floor is illuminated by a specialized light pattern off the camera axis to enhance floor texture. The robot begins by collecting images of the entire floor in the robot's workspace using this camera.

Once the complete image mosaic is stored, the robot can travel any trajectory on the floor while tracking its own position without difficulty. Localization is performed by simply recording one image, performing action update, then performing perception update by matching the image to the mosaic database using simple techniques based on image database matching. The main drawback of this method is the requirement of micro-fractures on the floor that generate sufficient texture for correlation [SN04].

Dellaert et al. [DBFT99] generate an image mosaic of the ceiling of the environment in which the robot needs to operate, and use it during operation to localize the robot. The distribution of light intensity is successfully utilized to localize the indoor robot MINERVA. If the light structure is uniform then such an approach cannot be used [UNMK03].

2.6.4 Position Estimation Based on Camera Calibration Techniques

Research into estimating internal and external camera parameters is quite mature [Zol03]. These methods have been used to localize robots equipped with cameras. Quan and Lan [QL99] present a camera localization method that requires 5-point correspondences. This method can be generalized to N-points. Stella and Distante [SD95] report location estimation of the center of projection (CP) of the camera and orientation of optical axis using 3 landmarks. Similarly, Liu et al. [LHF90] discuss camera location estimation using line correspondences. They discuss location estimation using linear and non-linear algorithms. Eight or more line correspondences are required for the linear algorithm, whereas, three or more line correspondences are required by the non-linear algorithm. First the orientation of the robot is calculated which is then used to calculate the position.

2.7 Discussion

A review of robot self-localization was presented in this chapter. The scope of the review was limited to approaches that are applicable in known or partially known environments.

Approaches that address the localization problem in unknown environments were largely considered as out of scope.

Due to the requirement of global self-localization it is not possible to use methods that are only based on local sensors. The environment cannot be modified with navigational aids, which rules out the possibility of using approaches that use artificial landmarks or active beacons. The requirements to be able to work in a wall free environment and low power consumption mean that the robot has to rely on vision sensor. Another reason of using vision is as stated by Lamon et al. [LNJS01] that ”Simple ranging devices require integration over time and high-level reasoning to accomplish localization. In contrast, vision has the potential to provide enough information to uniquely identify the robot’s position”. The constraint on the maximum size of the robot does not allow the use of an omni-directional vision system. Furthermore, it is undesirable to enforce constraints on feature locations in the environment. Features are supposed to be scarce and it is required that the localization is based on as few features as possible.

Methods that are based on global vision cannot be applied as all sensing and processing has to be onboard. The maximum size of the robot and the processing power is limited. Among the existing approaches related ideas in landmark extraction, feature based global self-localization and probabilistic position estimation could be used in this work. The differences between existing approaches and the proposed one is the subject of Chapter 3, Chapter 4 and Chapter 5. To the best of our knowledge there is no existing method that can address the problem in totality.

Chapter 3

Vision Based Feature Extraction

The robot environment consists of features such as line segments, corners, junctions and line intersections. Additionally, there could be distinct color patches and the transition between different colors can be used as landmarks. Straight lines are strong candidates to be used as landmarks since lines are determined by a large number of pixels which makes it possible to locate them accurately. Line features can be detected even if they are partially occluded, which makes lines an effective and reliable image feature. Lines are natural in the sense that a number of prominent lines can be found in indoor environments [DYOH03, DYOC04].

The following chapter presents feature extraction for self-localization using the stereo vision system. After reviewing the related work done in this field, the stereo vision system will be introduced.

3.1 Related Work

Currently, soccer robots of the size of Tinyphoon are marked on their top with some color patterns, which are then tracked for position estimation using a global camera and a host computer [BWN04, DBBD03]. This work aims at a shift towards complete autonomy, where all sensing and processing is done by the onboard sensors. However, feature extraction techniques used for localization of other (bigger and slower) soccer robots and also of indoor mobile robots with methods that could be applied to this domain are reviewed briefly.

Gutmann and Schlegel [GS96] use LRF mounted on top of the robot with a $360\,^{\circ}$viewing angle to obtain dense range scans of the surrounding walls in the environment. The robot position is estimated by matching a new scan to a reference map of scans. Dead reckoning information is combined with the measurements from the scan matcher with a KF. Similarly, Iocchi and Nardi [LN02] use the HT [DH72] to extract line segments from range data. These line segments are then matched with the reference map of the environment. This match is then finally integrated with odometry information by means of an EKF. The drawback of using a range sensor is that it requires the environment to be surrounded by walls. In a more natural wall free environment, vision seems to be the only effective sensor [UNMK03].

Vision based approaches reported in [ERF^{+}00, MMH04, SSB03] are based on detecting color marked features such as goal posts and corners. In such context, localization depends highly on robust recognition of color markings that have to be explored around the field. For robots with common cameras, searching these marks distracts them from concentrating on game objects [Hue04].

Adorni et al. [ACM99] use two wide angle cameras for localization of their goal keeper robot. The robot uses the white lines of the penalty area as landmarks which are detected by applying the HT to those image pixels that are segmented as white (after a color segmentation) and belong to an edge. Lines extracted from both images are symmetric if the robot is at its default position. The analysis of the asymmetry of the two views make it possible to estimate the robot's absolute position. The very specific details of the method make it difficult to be adopted by other robots and furthermore it is hard for the same robot to localize itself in other parts of the field.

Utz et al. [UNMK03] detect image pixels belonging to field markings on the basis of color transition, which are then transformed into the Hough parameter space. Filtering feature points that may belong to line segments on the basis of color transitions makes the method dependent on color segmentation.

Jong et al. [JCBJ02] divide the edge image (edges are marked as color transitions in the color segmented image) into small sub-images. Lines in each sub-image are detected by applying the HT and selecting the best two peaks from each sub-image. This gives a rough estimate of the lines. Next, a sub-pixel edge detector is run on the red component of the RGB image perpendicular to the lines found with the HT. The red component is chosen because it gives the best contrast between green and white. The edge pixels are then converted to calibrated camera coordinates and lines are fitted through these points (converted edge pixels) with the least square method.

Huebner [Hue04] detects pixels belonging to field lines based on their symmetry in the image. These pixels are then grouped into line segments using local operations. This method can only detect field markings and can not scale to detect other features such as the field boundary.

Bandlow et al. [BKHS00] detect the field boundary area by a dilatation operation applied on both the wall and the field region followed by an intersection of the two resulting regions. They use two different approaches; the first approach transforms the skeleton of the boundary area into contours, whereas, in the second method a sub-pixel accurate edge filter is applied to the y-channel within the boundary area. These edges are then transformed into a set of contours. Contours are mapped into the camera coordinate system to be used for position estimation of the robot.

Herrero-Pérez et al. [HPMBS05a] detect corner features made by field lines. Corners are detected using a method from Sojka [Soj02] which is based on the variance of the gradient of the brightness. Corners produced by this method are then filtered depending on whether they come from a white line segment or not.

Christensen et al. [CKKK94] use the current position estimate of the robot to make projections from the known environment corresponding to a prediction of what a camera is expected to see from a given view-point. These projections are then used for matching against line segments extracted from images. Line segments in the image are detected by using least square optimization of edge points along the predicted line. Such method would only work if the current position estimate is close to the actual one.

Marques et al. [ML01] use an omni-directional camera to get 360°view of the field. They apply the HT to detect field lines and then correlate them with the field model to deduce the robot position. The drawback of such a setup is the low resolution of the camera [UNMK03] and requirement for the extra-space to fit the camera and mirror.

Different local line detection schemes [BHR86, GSN04, CB03] together with variants of the HT such as Randomized Hough Transform (RHT) [XOK90], Connective Randomized Hough Transform (CRHT) [KH97], Extended Connective Randomized Hough Transform (ECRHT) [KK00], Probabilistic Hough Transform (PHT) [KEB91] (see [Lea93] and [KHXO95]) were reviewed. The probabilistic HT methods are the recent developments but have the drawback that they require some stopping criteria, which is utilized to stop the random sampling of pixel pairs. The stopping criterion could be a specified number of pixel pairs. When the number of sampled pixel pairs reaches the specified number, the process would stop. In a practical implementation, a suitable stopping criterion is selected by the users through the tests of different numbers of pixel pairs [CG99].

The main difference between the proposed method and previous methods lies in detection of line segments and their classification if they belong to field markings. Corners, junctions and line intersections are detected using semantic interpretations of these segments. These features are then put into a stereo algorithm to compute their three dimensional position, which is finally used in estimation of the robot position and orientation.

3.2 The Stereo Vision System

In the following working of the stereo vision system is discussed. Using the notations and coordinate systems introduced in Appendix A, the pose of a robot moving on a flat surface has three degrees of freedom as $\mathbf{p} = \begin{bmatrix} x & y & \theta \end{bmatrix}^T$. The robot pose is estimated by finding a transformation between landmark features in robot and world coordinate systems. For detection of landmark features, the robot is equipped with a stereo vision system. Fig. 3.1 shows a simplified construction of the robot vision system. In this figure, the rotation angle (γ) of the robot head and tilt of the vision system (β) are set to zero, which result in alignment of C0, C1, and C2. C0 and C1 are overlapping, whereas, C2 is translated along the x-axis. L and R represent the origins of the left and right cameras coordinate systems, respectively. Image pixels are denoted by (u_l, v_l) and (u_r, v_r), where the subscript l and r refer to the left and right image. The u and v axes of the image plane are in opposite direction to y^{C2} and z^{C2} axes respectively. The separation between the two cameras (the stereo baseline) is denoted by b and f is the focal length. Assuming identical cameras, parallel image planes and aligned epipolar lines, a point $\mathbf{p}^{\mathrm{C2}} = \begin{bmatrix} x^{\mathrm{C2}} & y^{\mathrm{C2}} & z^{\mathrm{C2}} \end{bmatrix}^T$ in C2 and its projections $\begin{bmatrix} u_l & v_l \end{bmatrix}^T$ and $\begin{bmatrix} u_r & v_r \end{bmatrix}^T$ in the left and right image can be related under perspective transformation [TV98]. To illustrate the relationship between $\mathbf{p}^{\mathrm{C2}}$ and its projections on the image planes, the similar triangles of Fig. 3.1 are redrawn in Fig. 3.2. From $\Delta \mathrm{A}_3\mathrm{A}_4\mathrm{A}_5$ and $\Delta \mathrm{A}_1\mathrm{A}_2\mathrm{A}_4$ as shown in Fig. 3.2, (3.1) can be obtained by equating ratios of the corresponding sides. Similarly, (3.2) and (3.3) result from equating side ratios from $\Delta \mathrm{B}_3\mathrm{B}_4\mathrm{B}_5$ and $\Delta \mathrm{B}_1\mathrm{B}_2\mathrm{B}_4$ and $\Delta \mathrm{C}_3\mathrm{C}_4\mathrm{C}_5$ and

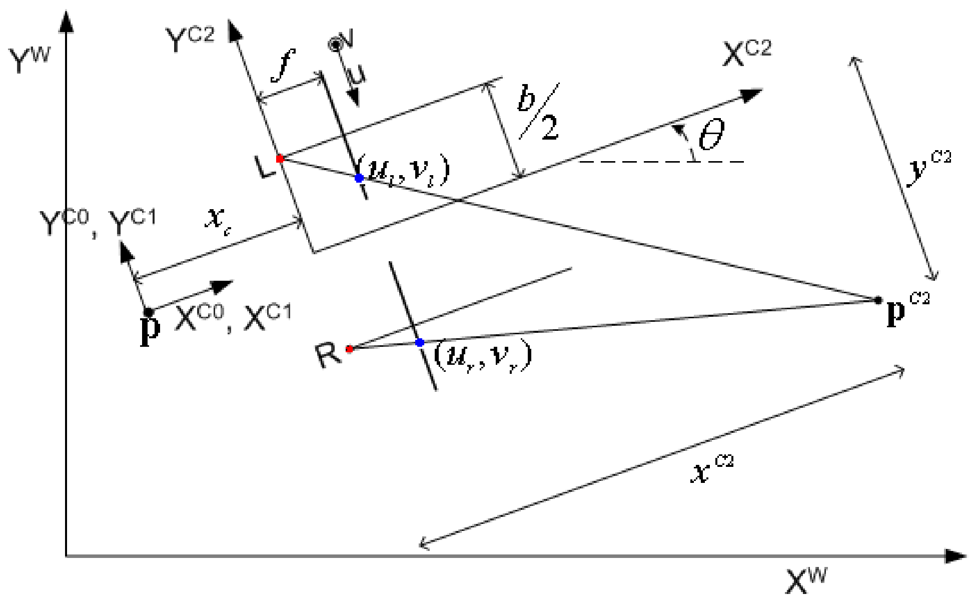


Figure 3.1: Geometric construction of the robot's stereo vision system

$\Delta C_1C_2C_4$, respectively.

$$\frac{f}{u_l - o_u} = \frac{x^{C2}}{y^{C2} + \frac{b}{2}} \tag{3.1}$$

$$\frac{f}{u_r - o_u} = \frac{x^{C2}}{y^{C2} - \frac{b}{2}} \tag{3.2}$$

$$\frac{f}{v_r - o_v} = \frac{x^{C2}}{z^{C2}} \tag{3.3}$$

Solution of (3.1), (3.2) and (3.3) results in (3.4), which is given as follows:

$$\mathbf{p}^{C2} = \begin{bmatrix} x^{C2} \\ y^{C2} \\ z^{C2} \end{bmatrix} = \mathbf{f}\left(\begin{bmatrix} u_l \\ v_l \end{bmatrix}, \begin{bmatrix} u_r \\ v_r \end{bmatrix}\right) = \begin{bmatrix} \frac{fb}{u_l - u_r} \\ \frac{-b}{2}\frac{u_l+u_r-2o_u}{u_l-u_r} \\ \frac{-b(v_r-o_v)}{u_l-u_r} \end{bmatrix} \tag{3.4}$$

where $[o_u \ o_v]^T$ is the image center. Using (A.4), the $\mathbf{p}^{C2}$ is transformed from C2 into C0 as follows:

$$\mathbf{p}^{C0} = \begin{bmatrix} x^{C0} \\ y^{C0} \\ z^{C0} \end{bmatrix} = \begin{bmatrix} \cos\beta & 0 & -\sin\beta \\ 0 & 1 & 0 \\ \sin\beta & 0 & \cos\beta \end{bmatrix} \left(\begin{bmatrix} \cos\gamma & \sin\gamma & 0 \\ -\sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x^{C2} \\ y^{C2} \\ z^{C2} \end{bmatrix} + \begin{bmatrix} x_c \\ 0 \\ 0 \end{bmatrix} \right) \tag{3.5}$$

where x_c is translation of the origin of C2 along x-axis, γ is the rotation angle of the robot head and β is the tilt of the vision system about y-axis.

The values x^{C0} and y^{C0} with respect to a landmark $\mathbf{p}_l = [x_l \ y_l]^T$ are used for calculating range r and angle ψ as illustrated in Fig. 3.3 and given by (3.6) and (3.7).

$$r = \sqrt{(x^{C0})^2 + (y^{C0})^2} \tag{3.6}$$

$$\psi = atan2(y^{C0}, x^{C0}) \tag{3.7}$$

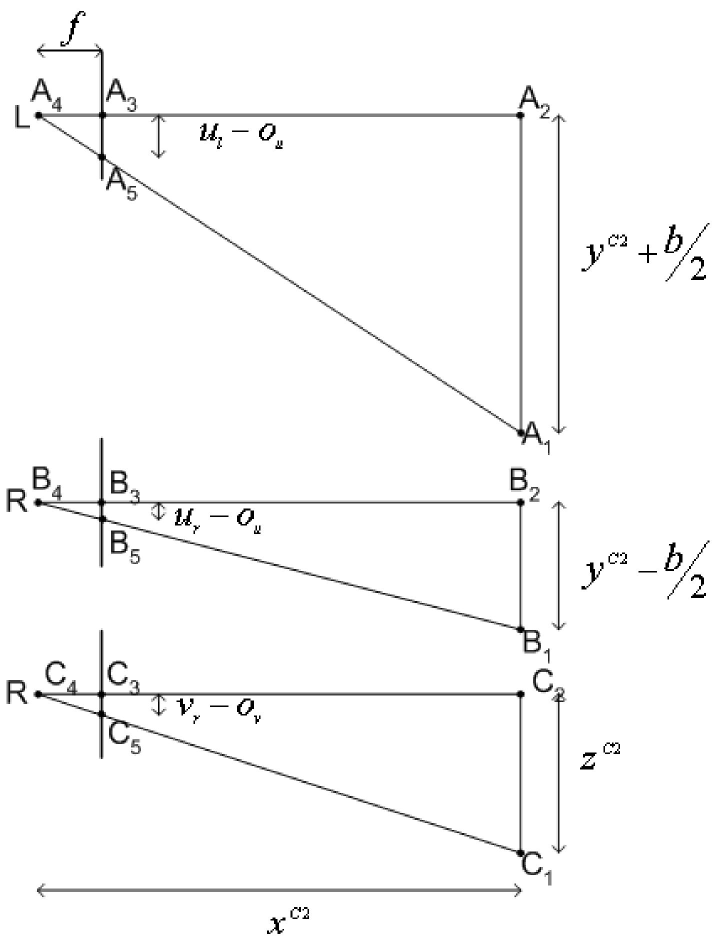


Figure 3.2: Similar triangles from Fig. 3.1 illustrating relationship between $\mathbf{p}^{C2}$ and its projections on the image planes

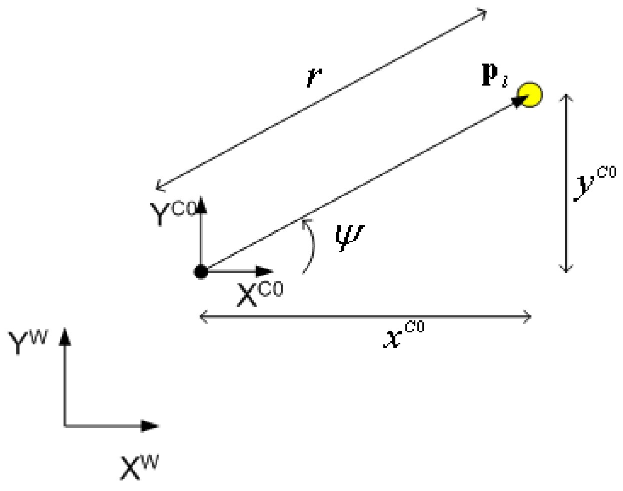


Figure 3.3: Illustration of the robot range and angle measurements with respect to a landmark

3.3 Detecting Line Features

The primary target is to extract the global structure of the line segments, in the presence of heavy occlusions. Local methods were dropped as they could not handle a high level of occlusion. The GBHT is selected for its relative robustness and soundness against noise especially occlusion and the need to extract a global line structure [MA98].

3.3.1 Line Detection with Hough Transform

The HT is an effective and robust method for finding straight lines fitting 2D points in images. When introduced for the first time the slope-intercept representation of line (3.8) was used [Hou62]. This representation has the disadvantage that both the slope (m) and intercept (c) tend towards infinity as a line approaches the vertical. Duda and Hart [DH72] suggested a parametrization of the straight line (3.9), which says that any point (x,y) can be represented by the length (ρ) and angle (θ) of the normal vector to the line passing through this point from the center of the coordinate system as shown in Fig. 3.4.

$$y = mx + c \tag{3.8}$$
$$\rho(\theta) = x\cos\theta + y\sin\theta \tag{3.9}$$

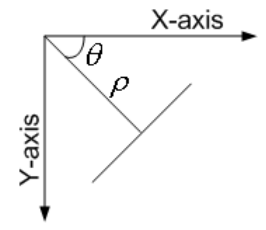


Figure 3.4: Line parametrization with ρ and θ

According to (3.9) each point (x, y) in the image corresponds to a curve, of period 2π, in the parameter space. Curves corresponding to collinear points of a line with parameters (ρ_1, θ_1) will cross each other at point (ρ_1, θ_1) in the parameter space [Imm98]. It is clear from (3.9) that $\rho(\theta) = -\rho(\theta + \pi)$. This means that the parameter space with $\theta \geq \pi$ is redundant. The parameter subspace with $0 \leq \theta < \pi$ is non-redundant and is sufficient to construct the entire two-dimensional parameter space [Imm98].

Both θ and ρ are continuous variables which are quantized with step sizes $\delta\theta$ and $\delta\rho$, respectively. This quantization results in a two-dimensional accumulator array, A. The size of A, $N_\theta \times N_\rho$, for the parameter subspace is given as follows [Imm98]:

$$N_\theta = \frac{\pi}{\delta\theta} \tag{3.10}$$
$$N_\rho = \frac{\rho_{max} - \rho_{min}}{\delta\rho} \tag{3.11}$$

where ρ_{min} and ρ_{max} are the minimum and maximum values of ρ for any combination of x, y, and θ. At the start of the process the accumulator array is initialized to all zeros. Then for each image point (x, y), the values of $\rho(\theta)$ are calculated using (3.9) for $0 \leq \theta < \pi$ sampled at every $\delta\theta$. For a certain sampled value of θ, say θ_i, the resulting value of ρ is rounded to ρ_i.

Each cell in the accumulator array counts the number of sinusoidal curves passing through the corresponding area (of size $\delta\theta \times \delta\rho$) in the parameter space. From the global perspective, an image pixel could be part of infinite number of lines which means that (3.9) will be calculated for all values of θ ranging from 0 to π (excluding π). However, image points that are supposed to be transformed into the parameter space, are the output of an edge detector which also gives orientation of the edge passing though that point. This information about the orientation reduces this range considerably. Hence, it is possible to calculate $\rho(\theta)$ for those values of θ which are close to the direction of edge normal, ϕ [KBS75]. This closeness could be determined by a given tolerance.

The use of gradient information is illustrated in Fig. 3.5. A polygonal shaped object is shown in the gray scale image of Fig. 3.5(a). The gradient information can be computed by passing the gray scale image through a sobel edge detector [GW02]. The gradient magnitude and orientation is shown in Fig. 3.5(b) and Fig. 3.5(c). Edge pixels that are transformed into the Hough parameter space are shown by the white lines in Fig. 3.5(b). In Fig. 3.5(c) the values are from -180 to 180, with low values shown black. Negative values are transformed into the range 0 to π before they are used in generation of the parameter space. Fig. 3.5(d) shows the parameter space being generated with the whole range of θ values, whereas Fig. 3.5(e) shows the case where the parameter space is generated using only values of θ that are close to ϕ. In this particular case the tolerance is within $\pm 15^\circ$ of the transformed value of ϕ. Apart from the obvious gain in speed this is a major aid in finding peaks in the parameter space as the interference from other lines is reduced.

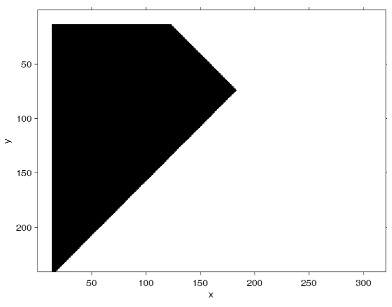
3.3.2 Peak Detection

A single point in the image corresponds to a curve in the parameter space. Curves corresponding to multiple collinear points cross each other at one single point. From this perspective, peak detection could be seen as the inverse of the HT as peaks in the accumulator array represent collinear points in the image which may belong to lines in the image. Peak detection is accomplished by searching through the accumulator array and choosing cells that meet certain criteria. For peak detection, an iterative process is implemented, which proceeds as outlined in Algorithm 1 [GWE03].

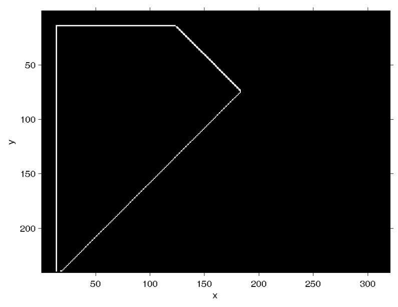
Algorithm 1 PeakDetection(H, n, $threshold$)

```
$Peaks \Leftarrow 0$
for $i = 1$ to $n$ do
   $p \Leftarrow \mathbf{max}(H)$
   if $p.value > threshold$ then
      add $p$ to $Peaks$
      clear neighborhood of $p$ in $H$
   else
      return $Peaks$
   end if
end for
return $Peaks$
```

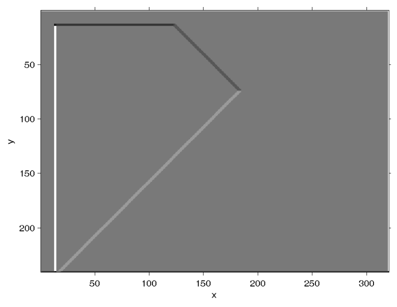
In Algorithm 1, H denotes the parameter space, n is the number of line segments sought and *threshold* is the threshold used in peak detection. The routine **max** returns a structure



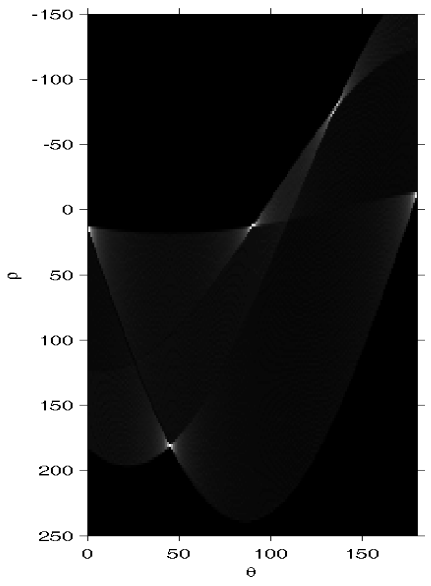
(a) Gray scale image



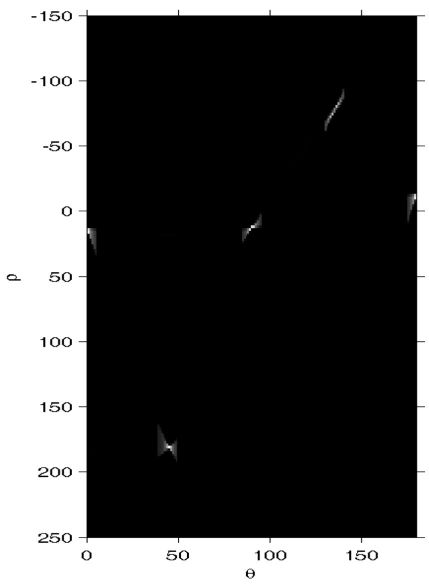
(b) Edge gradient magnitude



(c) Edge gradient orientation



(d) Without using gradient orientation



(e) Parameter space generated taking gradient orientation into consideration

Figure 3.5: Using edge orientation for calculating HT

containing *value*, ρ, and θ corresponding to the maximum value in H. The algorithm returns *Peaks*, which is a list of $(\theta, \rho, value)$ values. Peaks in the parameter subspace represent potential line segments in the image. Peaks could be spurious. This is a simple way of peak detection and better results could be achieved by filtering (e.g. the butterfly filter [LB87]) the parameter space before searching for peaks. The *threshold* is given as input, however, it can be specified as some percentage of the highest value in the accumulator array. The predefined neighborhood is a rectangular area around the peak. By clearing a window around the peak, the risk of selecting spurious values close to this peak is reduced. The size of the window is critical and is difficult to determine [Ågr03]. A too small size will leave spurious peaks, whereas, a too high value will risk eliminating a true peak.

The next section discusses verification of peaks detected and also completes the line description. The parameters (θ, ρ), length l and the coordinates of the end points (x_1, y_1) and (x_2, y_2) of a straight line constitute the complete line segment description [AA94].

3.3.3 Peak Verification and Completing Line Segment Description

After peaks are detected in the parameter space, they are tested if they represent true lines. Input to this process is the list of all feature points (FP), a list containing the detected peaks ($Peaks$) and the minimum length of a segment (L_{min}). Pixels in one group are separated into multiple line segments if their separation is greater than Gap_{min}, whereas, two line segments from a group are merged if the separation between them is less than Gap_{max}. The parameter list $Peaks$ is sorted in descending order. Verification of peaks is outlined in Algorithm 2 [BSN05].

For each peak, the loop listing from Line 6 to Line 13 of Algorithm 2, iterates through the whole list of edge pixels to find pixels that might have resulted in that peak. These pixels are added to F if their edge orientation is in compliance with θ (see Line 8). If the number of pixels corresponding to this peak are less than the minimum length, the peak is declared spurious.

After a successful minimum line length test, all pixels in F are rotated by Ω in Line 20. This is required to find gaps between pixels since the two dimensional problem of finding separation between pixels is transformed into a one dimensional problem. The rotated pixels have same y-value, which is equal to ρ. Pixels in the main group F are divided into multiple sub-groups if the separation between x-values of the rotated pixels is more than Gap_{min}. Each sub-group of pixels is tested against L_{min}. Line segments in different sub-groups are merged if there separation is less than Gap_{max}. Rotation is in clockwise direction for positive values of Ω. The process of rotating pixels is illustrated in Fig. 3.6. The actual line segments are shown by thin solid lines, whereas, their rotated versions are shown by thick solid lines. The dotted lines show any required extension of the line segments to meet the normal. Fig. 3.6(a) shows the case for positive value of ρ and $\theta < 90\,^\circ$. Similarly, the case for $\theta > 90\,^\circ$ is shown in Fig. 3.6(c). Here Ω is negative and the rotation is in counter clockwise direction. Finally, Fig. 3.6(e) shows rotation of a line segment with negative value of ρ and $\theta > 90\,^\circ$.

The loop from Line 23 to Line 36 divides pixels from the same group into multiple line segments if separation between the rotated x-values is greater than Gap_{min}. For each segment the minimum length is tested in order to remove pixels that could have accidentally aligned with other pixels. The code from Line 38 to Line 49 merges two line segments if their separation is less than Gap_{max}. If two line segments are merged then length of the new line segment is the length of the two segments plus the gap between them. Removing pixels from the main list of feature points (see Line 50) helps accelerate the process and reduces the possible interference that they may cause in the verification of other line segments [Dav92].

The above method of line verification and calculation of additional parameters such as line length and end points is iterative in nature. There are other state of the art non-iterative approaches that are based on the fact that the spread of votes in the accumulator array is dependent on the length and position of line segments. Akhtar and Atiquzzaman [AA92] determine the extent of the spread of votes in any column of the accumulator array to calculate the length of a single line segment in the image. The limitation of this algorithm is its inability to determine the end points of the line segment. Atiquzzaman and Akhtar [AA94] calculate the complete description of single line segment in an image. They select some arbitrary column in the accumulator array at a known distance from the detected peak. This column is then scanned for the first and last non-zero accumulator cells on which the calculation of end points of the line segment is based. This method can successfully calculate the end points

Algorithm 2 PeakVerification(FP, $Peaks$, L_{min}, Gap_{min}, Gap_{max})

```
for $i = 1$ to $Peaks.length$ do
    $F \Leftarrow 0$
    $\rho \Leftarrow Peaks(i).\rho$, $\theta \Leftarrow Peaks(i).\theta$
    $c \Leftarrow \cos\theta$, $s \Leftarrow \sin\theta$
    // finding pixels that might have resulted in this peak
    for $j = 1$ to $PF.length$ do
        if $PF(j).x \times c + PF(j).y \times s == \rho$ then
            if $PF(j).\phi - \theta \approx 0$ then
                add $PF(j)$ to $F$
                add connected pixels of $PF(j)$ to $F$ if orientation in compliance with $\theta$
            end if
        end if
    end for
    // test if this peak is spurious
    if $F.length < L_{min}$ then
        continue
    end if
    // all pixel in $F$ are rotated by $\Omega = 90^\circ - \theta$.
    // after rotation all pixels lie parallel to x-axis as shown in Fig. 3.6
    $F' \Leftarrow$ rotate $(F, \Omega)$
    $F' \Leftarrow$ sort $(F')$
    $L' \Leftarrow 0$, $l \Leftarrow 0$, $l.x_1 \Leftarrow F'(1).x$, $l.y_1 \Leftarrow F'(1).y$
    for $k = 2$ to $F'.length - 1$ do
        // pixels with the same $\rho$ and $\theta$ are divided into multiple line segments if gap between consecutive
        pixels is greater than $Gap_{min}$
        if $F'(k).x - F'(k-1).x > Gap_{min}$ then
            $l.x_2 \Leftarrow F'(k-1).x$
            $l.y_2 \Leftarrow F'(k-1).y$
            $l.length \Leftarrow l.x_2 - l.x_1$
            if $l.length \geq L_{min}$ then
                add $l$ to $L'$
            end if
            $l \Leftarrow 0$
            $l.x_1 \Leftarrow F'(k).x$
            $l.y_1 \Leftarrow F'(k).y$
        end if
    end for
    // line segments are merged if their separation is less than $Gap_{max}$
    for $k = 1$ to $L'.length - 1$ do
        if $L'(k+1).x_1 - F'(k).x_2 < Gap_{max}$ then
            $L'(k+1).x_1 \Leftarrow L'(k).x_1$
            $L'(k+1).length \Leftarrow L'(k+1).x_2 - L'(k).x_1$
        else
            $l \Leftarrow$ rotate $(L'(k), -\Omega)$
            add $l$ to $L$
        end if
    end for
    // rotation by $-\Omega$ brings the lines to their original position
    $l \Leftarrow$ rotate $(L'(L'.length), -\Omega)$
    add $l$ to $L$
    remove $F$ from $PF$
end for
return $L$
```

and consequently the length if there is only one line segment in the image. A more detailed analysis of the spreading of votes around a peak (formation of the *butterfly*) is reported by

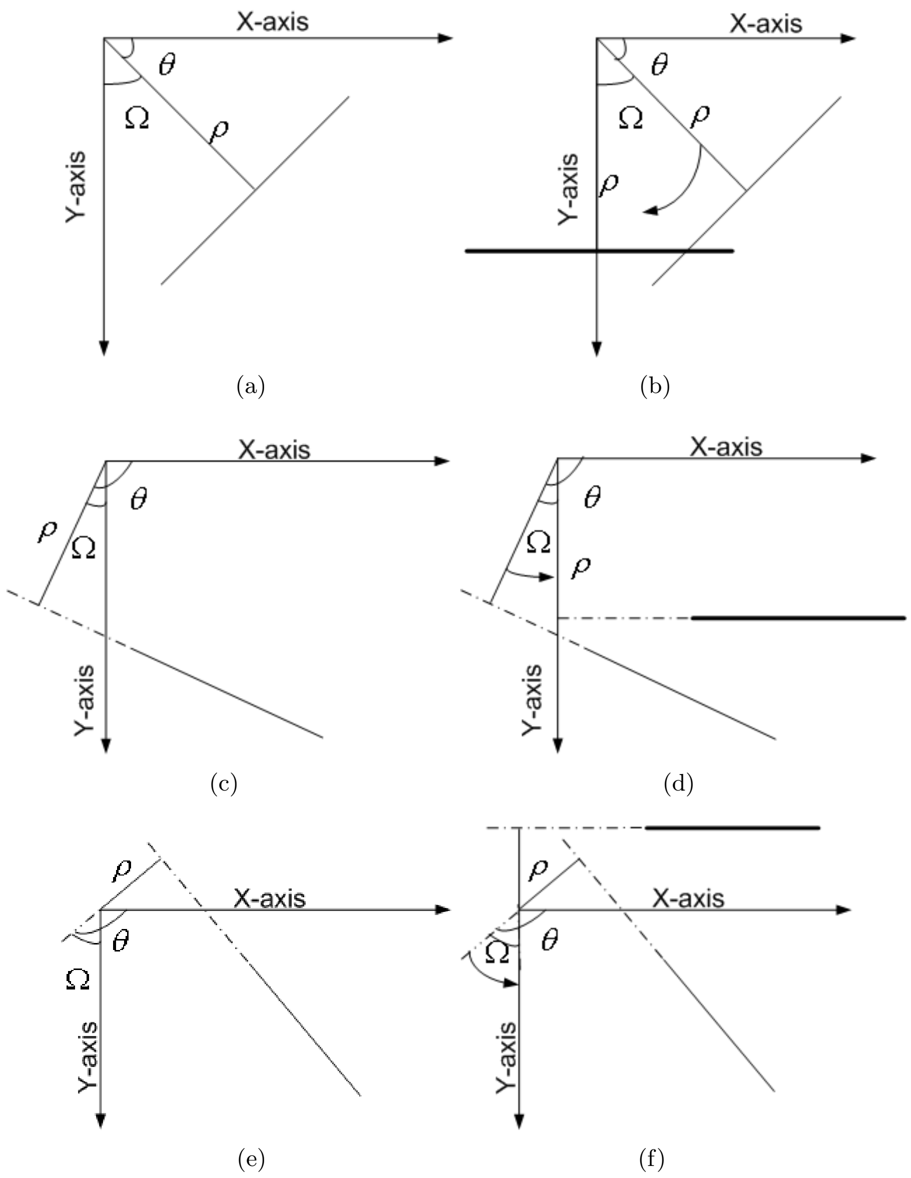


Figure 3.6: Rotating line segments for completing the line segment description

Kamat-Sadekar and Ganesan [KSG98]. This method is an extension of [AA94] to determine a complete line description of multiple line segments. They select two columns on each side of the detected peak. These columns are scanned from $\rho_{min}(\theta)$ to $\rho_{max}(\theta)$, which represent the limits on ρ for a given θ for an image consisting entirely of feature points.

The non-iterative method of finding end points and line length is computationally efficient as compared to the iterative method [AA94, KSG98]. However, real world images are too complex and result in distorted *butterflies* around peaks in the parameter space.

3.4 Detecting Field Markings

Classification of line segments detected in an image is discussed in this section. All line segments extracted are tested if they are field markings or belong to other objects. Field markings are white lines/arcs drawn on a dark surface. In terms of gray scale gradient, such markings can be seen as a sequence of a negative gradient followed by an opposite positive gradient of equal magnitude or vice versa as shown in Fig. 3.7. A synthetic image of the robot environment is shown in Fig. 3.7(a). Three line segments in this image can be classified as field markings. Passing the y-channel of this image through a sobel edge detector result in gradient magnitude and orientation as shown in Fig. 3.7(b) and Fig. 3.7(c), respectively [GW02]. The resulting dual edge features can be seen from the gradient magnitude and orientation images. The yellow and white walls also result in such edge points but they can be differentiated based on their separation. In Fig. 3.7(c) the values are from -180 °to 180 °, with low values shown in black.

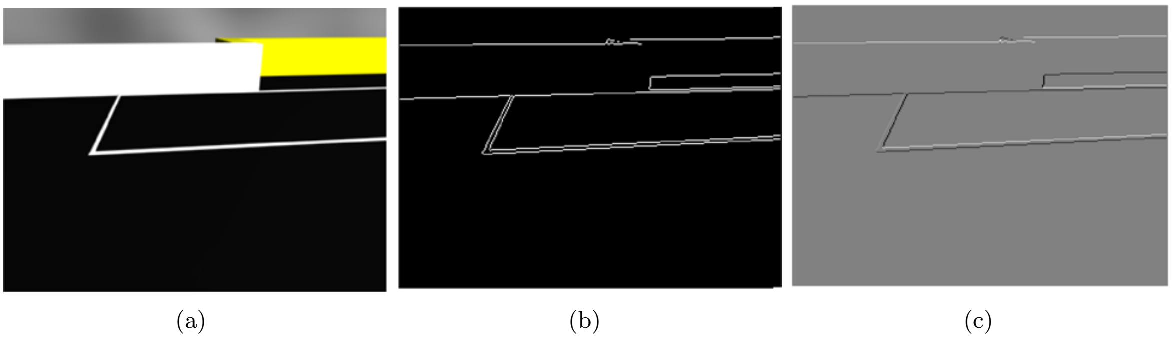


Figure 3.7: Differentiating between field marking and other straight lines

Classification of field marking is carried out after a successful minimum length test at Line 15 of Algorithm 2 as discussed in Section 3.3.3. The algorithm is outlined in Algorithm 3.4 and illustrated in Fig. 3.8. Input to the process of line classification is gradient magnitude G_m, gradient orientation G_o, maximum width of the line segment W' and a list L containing pixels of the line l to be classified. The value of W' depends on the maximum allowed width of a field marking in terms of image pixels.

As the classification is based on the existence of nearly parallel line segments of opposite gradient orientation, search direction is perpendicular to the line. There are two factors which help in deciding if a line segment should be classified as field marking or not. The algorithm tests a certain percentage of total pixels, say ξ_h%. However, a line segment is classified as field marking, if ξ_l% of the total pixels tested have dual edges. The requirement is $0 < \xi_l \leq \xi_h$ and $\xi_l \leq \xi_h \leq 1$. The values of ξ_l and ξ_h are determined empirically.

Fig. 3.8 illustrates the process of classification of a pixel. For a pixel in L, the inner loop (starting at Line 5) of Algorithm 3.4 iterates from 1 to W'. x' is the loop counter for which y' is calculated as given at Line 6. In each iteration an edge pixel of opposite gradient orientation is searched. If an edge pixel of opposite gradient orientation is found, it is classified as marking and the inner loop is terminated. The search is carried out in both directions perpendicular to l.

Algorithm 3 LineClassification(G_m, G_o, W', L)

```
count ⇐ 0
for $i = 1$ to $\xi_h \times L.length$ do
    $x \Leftarrow L(i).x$
    $y \Leftarrow L(i).y$
    for $x' = 1$ to $W'$ do
        $y' \Leftarrow round(x' \times \tan\theta)$
        // test if there is a dual edge at $(x + x', y + y')$
        if $G_m(x + x', y + y') \neq 0$ & $abs(G_o(x + x', y + y') - G_o(x, y)) \approx 180^\circ$ then
            $count \Leftarrow count + 1$
            // break inner loop
            break
        else if $G_m(x - x', y - y') \neq 0$ & $abs(G_o(x - x', y - y') - G_o(x, y)) \approx 180^\circ$ then
            $count \Leftarrow count + 1$
            // break inner loop
            break
        end if
    end for
    // testing if "enough" dual edges have been found
    if $count \geq \xi_l \times L.length$ then
        return true
    end if
end for
return false
```

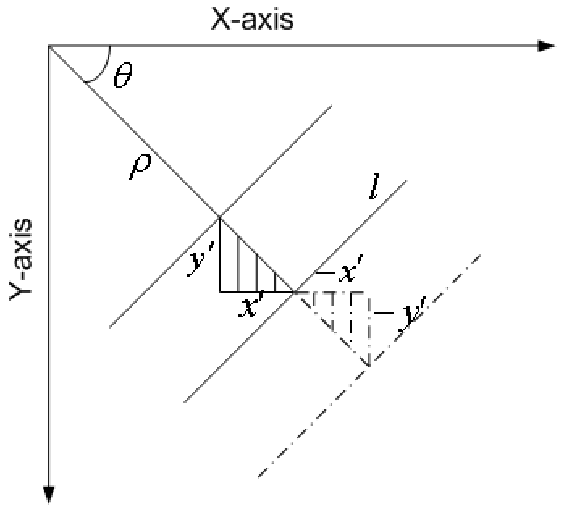


Figure 3.8: Detection of field markings

3.5 Detecting Corners and Junctions

The fully described line segments have end points, length, length of normal and orientation as attributes, which are used to detect junctions (T-junction and Y-junction), corners and line intersections. n line segments in an image may result in ${}^nC_2 = \frac{n(n-1)}{2}$ intersection points. These intersection points can be calculated by solving (3.9) for all combinations of two line segments. The intersection point p may not lie close to one or both of the line segments. Fig. 3.9(a) shows an example where p lies on both l_1 and l_2, whereas, in Fig. 3.9(c) it lies only on l_2. The length of normal and orientation of l_1 and l_2 are denoted by ρ_1 and θ_1 and

ρ_2 and θ_2, respectively. The process of calculating the intersection point and its verification for a pair of line segments l_1 and l_2 is given in Algorithm 4. Illustration of the algorithm is

Algorithm 4 LineIntersection(l_1, l_2)

```
$p' \leftarrow 0$
// calculate intersection $p$ of $l_1$ and $l_2$ using (3.9) for $(\rho_1,\theta_1)$ and $(\rho_2,\theta_2)$
```

$$p.x \Leftarrow \frac{\rho_1 \sin\theta_2 - \rho_2 \sin\theta_1}{\cos\theta_1 \sin\theta_2 - \cos\theta_2 \sin\theta_1} \tag{3.12}$$

$$p.y \Leftarrow \frac{\rho_1 \cos\theta_2 - \rho_2 \cos\theta_1}{\sin\theta_1 \cos\theta_2 - \sin\theta_2 \cos\theta_1} \tag{3.13}$$

```
if $p$ within image boundary then
  // rotate $l_1$ and $p$ by $\Omega_1 = \theta_1$ - 90 °
```

$$l'_1 \Leftarrow \begin{bmatrix} l_1.x_1 & l_1.y_1 \\ l_1.x_2 & l_1.y_2 \end{bmatrix} \begin{bmatrix} \cos\Omega_1 & -\sin\Omega_1 \\ \sin\Omega_1 & \cos\Omega_1 \end{bmatrix} \tag{3.14}$$

$$p' \Leftarrow \begin{bmatrix} p.x & p.y \end{bmatrix} \begin{bmatrix} \cos\Omega_1 & -\sin\Omega_1 \\ \sin\Omega_1 & \cos\Omega_1 \end{bmatrix} \tag{3.15}$$

```
  if $l'_1.x_1 - \delta \leq p'.x \leq l'_1.x_2 + \delta$ then
    // rotate end points of $l_2$
```

$$l'_2 \Leftarrow \begin{bmatrix} l_2.x_1 & l_2.y_1 \\ l_2.x_2 & l_2.y_2 \end{bmatrix} \begin{bmatrix} \cos\Omega_2 & -\sin\Omega_2 \\ \sin\Omega_2 & \cos\Omega_2 \end{bmatrix} \tag{3.16}$$

```
    if $l'_2.x_1 - \delta \leq p'.x \leq l'_2.x_2 + \delta$ then
      return $p$
    end if
  else
    return 0
  end if
else
  return 0
end if
```

given in Fig. 3.9. The lines in Fig. 3.9(a) intersect at point p and satisfy all the conditions of the algorithm. The intersection shown in Fig. 3.9(c) is not real and is detected when the first line is rotated around the origin.

The tolerance δ is dependent on the amount of distortion that can be tolerated in the position of an intersection. The position of an intersection point with respect to the end points of the line segments helps to classify an intersection as a corner or a junction of a given type. Furthermore, the information about the type of line segments (as a field marking or not) helps to recognize intersection points uniquely. In case there are m (with $m > 2$) line segments meeting at one point there are ${}^mC_2 = \frac{m(m-1)}{2}$ intersection points close to one another. The average is taken as the required position of the junction if two or more intersections are within a given tolerance. The rotated line segments are shown by thick solid lines in Fig.3.9. The information about the end points of line segments allows us to detect corners and junctions with this method. On the average there are $10 - 15$ line segments which makes this method attractive [BSN05].

Davies [Dav88] introduced an approach to corner detection based on the Generalized Hough

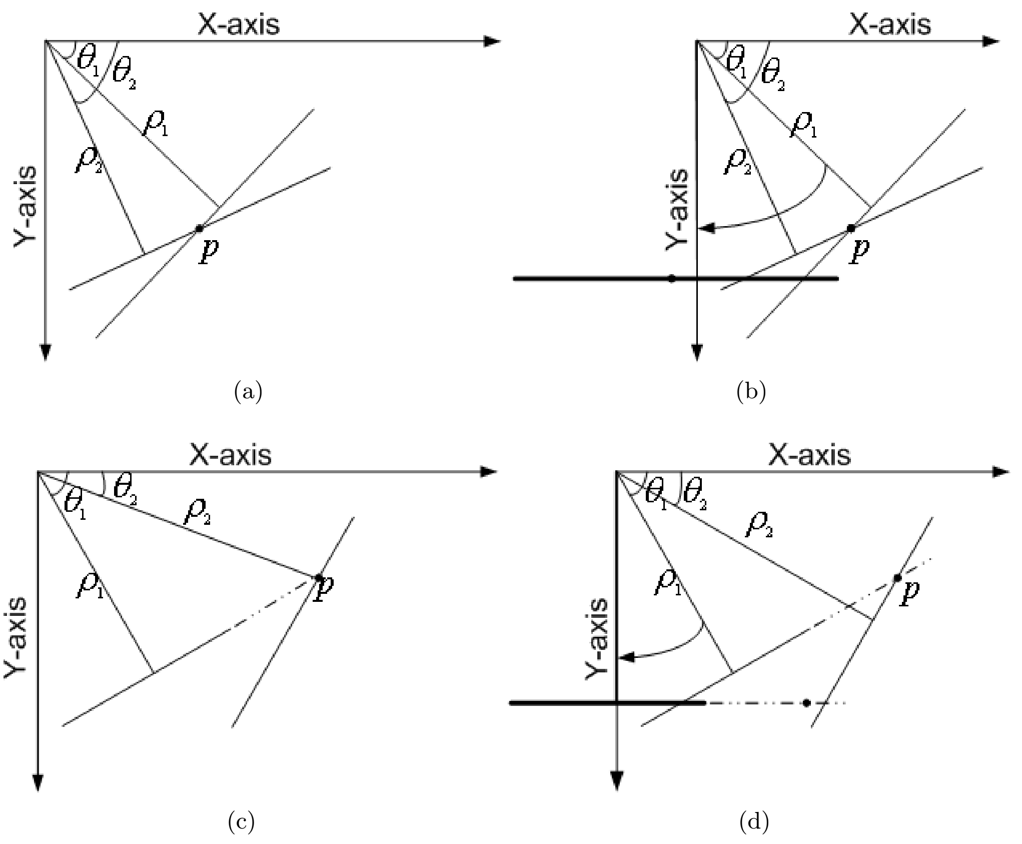


Figure 3.9: Finding line intersection (a) line segments intersect each other at p (b) rotated x-coordinate of p lies in between the x-components of the rotated end points (c) virtual intersection (d) rotated line together with the virtual intersection

Transform (GHT) [Bal81]. He uses a form of the generalized HT in which the parametrization is optimized for shapes having straight edges [Dav87]. This method has the advantage that it can be used to find corners which may be blunt or occluded.

Barret and Peterson [BP01] detect corners, junctions and line intersection by exploiting the accumulator array of the HT. After the generation of the parameter space and detection of peaks they perform a second pass through the edge map and compute the line integral over each sinusoid that corresponds to the current edge point. If a sinusoid passes through more than 2 peaks, the sum/integral is stored into a new accumulator array which has direct one-to-one correspondence with the original image. This method could also be applied to detect a virtual junction if all image pixels are considered as edge pixels in the second pass.

Another approach is that of detecting the corners directly by template matching technique. However, if the corners are not sharp the templates would be so large that the resulting search would be intractable. This problem is further complicated if corners of different orientations and different angles have to be searched. For example $Q \times M \times N^2 \times n^2$ operations have to be performed to detect corners using M $n \times n$ pixel templates corresponding to M possible orientations of a corner with Q different angles in an $N \times N$ pixel image [Dav88].

3.6 Detecting Color Transitions

Goals are marked with different colors (blue and yellow). The vertical edges of the goal corners are normally missed during edge detection and subsequent line segment extraction as the change in y-channel value between white and yellow is not significant and the length of the edge is small as compared to other lines in the environment. Therefore, goal corners are extracted based on color transitions. The process is outlined in the following paragraphs.

In the left camera image, pixels are tested if they belong to either blue or yellow color. This ’segmentation’ is done at a lower scale. Every fourth pixel in a row of every fourth row is tested, which results in a rectangular window around the blue or yellow color patches, if any. The neighborhood of this rectangular window is searched for color transition (color transition from white to yellow, yellow to white, white to blue, or blue to white represents a goal corner), using a full scale. If a color transition is detected in the left image, the corresponding feature points are searched in the right image. If the corresponding feature point is detected in the right image, its distance from the current robot position is calculated. Detection of two such points determine the robot position as shown in Fig. 4.1(b).

The use of two colors to detect a transition makes the process robust against outliers. All rows inside the rectangular window are searched for transition pixels. One value in a group of pixels is taken as the x-component of the edge between the wall and the colored goal. Outliers in the group are eliminated using simple statistical measures. The calculated stereo range is used to estimate robot position and orientation as discussed in the next chapters.

3.7 Discussion

The global nature of the HT extracts the strongest groups of collinear pixels. Within each group, locally significant but spatially separated (sub-groups of) pixels are merged based on parameters minimum gap G_{min}, maximum gap G_{max} and the minimum length of the line segment L_{min}. These parameters make the merging process robust against random noise.

The distinct and bright color of the goals makes them strong candidates to be selected as landmarks. Furthermore, calculating robot position and orientation with respect to goal corners is efficient since only $N/16$ pixels are tested to determine the rectangular boundaries around the color patches (if any), N being the total number of pixels. This results in localization of color patches which are then searched for the actual transitions.

The intersection point of two line segments is calculated based on ρ and θ values of the selected peaks. Further improvements in performance could be achieved if these values are recalculated once the line segments are extracted. The next chapter presents global self-localization using these landmarks.

Chapter 4

Feature Based Global Self-localization

The discussion here builds on the previous chapter and presents landmark based global self-localization. Global localization is required for an initial startup or at a later stage when the robot loses track of its pose during navigation. Existing approaches are based on dense range scans [GS96, GWN01], active beacon systems [HTMA03, Kle92], artificial landmarks [BSB$^+$00, JKLK02], bearing measurements using omni-directional cameras [JIPB03, BAK04] or bearing/range calculation using single frontal cameras [BKHS00, CRK99], while a feature based stereo vision system for range calculation is presented here. Location of the robot is estimated using range measurements with respect to distinct landmarks. Unlike methods based on angle measurements, this method requires only two distinct landmarks [BS06].

Distinct landmark features are sparse in our application domain and are frequently occluded by other robots. This makes simultaneous acquisition of two or more landmarks difficult. Therefore, another method is proposed that requires only one distinct landmark [BSG06]. The method requires independent estimation of the robot absolute orientation. To enable the robot to estimate its absolute orientation, it is equipped with a compass in addition to odometry and the stereo vision system [NM05, NCB$^+$06]. Uncertainty analysis of both the methods is also presented in this chapter.

4.1 Related Work

If the robot can only measure angles between landmarks then a minimum of three distinct landmarks are required to triangulate the robot position on a planar surface [YM05, Sug88]. Furthermore, methods based on angle measurements are sensitive to the relative position of robot with respect to the landmarks [SB93]. Whereas, when range measurements are available this requirement drops to two if ordering of the landmarks with respect to the robot is possible [BS06].

There have been approaches to maximize the chances of simultaneous acquisition of multiple landmarks using omni-directional cameras with viewing angle of 360° [SB05, JIPB03]. A camera looking at a mirror of special shape provides the robot with visual information in all directions simultaneously. The work reported in [ML01] is an example of such a setup used for localization of soccer robots. As the resolution and camera range depend on the mirror geometry, the resolution of objects is usually smaller than that of directional images [UNMK03]. The mirror should be fitted at a required height in order to have a global view of the environment. This could be a drawback if the maximum height of the robot is constrained.

Approaches using single frontal cameras in conjunction with odometry are widely used for self-localization. These methods are either based on calculating range and/or bearing based on known shape and size of landmarks or enforce special constraints on environment features [BKHS00, CRK99]. Choi et al. [CRK99] present a localization system for mobile indoor robot using a monocular vision system. They use calibrated camera to acquire depth of features. The robot's actual position is computed by referencing door corners over the floor edge, and heading by direction of the floor edge lines. With frontal cameras one can have high resolution but the field of view is limited [UNMK03]. Furthermore, range measurement using single image is not always robust especially for targets near the altitude of the sensor [NJW+98].

Error in robot observations results in uncertain position estimate. In addition to estimating the position, it is also important to know the reliability of this estimate. One method to accommodate observation and landmark position error, an observation tolerance ε and landmark position tolerances σ_i, $i = 1, 2, \ldots$ can be used [AH93]. ε is a maximal error in sensor measurement, and likewise σ_i is a maximal error in the absolute position of landmark i. Using the notation as introduced in [AH93], landmark i can be defined as $\mathbf{p}_i = \begin{bmatrix} p_i - \sigma_i & p_i + \sigma_i \end{bmatrix}$, and an observation as $\mathbf{o}_i = \begin{bmatrix} o_i - \varepsilon & o_i + \varepsilon \end{bmatrix}$. The basic idea of the algorithm reported in [AH93] is to recognize in the camera image those entities that stay invariant with respect to the position and orientation of the robot as it travels in its environment. A triple of point landmarks on a wall in the environment is an example of such invariant entities [DK02]. The algorithm cannot be run if the number of observed landmark points is less than three [AH93].

Matthies and Shafer [MS87] use a stereo vision system to estimate translation and rotation between two three dimensional sets of points. The robot position is estimated as a succession of each transformation between frames. Similarly, Kriegman et al. [KTB89] use a stereo vision system for robot navigation in indoor environments. The robot position is tracked, based on information gathered from features which move between each successive motion of the robot, while its position uncertainty is calculated using first order approximation. The drawback of such an approach is that without referencing global features in the map, uncertainty increases with the passage of time.

Fuzzy logic based methods [Saf97] can also be used to account for error in odometry and external sensors and to represent uncertainty in robot location. Buschka et al. [BSW00] adopt this technique for localization of Sony Aibo robots using lines and color marking in the environment, whereas Herrero-Perez et al. [HPMBS05b] use corner formed by field lines.

To address these limitations a stereo vision based range estimation is proposed. This approach enables range estimation of landmarks and the use of bi/trilateration approach helps to calculate robot position. Moveover, the pivoted camera head enables the robot to have a 360°view of its environment. The robot could acquire additional information from the environment by rotating its head. The major advantage of the approach is that it requires less landmarks as compared to the angle based methods [BS06, BSG06].

For uncertainty analysis a basic assumption about error distribution in the left and right camera images is made and this error is then propagated into the robot position estimate. For error propagation, it is further assumed that the non-linear models that link the robot observations to its position can be represented by the first two components of the Taylor series expansion around the estimated observation. Experimental results have shown that this model can adequately capture the uncertainty in robot position that arise from sensor imperfections, whereas, error in landmark location in the global map and correspondence analysis is ignored [BSKN07].

4.2 Landmark Based Position Estimation

In the following sections different methods of position estimation using range or angle measurements are discussed. The discussion provides basis for derivation of the robot position in Section 4.3.

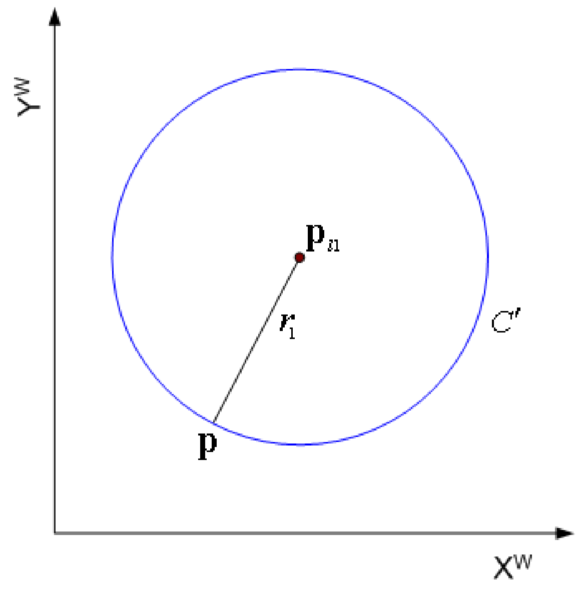
4.2.1 Position Estimation using Range Measurements

Let us consider that while navigating in an environment, a robot detects a unique landmark feature $\mathbf{p}_{l1}$ whose location is known in the global map and estimates its distance r_1. With this information the robot position is constrained to a circle C' of radius r_1 and center at $\mathbf{p}_{l1}$, considering that the robot is moving on a flat surface. This phenomena is illustrated in Fig. 4.1(a).

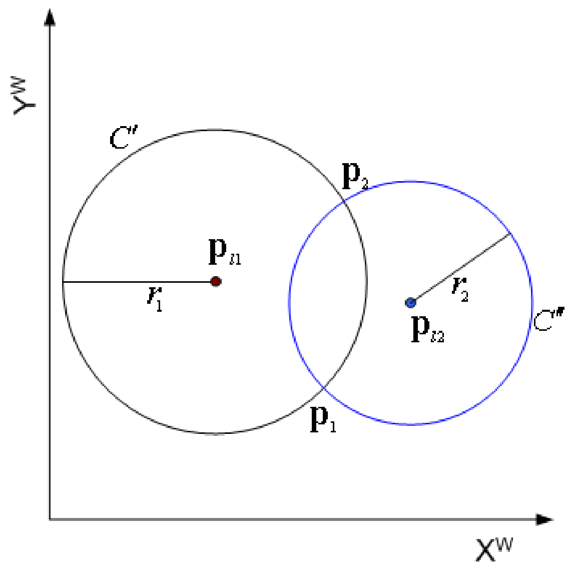
Similarly, detection of landmark point $\mathbf{p}_{l2}$ and its range estimate r_2 will constrain robot position to circle C''. Concurrent identification and range estimation of two landmark points will constrain the robot position to the intersection of two points. Fig.4.1(b) shows that after identification of two landmark points $\mathbf{p}_{l1}$ and $\mathbf{p}_{l2}$, the robot is at one of the two possible positions $\mathbf{p}_1$ or $\mathbf{p}_2$, which are determined by the intersection of circle C' and C''. The correct solution among the two possible candidates can be determined by assuming a fixed order of landmarks with respect to the robot [Sug88]. For example it may be known from the environment map that landmark point $\mathbf{p}_{l1}$ appears to the left or right of $\mathbf{p}_{l2}$ from certain areas in the environment or one of the solutions lies at impossible locations.

The ambiguity can also be resolved by identifying three landmarks simultaneously. Fig. 4.1(c) shows that the robot identifies landmark point $\mathbf{p}_{l3}$ and measures its range r_3. Identification of $\mathbf{p}_{l3}$ and its range estimation constrains the robot position to circle C'''. It is clear from the figure that when all the three points are extracted concurrently the robot position is at the intersection of C', C'' and C'''.

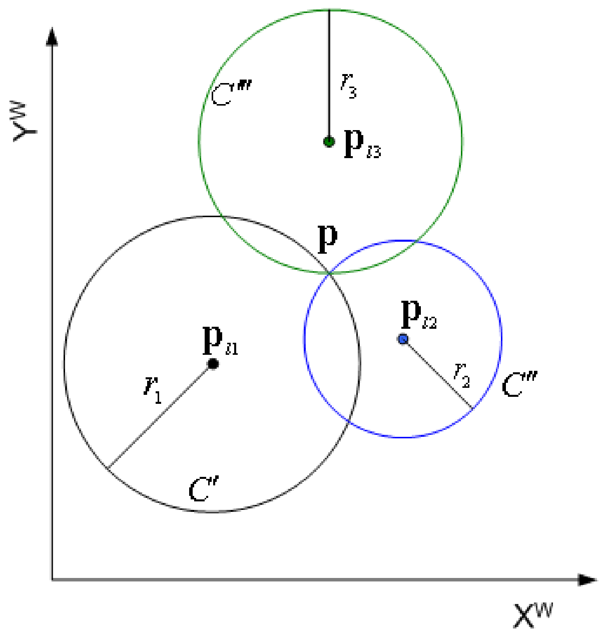
From the above discussion it is clear that with perfect distance measurement to three distinct landmarks or to two distinct landmarks with ordering constraint, the robot position can be calculated uniquely. However, depending on the onboard sensors, the robot may not be able to estimate distance to landmarks in its environment. If it can only estimate angle between landmarks then a minimum of three landmarks are required for position calculation. This is explained in the following section.



(a)



(b)



(c)

Figure 4.1: Position estimation using range measurements (a) with range estimation to a single landmark the robot position is constrained to a circle. If the robot is able to identify two landmarks and measure there range its position is constrained to two points as in (b) or to a single point when three landmarks are extracted concurrently as in (c)

4.2.2 Position Estimation using Angle Measurements

Fig. 4.2(a) shows a case where the robot identifies two distinct landmarks $\mathbf{p}_{l1}$ and $\mathbf{p}_{l2}$ in its environment and measures the angle α between them. The angle between the two landmark points remains equal to α as long as the robot is on the circular arc C' or C'' (shown dashed in Fig. 4.2(a)) [Sug88, SB93]. The angle, α, formed by the rays from robot position to each landmark is called the visual angle [SB93]. Again, as in the case with distance measurement the ambiguity between C' and C'' can be resolved by considering a fixed order of landmark points.

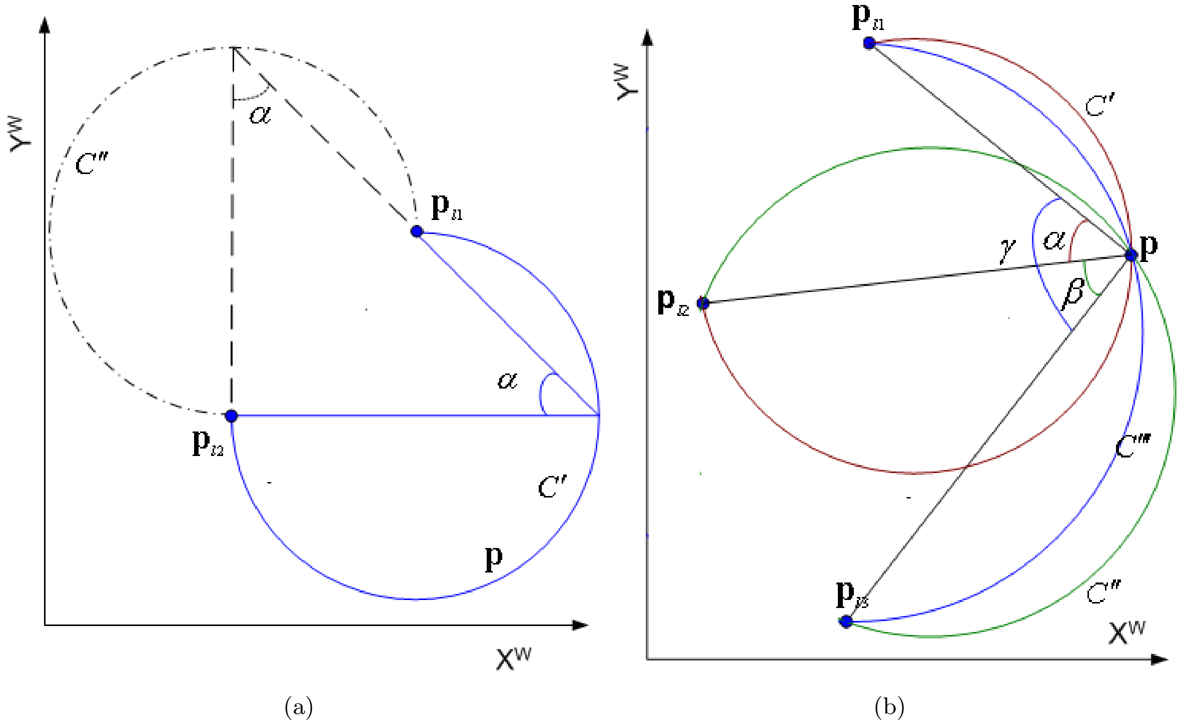


Figure 4.2: Angle based position estimation (a) identifying two landmarks and measuring angle α between them constrains the robot position to circular arc C' or C'' (b) using three landmarks and angle measurements the robot position can be calculated uniquely. The robot $\mathbf{p}$ is determined by the intersection of the three circular arcs C', C'', and C'''

In this case there are infinite number of candidate positions and the robot must acquire a third landmark point in its environment. Since the robot is moving on a flat terrain, visual angles between three landmark points will constrain robot to the intersection of three circles, unless all three points and the robot lie on the same circle [Sug88, Kro89, SB93]. Fig. 4.2(b) illustrates that the robot identifies three landmark points $\mathbf{p}_{l1}$, $\mathbf{p}_{l2}$ and $\mathbf{p}_{l3}$ and measures the visual angles α, β and γ. Angle α is formed between $\mathbf{p}_{l1}$ and $\mathbf{p}_{l2}$. The angle α stays the same as long as the robot is somewhere along the circular arc C' shown in red. Similarly, β is formed between $\mathbf{p}_{l2}$ and $\mathbf{p}_{l3}$ which constrains the robot position to C''. The angle γ constrains the robot position to C''' and is formed between $\mathbf{p}_{l1}$ and $\mathbf{p}_{l3}$. The intersection of the three circular arcs result in the position $\mathbf{p}$.

4.2.3 Adding Absolute Orientation of the Robot

Absolute robot orientation can be obtained by equipping the robot with sensors such as a magnetic compass. This information provides a constraint which results in reduction of minimum number of required landmarks for a unique position estimate if combined with landmark sightings. As can be seen in Fig. 4.3(a) knowledge of absolute orientation of the robot and identification of landmark point $\mathbf{p}_{l1}$ constrains the robot orientation to line L'. When this constraint is combined with the one shown in Fig. 4.1(a), the robot position can be calculated by the intersection of line L' and circle C'. Further details are given in Section 4.3.2.

Similarly, additional knowledge of robot orientation in Fig. 4.2(a) will constrain the robot position to a single point. In Fig. 4.2(a) the robot detects landmarks $\mathbf{p}_{l1}$ and $\mathbf{p}_{l2}$ and measures the visual angle α, which constrains the robot position to the circular arc C'. When this knowledge is combined with absolute orientation of the robot (θ), the robot position is constrained to a single point on the circular arc. This phenomenon is illustrated in Fig. 4.3(b) as intersection of L_1 and L_2. Detection of $\mathbf{p}_{l1}$ and measurement of angle ψ_1 when combined with θ constrains the robot position to L_1, whereas detection of $\mathbf{p}_{l2}$ and measurement of angle ψ_2 defines the constraint L_2. Simultaneous detection of $\mathbf{p}_{l1}$ and $\mathbf{p}_{l2}$ and measurement of angles ψ_1 and ψ_2, in addition to θ define a single point $\mathbf{p}$ as the robot location. This point can be calculated as the intersection of lines L_1 and L_2 shown in blue and red color in Fig. 4.3(b).

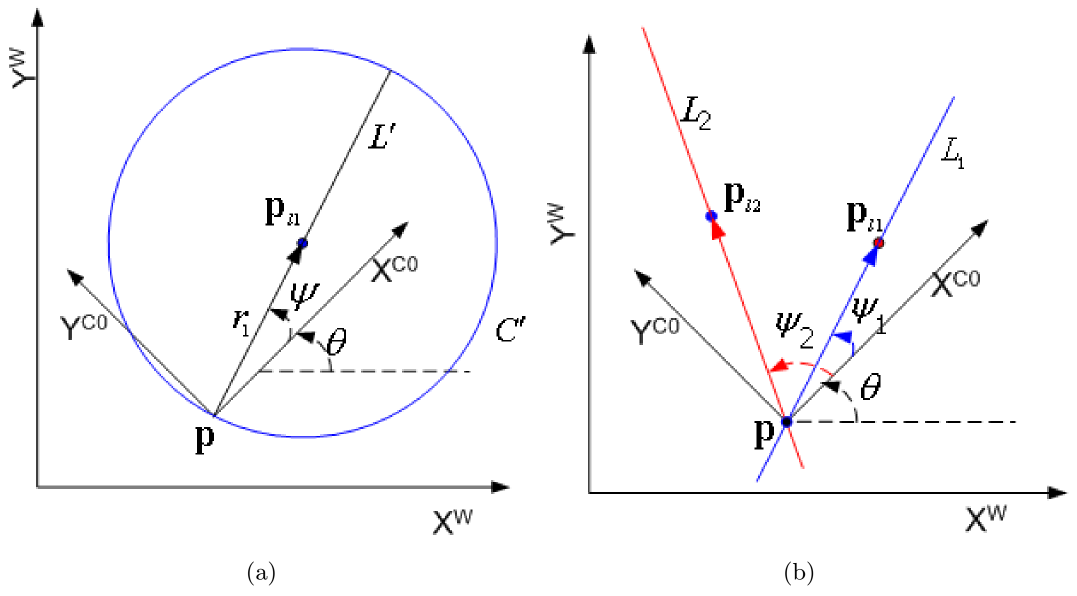


Figure 4.3: Typically, the minimum number of landmarks can be reduced by one if absolute orientation of the robot is available (a) combining range estimation with absolute orientation of robot (b) two landmarks are required when only angle between landmarks can be estimated

4.2.4 Using a Virtual Landmark

Range measurement with respect to a distinct landmark and knowledge of absolute orientation is sufficient to calculate the robot orientation. Another approach of position estimation using a single landmark uses the concept of *runningfix* [Cas86]. The underlying principle of the running fix is that an angle or range obtained from a landmark at time $k-1$ can be utilized at time k, as long as the cumulative movement vector, recorded since the reading was obtained, is added to the position vector of the landmark, thus creating a virtual landmark. This approach is explained in the following paragraphs [BSG06].

Going back to the case shown in Fig. 4.1(a), where the robot detects a distinct landmark point $\mathbf{p}_{l1}$ and measures its distance r_1 from it, after detecting this landmark the robot tries to identify other landmarks. However, if it is the only landmark that can be detected from the current position of the robot, it moves to a new point and tries to detect the same landmark again. Movement of the robot by δ_x and δ_y can be seen as motion of the circle C' by the same amount. Radius of the circle remains equal to r_1 as all candidate positions undergo the same displacement. This moment of the circle can be described as if the circle's center has moved to a new location, namely, $\mathbf{p}_{l2}$ as shown in Fig. 4.4(a). $\mathbf{p}_{l2}$ can be considered as a virtual landmark [Cas86, BSG06]. The robot is now somewhere on circle C''. The original circle is labeled C and is shown dotted in Fig. 4.4(a).

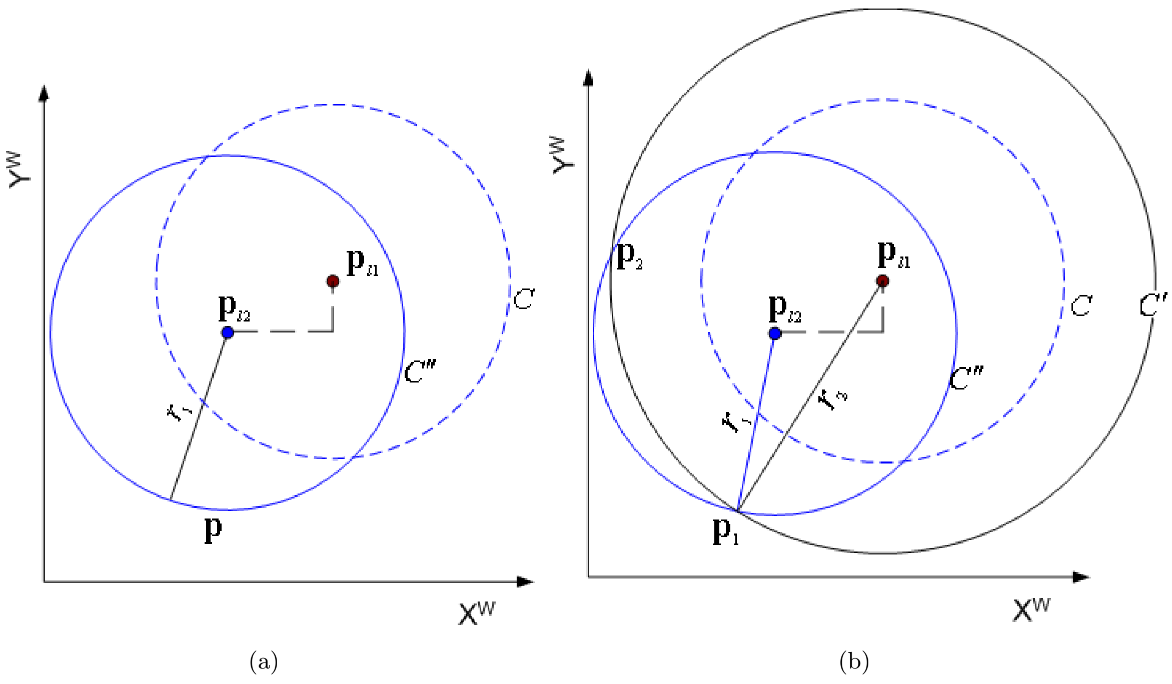


Figure 4.4: Position estimation using a virtual landmark (a) robot's movement is effectively motion of the circle by the same amount (b) robot is at the intersection of C' and C''

Suppose, that after moving to this new location the robot detects the same landmark $\mathbf{p}_{l1}$ again and measures its range r_2. Identification of this landmark and its range measurement will constrain the robot position to a new circle, C', which is shown in Fig. 4.4(b). The illustration in Fig. 4.4(b) shows that the intersection of circle C' and C'' will give possible robot position in the global coordinate system. One of the intersection points $\mathbf{p}_1$ or $\mathbf{p}_2$ will qualify for the possible robot position. The ambiguity between the two possible positions of the robot can be resolved by a rough estimate of the robot position or by tracking both candidate positions until further information is available. Resolving the ambiguity between the two position is not always possible, nevertheless, the robot position belief is concentrated to two points on the circle.

4.2.5 Position Uncertainty Due to Erroneous Measurements

So far the discussion is based on perfect identification of landmarks and error free measurements, hence, perfect localization. However, measurements are never perfect, which result in an uncertain position estimate [SB93]. Errors in landmark location, range and visual angle, absolute orientation of the robot can vary significantly in magnitude and would be due to a variety of reasons ranging from sensor imperfections to errors in correspondence [SB93].

For example, as illustrated in Fig. 4.5(a), with an erroneous range measurement the robot can be anywhere on the thick ring instead of a perfect circle of Fig. 4.1(a). The intersection of such thickened rings will determine the uncertainty in robot position when two or more landmarks are used. The area of uncertainty is shown shaded in Fig. 4.5(b) and Fig. 4.5(c) for the case of two or three landmarks, respectively.

Uncertainty in visual angle estimate will constrain the robot position to a thickened ring as shown in Fig. 4.6(a). The thickness of the ring is determined by the amount of error in visual angle [Kro89]. When three landmarks are used, error in estimate of visual angles α, β and γ will constrain the robot position to the intersection of such thick rings as shown in Fig. 4.6(b).

Similarly, landmark identification and uncertain absolute orientation of the robot constrains its location to a cone shaped area. The intersection of such a cone and a thick circle will determine the uncertainty of robot location if this information is combined with an erroneous range estimate as shown in Fig. 4.7(a). If the only information is the absolute orientation of the robot and the robot detects two landmarks, then its position is constrained to the intersection of two cones, labeled L_1 and L_2 as shown in Fig. 4.7(b).

4.2.6 Error in Landmark Identification and Matching

In addition to measurement errors, there could be error in landmark identification and matching with the world map. Moreover, some landmarks may not be detected, and some spurious landmarks may be detected. A missing landmark may not effect the localization if enough landmarks are detected. However, if a landmark point is spurious then the localization algorithm may determine an incorrect solution [Kro89]. Additionally, there could be error in landmark location in the global map of the environment and error in correspondence analysis. Errors in correspondence is that the point which has been identified as point x on the map may really be point y [SB93]. Furthermore, the assumption that the landmarks are distinguishable may not always hold.

4.2.7 Sensitivity to Landmark Configuration

Landmark based methods are sensitive to its configuration which is a major drawback [SS00]. The angle based methods are not applicable when all the landmarks and the robot lie on a circle or in a straight line. For angle based methods the most accurate position estimate occurs when the robot is at the center of the circular zone constructed by the three landmarks [SZL95]. Errors increase proportionally to the distance from the circle center. Also, there are singularities in which the errors reach extreme values. These singularities are proven to be on the bounding circle constructed by the three landmarks/beacons [SS00, TA93].

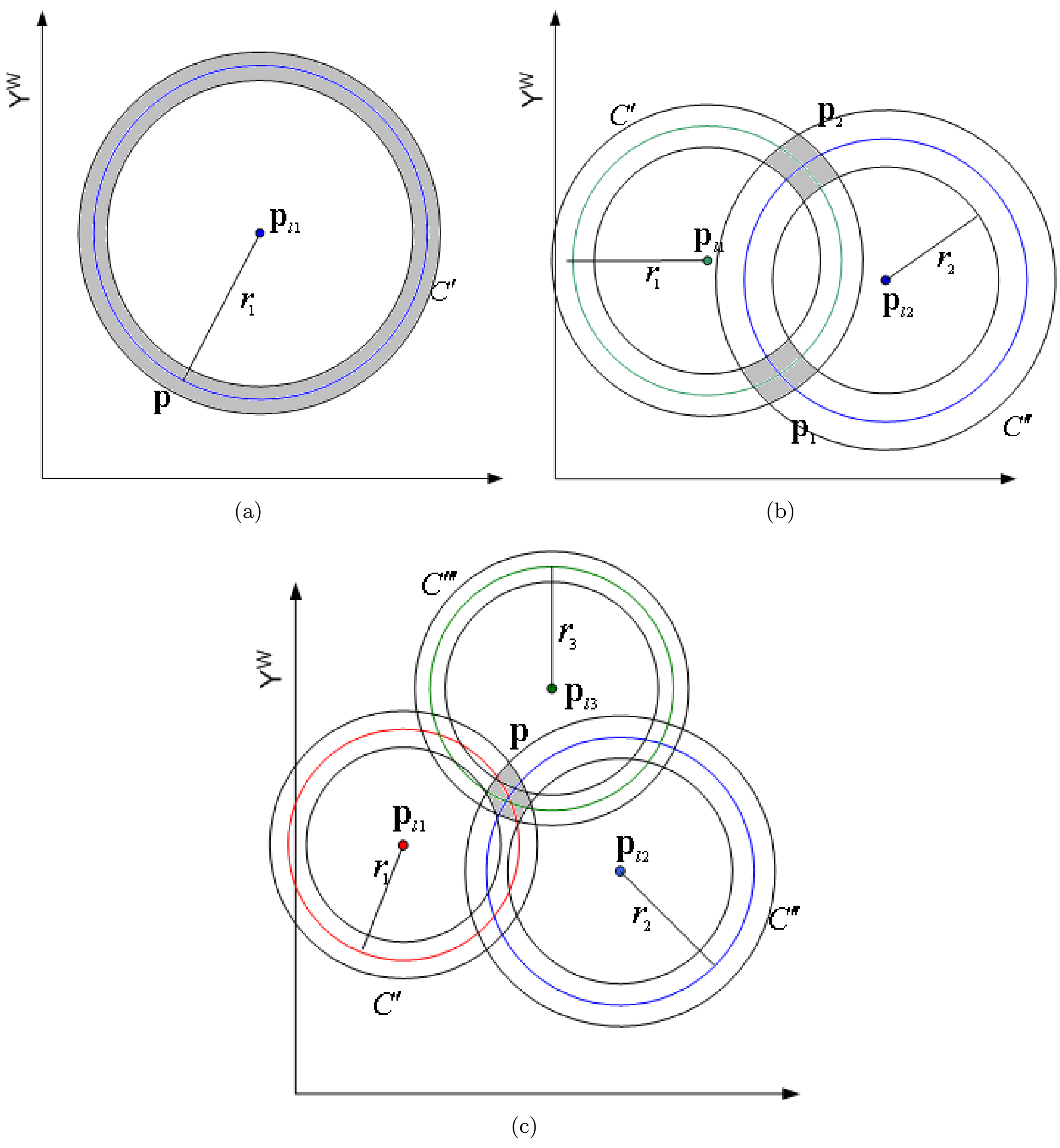
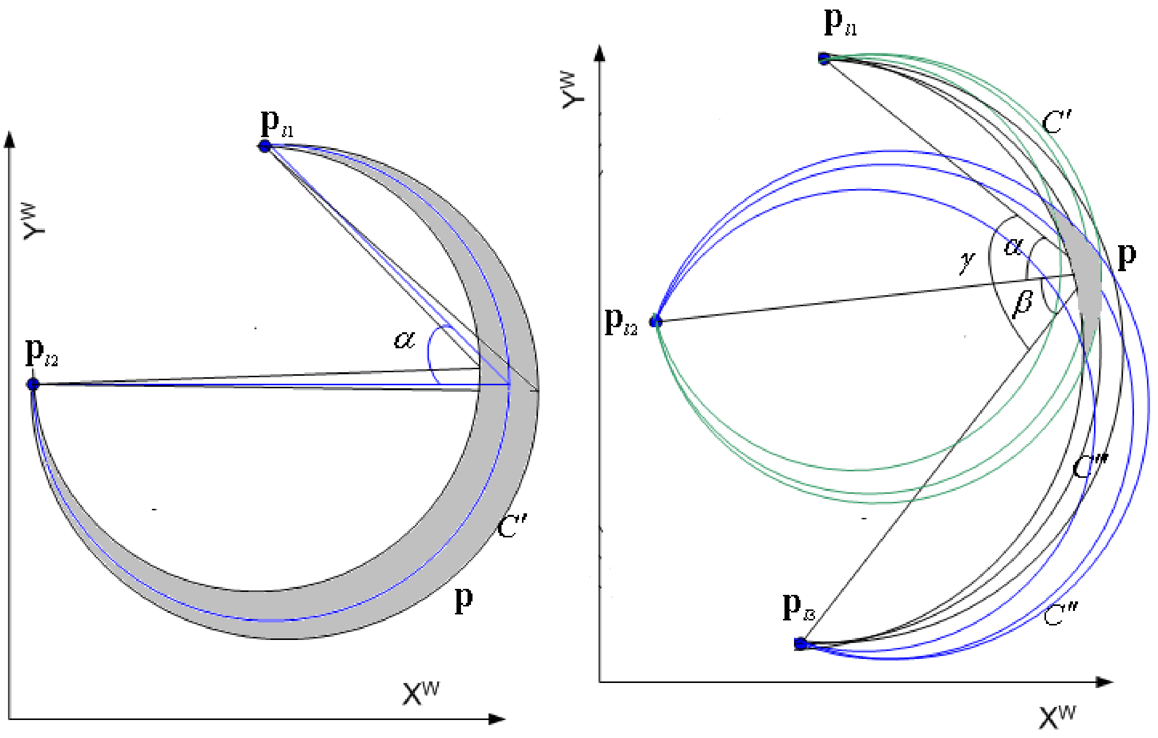


Figure 4.5: Uncertainty due to erroneous range measurements (a) the position of the robot is restricted to a thickened ring if there is error in distance measurement to the landmark. Intersection of such rings will give position uncertainty when two or three landmarks are detected as shown in (b) and (c)

Sutherland and Thompson [SB93] address the localization problem using distinguishable landmarks in the environment. They analyze the areas of uncertainty for different configuration of three landmarks and the errors in visual angle estimates to those landmarks, and show that for a given error in angle measurement the size of the localization error varies depending on the configuration of landmarks.



(a) (b)

Figure 4.6: Uncertainty due to error in estimate of visual angle (a) the robot is somewhere on the thick ring when the angle between two landmarks is in error. Intersection of such thick rings will determine the uncertainty as shown (b)

4.2.8 Position Estimation with More Than Three Landmarks

For much of the reasons stated in the previous sections, landmark based methods for position estimation with three landmarks does not yield an optimal position estimation. To overcome this problem, more than three landmarks have to be identified in the environment, and all the measurements have to be used to get an optimal estimate for the robot's position. The geometric method does not lend itself naturally to more than three measurements. Thus, costly nonlinear minimization techniques have to be employed [Shi02]. Given n landmarks, there are $\frac{n!}{3!(n-3)!}$ combinations of three landmarks that can be used to compute $\frac{n!}{3!(n-3)!}$ position estimates by the triangulation method. One way of combining these estimates —to obtain a final position estimate— is to compute their average [BG97].

Sugihara [Sug88] present an algorithm that tries to find the location of the robot inside a room using vertical edges extracted from an image taken by an onboard camera whose optical axis is kept parallel to the ground. The robot measures the direction of rays each of which emanates from the robot's position and pierces through at least one of the points in the environment. The measurements are considered to be error free. The algorithm runs in time $\mathcal{O}(n^3 \lg(n))$, where n is number of identical landmark points.

Krotkov [Kro89] followed the approach of Sugihara and formulated the positioning problem as a search in a tree of interpretation (pairing of landmark directions and landmark points). He developed an algorithm to search the tree efficiently and to determine the solution positions,

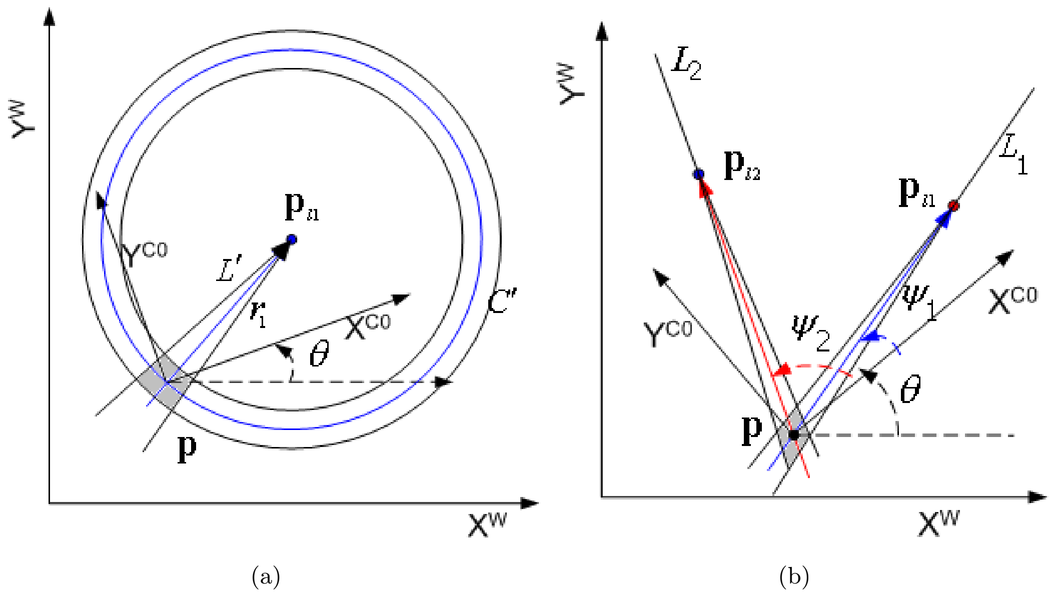


Figure 4.7: Error in absolute orientation of the robot result in an uncertain position (a) robot position is constrained to the intersection of the ring and cone (b) with two landmarks and erroneous absolute orientation the robot location is determined by the intersection of two cones

taking into account the errors in the landmark direction angle. According to this analysis, if the angle between two landmark points is in error, then the possible robot location does not lie on circular arc, but in a thickened region as shown in Fig. 4.5(b).

Atiya and Hager [AH93], developed a real-time vision-based algorithm for self-localization under the same assumptions. They mention both the lack of treatment of observation error in [Sug88] and the disregard with false positives — detecting features that do not correspond to a known landmark — in [Kro89]. They propose to represent the sensory error by a tolerance, which leads to a set-based algorithm for solving the matching problem and computing the absolute location of a mobile robot for indoor navigation.

As triangulation with noisy data is based on solving nonlinear equations with complicated closed-form solutions, Betke and Gurvits [BG97] argue that standard algorithms that provide least squares solutions for large numbers of nonlinear equations take too long for real-time robot navigation. They represent landmarks as complex numbers, which is the key idea to get a set of n linear equations, where n is the number of landmark points. This set of linear equations is represented as a vector equation, which is solved in $\mathcal{O}(n)$ time. Their robot is equipped with a camera that points upwards onto a reflective ball which acts like a mirror of the surroundings. Only a circular, one-dimensional strip of the brightness of each image is analyzed. The strips provide information on the angle of one landmark relative to another landmark but not on the distance of the landmarks to the camera. At least three landmarks are required to find a solution.

Another method to reduce the time complexity of solving the non-linear equations is reported by Shimshoni [Shi02]. He uses each angle measured to define a linear constraint on the position and orientation of the robot. For three or more angle measurements the same number of linear constraints are defined. This linear system is solved using Singular Value Decomposition (SVD) [PFTV92]. They also estimate the accuracy of the estimate by estimating its covariance matrix. They propose scaling of the input data to improve quality of results.

4.3 Position Estimation Using Vision Based Range

The presentation of landmark based position estimation in the previous section was based on a hypothetical sensor that would provide range and angle with respect to point landmarks. The following builds on it by using stereo vision based range estimates to distinctive environment features as discussed in Chapter 3. First robot position is calculated using two landmarks. This is followed by another derivation based on range measurement to a single landmark combined with the absolute orientation of the robot.

4.3.1 Calculating Position Using Two Landmarks

Using (3.6), range r_1 and r_2 with respect to landmarks $\mathbf{p}_{l1}$ and $\mathbf{p}_{l2}$ in Fig. 4.1(b) is given by (4.1) and (4.2) as follows:

$$r_1 = \sqrt{(x_1^{C0})^2 + (y_1^{C0})^2} \tag{4.1}$$

$$r_2 = \sqrt{(x_2^{C0})^2 + (y_2^{C0})^2} \tag{4.2}$$

The circle C' and C'' can be described using (4.3) and (4.4):

$$r_1^2 = (x - x_{l1})^2 + (y - y_{l1})^2 \tag{4.3}$$

$$r_2^2 = (x - x_{l2})^2 + (y - y_{l2})^2 \tag{4.4}$$

where (x_{l1}, y_{l1}) and (x_{l2}, y_{l2}) are the elements of $\mathbf{p}_{l1}$ and $\mathbf{p}_{l2}$ representing their location in the world coordinate system and centers of C' and C''. The intersection of the two circles can be found by solving (4.3) and (4.4), which will give the robot position in global coordinate system. Subtraction of (4.3) from (4.4) and re-arranging terms results in the following expression for x and y:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \frac{-D \pm \sqrt{D^2 - 4CE}}{2C} \\ A + B\frac{(D \pm \sqrt{D^2 - 4CE}}{2C} \end{bmatrix} \tag{4.5}$$

where

$$\begin{aligned} A &= \frac{r_1^2 - r_2^2 + x_{l2}^2 - x_{l1}^2 + y_{l2}^2 - y_{l1}^2}{2a} \\ B &= \frac{x_{l1} - x_{l2}}{a} \\ C &= B^2 + 1 \\ D &= 2AB - 2y_{l1}B - 2x_{l1} \\ E &= A^2 + x_{l1}^2 + y_{l1}^2 - 2y_{l1}A - r_1^2 \\ a &= y_{l2} - y_{l1} \end{aligned}$$

The subscripts 1 and 2 differentiate between quantities related to the two landmarks. One of the solution pairs (if any) from (4.5) will qualify for the robot position. The ambiguity between the two positions of the robot is resolved by using the information that the landmarks appear in a fixed order with respect to the robot.

The robot orientation θ can be calculated using its position and position of one of the landmarks. From illustration of Fig. 4.8 the following expression for the robot orientation can be obtained:

$$\theta = \varphi - \psi \tag{4.6}$$

where

$$\begin{aligned} \varphi &= atan2\left(y_{l1} - y, x_{l1} - x\right) \\ \psi &= atan2\left(y^{C0}, x^{C0}\right) \end{aligned}$$

In these equations (x_{l1}, y_{l1}) is the landmark location in the global coordinate system, (x, y) is the robot position and (x^{C0}, y^{C0}) is the location of the selected landmark in robot coordinate system. After having derived expression for all components of the robot pose, it can be

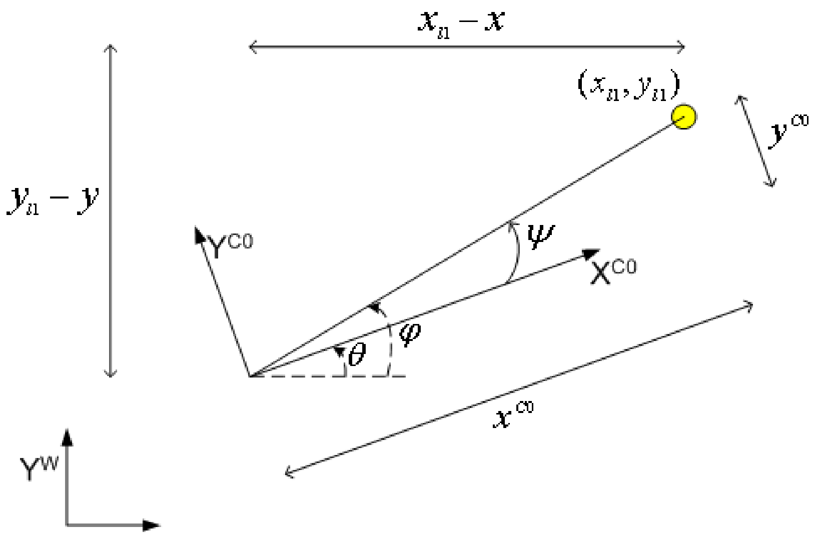


Figure 4.8: Calculating robot orientation

written in vector form as follows:

$$\mathbf{p} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} = \begin{bmatrix} {}^{\left(-D\pm\sqrt{D^2-4CE}\right)}\!/_{2C} \\ A + B\,{}^{\left(-D\pm\sqrt{D^2-4CE}\right)}\!/_{2C} \\ atan2\left(y_{l1} - y, x_{l1} - x\right) - atan2\left(y^{C0}, x^{C0}\right) \end{bmatrix} \tag{4.7}$$

4.3.2 Position Estimation Using Single Landmark

The previous subsection presented position estimation using two landmarks. Simultaneous acquisition of two or more landmarks is troublesome in our application environment where landmarks are scarce and they are frequently occluded by other robots for longer durations. Therefore, the localization algorithm should be based on as few landmarks as possible. This section builds on Section 4.2.3, where absolute orientation of the robot is used in position estimation.

As stated earlier and shown in Fig. 4.3(a), finding location of a single landmark in robot coordinate system is sufficient for robot position calculation if the absolute orientation of the robot is known. The line L' of Fig. 4.3(a) can be described using the slope-point form of line equations as given by (4.8), whereas, the circle C' is described by (4.3).

$$y - y_{l1} = m\,(x - x_{l1}) \tag{4.8}$$

The slope of the line m is equal to $\tan(\theta + \psi)$ and $\psi = atan2(y^{C0}, x^{C0})$. The intersection of L' and C' can be obtained as solution of (4.8) and (4.3), which is given by the following expression:

$$\mathbf{p} = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x_{ll} \pm \frac{r_1}{\sqrt{1+m^2}} \\ y_{ll} \pm \frac{mr_1}{1+m^2} \end{bmatrix} \tag{4.9}$$

Equation (4.9) results in two candidate positions for each intersection of the line with circle, one of which qualifies for the robot position. In our case the line segment always originates at the robot location and terminates at the landmark. This information is used to resolve the ambiguity between the two candidate positions. This is further illustrated in Fig. 4.9. For this case the robot position is given by the following expression:

$$\mathbf{p} = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x_{ll} - r_1 \cos\varphi \\ y_{ll} - r_1 \sin\varphi \end{bmatrix} \tag{4.10}$$

where $\varphi = \psi + \theta$. For measuring absolute orientation our robot is equipped with a magnetic

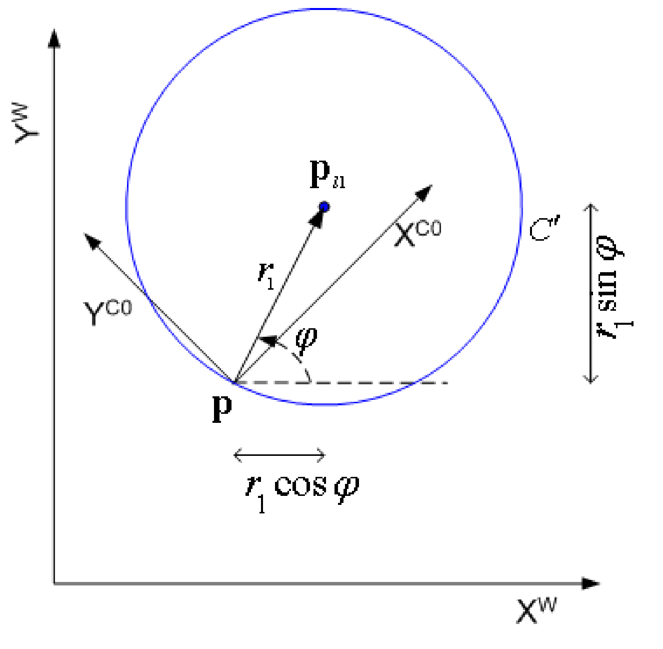


Figure 4.9: Single landmark based localization: robot position constrained to the intersection of line and circle

compass. It is most accurate ($< 3\,°$) when the robot is not moving and there are no other robots or magnetic objects in its close proximity [NM05].

4.4 Uncertainty Analysis

The previous section presented two methods for robot position estimation assuming error free measurements. However, as stated earlier error in sensor readings, landmark locations and in correspondence results in an uncertain position estimate. Therefore, in addition to knowing the position of the robot it is also important to measure reliability of this estimate.

Ignoring the landmark location error in the global coordinate system and the error in correspondence analysis, any error in estimating the location of the landmark projections in the left and right image together with an error in absolute orientation of the robot will determine the uncertainty in the robot location.

The major source of error in location of landmark points in the images is due to image quantization [MS87]. As shown in Fig. 4.10, all points in the light gray areas result in same pixels in the image plane. The uncertainty area grows as the distance from the image plane increases. This uncertainty can be adequately captured by assuming Gaussian error distribution in estimating the image coordinates and propagating it to the 3D location of the point using first order approximation of (3.4). This results in a 3D gaussian probability distribution for the 3D coordinates. A 2D view of the uncertainty ellipses are shown in black color in Fig. 4.10 [MS87]. However, as noted by Kriegman et al. [KTB89] the linearization of the perspective transformation does not hold for distant point correspondence as the disparity decreases and higher order terms will dominate.

4.4.1 Robot Location Uncertainty Using Single Landmark

Substituting values of r_1 and φ in (4.10) the following expression for robot position is obtained:

$$\mathbf{p} = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x_{ll} - \sqrt{(x_1^{C0})^2 + (y_1^{C0})^2}\cos\left(\theta + atan2\left(y^{C0}, x^{C0}\right)\right) \\ y_{ll} - \sqrt{(x_1^{C0})^2 + (y_1^{C0})^2}\sin\left(\theta + atan2\left(y^{C0}, x^{C0}\right)\right) \end{bmatrix} \tag{4.11}$$

It is clear from (4.11) that the robot position is dependent on x_1^{C0}, y_1^{C0} and θ. Whereas, (3.4) and (3.5) reveal that x_1^{C0} and y_1^{C0} are in turn dependent on u_{l1} and u_{r1}, hence, the input vector for position estimate using this method can be formed as follows:

$$\mathbf{i} = \begin{bmatrix} u_l \\ u_r \\ \theta \end{bmatrix} \tag{4.12}$$

and formulate the robot pose as:

$$\mathbf{p} = \mathbf{g}\left(\mathbf{i}\right) \tag{4.13}$$

The imperfections of the input quantities will result in an uncertain position estimate as shown in Fig. 4.7(a). It is assumed that error in input vector is zero mean Gaussian with the following covariance:

$$\mathbf{\Sigma}_i = \begin{bmatrix} \sigma_{uu}^2 & 0 & 0 \\ 0 & \sigma_{uu}^2 & 0 \\ 0 & 0 & \sigma_{\theta\theta}^2 \end{bmatrix} \tag{4.14}$$

where σ_{uu}^2 is the variance of error distribution of u_l or u_r and $\sigma_{\theta\theta}^2$ is the variance of error distribution in absolute orientation of the robot. It is reasonable to assume that the three

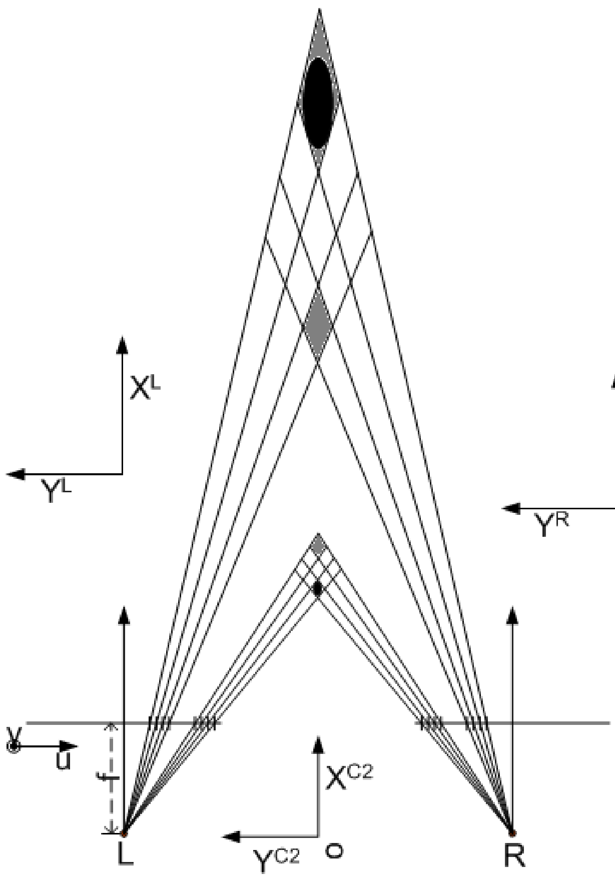


Figure 4.10: Stereo triangulation error due to image quantization (adapted from [MS87])

components of the input vector are not correlated as they are estimated independently. Furthermore, due to the identical camera assumption the error distribution of u_l and u_r are supposed to be identical. This error is propagated into the position estimate by the transformation (4.11) through (3.4) and (3.5). The systems given by (4.11), (3.4) and (3.5) are nonlinear and the resulting error distribution will not be a Gaussian. However, it is assumed that it can be adequately represented by the first two terms of Taylor series expansion around estimated input, $\widehat{\mathbf{i}}$, which results in the following expression:

$$\mathbf{p} = \mathbf{g}(\widehat{\mathbf{i}}) + \mathbf{J}_i\widetilde{\mathbf{i}} + \ldots \tag{4.15}$$

where

$$\begin{aligned} \widehat{\mathbf{p}} &= \mathbf{g}(\widehat{\mathbf{i}}) \\ \widetilde{\mathbf{p}} &\approx \mathbf{J}_i\widetilde{\mathbf{i}} \\ \mathbf{J}_i &= \frac{\partial \mathbf{p}}{\partial \mathbf{i}} \end{aligned}$$

In the above equations $\widehat{\mathbf{p}}$ is the position estimate, $\widetilde{\mathbf{p}}$ its error and $\mathbf{J}_i$ is the jacobian of $\mathbf{p}$ with respect to $\mathbf{i}$ evaluated at its estimated value $\widehat{\mathbf{i}}$. $\mathbf{J}_i$ is having the following elements:

$$\begin{aligned}
{}^{\partial x}/_{\partial u_l} &= {}^{1}/_{rd^2}[u_{ro}(bsx^{C0} - cby^{C0}) + fb(sy^{C0} + cx^{C0})] \\
{}^{\partial x}/_{\partial u_r} &= {}^{-1}/_{rd^2}[u_{lo}(bsx^{C0} - cby^{C0}) + fb(sy^{C0} + cx^{C0})] \\
{}^{\partial x}/_{\partial \theta} &= rs \\
{}^{\partial y}/_{\partial u_l} &= {}^{-1}/_{rd^2}[u_{ro}(bcx^{C0} + sby^{C0}) + fb(cy^{C0} - sx^{C0})] \\
{}^{\partial y}/_{\partial u_r} &= {}^{1}/_{rd^2}[u_{lo}(bcx^{C0} + sby^{C0}) + fb(cy^{C0} - sx^{C0})] \\
{}^{\partial y}/_{\partial \theta} &= -rc
\end{aligned}$$

where $d = u_l - u_r$, $u_{ro} = u_r - o_u$ and $u_{lo} = u_l - o_u$. Using the approximation from (4.15) expression for the covariance matrix of position estimate can be derived as follows:

$$\mathbf{\Sigma}_p = E\{\widetilde{\mathbf{p}}\widetilde{\mathbf{p}}^T\} = \mathbf{J}_i\mathbf{\Sigma}_i\mathbf{J}_i^T$$

4.4.2 Location Uncertainty Using Two Landmarks

Following the same model of the previous subsection for the uncertainty of the robot position estimation using two landmarks the following input vector can be formed:

$$\mathbf{i} = \begin{bmatrix} u_{l1} \\ u_{r1} \\ u_{l2} \\ u_{r2} \end{bmatrix} \tag{4.16}$$

The input vector in this case has four components which correspond to the location of the two landmarks in the left and right camera images. The imperfection in its estimation is propagated into the robot pose using (4.7) which result in an uncertain position estimate as illustrated in Fig. 4.5(b) and Fig. 4.11. This imperfection is modeled with a zero mean Gaussian having the following covariance matrix:

$$\mathbf{\Sigma}_i = \sigma_{uu}^2 \mathbf{I}_{4\times 4} \tag{4.17}$$

where $\mathbf{I}_{4\times 4}$ is 4×4 identity matrix. Using the same principles as adapted in the previous section the following expression for the covariance matrix of position estimate is derived using the new method:

$$\mathbf{\Sigma}_p = \mathbf{J}_i\mathbf{\Sigma}_i\mathbf{J}_i^T \tag{4.18}$$

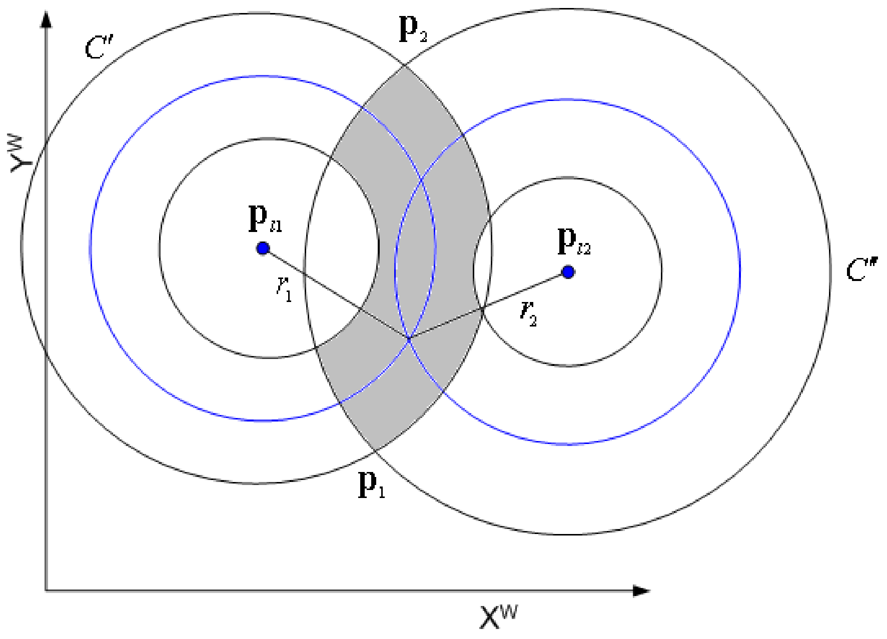


Figure 4.11: The resulting uncertainty in robot position is irregular in shape

Elements of the updated jacobian matrix are given by the following expressions:

$$
\begin{aligned}
{}^{\partial x}/_{\partial u_{l1}} &= {}^{(-fbx_1^{C0}+b(u_{r1}-o_u)y_1^{C0})T_1}/_{aCd_1^2} \\
{}^{\partial x}/_{\partial u_{r1}} &= {}^{(fbx_1^{C0}-b(u_{l1}-o_u)y_1^{C0})T_1}/_{aCd_1^2} \\
{}^{\partial x}/_{\partial u_{l2}} &= {}^{(-fbx_2^{C0}+b(u_{r2}-o_u)y_2^{C0})T_2}/_{aCd_2^2} \\
{}^{\partial x}/_{\partial u_{r2}} &= {}^{(fbx_2^{C0}-b(u_{l2}-o_u)y_2^{C0})T_2}/_{aCd_2^2} \\
{}^{\partial y}/_{\partial u_{l1}} &= {}^{(-fbx_1^{C0}+b(u_{r1}-o_u)y_1^{C0})}/_{ad_1^2} + B\,{}^{\partial x}/_{\partial u_{l1}} \\
{}^{\partial y}/_{\partial u_{r1}} &= {}^{(fbx_1^{C0}-b(u_{l1}-o_u)y_1^{C0})}/_{ad_1^2} + B\,{}^{\partial x}/_{\partial u_{r1}} \\
{}^{\partial y}/_{\partial u_{l2}} &= {}^{(fbx_2^{C0}-b(u_{r2}-o_u)y_2^{C0})}/_{ad_1^2} + B\,{}^{\partial x}/_{\partial u_{l2}} \\
{}^{\partial y}/_{\partial u_{r2}} &= {}^{(-fbx_2^{C0}+b(u_{l2}-o_u)y_2^{C0})}/_{ad_2^2} + B\,{}^{\partial x}/_{\partial u_{r2}} \\
{}^{\partial \theta}/_{\partial u_{l1}} &= {}^{((y_{l1}-y)({}^{\partial x}/_{\partial u_{l1}})-(x_{l1}-x)({}^{\partial y}/_{\partial u_{l1}}))}/_{r_{pl}^2} \\
&\quad - {}^{(bx_1^{C0}(u_{r1}-o_u)+fby_1^{C0})}/_{r_1^2 d_1^2} \\
{}^{\partial \theta}/_{\partial u_{r1}} &= {}^{((y_{l1}-y)({}^{\partial x}/_{\partial u_{r1}})-(x_{l1}-x)({}^{\partial y}/_{\partial u_{r1}}))}/_{r_{pl}^2} \\
&\quad + {}^{(bx_1^{C0}(u_{l1}-o_u)+fby_1^{C0})}/_{r_1^2 d_1^2} \\
{}^{\partial \theta}/_{\partial u_{l2}} &= {}^{((y_{l1}-y)({}^{\partial x}/_{\partial u_{l2}})-(x_{l1}-x)({}^{\partial y}/_{\partial u_{l2}}))}/_{r_{pl}^2} \\
{}^{\partial \theta}/_{\partial u_{r2}} &= {}^{((y_{l1}-y)({}^{\partial x}/_{\partial u_{l2}})-(x_{l1}-x)({}^{\partial y}/_{\partial u_{l2}}))}/_{r_{pl}^2}
\end{aligned}
$$

where $T_1 = -B \pm {}^{1}/_{\sqrt{D^2-4CE}}(BD-2C(A-y_{l1}-a))$, $T_2 = B \pm {}^{1}/_{\sqrt{D^2-4CE}}(-BD+2C(A-y_{l1}))$ and $r_{pl} = ((x_{l1}-x)^2 + (y_{l1}-y)^2)$.

The error analysis method presented in this section results in a measure of uncertainty in the form of a covariance matrix. Such a form is required to initialize a location tracking algorithm such as the EKF [New05].

4.5 Discussion

The methods presented in this chapter give the robot the ability to globally localize itself with two distinct landmarks or with one landmark if absolute orientation of the robot is available.

Global self-localization is sometimes impossible if enough features are not available. Therefore, it is required to track the robot position once it is estimated. During tracking, the uncertainty grows, which can be suppressed when external observation of landmark features are available. The position estimate and its corresponding covariance matrix provide the required information to initialize a location tracker such as an EKF. This is the subject of the next chapter.

Chapter 5

Position Tracking and Information Fusion

The discussion in the previous two chapters reveals that the global position of the robot can be estimated whenever range estimate of a single landmark and absolute orientation of the robot are available [BSGM06]. If the robot orientation cannot be estimated independently then two distinct landmarks are required [BS06]. However, instantaneous acquisition of distinct landmarks at all times is not possible as landmarks are few in number and are frequently occluded. Furthermore, global position estimation using only external sensors is time consuming [BEF96]. Similarly, it is shown in Section 2.1 that only local sensors are not enough for position estimation as it requires knowledge of the initial position and the errors accumulate unbounded. Therefore, it is required to use a combination of local and external sensors and fuse the information obtained from them. In order to have an all time position estimate, this chapter aims at tracking the robot position. The growing position error is suppressed by acquiring features from the robot environment whenever possible.

The information obtained from both the internal and external sensors are uncertain and incomplete, therefore it is required to use error models for their representation. The robot local sensors are calibrated for systematic errors, whereas, for the non-systematic error, it is assumed that it can be represented by zero mean Gaussian distribution. Similarly, the robot observations can also be adequately represented by a Gaussian distribution [BSGK07].

A simplified assumption of our application environment is that there exist features that are globally distinguishable. This resolves the problem of uniquely identifying areas in the environment, helps us avoid local minima traps and saves us from tracking multiple hypotheses or to maintain a multi-modal position belief. Therefore, the position belief can also be modeled by a Gaussian distribution. Keeping these properties of environment in view, the EKF is considered to be best suited for our application [BDN07]. Furthermore, the limited processing capabilities of the robot warrants the use of methods that need less memory and demand comparatively less computational power. This requirement favors EKF over ML and MCL [GF02].

The main weakness of the unimodal Gaussian representation of the belief is compensated by the global localization methods as discussed in the previous chapter. These methods are used to provide the robot an initial position at startup. The kidnapped robot problem or tracking failures can be detected by applying a validation gate to the innovation sequence.

After detecting such a failure the global localization procedure can be initiated. However, as shown in our comparison paper [BDN07] and discussed in the next chapter, the EKF also behaves gracefully during the initial position estimation. The convergence of the estimated position towards the true position is comparable to the one achieved with MCL method.

Implementation of a PF based method for position estimation and information fusion is also carried out in this chapter. The results of the comparison study are presented in the next chapter, which illustrates the performance and suitability of the EKF for the proposed application. Construction of the robot observation vector and its uncertainty analysis is also discussed in this chapter.

5.1 Extended Kalman Filter

Position tracking using EKF is the subject of this section. Detailed discussion about derivation of EKF and its application to robot navigation problems can be found in [Cro95] and [New05]. Using the global localization techniques, the filter is provided with optimal initial conditions (in the sense of minimum mean square error). State transition and observation models in the framework of an EKF are explained in the following sections. Presentation about geometric construction of differential drive robot, formation of the control vector and its uncertainty analysis is given in Appendix B.

5.1.1 State Transition Model

Beginning with the assumption of having a function $\mathbf{f}$ that models the transition from state $\mathbf{p}_{k-1}$[1] to $\mathbf{p}_k$ in the presence of control vector $\mathbf{u}_{k-1}$[2] at time k. The control vector is independent of state $\mathbf{p}_{k-1}$ and is supposed to be corrupted by an additive zero mean Gaussian noise $\widetilde{\mathbf{u}}_k$ of strength $\mathbf{U}_k$. This model is formally stated by the following equation:

$$\mathbf{p}_k = \mathbf{f}\left(\mathbf{p}_{k-1}, \mathbf{u}_{k-1}, k\right) \tag{5.1}$$

where $\mathbf{u}_k = \widehat{\mathbf{u}}_k + \widetilde{\mathbf{u}}_k$. The quantities $\mathbf{p}_{k-1}$ and $\mathbf{p}_k$ are the desired (unknown) states of the robot. Given robot observations $\mathbf{Z}_{k-1}$, a minimum mean square estimate of the robot state $\mathbf{p}_{k-1}$ at time $k-1$ is defined as [New05]:

$$\widehat{\mathbf{p}}_{k-1|k-1} = E\{\mathbf{p}_{k-1}|\mathbf{Z}_{k-1}\} \tag{5.2}$$

The uncertainty of this estimate is denoted by $\mathbf{P}_{k-1|k-1}$ and can be written as follows:

$$\mathbf{P}_{k-1|k-1} = E\{\widetilde{\mathbf{p}}_{k-1|k-1}\widetilde{\mathbf{p}}^T_{k-1|k-1}|\mathbf{Z}_{k-1}\} \tag{5.3}$$

where

$$\widetilde{\mathbf{p}}_{k-1|k-1} = \mathbf{p}_{k-1} - \widehat{\mathbf{p}}_{k-1|k-1} \tag{5.4}$$

is the estimation error. Here $\widetilde{\mathbf{p}}_{k-1|k-1}$ is supposed to be zero mean.

[1]The notation used here is a slightly changed version of [New05]. In our case $\widehat{\mathbf{p}}_{i|j}$ is the minimum mean square estimate of $\mathbf{p}_i$ given observation until time j i.e. $\mathbf{Z}_j$, where $j \leq i$. Similarly, $\mathbf{P}_{i|j}$ is the uncertainty of the estimate $\widehat{\mathbf{p}}_{i|j}$

[2]The control vector is the state change in robot frame of reference

The next step is to derive expression for $\widehat{\mathbf{p}}_{k|k-1}$, an estimate of $\mathbf{p}_k$ using all measurements but the one at time k and its uncertainty $\mathbf{P}_{k|k-1}$ using (5.1). This is called *prediction*. Similar to (5.2) this estimate can be written as follows:

$$\widehat{\mathbf{p}}_{k|k-1} = E\{\mathbf{p}_k|\mathbf{Z}_{k-1}\} \tag{5.5}$$

Using multivariate Taylor series expansion $\mathbf{p}_k$ is linearized around $\widehat{\mathbf{p}}_{k-1|k-1}$ and $\widehat{\mathbf{u}}_k$ as given by the following expression:

$$\mathbf{p}_k = \mathbf{f}(\widehat{\mathbf{p}}_{k-1|k-1}, \widehat{\mathbf{u}}_k, k) + \mathbf{J}_1(\mathbf{p}_{k-1} - \widehat{\mathbf{p}}_{k-1|k-1}) + \mathbf{J}_2(\mathbf{u}_k - \widehat{\mathbf{u}}_k) + \dots \tag{5.6}$$

where $\mathbf{J}_1$ and $\mathbf{J}_2$ are the jacobians of (5.1) w.r.t $\mathbf{p}_{k-1}$ and $\mathbf{u}_k$ evaluated at $\widehat{\mathbf{p}}_{k-1|k-1}$ and $\widehat{\mathbf{u}}_k$, respectively. Substitution of (5.6) in (5.5) results in the following:

$$\begin{aligned}
\widehat{\mathbf{p}}_{k|k-1} &= E\{\mathbf{p}_k|\mathbf{Z}_{k-1}\} \\
&\approx E\{\mathbf{f}(\widehat{\mathbf{p}}_{k-1|k-1}, \widehat{\mathbf{u}}_k, k) + \mathbf{J}_1(\mathbf{p}_{k-1} - \widehat{\mathbf{p}}_{k-1|k-1}) + \mathbf{J}_2\widetilde{\mathbf{u}}_k|\mathbf{Z}_{k-1}\} \\
&= \mathbf{f}(\widehat{\mathbf{p}}_{k-1|k-1}, \widehat{\mathbf{u}}_k, k) + \mathbf{J}_1 E\{\widetilde{\mathbf{p}}_{k-1|k-1}|\mathbf{Z}_{k-1}\} + \mathbf{J}_2 E\{\widetilde{\mathbf{u}}_k|\mathbf{Z}_{k-1}\} \\
&= \mathbf{f}(\widehat{\mathbf{p}}_{k-1|k-1}, \widehat{\mathbf{u}}_k, k)
\end{aligned} \tag{5.7}$$

From (5.6) and (5.7) expression for prediction error can be derived as follows:

$$\begin{aligned}
\widetilde{\mathbf{p}}_{k|k-1} &= \mathbf{p}_k - \widehat{\mathbf{p}}_{k|k-1} \\
&\approx \mathbf{f}(\widehat{\mathbf{p}}_{k-1|k-1}, \widehat{\mathbf{u}}_k, k) + \mathbf{J}_1\widetilde{\mathbf{p}}_{k-1|k-1} + \mathbf{J}_2\widetilde{\mathbf{u}}_k - \mathbf{f}(\widehat{\mathbf{p}}_{k-1|k-1}, \widehat{\mathbf{u}}_k, k) \\
&= \mathbf{J}_1\widetilde{\mathbf{p}}_{k-1|k-1} + \mathbf{J}_2\widetilde{\mathbf{u}}_k
\end{aligned} \tag{5.8}$$

The prediction error is then used to calculate prediction covariance as given below:

$$\begin{aligned}
\mathbf{P}_{k|k-1} &= E\{\widetilde{\mathbf{p}}_{k|k-1}\widetilde{\mathbf{p}}_{k|k-1}^T|\mathbf{Z}_{k-1}\} \\
&= E\{(\mathbf{J}_1\widetilde{\mathbf{p}}_{k-1|k-1} + \mathbf{J}_2\widetilde{\mathbf{u}}_k)(\mathbf{J}_1\widetilde{\mathbf{p}}_{k-1|k-1} + \mathbf{J}_2\widetilde{\mathbf{u}}_k)^T|\mathbf{Z}_{k-1}\} \\
&= \mathbf{J}_1 E\{\widetilde{\mathbf{p}}_{k-1|k-1}\widetilde{\mathbf{p}}_{k-1|k-1}^T|\mathbf{Z}_{k-1}\}\mathbf{J}_1^T + \mathbf{J}_2 E\{\widetilde{\mathbf{u}}_k\widetilde{\mathbf{u}}_k^T|\mathbf{Z}_{k-1}\}\mathbf{J}_2^T \\
&\quad + \mathbf{J}_1 E\{\widetilde{\mathbf{p}}_{k-1|k-1}\widetilde{\mathbf{u}}_k^T|\mathbf{Z}_{k-1}\}\mathbf{J}_2^T + \mathbf{J}_2 E\{\widetilde{\mathbf{u}}_k\widetilde{\mathbf{p}}_{k-1|k-1}^T|\mathbf{Z}_{k-1}\}\mathbf{J}_1^T \\
&= \mathbf{J}_1\mathbf{P}_{k-1|k-1}\mathbf{J}_1^T + \mathbf{J}_2\mathbf{U}_k\mathbf{J}_2^T
\end{aligned} \tag{5.9}$$

Equation (5.9) is based on the fact that $\widetilde{\mathbf{u}}_k$ and $\widetilde{\mathbf{p}}_{k-1|k-1}$ are not correlated. Detailed discussion on state transition model for a two wheeled differential drive robot [NM05] is given in Appendix B, where expressions for all terms used in this section are derived.

5.1.2 Observation Model

The robot observation model links the current state of the robot $\mathbf{p}_k$ with its observation $\mathbf{z}_k$ and is given by the following transformation:

$$\mathbf{z}_k = \mathbf{h}(\mathbf{p}_k, k) + \mathbf{w}_k \tag{5.10}$$

Robot observation is assumed to be corrupted by zero mean Gaussian noise $\mathbf{w}_k$ of strength $\mathbf{R}$. Using Taylor series expansion of (5.10) around the predicted state and retaining only the first two terms, the following expression of the observation model can be obtained:

$$\mathbf{z}_k \approx \mathbf{h}\left(\widehat{\mathbf{p}}_{k|k-1}\right) + \mathbf{J}_{zp}(\widehat{\mathbf{p}}_{k|k-1} - \mathbf{p}_k) + \mathbf{w}_k \tag{5.11}$$

where

$$\mathbf{z}_{k|k-1} = E\{\mathbf{z}_k|\mathbf{Z}_{k-1}\} = \mathbf{h}\left(\widehat{\mathbf{p}}_{k|k-1}\right) \tag{5.12}$$

is the observation prediction and $\mathbf{J}_{zp}$ is the observation jacobian evaluated at $\widehat{\mathbf{p}}_{k|k-1}$. The deviation of the actual observation from the predicted one is called *innovation* and is evaluated as follows:

$$\begin{aligned}\widetilde{\mathbf{z}}_{k|k-1} &= \mathbf{z}_k - \mathbf{z}_{k|k-1}\\ &\approx \mathbf{h}(\widehat{\mathbf{p}}_{k|k-1}) + \mathbf{J}_{zp}(\widehat{\mathbf{p}}_{k|k-1} - \mathbf{p}_k) + \mathbf{w}_k - \mathbf{h}(\widehat{\mathbf{p}}_{k|k-1})\\ &= \mathbf{J}_{zp}\widetilde{\mathbf{p}}_{k|k-1} + \mathbf{w}_k \end{aligned}\tag{5.13}$$

After having derived expression for *innovation*, expression for its covariance matrix can be derived as follows:

$$\begin{aligned}\mathbf{S} &= E\{\widetilde{\mathbf{z}}_{k|k-1}\widetilde{\mathbf{z}}_{k|k-1}^T|\mathbf{Z}_{k-1}\}\\ &= E\{(\mathbf{J}_{zp}\widetilde{\mathbf{p}}_{k|k-1} + \mathbf{w}_k)(\mathbf{J}_{zp}\widetilde{\mathbf{p}}_{k|k-1} + \mathbf{w}_k)^T|\mathbf{Z}_{k-1}\}\\ &= \mathbf{J}_{zp}E\{\widetilde{\mathbf{p}}_{k|k-1}\widetilde{\mathbf{p}}_{k|k-1}^T|\mathbf{Z}_{k-1}\}\mathbf{J}_{zp}^T + E\{\mathbf{w}_k\mathbf{w}_k^T|\mathbf{Z}_{k-1}\}\\ &= \mathbf{J}_{zp}\mathbf{P}_{k|k-1}\mathbf{J}_{zp}^T + \mathbf{R}\end{aligned}\tag{5.14}$$

Innovation is a measure of disagreement between the actual and predicted state of the robot and is used to evaluate the state estimate $\widehat{\mathbf{p}}_{k|k}$ as follows [New05]:

$$\widehat{\mathbf{p}}_{k|k} = \widehat{\mathbf{p}}_{k|k-1} + \mathbf{K}(\mathbf{z}_k - \mathbf{z}_{k|k-1}) \tag{5.15}$$

where $\mathbf{K}$ is the gain. A value of $\mathbf{K}$ that minimizes the mean square estimation error is called Kalman gain [SSGS02, New05]. Using (5.15), estimation error can be calculated as follows:

$$\begin{aligned}\widetilde{\mathbf{p}}_{k|k} &= \mathbf{p}_k - \widehat{\mathbf{p}}_{k|k}\\ &= \mathbf{p}_k - \widehat{\mathbf{p}}_{k|k-1} - \mathbf{K}(\mathbf{z}_k - \mathbf{z}_{k|k-1})\\ &= \mathbf{p}_k - \widehat{\mathbf{p}}_{k|k-1} - \mathbf{K}\widetilde{\mathbf{z}}_{k|k-1}\\ &= \widetilde{\mathbf{p}}_{k|k-1} - \mathbf{K}\widetilde{\mathbf{z}}_{k|k-1}\\ &= \widetilde{\mathbf{p}}_{k|k-1} - \mathbf{K}(\mathbf{J}_{zp}\widetilde{\mathbf{p}}_{k|k-1} + \mathbf{w}_k)\\ &= (\mathbf{I} - \mathbf{K}\mathbf{J}_{zp})\widetilde{\mathbf{p}}_{k|k-1} - \mathbf{K}\mathbf{w}_k\end{aligned}\tag{5.16}$$

and its covariance matrix by the following expression:

$$\begin{aligned}\mathbf{P}_{k|k} &= E\{\widetilde{\mathbf{p}}_{k|k}\widetilde{\mathbf{p}}_{k|k}^T|\mathbf{Z}_k\}\\ &= E\{((\mathbf{I} - \mathbf{K}\mathbf{J}_{zp})\widetilde{\mathbf{p}}_{k|k-1} - \mathbf{K}\mathbf{w}_k)((\mathbf{I} - \mathbf{K}\mathbf{J}_{zp})\widetilde{\mathbf{p}}_{k|k-1} - \mathbf{K}\mathbf{w}_k)^T|\mathbf{Z}_k\}\\ &= (\mathbf{I} - \mathbf{K}\mathbf{J}_{zp})E\{\widetilde{\mathbf{p}}_{k|k-1}\widetilde{\mathbf{p}}_{k|k-1}^T|\mathbf{Z}_k\}(\mathbf{I} - \mathbf{K}\mathbf{J}_{zp})^T\\ &\quad + \mathbf{K}E\{\mathbf{w}_k\mathbf{w}_k^T|\mathbf{Z}_k\}\mathbf{K}^T\\ &\quad - (\mathbf{I} - \mathbf{K}\mathbf{J}_{zp})E\{\widetilde{\mathbf{p}}_{k|k-1}\mathbf{w}_k^T|\mathbf{Z}_k\}\mathbf{K}^T\\ &\quad - \mathbf{K}E\{\mathbf{w}_k\widetilde{\mathbf{p}}_{k|k-1}^T|\mathbf{Z}_k\}(\mathbf{I} - \mathbf{K}\mathbf{J}_{zp})^T\\ &= (\mathbf{I} - \mathbf{K}\mathbf{J}_{zp})\mathbf{P}_{k|k-1}(\mathbf{I} - \mathbf{K}\mathbf{J}_{zp})^T + \mathbf{K}\mathbf{R}\mathbf{K}^T\end{aligned}\tag{5.17}$$

To find the Kalman gain $\mathbf{K}$ the following expression for mean square estimation error is evaluated [New05]:

$$\begin{aligned}p_{mmse} &= E\{\widetilde{\mathbf{p}}_{k|k}^T\widetilde{\mathbf{p}}_{k|k}|\mathbf{Z}_k\}\\ &= Tr(\mathbf{P}_{k|k})\\ &= Tr((\mathbf{I} - \mathbf{K}\mathbf{J}_{zp})\mathbf{P}_{k|k-1}(\mathbf{I} - \mathbf{K}\mathbf{J}_{zp})^T) + Tr(\mathbf{K}\mathbf{R}\mathbf{K}^T)\end{aligned}\tag{5.18}$$

where Tr stands for trace. Differentiating (5.18) with respect to $\mathbf{K}$ and equating the result to zero:

$$\begin{aligned}
-2(\mathbf{I}-\mathbf{K}\mathbf{J}_{zp})]\mathbf{P}_{k|k-1}\mathbf{J}_{zp}^T+2\mathbf{K}\mathbf{R} &= 0\\
-\mathbf{P}_{k|k-1}\mathbf{J}_{zp}^T+\mathbf{K}\mathbf{J}_{zp}\mathbf{P}_{k|k-1}\mathbf{J}_{zp}^T+\mathbf{K}\mathbf{R} &= 0\\
\mathbf{K}(\mathbf{J}_{zp}\mathbf{P}_{k|k-1}\mathbf{J}_{zp}^T+\mathbf{R}) &= \mathbf{P}_{k|k-1}\mathbf{J}_{zp}^T\\
\mathbf{K}\mathbf{S} &= \mathbf{P}_{k|k-1}\mathbf{J}_{zp}^T\\
\mathbf{K} &= \mathbf{P}_{k|k-1}\mathbf{J}_{zp}^T\mathbf{S}^{-1} \qquad (5.19)
\end{aligned}$$

The algorithm for EKF is outlined in Algorithm 5 and its flow chart is shown in Fig. 5.1. It is divided into two phases which are processed periodically. In the first phase which is called the "time update" or the "prediction phase", system state at k is predicted based upon the previous state estimation $\widehat{\mathbf{p}}_{k-1|k-1}$ and the estimated value of the system input or control vector $\widehat{\mathbf{u}}_{k-1}$. As given by (5.9), the uncertainty of the prediction is increased by the process noise $\mathbf{J}_2\mathbf{U}_k\mathbf{J}_2^T$.

Algorithm 5 Extended Kalman filter algorithm [BDN07]

```
upon initialization do
  input $\widehat{\mathbf{p}}_{0|0}$
  input $\mathbf{P}_{0|0}$
  $k \leftarrow 1$
end upon
loop
  if new control input available then
    // time update stage
    // predicting future position and its uncertainty based on the new control input
        $\widehat{\mathbf{p}}_{k|k-1} \leftarrow \mathbf{f}(\widehat{\mathbf{p}}_{k-1|k-1}, \widehat{\mathbf{u}}_k, k)$
        $\mathbf{U}_k \leftarrow \mathbf{J}_u\mathbf{\Sigma}_v\mathbf{J}_u^T$
        $\mathbf{P}_{k|k-1} \leftarrow \mathbf{J}_1\mathbf{P}_{k-1|k-1}\mathbf{J}_1^T + \mathbf{J}_2\mathbf{U}_k\mathbf{J}_2^T$
  else
        $\widehat{\mathbf{p}}_{k|k-1} \leftarrow \widehat{\mathbf{p}}_{k-1|k-1}$
        $\mathbf{P}_{k|k-1} \leftarrow \mathbf{P}_{k|k-1}$
  end if
  if new observation available then
    // updating position based on new observation
        $\mathbf{K} \leftarrow \mathbf{P}_{k|k-1}\mathbf{J}_{zp}\mathbf{S}^{-1}$
        $\mathbf{R} \leftarrow \mathbf{J}_z\mathbf{\Sigma}_i\mathbf{J}_z^T$
        $\widehat{\mathbf{p}}_{k|k} \leftarrow \widehat{\mathbf{p}}_{k|k-1} + \mathbf{K}(\mathbf{z}_k - \mathbf{z}_{k|k-1})$
        $\mathbf{P}_{k|k} \leftarrow (\mathbf{I}-\mathbf{K}\mathbf{J}_{zp})\mathbf{P}_{k|k-1}(\mathbf{I}-\mathbf{K}\mathbf{J}_{zp})^T + \mathbf{K}\mathbf{R}\mathbf{K}^T$
  else
        $\widehat{\mathbf{p}}_{k|k} \leftarrow \widehat{\mathbf{p}}_{k|k-1}$
        $\mathbf{P}_{k|k} \leftarrow \mathbf{P}_{k|k-1}$
  end if
  $k \leftarrow k+1$
end loop
```

The second phase, also known as the "measurement update" or the "correction phase", corrects prediction using the noisy measurements. The Kalman gain **K** (5.19) gives the proportion between how much the predicted state can be trusted, and up to what extent the new measurements can be taken into account in a way that results in a minimum mean square estimation error [New05]. The observation uncertainty **R** is based on the assumption that errors in estimating landmark location in the left and right camera are independent identically distributed Gaussian random variables having zero mean and covariance matrix $\mathbf{\Sigma}_i$. The robot observation and its uncertainty analysis is given in Section 5.2.

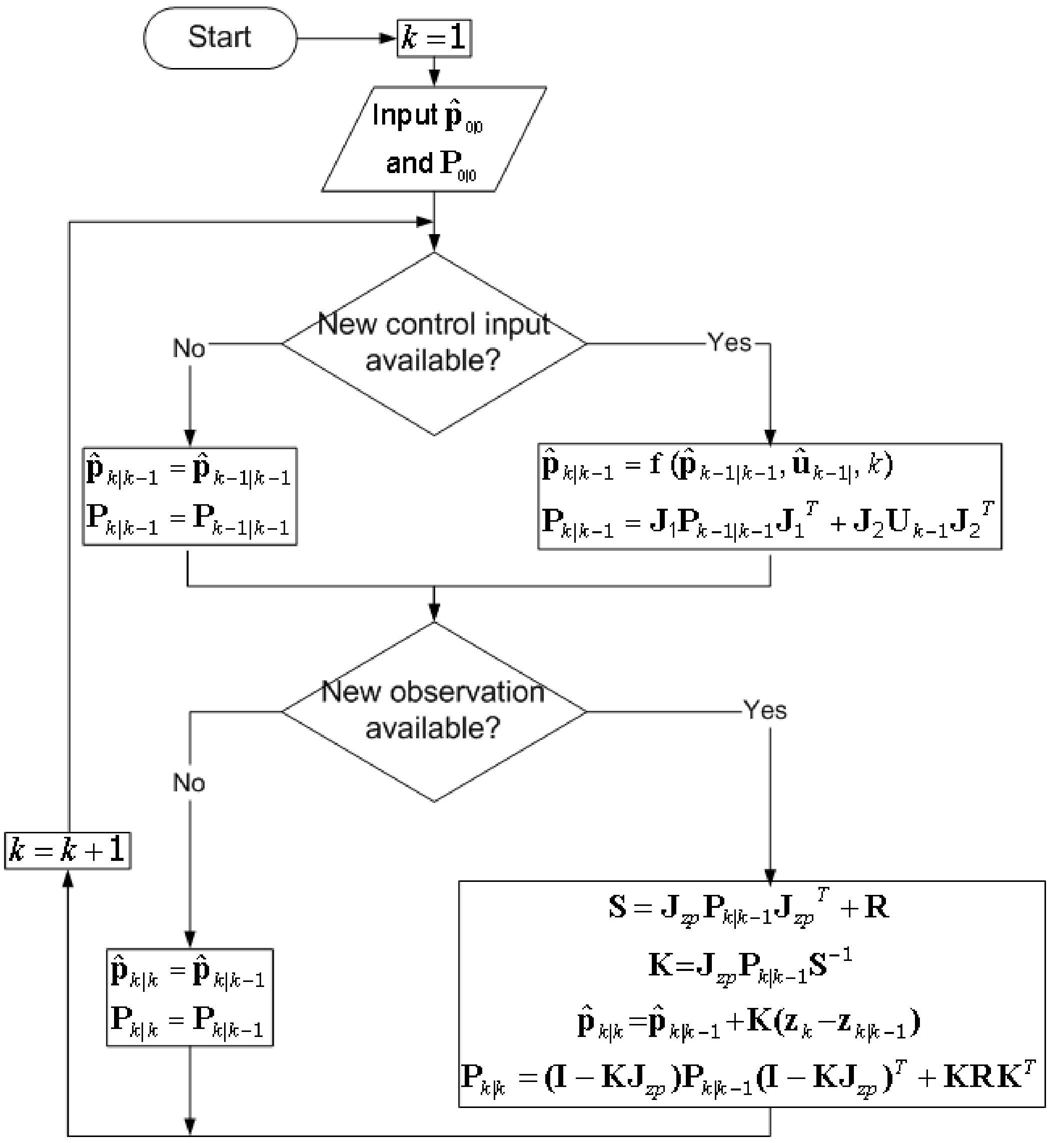

Start
$k=1$
Input $\hat{\mathbf{p}}_{0|0}$ and $\mathbf{P}_{0|0}$
New control input available?
No
Yes
$\hat{\mathbf{p}}_{k|k-1} = \hat{\mathbf{p}}_{k-1|k-1}$
$\mathbf{P}_{k|k-1} = \mathbf{P}_{k-1|k-1}$
$\hat{\mathbf{p}}_{k|k-1} = \mathbf{f}(\hat{\mathbf{p}}_{k-1|k-1}, \hat{\mathbf{u}}_{k-1|}, k)$
$\mathbf{P}_{k|k-1} = \mathbf{J}_1\mathbf{P}_{k-1|k-1}\mathbf{J}_1^T + \mathbf{J}_2\mathbf{U}_{k-1}\mathbf{J}_2^T$
New observation available?
No
Yes
$k=k+1$
$\hat{\mathbf{p}}_{k|k} = \hat{\mathbf{p}}_{k|k-1}$
$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1}$
$\mathbf{S} = \mathbf{J}_{zp}\mathbf{P}_{k|k-1}\mathbf{J}_{zp}^T + \mathbf{R}$
$\mathbf{K} = \mathbf{J}_{zp}\mathbf{P}_{k|k-1}\mathbf{S}^{-1}$
$\hat{\mathbf{p}}_{k|k} = \hat{\mathbf{p}}_{k|k-1} + \mathbf{K}(\mathbf{z}_k - \mathbf{z}_{k|k-1})$
$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}\mathbf{J}_{zp})\mathbf{P}_{k|k-1}(\mathbf{I} - \mathbf{K}\mathbf{J}_{zp})^T + \mathbf{K}\mathbf{R}\mathbf{K}^T$

Figure 5.1: Flow chart of the EKF [New05]

5.1.3 Kidnapped Robot Problem

An example of the kidnapped robot problem is illustrated in Fig. 5.2. Here the robot starts at point 'A' and moves along the linear path in reverse direction towards point 'B'. The actual path of the robot is shown by the blue line in Fig. 5.2. At point 'A' the robot calculates its global position using the single landmark based global position estimation method. The robot then tracks this position until it reaches point 'B'. The estimated path between point 'A' and point 'B' is shown in black color. After reaching point 'B', the robot is brought back to point 'A' without giving it the information that it is being displaced. The robot has a high belief that it is at point 'B', while it is moved to point 'A'. This is called a kidnapped problem and the robot has to detect it.

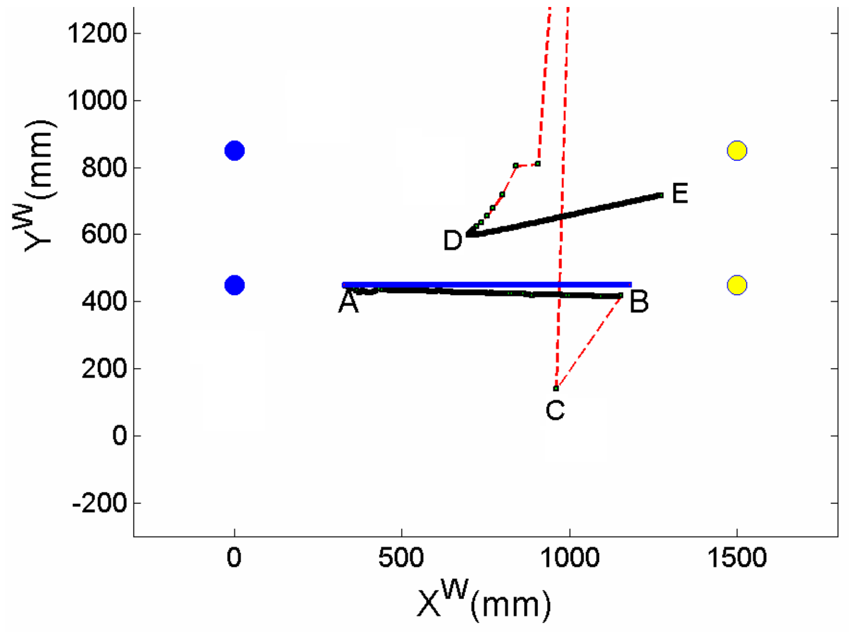


Figure 5.2: Illustrating the kidnapped robot problem. The robot is kidnapped after reaching point 'B'

The robot initially covers this distance between 'A' and 'B' in 100 steps and then repeats the whole process in another 100 steps. Between point 'A' and 'B', the position belief and estimate are in sync and the update process is working properly. The kidnap causes a disagreement between position belief and the actual estimate, which results in haphazard position estimation. The robot estimates that it is a point 'C'. After several jumps here and there the position estimate steadies at point 'D'. From point 'D' to 'E' the robot position estimates lie along a line but the error still remain high.

Fig. 5.3 shows pose error along with the $\pm 3\sigma$ uncertainty bound. The error is plotted in blue, whereas, the $\pm 3\sigma$ uncertainty bound is shown in red. As can be clearly seen from this figure that until step 100, the error is well-bounded by the position uncertainty. At step 100 the robot is kidnapped (brought back to point 'A'). Large error in all three components of the robot pose (x, y and θ) is the result of the kidnap. Beyond step 100, the robot has a false belief about its position.

Such a situation has to be identified and consequently the robot has to be declared lost. Once a kidnap is identified, corrective measures can be applied, which include adjusting the robot

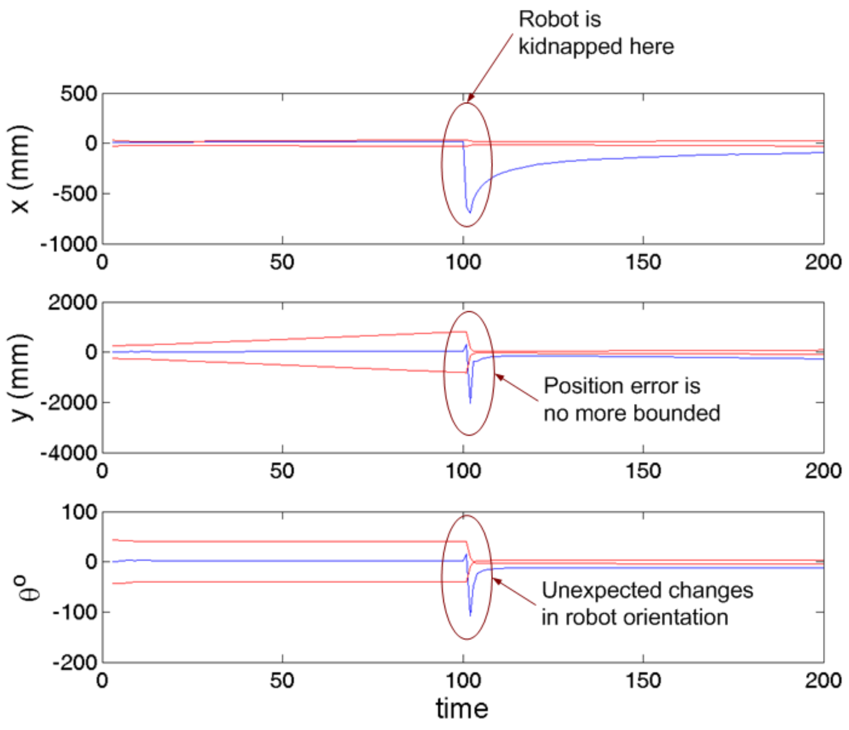


Figure 5.3: Error in x, y, and z components of the robot pose and its corresponding uncertainty for the kidnapped robot case. At step 100, the robot is kidnapped. Beyond this point the robot has a false belief of its position

position uncertainty or initiating the global position estimation. An unexpected situation can be detected by monitoring the innovation sequence (the difference between actual and expected observation) i.e. by implementing a validation gate [BS87]. The observation residual or innovation for the above example is shown in Fig.5.4. Whenever the robot makes an observation, it is also predicting what it should see from its prediction. As stated earlier, this difference in the actual and predicted observation is way to correct the predicted position estimate. The innovation is shown in blue color, whereas, its $\pm\sigma$ uncertainty bound is shown in red. From step 1 to step 100 the innovation sequence is well-bounded, however, as soon as the robot is displaced, the measurements are unexpected and hence unbounded. The situation in this figure is such that the robot position uncertainty is not adjusted or the global position estimation is not initiated.

The implementation of a validation gate based on the innovation is effectively comparing the predicted and actual observations. The objective is to determine the normalized difference between the predicted and actual observations. This difference is a quadratic form known as the squared Mahalanobis distance and is given by the following equation [Cro95, RB00a]:

$$e^2 = \widetilde{\mathbf{z}}_{k|k-1}^T \mathbf{S}^{-1} \widetilde{\mathbf{z}}_{k|k-1} \tag{5.20}$$

With Gaussian innovation, the resulting distribution of e^2 is chi-square [RB00a]. The values of e^2 for each step of the above example are shown in Fig. 5.5. The normalized error stays very low until step 100. The big jump in error is the result of kidnapping the robot. The error stays high and unpredictable for several observation. It is clear from the figure that with the help of this test the algorithm can decide if a given observation is unexpected or not. Several

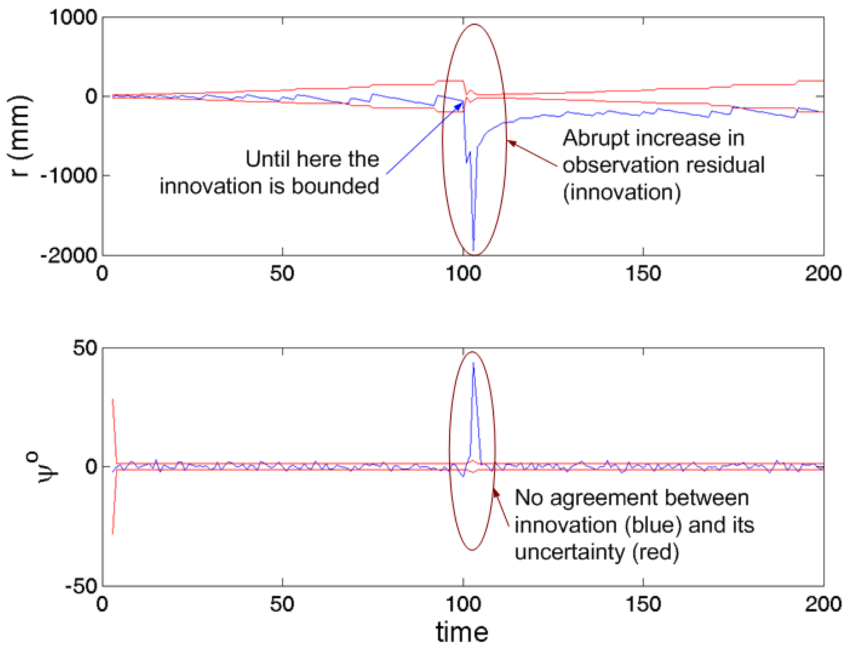


Figure 5.4: Observation residual or innovation for the case shown in Fig. 5.2

unexpected observations will further increase the probability of a kidnap. Fig. 5.6 shows robot position estimates when the robot position is readjusted after detecting an unexpected behavior using (5.20).

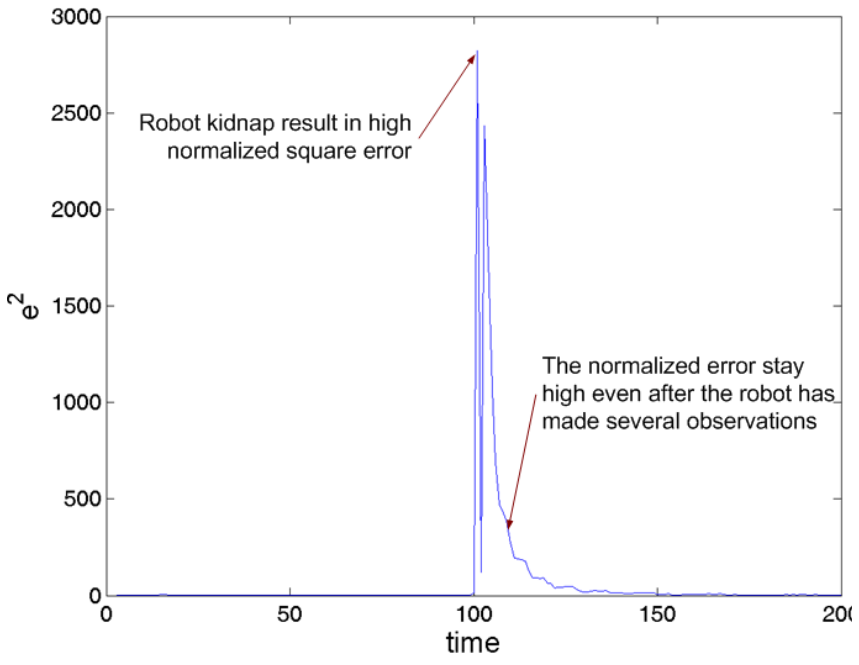


Figure 5.5: The Mahalanobis squared distance for example shown in Fig. 5.2

In this example when the filter detected that something is wrong, it ignored that observation

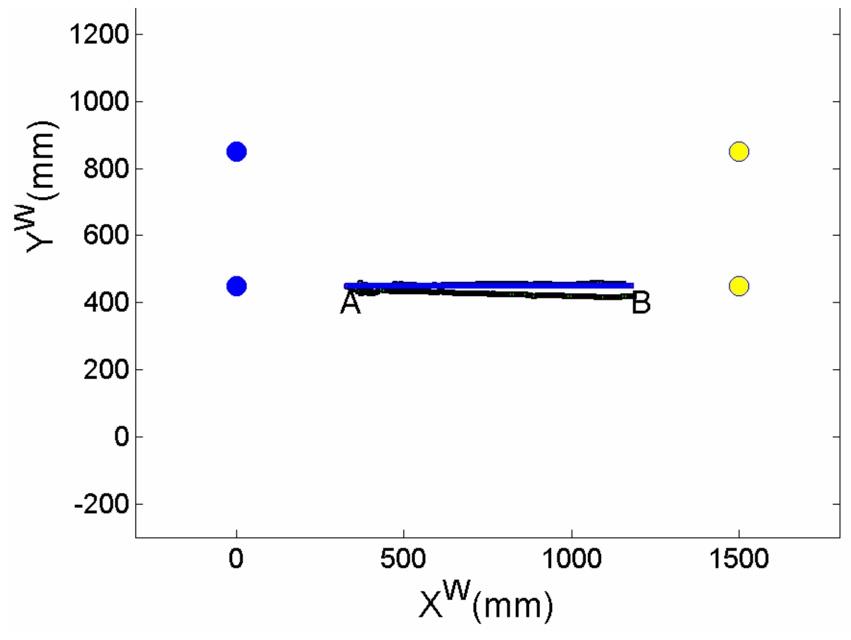


Figure 5.6: Repeating the estimation shown in Fig. 5.2. Here global position estimation is initiated after detecting that the robot is being kidnapped

and waited for a next one. After receiving several unexpected observation, the robot is declared lost and global position estimation is initiated. Until the global position is estimated, the robot position is updated only based on odometry information.

In this example observations are frequently made. The innovation increases when observations are not frequently made. This increase is accounted for by increase in the innovation covariance, which is dependant on the position uncertainty as given by (5.14). This means that the validation gate based on the normalized distance will work even if observations are not frequently available. However, as noted by Negenborn [Neg03] the unexpected observation are not only due to kidnapped robots, other sources may include sensor failure or increased sensor noise.

5.1.4 Representing Uniform Position Belief

Under global uncertainty the robot waits until its position is estimated using one of the global self-localization approaches. An alternative approach for representing the global uncertainty is to use a very high uncertainty for the initial position so that it approximates a uniform distribution. However, it is difficult to decide the amount of initial uncertainty so that it approximates a uniform distribution. The uncertainty grows even further until observations are made. Once observations are made, the uncertainty will drop. This decrease is inversely proportional to the initial uncertainty. Furthermore, the amount of observation residual that is taken into account is also decided on the basis of initial uncertainty as given by (5.19). An equivalent approach to approximate the uniform distribution is that the robot simply drives around until it receives its first observation of landmark features in its environment [Neg03]. When this measurement arrives it is taken into account in its totality. In the limiting case, when the initial uncertainty approaches infinity the innovation covariance (5.14) can be written as:

$$\mathbf{S} \approx \mathbf{J}_{zp}\mathbf{P}_{k|k-1}\mathbf{J}_{zp}^T \tag{5.21}$$

With the modified innovation covariance the Kalman gain can be approximated by the following expression:

$$\begin{aligned}
\mathbf{K} &= \mathbf{P}_{k|k-1}\mathbf{J}_{zp}^T\mathbf{S}^{-1} \\
&\approx \mathbf{P}_{k|k-1}\mathbf{J}_{zp}^T(\mathbf{J}_{zp}\mathbf{P}_{k|k-1}\mathbf{J}_{zp}^T)^{-1} \qquad (5.22) \\
&= \mathbf{J}_{zp}^{-1} \qquad (5.23)
\end{aligned}$$

and (5.17) becomes:

$$\mathbf{P}_{k|k} \approx \mathbf{J}_{zp}^{-1}\mathbf{R}(\mathbf{J}_{zp}^{-1})^T \tag{5.24}$$

With the modified Kalman gain the position update can be written as follows [Neg03]:

$$\begin{aligned}
\widehat{\mathbf{p}}_{k|k} &= \widehat{\mathbf{p}}_{k|k-1} + \mathbf{K}(\mathbf{z}_k - \mathbf{z}_{k|k-1}) \\
&\approx \widehat{\mathbf{p}}_{k|k-1} + \mathbf{J}_{zp}^{-1}(\mathbf{z}_k - \mathbf{z}_{k|k-1}) \\
&= \widehat{\mathbf{p}}_{k|k-1} + \mathbf{J}_{zp}^{-1}\mathbf{z}_k - \mathbf{J}_{zp}^{-1}\mathbf{z}_{k|k-1}) \\
&= \widehat{\mathbf{p}}_{k|k-1} + \mathbf{J}_{zp}^{-1}\mathbf{z}_k - \widehat{\mathbf{p}}_{k|k-1} \\
&= \mathbf{J}_{zp}^{-1}\mathbf{z}_k \qquad (5.25)
\end{aligned}$$

However, (5.25) can only be used if $\mathbf{J}_{zp}$ is invertible. Further discussion on the initialization issue is presented in the next chapter, where different alternatives are evaluated experimentally.

5.2 Robot Observation

The values x^{C0} and y^{C0} with respect to a landmark $\mathbf{p}_l = [x_l\ y_l]^T$ (see Section 3.2) are used to construct robot observation vector $\mathbf{z}_k$. The construction of the robot observation vector is illustrated in Fig. 5.7 and is given by (5.26).

$$\mathbf{z}_k = \begin{bmatrix} r_k \\ \psi_k \end{bmatrix} = \begin{bmatrix} \sqrt{(x^{C0})^2 + (y^{C0})^2} \\ atan2(y^{C0}, x^{C0}) \end{bmatrix} \tag{5.26}$$

Equation (5.12) is making use of observation prediction. From its predicted position $\widehat{\mathbf{p}}_{k|k-1}$, robot predicts what it may see, which is used for position update. Construction of the robot observation prediction vector is illustrated in Fig. 5.7 and is given by the following expression:

$$\mathbf{z}_{k|k-1} = \begin{bmatrix} r_{k|k-1} \\ \psi_{k|k-1} \end{bmatrix} = \begin{bmatrix} \sqrt{(x_l - x_{k|k-1})^2 + (y_l - y_{k|k-1})^2} \\ atan2(y_l - y_{k|k-1}, x_l - x_{k|k-1}) - \theta_{k|k-1} \end{bmatrix} \tag{5.27}$$

For robot observation uncertainty it is assumed that the error $\begin{bmatrix}\widetilde{u}_l & \widetilde{u}_r\end{bmatrix}^T$ in $\begin{bmatrix}u_l & u_r\end{bmatrix}^T$ is zero mean Gaussian. The covariance matrix of this error is give by the following expression:

$$\mathbf{\Sigma}_i = E\{\begin{bmatrix}\widetilde{u}_l & \widetilde{u}_r\end{bmatrix}^T \begin{bmatrix}\widetilde{u}_l & \widetilde{u}_r\end{bmatrix}\} = \sigma_{uu}^2 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{5.28}$$

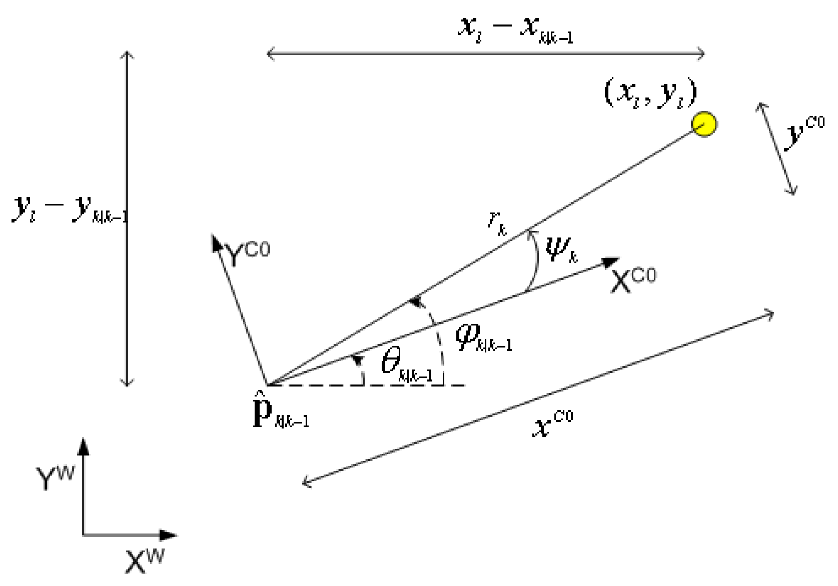


Figure 5.7: Illustration of the robot observation: The robot stereo vision system is used to estimate range and bearing with respect to a landmark feature. Four distinct landmark features are shown here

where σ_{uu}^2 is the variance of u_l or u_r. In deriving (5.28) it is assumed that $\widetilde{u}_l$ and $\widetilde{u}_r$ are not correlated. Error in u_l and u_r is propagated to observation $\mathbf{z}_k$ by the transformation (5.26) through (3.4) and (3.5). It is assumed that these systems can be represented by the first two terms of Taylor series expansion around estimated value of u_l and u_r. This assumption leads to following expression for observation model:

$$\mathbf{z}_k \approx \widehat{\mathbf{z}}_k + \mathbf{J}_z \begin{bmatrix}\widetilde{u}_l & \widetilde{u}_r\end{bmatrix}^T \tag{5.29}$$

where

$$\mathbf{J_z} = \begin{bmatrix} \frac{-b}{r_k d^2}(fx^{C0} - (u_r - o_u)y^{C0}) & \frac{b}{r_k d^2}(fx^{C0} - (u_l - o_u)y^{C0}) \\ \frac{b}{r_k^2 d^2}(x^{C0}(u_r - o_u) + x^{C0}f) & \frac{-b}{r_k^2 d^2}(x^{C0}(u_l - o_u) + y^{C0}f) \end{bmatrix} \tag{5.30}$$

is the jacobian of (5.26) with respect to $\begin{bmatrix}u_l & u_r\end{bmatrix}^T$ and $\widehat{\mathbf{z}}_k$ is the estimated observation. From (5.29) the observation error can be written as follows:

$$\begin{aligned} \widetilde{\mathbf{z}}_k &= \mathbf{z}_k - \widehat{\mathbf{z}}_k \\ &= \mathbf{J}_z \begin{bmatrix}\widetilde{u}_l & \widetilde{u}_r\end{bmatrix}^T \end{aligned} \tag{5.31}$$

Using (5.31) expression for observation covariance matrix is derived as follows:

$$\begin{aligned} \mathbf{R} &= E\{\widetilde{\mathbf{z}}_k \widetilde{\mathbf{z}}_k^T\} \\ &= E\{(\mathbf{J}_z \begin{bmatrix}\widetilde{u}_l & \widetilde{u}_r\end{bmatrix}^T)(\mathbf{J}_z \begin{bmatrix}\widetilde{u}_l & \widetilde{u}_r\end{bmatrix}^T)^T\} \\ &= \mathbf{J}_z E\{\begin{bmatrix}\widetilde{u}_l & \widetilde{u}_r\end{bmatrix}^T \begin{bmatrix}\widetilde{u}_l & \widetilde{u}_r\end{bmatrix}\}\mathbf{J}_z^T \\ &= \mathbf{J}_z \mathbf{\Sigma}_i \mathbf{J}_z^T \end{aligned} \tag{5.32}$$

Substitution of $\mathbf{\Sigma}_i$ and $\mathbf{J}_z$ from (5.28) and (5.30) into (5.32) results in the following expressions for different elements of $\mathbf{R}$:

$$
\begin{aligned}
R_{11} &= \frac{b^2\sigma_{uu}^2}{r_k^2 d^4}(2f^2(x^{C0})^2 + (y^{C0})^2\{(u_l - o_u)^2 + (u_r - o_u)^2\} - 2fx^{C0}y^{C0}(u_l + u_r - 2o_u)) \\
R_{12} &= \frac{-b^2\sigma_{uu}^2}{r_k^3 d^4}(f((x^{C0})^2 - (y^{C0})^2)(u_l + u_r - 2o_u) \\
&\quad +2f(x^{C0})(y^{C0}) - x^{C0}y^{C0}\{(u_l - o_u)^2 + (u_r - o_u)^2\}) \\
R_{21} &= R_{12} \\
R_{22} &= \frac{b^2\sigma_{uu}^2}{r_k^4 d^4}((x^{C0})^2\{(u_l - o_u)^2 + (u_r - o_u)^2\} + 2f^2(y^{C0})^2 + 2fx^{C0}y^{C0}(u_l + u_r - 2o_u))
\end{aligned}
$$

where R_{ij} represent an element at ith row and jth column. The error model discussed above captures the uncertainty due to image quantization, yet it fails to handle gross segmentation problems [MS87].

5.3 Particle Filter Based Position Estimation

In the following implementation of PF based position estimation algorithm is discussed. The discussion here builds on theoretical background of Section 2.4.2. Details of the algorithm and motivation behind its use can be found in [Deu07, BDN07].

The basic concept of the algorithm is a Monte Carlo approximation with Sequential Importance Sampling (SIS). At startup, random particles are generated from the initial distribution and their weights are set to the reciprocal of the number of particles. In each iteration, the weight of a particle is calculated by multiplying its old weight with the probability that this particle represents the real state in accordance with the new sensor data. After weight normalization, the new state is calculated using these particles. The SIS leads to degeneration, since the current weight of particles influences the weight update process, strong particles dominate the overall belief even if sensor readings suggest otherwise. This problem could be avoided by replacing SIS with SIR. This introduces another source of degeneration, which could be addressed by re-injection as discussed in Section 2.4.2.

Algorithm 6 shows a classical PF with SIR. During initialization, N particles are drawn from $p(\mathbf{p}_0)$. If the robot has no knowledge of its position the distribution $p(\mathbf{p}_0)$ is uniform over the entire state space. When new control input is available, temporary particles $\tilde{\mathbf{p}_k}^{(i)}$ are generated according to the distribution given by (5.33). Using the actual sensor data $\mathbf{z}_k$, the weight $\tilde{w_k}^{(i)}$ for the particles is calculated in (5.34). In (5.37), the normalized weights of (5.36) are used to calculate the state estimation for $\mathbf{p}_k$ conditioned on all previous measurements $\mathbf{z}_{1:k}$. As a preparation for the re-sampling step, all tuples $\left\{\tilde{\mathbf{p}_k}^{(i)}, \tilde{w_k}^{(i)}\right\}$ are sorted in descending order with the highest weight-value first. During re-sampling, N times a random number j is drawn in such a way that tuples with a higher weight are preferred. For each j, the corresponding $\tilde{\mathbf{p}_k}^{(j)}$ is drawn and assigned to the final state $\mathbf{p}_k^{(i)}$ as described by (5.38). All final state vectors are assigned a uniform weight. To avoid local maxima, finally, T particles are replaced with new uniformly distributed random particles. T is much smaller than N, typically 100 times or more.

Algorithm 6 Particle filter algorithm [BDN07]

```
upon initialization do
  // generate random particles
  for $i = 1$ to $N$ do
    generate $\mathbf{p}_0^{(i)} \sim p(\mathbf{p}_0)$
  end for
  $k \leftarrow 1$
end upon
loop
  if new control input available then
    for $i = 1$ to $N$ do
      // generate particle
            $$\tilde{\mathbf{p}_k}^{(i)} \leftarrow p\left(\mathbf{p}_k|\mathbf{u}_{k-1}, \mathbf{p}_{k-1}^{(i)}\right) \quad (5.33)$$
      // calculate weight/probability of particle
            $$\tilde{w_k}^{(i)} \leftarrow p\left(\mathbf{z}_k|\tilde{\mathbf{p}_k}^{(i)}\right) \quad (5.34)$$
    end for
  end if
            $$w_{sum} \leftarrow \left[\sum_{j=1}^{N} \tilde{w_k}^{(j)}\right] \quad (5.35)$$
  // normalize weights
  for $i = 1$ to $N$ do
            $$\tilde{w_k}^{(i)} \leftarrow \frac{\tilde{w_k}^{(i)}}{w_{sum}} \quad (5.36)$$
  end for
  // estimate current state
            $$E(\mathbf{p}_k|\mathbf{z}_{1:k}) \leftarrow \sum_{j=1}^{N} \tilde{\mathbf{p}}_k^{(j)} \tilde{w}_k^{(j)} \quad (5.37)$$
  // resample particles
  sort $\left\{\tilde{\mathbf{p}_k}^{(i)}, \tilde{w_k}^{(i)}\right\}_{i=1}^{N}$ such that $\tilde{w_k}^{(i)} > \tilde{w_k}^{(i+1)}$
  for $i = 1$ to $N$ do
    draw j with probability $\tilde{w_k}$
            $$\left\{\mathbf{p}_k^{(i)}, \frac{1}{N}\right\} \leftarrow \left\{\tilde{\mathbf{p}_k}^{(j)}, \tilde{w_k}^{(j)}\right\} \quad (5.38)$$
  end for
  // re-inject random particles
  for $i = 1$ to $T$ do
    replace $\mathbf{p}_k^{(i)}$ with new random particle
  end for
  $k \leftarrow k + 1$
end loop
```

Chapter 6

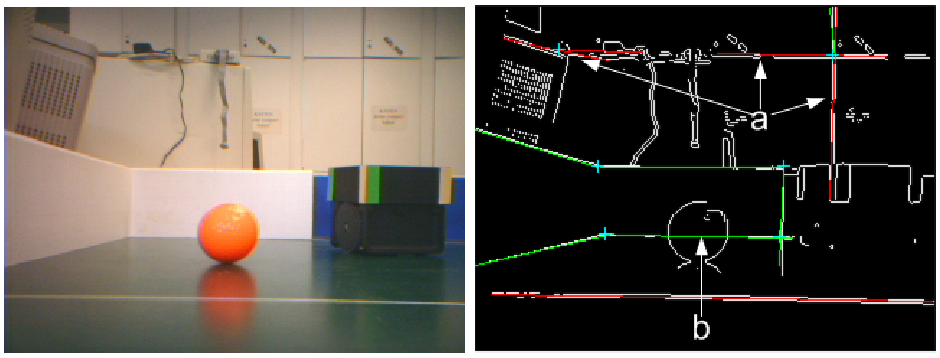
Results and Evaluation

In the following chapter the performance of all algorithms given in the previous chapters is tested with simulations. The parameters for simulations have been chosen to match the Tinyphoon robot [NM05, NCB$^+$06]. For example, matching parameters of Tinyphoon, the simulated robot is a two wheeled differential drive, with wheel base of 75 mm. The stereo vision system is mounted at a height of 70 mm. The separation between the two cameras (stereo baseline) is 30 mm. This comes from the construction of the robot as the maximum allowed size of the robot is 75 mm. The cameras are capable to provide images of resolution 640×480 pixels or 320×240 pixels. For experiments reported in this thesis the resolution was set at 320×240 pixels. The size of the robot and dimensions of the robot environment conforms with the Federation of International Robot-soccer Association (FIRA) Micro Robot World Cup Soccer Tournament (MiroSot) Small League [FIR07b].

6.1 Feature Extraction Results

Beginning with Fig. 6.1(a) which shows a color image of the soccer field in our laboratory. The ball and a robot can be seen on a flat surface. Field walls, the blue goal and the center line can also be seen in the image. Fig. 6.1(b) shows the edge map detected by applying the Sobel edge detector to the y-channel of this image. The detected line segments, corners and line intersections are superimposed on the edge map. The line segments which belong to field markings are shown red in Fig. 6.1(b). Corners and line intersections are shown with a (+) sign. It can be seen in the figure that certain features have been detected that do not belong to objects in the field. These features are marked with arrows starting at 'a' and can be discarded by using information based on the computed range information. A convincing result of the algorithm is the detection of the borderline (occluded by the ball, marked as 'b') as one line segment. This was achieved by the GBHT which classified all the pixels belonging to the two line segments in one group. The two line segments were merged using the two thresholds to merge line segments and pixels within the same group of collinear pixels.

Images of the yellow goal are shown in Fig. 6.2. In Fig. 6.2(b) two corners and one intersection are detected. The intersection point in the top-left corner (marked as 'a') is spurious. The reason for detection of this intersection point is that the two line segments are within allowed tolerance for detection of line intersection. The vertical line segments in the top middle of the image are not detected since the length of these segments is less than the minimum length



(a) (b)

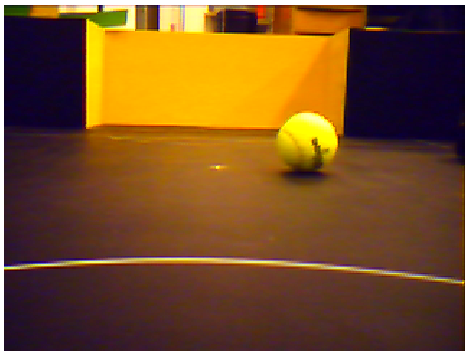
Figure 6.1: Real images with features superimposed (a) color camera image showing a robot, ball and the goal area (b) edge image with the detected lines, corners and line intersections superimposed on it. Features that are classified as field markings are shown in red color, while other line segments are shown in green color

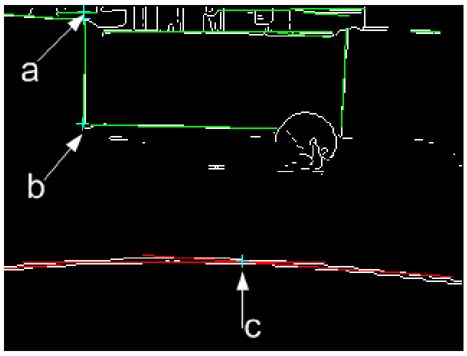
allowed. It can be seen from this figure that the line segments are successfully classified. The corner formed by the two field markings ('c') can also be seen in the figure. This type of corners can be identified from the angle between the two line segments. Fig. 6.2(c) and Fig. 6.2(d) show the same scene from slightly different position. Two robots with red objects on their top can also be seen in Fig. 6.2(c). In this case the extracted feature points are superimposed on the edge map. The line joining the top of the two robots is not separately identified but merged and seen as one single line as shown in Fig. 6.2(d). This line segment is marked as 'a'. Such features have to be detected and discarded during correspondence analysis. Similarly, Fig. 6.2(e) and Fig. 6.2(g) show two more images of the same scene with the ball at different positions. The corresponding edge maps and detected features for the two images are shown in Fig. 6.2(f) and Fig. 6.2(h) respectively. The merging of the line segments occluded by the ball in Fig. 6.2(h) is remarkable.

Results from application of the algorithm to synthetic images are shown in Fig. 6.3. An image of the field corner is seen in Fig. 6.3(a). One can see a corner (pointed to by arrow 'a') formed by the two walls and another formed by the two field markings ('b'). The junction formed by the field marking and the border can also be seen in Fig. 6.3(a). The junction is pointed to by arrow 'c' in Fig. 6.3(b). The corner formed by the two walls is blunt as solid 7 cm $\times$ 7 cm isosceles triangles are fixed at the four corners of the field to avoid the ball getting cornered [FIR07b]. The extracted features from this image are superimposed on its edge map in Fig. 6.3(b). The field marking (shown in red) are successfully classified by the algorithm. Fig. 6.3(c) and Fig. 6.3(e) show two other images of the field, while the extracted features are shown in Fig. 6.3(d) and Fig. 6.3(f) respectively. Fig. 6.3(f) shows a corner formation from two segments of the arc. This type of corners can be identified by the angle between the two line segments, since the real corners from the field lines have 90 °angle.

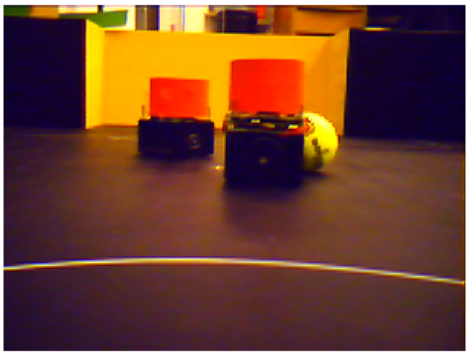
Feature extraction summary for the real world images and synthetic images is shown in Table 6.1 and Table 6.2. The summary is based on manual classification of features in images shown in this section.

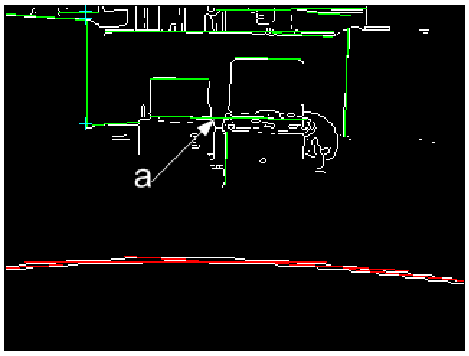
For the results presented in this section, the image resolution was set at 320 $\times$ 240 pixels. The quantization steps for θ and ρ were 1 ° and 3 pixels, respectively. The tolerance used


(a)

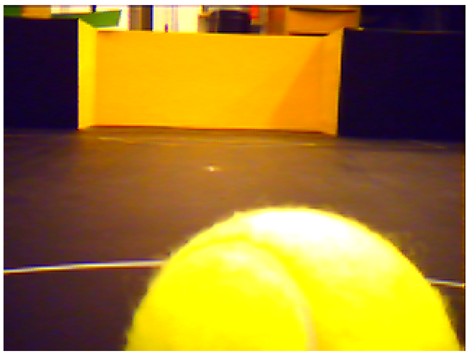


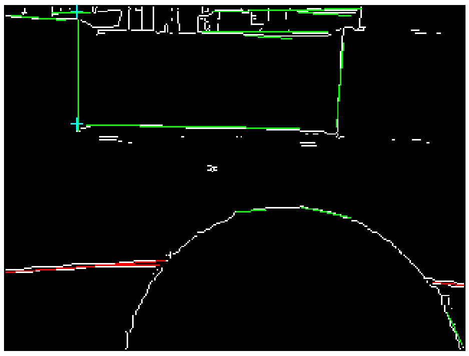
(b)

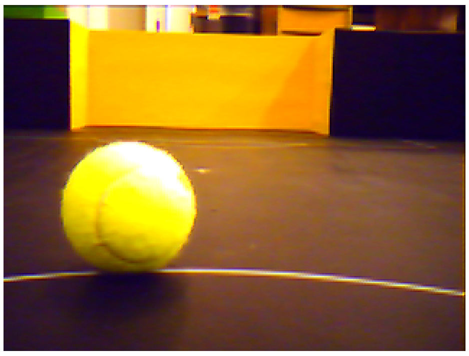

(c)



(d)


(e)


(f)


(g)

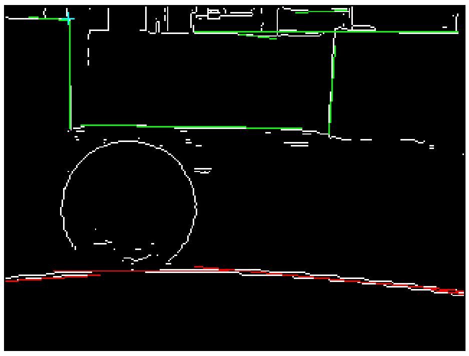

(h)

Figure 6.2: Image of the yellow goal from varying distances under bad illumination conditions. Color images are shown in (a), (c), (e) and (g), while their edge maps with the super imposed features are shown in (b), (d), (f) and (h), respectively

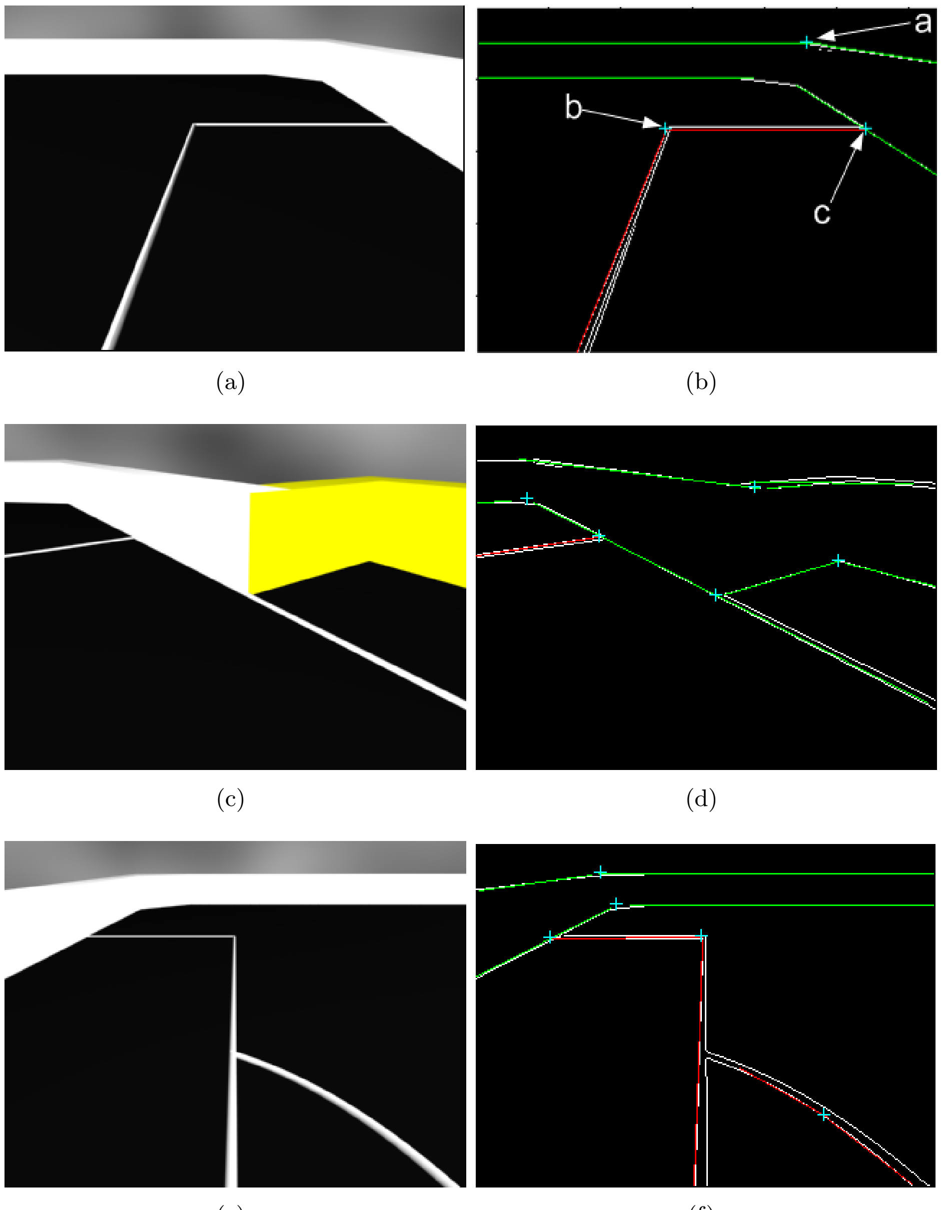


Figure 6.3: Synthetic images of the virtual field (a) colored image showing field corner (b) detected features super imposed on the edge map (c) image showing the goal area (d) edge map with detected features (e) image showing corners, field lines and arc (f) a corner formation by two line segments of the arc

Table 6.1: Feature extraction summary for real world images

	Total	**Detected**	**Missed**	**Spurious**
Lines	48	50	3	5
Field markings	10	10	0	0
Corners	16	13	5	2
Junctions	0	1	0	1
Line intersection	1	1	0	0

Table 6.2: Feature extraction summary for synthetic images

	Total	**Detected**	**Missed**	**Spurious**
Lines	20	21	1	2
Field markings	6	6	0	0
Corners	9	9	1	1
Junctions	4	4	0	0
Line intersection	0	0	0	0

for the GBHT is $\pm 5^\circ$. Gap_{min} and Gap_{max} (see Section 3.3.3) were set to 3 and 50 pixels, respectively. The minimum length of line segments was 30 pixels whereas, the tolerance for corners and junctions to be selected was 8 pixels of the rotated lines. These are empirical values and have to be tuned depending on the application.

6.2 Statistical Results for Vision Based Range Estimation

Statistical results from stereo vision based range estimation are presented in this section. Range estimation is based on the extracted feature points in images generated from an environment model of the FIRA MiroSot Small League [FIR07b]. Features used are color transitions. A sample image is shown in Fig. 6.4.

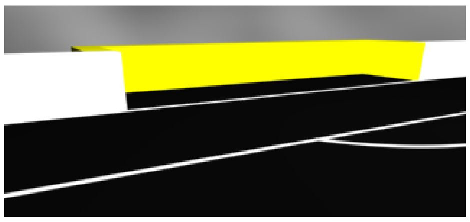


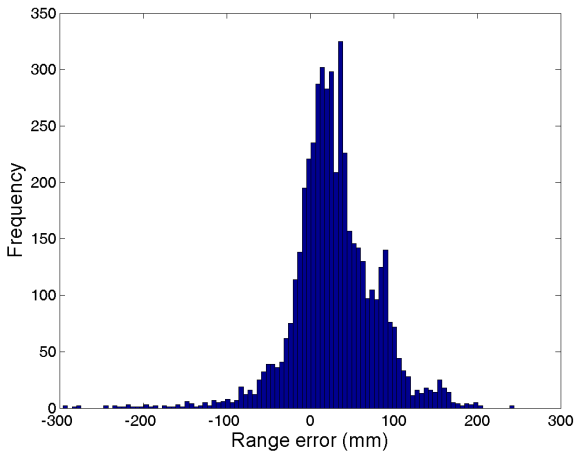
Figure 6.4: Incorrect range estimation due to concave structures in the environment

Results from range estimation error are shown in Table 6.3. The first column shows values for the mean error. The standard deviation (Std), minimum (Min) and maximum (Max) values are presented in the second, third and fourth columns. The absolute error is expressed in millimeters (mm). The normalized error is calculated by dividing the range estimation error by the actual range.

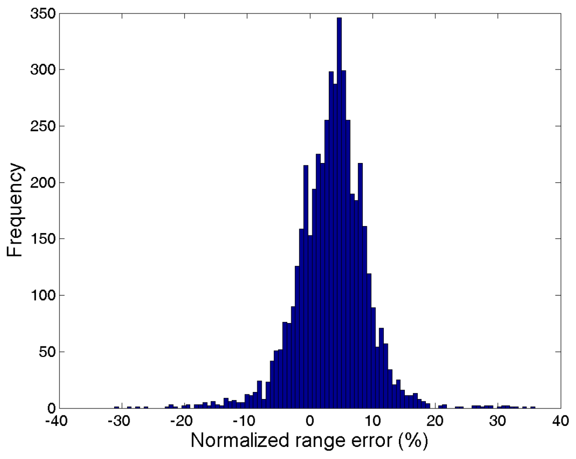
Table 6.3: Range estimation error

	Mean	Std	Min	Max
Absolute error	43.91	39.06	0.0007	295.97
Normalized error	5.18%	4.04%	0.0001%	35.91%

The results presented in Table 6.3 are based on 4893 range measurements which were made from different locations in the robot environment. Histograms for the actual and normalized range measurement error are shown in Fig. 6.5(a) and Fig. 6.5(b), respectively. In these figures, numbers along the x-axis represent the actual (millimeters) and normalized (%) range measurement error, whereas, those along the y-axis show the counts for the corresponding bin of the histogram. The histogram for normalized error shows that for some measurements the error is even higher than 20% of the true range. There are several reasons for this high error. The baseline of the stereo vision system is narrow as the construction of the robot does not allow the use of a wide baseline. This makes range estimation sensitive to segmentation problems, especially for distant points. The images used are low resolution due to the limited processing capabilities of the robot. Moreover, due to the size and concavity of the goal, it is often difficult to determine which point on the goal is being observed or to get an accurate range measurement. Fig 6.4 shows an example where range of the left goal corner is variable. This results in inconsistent ranges and inconsistent landmark positions [SSB03]. Performance improvement can be achieved by using high resolution images [BDN07].



(a)



(b)

Figure 6.5: Histogram of stereo range estimation error (a) absolute difference between true and estimated range (b) normalized range estimation error

6.3 Feature Based Global Localization Using Two Landmarks

In the following global self-localization (based on algorithms discussed in Chapter 4) is presented. It is assumed that landmarks appear in a fixed order with respect to the robot. The discussion in this section aims at evaluating performance of the global self-localization method and does not incorporate any kind of tracking of landmarks or of its own position.

This means that the position at each is step does not use information acquired at any other step.

To cover different possibilities of the instantaneous pose of the robot, 30 trials each consisting of 100 steps were conducted. These trials are grouped into five categories: motion along rectangular paths with no rotation, single point rotation of 360 °, 180 °or 90 °with no motion, motion along rectangular paths with 360 °rotation, motion along linear paths with 20 °rotation and motion along linear paths with no rotation. At every step the robot takes images of its environment, searches for landmark feature (s) and calculates its position if it finds the minimum required landmarks.

6.3.1 Motion without Rotation

Fig. 6.6 shows the simulations of the case when the robot moves along rectangular paths of different dimensions. In all these cases the robot orientation was fixed at 0 °or 180 °along the x-axis but this a priori known orientation was not used as input for position estimation. The locations from where the position estimation was attempted are marked with crosses (x) in gray color. The blue plus (+) signs mark the true location where the robot found both of the required features and was able to estimate its position and orientation. The estimated locations are marked with a blue dot (·). The difference between the estimated position and true position is indicated by the line segments shown in red. The small circles (blue and yellow) on the left and right sides show the landmark features to which the robot position is estimated.

In Fig. 6.6(a), dimensions of the rectangular path taken by the robot is 900×600. The starting step for this trial is at $(1200, 950)$ and the robot moves in counter clockwise direction along the rectangle. The step size in each trial stays the same. Orientation of the robot is fixed at 180 °(robot is facing towards left). Dimensions of the rectangular path in Fig. 6.6(b) is 750×500. The starting step is at $(1125, 900)$ and orientation is fixed at 180 °. Similarly, in Fig. 6.6(c), the starting step is at $(1050, 850)$, while the next steps are selected along the rectangular path having dimensions 600×400. The orientation of the robot in this case is set at 0 °(robot is facing towards right here). In the final trial of this category the robot starts at $(975, 800)$ with orientation at 0 °and moves along the rectangular path having dimensions of 450×300. Statistical results for the absolute error in all three components of the robot pose for the category discussed above are shown in Table 6.4. The first column shows the average (Mean) values. The standard deviation (Std), minimum (Min) and maximum (Max) values are presented in the second, third and fourth columns. The starting locations, dimensions and the position errors are expressed in millimeters (mm), whereas, the orientation error is expressed in degrees (°).

Table 6.4: Estimation error, based on the two landmarks method, in each component (x, y and θ)of the robot pose. The robot positions are along rectangular paths and there is no rotation involved

	Mean	Std	Min	Max
δx (mm)	78.56	56.88	0.17	344.06
δy (mm)	113.89	144.45	0.015	630.47
$\delta\theta$ (°)	6.40	8.25	0.0004	36.36

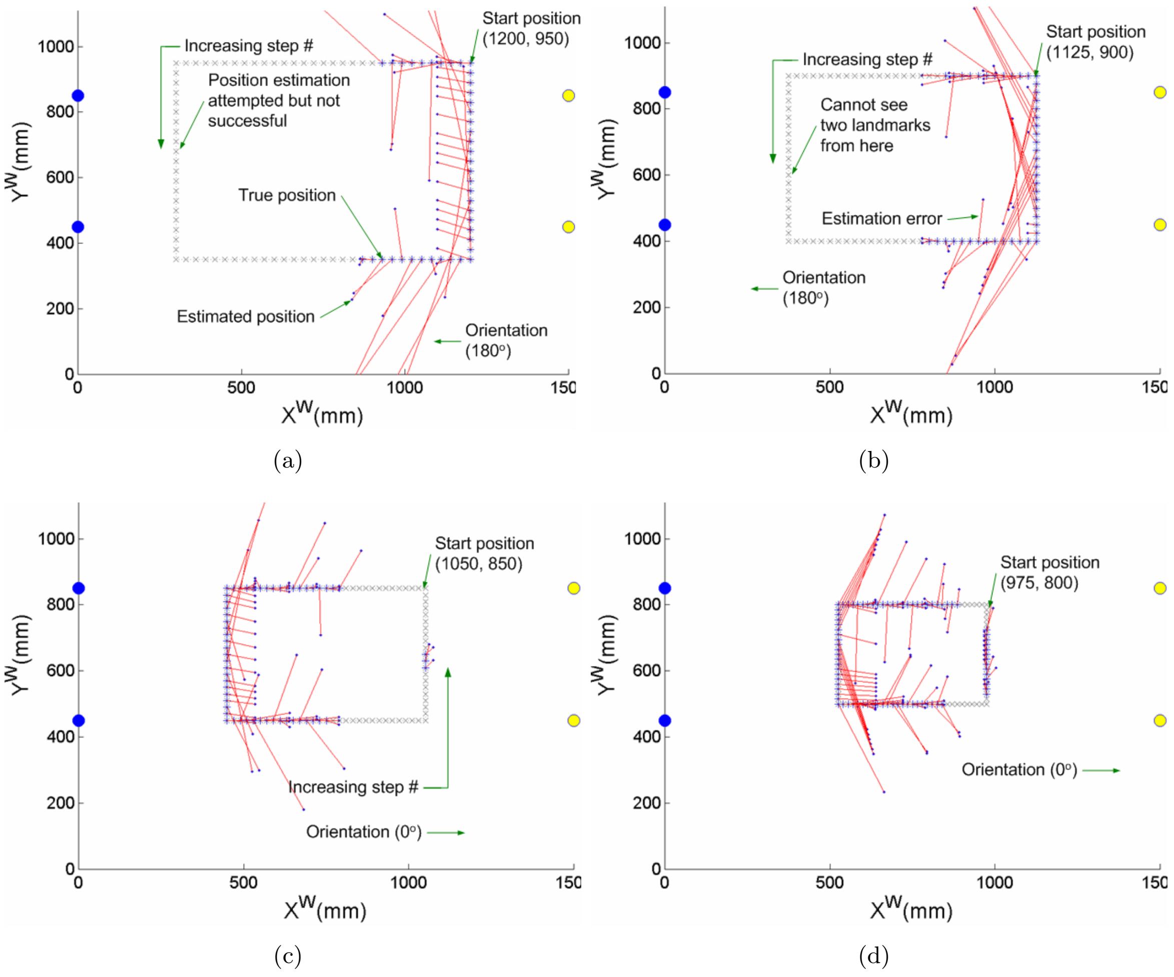


Figure 6.6: The robot is following rectangular paths of different dimensions. However, its orientation is fixed at 0 °or 180 °. Locations from where position estimation was attempted are marked with cross (x), the plus (+) signs mark the true location of successful estimation. The estimated position is indicated by dot (·) and the position error is shown by the line segment

While estimating its position, the robot also calculates its uncertainty. The advantage of uncertainty calculation is two-fold. It helps the robot decide if it wants to use an estimation or ignore it. Secondly, the uncertainty is calculated in the form of a covariance matrix, which is required to initialize a location tracker such as EKF. As can be seen from Table 6.4, the error is high. The reason of this high error is that most of the position estimates are made with respect to distant landmarks, where a small error in pixel location causes large position error. Uncertainty calculation for each position gives the robot with an option if it wants to reject estimates with high uncertainty. To compare error in each component of the robot pose with its corresponding $\pm\sigma$ error bound all measurements in all trials of this category are stacked together and shown in Fig. 6.7. The four trials are separated by the green bars and marked as (a), (b), (c), and (d). The marker (a) in Fig. 6.7 corresponds to Fig. 6.6(a), (b) to Fig. 6.6(b) and so on. Error values are plotted as bars in black color, while the corresponding $\pm\sigma$ uncertainty bound is shown in red. Position error increases as the robot is getting further from landmark features, which is captured by a corresponding increase in uncertainty. The simulations results show that the robot can estimate its position if it can find two landmarks. Even estimates with unacceptable error are useful in the sense that it gives the robot a rough estimate which can be refined later.

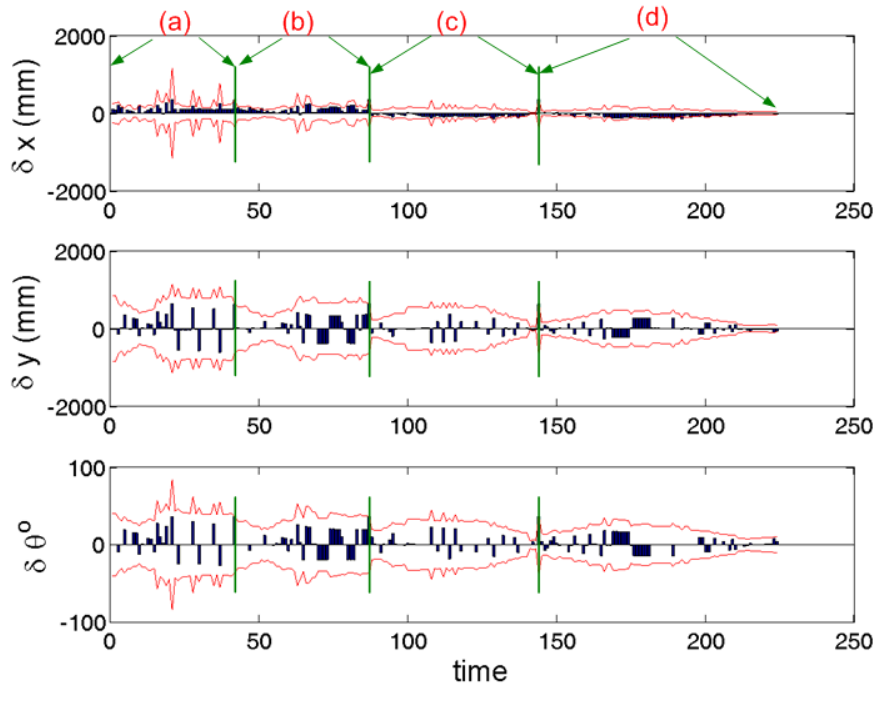


Figure 6.7: Comparison of position estimation error with corresponding $\pm\sigma$ uncertainty bound. Error from position estimations shown in Fig. 6.6 is stacked together and shown by black bars while the red curve give the $\pm\sigma$ uncertainty bound. The four trials of Fig. 6.6 are separated by the green bars and marked as (a), (b), (c), and (d). The marker (a) corresponds to Fig. 6.6(a), (b) to Fig. 6.6(b) and so on

6.3.2 Single Point Rotation

Fig. 6.8 shows results from position estimation in the category wherein the robot rotates around a single point without moving. The robot is placed at different positions in the environment. Between different steps in each trial only orientation of the robot changes, whereas, its position remains fixed. However, this knowledge is not used and global position estimation is attempted at each step. In Fig. 6.8(a), the robot is placed at $(750, 650)$. During this trial the robot rotates 360 °in clockwise direction in 100 steps. The robot orientation at step 1 is -90 ° i.e., the robot begins by facing downwards. Two landmarks are not visible to both the cameras during the first 22 steps. The first time two landmarks are extracted and position estimated is at step 23. From step 23 to step 29, the two landmarks on the left side are visible to the robot. The landmarks on the right side become available from step 73 to 79.

In Fig. 6.8(b), the robot is placed in the lower-left corner at $(350, 350)$. The robot completes a 360 °clockwise rotation in 100 steps, with its initial orientation set at -90 °. In this trial the robot position was estimated only from step 67 to step 77 with respect to landmarks on the right side. Similarly, Fig. 6.8(c) to Fig. 6.8(e) show trials where the robot is placed at the other three corners. Fig. 6.8(f) repeats the case presented in Fig. 6.8(a). In this case the robot orientation at start is -135 ° and it completes a 90 °clockwise rotation in 100 steps. The robot is able to estimate its position in almost 30% of the steps with respect to landmarks on the left side. Also in Fig.6.8(g), the robot is placed at the center but has different orientation at startup and completes 180 °clockwise rotation in 100 steps. In the last trial of this category shown in Fig.6.8(h), the robot is placed at a center right location at $(1050, 650)$ from where it completes a clockwise rotation of 90 °. The numbers for starting locations represent the absolute position of the robot and are expressed in millimeters (mm). Similarly, dimensions and position errors are also expressed in millimeters (mm).

Statistical results for the rotation-only case are shown in Table 6.5. The column headers and units in this table are the same as those in Table 6.4. The robot could estimate its pose only in less than 20 % of the locations where it searched for landmarks. A little less than 15% of the position estimates have distance between the true and estimated position greater than 400 mm. Similar to Fig. 6.7, a comparison of the estimation error for all estimates of Fig. 6.8 and the corresponding $\pm\sigma$ error bound is shown in Fig. 6.9. The eight trials of Fig. 6.8 are separated by the green bars and marked with the name of the sub-figure. For example, all the estimates of the trial shown in Fig. 6.8(a) are marked as (a) in Fig. 6.9.

Table 6.5: Estimation error, based on the two landmarks method, in each component (x, y and θ)of the robot pose. The robot is placed at different positions in its environment. Between different steps only robot orientation changes, while its position remains fixed

	Mean	Std	Min	Max
δx (mm)	58.22	37.79	1.13	231.75
δy (mm)	140.39	136.30	0.86	623.08
$\delta\theta$ (°)	7.41	6.69	0.11	28.75

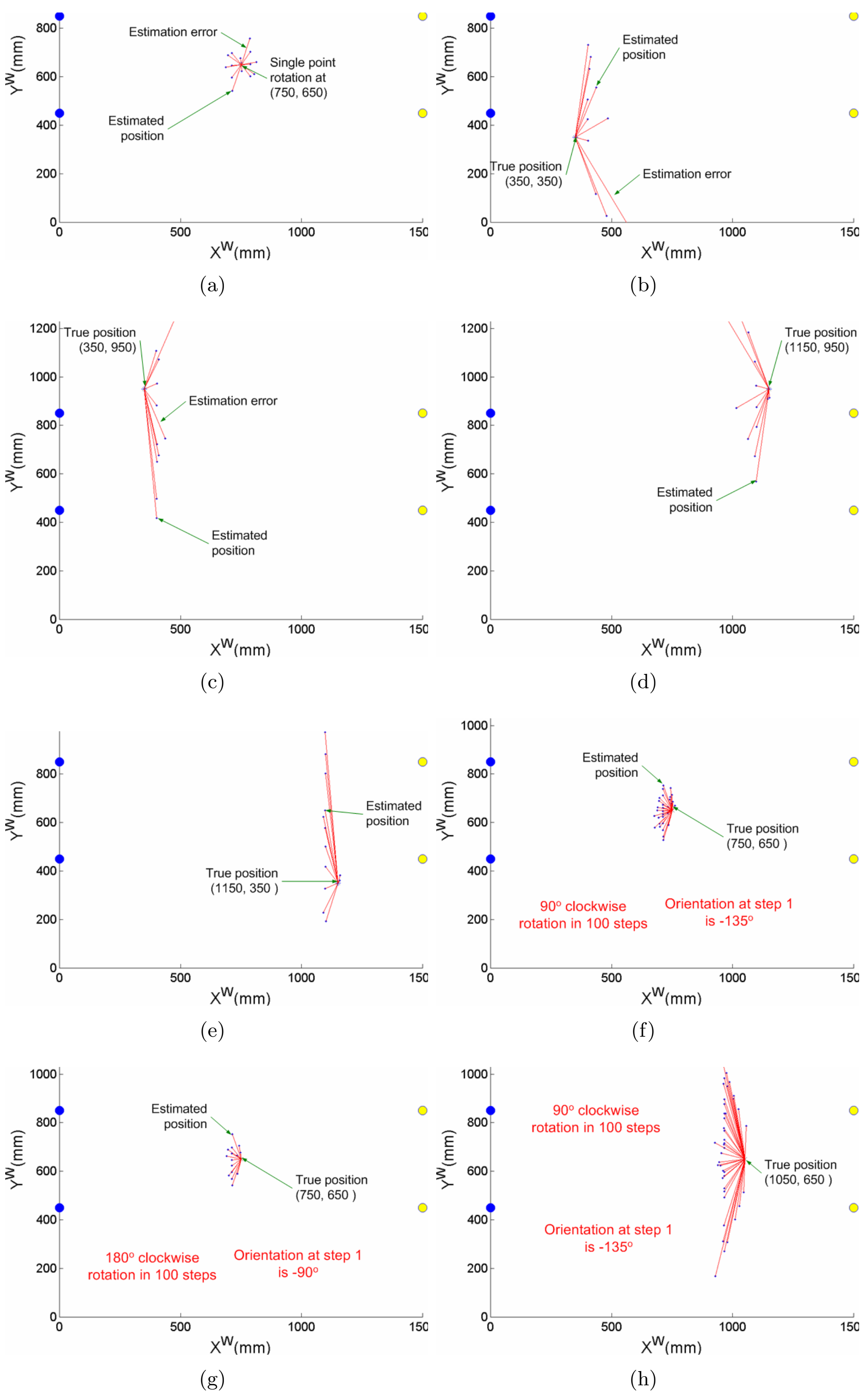


Figure 6.8: The robot is placed at different locations in its environment. In this case the robot is rotating in small steps but without any motion. The robot completes 360 °rotation in (a), (b), (c), (d) and (e), whereas, it completes 180 °rotation in (g) and 90 °in (f) and (h)

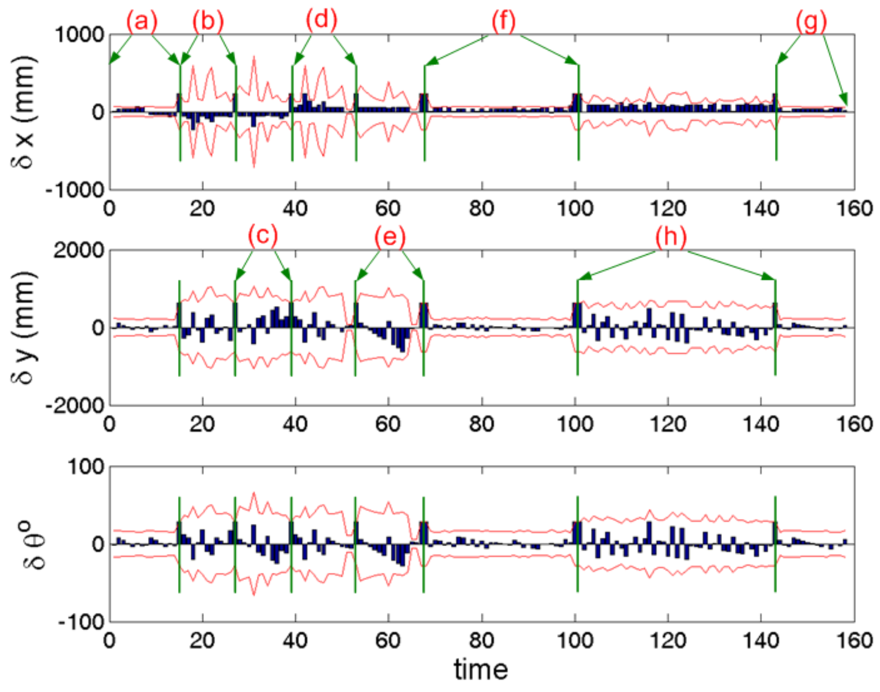


Figure 6.9: Comparison of position estimation error with corresponding $\pm\sigma$ uncertainty bound for position estimates shown in Fig. 6.8. The eight trials of Fig. 6.8 are separated by the green bars and marked as (a), (b), (c), (d), (e), (f), (g) and (h). The marker (a) corresponds to Fig. 6.8(a), (b) to Fig. 6.8(b) and so on

6.3.3 Motion and Rotation

The third category is shown in Fig. 6.10 wherein the robot moves in rectangular paths and at the same time is able to rotate through 360°. In addition to changing position the robot orientation is also changed. In each trial of this category the robot completes a 360°rotation in clockwise direction. In Fig. 6.10(a), dimensions of the rectangle path is 1200×800. The starting step for this trial is at $(1350, 1050)$. The next steps from here are in counter clockwise direction along the rectangle. The step size in these measurements is also kept the same. Between consecutive steps there is a fixed change of 3.6°in the robot orientation. At startup the robot orientation is -90°. Simulation results show that the robot is able to estimate its position from step 15 onwards and not before. From 15 to 20, the robot can estimate its position with respect to landmarks on the left side. The next time it is able to estimate position is from step 65 to 70, where position is estimated with respect to landmarks on the right side.

In Fig. 6.10(b) the robot follows a rectangular path of dimensions 800×600. The starting step for this trial is at $(1150, 950)$. Initial orientation and direction of change in position and orientation remains the same. In this case the robot position was estimated between steps 16 to 22 and 66 to 72. Similarly, in Fig. 6.10(c), Fig. 6.10(d) and Fig. 6.10(e) the same pattern is followed. Statistical results from this category are shown in Table 6.6.

It can be seen from Table 6.6 that in these measurements the error is lower as compared to the previous two cases. The reason for this low error is that the position is estimated with

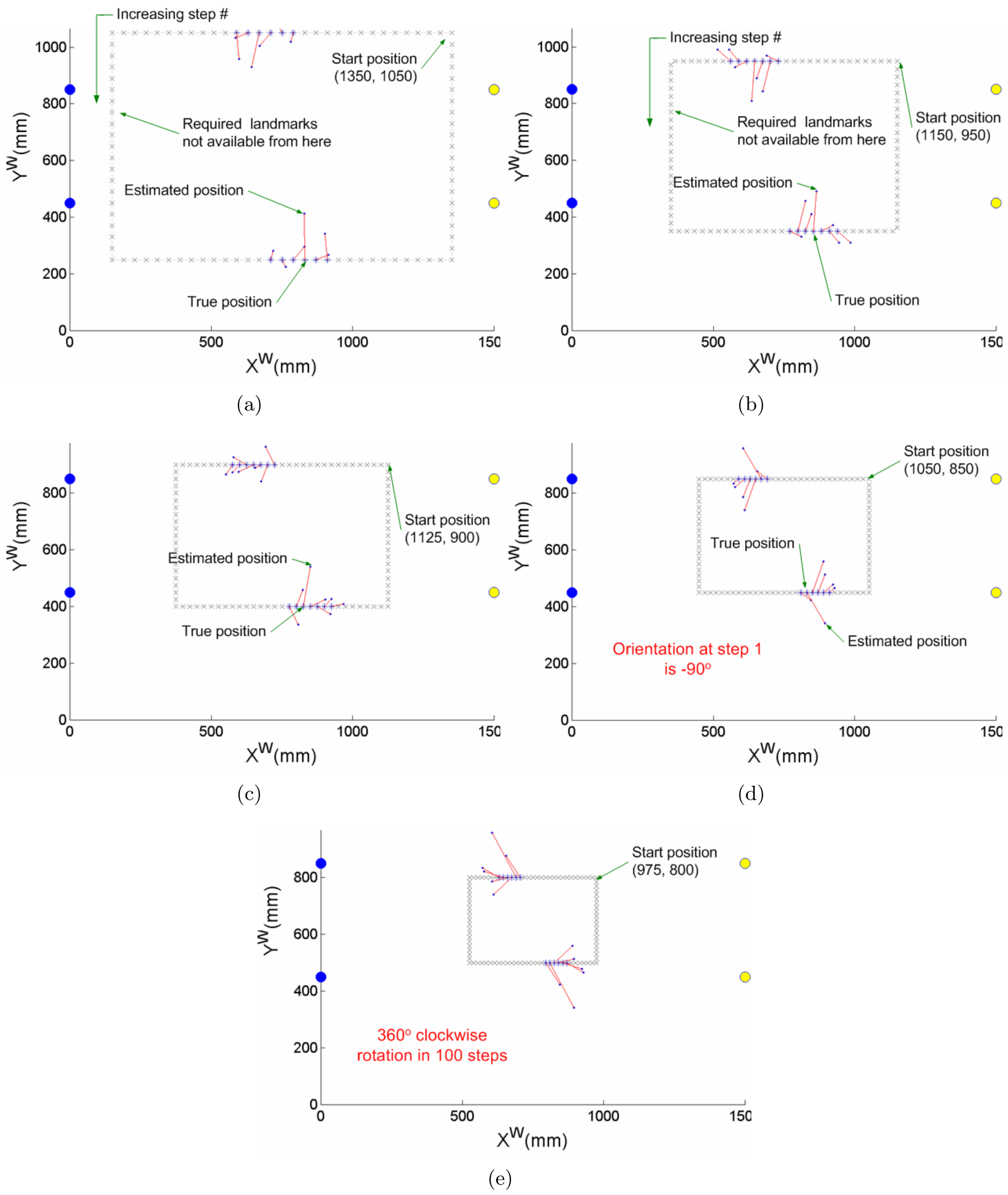


Figure 6.10: Here in this case the robot is following rectangular paths and is rotating as well. The start position is at the top-right corner of the rectangle and the starting orientation is -90 °. The rectangular paths are divided into 100 steps each. The robot position is changing in counter clockwise direction and it completes 360 °clockwise rotation in each trial

Table 6.6: Estimation error, based on the two landmarks method, in each component (x, y and θ)of the robot pose. The robot moves along rectangular paths of different dimensions and is also changing its orientation

	Mean	Std	Min	Max
δx (mm)	36.90	18.88	1.90	84.77
δy (mm)	56.10	42.51	9.66	161.73
$\delta\theta$ ($^\circ$)	3.60	2.76	0.02	9.70

respect to landmarks at moderate distances. Uncertainty as calculated for each step in this category is shown in Fig. 6.11. Different trials of this category are also separated by the green bars and marked in a similar fashion.

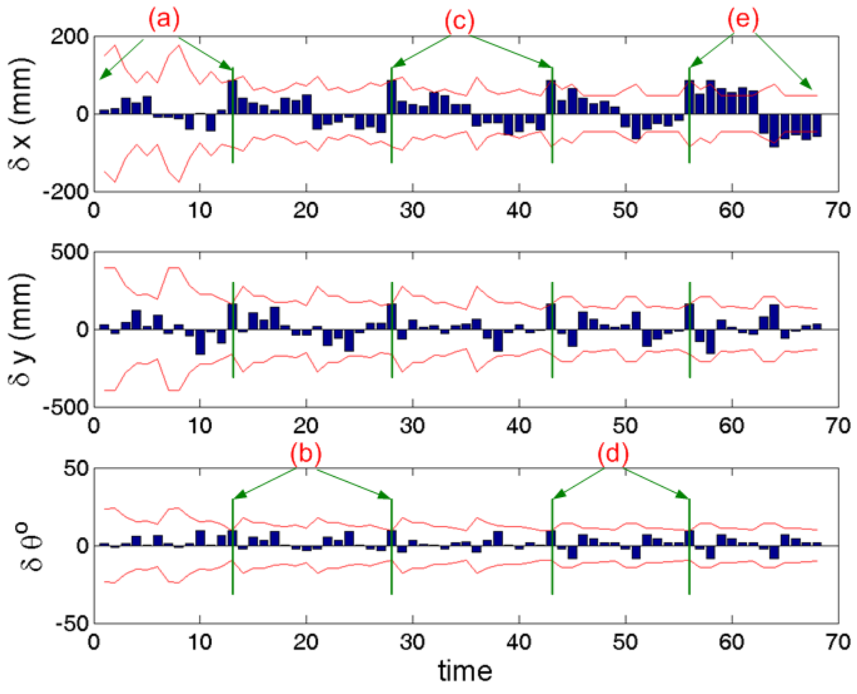


Figure 6.11: Comparison of position estimation error with corresponding $\pm\sigma$ uncertainty bound for position estimates shown in Fig. 6.10. The five trials of Fig. 6.10 are separated by the green bars and marked as (a), (b), (c), (d), and (e). The marker (a) corresponds to Fig. 6.10(a), (b) to Fig. 6.10(b) and so on

6.3.4 Linear Motion with Restricted Rotation

Here, the robot follows linear paths but its orientation remains fixed at 0°or 180°along the x-axis. In total 6 trials were made in this category which are shown in Fig. 6.12. In Fig. 6.12(a), the starting position is at $(321, 650)$, where the robot faces the goal on the left side while moves backwards towards the right. At each step the robot moves away from the goal. The only change between steps is along the x-axis, while orientation and location along y-axis remain fixed. The step size is ≈9 mm. Increase in position error with distance to the landmarks can be seen from this figure. The first time robot acquires both the landmarks is at step 9. The landmarks (the two goals) remain in sight until the end of the trial.

Fig. 6.12(b) shows the location estimation from the same position but in this case the robot's orientation is fixed at $0\,^{\circ}$. In this case the robot faces the landmarks on the right hand side (the yellow goal) and moves closer to them with each step. Similarly, Fig. 6.12(c) and Fig. 6.12(d) follow the same motion pattern of Fig. 6.12(a) Fig. 6.12(b), respectively, but having different start positions. In Fig. 6.12(c) the start position is at $(321, 450)$, whereas, in Fig. 6.12(d) the robot start at $(321, 850)$. Fig. 6.12(e) shows a different pattern. In this case the change between different steps is along the y-axis. The orientation of the robot remains fixed at $0\,^{\circ}$. The start position is in the middle of the field on the lower side at $(750, 222)$. During the next steps there is a constant change of $\approx$9 mm along y-axis. The first time the robot is able to estimate its position is at step 26. From 26 to 76, the robot can estimate its position with respect to landmarks on the right side.

The final trial of this category is shown in Fig. 6.12(f), where the robot follows a similar motion pattern as shown in Fig. 6.12(a), but with a different start position, step size and direction of motion. In this case the robot begins moving from $(1300, 650)$. With each step the robot gets closer to the landmarks on the left side. Table 6.7 shows statistical results from this category, whereas, uncertainty calculations for each estimates are drawn in Fig. 6.13. The rise and fall of error and the corresponding uncertainty values with distance from landmarks can be seen from the figure. The almost constant level of uncertainty, marked as (e), is for the trial shown in Fig. 6.12(e).

Table 6.7: Estimation error, based on the two landmarks method, in each component (x, y and θ)of the robot pose. The robot positions are along linear paths, while its orientation remain fixed along x-axis

	Mean	Std	Min	Max
δx (mm)	69.39	55.88	1.27	270.28
δy (mm)	96.00	130.43	0.011	546.74
$\delta\theta$ ($^{\circ}$)	6.23	7.64	0.0024	28.07

In the above discussion the robot always looks at one of landmarks on the left or right side, due to which position estimates were made at more than 70% of the locations where it was attempted. A little less than 6% of the total estimates are such that the distance between the true and estimated position is greater than 400 mm.

6.3.5 Linear Motion with Rotation

Fig.6.14 shows 5 trials of the last category. Here, the robot motion is along linear paths but in addition to change in the position there is a change in orientation too. Between two consecutive steps there is a small change of $0.2\,^{\circ}$in robot orientation. In Fig. 6.14(a) the robot starts at $(950, 452)$ and moves upward in small steps of $\approx$9 mm. The robot orientation at startup is -170$\,^{\circ}$, which changes to -190$\,^{\circ}$or 170$\,^{\circ}$by the end of the trial. The same motion pattern is followed in rest of the trials, however, with a different start location.

Table 6.8 shows results from the above data. It was possible to estimate the robot position in more than 55% of the locations where features were searched. In 55% of the total estimates

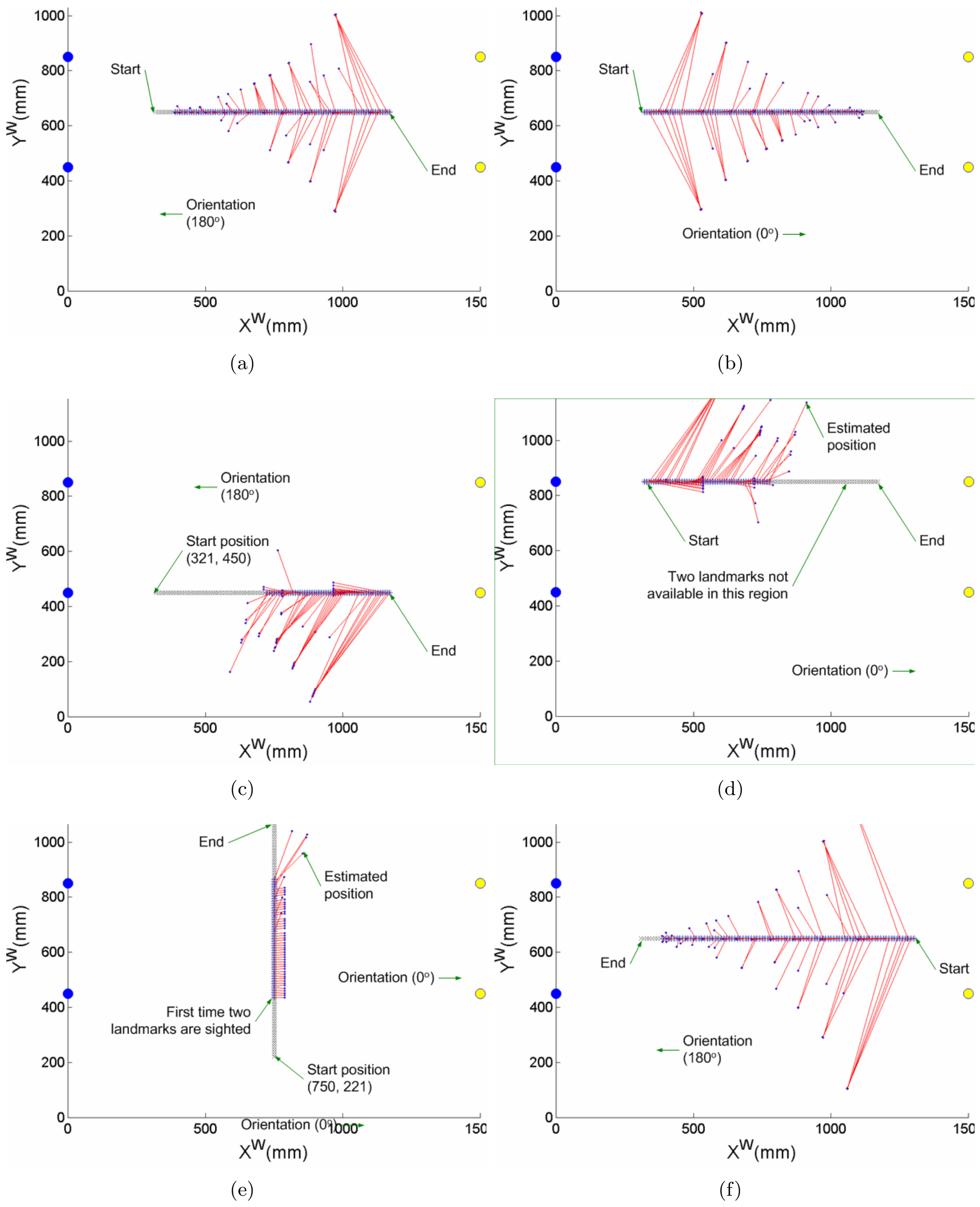


Figure 6.12: The robot is moving along linear paths, while its orientation is fixed along x-axis (a) the robot is looking at landmarks on the left side and moves away towards the right (b) the robot is looking at landmarks on the right side. At each step it is getting closer to the landmarks. In (c) and (d) the motion pattern is as similar to (a) and (b) but positions are different (e) the robot start at the bottom position and moves upward while looking at landmarks on the left side (f) the robot is looking at and moving towards left

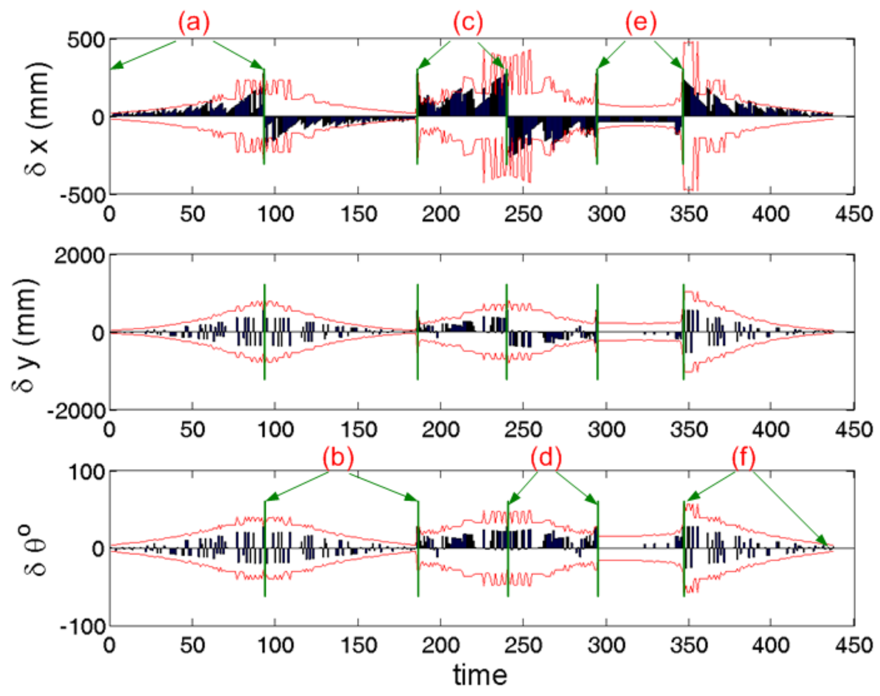


Figure 6.13: Comparison of position estimation error with corresponding $\pm\sigma$ uncertainty bound for position estimates shown in Fig. 6.12. The six trials of Fig. 6.12 are separated by the green bars and marked as (a), (b), (c), (d), (e) and (f). The marker (a) corresponds to Fig. 6.12(a), (b) to Fig. 6.12(b) and so on

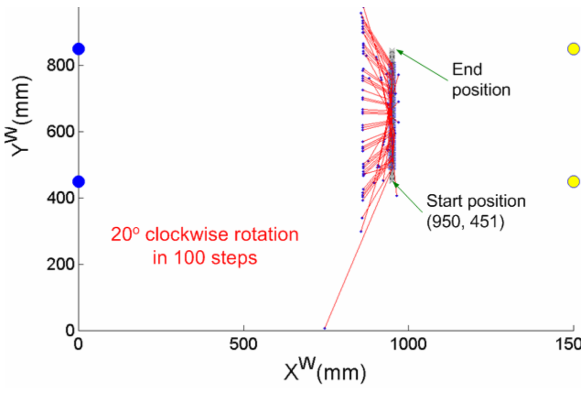
the distance between the true and calculated positions is less than 100 mm. The error is between 100 mm and 200 mm for 35% of the time. Only 4 estimates are such where error is greater than 400 mm. Results from uncertainty analysis are shown in Fig. 6.15.

Table 6.8: Estimation error, based on the two landmarks method, in each component (x, y and θ)of the robot pose. The robot positions are along linear paths. The robot orientation is also changing

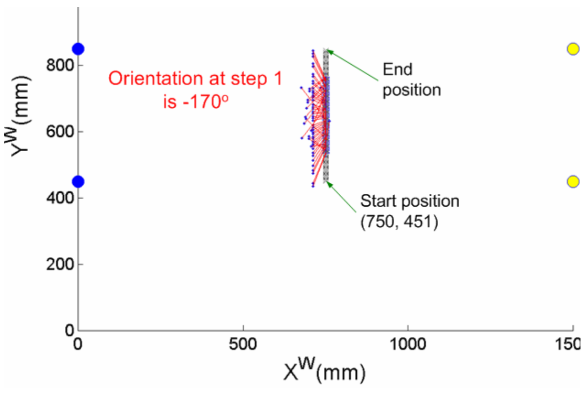
	Mean	Std	Min	Max
δx (mm)	56.26	36.33	0.15	274.40
δy (mm)	86.45	98.21	0.07	804.47
$\delta\theta$ ($^\circ$)	4.99	4.89	0.03	32.59

6.4 Location Estimation Using One Landmark

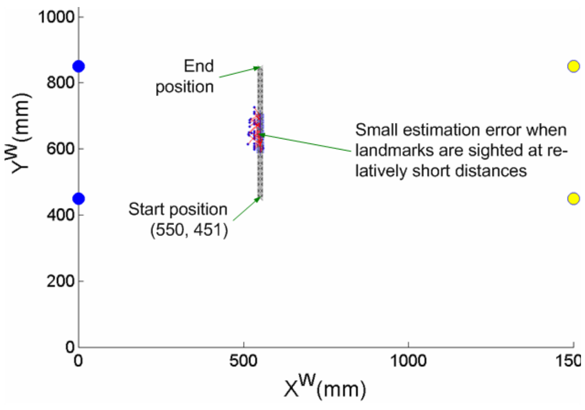
In the following section a performance evaluation of the single landmark based position estimation is presented. For this study it is assumed that information about the absolute orientation of the robot is always available. The locations from where position estimation have been attempted are the same as in the case for two landmark. However, in this case the position is estimated whenever the robot could find a single distinct landmark. The experiment in each category is repeated for varying level of noise in the robot's absolute



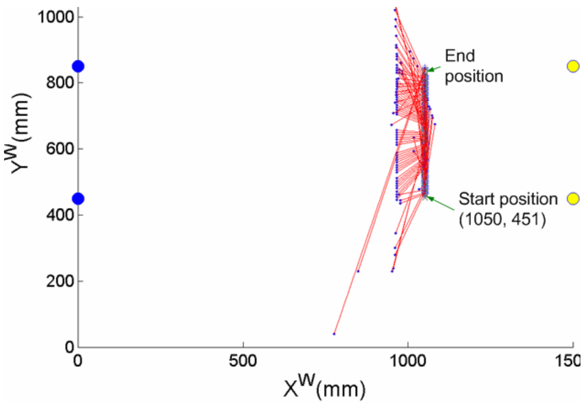
(a)



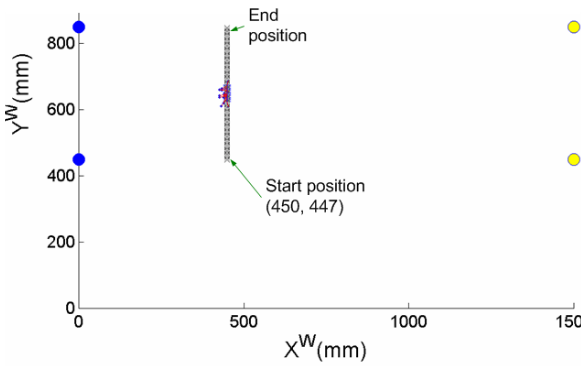
(b)



(c)



(d)



(e)

Figure 6.14: Motion along linear path. In this case the robot is also changing its orientation

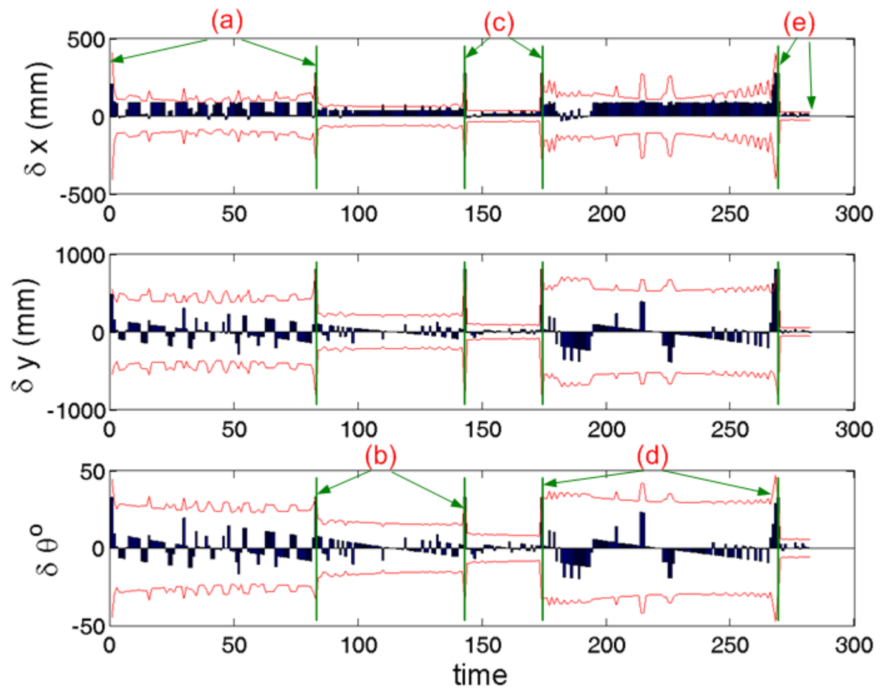


Figure 6.15: Comparison of position estimation error with corresponding $\pm\sigma$ uncertainty bound for position estimates shown in Fig. 6.14. The five trials of Fig. 6.14 are separated by the green bars and marked as (a), (b), (c), (d) and (e). The marker (a) corresponds to Fig. 6.14(a), (b) to Fig. 6.14(b) and so on

orientation, which is additive zero mean Gaussian. Figures shown in this section are generated for orientation error of zero mean and 3.5 °standard deviation.

6.4.1 Motion without Rotation

Fig. 6.16 shows single landmark position estimation for the case where locations from where robot position estimation was attempted lie along rectangular paths of different dimensions. In Fig. 6.16(a), it is only from step 27 to 33, 39 to 41 and 48 to 53 that no landmark can be acquired. Dimensions of the rectangle in Fig. 6.16(b) is 750×500, where the true positions lie more in the field as compared to the first trial. The result is that a single landmark is available in 97 out of 100 steps, whereas, in Fig. 6.16(c) and Fig. 6.16(d), one landmark is always available.

Statistical results for the absolute error in x and y components of the robot pose for the above category is shown in Table 6.9. Position estimates are made frequently here since the probability of sighting a single landmark is higher as compared to the case with two landmarks. The mean as well as standard deviation error is also less.

The effect of rising the level of angular noise can be seen from Table 6.9. Mean as well as standard deviation increases with the increase in angular noise. Fig. 6.7 shows the error in each component of the robot pose and the corresponding uncertainty values. Similar to the two landmarks method, error values are plotted as bars in black color, while the corresponding $\pm\sigma$ uncertainty bound is shown in red. Increase in the position error is accounted for a

Table 6.9: Estimation error, based on the single landmark method, in each component (x, y and θ) of the robot pose. The robot positions are along rectangular paths and there is no rotation involved. The experiment is repeated for varying level of noise in robot orientation

	Angular noise (Std)	Mean (mm)	Std (mm)	Min (mm)	Max (mm)
δx	0°	57.03	44.76	0.15	174.70
δy		11.48	11.85	0	52.47
δx	1°	57.11	45.07	0.18	177.39
δy		16.02	12.81	0.17	78.78
δx	2°	57.95	45.44	0.04	179.21
δy		24.97	20.89	0.04	112.37
δx	3°	58.47	45.35	0.01	184.69
δy		32.42	24.47	0.16	154.09
δx	3.5°	57.95	45.64	0.06	195.36
δy		38.11	30.52	0.06	157.84
δx	4°	58.47	46.53	1.06	200.63
δy		41.11	33.40	0.10	176.61
δx	5°	59.50	46.10	0.01	205.77
δy		52.44	43.38	0.24	222.68
δx	6°	60.96	48.06	0.04	291.74
δy		61.67	53.56	0.08	274.49
δx	7°	63.00	52.19	0.15	345.40
δy		68.78	59.17	0.15	325.90
δx	8°	64.36	50.33	0.23	285.68
δy		76.25	62.74	0.52	340.44
δx	9°	66.07	53.97	0.28	355.34
δy		84.82	67.51	0.71	397.81
δx	10°	66.41	50.71	0.04	251.26
δy		91.51	81.63	0.07	395.84

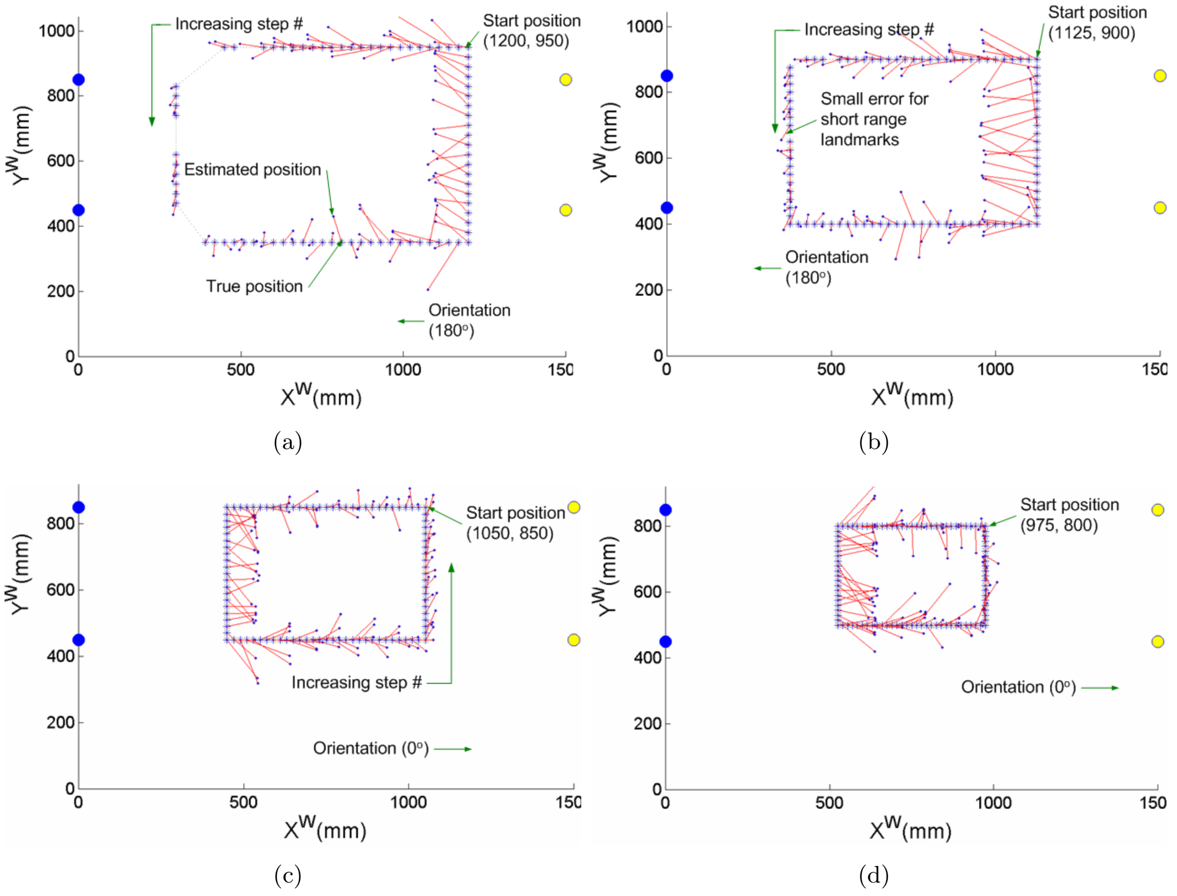


Figure 6.16: Single landmark based position estimation. The robot is following rectangular paths of different dimensions. However, its orientation is fixed at 0 °or 180 °. The starting position in each trial is at the top-right corner of the rectangular path. Locations in next steps are lie is counter clockwise direction. In (a) and (b) the robot is looking towards landmarks on the left side, whereas, in (c) and (d) it is looking towards landmarks on the right side

corresponding increase in uncertainty. The four trials of Fig. 6.16 are separated by the green bars and marked with the name of the sub-figure. For example, all the estimates of the trial shown in Fig. 6.16(a) are marked as (a) in Fig. 6.9.

6.4.2 Single Point Rotation

Fig. 6.18 shows results from position estimation in the second category wherein the robot rotates around a single point looking for landmarks. Details of robot positions and changes in its orientation between steps has already been explained in the previous section. Fig. 6.18(a) shows a 360 °clockwise rotation from the center of the field. The robot orientation at step 1 is −90 °. No landmark can be sighted during the first 14 steps. The first time a landmark can be extracted and position estimated is at step 15. From step 15 to step 36, at least one of the landmarks on the left side is visible to the robot. After this phase no landmark can be extracted from step 37 to 63. At least one of the landmarks on the right side becomes available from step 64 to 86.

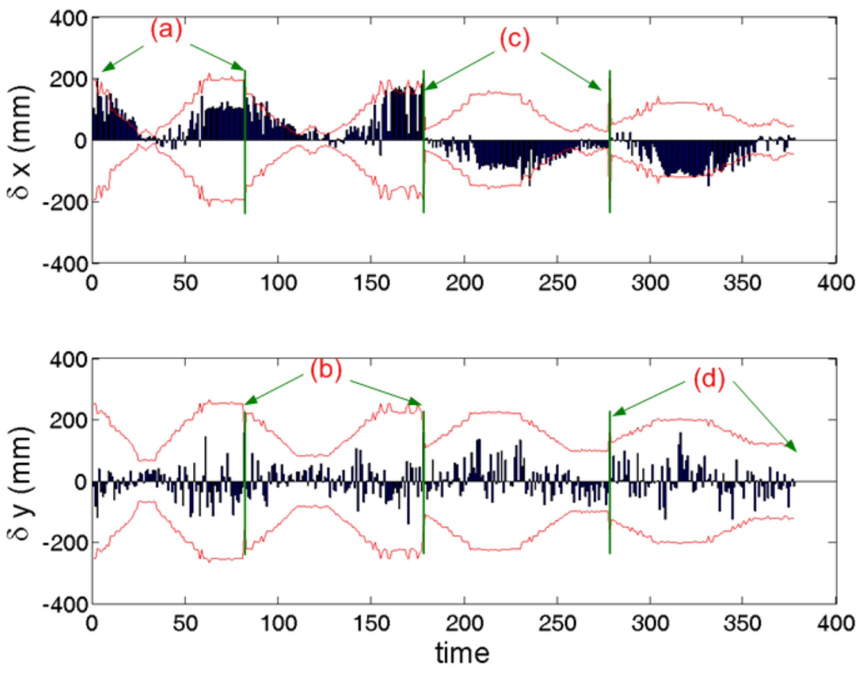
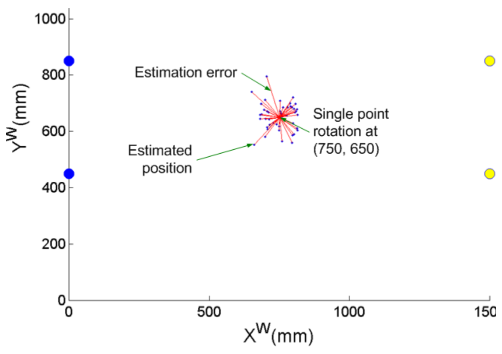


Figure 6.17: Comparison of position estimation error with corresponding $\pm\sigma$ uncertainty bound. Error from position estimations shown in Fig. 6.16 is stacked together and shown by black bars while the red curve give the $\pm\sigma$ uncertainty bound. The four trials of Fig. 6.16 are separated by the green bars and marked as (a), (b), (c), and (d). The marker (a) corresponds to Fig. 6.16(a), (b) to Fig. 6.16(b) and so on

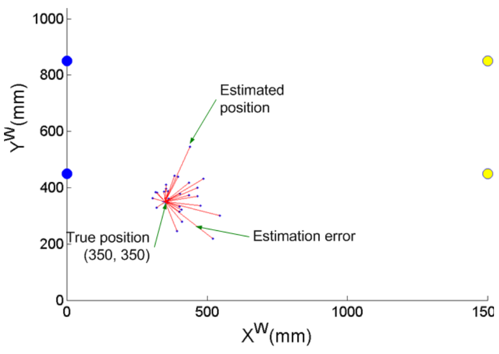
Fig. 6.18(b) shows the second trial of this category. In this trial the robot position was estimated at step 34, 37, 42 to 48 and then from step 64 to 80. Position estimates from step 64 to 80 were made with respect to landmarks on the right side. Similarly, Fig. 6.18(c) to Fig. 6.18(e) show trials where the robot is placed at the other three corners. Fig. 6.18(f) repeats the case presented in Fig. 6.18(a). Here the robot orientation at step 1 is $-135\,^\circ$ and it completes a $90\,^\circ$clockwise rotation in 100 steps. In this case, the robot is able to estimate its position from step 7 to 94, where one of the landmarks on the left side is visible. Statistical results for the rotation-only case are shown in Table 6.10. In this category the robot could estimate its position in more than 45% of the locations where it searched for landmarks compared to less than 20 % of the two landmark case. Additionally, the position error is less as compared to the error shown in Table 6.5. $\pm\sigma$ uncertainty bound for the rotation-only case is shown in Fig. 6.19.

6.4.3 Motion and Rotation

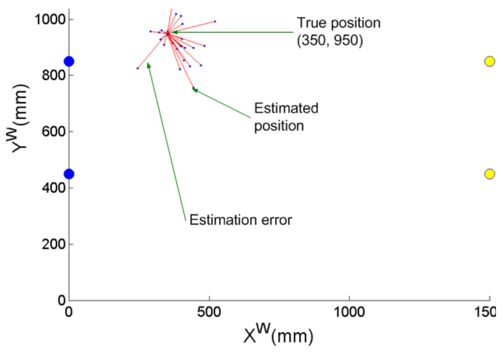
The third category is shown in Fig. 6.20. Here, in addition to changing the position, the robot orientation is also changed. In Fig. 6.20(a), the robot starts at $(1350, 1050)$. Its next steps lie along the rectangular path of size 1200×800 in counter clockwise direction. The lower landmark on the right side is visible during the first two steps. From step 11 to 24 the robot can see one of the landmarks on the left side. However, it fails to estimate its position



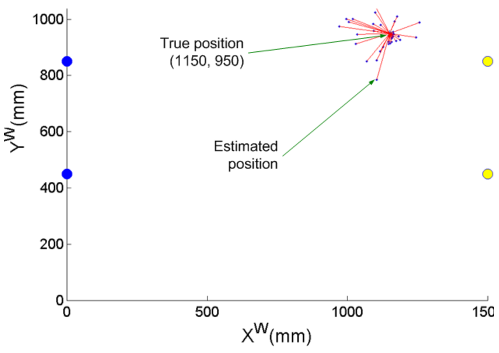
(a)



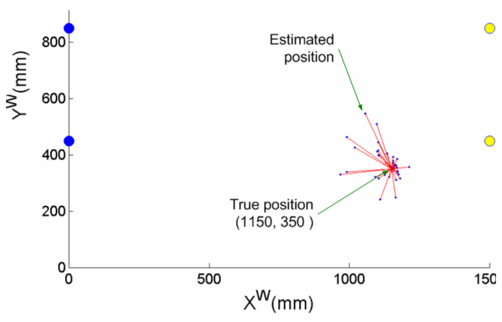
(b)



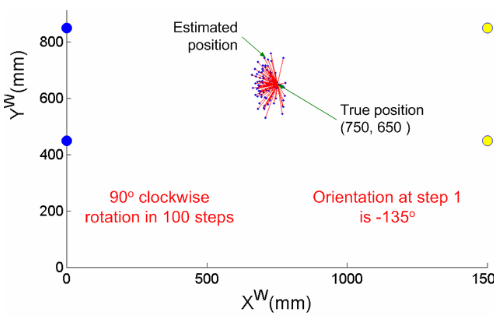
(c)



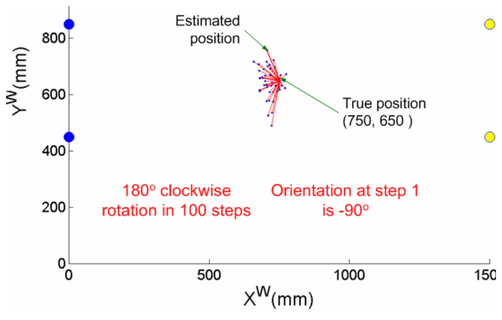
(d)



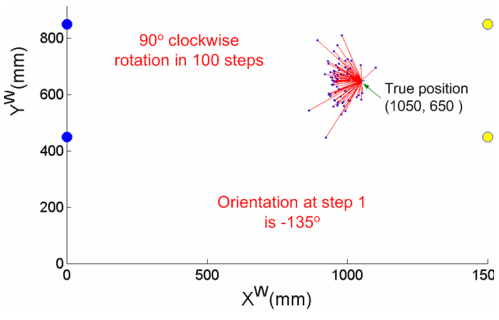
(e)



(f)



(g)



(h)

Figure 6.18: Single landmark based position estimation. The robot is placed at different locations in its environment. In this case the robot is rotating in small steps but without any motion. The robot completes 360 °rotation in (a), (b), (c), (d) and (e), whereas, it completes 180 °rotation in (g) and 90 °in (f) and (h)

Table 6.10: Estimation error, based on the single landmark method, in each component (x, y and θ) of the robot pose. The robot is placed at different positions in its environment. Between different steps only robot orientation changes, while its position remains fixed

	Angular noise (Std)	Mean (mm)	Std (mm)	Min (mm)	Max (mm)
δx	0°	42.52	35.69	0.026	163.89
δy		14.68	13.03	0.005	67.68
δx	1°	42.43	35.76	0.17	163.19
δy		18.26	16.18	0.003	95.46
δx	2°	43.40	36.08	0.11	178.55
δy		23.92	22.13	0.03	179.72
δx	3°	43.95	35.70	0.04	181.18
δy		31.67	28.14	0.28	177.74
δx	3.5°	44.75	37.46	0.23	196.04
δy		39.65	35.73	0.14	202.99
δx	4°	45.86	36.23	0.24	166.90
δy		40.64	35.51	0.013	196.82
δx	5°	46.95	38.23	0.131	218.24
δy		47.77	42.43	0.32	271.06
δx	6°	51.04	44.12	0.37	404.81
δy		60.88	56.82	0.10	314.65
δx	7°	50.73	42.42	0.21	248.07
δy		67.34	62.45	0.04	403.82
δx	8°	54.05	43.90	0.07	282.74
δy		78.58	72.74	0.09	406.91
δx	9°	57.55	52.65	0.05	386.74
δy		85.34	79.82	0.05	501.56
δx	10°	58.71	51.00	0.116	316.76
δy		93.6964	82.79	0.05	435.09

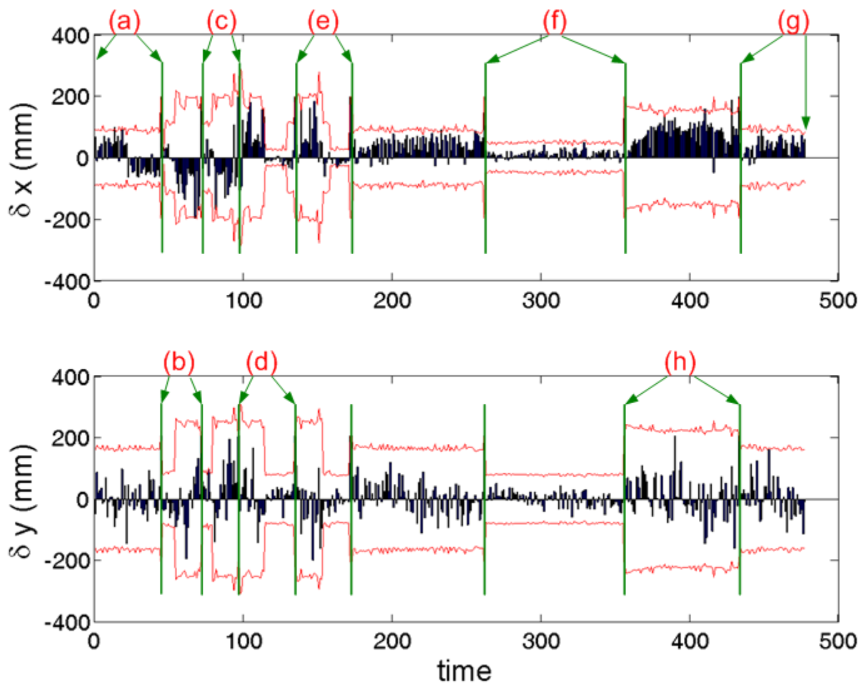


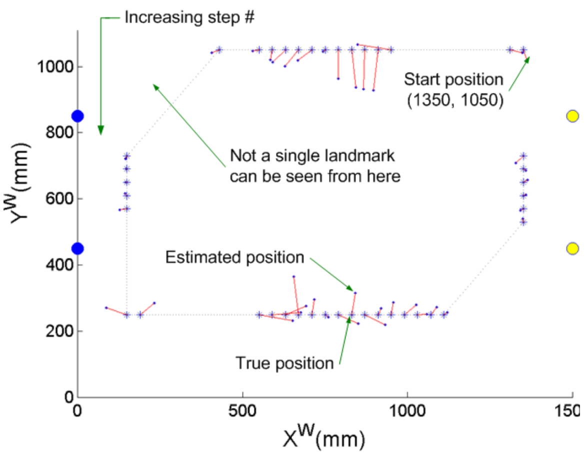
Figure 6.19: Comparison of position estimation error with corresponding $\pm\sigma$ uncertainty bound for position estimates shown in Fig. 6.18. The eight trials of Fig. 6.18 are separated by the green bars and marked as (a), (b), (c), (d), (e), (f), (g) and (h). The marker (a) corresponds to Fig. 6.18(a), (b) to Fig. 6.18(b) and so on

at step 22 and 23 due to segmentation problems. Similarly, motion patterns are shown in Fig. 6.20(b) to Fig. 6.20(e). Statistical results from this category are shown in Table 6.11, whereas, uncertainty as calculated for each step in this category is shown in Fig. 6.21.

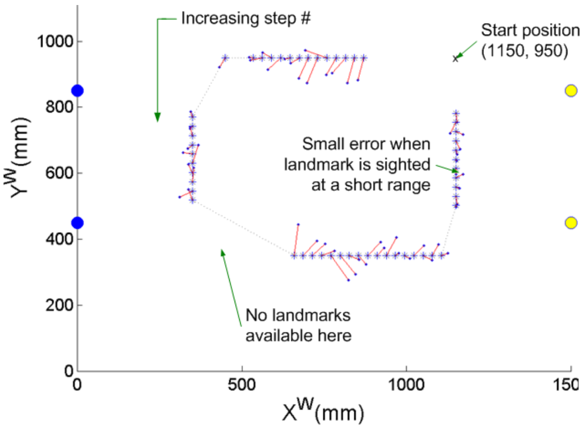
6.4.4 Linear Motion with Restricted Rotation

The fourth category consisting of 6 trials is shown in Fig. 6.22. Here, the robot is following linear paths but its orientation remains fixed at $0\,°$or $180\,°$along the x-axis. In Fig. 6.22(a), the starting position is at $(321, 650)$, where the robot is looking at the goal on the left side, with each step the robot moves away from the goal. The only change between steps is along the x-axis, while orientation and location along y-axis remain fixed. The step size is $\approx$9 mm. Increase in position error with distance to the landmarks can be seen from this figure. The first time robot can acquire at least one landmark is at step 9, which remains in sight until end of the trial.

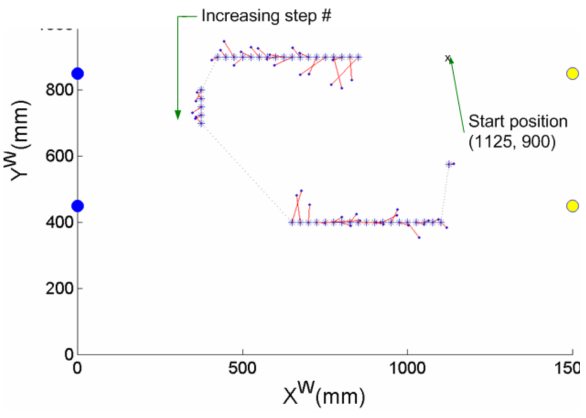
Fig. 6.22(b) shows the location estimation from the same positions but here the robot is looking in the opposite direction. In this case the robot is getting closer to the yellow goal on the right side in each step. In Fig. 6.22(c) the robot starts at $(321, 450)$, and is moving away from landmarks on the left side. Similarly, Fig. 6.22(d) the robot start at $(321, 850)$ and follows the pattern shown in Fig. 6.22(b). Table 6.12 shows statistical results from this category, whereas, uncertainty calculations for each estimates are drawn in Fig. 6.23. In this category the robot is able to estimate its position more than 95% of the time.



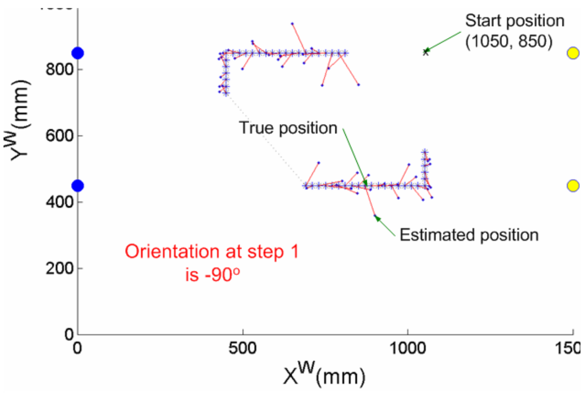
(a)



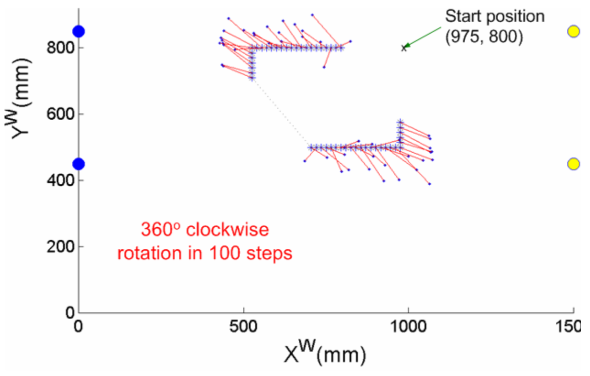
(b)



(c)



(d)



(e)

Figure 6.20: Single landmark based position estimation. Here in this case the robot is following rectangular paths and is rotating as well. The start position is at the top-right corner of the rectangle and the starting orientation is -90 °. The rectangular paths are divided into 100 steps each. The robot position is changing in counter clockwise direction and it completes 360 °clockwise rotation in each trial

Table 6.11: Estimation error, based on the single landmark method, in each component (x, y and θ)of the robot pose. The robot moves along rectangular paths of different dimensions and is also changing its orientation

	Angular noise (Std)	Mean (mm)	Std (mm)	Min (mm)	Max (mm)
δx	0°	34.31	25.76	0.30	98.83
δy		19.70	17.51	0.08	64.98
δx	1°	34.68	26.03	0.09	98.90
δy		21.99	18.63	0.003	85.14
δx	2°	35.95	26.09	0.33	99.79
δy		24.86	19.66	0.08	85.14
δx	3°	36.55	27.13	0.014	127.79
δy		27.97	22.45	0.105	113.90
δx	3.5°	36.27	27.56	0.03	106.15
δy		32.06	27.74	0.014	121.74
δx	4°	38.34	28.70	0.65	147.28
δy		35.45	28.14	0.41	177.09
δx	5°	37.43	27.75	1.23	135.25
δy		39.68	33.24	0.08	169.00
δx	6°	42.84	35.30	0.52	276.81
δy		46.93	49.33	0.19	479.26
δx	7°	43.39	32.44	1.09	180.61
δy		56.44	46.93	0.37	285.49
δx	8°	46.46	37.76	1.56	258.58
δy		69.26	55.76	0.13	259.66
δx	9°	45.95	35.97	1.14	177.13
δy		65.75	52.61	0.40	284.58
δx	10°	53.19	42.97	1.21	270.52
δy		80.46	67.27	0.67	378.00

Table 6.12: Estimation error, based on the single landmark method, in each component (x, y and θ) of the robot pose. The robot positions are along linear paths, while its orientation remain fixed along x-axis

	Angular noise (Std)	Mean (mm)	Std (mm)	Min (mm)	Max (mm)
δx	0°	60.58	41.21	0.73	204.61
δy		3.11	3.57	0	19.98
δx	1°	60.80	41.16	0.94	204.67
δy		10.81	9.15	0.0012	51.26
δx	2°	61.22	41.47	1.19	205.04
δy		19.63	15.96	0.02	95.67
δx	3°	61.60	42.33	0.47	206.96
δy		30.43	24.65	0.08	172.82
δx	3.5°	61.04	41.98	0.23	208.77
δy		31.99	28.08	0.04	217.09
δx	4°	61.98	42.42	1.84	215.08
δy		40.56	34.41	0.45	222.17
δx	5°	62.41	43.56	0.07	213.28
δy		48.70	40.76	0.08	222.48
δx	6°	64.16	43.75	0.28	225.79
δy		60.21	48.83	0.03	248.67
δx	7°	67.35	45.89	0.02	255.98
δy		70.64	58.25	0.19	356.68
δx	8°	66.74	45.15	0.16	219.35
δy		73.98	58.83	0.04	306.21
δx	9°	69.99	48.99	0.34	306.73
δy		89.77	72.73	0.18	405.94
δx	10°	75.35	52.57	0.98	251.09
δy		101.81	82.91	0.17	570.67

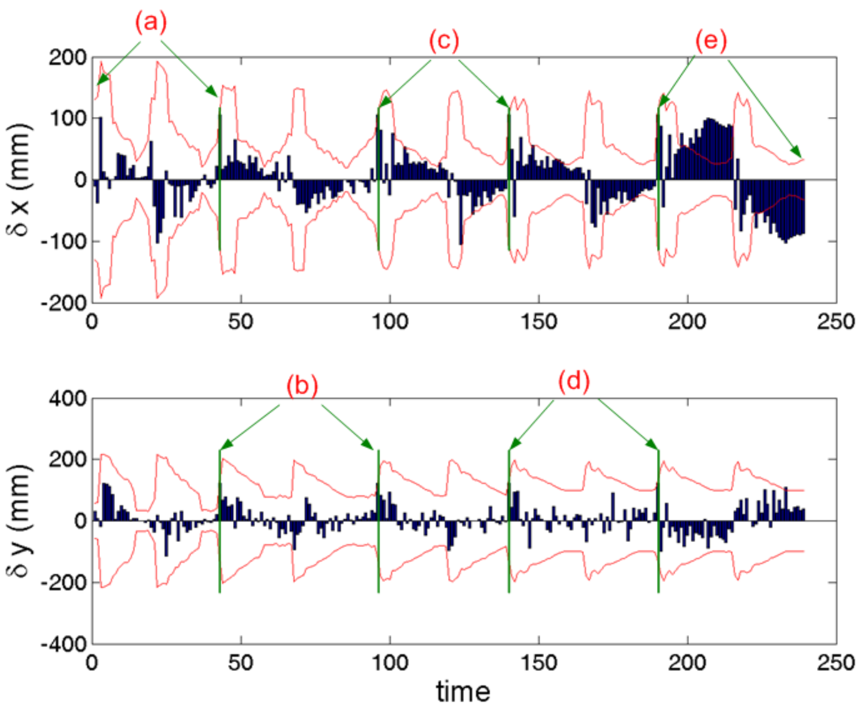


Figure 6.21: Comparison of position estimation error with corresponding $\pm\sigma$ uncertainty bound for position estimates shown in Fig. 6.20. The five trials of Fig. 6.20 are separated by the green bars and marked as (a), (b), (c), (d), and (e). The marker (a) corresponds to Fig. 6.20(a), (b) to Fig. 6.20(b) and so on

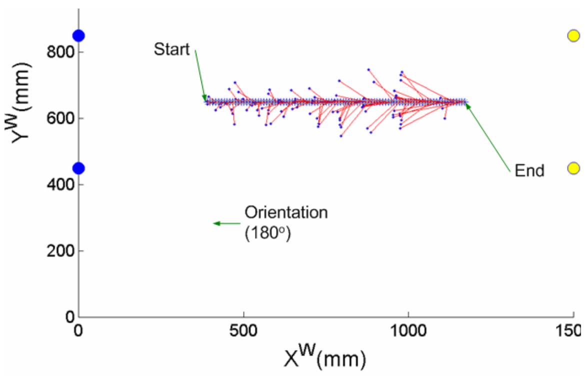
6.4.5 Linear Motion with Rotation

Fig.6.24 shows the last category where the robot moves in linear path and is rotating as well. In Fig. 6.24(a) the robot starts at $(950, 452)$ and moves upward in small steps of ≈9 mm. The robot orientation at startup is -170 °, which changes to -190 °or 170 °by the end of the trial. The starting location in Fig. 6.24(b) is $(750, 452)$. Here the robot follows a similar motion pattern as in Fig.6.24(a). At least one landmark can always be acquired during the entire trial.

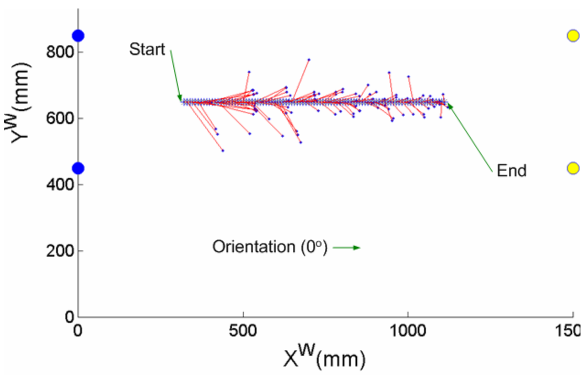
Table 6.13 shows results from the category discussed above. It was possible to estimate the robot position in 100% of the locations where features were searched compared to 55% when two landmarks were used. Results from uncertainty analysis are shown in Fig. 6.25.

6.5 Position Tracking and Information Fusion

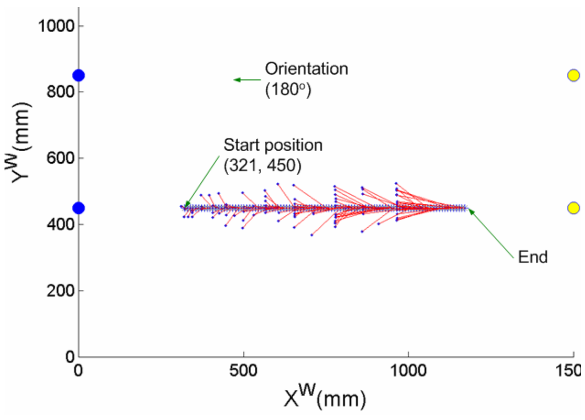
The trials for the position tracking experiment are grouped into three categories according to the trajectory followed by the robot: circular motion, linear motion and rotation about a single point. First the curved trajectory and then its two special cases are tested. In order to improve the statistics regarding the growth of uncertainty and robot position, different trials are conducted in each category and every trial consists of 100 steps.



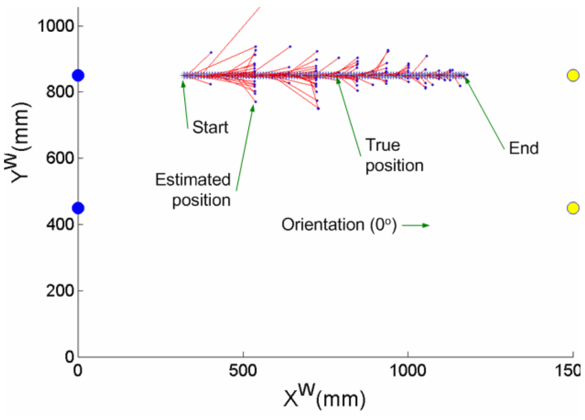
(a)



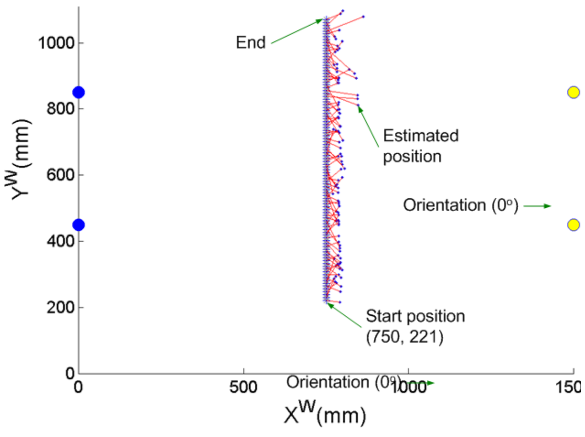
(b)



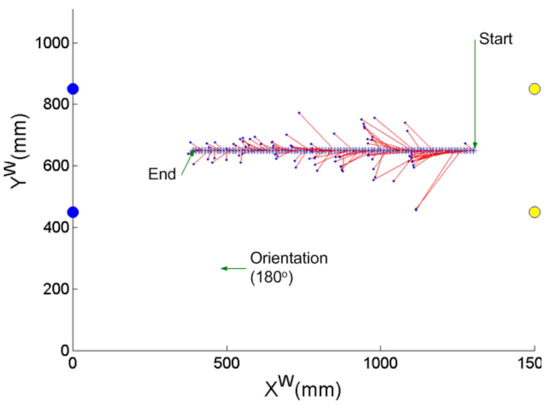
(c)



(d)



(e)



(f)

Figure 6.22: Single landmark based position estimation. The robot is moving along linear paths, while its orientation is fixed along x-axis (a) the robot is looking at landmarks on the left side and moves away towards the right (b) the robot is looking at landmarks on the right side. At each step it is getting closer to the landmarks. In (c) and (d) the motion pattern is as similar to (a) and (b) but positions are different (e) the robot start at the bottom position and moves upward while looking at landmarks on the left side (f) the robot is looking at and moving towards left

Table 6.13: Estimation error, based on the single landmark method, in each component (x, y and θ) of the robot pose. The robot positions are along linear paths. The robot orientation is also changing

	Angular noise (Std)	Mean (mm)	Std (mm)	Min (mm)	Max (mm)
δx	0°	49.01	31.80	0.03	173.73
δy		6.89	24.06	0.0042	378.59
δx	1°	49.15	31.86	0.12	173.65
δy		12.48	26.16	0.08	394.69
δx	2°	49.71	32.37	0.42	174.75
δy		22.32	27.94	0.0031	379.47
δx	3°	50.07	33.22	0.34	175.01
δy		29.97	33.94	0.10	422.09
δx	3.5°	50.36	33.9883	0.02	173.90
δy		34.86	37.48	0.012	411.98
δx	4°	50.87	34.30	0.012	175.28
δy		42.74	41.88	0.12	444.12
δx	5°	52.94	34.70	0.08	176.09
δy		50.93	46.93	0.11	337.05
δx	6°	54.23	35.63	0.65	179.04
δy		62.50	56.22	0.02	487.67
δx	7°	55.05	36.90	0.14	181.84
δy		70.54	62.20	0.16	510.14
δx	8°	57.21	40.38	0.05	282.57
δy		76.96	68.63	0.37	569.72
δx	9°	59.89	43.46	0.002	244.21
δy		93.17	76.20	0.31	469.05
δx	10°	61.07	44.09	0.0014	258.40
δy		91.68	76.36	0.23	419.38

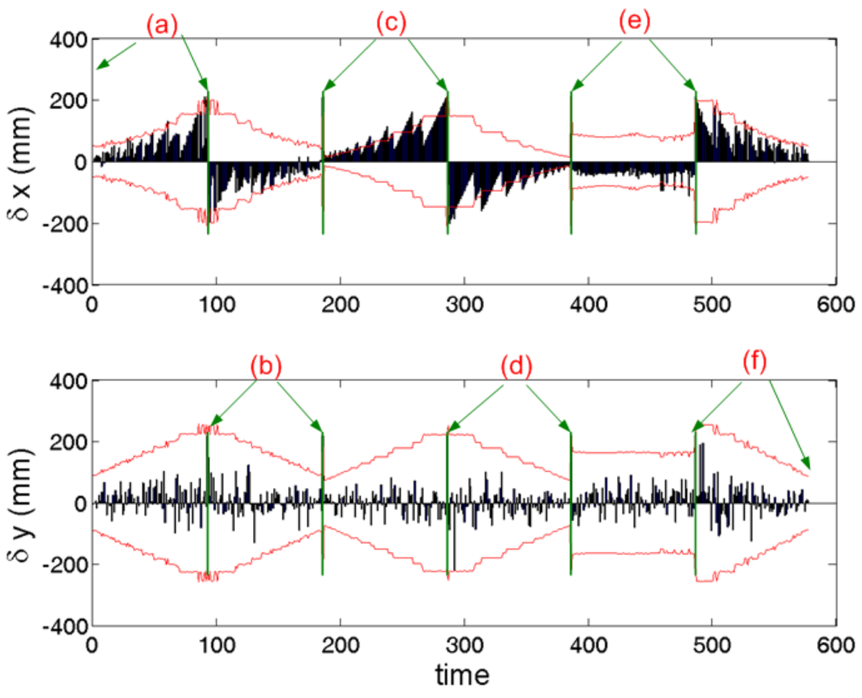
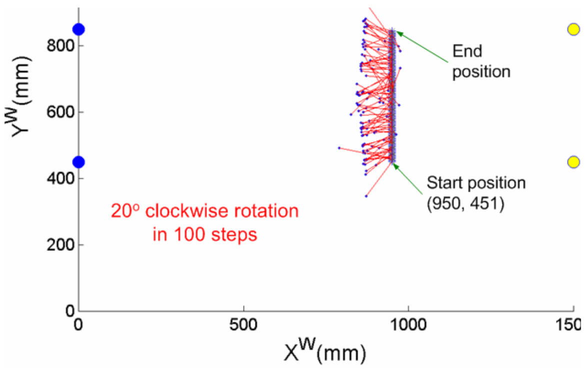


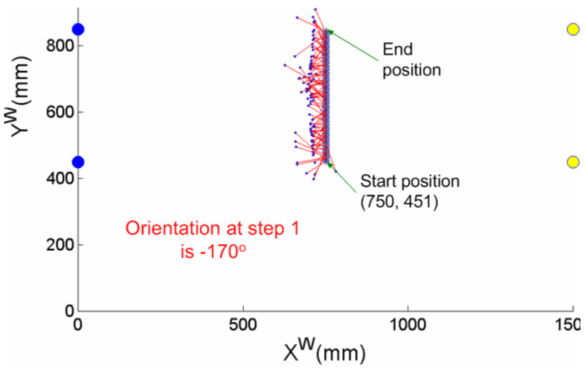
Figure 6.23: Comparison of position estimation error with corresponding $\pm\sigma$ uncertainty bound for position estimates shown in Fig. 6.22. The six trials of Fig. 6.22 are separated by the green bars and marked as (a), (b), (c), (d), (e) and (f). The marker (a) corresponds to Fig. 6.22(a), (b) to Fig. 6.22(b) and so on

For extensive testing of the algorithm the experiments are repeated in three different scenarios. Firstly, the position tracker is initialized manually with a starting position and its corresponding uncertainty. The initial position is obtained by adding random noise to the true position. Secondly, the robot is given a random start position with a high uncertainty. This is to study the convergence of the robot position belief when features are available. Providing the starting position and the covariance matrix violates the robot autonomy. For autonomous behavior the robot must be equipped with tools that enable it to localize itself from scratch. This capability of the algorithm is demonstrated in the final run of the experiment where initialization of the tracker is done with the help of global position estimation methods discussed in Chapter 4. At startup the robot does not initialize its position tracker and actively searches landmarks until it finds enough that are required for global localization. After finding its position on the global map, the robot initializes its location tracker with this position and its corresponding covariance.

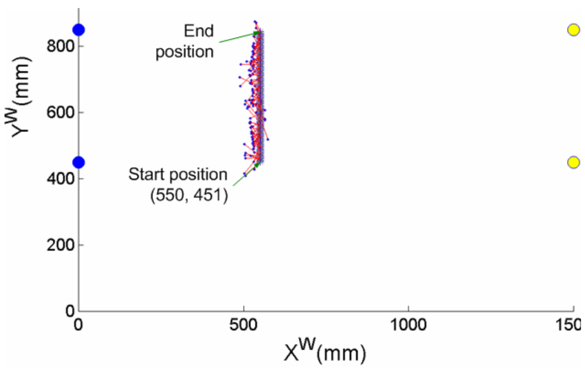
The first category consists of different trials where the robot follows a circular path of different diameters. A sample trial of this category is shown in Fig. 6.26. Here radius of the circle is 500 mm. As can be seen from Fig. 6.26(a) the ideal path that the robot is supposed to follow is a perfect circle. However, due to imperfections of its sensors the robot deviates from the true path. In this trial the true starting position of the robot is $\begin{bmatrix}1350 & 650 & 90^\circ\end{bmatrix}$. In 28 of the 100 steps, at least one landmark is visible. Two landmarks were visible in four of these steps. The first landmark sighting is at step 32. Due to the fact that the robot travels on the same path and performs the same relative movement at each step, a landmark is visible at the same step(s) in every run of the same trial. In Fig. 6.26(a) the robot starts at a known



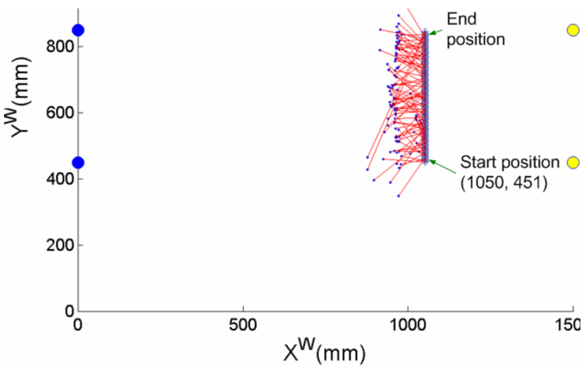
(a)



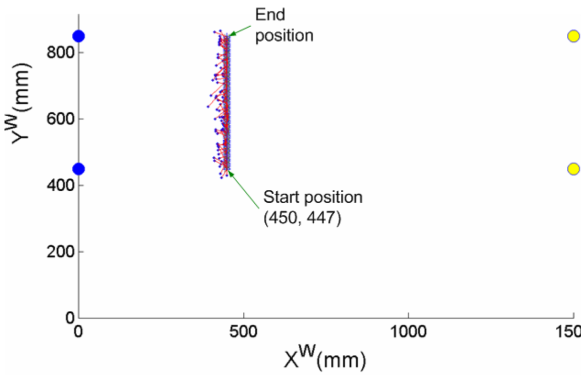
(b)



(c)



(d)



(e)

Figure 6.24: Single landmark based position estimation. Locations from where position estimation is attempted lie along linear paths. The robot orientation at startup is -170°, which changes to -190°or 170°by the end of the trial. In all of these trials, y-component of robot starting position is ≈400 and the step size is ≈4 mm. The x-component of the positions is starting is 950, 750, 550, 1050 and 450 in (a), (b), (c), (d) and (e), respectively

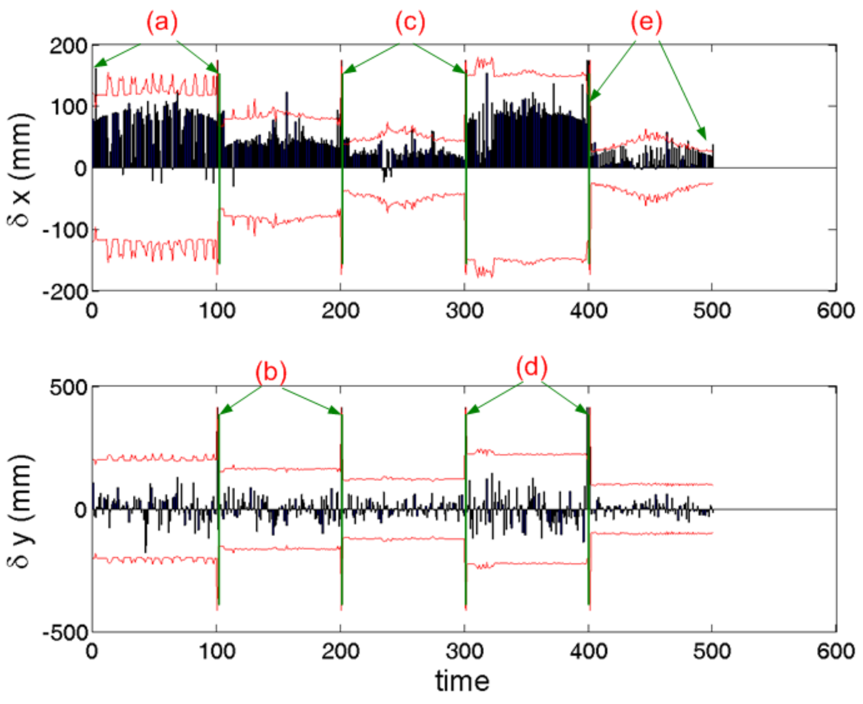
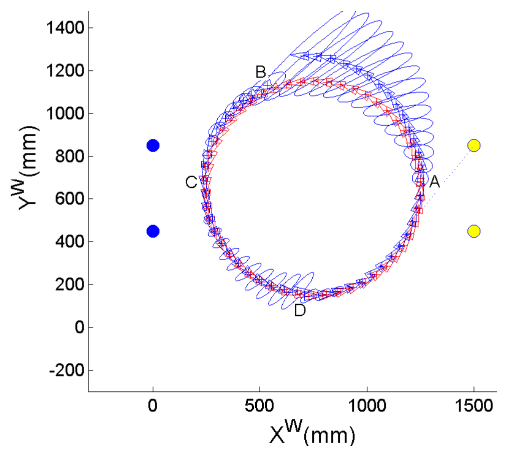


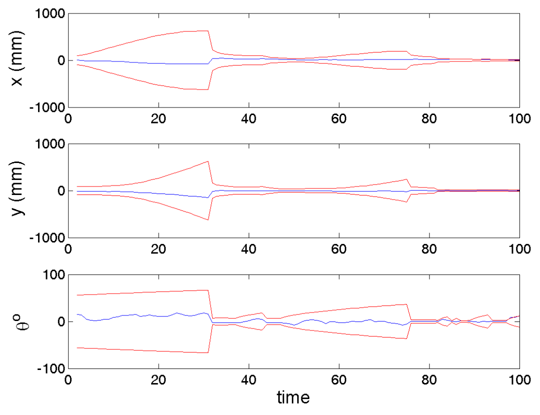
Figure 6.25: Comparison of position estimation error with corresponding $\pm\sigma$ uncertainty bound for position estimates shown in Fig. 6.24. The five trials of Fig. 6.24 are separated by the green bars and marked as (a), (b), (c), (d) and (e). The marker (a) corresponds to Fig. 6.24(a), (b) to Fig. 6.24(b) and so on

position with low uncertainty and moves in counterclockwise direction. The starting position is marked as 'A' (in Fig. 6.26(c) the random start point is marked 'A1' and the true start as 'A2'). The robot observes its first landmark feature when it has covered more than a quarter of its intended path marked as 'B' (the estimated and true points in Fig. 6.26(c) and Fig. 6.26(e) are marked as 'B1' and 'B2' respectively). As can be seen in Fig. 6.26(a), the uncertainty of the robot position is growing continuously and the pose drifts away from the true value. When the robot observes one of the two features on the left side, its uncertainty is reduced and position adjusted. However, as soon as the landmarks are out of sight, drift from the true position starts and the uncertainty increases. Between 'C' and 'D' robot position uncertainty is increasing unbounded. This drift is corrected when the landmarks are visible again from 'D' onward.

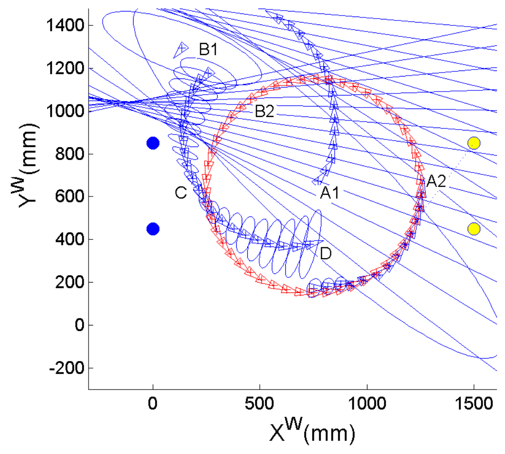
Fig. 6.26(c) illustrates the case where the robot is given a random starting position with a high uncertainty. The true path is shown in red whereas, the estimated path is shown in blue. As can be seen in the figure the robot is constantly moving away from its intended path and its uncertainty is growing rapidly. However, when landmark features become available to the robot, corrective steps are applied and the robot gets closer and closer to its true path. Since, only odometric data is available during the first thirty two steps the relative movement of the robot is in wrong direction. After the first sighting of landmark at position marked as 'B2', the estimated position jumps to the upper right of the field ('B1'). Consecutive landmark sightings reduce the uncertainty and improve estimation. Due to scarcity of landmarks the robot takes long to rectify its position belief. However, the assumption of the existence of globally distinct landmarks helps the robot converge to its correct position when features are



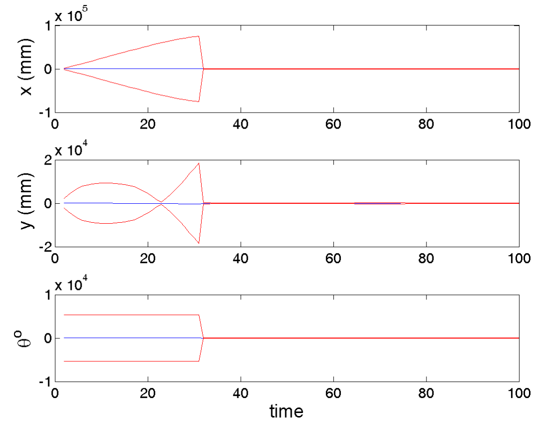
(a)



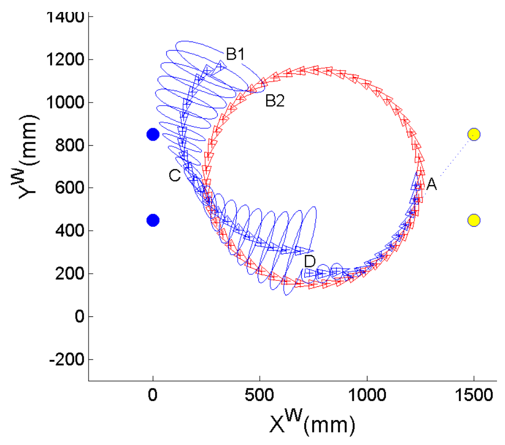
(b)



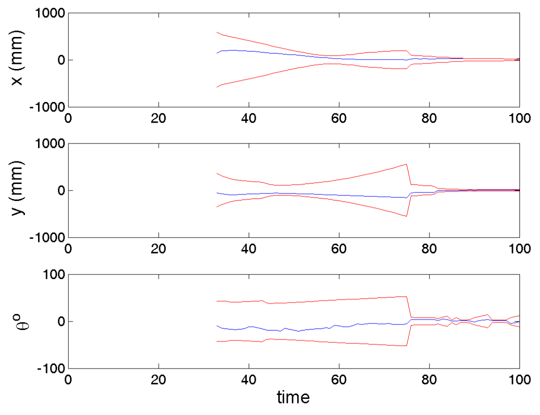
(c)



(d)



(e)



(f)

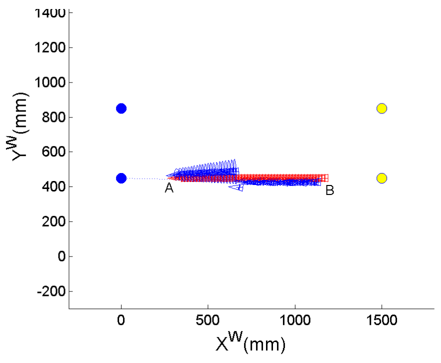
Figure 6.26: Path estimation using EKF. Robot desired trajectory is a circle of radius 500 mm (a) the robot starts at a known position with low uncertainty (b) $\pm 3\sigma$ bound on error in x, y and θ when the robot starts at a known location (c) the robot starts at a random position with a very high uncertainty (d) error bounds for random starting position (e) start position and uncertainty estimated using single landmark method (f) uncertainty bounds when the robot position is initialized with global localization method

observed. The process is rather slow and the global position estimation should be carried out at startup and whenever it loses track of its position as it results in a better performance. Global self-localization for the same path is shown in Fig. 6.26(e). Between 'A' and 'B2' no landmarks are available and the robot does not do any estimation of its position but only searches for landmarks. The global position is estimated using the single landmark method at 'B1'. Between 'B1' and 'C' the robot position belief is continuously improved since it can sight at least one landmark. From 'C' onwards the robot cannot acquire any of the landmarks and the uncertainty is growing again. From point 'D' onwards the robot position is continuously adjusted. $\pm 3\sigma$ error bound on each component of the robot pose for all the three runs of this trial are shown in Fig. 6.26(b), Fig. 6.26(d) and Fig. 6.26(f). The error in each component of the robot pose is shown in blue, whereas, the corresponding $\pm 3\sigma$ uncertainty bound is shown in red. It can be seen from these figures that the error is bounded.

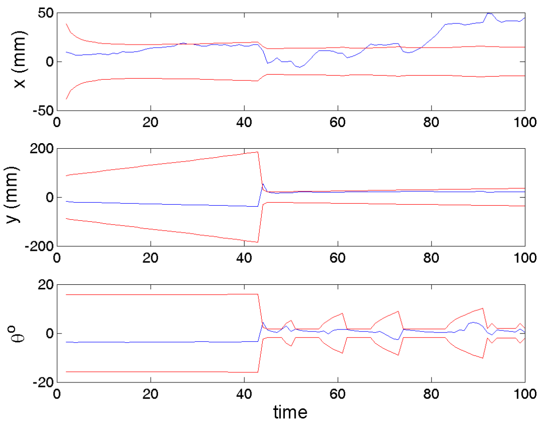
The second category consists of trials to test the limiting case when the two wheels are moving with approximately the same velocity. The difference between different trials is in starting position of the robot and/or direction of motion. A single trial from this category is shown in Fig. 6.27. The robot orientation is fixed at 180 °and it starts at a point 'A' on the left side and moves in reverse direction towards the right in small steps of 8.9 mm each. The end point is marked as 'B'. The orientation of the robot is such that in 99 out of the 100 steps, at least one landmark is visible. The first landmark sighting is at step 2. From step 44 onwards two landmarks are visible. First run of this trial where the position tracker is manually initialized is in Fig 6.27(a). The robot starts at a known position with low uncertainty. When the robot moves further away from the landmarks, the effect of robot observation decreases as its uncertainty increases and the accumulation of odometry error dominates. As predicted by (B.23) uncertainty in direction perpendicular to the direction of motion is growing much faster than uncertainty in x or θ. The uncertainty is reduced when the second landmark becomes available. The effect of deteriorating observation can be seen in Fig. 6.27(b) where error is not bounded. Also in this case the experiment is repeated for a random start location with high uncertainty and global position estimation as shown in Fig. 6.27(c) and Fig. 6.27(e), respectively. In Fig. 6.27(c) the random start position is marked as 'C'. Since, observations are frequently available, the position error is quickly reduced. Further reduction in error is achieved when the second landmark is sighted. The error in each component of the robot pose and their corresponding $\pm 3\sigma$ uncertainty bounds are shown in Fig. 6.27(b), Fig. 6.27(d) and Fig. 6.27(f).

The trials in the last category simulate the robot behavior when the two wheels are rotating with the same velocity but in opposite direction. Under such a scenario the robot rotates around its center of mass. During each trial the robot may or may not complete a full 360 °rotation. Fig. 6.28 shows a single trial of this category. Here the robot starts at $\begin{bmatrix} 750 & 650 & -90\,° \end{bmatrix}$ and completes a 180 °rotation in 100 steps in clockwise direction. During this trial the robot observes at least one landmark in 44 out of 100 steps (between step 29 and step 72). Two landmarks are available only in 15 of the 44 observations. Similar to the previous two categories the experiment is repeated for random and unknown starting position. Even though the randomly chosen position in Fig 6.28(c) matches with the true position, the uncertainty stays high.

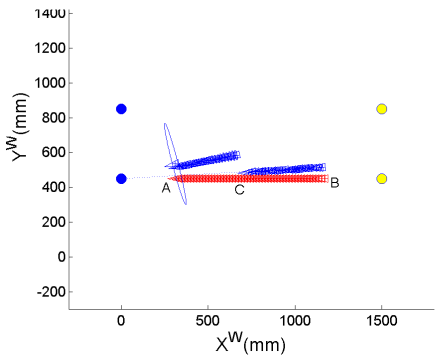
The above experiment is compared with a PF based position estimation method. Details of the comparison study are reported in [BDN07, Deu07]. The PF algorithm uses random values and hence a single run is not precisely repeated. The results presented here are generated by using 25 different runs for the same input data set. Fig. 6.26 is repeated in Fig. 6.29. The



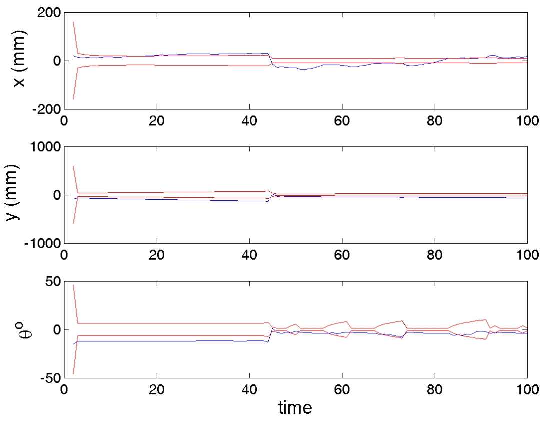
(a)



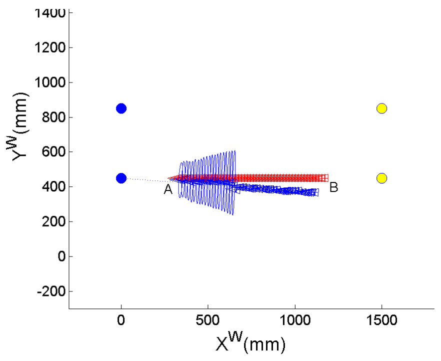
(b)



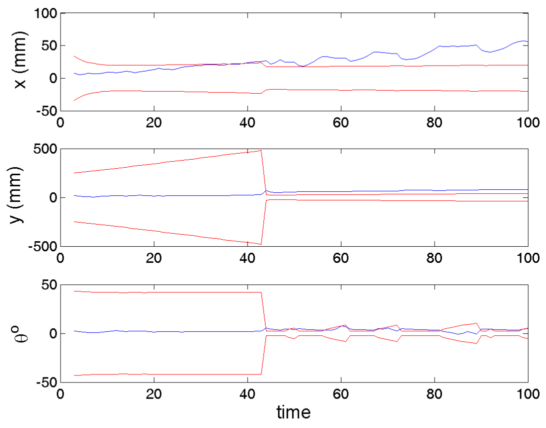
(c)



(d)



(e)



(f)

Figure 6.27: Path estimation using EKF. The robot's desired path is a straight line (a) known starting position (b) error and $\pm 3\sigma$ error bound for known starting location (c) starting at a random position with high uncertainty (d) error and $\pm 3\sigma$ bound on error in x, y and θ for a random start position (e) the location tracker is initialized by estimating the robot location and its uncertainty using global position estimation (f) location error and $\pm 3\sigma$ error bound when the location tracker is initialized with a global position estimate

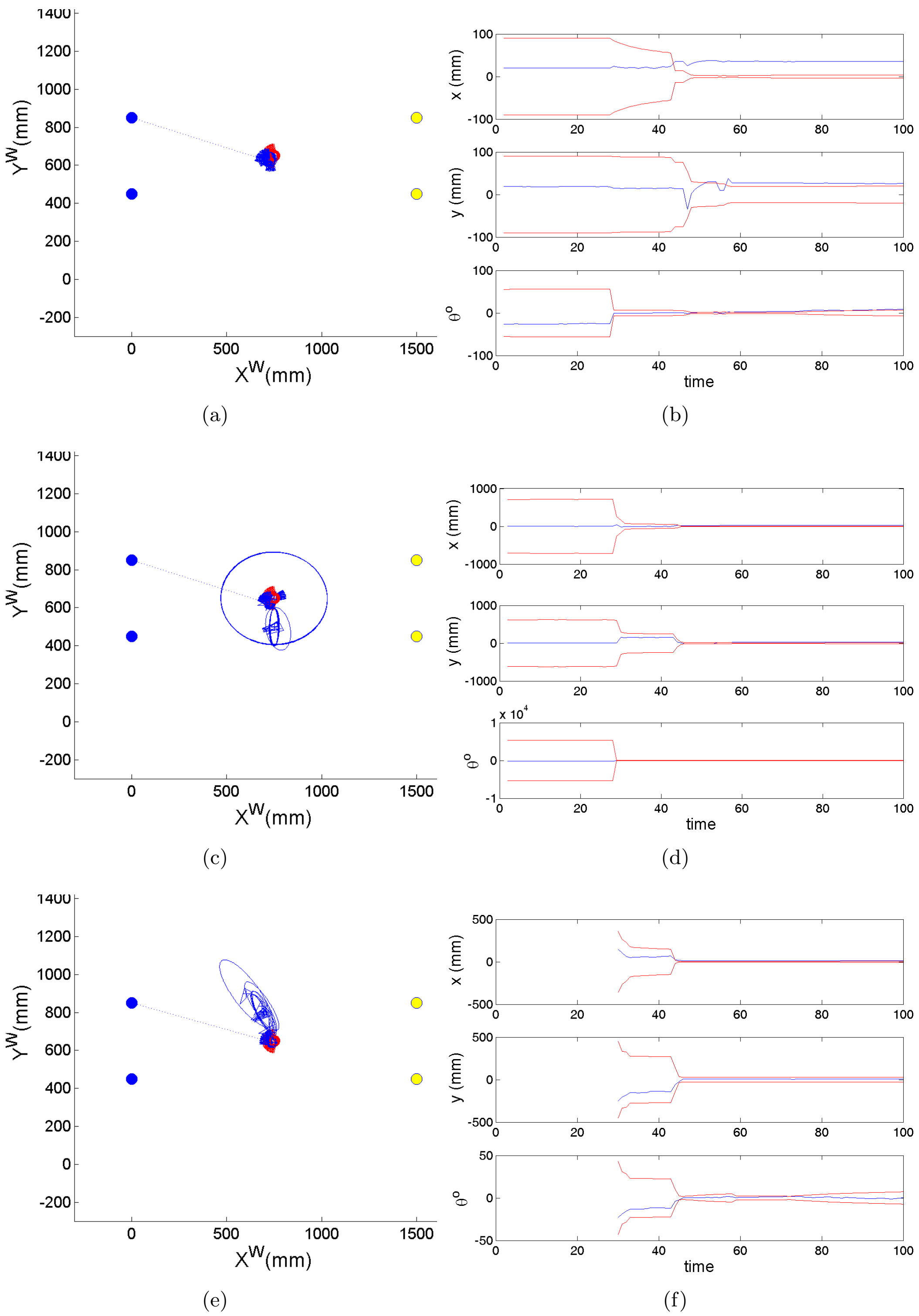


Figure 6.28: Path estimation using EKF. The robot is rotating about its center of mass (a) starting at a known position (b) $\pm 3\sigma$ error bound for known starting position (c) starting at a random position (d) $\pm 3\sigma$ error bound for random start (e) initial position is estimated using the single landmark method (f) $\pm 3\sigma$ error bound for global initialization of the filter

estimated path and absolute error in x, y and θ for the EKF and MCL is shown blue dotted and red dashed, respectively. The gray line in Fig. 6.29(b) shows the 100 mm error reference in x and y and $7.5\,^{\circ}$in θ. The true path is shown solid in gray color. Both methods are using the same data.

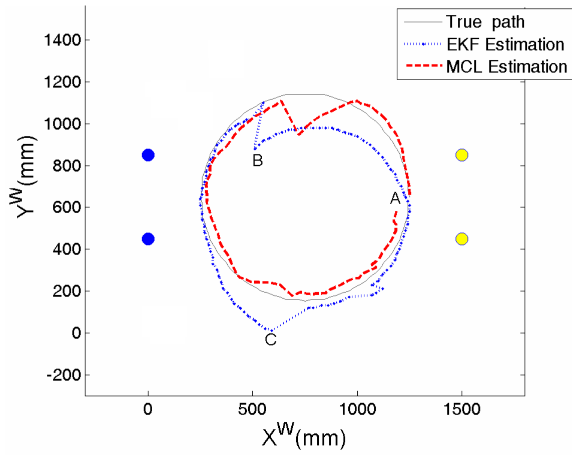
Path estimation with the EKF has already been explained in the previous paragraphs. In Fig. 6.29(a), both methods are provided with a known start position. The starting position is marked as 'A'. It can be seen from the figure that the path estimated by both methods drifts from the true path until landmarks are available at point 'B'. Random initialization is shown in Fig. 6.29(c). The start position of the PF is marked 'A1', while the true start is marked as 'A2'. After the first sighting of landmark, PF based estimated position jumps to the upper left of the field marked as 'B'. Beyond this point the estimated position is getting closer to the true position but starts drifting away when no landmarks can be sighted. Improvement in position estimate is rather slow until landmark on the right side is sighted. This position is marked as 'C' in the figure. The path estimation done by the EKF is corrected by the new sightings towards the real path, which fits the true path well until the end of the round.

The algorithm is initialized with a set of 1000 uniformly distributed position estimations (particles). The particles at each step move according to the odometric data. Along with the particle resampling/reinjection, the estimated position stays close to the center. With the first landmark sighting at position marked 'B', the estimation gets closer to the true path. By the time the robot reached the point marked with 'C', fifteen steps without a landmark sighting were passed. This causes a spread of the particles and overlaps the borders of the playground. Similar to the EKF, with the first landmark sighting of the other goal at point marked as 'C', the estimated path approaches the true path.

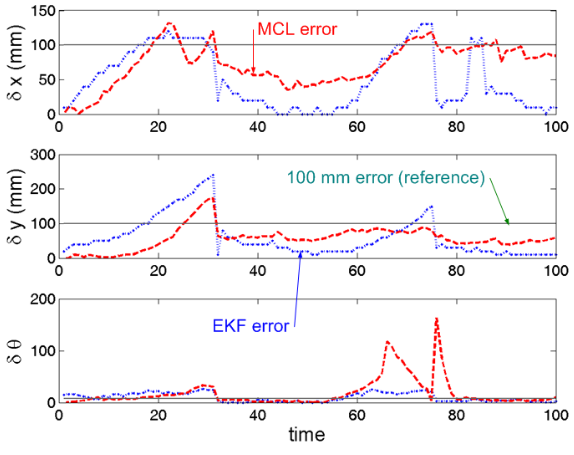
The estimation done by the EKF algorithm is smoother as compared with the one achieved with the MCL. As shown in Fig. 6.29(c), the path of the EKF is following a similar path as the true one. An individual particle also follows a path based on the control input, however, the resultant path estimate of the robot does not look like a circular arc as N circular paths contribute to it. Comparison for motion along a straight line and single rotation is shown in Fig. 6.30 and Fig. 6.31, respectively.

Comparison results of the EKF algorithm and the MCL algorithm for the last 20 steps of the trials discussed above are shown in Table 6.14, Table 6.15 and Table 6.16. The tables also show results from after landmark sighting until step 79. Both the algorithms are getting closer to the true values. The two categories where the EKF is performing better than the MCL are the maximum distance and the standard deviation. Bad estimations in Table 6.15 are the result of erroneous observation for line motion. The terms 'Known', 'Random', and 'Global' refer to the way EKF is initialized. 'Known' means the filter is initialized manually by providing it the starting position and uncertainty, 'Random' means random start position and high uncertainty, while in 'Global' the start position and uncertainty is estimated using global self-localization techniques discussed earlier.

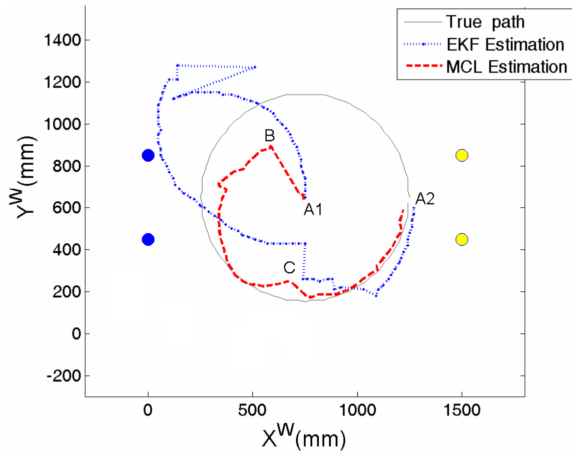
Statistical results for the error in robot observation are shown in Table 6.17. In this table δr and $\delta\varphi$ refer to the range and bearing components of the observation vector. δr is expressed in millimeters and $\delta\varphi$ is given in degrees. The first row show values for the average error. The standard deviation (Std), minimum (Min) and maximum (Max) values for each group are presented in the second, third and fourth row. The Min and Max shown are the absolute values. The first two columns list error in range and bearing in the category where robot follows circular paths of different diameters. Results from single point rotation and straight



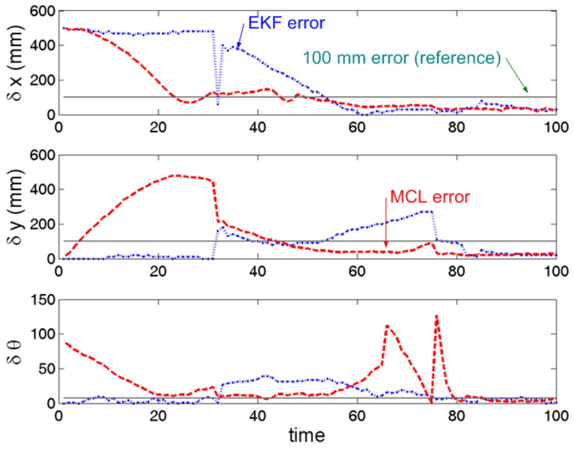
(a)



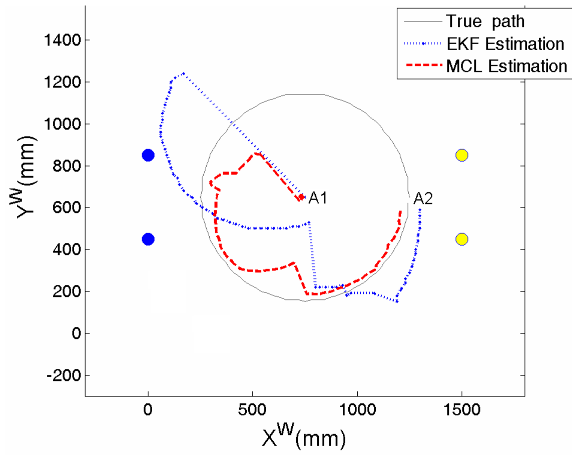
(b)



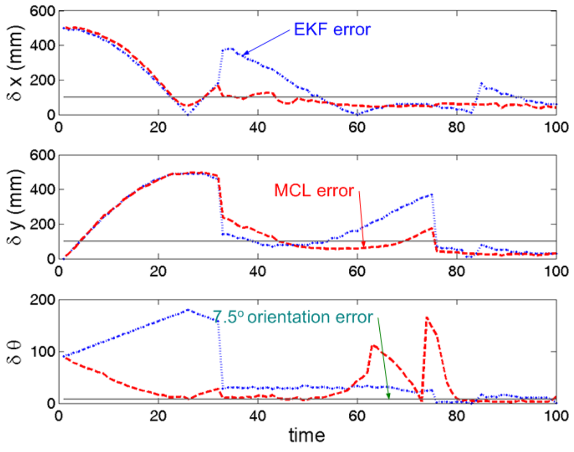
(c)



(d)

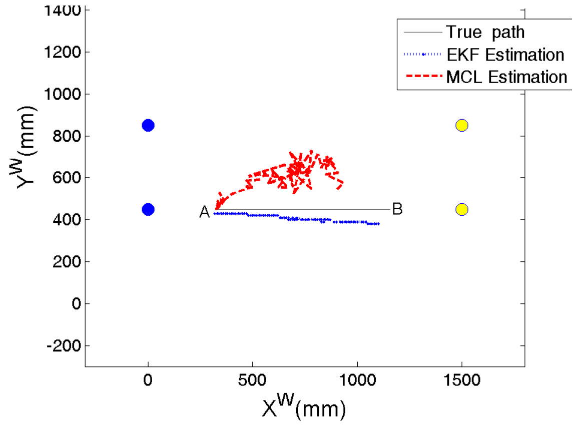


(e)

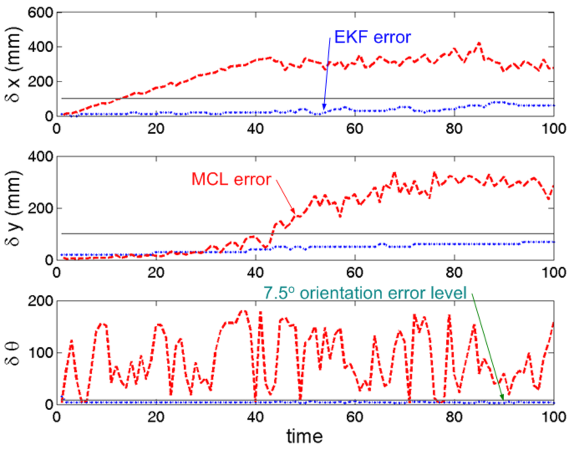


(f)

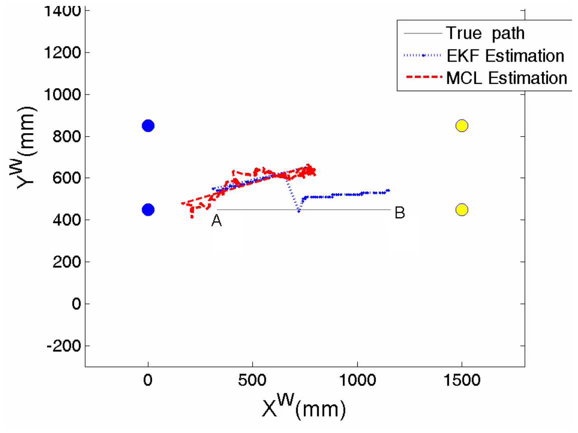
Figure 6.29: Path estimation comparison for the case shown in Fig. 6.26. The estimated path and absolute error in x, y and θ for the EKF and MCL is shown blue dotted and red dashed, respectively. The gray lines in (b), (d) and (f) shows the 100 mm error limit in x and y and 7.5 °in θ (a) known start position (b) error in x, y, and θ (c) random start position (d) error comparison (e) start position estimated with with single landmark based method (f) error



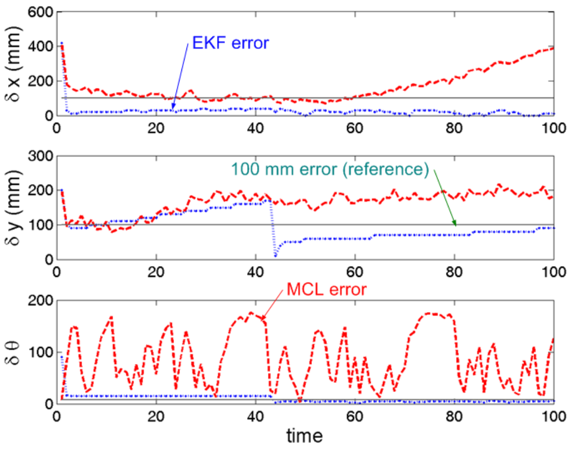
(a)



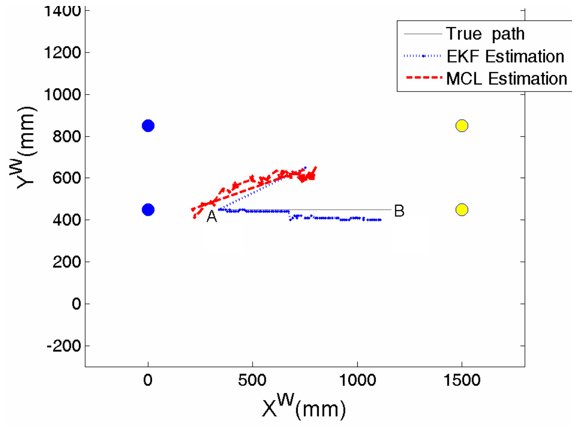
(b)



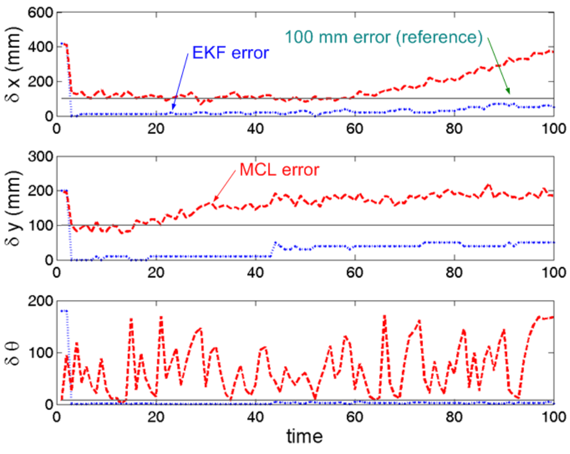
(c)



(d)

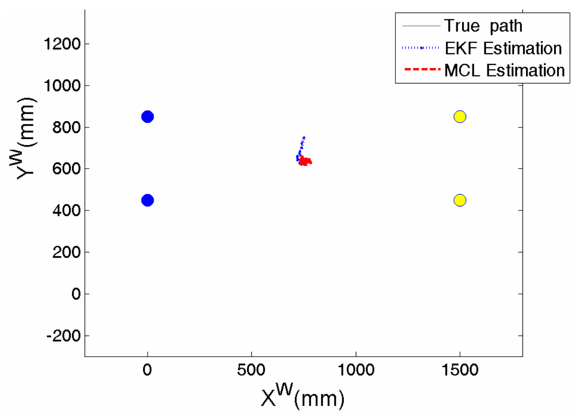


(e)

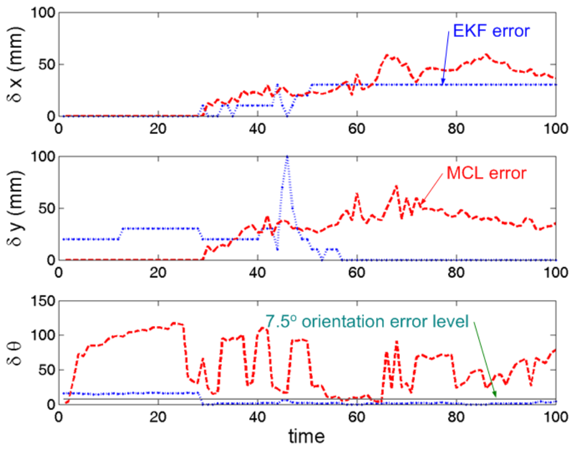


(f)

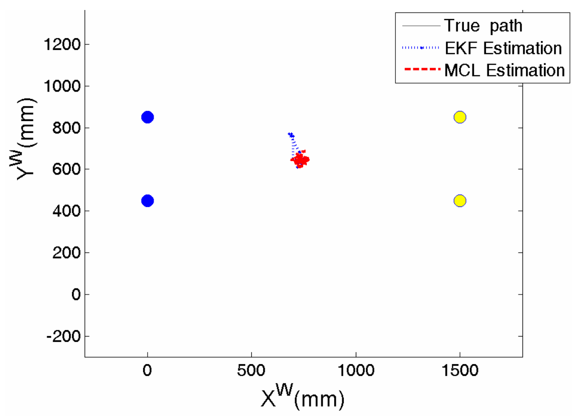
Figure 6.30: Path estimation comparison for the case shown in Fig. 6.27



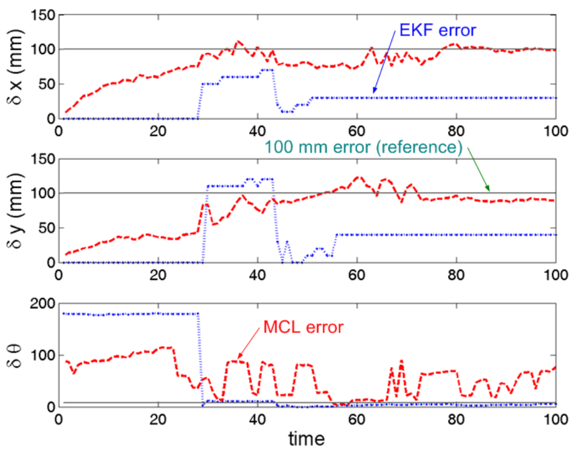
(a)



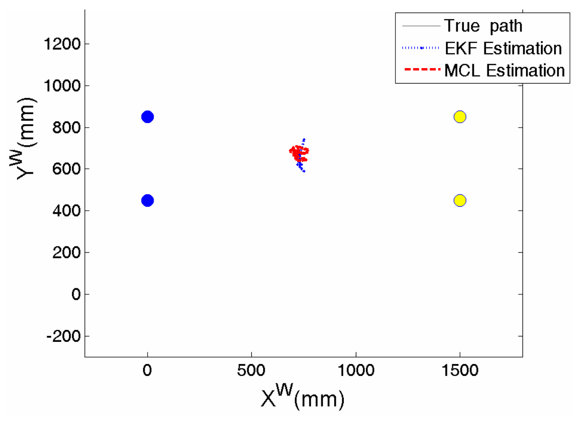
(b)



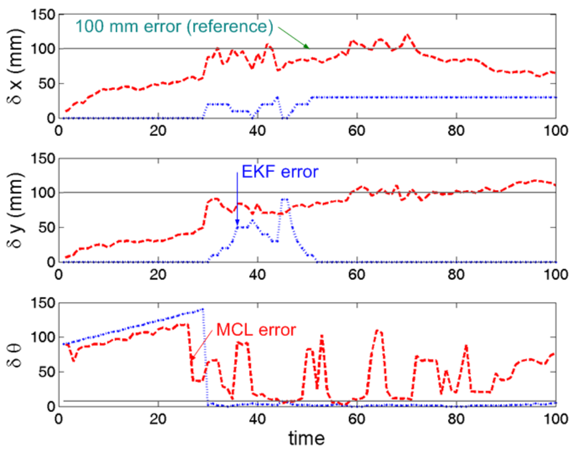
(c)



(d)



(e)



(f)

Figure 6.31: Path estimation comparison for the case shown in Fig. 6.28

Table 6.14: Statistical results (distance to the true position) for parts of the circular path shown in Fig. 6.29. First landmark is sighted at step 32

Start position	Method	Steps	Mean (mm)	Std (mm)	Min (mm)	Max (mm)
Known	EKF	33—79	60.41	53.63	10.40	197.60
	MCL	33—79	105.83	28.11	65.71	169.01
	EKF	80—100	33.18	32.33	10.40	114.40
	MCL	80—100	107.81	8.53	89.42	125.34
Random	EKF	33—79	235.66	92.95	93.60	457.60
	MCL	33—79	120.24	64.13	41.41	266.96
	EKF	80—100	51.01	19.43	31.20	93.60
	MCL	80—100	40.95	4.66	31.49	51.07
Global	EKF	33—79	241.41	102.95	62.40	416.00
	MCL	33—79	140.52	58.01	77.28	283.22
	EKF	80—100	96.08	46.85	10.40	197.60
	MCL	80—100	61.54	8.91	44.66	74.85

Table 6.15: Statistical results (distance to the true position) for parts of the linear path shown in Fig. 6.30

Start position	Method	Steps	Mean (mm)	Std (mm)	Min (mm)	Max (mm)
Known	EKF	10—79	49.03	17.27	20.80	72.80
	MCL	10—79	337.38	133.00	73.73	529.26
	EKF	80—100	90.63	5.83	83.20	104.00
	MCL	80—100	486.13	44.09	408.14	579.04
Random	EKF	10—79	95.83	41.37	31.20	166.40
	MCL	10—79	218.99	38.71	151.54	314.06
	EKF	80—100	72.80	6.58	62.40	83.20
	MCL	80—100	394.09	47.53	312.22	472.50
Global	EKF	10—79	30.16	18.24	10.40	62.40
	MCL	10—79	219.24	43.83	142.78	338.27
	EKF	80—100	70.82	7.07	62.40	83.20
	MCL	80—100	392.75	44.32	312.54	457.97

line motion are shown from third to sixth column respectively. Overall results from all the three categories are shown in the last two columns.

The rise in observation error with distance from landmark features is adequately captured by a corresponding increase in uncertainty. However, the error that arises from segmentation problems is not handled. The observation error is comparatively low for the case when robot is moving along circular paths. This is due to the fact that in this case landmarks are observed from relatively short distances.

The main problem that stems from the high observation error is the slow repositioning of the robot if it has a wrong position with low uncertainty. The fusion of an erroneous observation reduces the robot position uncertainty even further but causes less improvement in its error. To tackle this problem a validation gate is implemented which is based on innovation. If innovation is above a certain threshold the robot is declared lost and global position estimation is forced. After global position estimation the location tracker is initialized with the new

Table 6.16: Statistical results (distance to the true position) for parts of the single point point rotation shown in Fig. 6.31. First landmark is sighted at step 29

Start position	Method	Steps	Mean (mm)	Std (mm)	Min (mm)	Max (mm)
Known	EKF	30—79	33.28	12.78	20.80	104.00
	MCL	30—79	53.92	20.29	15.38	102.02
	EKF	80—100	31.20	0.00	31.20	31.20
	MCL	80—100	69.71	8.56	56.40	83.41
Random	EKF	30—79	70.10	42.59	10.40	145.60
	MCL	30—79	149.27	11.40	121.14	176.62
	EKF	80—100	52.00	0.00	52.00	52.00
	MCL	80—100	156.67	3.50	150.77	166.06
Global	EKF	30—79	37.44	14.25	20.80	93.60
	MCL	30—79	150.24	15.59	118.86	178.24
	EKF	80—100	31.20	0.00	31.20	31.20
	MCL	80—100	155.38	3.93	147.39	160.83

Table 6.17: The robot observation error

	Circular path		Point		Line		All	
	δr (mm)	$\delta\varphi$	δr (mm)	$\delta\varphi$	δr (mm)	$\delta\varphi$	δr (mm)	$\delta\varphi$
Mean	-4.67	-0.19 °	-38.31	-0.17 °	-32.16	-0.37 °	-31.04	-0.30 °
Std	22.43	0.17 °	43.52	0.85 °	46.61	0.70 °	44.83	0.72 °
Min	0.25	0.0007 °	0.007	0.0003 °	0.07	0.0006 °	0.007	0.0003 °
Max	70.12	0.53 °	163.17	1.93 °	204.60	1.78 °	204.60	1.93 °

estimate and its corresponding uncertainty. A similar problem has already been discussed in Section 5.1.3, where a robot which is certain about its position is displaced. The robot detected such a situation by analyzing the innovation sequence.

Simulation results illustrate that the algorithm can successfully localize the robot despite of the fact that landmarks are sparse. The linearization of the observation model introduces problems for distant features. The robot position error is perfectly bounded when it is following a circular path. However, the error is not always bounded in the latter two categories. The reason for this is that in motion along the circle, features are always observed from a short distance whereas in the other cases features are observed from a larger distance towards the end of the trials. The prominence of this problem comes from the use of a narrow baseline stereo.

The method that requires two landmarks for global self-localization is the desired one since it does not require the robot to known its absolute orientation. The drawback of this method is the difficulty to acquire two landmarks simultaneously, specially from short distances. As shown by the simulation results that the probably of acquiring single landmark is high as compared to two landmarks and the method can be used from most locations in the robot environment. However, estimation of the absolute orientation of the robot in indoor environments requires tedious calibration and is inaccurate if other robots are in close vicinity of the robot.

The comparison study concludes that both algorithms perform well for their average estimates. The comparison shows that estimates obtained by EKF have fewer errors and are closer to the true values than MCL. In MCL (for circular path) the mean error is about 6 cm and it increases with decrease of number of particles. Further, tests to determine the optimal number of particles for the PF based localization approach have been performed. 3,000 particles mark the point above which no additional quality could be gained and below which the result is varying [BDN07]. These variations are caused by the initial distribution of the particles and values generated by the random number generator. With fewer particles the shape of the path is disturbed by many peaks and the robot appears to move back and forth [Deu07].

Chapter 7

Discussion and Future Work

Concluding remarks and direction for future work is presented in this chapter. Emphasis is given to the requirements listed in the Introduction chapter of this thesis.

7.1 Discussion

The work presented in this thesis is a step towards realization of tiny autonomous mobile robots. It has been shown that the robot can successfully localize itself within its environment, which is the basic requirement for autonomous behavior. The discussed approach to self-localization is a combination of global and local position estimation methods. Methods based on local sensors have not been used since they require a manual starting position and, moreover, position error accumulate unbounded as the robot move in its environment. The robot position at startup (a solution to the bootstrap problem) is estimated using global position estimation methods, which extract distinct features that already exist in the robot environment. Globally distinct features being scarce, localization algorithms were required to use as few landmarks as possible. The localization algorithm is not dependent on specially designed landmarks or on there placement at a specific location in the robot environment. Reduction in number of landmarks has been achieved by using range based methods. It has been shown that just one landmark is required if absolute orientation of the robot is estimated independently.

Global self-localization requires extraction of at least one global distinct landmark or two if the absolute orientation of the robot is not known, which may not be possible at every step as landmarks are scarce and are not available through the entire state space. Therefore, the robot position is tracked once it is estimated. During tracking, the uncertainty grows, which is suppressed when external observations of landmark features are available. The global position estimation methods and their uncertainty analysis provide the required information to initialize a location tracker such as an EKF, which results in significant performance improvement over random initialization. The global position estimation also compensates the drawback of EKF to bootstrap in a recurrent environment due to its unimodal nature.

Comparison of EKF and MCL has shown that EKF performs better or at least comparable to MCL. The plus point of EKF over MCL is that its computation time is at least 10 times less than that compared to MCL [GF02]. Comparison of computation time for both the methods on the Tinyphoon is within the scope of future work. The kidnap robot problem has been addressed using a validation gate on the innovation sequence. Detection of a kidnap case and successful re-localization of the robot has been demonstrated.

The stereo vision system has been used as the robot primary external sensor, which solves the problem of range computation without enforcing special constraints on object positions or size. Using a single image for range computation whole object has to be visible in camera images. For example the entire goal area has to be visible to compute range to it. This is something unlikely as goals are frequently occluded. Features used are line based and color transitions. Line based features are extracted using GBHT while color transitions are detected using color segmentation methods.

Details of constructing a sparse 3D map have been reported in our work [Ent05]. Implementation details of edge detection, blob detection and extraction of 3D lines on the Tinyphoon robot has been reported in [Bad07]. It has been shown that a frame rate of 5 Hz can be achieved using the onboard processors of the Tinyphoon robot. Currently, a frame rate of 12 Hz can be reached by narrowing the window of interest. This frame rate is adequate for self-localization but not for detection of game ball and other robots. A game where the robot should play against state-of-the-art robots with vision systems above the playing field (MiroSot Small and Middle-Size Leagues [FIR07b, FIR07a]) might result in a failure for the Tinyphoon robot. The game ball speed (in MiroSot up to 60km/h) makes the vision sensor less effective [Bad07]. However, the new proposed league, AMiroSot, is an environment where the robot can face opponents with similar problems [KDB$^+$06]. Tinyphoon's capabilities to play in such an environment have already been demonstrated at the FIRA World championship 2006 at Dortmund in the RoboSot league [FIR06]. Nevertheless, fast vision computations are required. One way of achieving a fast first computation is to use scale-space methods which is discussed in the next section.

7.2 Future Work

The hard real-time requirements warrant research to introduce an efficient first computation that can be refined later. It is proposed as a future work to develop a new scale-space based stereo algorithm that allows the computation of the range images at different scales which is essential in real-time environments. First a course and fast estimation of the depth is obtained, and then detailed and more accurate results are computed.

A basic property of real-world objects is that they only exist as meaningful entities over certain ranges of scale [Lin94b]. For example when looking at moving people at a distance (coarse scale) it is important to identify them as a woman or man or to determine direction of their movement. At a fine scale it is more appropriate to talk about small features of the individual person, like the hair style or the face. To model the structure of the real world the concept of scale plays an important role in computer vision. If the scale is too low for a certain problem it can be refined by foveating interesting structures, or if necessary, moving closer to the interesting object. This movement process makes it possible to acquire additional information about the three-dimensional structure of an object. So the main tasks to be solved by vision algorithms are as to what kind of information should be extracted at the initial stages, and which operations should be performed on the data that reach the visual sensors.

The general idea of representing a signal on multiple scales is not new. Early work was performed by Rosenfeld in 1971 [RT71], who observed the advantage of using operators of different sizes in edge detection. Several authors used different levels of resolution [Kli71, Uhr72]. These approaches have been developed further by Burt [BA83], Crowley [CS87], Kropatsch, et al. [Kro91] to the *image pyramid*. The main advantage of the pyramid representation is the rapidly decreasing image size, thus reducing the computational work in pre-processing. There is vast literature on different aspects of pyramid representation [Bur81, BA83, CS87, MBR87, Nac95, KY96]. An overview can be found in [Kro91] and [JR94].

The scale-space representation is a special type of multi-scale representation that comprises of a continuous scale parameter and preserves the same spatial sampling at all scales. It is an embedding of the original signal into a one-parameter family of derived signals constructed by convolution with a one-parameter family of Gaussian kernels of increasing width [Lin94b]. This operation is known by the term *scale-space smoothing* [Lin94a]. In this representation the fine scale information is successively suppressed, preserving the same resolution. One of the major reasons for using a multi-scale representation is to remove fine details for pre-processing in order to have low noise conditions and to eliminate unnecessary scene details. An important property of scale-space is that features at different scales can be related to each other in a precise manner [Lin94a]. In addition to the constraints and consistency checks several control strategies have been proposed by many researchers to reduce ambiguous correspondences and enhance stereo matching. These include, for instance, hierarchical matching strategies [MT89, MS98]. It is required to use the scale-space approach for range estimation and adapt it to feature-based stereo, since it speeds up the entire range computation while preserving the accuracy. So far this approach is only used for area-based correspondence analysis.

The work reported in this dissertation address only the position estimation and assume that the environment is known. In future this work could be extended in a way that the robot is able to explore its environment and build a map of it. Map building is the dual of position estimation as it requires knowledge of the robot position. Therefore, there is a need to address both of them simultaneously i.e. Simultaneous Localization and Map Building (SLAM). Realization of a tiny robot with vision as its primary sensor and capable of learning its environment for the purpose of navigation will be an interesting and challenging research topic.

An extended version of this work could be applied to lateral positioning of vehicles in a lane. Although GPS is used for position calculation in out door environments but GPS signals may be obstructed by tall and large buildings. One significant limitation of feature based binocular stereo vision is that it cannot match pixels on line segments if they lie on or close to the epipolar lines. This ambiguity can be resolved by adding a third camera [WT99]. However, such extension could be made only if it can be supported by the on-board processing unit.

Appendix A

Notations and Coordinate Systems

A.1 Notations

The notation used in this thesis broadly follows conventions in computer vision and robotics. However, exact meanings of certain representations is described here.

Coordinate Systems

A coordinate system is an imaginary alignment of x,y,z axes in Euclidean 3D space. In this thesis it is denoted by a capital letter such as W or a capital letter and a number such as C0.

Vectors

Vectors are represented in lower case bold type such as $\mathbf{p}$. Superscripts are used to identify the coordinate system of interest in which the vector are represented. The superscripts are however dropped to simplify notation in cases where the coordinate system is obvious from the context or the vector is independent of any coordinate system introduced in this thesis. Lower case subscripts such as l are sometimes used to differentiate between different vectors of the same name, whereas, numbers in the subscript mean multiple instances of the same type of vector. k in the subscript refers to step number (time).

Matrices

Matrices are denoted in upper case bold type such as $\mathbf{U}$ or $\mathbf{\Sigma}$. Subscripts are sometimes used to differentiate between different vectors though ones with a similar role. Different elements in a matrix are differentiated with subscript ij representing a particular element at ith row and jth column e.g. U_{23} is an element of $\mathbf{U}$ at 2nd row and 3rd column.

Rotations

Rotations are special matrices that transform a vector from one coordinate system to another. A rotation matrix which produces the components of a vector in frame C0 from its components in frame C1 is defined as $\mathbf{R}^{\mathrm{C0C1}}$ and can be written as follows:

$$\mathbf{a}^{\mathrm{C0}} = \mathbf{R}^{\mathrm{C0C1}}\mathbf{a}^{\mathrm{C1}} \tag{A.1}$$

Similarly, the reverse transformation is simply written as:

$$\mathbf{a}^{\mathrm{C1}} = \mathbf{R}^{\mathrm{C1C0}}\mathbf{a}^{\mathrm{C0}} \tag{A.2}$$

where $\mathbf{R}^{\mathrm{C1C0}} = (\mathbf{R}^{\mathrm{C0C1}})^T$ due to the fact that a rotation matrix is orthogonal.

A.2 Coordinate Systems

The coordinate systems used in this thesis are shown in Fig. A.1. The world coordinate system is denoted with superscript W. Objects in the world coordinate system have coordinates x, y, and z. The robot motion is always assumed to be a flat surface therefore the robot position is denoted by x, y, and θ. Besides this coordinate system five other coordinate systems are introduced as follows:

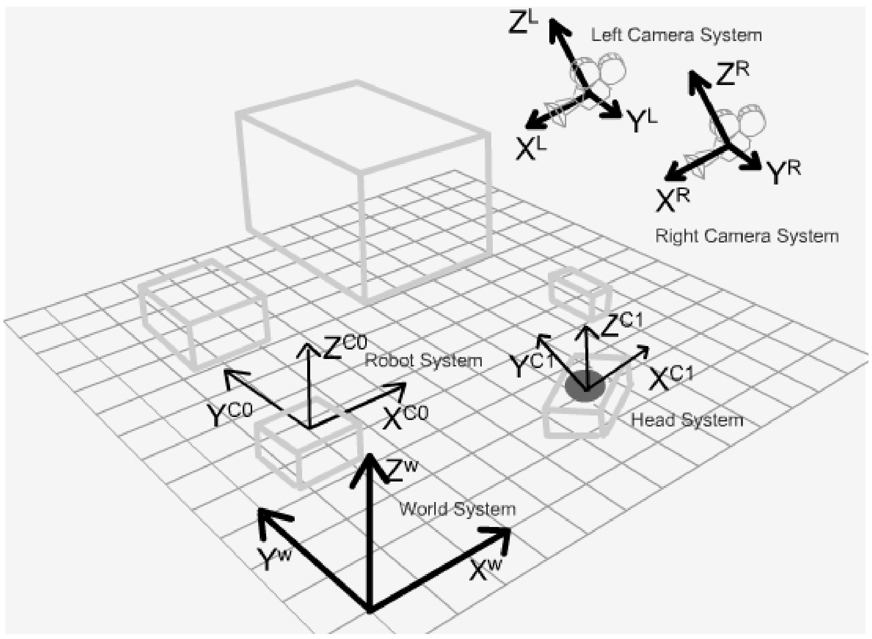


Figure A.1: Coordinate Systems

Robot Coordinate System

The robot coordinate system is represented by C0. It is fixed with respect to the center. The origin of this coordinate system lies at the robot center of mass. Its x-axis lies in the forward direction with the z-axis pointing upward and y-axis to the left. The origin of this coordinate system with respect to the origin of the world coordinate system gives the position and its orientation with the world coordinate defines the orientation of the robot. Objects in the robot coordinate system are represented by x^{C0}, y^{C0}, and z^{C0} coordinates.

Head Coordinate System

The head coordinate system is represented by C1. This coordinate system is fixed to the robot head and is rotated with respect to C0 by rotation angle γ. With $\gamma = 0$, both C0 and C1 are aligned. The origin of this coordinate system lies at the robot center of mass.

Vision Coordinate System

The stereo vision coordinate system is represented by C2. C2 is translated and rotated with respect to C1. The translation and rotation is constant along x-axis and y-axis, respectively.

Cameras Coordinate Systems

The left and right cameras coordinate systems are represented by L and R respectively. These coordinate systems are fixed relative to the two cameras. Their x-axes are aligned with the camera optic axes, and their y and z axes lie parallel to their image planes. These coordinate systems are aligned with C2.

The axes of the image plane are represented by u, v. An image pixel irrespective of the camera will be denoted as (u, v). Image pixels from the left and right cameras are differentiated by subscripts l and r, respectively. Multiple instances of the same type are differentiated by a number in the subscript. For example two points in the robot coordinate system are represented by $\begin{bmatrix} x_1^{\mathrm{C0}} & y_1^{\mathrm{C0}} & z_1^{\mathrm{C0}} \end{bmatrix}^T$ and $\begin{bmatrix} x_2^{\mathrm{C0}} & y_2^{\mathrm{C0}} & z_2^{\mathrm{C0}} \end{bmatrix}^T$, while two pixels in the left camera image are denoted by $\begin{bmatrix} u_{l1} & v_{l1} \end{bmatrix}^T$ and $\begin{bmatrix} u_{l2} & v_{l2} \end{bmatrix}^T$.

Position of the head of the robot is fixed with respect to its center. It can only rotate about z-axis. The transformation of a vector $\mathbf{a}^{\mathrm{C0}}$ to vision coordinate system is formulated as follows:

$$\mathbf{a}^{\mathrm{C2}} = \mathbf{R}^{\mathrm{C2C1}}((\mathbf{R}^{\mathrm{C1C0}}\mathbf{a}^{\mathrm{C0}}) + \mathbf{t}) \tag{A.3}$$

Using (A.2) the reverse transformation can be written as:

$$\mathbf{a}^{\mathrm{C0}} = \mathbf{R}^{\mathrm{C0C1}}(\mathbf{R}^{\mathrm{C1C2}}\mathbf{a}^{\mathrm{C2}} - \mathbf{t}) \tag{A.4}$$

where

$$\mathbf{t} = \begin{bmatrix} x_c & 0 & 0 \end{bmatrix}^T \tag{A.5}$$

$$\mathbf{R}^{\mathrm{C2C1}} = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{A.6}$$

$$\mathbf{R}^{\mathrm{C1C0}} = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \tag{A.7}$$

$$\mathbf{R}^{\mathrm{C0C1}} = (\mathbf{R}^{\mathrm{C1C0}})^T \tag{A.8}$$

$$\mathbf{R}^{\mathrm{C1C2}} = (\mathbf{R}^{\mathrm{C2C1}})^T \tag{A.9}$$

In the above equation x_c and β are constants, whereas, γ is a variable.

Appendix B

Odometry Model

B.1 Geometric Construction of a Differential Drive Robot

It is assumed that the two wheeled differential drive robot is moving on flat surface with the robot pose[1] having 3 degrees of freedom as shown in Fig. B.1. The separation between the two wheels of the robot is w. The robot center is represented by o. Movement of the robot center is considered as motion of the whole robot. The type of trajectory followed by the robot depends on the velocity of each wheel. If the two values are equal the robot travels along a straight line. A curved path is traversed if the two wheels rotate at different velocities, whereas the robot rotates around its center of mass if the velocities are equal in magnitude but opposite in direction.

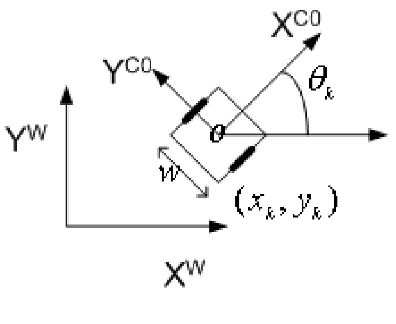


Figure B.1: Representation of the robot pose

Fig. B.2 shows the robot's trajectory between time step $k-1$ and k. The distance covered (per unit time) by the left and right wheels of the robot is denoted by v_{lk} and v_{rk} respectively. The three components of state change in robot coordinate system are represented by δx_k, δy_k and $\delta\theta_k$. Whereas, α_k denotes change in robot orientation, c_k the radius of curvature and v_k is the robot velocity. The relations between these quantities and the distance traveled by the two wheels are given by (B.1), (B.2) and (B.3). It is assumed that between time $k-1$ and k, the robot moves with constant velocity, hence, the robot follows a path of constant radius of curvature. The robot wheel encoders provide wheel counts which are transformed into the distance covered by the two wheels. The relationship between distance covered by each

[1]If not specified, state, pose and position refer to the same quantity

wheel of the robot with the robot velocity and radius of curvature is illustrated in Fig. B.2 and given by the following expressions:

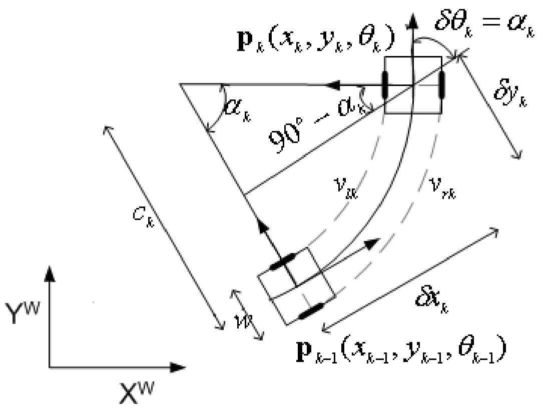

$\mathbf{p}_k(x_k, y_k, \theta_k)$
$\delta\theta_k = \alpha_k$
δy_k
α_k
$90^\circ - \alpha_k$
c_k
v_{lk}
v_{rk}
δx_k
w
$\mathbf{p}_{k-1}(x_{k-1}, y_{k-1}, \theta_{k-1})$
Y^W
X^W

Figure B.2: Geometric construction of differential drive robot

$$\alpha_k = \frac{v_{rk} - v_{lk}}{w} \tag{B.1}$$

$$v_k = \frac{v_{rk} + v_{lk}}{2} \tag{B.2}$$

$$c_k = \frac{w(v_{rk} + v_{lk})}{2(v_{rk} - v_{lk})} \tag{B.3}$$

These expressions are used to formulate the control vector $\mathbf{u}_k$, which is the state change in robot frame of reference. This is called control vector as it is based only on v_{rk} and v_{lk}. Further illustration of the construction of control vector is given in Fig. B.3, where only relevant elements of Fig. B.2 are shown. Using geometric relationship between different elements of Fig. B.3 the control vector can be formally stated as follows:

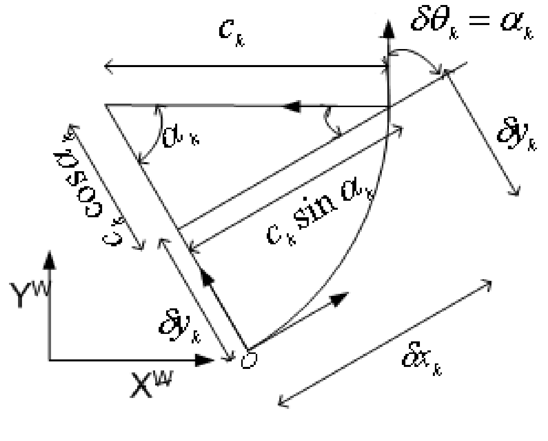


Figure B.3: Construction of the robot's control vector

$$\mathbf{u}_k = \begin{bmatrix} \delta x_k \\ \delta y_k \\ \delta\theta_k \end{bmatrix} = \mathbf{g}\left(\begin{bmatrix} v_{lk} \\ v_{rk} \end{bmatrix}\right) = \begin{bmatrix} c_k sin\alpha_k \\ c_k(1 - cos\alpha_k) \\ \alpha_k \end{bmatrix} \tag{B.4}$$

or in more detail by the following equation:

$$\mathbf{u}_k = \begin{bmatrix} \delta x_k \\ \delta y_k \\ \delta \theta_k \end{bmatrix} = \begin{bmatrix} \frac{w(v_{rk}+v_{lk})}{2(v_{rk}-v_{lk})} sin(\frac{v_{rk}-v_{lk}}{w}) \\ \frac{w(v_{rk}+v_{lk})}{2(v_{rk}-v_{lk})}(1 - cos(\frac{v_{rk}-v_{lk}}{w})) \\ \frac{v_{rk}-v_{lk}}{w} \end{bmatrix} \tag{B.5}$$

This is a generalized expression for the control vector for the case when the robot is following a curved path and is required to predict future state of the robot as given by (5.7). For calculation of prediction uncertainty, expressions for $\mathbf{U}_k$, $\mathbf{J}_1$ and $\mathbf{J}_2$ are required. Uncertainty analysis of the control vector is given in the next section, whereas, expressions for $\mathbf{J}_1$ and $\mathbf{J}_2$ are derived in Section B.3 along with compounding of the control vector with the current location to arrive at the new location at time k. In this discussion it is assumed that at time $k-1$ an estimate $\widehat{\mathbf{p}}_{k-1|k-1}$ of the robot pose and its uncertainty $\mathbf{P}_{k-1|k-1}$ is available.

B.2 Control Vector Uncertainty

For uncertainty analysis of the control vector it is assumed that the robot's odometry is calibrated for systematic errors using methods such as the University of Michigan Benchmark (UMBmark) [BF96]. Analysis of the non-systematic errors start by making a basic assumption about error in distance traveled by the robot and propagate it to the robot control vector as discussed in the following paragraphs. Suppose the deviation $\widetilde{\mathbf{v}}_k$ of the estimated velocity vector $\widehat{\mathbf{v}}_k$ from its true value $\mathbf{v}_k$ is a random vector of zero mean and covariance $\mathbf{\Sigma}_v$. As the two wheels are driven by two different motors and the distance covered by each wheel is measured by two independent encoders, it is reasonable to assume that error in distance covered by the left wheel is independent of the error in distance covered by the right wheel [CK97]. The covariance matrix of this error is given by the following equation:

$$\mathbf{\Sigma}_v = E\{\widetilde{\mathbf{v}}_k \widetilde{\mathbf{v}}_k^T\} = \begin{bmatrix} \sigma_l^2 & 0 \\ 0 & \sigma_r^2 \end{bmatrix} \tag{B.6}$$

where σ_l^2 and σ_r^2 are the variances of v_{lk} and v_{rk} and are proportional to their absolute values.

Expression for covariance matrix of error in $\mathbf{u}_k$ can be derived by writing (B.4) using Taylor series expansion around $\widehat{\mathbf{v}}_k$ as follows:

$$\mathbf{u}_k = \mathbf{g}(\widehat{\mathbf{v}}_k) + \mathbf{J}_u[\widetilde{\mathbf{v}}_k] + \dots \tag{B.7}$$

where $\widehat{\mathbf{u}}_k = \mathbf{g}(\widehat{\mathbf{v}}_k)$ and

$$\mathbf{J}_u = \begin{bmatrix} \frac{-(v_{rk}^2-v_{lk}^2)\cos(\frac{v_{rk}-v_{lk}}{w})+2wv_{rk}\sin(\frac{v_{rk}-v_{lk}}{w})}{2(v_{rk}-v_{lk})^2} & \frac{(v_{rk}^2-v_{lk}^2)\cos(\frac{v_{rk}-v_{lk}}{w})-2wv_{lk}\sin(\frac{v_{rk}-v_{lk}}{w})}{2(v_{rk}-v_{lk})^2} \\ \frac{-(v_{rk}^2-v_{lk}^2)\sin(\frac{v_{rk}-v_{lk}}{w})+2wv_{rk}(1-\cos(\frac{v_{rk}-v_{lk}}{w}))}{2(v_{rk}-v_{lk})^2} & \frac{(v_{rk}^2-v_{lk}^2)\sin(\frac{v_{rk}-v_{lk}}{w})-2wv_{lk}(1-\cos(\frac{v_{rk}-v_{lk}}{w}))}{2(v_{rk}-v_{lk})^2} \\ \frac{-1}{w} & \frac{1}{w} \end{bmatrix} \tag{B.8}$$

is the jacobian of $\mathbf{u}_k$ with respect to $\mathbf{v}_k$ evaluated at $\widehat{\mathbf{v}}_k$. Retaining only the first two terms of (B.7), expression for $\widetilde{\mathbf{u}}_k$ becomes:

$$\begin{aligned} \widetilde{\mathbf{u}}_k &= \mathbf{u}_k - \widehat{\mathbf{u}}_k \\ &\approx \mathbf{g}(\widehat{\mathbf{v}}_k) + \mathbf{J}_u \widetilde{\mathbf{v}}_k - \mathbf{g}(\widehat{\mathbf{v}}_k) \\ &= \mathbf{J}_u \widetilde{\mathbf{v}}_k \end{aligned} \tag{B.9}$$

Hence, $\mathbf{U}_k$ can be written as follows:

$$\begin{aligned}\mathbf{U}_k &= E\{\widetilde{\mathbf{u}}_k\widetilde{\mathbf{u}}_k^T\}\\ &= E\{(\mathbf{J}_u\widetilde{\mathbf{v}}_k)(\mathbf{J}_u\widetilde{\mathbf{v}}_k)^T\}\\ &= \mathbf{J}_u E\{\widetilde{\mathbf{v}}_k\widetilde{\mathbf{v}}_k^T\}\mathbf{J}_u^T\\ &= \mathbf{J}_u\mathbf{\Sigma}_v\mathbf{J}_u^T \qquad (\text{B.10})\end{aligned}$$

Substitution of (B.6) and (B.8) in (B.10) results in the following elements of $\mathbf{U}_k$, where U_{ij} represents an element at ith row and jth column:

$$\begin{aligned}
U_{11} &= \frac{1}{4(v_{rk}-v_{lk})^4}((v_{rk}^2-v_{lk}^2)^2(\sigma_l^2+\sigma_r^2)\cos^2\alpha_k + 4w^2(v_{rk}^2\sigma_l^2+v_{lk}^2\sigma_r^2)\sin^2\alpha_k\\
&\quad -4w(v_{rk}^2-v_{lk}^2)(v_{rk}\sigma_l^2+v_{lk}\sigma_r^2)\sin\alpha_k\cos\alpha_k)\\
U_{12} &= \frac{1}{4(v_{rk}-v_{lk})^4}((v_{rk}^2-v_{lk}^2)^2(\sigma_l^2+\sigma_r^2)\cos\alpha_k\sin\alpha_k\\
&\quad +4w^2(v_{rk}^2\sigma_l^2+v_{lk}^2\sigma_r^2)\sin\alpha_k(1-\cos\alpha_k)\\
&\quad -2w(v_{rk}^2-v_{lk}^2)(v_{rk}\sigma_l^2+v_{lk}\sigma_r^2)(\cos\alpha_k(1-\cos\alpha_k)+\sin^2\alpha_k))\\
U_{13} &= \frac{1}{2w(v_{rk}-v_{lk})^2}((v_{rk}^2-v_{lk}^2)(\sigma_l^2+\sigma_r^2)\cos\alpha_k - 2w(v_{rk}\sigma_l^2+v_{lk}\sigma_r^2)\sin\alpha_k)\\
U_{21} &= U_{12}\\
U_{22} &= \frac{1}{4(v_{rk}-v_{lk})^4}((v_{rk}^2-v_{lk}^2)^2(\sigma_l^2+\sigma_r^2)\sin^2\alpha_k + 4w^2(v_{rk}^2\sigma_l^2+v_{lk}^2\sigma_r^2)(1-\cos\alpha_k)^2\\
&\quad -4w(v_{rk}^2-v_{lk}^2)(v_{rk}\sigma_l^2+v_{lk}\sigma_r^2)\sin\alpha_k(1-\cos\alpha_k))\\
U_{23} &= \frac{1}{2w(v_{rk}-v_{lk})^2}((v_{rk}^2-v_{lk}^2)(\sigma_l^2+\sigma_r^2)\sin\alpha_k - 2w(v_{rk}\sigma_l^2+v_{lk}\sigma_r^2)(1-\cos\alpha_k))\\
U_{31} &= U_{13}\\
U_{32} &= U_{23}\\
U_{33} &= \frac{1}{w^2}(\sigma_l^2+\sigma_r^2)
\end{aligned}$$

B.3 Compounding of Transformation

Motion model (5.1) states that the robot pose changes from $\mathbf{p}_{k-1}$ to $\mathbf{p}_k$ under the influence of control vector $\mathbf{u}_k$. In our application a simplified version of this transition is illustrated in Fig. B.4, which helps us in incorporating the control vector with the current state to arrive at a new state of the robot. The illustration in Fig. B.4 show the following relationship between different terms:

$$\begin{aligned}d_k &= \sqrt{\delta x_k^2+\delta y_k^2}\\ &= \sqrt{dx_k^2+dy_k^2} \qquad (\text{B.11})\end{aligned}$$

where dx, dy are the total change in x and y and δx, δy are components of the control vector. Similarly:

$$\delta x_k = d_k\cos\alpha_k \qquad (\text{B.12})$$

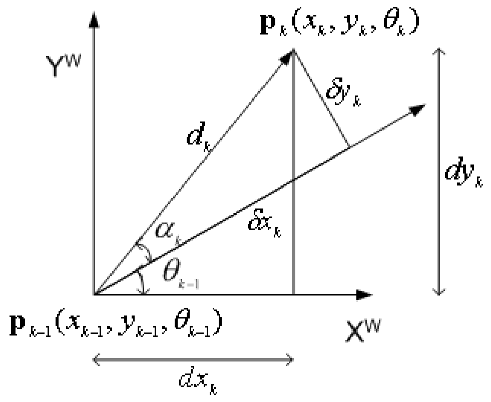

$\mathbf{p}_k(x_k, y_k, \theta_k)$
Y^W
δy_k
d_k
dy_k
α_k
δx_k
θ_{k-1}
$\mathbf{p}_{k-1}(x_{k-1}, y_{k-1}, \theta_{k-1})$
X^W
dx_k

Figure B.4: Compounding of transformation

$$\delta y_k = d_k \sin \alpha_k \tag{B.13}$$

and

$$\begin{aligned} dx_k &= d_k \cos(\alpha_k + \theta_{k-1}) \\ &= d_k \cos(\alpha_k) \cos(\theta_{k-1}) - d_k \sin(\alpha_k) \sin(\theta_{k-1}) \\ &= \delta x_k \cos(\theta_{k-1}) - \delta y_k \sin(\theta_{k-1}) \end{aligned} \tag{B.14}$$

$$\begin{aligned} dy_k &= d_k \sin(\alpha_k + \theta_{k-1}) \\ &= d_k \sin(\alpha_k) \cos(\theta_{k-1}) + d_k \cos(\alpha_k) \sin(\theta_{k-1}) \\ &= \delta y_k \cos(\theta_{k-1}) + \delta x_k \sin(\theta_{k-1}) \end{aligned} \tag{B.15}$$

Using the illustration of Fig. B.4 and (B.14) and (B.15), (5.1) may be written as follows:

$$\mathbf{p}_k = \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} = \begin{bmatrix} x_{k-1} + \delta x_k cos(\theta_{k-1}) - \delta y_k sin(\theta_{k-1}) \\ y_{k-1} + \delta x_k sin(\theta_{k-1}) + \delta y_k cos(\theta_{k-1}) \\ \theta_{k-1} + \alpha_k \end{bmatrix} \tag{B.16}$$

and prediction (5.7) in the following form:

$$\widehat{\mathbf{p}}_{k|k-1} = \begin{bmatrix} x_{k|k-1} \\ y_{k|k-1} \\ \theta_{k|k-1} \end{bmatrix} = \begin{bmatrix} x_{k-1|k-1} + \delta x_k cos(\theta_{k-1|k-1}) - \delta y_k sin(\theta_{k-1|k-1}) \\ y_{k-1|k-1} + \delta x_k sin(\theta_{k-1|k-1}) + \delta y_k cos(\theta_{k-1|k-1}) \\ \theta_{k-1|k-1} + \alpha_k \end{bmatrix} \tag{B.17}$$

After having derived the above equations, it is time to write expressions for $\mathbf{J}_1$ and $\mathbf{J}_2$ as used in (5.6) and given by (B.18) and (B.19) below:

$$\mathbf{J}_1 = \begin{bmatrix} 1 & 0 & -\delta x_k \sin(\theta_{k-1|k-1}) - \delta y_k \cos(\theta_{k-1|k-1}) \\ 0 & 1 & \delta x_k \cos(\theta_{k-1|k-1}) - \delta y_k \sin(\theta_{k-1|k-1}) \\ 0 & 0 & 1 \end{bmatrix} \tag{B.18}$$

$$\mathbf{J}_2 = \begin{bmatrix} \cos(\theta_{k-1|k-1}) & -\sin(\theta_{k-1|k-1}) & 0 \\ \sin(\theta_{k-1|k-1}) & \cos(\theta_{k-1|k-1}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{B.19}$$

The above equations state closed form expressions which are applicable when the two wheels of the robot are turning with different velocities and the robot is following a circular arc with constant radius of curvature. However, if the robot is moving along a straight line or is rotating about its center of mass, the above expressions can be further simplified as explained in the next sub-section.

B.4 Special Cases

The robot follows a curved path when its wheels are moving at different velocities. Its trajectory approaches a straight line when the two wheels start turning at the same velocities and rotates about its center of mass when the two velocities are same in magnitude but opposite in direction.

B.4.1 Motion Along a Straight Line

Robot trajectory approaches a straight line when $v_{rk} - v_{lk} \rightarrow 0$. Application of this condition to (B.1) and (B.3) results in $\alpha_k \rightarrow 0$ and $c_k \rightarrow \infty$, respectively. The control vector (B.5) becomes:

$$\mathbf{u_k} = \begin{bmatrix} \delta x_k \\ \delta y_k \\ \delta\theta_k \end{bmatrix} = \begin{bmatrix} v_k \\ 0 \\ 0 \end{bmatrix} \tag{B.20}$$

Using the new control vector prediction estimate (B.17) takes the following form:

$$\widehat{\mathbf{p}}_{k|k-1} = \begin{bmatrix} x_{k|k-1} \\ y_{k|k-1} \\ \theta_{k|k-1} \end{bmatrix} = \begin{bmatrix} x_{k-1|k-1} + v_k cos(\theta_{k-1|k-1}) \\ y_{k-1|k-1} + \delta x_k sin(\theta_{k-1|k-1}) \\ \theta_{k-1|k-1} + \alpha_k \end{bmatrix} \tag{B.21}$$

It is assumed that $sin(\alpha_k) \approx \alpha_k$ and $cos(\alpha_k) \approx 1$ for small values of α_k. Using these approximations the jacobian $\mathbf{J}_u$ and covariance matrix $\mathbf{U}_k$ of (B.8) and (B.10) are simplified as follows:

$$\mathbf{J}_u = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{-v_k}{w} & \frac{v_k}{w} \\ \frac{-1}{w} & \frac{1}{w} \end{bmatrix} \tag{B.22}$$

$$\mathbf{U}_k = \begin{bmatrix} \frac{\sigma_l^2+\sigma_r^2}{4} & \frac{v_k(\sigma_r^2-\sigma_l^2)}{2w} & \frac{\sigma_r^2-\sigma_l^2}{2w} \\ \frac{v_k(\sigma_r^2-\sigma_l^2)}{2w} & \frac{v_k^2(\sigma_l^2+\sigma_r^2)}{w^2} & \frac{v_k(\sigma_l^2+\sigma_r^2)}{w^2} \\ \frac{\sigma_r^2-\sigma_l^2}{2w} & \frac{v_k(\sigma_l^2+\sigma_r^2)}{w^2} & \frac{\sigma_l^2+\sigma_r^2}{w^2} \end{bmatrix} \tag{B.23}$$

The above equations are using the fact that for straight line motion $v_k \approx v_{lk} \approx v_{rk}$. As revealed by (B.23) uncertainty in direction perpendicular to the direction of motion is proportional to the cube of distance traveled.

B.4.2 Rotation Around Center of Mass

The robot starts rotating around its center of mass when the two velocities are equal in magnitude but opposite in direction. Suppose $v_{rk} \to -v_{lk}$ or $v_{rk} + v_{lk} \to 0$, in such a case (B.1) and (B.3) result in $\alpha_k \to \frac{2v_{rk}}{w}$ and $c_k \to 0$, respectively. The control vector of (B.5) becomes:

$$\mathbf{u_k} = \begin{bmatrix} \delta x_k \\ \delta y_k \\ \delta\theta_k \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \frac{2v_{rk}}{w} \end{bmatrix} \tag{B.24}$$

whereas, the prediction (B.17) takes the following form:

$$\widehat{\mathbf{p}}_{k|k-1} = \begin{bmatrix} x_{k|k-1} \\ y_{k|k-1} \\ \theta_{k|k-1} \end{bmatrix} = \begin{bmatrix} x_{k-1|k-1} \\ y_{k-1|k-1} \\ \theta_{k-1|k-1} + \frac{2v_{rk}}{w} \end{bmatrix} \tag{B.25}$$

Substituting $-v_{rk}$ for v_{lk}, $\mathbf{J}_u$ and $\mathbf{U}_k$ reduces to the following:

$$\mathbf{J}_u = \begin{bmatrix} \frac{w\sin\alpha_k}{4v_{rk}} & \frac{w\sin\alpha_k}{4v_{rk}} \\ \frac{w(1-\cos\alpha_k)}{4v_{rk}} & \frac{w(1-\cos\alpha_k)}{4v_{rk}} \\ \frac{-1}{w} & \frac{1}{w} \end{bmatrix} \tag{B.26}$$

$$\mathbf{U}_k = \begin{bmatrix} \frac{w^2(\sigma_l^2+\sigma_r^2)\sin^2\alpha_k}{16v_{rk}^2} & \frac{w^2(\sigma_l^2+\sigma_r^2)\sin\alpha_k(1-\cos\alpha_k)}{16v_{rk}^2} & \frac{(\sigma_r^2-\sigma_l^2)\sin\alpha_k}{4v_{rk}} \\ \frac{w^2(\sigma_l^2+\sigma_r^2)\sin\alpha_k(1-\cos\alpha_k)}{16v_{rk}^2} & \frac{w^2(\sigma_l^2+\sigma_r^2)(1-\cos\alpha_k)^2}{16v_{rk}^2} & \frac{(\sigma_r^2-\sigma_l^2)(1-\cos\alpha_k)}{4v_{rk}} \\ \frac{(\sigma_r^2-\sigma_l^2)\sin\alpha_k}{4v_{rk}} & \frac{(\sigma_r^2-\sigma_l^2)(1-\cos\alpha_k)}{4v_{rk}} & \frac{\sigma_l^2+\sigma_r^2}{w^2} \end{bmatrix} \tag{B.27}$$

Bibliography

[AA92] AKHTAR, M. W. ; ATIQUZZAMAN, M.: Determination of Line Length Using Hough Transform. In: *Electronics Letters* 28 (1992), January, Nr. 1, S. 94–96

[AA94] ATIQUZZAMAN, M. ; AKHTAR, M. W.: Complete Line Segment Description Using the Hough Transform. In: *Image and Vision Computing* 12 (1994), June, Nr. 5, S. 267–273

[ACM99] ADORNI, G. ; CAGNONI, S. ; MORDONINI, M.: Landmark-Based Robot Self-Localization: a Case Study for the RoboCup Goal-Keeper. In: *Proceedings of the International Conference on Information Intelligence and Systems.* Bethesda, MD , USA, October 1999, S. 164–171

[ACSS03] ARRAS, K. O. ; CASTELLANOS, J. A. ; SCHILT, M. ; SIEGWART, R.: Feature-Based Multi-Hypothesis Localization and Tracking Using Geometric Constraints. In: *Robotics and Autonomous Systems* 44 (2003), S. 41 – 53

[Ågr03] ÅGREN, E.: *Lateral Position Detection Using a Vehicle-Mounted Camera.* Linköping, Sweden, Linköpings Universitet, Diplomarbeit, 2003

[AH93] ATIYA, S. ; HAGER, G.D.: Real-Time Vision-Based Robot Localization. In: *IEEE Transactions on Robotics and Automation* 9 (1993), Nr. 6, S. 785–800

[AHC05] AIDER, O. A. ; HOPPENOT, P. ; COLLE, E.: A Model-Based Method for Indoor Mobile Robot Localization Using Monocular Vision and Straight-Line Correspondences. In: *Robotics and Autonomous Systems* 52 (2005), August, Nr. 2-3, S. 229–246

[AR98] ARSENIO, A. ; RIBEIRO, M.I.: Absolute Localization of Mobile Robots Using Natural Landmarks. In: *Proceedings IEEE International Conference on Electronics, Circuits and Systems* Bd. 2, 1998, S. 483 – 486

[AT99] ARRAS, K.O. ; TOMATIS, N.: Improving Robustness and Precision in Mobile Robot Localization by Using Laser Range Finding and Monocular Vision. In: *Proceedings of the European Workshop on Advanced Mobile Robots.* Zurich, Switzerland, September 1999

[ATJS01] ARRAS, K.O. ; TOMATIS, N. ; JENSEN, B. ; SIEGWART, R.: Multisensor On-the-Fly Localization: Precision and Reliability for Applications. In: *Robotics and Autonomous Systems* 34 (2001), February, Nr. 2-3, S. 131–143

[AVT+00] Asada, M. ; Veloso, M.M. ; Tambe, M. ; Noda, I. ; Kitano, H. ; Kraetzschmar, G.K.: Overview of RoboCup-98. In: *Artificial Intelligence Magazine* 21 (2000), Nr. 1, S. 9–19

[BA83] Burt, P.J. ; Adelson, E.: The Laplacian Pyramid as a Compact Image Code. In: *IEEE Transactions on Communications* 9 (1983), Nr. 4, S. 532–540

[Bad07] Bader, M.: *Feature-Based Real-Time Stereo Vision on a Dual Core DSP with an Object Detection Algorithm*. Vienna, Austria, Pattern Recongnition and Impage Processing Group, Institute of Computer Aided Automation, Vienna University of Technology, Diplomarbeit, March 2007

[BAK04] Bekris, K.E. ; Argyros, A. A. ; Kavraki, L. E.: Angle-Based Methods for Mobile Robot Navigation: Reaching the Entire Plane. In: *Proceedings of the IEEE International Conference on Robotics and Automation*, 2004, S. 2373 – 2378

[Bal81] Ballard, D. H.: Generalizing the Hough Transform to Detect Arbitrary Shapes. In: *Pattern Recognition* 13 (1981), Nr. 2, S. 111–122

[BDN07] Bais, A. ; Deutsch, T. ; Novak, G.: Comparison of Self-Localization Methods for Soccer Robots. In: *Proceedings of the IEEE International Conference on Industrial Informatics*, 2007

[BDW93] B.Barshan ; Durrant-Whyte, H.F.: An Inertial Navigation System for a Mobile Robot. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. Yokohama, Japan, July 1993, S. 2243–2248

[BEF96] Borenstein, J. ; Everett, H. R. ; Feng, L.: *Navigating Mobile Robots: Systems and Techniques*. A. K. Peters, Ltd., 1996. – ISBN 1–5688–1058–0

[BF82] Barnard, S.T. ; Fischler, M.A.: Computational Stereo. In: *Surveys* 14 (1982), December, Nr. 4, S. 553–572

[BF96] Borenstein, J. ; Feng, L.: Measurement and Correction of Systematic Odometry Errors in Mobile Robots. In: *IEEE Transactions on Robotics and Automation* 12 (1996), December, Nr. 6, S. 869 – 880

[BFHS96] Burgard, W. ; Fox, D. ; Hennig, D. ; Schmidt, T.: Estimating the Absolute Position of a Mobile Robot Using Position Probability Grids. In: *Proceedings of the National Conference on Artificial Intelligence* Bd. 2, 1996, S. 896–901

[BG97] Betke, M. ; Gurvits, L.: Mobile Robot Localization Using Landmarks. In: *IEEE Transactions on Robotics and Automation* 13 (1997), April, Nr. 2, S. 251 – 263

[BHR86] Burns, J. B. ; Hanson, A. R. ; Riseman, E. M.: Extracting Straight Lines. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8 (1986), Nr. 4, S. 425–455

[BKHS00] Bandlow, T. ; Klupsch, M. ; Hanek, R. ; Schmitt, Thorsten: Fast Image Segmentation, Object Recognition and Localization in a RoboCup Scenario. In: *RoboCup 1999: Robot Soccer World Cup III*, 2000, S. 174–185

[Bor87] Borenstein, J.: *The Nursing Robot System*, Technion, Haifa, Israel, Diss., June 1987

[Bor98] Borenstein, J.: Experimental Results from Internal Odometry Error Correction with the OmniMate Mobile Robot. In: *IEEE Transactions on Robotics and Automation* 14 (1998), December, Nr. 6, S. 963 – 969

[BP01] Barrett, W. A. ; Petersen, K. D.: Houghing the Hough: Peak Collection for Detection of Corners, Junctions and Line Intersections. In: *Proceedings of International Conference on Computer Vision and Pattern Recognition* Bd. 2, 2001, S. 302–309

[Bra93] Braunegg, D. J.: MARVEL: A System That Recognizes World Locations with Stereo Vision. In: *IEEE Transactions on Robotics and Automation* 9 (1993), June, Nr. 3, S. 303–308

[Bra96] Braendle, N.: Rektifizierung oder direkte Tiefenberechnung? Untersuchungen zur digitalen Objektvermessung mit dem Stereoverfahren / PRIP, TU Wien. 1996 (PRIP-TR-043). – Forschungsbericht

[BS87] Bar-Shalom, Y.: *Tracking and Data association*. San Diego, CA, USA : Academic Press Professional, Inc., 1987. – ISBN 0–120–79760–7

[BS06] Bais, A. ; Sablatnig, R.: Landmark Based Global Self-Localization of Mobile Soccer Robots. In: Narayanan, P. J. (Hrsg.) ; Nayar, S. K. (Hrsg.) ; Shum, H-Yeung (Hrsg.): *Proceedings of the Asian Conference on Computer Vision* Bd. 3852. Hyderabad, India : Springer-Verlag GmbH, January 2006, S. 842 – 851

[BSB$^+$00] Briggs, A. J. ; Scharstein, D. ; Braziunas, D. ; Dima, C. ; Wall, P.: Mobile Robot Navigation Using Self-Similar Landmarks. In: *Proceedings of the IEEE International Conference on Robotics and Automation*, 2000, S. 1428–1434

[BSG06] Bais, A. ; Sablatnig, R. ; Gu, J.: Single Landmark Based Self-Localization of Mobile Robots. In: *Proceedings of the Canadian Conference on Computer and Robot Vision*. Quebec City, Canada : IEEE Computer Society Press, June 2006, S. 67

[BSGK07] Bais, A. ; Sablatnig, R. ; Gu, J. ; Khawaja, Y. M.: Location Tracker for a Mobile Robot. In: *Proceedings of the IEEE International Conference on Industrial Informatics*, 2007

[BSGM06] Bais, A. ; Sablatnig, R. ; Gu, J. ; Mahlknecht, S.: Active Single Landmark Based Global Localization of Autonomous Mobile Robots. In: et al., G. B. (Hrsg.): *Proceedings of the International Conference on Visual Computing*. Lake Tahoe, Nevada, USA : Springer-Verlag Berlin Heidelberg, November 2006 (Lecture Notes in Computer Science 4291), S. 202 – 211

[BSKN07] Bais, A. ; Sablatnig, R. ; Khawaja, Y. M. ; Novak, G.: Propagation of Uncertainty in Landmark Based Self-Localization of Autonomous Mobile Robots. In: *Proceedings of IAPR Conference on Machine Vision Applications*, 2007

[BSN05] Bais, A. ; Sablatnig, R. ; Novak, G.: Line-Based Landmark Recognition for Self-Localization of Soccer Robots. In: *Proceedings of the IEEE International Conference on Emerging Technologies*. Islamabad, Pakistan, September 2005, S. 132–137

[BSW00] Buschka, P. ; Saffiotti, A. ; Wasik, Z.: Fuzzy Landmark-Based Localization for a Legged Robot. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2000, S. 1205 – 1210

[Bur81] Burt, P.J.: Fast Filter Transforms for Image Processing. In: *Computer Vision, Graphics, and Image Processing* 16 (1981), S. 20–51

[BV03] Bruce, J. ; Veloso, M.: Fast and Accurate Vision-Based Pattern Detection and Identification. In: *Proceedings of the IEEE International Conference on Robotics and Automation* Bd. 1, 2003, S. 1277 – 1282

[BWN04] Ball, D. ; Wyeth, G. ; Nuske, S.: A Global Vision System for a Robot Soccer Team. In: *Proceedings of the Australian Conference on Robotics and Automation*, 2004

[Cas86] Case, M.: Single Landmark Navigation by Mobile Robot. In: *Proceedings of the SPIE Conference on Mobile Robots* Bd. 727. Cambridge, MA, October 1986, S. 231–237

[CB03] Climer, S. ; Bhatia, S. K.: Local Lines: A Linear Time Line Detector. In: *Pattern Recognition Letters* 24 (2003), S. 2291–2300

[CC92] Chenavier, F. ; Crowley, J. L.: Position Estimation for a Mobile Robot Using Vision and Odometry. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. Nice, France, May 1992, S. 2588–2593

[CDPB00] Clerentin, A. ; Delahoche, L. ; Pegard, C. ; Brassart, E.: A Localization Method Based on Two Omnidirectional Perception Systems cooperation. In: *Proceedings of the IEEE International Conference on Robotics and Automation*, 2000, S. 1219–1224

[CG99] Chutatape, O. ; Guo, L.: A Modified Hough Transform for Line Detection and its Performance. In: *Pattern Recognition* 32 (1999), S. 181 – 192

[CK97] Chong, K. S. ; Kleeman, L.: Accurate Odometry and Error Modelling for a Mobile Robot. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. Albuquerque, New Mexico, April 1997, S. 2783 – 2788

[CKK96] Cassandra, A. R. ; Kaelbling, L. P. ; Kurien, J. A.: Acting Under Uncertainty: Discrete Bayesian Models for Mobile Robot Navigation. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems* Bd. 2. Osaka Japan, November 1996, S. 963–972

[CKKK94] Christensen, H. I. ; Kirkeby, N. O. ; Kristensen, S. ; Knudsen, L.: Model-Driven Vision for Indoor Navigation. In: *Robotics and Autonomous Systems* 12 (1994), April, Nr. 3-4, S. 199–207

[CKP95] Chang, H. D. ; Kim, K. I. ; Poston, T.: An Accurate 3D Localization of a Camera Using a Guide-Mark. In: *Pattern Recognition Letters* 16 (1995), July, Nr. 7, S. 749–757

[CKS+00] Coradeschi, S. ; Karlsson, L. ; Stone, P. ; Balch, T. ; Kraetzschmar, G.K. ; Asada, M.: Overview of RoboCup-99. In: *Artificial Intelligence Magazine* 21 (2000), Nr. 3, S. 11–18

[CL94] Cox, I. J. ; Leonard, J. J.: Modeling a Dynamic Environment Using a Bayesian Multiple Hypothesis Approach. In: *Artificial Intelligence* 66 (1994), Nr. 2, S. 311–344

[COB01] Chung, H. ; Ojeda, L. ; Borenstein, J.: Accurate Mobile Robot Dead-Reckoning with a Precision-Calibrated Fiber-Optic Gyroscope. In: *IEEE Transactions on Robotics and Automation* 17 (2001), February, Nr. 1, S. 80 – 84

[Cox91] Cox., I. J.: Blanche — An Experiment in Guidance and Navigation of an Autonomous Robot Vehicle. In: *IEEE Transactions on Robotics and Automation* 7 (1991), Nr. 2, S. 193–204

[CRK99] Choi, W. ; Ryu, C. ; Kim, H.: Navigation of a Mobile Robot Using mono-Vision and mono-audition. In: *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics* Bd. 4, 1999, S. 686–691

[Cro89] Crowley, J. L.: World Modeling and Position Estimation for a Mobile Robot Using Ultrasonic Ranging. In: *Proceedings of the IEEE International Conference on Robotics and Automation.* Scottsdale, AZ, May 1989, S. 674–680

[Cro95] Crowley, J. L.: Mathematical Foundations of Navigation and Perception for an Autonomous Mobile Robot. In: *Proceedings of the International Workshop on Reasoning with Uncertainty in Robotics*, 1995, S. 9–51

[CS87] Crowley, J.L. ; Sanderson, A.C.: Multiple Resolution Representation and Probabilistic Matching of 2-D gray-scale Shape. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 9 (1987), Nr. 1, S. 113–121

[Dav87] Davies, E. R.: A New parametrisation of the Straight Line and its Application for the Optimal Detection of objects with Straight Edges. In: *Pattern Recognition Letters* 6 (1987), June, Nr. 1, S. 9–14

[Dav88] Davies, E. R.: Application of the Generalised Hough Transform to Corner Detection. In: *Computers and Digital Techniques, IEE Proceedings-E* 135 (1988), January, Nr. 1, S. 49 – 54

[Dav92] Davies, E. R.: Simple Two-stage Method for the Accurate Location of Hough Transform Peaks. In: *Computers and Digital Techniques, IEE Proceedings-E* 139 (1992), May, Nr. 3, S. 242 – 248

[DBBD03] Douret, J. ; Benosman, R. ; Bouzar, S. ; Devars, J.: Localization of Robots in F180 League Using Projective Geometry. In: *RoboCup 2002: Robot Soccer World Cup VI* Bd. 2752, 2003, S. 312–318

[DBFT99] Dellaert, F. ; Burgard, W. ; Fox, D. ; Thrun, S.: Using the condensation Algorithm for Robust, Vision-Based Mobile Robot Localization. In: *Proceedings of International Conference on Computer Vision and Pattern Recognition* Bd. 2. Fort Collins, Colorado, June 1999, S. 2588–2594

[Deu07] Deutsch, T.: *Geometric World ModelRepository and Localization for Autonomous Mobile Robots*. Vienna, Austria, Institute of Computer Technology, Vienna University of Technology, Diplomarbeit, March 2007

[DFBT99] Dellaert, F. ; Fox, D. ; Burgard, W. ; Thrun, S.: Monte Carlo Localization for Mobile Robots(ICRA' 99). In: *Proceedings of the IEEE International Conference on Robotics and Automation*, 1999

[DH72] Duda, R. ; Hart, P.: Use of the Hough Transformation to Detect Lines and Curves in the Pictures. In: *Communications of the Association for Computing Machinery* 15 (1972), Nr. 1, S. 11–15

[DJ97] Dulimarta, H. S. ; Jain, A. K.: Mobile Robot Localization in Indoor Environment. In: *Pattern Recognition* 30 (1997), Nr. 1, S. 99–111

[DK02] DeSouza, G. N. ; Kak, A. C.: Vision for Mobile Robot Navigation: A Survey. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24 (2002), February, Nr. 2, S. 237–267

[DYOC04] Dao, N.X. ; You, B.-J. ; Oh, S.-R. ; Choi, Y.J.: Simple Visual Self-Localization for Indoor Mobile Robots Using Single Video Camera. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems* Bd. 4, 2004, S. 3767–3772

[DYOH03] Dao, N.X. ; You, B.-J. ; Oh, S.-R. ; Hwangbo, M.: Visual Self-Localization for Indoor Mobile Robots Using Natural Lines. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems* Bd. 2, 2003, S. 1252– 1257

[EM92] Engelson, S. ; McDermott, D.: Error Correction in Mobile Robot Map Learning. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. Nice, France, May 1992, S. 2555–2560

[Ent05] Entner, H.: *Real-Time 3D Reconstruction for Autonomous Football Playing Robots Using a Feature Based Stereo Approach*. Vienna, Austria, Vienna University of Technology, Diplomarbeit, January 2005

[ERF$^+$00] Enderle, S. ; Ritter, M. ; Fox, D. ; Sablatnög, S. ; Kraetzschmar, G. ; Palm, G.: Soccer Robot Localization Using Sporadic Visual Features. In: et al., E. P. (Hrsg.): *Proceedings of the International Conference on Intelligent Autonomous Systems*, 2000, S. 959–966

[FBDT99a] Fox, D. ; Burgard, W. ; Dellaert, F. ; Thrun, S.: Markov Localization for Mobile Robots in Dynamic Environments. In: *Artificial Intelligence* 11 (1999), S. 391–427

[FBDT99b] Fox, D. ; Burgard, W. ; Dellaert, F. ; Thrun, S.: Monte Carlo Localization: Efficient Position Estimation for Mobile Robots. In: *Proceedings of the National Conference on Artificial Intelligence*, 1999

[FBKT99] Fox, D. ; Burgard, W. ; Kruppa, H. ; Thrun, S.: A Monte Carlo Algorithm for Multi-Robot Localization / School of Computer Science, Carnegie Mellon University. Pittsburgh, PA 15213, March 1999 (CMU-CS-99-120). – Forschungsbericht

[FBKT00] Fox, D. ; Burgard, W. ; Kruppa, H. ; Thrun, S.: A Probabilistic Approach to Collaborative Multi-Robot Localization. In: *Autonomous Robots* 8 (2000), Nr. 3, S. 325–344

[FBT98] Fox, D. ; Burgard, W. ; Thrun, S.: Active Markov Localization for Mobile Robots. In: *Robotics and Autonomous Systems* 25 (1998), November, Nr. 3-4, S. 195–207

[FBTC98] Fox, D. ; Burgard, W. ; Thrun, S. ; Cremers, A.B.: Position Estimation for Mobile Robots in Dynamic Environments. In: *Proceedings of the National Conference on Artificial Intelligence*, 1998, S. 983–988

[FFK92] Feng, L. ; Fainman, Y. ; Koren, Y.: Estimation of the Absolute Position of Mobile Systems by an Optoelectronic Processor. In: *IEEE Transactions on Systems, Man, and Cybernetics* 22 (1992), September/October, Nr. 5, S. 953–963

[FG03] Fichtner, M. ; Grossmann, A.: A Visual-Sensor Model for Mobile Robot Localisation / Artificial Intelligence Institute, Department of Computer Science, Technische Universitaet Dresden. Dresden, Germany, 2003 (WV-03-03/CL-2003-02). – Forschungsbericht

[FHR$^+$90] Fennema, C. ; Hanson, A. ; Riseman, E. ; Beveridge, J. ; Kumar, R.: Model-Directed Mobile Robot Navigation. In: *IEEE Transactions on Systems, Man, and Cybernetics* 20 (1990), Nr. 6, S. 1352–1369

[FLW95] Forsberg, J. ; Larsson, U. ; Wernersson, A.: Mobile Robot Navigation Using the Range-Weighted Hough Transform. In: *IEEE Robotics and Automation Magazine* (1995), March, S. 18–26

[Fox98] Fox, D.: *Markov Localization: A Probabilistic Framework for Mobile Robot Localization and Navigation*, Institute of Computer Science III, University of Bonn, Germany, Diss., December 1998

[Fox03] Fox, D.: Adapting the Sample Size in Particle Filters Through KLD-Sampling. In: *The International Journal of Robotics Research* 22 (2003), Nr. 12, S. 985–1003

[FTBD00] Fox, D. ; Thrun, S. ; Burgard, W. ; Dellaert, F.: Particle Filters for Mobile Robot Localization. In: Doucet, Arnaud (Hrsg.) ; de Freitas, Nando (Hrsg.) ; Gordon, Neil (Hrsg.): *Sequential Monte Carlo Methods in Practice.* New York : Springer, 2000, S. 2582–2587

[GBFK98] Gutmann, J. S. ; Burgard, W. ; Fox, D. ; Konolige, K.: An Experimental Comparison of Localization Methods. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1998

[GDCT04] Giremus, A. ; Doucet, A. ; Calmettes, V. ; Tourneret, J.-Y.: A Rao-Blackwellized Particle Filter for INS/GPS Integration. In: *Proceedings of the IEEE International Conference on Acoustics,Speech, and Signal Processing* Bd. 3, 2004, S. 964–967

[GF02] Gutmann, J.S. ; Fox, D.: An Experimental Comparison of Localization Methods Continued. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2002

[GHH+99] Gutmann, J. S. ; Hatzack, W. ; Herrmann, I. ; Nebel, B. ; Rittinger, F. ; Topor, A. ; Weigel, T. ; Welsch, B.: The CS Freiburg Robotic Soccer Team: Reliable Self-Localization, Multirobot Sensor Integration, and Basic Soccer Skills. In: Asada, M. (Hrsg.): *RoboCup 1998: Robot Soccer World Cup II*. Berlin, Heidelberg, New York : Springer-Verlag, 1999

[GS96] Gutmann, J.S. ; Schlegel, C.: AMOS: Comparison of Scan Matching Approaches for Self-Localization in Indoor Environments. In: *Proceedings of the Euromicro Workshop on Advanced Mobile Robots*, IEEE Computer Society Press, 1996

[GSN04] Guru, D. S. ; Shekar, B. H. ; Nagabhushan, P.: A Simple and Robust Line Detection Algorithm Based on Small Eigenvalue Analysis. In: *Pattern Recognition Letters* 25 (2004), Nr. 1, S. 1–13. – ISSN 0167–8655

[GSO92] Gonzalez, J. ; Stentz, A. ; Ollero, A.: An Iconic Position Estimator for a 2D Laser Range Finder. In: *Proceedings of the IEEE International Conference on Robotics and Automation* Bd. 3. Nice, France, May 1992, S. 2646 – 2651

[GTN99] Gutmann, J.S. ; T.Weigel ; Nebel, B.: Fast, Accurate, and Robust Self-Localization in Polygonal Environments. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. Kyongju, Korea, October 1999

[GW02] Gonzalez, R. C. ; Woods, R. E.: *Digital Image Processing*. 2nd. Prentice Hall, January 2002. – ISBN 0201180758

[GWE03] Gonzalez, R. C. ; Woods, R. E. ; Eddins, S. L.: *Digital Image Processing Using MATLAB*. 1st. Prentice Hall, September 2003. – ISBN 0130085197

[GWN01] Gutmann, J.S. ; Weigel, T. ; Nebel, B.: A Fast, Accurate, and Robust Method for Self-Localization in Polygonal Environments Using Laser-Range-Finders. In: *Advanced Robotics Journal* 14 (2001), Nr. 8, S. 651–668

[HE85] H.P.Moravec ; Elfes, A.: High Resolution Maps from Wide Angle Sonar. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. Washington, D.C., 1985, S. 116–121

[HK96] Hertzberg, J. ; Kirchner, F.: Landmark-Based Autonomous Navigation in Sewerage Pipes. In: *Proceedings of the Euromicro Workshop on Advanced Mobile Robots*, 1996, S. 68–73

[HK99] Howard, A. ; Kitchen, L.: Navigation Using Natural Landmarks. In: *Robotics and Autonomous Systems* 26 (1999), S. 99 – 115

[HKYR97] Hager, G. ; Kriegman, D. ; Yeh, E. ; Rasmussen, C.: Image-Based Prediction of Landmark Features for Mobile Robot Navigation. In: *Proceedings of the IEEE International Conference on Robotics and Automation*, 1997, S. 1040–1046

[Hou62] Hough, P. V. C. *Method and Means for Recognizing Complex Patterns.* 1962

[HPMBS05a] Herrero-Pérez, D. ; Martínez-Barberá, H. ; Saffiotti, A.: Fuzzy Self-Localization Using Natural Features in the Four-Legged League. In: et al., D. N. (Hrsg.): *RoboCup 2004: Robot Soccer World Cup VIII*, Springer-Verlag, 2005 (Lecture Notes in Computer Science), S. 110 – 121

[HPMBS05b] Herrero-Pérez, D. ; Martínez-Barberá, H. ; Saffiotti, A.: Fuzzy Self-Localization Using Natural Features in the Four-Legged League. In: *RoboCup 2004: Robot Soccer World Cup VIII*, 2005 (LNCS), S. 110 – 121

[HS93] Haralick, R. M. ; Shapiro, L. G.: *Computer and Robot Vision.* Bd. II. Addison Wesley, 1993

[HS00] Hanek, R. ; Schmitt, T.: Vision-Based Localization and Data Fusion in a System of Cooperating Mobile Robots. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2000, S. 1199–1204

[HTMA03] Hernandez, S. ; Torres, J.M. ; Morales, C.A. ; Acosta, L.: A New Low Cost System for Autonomous Robot Heading and Position Localization in a Closed Area. In: *Autonomous Robots* 15 (2003), September, Nr. 2, S. 99–110

[Hue04] Huebner, K.: A Symmetry Operator and Its Application to the RoboCup. In: et al., D. P. (Hrsg.): *RoboCup 2003: Robot Soccer World Cup VII* Bd. 3020, 2004, S. 274 – 283

[Imm98] Immerkær, J.: Some Remarks on the Straight Line Hough Transform. In: *Pattern Recognition Letters* 19 (1998), October, Nr. 12, S. 1133–1135

[IN00] Iocchi, L. ; Nardi, D.: Self-Localization in the RoboCup Environment. In: Veloso, Manuela (Hrsg.) ; Pagello, Enrico (Hrsg.) ; Kitano, Hiroaki (Hrsg.): *RoboCup 1999: Robot Soccer World Cup III.* Berlin : Springer-Verlag, 2000 (Lecture Notes in Artificial Intelligence 1856), S. 318–330

[JAWA00] Jensfelt, P. ; Austin, D. J. ; Wijk, O. ; Andersson, M.: Feature Based Condensation for Mobile Robot Localization. In: *Proceedings of the IEEE International Conference on Robotics and Automation.* San Francisco, CA, April 2000, S. 2531–2537

[JC01] Jensfelt, P. ; Christensen, H. I.: Pose Tracking Using Laser Scanning and Minimalistic Environmental Models. In: *IEEE Transactions on Robotics and Automation* 17 (2001), April, Nr. 2, S. 138–147

[JCBJ02] de Jong, F. ; Caarls, J. ; Bartelds, R. ; Jonker, P.: A Two-Tiered Approach to Self-Localization. In: *RoboCup 2001: Robot Soccer World Cup V*, Springer-Verlag, 2002 (Lecture Notes in Artificial Intelligence), S. 405–410

[JIPB03] Ji, J. ; Indiveri, G. ; Ploeger, P. ; Bredenfeld, A.: An Omni-Vision Based Self-Localization Method for Soccer Robot. In: *Proceedings of the IEEE Symposium on Intelligent Vehicles.* Columbus, Ohio, USA, June 2003

[JK01] Jensfelt, P. ; Kristensen, S.: Active Global Localization for a Mobile Robot Using Multiple Hypothesis Tracking. In: *IEEE Transactions on Robotics and Automation* 17 (2001), October, Nr. 5, S. 748–760

[JKLK02] Jang, G. ; Kim, S. ; Lee, W. ; Kweon, I.: Color Landmark Based Self-Localization for Indoor Mobile Robots. In: *Proceedings of the IEEE International Conference on Robotics and Automation*, 2002, S. 1037 – 1042

[JML99] Jennings, C. ; Murray, D. ; Little, J. J.: Cooperative Robot Localization with Vision-Based Mapping. In: *Proceedings of the IEEE International Conference on Robotics and Automation*, 1999, S. 2659 – 2665

[JR94] Jolion, J. ; Rosenfeld, A.: *A Pyramid Framework for Early Vision.* Kluwer, 1994

[JU97] Julier, S. J. ; Uhlmann, J. K.: A New Extension of the Kalman Filter to Non-Linear Systems. In: *Proceedings of the International Symposium on Aerospace Defence Sensing, Simulation and Controls*, 1997

[KA87] Kabuka, M. R. ; Arenas, A. E.: Position Verification of a Mobile Robot Using Standard Pattern. In: *IEEE Journal of Robotics and Automation* RA-3 (1987), December, Nr. 6, S. 505–516

[KALAC90] Kak, A. ; Andress, K. ; L.-Abadia, C. ; Carroll, M.: Hierarchical Evidence Accumulation in the PSEIKI System and Experiments in Model-driven Mobile Robot Navigation. In: *Proceedings of the Conference on Uncertainty in Artificial Intelligence* Bd. 5, Elsevier Science Publishers B.V., North-Holland, 1990, S. 353–369

[KBS75] Kimme, C. ; Ballard, D. ; Sklansky, J.: Finding Circles by an Array of Accumulators. In: *Communications of the Association for Computing Machinery* 18 (1975), Nr. 2, S. 120–122. – ISSN 0001–0782

[KDB^{+}06] Krywult, S. ; Deutsch, T. ; Bader, M. ; Novak, G. ; Onrubia, A. G.: Autonomous MiroSot - the Autonomous Way of Playing MiroSot. In: *Proceedings of the FIRA RoboWorld Congress*, 2006, S. 163167

[KE02] Kraetzschmar, G. K. ; Enderle, Stefan: Self-Localization Using Sporadic Features. In: *Robotics and Autonomous Systems* 40 (2002), S. 111 – 119

[KEB91] KIRYATI, N. ; ELDAR, Y. ; BRUCKSTEIN, A.M.: A Probabilistic Hough Transform. In: *Pattern Recognition* 24 (1991), Nr. 4, S. 303 – 316

[Kel00] KELLY, A.: Pose Determination and Tracking in Image Mosaic-Based Vehicle Position Estimation. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. Takamatsu, Japan, October 2000

[KFM04] KWOK, C. ; FOX, D. ; MEIL, M.: Real-Time Particle Filters. In: *Proceedings of the IEEE* 92 (2004), March, Nr. 3, S. 469–484

[KH97] KÄLVIÄINEN, H. ; HIRVONEN, P.: An Extension to the Randomized Hough Transform Exploiting Connectivity. In: *Pattern Recognition Letters* 18 (1997), January, Nr. 1, S. 77 – 85

[KHXO95] KÄLVIÄINEN, H. ; HIRVONEN, P. ; XU, L. ; OJA, E.: Probabilistic and Non-Probabilistic Hough Transforms: Overview and Comparisons. In: *Image and Vision Computing* 13 (1995), May, Nr. 4, S. 239 – 252

[KK00] KYRKI, V. ; KÄLVIÄINEN, H.: Combination of Local and Global Line Extraction. In: *Real-Time Imaging* 6 (2000), S. 79–91

[KL92] KAY, M.G. ; LUO, R.C.: Camera Placement For Global Vision. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems* Bd. 2. Raleigh, NC, July 1992, S. 917 – 924

[KL93] KAY, M. ; LUO, R.: Global Vision for the Control of Free-Ranging AGV Systems. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. Atlanta, GA, May 1993, S. 14–19

[Kle92] KLEEMAN, L.: Optimal Estimation of Position and Heading for Mobile Robots Using Ultrasonic Beacons and Dead-Reckoning. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. Nice, France, May 1992, S. 2582–2587

[Kli71] KLINGER, A.: Pattern and Search Statistics. In: RUSTAGI, J.S. (Hrsg.): *Optimizing Methods in Statistics*. New York : Academic Press, 1971

[KO94] KOMORIYA, K. ; OYAMA, E.: Position Estimation of a Mobile Robot Using Optical Fiber Gyroscope (OFG). In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. Munich, Germany, September 1994, S. 143–149

[Kro89] KROTKOV, E.: Mobile Robot Localization Using a Single Image. In: *Proceedings of the IEEE International Conference on Robotics and Automation* Bd. 2, 1989, S. 978 – 983

[Kro91] KROPATSCH, Walter G.: Image Pyramids and Curves - An Overview / PRIP, TU Wien. 1991 (PRIP-TR-002). – Forschungsbericht

[KS98] KOENIG, S. ; SIMMONS, R. G.: Xavier: a Robot Navigation Architecture Based on Partially observable Markov decision process Models. (1998), S. 91–122. ISBN 0–262–61137–6

[KSG98] Kamat-Sadekar, V. ; Ganesan, S.: Complete Description of Multiple Line Segments Using the Hough Transform. In: *Image and Vision Computing* 16 (1998), July, Nr. 9-10, S. 597–613

[KTB89] Kriegman, J. ; Triendl, E. ; Binford, T.O.: Stereo Vision and Navigation in Buildings for Mobile Robots. In: *IEEE Transactions on Robotics and Automation* 5 (1989), December, Nr. 6, S. 792–803

[KY96] Kropatsch, W.G. ; Yacoub, S. B.: A Revision of Pyramid Segmentation. In: *Proceedings of IAPR International Conference on Pattern Recognition* Bd. B. Vienna, August 1996, S. 477–481

[LB87] Leavers, V. F. ; Boyce, J. F.: The Radon Transform and Its Application to Shape Parameterization in Machine Vision. In: *Image and Vision Computing* 5 (1987), May, Nr. 2, S. 161–166

[LDW91] Leonard, J.J. ; Durrant-Whyte, H.F.: Mobile Robot Localization by Tracking Geometric Beacons. In: *IEEE Transactions on Robotics and Automation* 7 (1991), Nr. 3, S. 376–382

[Lea93] Leavers, V. F.: Survey - Which Hough Transform? In: *Computer Vision, Graphics, and Image Processing* 58 (1993), S. 250 – 264

[LFW96] Larsson, U. ; Forsberg, J. ; Wemersson, Å.: Mobile Robot Localization: Integrating Measurements from a Time-of-Flight Laser. In: *IEEE Transactions on Industrial Electronics* 43 (1996), Nr. 3, S. 422 – 431

[LHF90] Liu, Y. ; Huang, T.S. ; Faugeras, O.D.: Determination of Camera Location from 2-D to 3-D Line and Point Correspondences. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 12 (1990), January, Nr. 1, S. 28–37

[Lin94a] Lindeberg, L.: *Scale-Space Theory in Computer Vision.* Boston/London/Dordrecht : Kluwer Academic Publishers, 1994

[Lin94b] Lindeberg, T.: Scale-Space Theory: A basic tool for analysing structures at different scales. In: *Journal of Applied Statistics* 21 (1994), Nr. 2, S. 224–270

[LK04] Libuda, L. ; Kraiss, K.-F.: Identification of Natural Landmarks for Vision Based Navigation. In: Drews, Prof. P. (Hrsg.): *Proceedings of the Conference on Mechatronics and Robotics* Bd. III. Aachen, Germany, September 2004, S. 877–882

[LL92] Lazanas, A. ; Latombe, J.-C.: Landmark-Based Robot Navigation. In: *Proceedings of the National Conference on Artificial Intelligence.* San Jose, California : AAAI Press, 1992, S. 816–822

[LN02] L.Iocchi ; Nardi, D.: Hough Localization for Mobile Robots in Polygonal Environments. In: *Robotics and Autonomous Systems* 40 (2002), July, Nr. 1, S. 43–58

[LNHS05] Lingemann, K. ; Nüchter, A. ; Hertzberg, J. ; Surmann, H.: High-speed Laser Localization for Mobile Robots. In: *Robotics and Autonomous Systems* 51 (2005), S. 275 – 296

[LNJS01] Lamon, P. ; Nourbakhsh, I. ; Jensenl, B. ; Siegwart, R.: Deriving and Matching Image fingerprint sequences for Mobile Robot Localization. In: *Proceedings of the IEEE International Conference on Robotics and Automation.* Seoul, Korea, May 2001, S. 1609–1614

[LSL$^+$03] Lee, J. M. ; Son, K. ; Lee, M. C. ; Choi, J. W. ; Han, S. H. ; Lee, M. H.: Localization of a Mobile Robot Using the Image of a Moving Object. In: *IEEE Transactions on Industrial Electronics* 50 (2003), June, Nr. 3, S. 612–619

[LST02] Li, X. J. ; So, A. T. P. ; Tso, S. K.: CAD-Vision-Range-Based Self-Localization for Mobile Robot Using One Landmark. In: *Journal of Intelligent and Robotic Systems* 35 (2002), S. 61–81

[LV00] Lenser, S. ; Veloso, M.: Sensor Resetting Localization for Poorly Modelled Mobile Robots. In: *Proceedings of the IEEE International Conference on Robotics and Automation*, 2000, S. 1225–1232

[MA98] McLaughlin, R. A. ; Alder, M. D.: The Hough Transform Versus the UpWrite. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20 (1998), Nr. 4, S. 396–400

[MAES99] Moreno, L. ; Armingol, J. M. ; de la Escalera, A. ; Salichs, M. A.: Global Integration of Ultrasonic Sensors Information in Mobile Robot Localization. In: *Proceedings of the International Conference on Advanced Robotics.* Tokyo, Japan, October 1999

[MAG$^+$02] Moreno, L. ; Armingol, J. M. ; Garrido, S. ; de la Escalera, A. ; Salichs, M. A.: A Genetic Algorithm for Mobile Robot Localization Using Ultrasonic Sensors. In: *Journal of Intelligent and Robotic Systems* 34 (2002), June, Nr. 2, S. 135 – 154

[May79] Maybeck, P. S.: *Mathematics in Science and Engineering.* Bd. 141: *Stochastic Models, Estimation, and control.* Academic Press, 1979. – ISBN 0124807038

[MBR87] Meer, P. ; Baugher, E. S. ; Rosenfeld, A.: Frequency Domain Analysis and Synthesis of Image Pyramid Generating Kernels. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 9 (1987), Nr. 4, S. 512–522

[ML01] Marques, C. F. ; Lima, P. U.: A Localization Method for a Soccer Robot Using a Vision-Based omni-directional sensor. In: et al., P. S. (Hrsg.): *RoboCup 2000: Robot Soccer World Cup IV*, 2001 (LNCS 2109), S. 96–107

[MMH04] Motomura, A. ; Matsuoka, T. ; Hasegawa, T.: Self-Localization Method Using Two Landmarks and Dead Reckoning for Autonomous Mobile Soccer Robots. In: *RoboCup 2003: Robot Soccer World Cup VII*, 2004 (LNCS), S. 526–533

[MPS01] Marando, F. ; Piaggio, M. ; Scalzo, A.: Real Time Self Localization Using a Single Frontal Camera. In: *Proceedings of the International Symposium on Intelligent Robotic Systems.* Toulouse, France, July 2001

[MS87] Matthies, L. ; Shafer, S.A.: Error Modeling in Stereo Navigation. In: *IEEE Transactions on Robotics and Automation* RA-3 (1987), June, Nr. 3, S. 239–248

[MS98] Menard, C. ; Sablatnig, R.: Adaptive Area-Based Stereo Matching. In: *Proceedings of the IS&T/SPIE Symposium on Three-Dimensional Image Capture and Applications* Bd. SPIE-Vol.3313, 1998, S. 14–24

[MT89] Marapane, S.B. ; Trivedi, M.M.: Region-Based Stereo Analysis for Robotic Applications. In: *IEEE Transactions on Systems, Man, and Cybernetics* 19 (1989), November/December, S. 1447–1464

[MZPI04] Menegatti, E. ; Zoccarato, M. ; Pagello, E. ; Ishiguro, H.: Image-Based Monte Carlo Localisation with Omnidirectional Images. In: *Robotics and Autonomous Systems* 48 (2004), Nr. 1, S. 17–30

[Nac95] Nacken, P.F.M.: Image Segmentation by Connectivity preserving Relinking in Hierarchical Graph Structures. In: *Pattern Recognition* 28 (1995), June, Nr. 6, S. 907–920

[NBM04] Novak, G. ; Bais, A. ; Mahlknecic, S.: Simple Stereo Vision System for Real-Time Object Recognition for an Autonomous Mobile Robot. In: *Proceedings of IEEE International Conference on Computational Cybernetics*, 2004, S. 213– 216

[NCB$^+$06] Novak, G. ; C.Roesener ; Bader, M. ; Deutsch, T. ; Jakubek, S. ; Krywult, S. ; M.Seyr: The Tinyphoon's Control Concept. In: *Proceedings of the IEEE International Conference on Mechatronics*, 2006, S. 625–630

[Neg03] Negenborn, R.: *Robot Localization and Kalman Filters: On Finding Your Position in a noisy World.* Utrecht, The Netherlands, Institute of Information and Computing Sciences, Utrecht University, Diplomarbeit, September 2003

[New05] Newman, P.: *An Introduction to Estimation and its Application to Mobile Robotics.* 2.0. Robotics Research Group, Department of Engineering Science, University of Oxford, October 2005. – Lecture notes

[NJW$^+$98] Nickerson, S. B. ; Jasiobedzki, P. ; Wilkes, D. ; Jenkin, M. ; Milios, E. ; Tsotsos, J. ; Jepson, A. ; Bains, O. N.: The ARK Project: Autonomous Mobile Robots for Known Industrial Environments. In: *Robotics and Autonomous Systems* 25 (1998), October, Nr. 1-2, S. 83–104

[NM05] Novak, G. ; Mahlknecht, S.: TINYPHOON A Tiny Autonomous Mobile Robot. In: *Proceedings of the IEEE International Symposium on Industrial Electronics*, 2005, S. 1533–1538

[Nov03] Novak, G.: Roboter Soccer: An Example for Autonomous Mobile Cooperating Robots. In: *Proceedings of the Workshop on Intelligent Solutions in Embedded Systems.* Vienna, Austria, June 2003, S. 107–118

[NPB95] Nourbakhsh, I. ; Powers, R. ; Birchfield, S.: Dervish: An Office-Navigating Robot. In: *Artificial Intelligence Magazine* 16 (1995), S. 53–60

[OKK98] Ohya, I. ; Kosaka, A. ; Kak, A.: Vision-Based Navigation by a Mobile Robot with Obstacle Avoidance Using Single-Camera Vision and Ultrasonic sensing. In: *IEEE Transactions on Robotics and Automation* 14 (1998), December, Nr. 6, S. 969 – 978

[OTY04] Ohno, K. ; Tsubouchi, T. ; Yuta, S.: Outdoor Map Building Based on Odometry and RTK-GPS Positioning Fusion. In: *Proceedings of the IEEE International Conference on Robotics and Automation* Bd. 1, 2004, S. 684 – 690

[PCCL96] Park, K. ; Chung, D. ; Chung, H. ; Lee, J. G.: Dead Reckoning Navigation of a Mobile Robot Using the Indirect Kalman Filter. In: *Proceedings of the IEEE/SICE/RSJ International Conference on Multisensor Fusion and Integration for Intelligent Systems*, 1996

[PFTV92] Press, W. H. ; Flannery, B. P. ; Teukolsky, S. A. ; Vetterling, W. T.: *Numerical Recipes in C : The Art of Scientific Computing.* 2. Cambridge University Press, October 1992. – ISBN 0521431085

[PHL98] Park, K. ; H.Chung ; Lee, J.: Dead Reckoning Navigation for Autonomous Mobile Robots. In: *Proceedings of the Intelligent Autonomous Vehicle.* Madrid, Spain, 1998, S. 775–781

[PPSU01] Panzieri, S. ; Pascucci, F. ; Setola, R. ; Ulivi, G.: A Low Cost Vision Based Localization System for Mobile Robots. In: *Proceedings of the Mediterranean Conference on Control and Automation.* Dubrovnik, Croatia, June 2001

[QL99] Quan, L. ; Lan, Z: Linear N-point Camera Pose Determination. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 21 (1999), August, Nr. 8, S. 774–780

[RB00a] Roumeliotis, S. I. ; Bekey, G. A.: SEGMENTS: A Layered, Dual-Kalman Filter Algorithm for Indoor Feature Extraction. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2000, S. 454–461

[RB00b] Roumeliotis, S.I. ; Bekey, G.A.: Bayesian Estimation and Kalman filtering: A unified Framework for Mobile Robot Localization. In: *Proceedings of the IEEE International Conference on Robotics and Automation*, 2000, S. 2985–2992

[RB02] Roumeliotis, S. I. ; Bekey, G. A.: Distributed multirobot Localization. In: *IEEE Transactions on Robotics and Automation* 18 (2002), October, Nr. 5, S. 781 – 795

[Ren93] Rencken, W.D.: Concurrent Localization and Map Building for Mobile Robots Using Ultrasonic Sensors. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems.* Yokohama, Japan, July 1993, S. 2192–2197

[Ren94] Rencken, W.D.: Autonomous Sonar Navigation in Indoor, Unknown, and Unstructured Environments. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems.* Munich, Germany, September 1994, S. 127–134

[RHD$^+$02] Ranganathan, P. ; Hayet, J.B. ; Devy, M. ; Hutchinson, S. ; Lerasle, F.: Topological Navigation and Qualitative Localization for Indoor Environment Using Multi-Sensory Perception. In: *Robotics and Autonomous Systems* 41 (2002), S. 137 – 144

[RJ04] Röfer, T. ; Jüngel, M.: Fast and Robust Edge-Based Localization in the Sony Four-Legged Robot League. In: *RoboCup 2003: Robot Soccer World Cup VII* Bd. 3020, Springer-Verlag GmbH, 2004, S. 262 – 273

[RR95] R.Bauer ; Rencken, W. D.: Sonar Feature Based Exploration. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems.* Pittsburgh, Pennsylvania, August 1995, S. 148–153

[RT71] Rosenfeld, A. ; Thurston, M.: Edge and Curve Detection for Visual Scene Analysis. In: *IEEE Transactions on Computers* 20 (1971), Nr. 5, S. 562–569

[Saf97] Saffiotti, A.: The Uses of Fuzzy Logic in Autonomous Robot Navigation. In: *Soft Computing - A Fusion of Foundations, Methodologies and Applications* 1 (1997), December, Nr. 4, S. 180 – 197

[SB93] Sutherland, K. T. ; B.Thompson, W.: Inexact Navigation. In: *Proceedings of the IEEE International Conference on Robotics and Automation*, 1993, S. 1–7

[SB05] Steinbauer, G. ; Bischof, H.: Illumination Insensitive Robot Self-Localization Using Panoramic Eigenspaces. In: *RoboCup 2004: Robot Soccer World Cup VIII* Bd. 3276, 2005, S. 84 – 96

[SC94] Schiele, B. ; Crowley, J. L.: A Comparison of Position Estimation Techniques Using Occupancy Grids. In: *Proceedings of the IEEE International Conference on Robotics and Automation* Bd. 2, 1994, S. 1628–1634

[SD95] Stella, E. ; Distante, A.: Self-Location of a Mobile Robot by Estimation of Camera parameters. In: *Robotics and Autonomous Systems* 15 (1995), S. 179–187

[SHB$^+$02] Schmitt, T. ; Hanek, R. ; Beetz, M. ; Buck, S. ; Radig, B.: Cooperative Probabilistic State Estimation for Vision-Based Autonomous Mobile Robots. In: *IEEE Transactions on Robotics and Automation* 18 (2002), October, Nr. 5, S. 670 – 684

[Shi02] Shimshoni, I.: On Mobile Robot Localization from Landmark Bearings. In: *IEEE Transactions on Robotics and Automation* 18 (2002), DECEMBER, Nr. 6, S. 971–976

[SK95] Simmons, R. ; Koenig, S.: Probabilistic Robot Navigation in Partially Observable Environments. In: *Proceedings of the International Joint Conference on Artificial Intelligence*, 1995, S. 1080–1087

[SN04] Siegwart, R. ; Nourbakhsh, I. R.: *Introduction to Autonomous Mobile Robots*. The MIT Press, April 2004. – ISBN 026219502X

[Soj02] Sojka, E.: A New and Efficient Algorithm for Detecting the Corners in Digital Images. In: Gool, L. V. (Hrsg.): *Proceedings of the DAGM Symposium* Bd. 2449, Springer, 2002, S. 125–132

[SS86] Steier, W. H. ; Shori, R. K.: Optical Hough Transform. In: *Applied Optics* 25 (1986), Nr. 16, S. 2734 – 2738

[SS00] Sinriech, D. ; Shoval, S.: Landmark Configuration for Absolute Positioning of Autonomous Vehicles. In: *IIE Transactions* 32 (2000), S. 613–624

[SSB03] Stroupe, A. W. ; Sikorski, K. ; Balch, T.: Constraint-Based Landmark Localization. In: Kaminka, G.A. (Hrsg.) ; Lima, P.U. (Hrsg.) ; Rojas, R. (Hrsg.): *RoboCup 2002: Robot Soccer World Cup VI* Bd. 2752, Springer-Verlag, November 2003, S. 8–24

[SSGS02] Singh, S. ; Srinivasan, T.P. ; Gupta, A. ; Srivastava, P. K.: Improving the Accuracy of High Resolution Image Data Products Using Kalman Filter. In: *Indian Cartographer* DAPI-13 (2002), S. 73

[Sug88] Sugihara, K.: Some Location Problems for Robot Navigation Using a Single Camera. In: *Computer Vision, Graphics, and Image Processing* 42 (1988), Nr. 1, S. 112–129

[SZL95] Shoval, S. ; Zeitoun, I. ; Lenz, E.: Layout of Beacon for Triangulation of AGV's in Industrial Environments. In: *Proceedings of the International Conference on Production Research*, 1995, S. 485 – 487

[SZO$^+$03] Sharon, A. ; Zentner, A. ; Oren, E. ; Goldberg, Y. ; Sharvit, Z. ; Kirkpatrick, S. ; Jonas, A.: *Robot Localization Using Ultrasonic and Radio Frequency Signals*. `humanrobot.kist.re.kr/New_papers/cjs/c-04.PDF`. 2003. – Engineering Project: School of Engineering and Computer Science The Hebrew University, Jerusalem

[TA90] Talluri, R. ; Aggarwal, J.K.: Position Estimation for a Mobile Robot in an Unstructured Environment. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1990, S. 159–166

[TA91] Talluri, R. ; Aggarwal, J.K.: Position Estimation of a Mobile Robot Using Edge Visibility Regions. In: *Proceedings of International Conference on Computer Vision and Pattern Recognition*, 1991, S. 714–715

[TA93] Talluri, R. ; Aggarwal, J.: Position Estimation Techniques for an Autonomous Mobile Robot - a Review. In: Chen, C. H. (Hrsg.): *Handbook of Pattern Recognition and Computer Vision*. World Scientific: Singapore, 1993, Kapitel 4.4, S. 769–801

[TFBD01] Thrun, S. ; Fox, D. ; Burgard, W. ; Dellaert, F.: Robust Monte Carlo Localization for Mobile Robots. In: *Artificial Intelligence* 128 (2001), S. 99–141

[Thr98] Thrun, S.: Bayesian Landmark Learning for Mobile Robot Localization. In: *Machine Learning* 33 (1998), Nr. 1, S. 41–76

[TNS03] Tang, Z. ; Nazeer, A. ; Sun, Z.: Robust Vision Localization For RoboCup F180. In: *Singapore Polytechnic Technical Journal* (2003), May

[TRM$^+$05] Tehrani, A. F. ; Rojas, R. ; Moballegh, H. R. ; Hosseini, I. ; Amini, P.: Analysis by Synthesis, a Novel Method in Mobile Robot Self-Localization. In: et al., G.A. K. (Hrsg.): *RoboCup 2004: Robot Soccer World Cup VIII* Bd. 3276, 2005, S. 586–593

[TTA94] Tonouchi, Y. ; Tsubouchi, T. ; Arimoto, S.: Fusion of Dead-Reckoning Positions With a Workspace Model for a Mobile Robot by Bayesian Inference. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1994, S. 1347–1354

[TV98] Trucco, E. ; Verri, A.: *Introductory TEchniques for 3-D Computer Vision*. Upper Saddle River, NJ, USA : Prentice Hall, 1998. – ISBN 0–13–261108–2

[Uhr72] Uhr, L.: Layered 'Recognition Cone' Networks that Preprocess, Classify and Describe. In: *IEEE Transactions on Computers* (1972), S. 759–768

[UNMK03] Utz, H. ; Neubeck, A. ; Mayer, G. ; Kraetzschmar, G.: Improving Vision-Based Self-Localization. In: et al., G. K. (Hrsg.): *RoboCup 2002: Robot Soccer World Cup VI*, Springer-Verlag, 2003 (Lecture Notes in Computer Science 2752), S. 25–40

[VXBA96] Vandorpe, J. ; Xu, H. ; Brussel, H. V. ; Aertbelien, E.: Positioning of the Mobile Robot LiAS Using Natural Landmarks and a 2D Range Finder. In: *Proceedings of the IEEE/SICE/RSJ International Conference on Multisensor Fusion and Integration for Intelligent Systems*, 1996

[Wan88] Wang, C.M.: Location Estimation and Uncertainty Analysis for Mobile Robots. In: *Proceedings of the IEEE International Conference on Robotics and Automation*, 1988, S. 1230–1235

[WB95] Welch, G. ; Bishop, G.: An Introduction to the Kalman Filter / University of North Carolina at Chapel Hill. Chapel Hill, NC, USA : University of North Carolina at Chapel Hill, 1995. – Forschungsbericht

[WBB05] Wolf, J. ; Burgard, W. ; Burkhardt, H.: Robust Vision-Based Localization by combining an Image-retrieval System with Monte Carlo Localization. In: *IEEE Transactions on Robotics* 21 (2005), April, Nr. 2, S. 208 – 216

[WD96] Weckesser, P. ; Dillmann, R.: Sensor-Fusion of intensity and LaserRange Images. In: *Proceedings of the IEEE/SICE/RSJ International Conference on Multisensor Fusion and Integration for Intelligent Systems*, 1996, S. 501–508

[WFJ$^+$01] Weber, J. ; Franken, L. ; Jorg, K.-W. ; Schmitt, K. ; Puttkamer, E.v.: An Integrative Framework for Global Self-Localization. In: *Proceedings of the IEEE/SICE/RSJ International Conference on Multisensor Fusion and Integration for Intelligent Systems*, 2001, S. 73 – 78

[WFJP02] Weber, J. ; Franken, L. ; Jorg, K.-W. ; Puttkamer, E.v.: Reference Scan Matching for Global Self-Localization. In: *Robotics and Autonomous Systems* 40 (2002), August, Nr. 2, S. 99–110

[Wic98] von Wichert, G.: Mobile Robot Localization Using a Self-organized Visual Environment Representation. In: *Robotics and Autonomous Systems* 25 (1998), S. 185 – 194

[WP03] Wolf, J. ; Pinz, A.: Particle Filter for Self Localization Using Panoramic Vision. In: *ÖGAI Journal* 22 (2003), May, Nr. 4, S. 8–15

[WT99] Williamson, T. ; Thorpe, C.: A Trinocular Stereo System for Highway Obstacle Detection. In: *Proceedings of the IEEE International Conference on Robotics and Automation* Bd. 3, 1999, S. 2267 – 2273

[XOK90] Xu, L. ; Oja, E. ; Kultanen, P.: A New Curve Detection Method: Randomized Hough Transform (RHT). In: *Pattern Recognition Letters* 11 (1990), S. 331–338

[YM05] Yuen, D. C. K. ; MacDonald, B. A.: Vision-Based Localization Algorithm Based on Landmark Matching, Triangulation, Reconstruction, and Comparison. In: *IEEE Transactions on Robotics* 21 (2005), April, Nr. 2, S. 217 – 226

[Zol03] Zollner, H.: A Calibration Technique for CCD Cameras Using Pose Estimation / PRIP, TU Wien. 2003 (PRIP-TR-085). – Forschungsbericht

[ZRJ03] Zezhong, X. ; Ronghua, L. ; Jilin, L.: Global Localization Based on Corner Point. In: *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation*, 2003, S. 843 – 847

Internet References

[FIR06] FIRA MiroSot WorldCup: *Official Website of the 11th FIRA RoboWorld Cup Germany 2006 in Dortmund and the 5th FIRA RoboWorld Congress 2006.* http://www.firaworldcup.de/. 2006. – [Online; accessed 10-April-2007]

[FIR07a] FIRA MiroSot: *Overall System:FIRA MiroSot Small League.* http://www.fira.net/Soccer/mirosot/appendix2.html. 2007. – [Online; accessed 21-February-2007]

[FIR07b] FIRA MiroSot: *Overview: FIRA MiroSot Small League.* http://www.fira.net/Soccer/mirosot/overview.html. 2007. – [Online; accessed 21-February-2007]

[Rob07a] RoboCup: *Overall System:RoboCup Small Size Robot League.* http://Small-size.informatik.uni-bremen.de/_detail/dataflow.png?id=start&cache=cache. 2007. – [Online; accessed 21-February-2007]

[Rob07b] RoboCup: *Overview: RoboCup Small Size Robot League (F180).* http://Small-size.informatik.uni-bremen.de/. 2007. – [Online; accessed 21-February-2007]

[Wik07a] Wikipedia: *Charge-Coupled Device — Wikipedia, The Free Encyclopedia.* http://en.wikipedia.org/. 2007. – [Online; accessed 21-February-2007]

[Wik07b] Wikipedia: *Kalman Filter — Wikipedia, The Free Encyclopedia.* http://en.wikipedia.org/. 2007. – [Online; accessed 11-February-2007]