# VISION-BASED ROBOT LOCALIZATION USING ARTIFICIAL AND NATURAL LANDMARKS

ZAFER ARICAN

AUGUST 2004

# VISION-BASED ROBOT LOCALIZATION USING ARTIFICIAL AND NATURAL LANDMARKS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ZAFER ARICAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

AUGUST 2004

Approval of the Graduate School of Natural and Applied Sciences

Prof. Dr. Canan Özgen
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Mübeccel Demirekler
Head of the Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Prof. Dr. Uğur Halıcı
Supervisor

Examining Committee Members

| | | |
|---|---|---|
| Prof. Dr. Kemal Leblebicioğlu | (METU,EE) | |
| Prof. Dr. Uğur Halıcı | (METU,EE) | |
| Prof. Dr. Turgut Tümer | (METU,ME) | |
| Assoc. Prof. Aydın Alatan | (METU,EE) | |
| Dr. Uğur Leloğlu | (BİLTEN) | |

# Abstract

VISION-BASED ROBOT LOCALIZATION USING ARTIFICIAL AND NATURAL LANDMARKS

Arıcan, Zafer
M.Sc., Department of Electrical and Electronics Engineering
Supervisor: Prof. Dr. Uğur Halıcı

August 2004, 76 pages

In mobile robot applications, it is an important issue for a robot to know where it is. Accurate localization becomes crucial for navigation and map building applications because both route to follow and positions of the objects to be inserted into the map highly depend on the position of the robot in the environment.

For localization, the robot uses the measurements that it takes by various devices such as laser rangefinders, sonars, odometry devices and vision. Generally these devices give the distances of the objects in the environment to the robot and proceesing these distance information, the robot finds its location in the environment.

In this thesis, two vision-based robot localization algorithms are implemented. The first algorithm uses artificial landmarks as the objects around the robot and by measuring the positions of these landmarks with respect to the camera system, the robot locates itself in the environment. Locations of these landmarks are known. The second algorithm instead of using artificial landmarks, estimates its location by measuring the positions of the objects that naturally exist in the environment. These objects are treated as natural landmarks and locations of these landmarks are not known initially.

A three-wheeled robot base on which a stereo camera system is mounted is used as the mobile robot unit. Processing and control tasks of the system is performed by a stationary PC. Experiments are performed on this robot system. The stereo camera system is the measurement device for this robot.

Keywords: Robot Localization, Artificial Landmarks, Natural Landmarks, Stereo Vision, Extended Kalman Filter, Coded Cylinder, Landmark Detection

# Öz

## YAPAY VE DOĞAL YER İŞARETLERİ KULLANARAK GÖRME TABANLI ROBOT YERİ BELİRLENMESİ

Arıcan, Zafer
Yüksek Lisans, Elektrik Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Uğur Halıcı

Ağustos 2004, 76 sayfa

Gezici robot uygulamalarında, robotun nerede olduğunu bilmesi önemli bir problemdir. Hassas robot yeri belirleme ilerleme ve harita oluşturma uygulamaları için hayati bir öneme sahiptir çünkü hem izlenecek rota hem de haritaya yerleştirilecek nesnelerin yerleri, robotun yerine çok bağlıdır.

Yer belirleme için robot, lazer telemetre, sonar telemetre, odometri ve camera gibi çeşitli ölçüm cihazları kullanır. Genelde bu cihazlar, çevredeki nesnelerin robota olan uzaklığını ölçerler ve robot, bu uzaklık bilgisini kullanarak bulunduğu ortam içindeki yerini hesaplar.

Bu tezde, görme tabanlı iki robot yeri belirleme algoritması hayata geçirilmiştir. Birinci algoritma, yapay yer işaretlerine göre pozisyonun ölçülüp, bulunduğu ortamdaki yerini bulmaya yöneliktir. Bu yapay yer işaretlerinin yerleri bilinmektedir. İkinci algoritma ise yapay yer işaretleri kullanmadan ortamda doğal olarak bulunan nesnelerin robota göre pozisyonunun bulunup, bu ölçümlere dayanarak robot yeri belirlemeye yöneliktir. evredeki nesneler doğal yer işaretleri olarak kabul edilir ve bu yer işaretlerinin yerleri, başlangıçta bilinmemektedir.

Üzerine stereo kamera sistemi yerleştirilmiş 3 tekerlekli bir robot aracı, hareketli ünite olarak kullanılmıştır. İşleme ve kontrol işlemleri sabit bir bilgisayarca yerine getirilmektedir. Deneyler bu sistem üzerinde yapılmıştır. Stereo kamera sistemi bu robot için ölçüm cihazı olarak kullanılmıştır.

Anahtar Kelimeler: Robot Yeri Belirleme, Yapay yer İşaretleri, Doğal Yer İşaretleri, Stereo Görme, Genişletilmiş Kalman Filtresi, Kodlanmış Silindir, Yer İşareti Bulma

To My Father, My Mother and My Lovely Sister

# Acknowledgments

This thesis has been conducted in Computer Vision and Intelligent Systems Research Laboratory in Electrical and Electronics Department and has been partly supported in projects BAP-2002-03-01-05 and BAP.04.03.01.02.

I would like to express my sincere gratitude to my supervisor, Prof. Dr. Uğur HALICI, for her guidance and encouragement throughout the thesis study. I also would like to thank Prof. Dr. Kemal Leblebicioğlu for his suggestions.

I would like to thank to my family for their support, patience and encouragement. Special thanks are for my lovely sister who supports me everytime, everywhere.

I am thankful to Dr.İlkay Ulusoy, for her guidance, support and studies which inspired me on choosing my thesis topic, to Mustafa, for his camera system, assistance, and invaluable friendship, and to his family for their hospitality in İzmir.

Thanks are for members of Computer Vision and Intelligent Systems Research Laboratory, and all my friends for their friendship and support.

# Table of Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Problem Definition and Motivation

In mobile robot applications, it is an important issue for the robot to know where it is. Accurate localization becomes crucial for navigation and map building because both route to follow and features to be inserted into the map, highly depends upon the position of the robot in the environment where the robot operates. Although some applications, such as automatic vacuum cleaner and lawnmower, do not need to localize themselves to carry out their tasks, knowing where they are can improve the performance and optimize time and cost. Robot localization is an important property for a robot which has autonomous capabilities and it has been referred to as "the most fundamental problem to providing a mobile robot with autonomous capabilities"[16].

The robot localization problem can be categorized in different ways. Approaches to solve the problem, used measurement devices and application area of the robot are the main criteria to categorize the problem.

Commonly, pose of the robot is represented by an uncertainty and problem is approached as probabilistic. If initial pose of the robot is known and estimation of the pose is performed by incrementally updating the pose, it is called "position tracking" or "incremental localization". Position tracking seeks to correct the incremental errors caused by the system. A second approach is the "global localization" which tries to solve the problem without any initial robot pose and uses all the data it obtained at once. Since it processes the data at once, it is not a real-time process.

Both position tracking and global localization needs a map of the environment to obtain the data for the estimation of the position. It is possible to give the map of the environment like a plan of the building in which the robot roams. Otherwise, the robot obtains the information about its position using the sensor mounted on it and uses the objects around it as reference. In this manner, specific objects called “artificial landmarks” can be used. These landmarks have specific properties generally determined by the users who use the robot system and positions of the landmarks are known. By this way, accurate and long-time localization of the robot can be established. These kinds of landmarks and localization is used in industry, because the environment is planned and artificial landmarks are placed according to this plan. Another approach is to use the objects that naturally exist in the environment and this kind of objects are called “natural landmarks”. Positions of these landmarks are usually not known. Using the relative positions of the landmarks with respect to the robot, localization of the robot can be reached but this is generally for short-term localization. If long-time localization and the building of the map of the environment is aimed another problem called “Simultaneous Localization and Map Building”(SLAM) occurs.

SLAM is based on the concurrent handling of the two tasks namely robot localization and map building which are dependent on each other. For a good map building, accurate position of the robot is needed, besides for a robot to know its position, a map of the environment is required and this is like “chicken and egg”. SLAM algorithms are like the ones introduced before and uses incremental and global approaches.

For both cases of the known and the unknown map, it is required for a robot to get data to make inference about its location. It is performed by taking measurement using different measurement devices. Another category of the robot localization is the measurement devices.

Most of the localization algorithms are based on the measurement devices such as laser rangefinders, sonars, odometry and vision. Laser rangefinders are accurate but slow. Sonar works similar in principle to laser rangefinders, however it is cheaper and faster with a cost of high uncertainty in measurements. Odometry is based on the determination of the location by using robot dynamics including wheel speeds, axes length, wheel diameter. This method may give erroneous results for long-term localization. Because systematic errors such as unequal wheel diameters, wheel misalignment and non-systematic errors including slippage and skidding, may propagate over time and causes a drift.

Vision systems are relatively cheap and give high resolution measurement data.

High information that it gives makes it convenient for other tasks such as object recognition and 3D reconstruction while performing operations used by localization process. Although complexity of the operations for calculation of depth is high in vision systems considering other methods stated above, recent advances in computer technology reduced the computation time of the image processing tasks required for the retrieval of the useful information required for the localization.

A good example of concurrent use of vision systems for both navigation and object recognition and map building is studied in [18]. In that study, a robot builds the cognitive map while navigating in the environment using the stereo camera systems. It recognize the objects which it represents in the map. However, experiments related to this study is performed in a virtual environment. In this environment, movement of the robot is perfect and at each step it is able to know its exact location.

In this thesis, our aim is to construct the robot system for future realization of the tasks performed in the virtual environment used in the study [18] and to study the robot localization problem to make the robot system able to estimate its location in the real environment.

## 1.2 Literature Review

Beginning from the second half of the 80s, interest on the robot localization problem increased. As stated in [16], it is accepted to be the fundamental problem to make a robot autonomous. First successful studies on the subject are [17, 15, 19]. Two approaches which are incremental localization and absolute localization are the most implemented ones to solve the localization problem. For incremental localization, Extended Kalman Filter(EKF) is widely used by many researchers in [3, 4, 19, 2]. In this studies, with the knowledge of the initial position of the robot, at each time step, robot position is estimated by updating the state of the position using the measurements. This method is convenient for real-time use because it uses the data at that time step only. Distributions of the random variables in this approach is Gaussian and this provide many useful properties for the systems using EKF.

For global localization, different methods similar to those studied in [6, 21, 28] are used. These methods do not need the initial position of the robot and in addition distributions of the random variables do not need to be Gaussian. Sample sets are used to represent and process non-Gaussian distributions.

Both incremental and global localization needs a map of the environment to estimate the position of the robot. If map of the environment is not known also, robot localization is investigated with the map building as SLAM problem. Some studies on this subject are [3, 4, 28, 32]. In [5], there is a good review for the SLAM problem. In this problem, while features in the environment are put into a map build by the system, robot position is estimated according to these features. Initial position of the robot is assumed to be the origin of the environment and mapping and localization is performed according to this frame of reference.

All the methods do need measurements either taken from the internal sensors or sensors to distance measurement devices. Studies in this field use sonars [6, 8], laser rangefinders [6, 12, 28] and vision [3, 4, 7, 24, 32].

Sonar, laser rangefinders and vision systems take measurement such that position of the features with respect to the robot is measured. For a proper localization, features (landmarks) should have some properties which provide the system to identify the landmarks easily. For this purpose, two types of landmarks namely artificial and natural landmarks are used. Artificial landmarks are specific to the system which uses it. Self-similar intensity patterns like barcode are used in [31] as artificial landmarks. In [26], black and white disks are used and projections of the disks on the image plane are processed using elliptic Hough transform. Zitová and Flusser [35] use two concentric circles as landmarks and present a recognition algoritm based on the affine moment invariants.

Natural landmarks are the features that are extracted from the objects that naturally exist in the environment. Kosaka and Kak [24] use vertical lines like doors, wall edges and tables as landmarks. Davison [3, 4] uses strong corners as features. Scale-invariant features (SIFT) is an innovative solution to invariant features which is used as natural landmarks in [32].

## 1.3 Thesis Outline

This thesis, studies the robot localization problem using stereo vision on the existence of artificial and natural landmarks in the environment. The organization of the thesis is as follows:

- Chapter 2: At each step, the location of the robot in this thesis is calculated using Extended Kalman Filter and this chapter gives information about the theory behind the state estimation and the Extended Kalman Filter.

- Chapter 3: Measurement device of the the robot system that we construct is the stereo cameras. This chapter gives the theoretical background about the stereo vision and the calculation of the position of the objects using the stereo vision.

- Chapter 4: In this chapter, the robot system that we build and use in this thesis is explained.

- Chapter 5: In this chapter, detection of the artificial and natural landmarks and matching methods are described. Properties of the artificial landmark type that we used is given.

- Chapter 6: One of the robot localization algorithm based on the landmarks that we implemented is the one which uses the artificial landmarks. In this chapter, the robot localization algorithm and the experiments to test the algorithm is explained.

- Chapter 7: The other robot localization algorithm that we implemented is the one which uses natural landmarks. This chapter describes the algorithm and gives information about the tests performed related to this algorithm

- Chapter 8: This chapter gives a summary of the results obtained during the thesis study and in the light of experience gained, offers some future work to improve the performance of the system.

# Chapter 2

# State Estimation

## 2.1 Introduction

Estimating the state in a linear or non-linear stochastic dynamic system is a problem in many applications like tracking, robot localization and navigation, signal processing. In 1960, R.E.Kalman introduced an algorithm which is called with his name for the linear filtering problem [23, 34] .This algorithm is composed of a set of mathematical equations and describes a recursive solution to the problem. Later, for the non-linear systems, an extension of this filter, namely Extended Kalman Filter (EKF) was developed[34].

This chapter explains these two filters with a review of linear and non-linear stochastic dynamic systems. Although, these filters work for both continuous and discrete-systems, for rest of the chapter, discrete-time stochastic systems will be investigated. Section 2.2 explains stochastic linear dynamic systems and introduces some assumption that will be used in rest of the chapter. In section 2.3 discrete-time linear Kalman filter is described and equations of the filter are given. Section 2.4 and 2.5, deals with non-linear stochastic dynamic systems and EKF.

## 2.2 Stochastic Linear Dynamic Systems

### 2.2.1 State Space Model

State-space model of a discrete-time linear stochastic dynamic system can be expressed as

$$\mathbf{x}(k+1) = \mathtt{F}(k)\mathbf{x}(k) + \mathtt{G}(k)\mathbf{u}(k) + \mathtt{B}(k)\mathbf{w}(k) \tag{2.1}$$

where $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{u} \in \mathbb{R}^m$ and $\mathbf{w} \in \mathbb{R}^l$ are *state*, *input* and *process noise* vectors, respectively. $\mathtt{F}(k)$ is an $n \times n$ matrix called *transition matrix*, $\mathtt{G}(k)$ is the $n \times m$ *input gain matrix*, and $\mathtt{B}(k)$ is the *process noise gain matrix* which has dimensions of $n \times l$.

Similarly, measurement equation of the system can be written as

$$\mathbf{z}(k) = \mathtt{H}(k)\mathbf{x}(k) + \mathtt{C}(k)\mathbf{v}(k) \tag{2.2}$$

where $\mathbf{z} \in \mathbb{R}^p$ is the *output* vector, $\mathbf{x} \in \mathbb{R}^n$ is the state vector, and $\mathbf{v} \in \mathbb{R}^g$ is the *measurement noise* vector. Reflection of the state to the output is accomplished by the *measurement matrix* $\mathtt{H}(k)$ of dimension $p \times n$. $\mathtt{C}(k)$ is the $p \times g$ *measurement noise gain matrix.*

Noise vectors, $\mathbf{w}(k)$, $\mathbf{v}(k)$ and related gain matrices are introduced in stochastic models to handle three uncertainties which deterministic system models cannot handle. These uncertainties are:

- Imperfections in models: Physical systems cannot be sufficiently characterized by mathematical models.
- Uncontrolled inputs or disturbances entering into system: Environment conditions affect the systems in uncontrolled manner.
- Inaccuracy in measurement devices: Noise in sensors may cause the device to take wrong measurements.

Considering these uncertainties, the noise vectors are random variables with appropriate probability density functions (pdf). Because the noise vectors in the state and measurement equations are random variables, state and output vectors are random variables, too.

Input vector, $\mathbf{u}(k)$ can be either deterministic or probabilistic depending on the system being open-loop or closed-loop. In open-loop systems input is known and independent of the output of the system. In robotic applications that have that kind of characteristics, input is predefined or given by an operator. In closed-loop systems, input is a function of the output $\mathbf{z}(k)$. Since output vector is probabilistic, input is also probabilistic. Automated servoing in robotics navigation applications is one of the examples to closed-loop systems.

### 2.2.2 Gauss-Markov Model and Assumptions

If random variables in Equation. 2.1 and 2.2 are Gaussian and $\mathbf{x}(k+1)$ depends on only present values of random variables $\mathbf{x}(k)$, $\mathbf{w}(k)$, $\mathbf{v}(k)$, the state-space model is a Gauss-Markov Model. Three basic assumptions about the system can guarantee Gauss-Markov property.

- First of all, random variables in the system model are Gaussian. That is, each random variable has a pdf of the form

  $$\mathbf{p}(\mathbf{x}) = \frac{1}{|2\pi\Sigma|^{-1/2}} e^{-1/2(\mathbf{x}-\overline{\mathbf{x}})^{\mathsf{T}}\Sigma^{-1}(\mathbf{x}-\overline{\mathbf{x}})} \tag{2.3}$$

  where

  $$\overline{\mathbf{x}} = E[\mathbf{x}] \tag{2.4}$$
  $$\Sigma = E\left[(\mathbf{x}-\overline{\mathbf{x}})(\mathbf{x}-\overline{\mathbf{x}})^{\mathsf{T}}\right] \tag{2.5}$$

  are mean and covariance matrix of the random vector $\mathbf{x}$, respectively.

- Second, process and measurement vectors are *white* noise vectors. In other words,

  $$E\left[\mathbf{w}(k)\mathbf{w}(j)^{\mathsf{T}}\right] = \begin{cases} 0 & k \neq j \\ Q(k) & k = j \end{cases} \tag{2.6}$$

  where $Q(k)$ is the covariance matrix.

  Whiteness implies that knowing the value of the noise at any time does not give any information about the value of the noise at any other time instant. This assumption is crucial for the Markov property of the system.

- Finally, random processes, $\mathbf{w}(k)$, $\mathbf{v}(k)$ are uncorrelated of each other. Similar to whiteness property, knowing $\mathbf{w}(k)$ at any time instant, does not help predicting the value of $\mathbf{v}(k)$.

Gauss-Markov property facilitates the computation of the posterior pdf of the state. Gaussian pdf can be completely determined by mean and covariance of the random variable, and pdf that is given in Equation. 2.3 can be represented as $N(\overline{\mathbf{x}}, \Sigma)$. Thus, to compute only mean and covariance of the random variable is enough to determine the pdf. Another important feature of the Gaussian pdf is after a linear transformation, output is still a Gaussian. That is if $\mathbf{x}(k)$ is Gaussian, $\mathtt{G}\mathbf{x}(k)$ is also Gaussian given that $\mathtt{G}$ is a linear transformation. Combined with the Markov property, this feature makes the posterior pdf of the state determinable recursively at each time instant.

## 2.3 Kalman Filter

*The Kalman filter* or *Kalman State Estimator* is optimal recursive linear state estimation algorithm provided that assumptions in Section 2.2.2 hold for the system and the system is linear with state and measurement equations are as Equation.2.1 and 2.2 with known input $\mathbf{u}(k)$ and noise vectors $\mathbf{w}(k)$, $\mathbf{v}(k)$ with normal probability

$$\begin{aligned} p(\mathbf{w}(k)) &\sim N(0, Q(k)) \\ p(\mathbf{v}(k)) &\sim N(0, R(k)). \end{aligned}$$

Dynamic estimation of the mean and covariance of the state is obtained by recursive static estimation equations

$$\hat{\mathbf{x}} = \overline{\mathbf{x}} + \Sigma_{xz}\Sigma_{zz}^{-1}(\mathbf{z} - \overline{\mathbf{z}}) \tag{2.7}$$

$$\Sigma_{xx|z} = \Sigma_{xx} - \Sigma_{xz}\Sigma_{zz}^{-1}\Sigma_{zx} \tag{2.8}$$

where $\hat{\mathbf{x}}$ is the estimated state mean, $\overline{\mathbf{x}}$ is the predicted state mean,$\mathbf{z}$ and $\overline{\mathbf{z}}$ are actual and predicted measurements, respectively. $\Sigma_{xz}$, $\Sigma_{zx}$, $\Sigma_{zz}$ are covariance and cross-covariance of the state and measurement vectors accordingly. For derivation of the above equations, see [1].

Terms in Equation 2.7 and 2.8 can be adapted to dynamic estimation in the following way.

- Predicted state
$$\overline{\mathbf{x}} \rightarrow \hat{\mathbf{x}}(k+1|k)$$
- Predicted measurement
$$\overline{\mathbf{z}} \rightarrow \hat{\mathbf{z}}(k+1|k)$$
- Corrected(Updated) state
$$\hat{\mathbf{x}} \rightarrow \hat{\mathbf{x}}(k+1|k+1)$$
- Residual or innovation
$$\nu(k+1) = \tilde{\mathbf{z}}(k+1|k) = \mathbf{z}(k+1) - \hat{\mathbf{z}}(k+1|k)$$
- State prediction error
$$\tilde{\mathbf{x}}(k+1|k) = \mathbf{x}(k+1) - \hat{\mathbf{x}}(k+1|k)$$

- Predicted state covariance

$$\Sigma_{xx} \rightarrow \mathtt{P}(k+1|k) \triangleq cov[\mathbf{x}(k+1)] = cov[\tilde{\mathbf{x}}(k+1|k)]$$

- Innovation covariance

$$\Sigma_{zz} \rightarrow \mathtt{S}(k+1) \triangleq cov[\mathbf{z}(\mathbf{k+1})] = cov[\tilde{\mathbf{z}}(k+1)]$$

- Updated state covariance

$$\Sigma_{xx|z} \rightarrow \mathtt{P}(k+1|k+1) \triangleq cov[\mathbf{x}(k+1)] = cov[\tilde{\mathbf{x}}(k+1|k+1)]$$

- Filter gain

$$\Sigma_{xz}\Sigma_{zz}^{-1} \rightarrow \mathtt{K}(k+1) \triangleq cov[\mathbf{x}(\mathbf{k+1}), \mathbf{z}(k+1)]\mathtt{S}(k+1)^{-1}$$

To obtain Kalman Filter equations, state and measurement Equations in Eqn. 2.1 and 2.2 are substituted into the covariance and mean definitions which are defined above. Using assumptions that noise vectors are zero-mean, white and uncorrelated with each other, following equations are obtained.

$$\begin{aligned}
\hat{\mathbf{x}}(k+1|k) &= \mathtt{F}(k)\hat{\mathbf{x}}(k|k) + \mathtt{G}(k)\mathbf{u}(\mathbf{k}) && (2.9)\\
\mathtt{P}(k+1|k) &= \mathtt{F}(k)\mathtt{P}(k|k)\mathtt{F}(k)^{\mathsf{T}} + \mathtt{B}(k)\mathtt{Q}(k)\mathtt{B}(k)^{\mathsf{T}} && (2.10)\\
\hat{\mathbf{z}}(k+1|k) &= \mathtt{H}(k+1)\hat{\mathbf{x}}(k+1)|k) && (2.11)\\
\mathtt{S}(k+1) &= \mathtt{H}(k+1)\mathtt{P}(k+1|k)\mathtt{H}(k+1)^{\mathsf{T}} \\
&\quad + \mathtt{C}(k+1)\mathtt{R}(k+1)\mathtt{C}(k+1)^{\mathsf{T}} && (2.12)\\
\mathtt{K}(k+1) &= \mathtt{P}(k+1|k)\mathtt{H}(k+1)^{\mathsf{T}}\mathtt{S}(k+1)^{-1} && (2.13)\\
\hat{\mathbf{x}}(k+1|k+1) &= \hat{\mathbf{x}}(k+1|k) + \mathtt{K}(k+1)(\mathbf{z}(\mathbf{k+1}) - \hat{\mathbf{z}}(k+1|k)) && (2.14)\\
\mathtt{P}(k+1|k+1) &= \mathtt{P}(k+1|k) - \mathtt{K}(k+1)\mathtt{S}(k+1)\mathtt{K}(k+1)^{\mathsf{T}} && (2.15)
\end{aligned}$$

In many systems, process noise gain matrix, $\mathtt{B}(k)$ and measurement noise gain matrix, $\mathtt{C}(k)$ are identity matrices and therefore can be removed from the equations in which they exist.

Kalman Filter is also known as *prediction-correction*[34] algorithm. In prediction phase, mean and covariance of the state and measurement vectors are projected from previous time step *k* to the next step *k+1* according to the state and measurement equations. In correction phase, predicted mean and covariances are corrected with taken measurements $\mathbf{z}(k+1)$. Figure 2.1 shows grouping of filter equations according

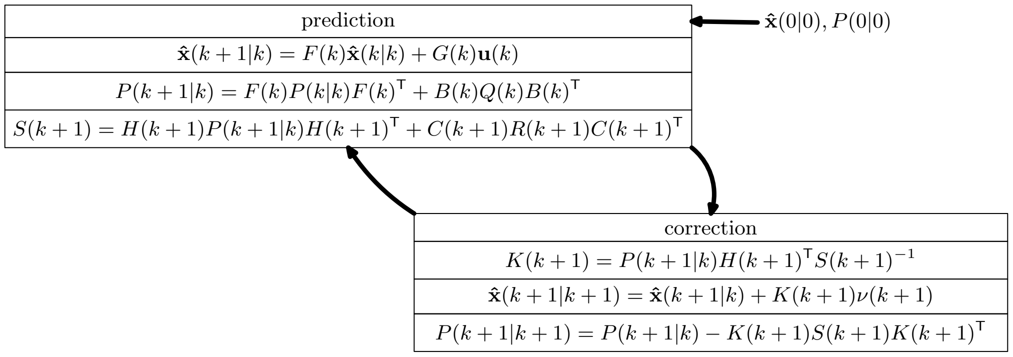

prediction
$\hat{\mathbf{x}}(k+1|k) = F(k)\hat{\mathbf{x}}(k|k) + G(k)\mathbf{u}(k)$
$P(k+1|k) = F(k)P(k|k)F(k)^{\mathsf{T}} + B(k)Q(k)B(k)^{\mathsf{T}}$
$S(k+1) = H(k+1)P(k+1|k)H(k+1)^{\mathsf{T}} + C(k+1)R(k+1)C(k+1)^{\mathsf{T}}$
$\hat{\mathbf{x}}(0|0), P(0|0)$
correction
$K(k+1) = P(k+1|k)H(k+1)^{\mathsf{T}}S(k+1)^{-1}$
$\hat{\mathbf{x}}(k+1|k+1) = \hat{\mathbf{x}}(k+1|k) + K(k+1)\nu(k+1)$
$P(k+1|k+1) = P(k+1|k) - K(k+1)S(k+1)K(k+1)^{\mathsf{T}}$

Figure 2.1: Kalman Filter cycle with prediction and correction equations

to being a prediction or correction equation and cycle between them starting from the initial state $\hat{\mathbf{x}}(0|0)$ and state covariance $\mathtt{P}(0|0)$.

Process and measurement noise covariances $\mathtt{Q}(k)$ and $\mathtt{R}(k)$ determine the trust on state and measurement models. If matrix norm $\|\mathtt{Q}(k)\|$ of $\mathtt{Q}(k)$ is smaller than the matrix norm $\|\mathtt{R}(k)\|$ of $\mathtt{R}(k)$, process model is more trusted than measurements and therefore effect of the taken measurement on the state estimate is small. In other case, if $\|\mathtt{R}(k)\|$ is smaller, measurements are more trusted, and highly affect the state estimate. Both of the covariances take place in the filter gain expression $\mathtt{K}(k+1)$ and affects the contribution of measurements on the state estimate. Thus, $\mathtt{Q}(k)$ and $\mathtt{R}(k)$ determines the characteristics of the filter and used for *fine tuning*[34] of the filter.

If the system is time invariant, gain will converge quickly, therefore, gain and covariance matrices can be computed offline. This reduces the computational time greatly and makes the algorithm more convenient for real-time applications.

## 2.4 Stochastic Non-linear Dynamic Systems and Linearization

The Kalman Filter deals with the problem of state estimation of a *linear* stochastic dynamic system. However, there are systems that show nonlinear characteristics and cannot be modelled as linear state space models in Eqn. 2.1 and 2.2. For this systems, state equation is

$$\mathbf{x}(k+1) = f(\mathbf{x}(k), \mathbf{u}(k), \mathbf{w}(k)) \tag{2.16}$$

where $f$ is the *non-linear* function that takes previous state $\mathbf{x}(k)$, known input $\mathbf{u}(k)$ and process noise $\mathbf{w}(k)$ as input and relates the state at time step *k+1* to the state at time step $k$. Measurement equation is

$$\mathbf{z}(k+1) = h(\mathbf{x}(k), \mathbf{v}(k)) \tag{2.17}$$

In fact, these two equations are the general expression of the dynamic systems. Linear models in Eqn. 2.1 and 2.2 are the special case of the above two equations. In the case that, one or both of the state and measurement functions are non-linear, dynamic system is non-linear and must be treated accordingly.

To simplify the computations, another assumption is introduced about the system that is $\mathbf{w}(k)$ and $\mathbf{v}(k)$ are additive noise without any multiplicative factor before them, that is

$$\mathbf{x}(k+1) = f(\mathbf{x}(k), \mathbf{u}(k)) + \mathbf{w}(k) \tag{2.18}$$
$$\mathbf{z}(k+1) = h(\mathbf{x}(k+1)) + \mathbf{v}(k) \tag{2.19}$$

With the same assumptions that process and measurement noises $\mathbf{w}(k)$ and $\mathbf{u}(k)$ are zero-mean, white Gaussian noises with covariances $\mathtt{Q}(k)$ and $\mathtt{R}(k)$ respectively and they are independent of each other, resultant state vector is no more Gaussian even entering state vector of the previous time step is Gaussian, because for non-linear transformation, Gaussian pdf is not conserved. Thus, state cannot be estimated recursively any more.

To regain advantages in linear systems, non-linear functions $f$ and $h$ are expanded in vector Taylor series around previous estimate $\hat{\mathbf{x}}(k|k)$ up to first-order. The resultant state and measurement equations are

$$\begin{aligned} \mathbf{x}(k+1) &= f(\hat{\mathbf{x}}(k|k), \mathbf{u}(k)) + \mathtt{f}_{\mathtt{x}}(k)(\mathbf{x}(k) - \hat{\mathbf{x}}(k|k)) + HOT + \mathbf{w}(k) \quad (2.20) \\ \mathbf{z}(k+1) &= h(\hat{\mathbf{x}}(k+1|k), \mathbf{u}(k)) + \mathtt{h}_{\mathtt{x}}(\mathbf{x}(k+1) - \hat{\mathbf{x}}(k+1|k)) \\ &\quad + HOT + \mathbf{v}(k) \quad (2.21) \end{aligned}$$

where $\mathtt{f}_{\mathtt{x}}(k)$ is the Jacobian matrix of the state function with respect to state vector $\mathbf{x}(k)$, that is

$$\mathtt{f}_{\mathtt{x}} \triangleq \left[\nabla_x f(\mathbf{x}, \mathbf{u})^{\mathsf{T}}\right]^{\mathsf{T}} \big|_{x=\hat{\mathbf{x}}(k|k)} \triangleq \frac{\partial \mathtt{f}}{\partial \mathtt{x}},$$

$h_x$ is the Jacobian matrix of the measurement function with respect to $\mathbf{x}$

$$\mathtt{h_x} \triangleq \left[\nabla_x h(\mathbf{x})^\mathsf{T}\right]^\mathsf{T} |_{x=\hat{\mathbf{x}}(k+1|k)} \triangleq \frac{\partial \mathtt{h}}{\partial \mathtt{x}},$$

and HOT represents the higher order terms in the series expansion.

This procedure is also known as *linearization* and if HOT terms are neglected new equations behave like linear equations. Although as higher order terms inserted into the equations, approximated equations approaches to real non-linear equations, its computational cost increases. Even adding second-order terms introduces Hessian of the state and measurement functions which is computationally expensive.

For simplification, higher order terms will be neglected and notations $\frac{\partial \mathtt{f}}{\partial \mathtt{x}}$ and $\frac{\partial \mathtt{h}}{\partial \mathtt{x}}$ without time step notation $k$ will be used for the rest of the chapter.

## 2.5 Extended Kalman Filter

For non-linear stochastic dynamic systems, it is not possible to use Kalman Filter. Thus, an extension of the Kalman Filter called Extended Kalman Filter (EKF) is developed for these systems. EKF is suboptimal state estimator which uses linearized state and measurement equations to estimate the state.

To obtain EKF equations, similar approach used in Section 2.3 is used. With same assumptions introduced in Section 2.2 and 2.3, applying simplified state and measurements in Equations 2.18, 2.19 and linearized Equations 2.20 and 2.21 into mean and covariance definitions in Section 2.3, following EKF equations are obtained.

- Predicted state $\hat{\mathbf{x}}(k+1|k)$ is

$$\hat{\mathbf{x}}(k+1|k) = f(\hat{x}(k|k), u(k)) \tag{2.22}$$

- Predicted measurement $\hat{\mathbf{z}}(k+1|k)$ is

$$\hat{\mathbf{z}}(k+1|k) = h(\hat{\mathbf{x}}(k|k)) \tag{2.23}$$

- Predicted state covariance $\mathtt{P}(k+1|k)$ is

$$\mathtt{P}(k+1|k) = \frac{\partial \mathtt{f}}{\partial \mathtt{x}} \mathtt{P}(k|k) \frac{\partial \mathtt{f}}{\partial \mathtt{x}}^\mathsf{T} + \mathtt{Q}(k) \tag{2.24}$$

- Innovation covariance $\mathtt{S}(k+1)$ is

$$\mathtt{S}(k+1) = \frac{\partial \mathtt{h}}{\partial \mathtt{x}} \mathtt{P}(k|k) \frac{\partial \mathtt{h}}{\partial \mathtt{x}}^\mathsf{T} + \mathtt{R}(k) \tag{2.25}$$

- Innovation or residual $\nu(k+1)$ is

$$\nu(k+1) = \mathbf{z}(k+1) - \hat{\mathbf{z}}(k+1|k) \tag{2.26}$$

where $\mathbf{z}(k+1)$ is the measurement taken at time step $k+1$

- Filter gain $\mathtt{K}(k+1)$ is

$$\mathtt{K}(k+1) = \mathtt{P}(k+1|k)\frac{\partial \mathtt{h}}{\partial \mathtt{x}}\mathtt{S}(k+1)^{-1} \tag{2.27}$$

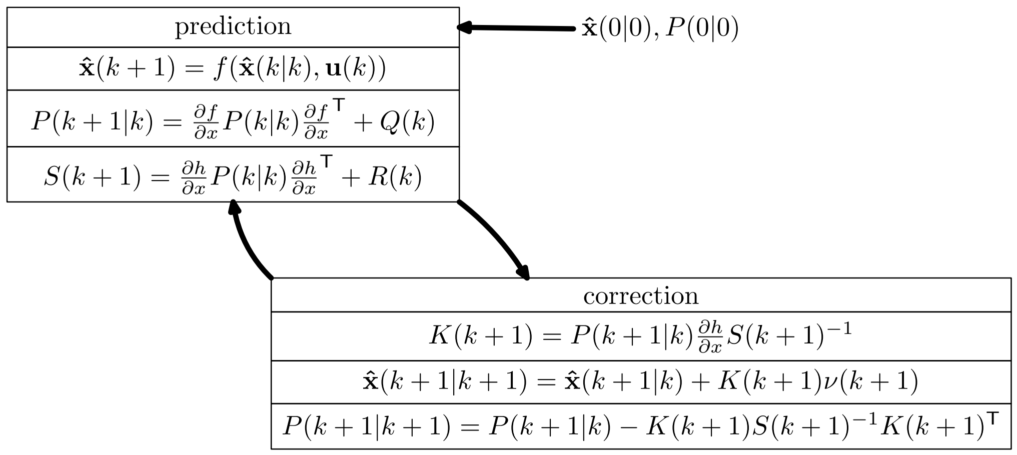
- Updated state $\hat{\mathbf{x}}(k+1|k+1)$ is

$$\hat{\mathbf{x}}(k+1|k+1) = \hat{\mathbf{x}}(k+1|k) + \mathtt{K}(k+1)\nu(k+1) \tag{2.28}$$

- Updated state covariance $\mathtt{P}(k+1|k+1)$ is

$$\mathtt{P}(k+1|k+1) = \mathtt{P}(k+1|k) - \mathtt{K}(k+1)\mathtt{S}(k+1)\mathtt{K}(k+1)^{\mathsf{T}} \tag{2.29}$$

Similar to Kalman Filter, prediction and correction phases exist in EKF. Figure 2.2 shows the EKF cycle starting from initial state mean $\hat{\mathbf{x}}(0|0)$ and covariance $\mathtt{P}(0|0)$ with prediction and correction equations.



prediction
$\hat{\mathbf{x}}(k+1) = f(\hat{\mathbf{x}}(k|k), \mathbf{u}(k))$
$P(k+1|k) = \frac{\partial f}{\partial x}P(k|k)\frac{\partial f}{\partial x}^{\mathsf{T}} + Q(k)$
$S(k+1) = \frac{\partial h}{\partial x}P(k|k)\frac{\partial h}{\partial x}^{\mathsf{T}} + R(k)$
$\hat{\mathbf{x}}(0|0), P(0|0)$
correction
$K(k+1) = P(k+1|k)\frac{\partial h}{\partial x}S(k+1)^{-1}$
$\hat{\mathbf{x}}(k+1|k+1) = \hat{\mathbf{x}}(k+1|k) + K(k+1)\nu(k+1)$
$P(k+1|k+1) = P(k+1|k) - K(k+1)S(k+1)^{-1}K(k+1)^{\mathsf{T}}$

Figure 2.2: EKF cycle with prediction and correction equations

There are some limitations in EKF caused by linearization. First of all, Jacobian expressions in the equations may lead the innovation covariance to become singular and therefore not invertible. Even being not singular, determinant of it that is near to zero may cause the system to diverge. Second, linearization may cause *biased* estimates, that is mean of the state prediction error, $E[\mathbf{x}(k+1) - \hat{\mathbf{x}}(k+1|k)]$ may be non-zero. Finally, if dimensions of the state and measurement vectors are high, computational cost increases. However, with proper selection of state and measurement functions and vectors, risks caused by above limitations can be overwhelmed.

Since the Jacobian matrices are calculated in each step, system is no more time-invariant. Therefore gain and covariances cannot be calculated offline and it may increase the computational time compared to time-invariant linear Kalman filter application.

Similar to Kalman filter, process and measurement noise covariances can be used to adjust the filter characteristics. However, in EKF, for protect consistency of the filter, these two covariances should be small. In other words, both process model and measurement model should be trusted.

## 2.6 Remarks

Kalman Filter and EKF are commonly used for state estimation of linear and non-linear stochastic dynamic system. Some final words on these filters can be stated as follows

- Initial state mean and covariance affect the filter performance. Particularly for EKF, initial covariance should be selected small for a good operation.
- Process and measurement noise covariances can be used to adjust the filter characteristics.
- Predicted measurement and innovation covariance are important for determining measurement range. Particularly, for feature search in vision applications, it can reduce the region to be searched. Feature search will be discussed in Chapter 3.
- Computational cost of the Kalman filter is $O(n^2)$ where $n$ is the bigger dimension of the state and measurement vectors.

For further details and derivations on the subject, see [1, 25, 27, 34]

# Chapter 3

# Stereo Vision

In this chapter, vision as depth perception method will be discussed. Starting from the basic pinhole camera, CCD cameras used mainly robotic applications will be modeled and on these models, techniques to find the distances of the objects in a scene seen by cameras will be discussed. Particularly, depth extraction methods for point features in the scene will be investigated.

Section 3.1 deals with two camera models, namely the basic pinhole camera model and finite projective model will be explained. At the rest of the chapter, finite projective model will be used. In Section 3.2, two camera systems and depth extraction using triangulation method will be explained.

## 3.1 Camera Models

A camera can be defined as a mapping from object space $\mathbb{R}^3$ to image space $\mathbb{R}^2$. In this section two camera models, namely the basic pinhole model, and finite projective model will be discussed. Starting from the basic pinhole model, terms about transformations and camera properties will be included to form finite projective model.

### 3.1.1 The Basic Pinhole Model

Let camera center be the origin of the Euclidian coordinate system and at $z = f$ there exist a plane called *image plane*. Projection of a point $\mathbf{X} = (X, Y, Z)^\mathsf{T}$ on the image plane is the intersection of image plane and the line joining camera center and $\mathbf{X}$. By using triangle similarity, it can be easily shown that point on the image is $(fX/Z, fY/Z)$. This is shown in figure 3.1.

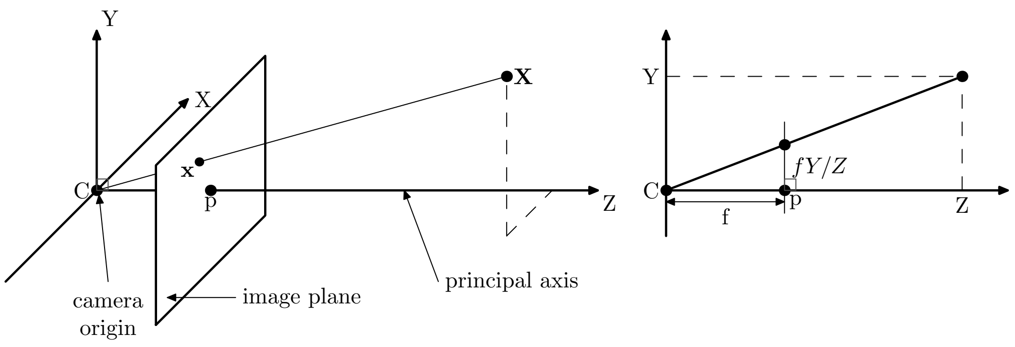


Figure 3.1: The basic pinhole model

The line from the camera center perpendicular to the image plane is the *principal axis*. Intersection point of the principal axis with image plane is the *principal point*. $\mathbf{x}$ is the projection of the $\mathbf{X}$ on the image plane.

### 3.1.2 Finite Projective Camera

In finite projective model, image point $\mathbf{x}$ and object point $\mathbf{X}$ is related by a *projection matrix* $\mathtt{P}$, that is,

$$\mathbf{x} = \mathtt{P}\mathbf{X} \tag{3.1}$$

$\mathbf{x}$ is the homogeneous image point $(x, y, 1)^{\mathsf{T}}$ and $\mathbf{X}$ is homogeneous object point $(X, Y, Z, 1)^{\mathsf{T}}$. $\mathtt{P}$ is the $3 \times 4$ projection matrix which is

$$\mathtt{P} = \mathtt{K}[\mathtt{R}|\mathbf{t}] \tag{3.2}$$

where $\mathtt{K}$ is the $3 \times 4$ *calibration matrix*, $\mathtt{R}$ is the $3 \times 3$ *rotation matrix* and $\mathbf{t}$ is the $3 \times 1$ *translation vector*. $[\mathtt{R}|\mathbf{t}]$ is the *transformation matrix* formed by concatenating translation vector as a column to the end of $\mathtt{R}$.

Calibration matrix $\mathtt{K}$ contains information about the camera properties and commonly defined for CCD cameras. It is in the form

$$\mathtt{K} = \begin{bmatrix} \alpha_x & s & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

$x_0$ and $y_0$ are the coordinates of the principal point on the image. In pinhole model, it is assumed that principal point is at the center of the image plane. However, it may not be the case. For this reason, $x_0$ and $y_0$ are added to the calibration matrix. In ideal case, they are not zero because image coordinate system differs from the camera coordinate system and it's unit is pixels. Origin in the image coordinate system is top-left corner of the image as shown in Figure 3.2. Thus, in ideal case, the principal point is at half of the image width and height.

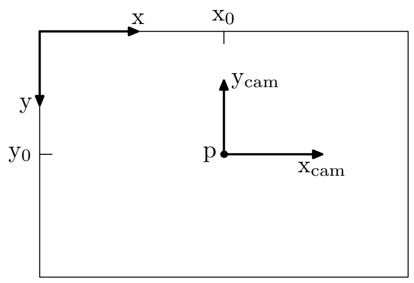


Figure 3.2: Image and Camera coordinate systems

Since image coordinates are in pixels, it is necessary to convert focal length into pixels. If $m_x$ and $m_y$ are number of pixels per unit distance, focal lengths can be written as $\alpha_x = fm_x$ and $\alpha_y = fm_y$. Two focal lengths are defined because in CCD cameras, it is possible that image pixels are not square.

$s$ is the skew parameter to model skewed image elements. It is often zero, however in the case when x and y axis are not perpendicular it is non-zero.

In pinhole camera model, camera center and origin of the coordinate frame in which object point $\mathbf{X}$ is defined. To generalize the model in finite projective model, points in space are defined in world coordinate frame. If object point $\mathbf{X}$ is expressed in terms of world coordinate system, then to find the projection of $\mathbf{X}$, it should be expressed in terms of camera coordinate system. For this reason, position of the camera with respect to world origin should be known. Rotation matrix R and translation vector $\mathbf{t}$ express the transformation from world coordinate frame to camera coordinate frame based on the position information of the camera in the world coordinate frame as illustrated in Figure 3.3.

If world coordinate frame and camera coordinate frame coincides, equation 3.2

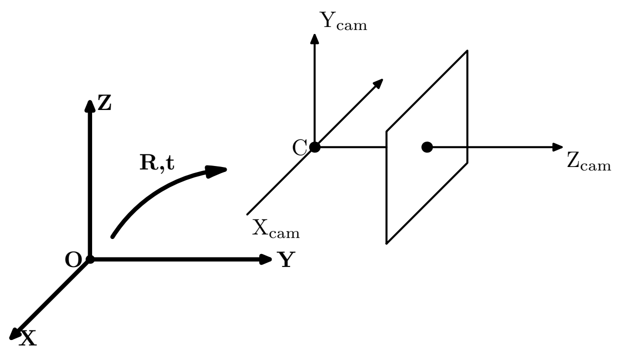


Figure 3.3: Transformation from world coordinate system to camera coordinate system

takes the form

$$\mathtt{P} = \mathtt{K}[\mathtt{I}|\mathbf{0}] \tag{3.4}$$

where $\mathtt{I}$ is the identity matrix and $\mathbf{0}$ is the zero vector.

Pinhole camera model is the special case of the finite camera model of the form

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

## 3.2 Stereo Vision and Depth

Single view of a scene is not sufficient to obtain distances of the objects seen in the scene to the camera. For this reason, multiple cameras seeing the same scene or multiple images taken by a single camera from different positions are required to obtain depth information. As in the human visual system, stereo cameras are the basic system to obtain depth. In Section 3.2.1, stereo cameras having coplanar image planes of the cameras will be explained. The general case in which cameras in arbitrary position and orientation will be discussed in Section 3.2.2. Triangulation technique to compute depth will be discussed in Section 3.2.3.

### 3.2.1 Coplanar Image Planes

Simplest case of the stereo vision is the two identical cameras separated in x direction by a *baseline distance* $b$ such that their image planes are coplanar. An object point $\mathbf{X} = (X, Y, Z)$ is seen on different positions in left and right cameras. Displacement between two image point coordinates is called *disparity* and denoted by $d$. In the case of coplanar image planes, disparity is only in one direction. See Figure 3.4.

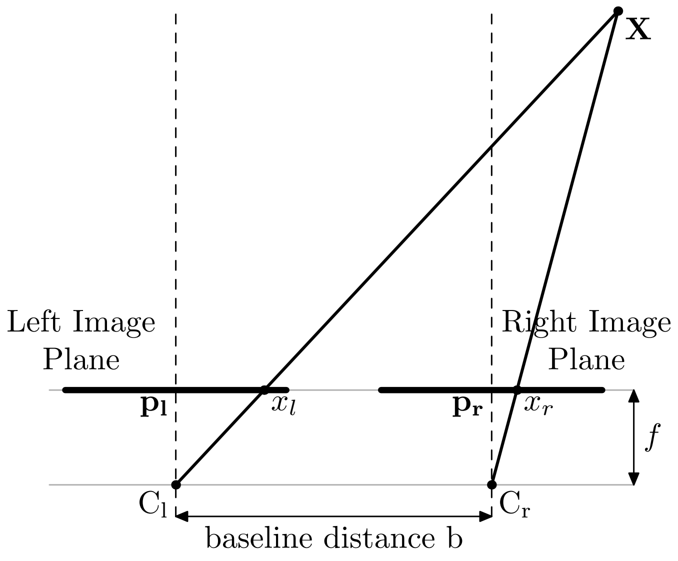


Figure 3.4: If an object can be seen by both cameras, displacement of image points in two cameras is called disparity

Using similar triangles, one can find the following basic stereo vision equations.

$$\begin{aligned} x_l &= f\frac{X+b}{Z} \\ x_r &= f\frac{X}{Z} \\ d &= x_l - x_r \\ X &= b\frac{x_r}{d} \\ Y &= b\frac{(y_l+y_r)/2}{d} \\ Z &= b\frac{f}{d} \end{aligned}$$

where $x_l$ and $x_r$ are the image points on the left and right image planes respectively.[20]

Note that while obtaining these equations, world coordinate is chosen to be right camera center. As it can be seen from the above equations, disparity $d = x_l - x_r$ determines object points and it is inversely proportional to depth $Z$. That is, if an object approaches to the camera system, disparity associated with this object increases. With known baseline distance and focal length, disparity is enough to find the depth in this model.

### 3.2.2 Epipolar Geometry

Epipolar geometry is the geometry of intersection of two image planes and with the plane defined by image points and camera centers. It is commonly used in correspondence search for stereo matching and therefore crucial for depth perception particularly when camera image planes are not coplanar. It depends only on camera internal parameters and relative positions and it is independent of the scene. Three terms, namely epipolar plane, epipole and epipolar line are defined in epipolar geometry.

Suppose that an object point $\mathbf{X}$ has views $\mathbf{x}$ and $\mathbf{x}'$ on two image planes, respectively. As shown in Figure 3.5, object point, its projection points on two images and camera centers lie in the same plane which is called *epipolar plane*. Epipolar plane passes through baseline which is the line joining camera centers.

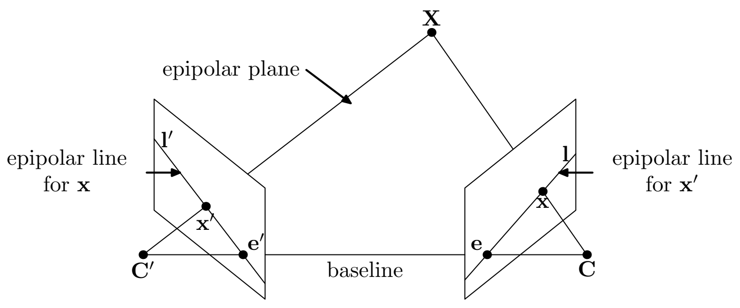


Figure 3.5: Epipolar plane, epipolar lines and epipoles

Intersection point of baseline with image plane is called *epipole*. In addition, it is projection of the camera center of one camera on the other image.

*Epipolar line* is the intersection of epipolar plane with image planes. Since object point, image points and epipolar lines are coplanar, corresponding image point of an image point on the other image plane lies in this line. This property is important for determining search region for stereo matching.

Relation between image points and epipolar lines is established by the *Fundamental Matrix* F. Fundamental matrix relates one point in one image plane to the corresponding epipolar line in the other image plane and two corresponding point of and object is related by the equation.

$$\mathbf{x}'\mathtt{F}\mathbf{x} = 0 \tag{3.5}$$

F matrix is a $3 \times 3$ matrix of rank 2 and thus epipolar line $\mathbf{l}'$ is computed by the equation

$$\mathbf{l}' = \mathtt{F}\mathbf{x}$$

As stated earlier, epipolar geometry depends on only camera internal parameters and relative pose of the cameras. F Matrix encapsulates all these information. Considering two cameras whose camera models are as in equation 3.1 and projection matrices are $\mathtt{P} = \mathtt{K}[\mathtt{I}|\mathbf{0}]$ and $\mathtt{P}' = \mathtt{K}'[\mathtt{R}|\mathbf{t}]$, fundamental matrix F is computed as

$$\mathtt{F} = \mathtt{K}'^{-\mathsf{T}}[\mathbf{t}]_\times \mathtt{R}\mathtt{K}^{-1}$$

Here $[\mathbf{t}]_\times$ denotes skew-symetric matrix which is defined as

$$[\mathbf{t}]_\times = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix}$$

for $\mathbf{t} = (t_x, t_y, t_z)^\mathsf{T}$.

Figure 3.6 shows two nearly coplanar images and epipolar lines of some points in the images. As shown in the figure, corresponding point of an image point in one image lies in the epipolar line calculated by equation 3.5.

In the case of coplanar image planes, baseline intersects image planes at infinity. Therefore, epipoles are at infinity and epipolar lines are parallel. In other words, image rows corresponds to epipolar lines. For corresponding point search, it is enough to search in vicinity of the row with the same $y$ value of the image point in the other image for this case.

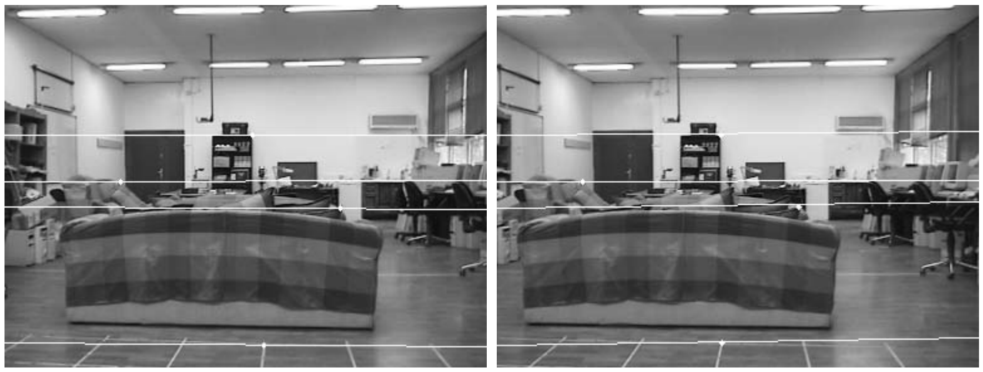


Figure 3.6: Two views taken from a nearly parallel cameras. Corresponding point of an image point in one view lies on the corresponding epipolar line. Note that epipolar lines are almost parallel.

### 3.2.3 Triangulation

Single view of a scene is not sufficient to find distances of objects in the scene to the camera. Object may be anywhere on the ray starting from camera center and passing through the image point of the object on the image plane. In other words, on the ray, every object point is corresponds to the same point on the image plane. Thus, minimum two views of the same scene is necessary. By this way, intersection point of the two rays gives the position of object in 3D. This procedure is known as *triangulation* or *back-projection*. Linear triangulation method will be described in this section to find object point.

Let two image points $\mathbf{x} = \mathtt{P}\mathbf{X}$ and $\mathbf{x}' = \mathtt{P}'\mathbf{X}$ of the same object point $\mathbf{X}$. These equations can be combined into a linear equation $\mathtt{A}\mathbf{X} = 0$. To obtain $\mathtt{A}$, first, three equations for each image point is found by cross product. Two of these equations are linearly independent. For the first image $\mathbf{x} \times (\mathtt{P}\mathbf{X})$ gives three equations, which are

$$\begin{aligned} x(\mathbf{p}^{3\mathsf{T}}\mathbf{X}) - (\mathbf{p}^{1\mathsf{T}}\mathbf{X}) &= 0 \\ y((\mathbf{p}^{3\mathsf{T}}\mathbf{X}) - (\mathbf{p}^{2\mathsf{T}}\mathbf{X}) &= 0 \\ x((\mathbf{p}^{2\mathsf{T}}\mathbf{X}) - y(\mathbf{p}^{1\mathsf{T}}\mathbf{X}) &= 0 \end{aligned}$$

where $\mathbf{p}^{i\mathsf{T}}$ are the rows of $\mathtt{P}$.

By selecting two of the three equations from each image,linear equation, $\mathtt{A}\mathbf{X} = 0$

is formed where $\mathtt{A}$ is of the form

$$\mathtt{A} = \begin{bmatrix} x\mathbf{p}^{3\mathsf{T}} - \mathbf{p}^{1\mathsf{T}} \\ y\mathbf{p}^{3\mathsf{T}} - \mathbf{p}^{2\mathsf{T}} \\ x'\mathbf{p}'^{3\mathsf{T}} - \mathbf{p}'^{1\mathsf{T}} \\ y'\mathbf{p}'^{3\mathsf{T}} - \mathbf{p}'^{2\mathsf{T}} \end{bmatrix}.$$

In most cases, because of noise and some distortions in the image, backprojected rays may not intersect. The optimal solution for these cases in least square sense is the last column of $\mathtt{V}$, where $\mathtt{A} = \mathtt{UDV}^{\mathsf{T}}$ is the Singular Value Decomposition(SVD) of $\mathtt{A}$.[14]

## 3.3 Remarks

Stereo vision as depth perception method is commonly used in many robotic applications including robot localization. Finding depth of features by vision is useful in the sense that other properties of the feature can be found by other vision tools at the same time without any need for data association. Main problem in vision based depth perception applications is finding corresponding pair associated with the object point. However, epipolar geometry provides useful constraints on the possible locations of the image points and simplifies the search procedure for the corresponding points of the features. For further details on topics discussed in this chapter, see [14, 20].

# Chapter 4

# Robot System and Models

## 4.1 System Composition

Our robot system is composed of three main components. These are

- The robot vehicle
- The stereo camera system
- The processing and control unit

Robot vehicle and cameras are mobile units of the system where processing and control unit is a stationary PC. Mobile part of the system which is the robot vehicle and stereo camera system mounted on the robot vehicle is shown in Figure 4.1. The processing and control unit is the PC which takes and processes the data coming from camera system and controls the robot. Following three subsections will describe the system components in detail.

### 4.1.1 The Processing and Control Unit

The Processing and Control Unit(PCU) is a stationary PC which runs Windows XP as the operating system and has the system properties which are shown in Table 4.1.

Some important tasks such as image processing to take measurements from the stereo camera system requires high capacity processor and high memory. To work near real-time, Pentium IV processor with 2.4 GHZ speed and 512 MB RAM is convenient. In addition architecture of the processor is optimized for the multimedia

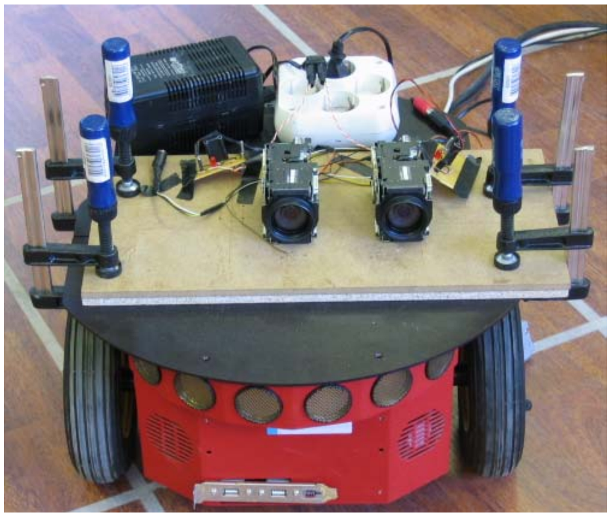


Figure 4.1: Mobile Part of the Robot System

applications and it speeds up the processing of multimedia such as image, video and sound. Secondary storage device, harddisk is enough to store temporary and permanent data and fast enough to permit fast operation. Although graphics card does not affect directly processing of the data in this thesis, it helps visualization of the data in the monitor.

Camera system which will be described in Section 4.1.3 gives analog output. To process data obtained from the cameras, analog data should be converted into digital data. Matrox Meteor II frame grabber mounted on the PCI slot of the PC, samples

Table 4.1: PC system properties

| | |
|---|---|
| Processor | Intel Pentium IV 2.4GHZ |
| Memory | 512 MB DDR RAM |
| Harddisk | 60 GB 7200 RPM |
| Graphics Card | 64 MB GeForce 4 MX440 |
| Frame Grabber | Matrox Meteor II Standart |

this analog output and produces digital data. Since there are two cameras in the vision system, two input ports are required in the frame grabber. 44 pin input slot of the frame grabber supports up to 12 video input and sufficient for two camera system. However, it multiplexes inputs and samples only one input at a time. This limits the operation of the system. This limitation and consequences will be discussed later.

Operating System (OS) of the PC is Windows XP. For three reasons this OS was chosen. First of all, support of our Matrox frame grabber is only for the Windows OS and this forced us to use Windows as the operating system. Secondly, Windows XP provides tools for multimedia processing and support large amount of hardware devices. Finally, most software related to the works in this thesis are written for the Windows OS.

C++ is the programming language and Visual C++.NET is the programming environment for this thesis. Interaction with robot vehicle, and frame grabber is established by two C++ libraries ARIA and MIL. ARIA is used for control of the robot vehicle and taking data about location of the robot obtained by the shaft encoders in the robot wheels. More information about the robot vehicle and ARIA will be given in Section 4.1.2.

Matrox Imaging Library(MIL) is a commercial library and sold separately from the frame grabber. MIL-lite, a limited version of MIL, is used in this thesis. MIL-lite includes only commands required for image and video acquisition and does not include any video and image processing commands which MIL includes. For image and video processing and also for mathematical calculations an open-source and free library, Intel Open Source Computer Vision Library (OpenCV) is used. This library includes many functions from corner and edge detection to complex matrix calculations. In addition, a wrapper class which converts data taken by the MIL-lite library to OpenCV structures is used. This class is written in our laboratory as part of another thesis[29].

### 4.1.2 The Robot Vehicle

Robot vehicle is a Pioneer 2 DX8 model, differential 3-wheeled robot base manufactured by ActivMedia Robotics Company. Two bigger wheels are at the front and are driven by two servo motors. Third wheel is smaller and it is at the back. This wheel is free-running and used for stabilization of the vehicle. All movements are established by the independent movement of the front two wheels. That is, wheel speeds and direction determines the movement of the vehicle. For example, for forward motion, wheel speeds of the two wheels are same, and they turn in the same direction. For turn around its own axis, same wheel speed but opposite turn directions are required. Movement of the vehicle is not restricted to this type of motion. Wheel structure gives 3 degrees of freedom for the motion of the vehicle. These are translation in two dimensions and rotation around the axis which is perpendicular to the plane of translation. Center of the robot vehicle is defined as the midpoint between the two front wheels. This point will be used to place the camera system on the vehicle.

Shaft encoders are placed to read the wheel revolutions of the vehicle. Readings are used to locate the robot according to displacement from the starting point. This way of locating robot is called odometry. Odometry is error-prone because it cannot handle the cases like wheel slippage or stall.

Communication with the vehicle is done by a RS-232 serial cable by the "pass through" principle. Three pins which are RX, TX and Ground are sufficient for communication and in our robot system these 3 pins are used.

A Hitachi microcontroller and controller board controls the servo motors, communication and odometry devices. Controller board, motors and other devices in the vehicle are feed by 3 rechargeable 12 volt batteries.

ARIA library provides a complete control over the vehicle. It includes commands not only for movement but also for state reflection. State reflection is the term used for getting information about the vehicle at that moment. Odometry readings, wheel speeds, battery voltages, are some of these information. ARIA converts the commands given in the program code to low level commands in the serial communication protocol. These low level commands are processed by the controller board and control of the vehicle is established.

On the robot base, at the front, there are sonar sensors used for depth and proximity sensing. In many robot localization algorithms, sonar sensors are used for depth perception. However, in this thesis, only vision based depth perception is used because sonar sensors mounted on our robot vehicle is designed for proximity sensing and therefore depth range is small and uncertainty of the depth values are high.

### 4.1.3 The Stereo Camera System

Stereo Camera System is developed in our laboratory as part of another thesis[29]. It consists of two identical Sony color board camera, composite video cables and a board on which cameras are fixed. The system is shown in Figure 4.2. Technical specifications of the cameras are given in Table 4.2. Cameras are fixed, that is, spacing and relative rotation of the cameras are constant. By this way, cameras are calibrated manually before mounted on the robot and during thesis work, no additional calibration is required.

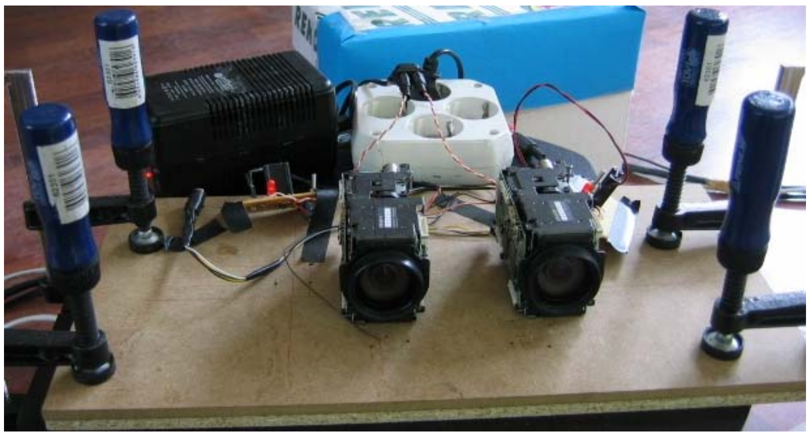


Figure 4.2: Camera system is developed in the laboratory and consists of two Sony board cameras.

Cameras give analog interlaced output which are transfered on composite video cables. Resolution of the cameras are $768 \times 576$ in PAL standart. $768 \times 576$ resolution is too big for real-time image processing and therefore images taken from the cameras are scaled down to $384 \times 288$ by taking only even fields the interlaced frame and scaling down y axis by two. Taking only even fields also prevents jitter caused by interlacing and provides clean view of the scene. Although cameras are color cameras, color components are not used and grayscale output is taken from the cameras and all processing operations are done on these grayscale images. Images are taken in wide angle mode, that is, no optic or digital zoom is used and calibration of the cameras done according to wide angle mode of the cameras.

Right camera is chosen as the reference camera. Camera center of the right

Table 4.2: Camera Technical Specifications

| | **Left** | **Right** |
|---|---|---|
| **Model** | Sony FCB-IX47AP | Sony FCB-IX47AP |
| **Serial No** | 1001511 | 1001512 |
| **Resolution** | $768 \times 576$ | $768 \times 576$ |
| **Optical Zoom** | $18\times$ | $18\times$ |
| **Digital Zoom** | $10\times$ | $10\times$ |
| **Focal length** | 3.1-31 mm | 3.1-31 mm |

camera is chosen as the origin of the camera coordinate frame. In addition, feature detection and tracking operations are performed on the images taken from the right camera. Left camera is the complementary camera to provide depth information of the features detected on the right camera.

Camera system is mounted on the robot vehicle such that robot's turn axis and camera center of the reference camera coincides. By this way, calculations are simplified and any translation of cameras during rotation of the robot around its own axis is avoided.

## 4.2 Mathematical Models

### 4.2.1 Vehicle Model

This thesis deals with robots that operate in indoor environments and it is assumed that robot vehicle moves on a ground plane. A world coordinate frame is defined such that its $x$ and $z$ axis lie in the ground plane that robot vehicle moves on. Its $y$ axis points towards vertically upwards. Robot's position in world coordinate is represented as $(x, z, \theta)^\mathsf{T}$ where $x$ and $z$ are the position in world coordinate frame and $\theta$ is the orientation of the robot relative to the world z axis. The point that is chosen as the reference on the robot for position is turn axis of the robot where reference camera is also placed. Figure 4.3 shows robot and world coordinate frames and representation of the robot position.

As described in section 4.1.2, motion of our robot vehicle is established by different

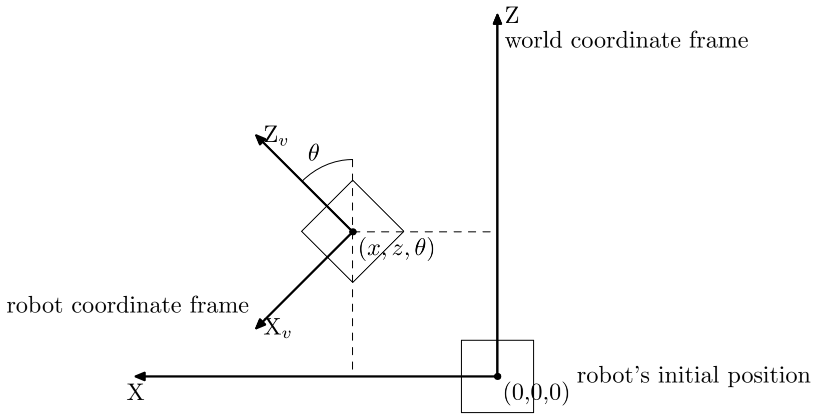


Figure 4.3: Representation of the robot's location in the world coordinate frame

wheel speed of the two front wheels. In some studies [11, 30], using same or similar robots, wheel speeds are given as input to control the robot. By this approach, it is possible to make the robot vehicle do arbitrary motion in the $x - z$ plane. However, for two reasons, a different approach similar to the one in [32] is used in this thesis.

First reason is, control with wheel speeds cannot handle displacement during acceleration and deceleration. To include acceleration in the model is difficult and makes the calculations complicated.

Second reason is more specific to our system. As stated earlier, frame grabber that we used, multiplexes input ports and takes only one input at a time. To get stereo view of the scene, channel swithcing is required. However, grabbing one frame and swithcing channel take about 110 ms in the PC system which is too long for grabbing one frame from each camera while the robot vehicle moves. After first frame from the right camera is taken, because of the delay caused by channel switching, frame from the left camera is taken from a different location caused by the displacement of the vehicle during the delay. For example, if the robot moves at speed 200 mm/sec in the forward direction, an extra 2 cm translation occurs between two camera views. It becomes more complicated when the vehicle is also turning. It can be handled by calculating displacement and including in the depth calculations. However, this makes camera model depend on the vehicle model which is not desirable.

For these reasons, robot vehicle motion is constrained to two types of motion, namely *move forward* and *turn*. Input vector $(d, \phi)^\mathsf{T}$ is defined to control the vehicle where $d$ represents the displacement in the forward direction and $\phi$ is the turn angle in radians. ARIA library includes commands to move forward in the desired $d$ and to turn in the desired $\phi$.

With known input $\mathbf{u} = (d(k), \phi(k))^\mathsf{T}$, robot's location can be calculated as

$$x(k+1) = x(k) + d(k)\sin\theta(k) \tag{4.1}$$

$$z(k+1) = z(k) + d(k)\cos\theta(k) \tag{4.2}$$

$$\theta(k+1) = \theta(k) + \phi(k) \tag{4.3}$$

If the above equations perfectly modeled the motion of the robot, there would be no need for the estimation of the robot location. However, there is an uncertainty in the motion and this should be included in the model. Error in the motion estimate can be modeled as a Gaussian with zero-mean and covariance $\mathtt{Q}(k)$. $\mathtt{Q}(k)$ can be calculated by using estimated robot location.

By defining estimated robot location as

$$\mathbf{f}_v = \begin{pmatrix} x(k+1) \\ z(k+1) \\ \theta(k+1) \end{pmatrix}$$

$\mathtt{Q}(k)$ is calculated by using first-order transfer of uncertainty and given as

$$\mathtt{Q}(k) = \frac{\partial \mathbf{f_v}}{\partial \mathbf{u}} \mathtt{U}(k) \frac{\partial \mathbf{f_v}}{\partial \mathbf{u}}^\mathsf{T}$$

where $\frac{\partial \mathbf{f_v}}{\partial \mathbf{u}}$ is the Jacobian of the $\mathbf{f}_v$ with respect to $\mathbf{u}$ and $\mathtt{U}(k)$ is the covariance matrix of the $\mathbf{u}$:

$$\mathtt{U}(k) = \begin{bmatrix} {\sigma_d}^2 & 0 \\ 0 & {\sigma_\phi}^2 \end{bmatrix} \quad , \quad \frac{\partial \mathbf{f_v}}{\partial \mathbf{u}} = \begin{bmatrix} \frac{\partial x(k+1)}{\partial d} & \frac{\partial x(k+1)}{\partial \phi} \\ \frac{\partial z(k+1)}{\partial d} & \frac{\partial z(k+1)}{\partial \phi} \\ \frac{\partial \theta(k+1)}{\partial d} & \frac{\partial \theta(k+1)}{\partial \phi} \end{bmatrix} .$$

Standart deviation for the turn, $\sigma_\phi$ is about $0.05\phi$. For $d$, standart deviation $\sigma_d$ is about $0.05d$.

Later, in the Chapters 6 and 7 uncertainty $\mathtt{Q}(k)$ will be used in EKF estimation algorithm and will be important in determining the characteristics of the filter. Although there are many factors affecting the motion of the vehicle, this uncertainty model is sufficient to handle all the errors and noise that exist in the system. In addition, adding other errors in the model increases the complexity without giving significant increase in the performance. This uncertainty is introduced to the model as an additive term without any gain factor. Details about the model with noise added, will be described in Chapters 6 and 7.

### 4.2.2 Camera Model

In this thesis, finite projective camera model which is explained in chapter 3, is used to model the camera system. In this section,values of the internal camera matrix, rotation Matrix and translation vector in the Equations 3.1,3.2 and 3.3 will be given and discussed.

As described in section 4.1.3, two same model Sony board cameras are used in the vision system. Although they seem to be identical, internal camera parameters differ and this difference should be included in the models. Table 4.3 shows internal camera parameters of each camera[29].

Table 4.3: Internal and External Camera Parameters of each camera

| | **Left** | **Right** |
|---|---|---|
| $\mathbf{f_u}$ | 454.189 | 452.501 |
| $\mathbf{f_v}$ | -455.073 | -453.15 |
| $\mathbf{s}$ | 0.015 | 0.174 |
| $\mathbf{u_0}$ | 181.604 | 180.565 |
| $\mathbf{v_0}$ | 151.733 | 147.697 |
| $t_x$ | -95.8369 | 0 |
| $t_y$ | 1.5005 | 0 |
| $t_z$ | -1.4762 | 0 |

Note that$f_v$ is negative to make camera and robot coordinate frames coincide. In addition there is a shift in the central points of the cameras and they are different from the midpoint (192,144) of the scaled images. (Scaled image size is $384 \times 288$).

Placement of the cameras also differ from the simple parallel camera placement. Although, it was aimed to place them parallel, in laboratory conditions, some relative rotation and extra translation could not be avoided. As taking right camera as reference, projection matrices for each camera are

$$\mathtt{P}' = \mathtt{K}' \left[ \begin{array}{ccc|c} 0.9992 & 0.0215 & -0.0330 & -95.8369 \\ -0.0214 & 0.9997 & 0.0032 & 1.5005 \\ 0.0331 & -0.0025 & 0.9995 & -1.4762 \end{array} \right] \quad , \quad \mathtt{P} = \mathtt{K} \left[ \begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array} \right]$$

where $\mathtt{P}', \mathtt{K}'$ are projection and internal camera matrices for left camera and $\mathtt{P}, \mathtt{K}$ are projection and internal camera matrices for the right camera. Since right camera is chosen to be reference, rotation matrix is identity matrix and translation vector is the zero vector.

Depth calculations are performed by using these parameters and matrices in the equations given in Chapter 3. Important point in these calculations is to find the corresponding image point pair which describes the 3D object. Chapter 5 will describe feature detection, finding corresponding point of the feature and depth perception.

# Chapter 5

# Landmark Detection and Matching

## 5.1 Artificial Landmarks

In robot localization, generally distances of objects to the robot is measured and inference about the location of the robot in the environment is reached by using this distance information. However, measurement devices find the positions of the objects in the robot coordinate system and to find the position of the robot in the world coordinate system, positions of the objects in the world coordinate system should be known. Generally for every object in the scene to give the position information is not feasible and increases the computational time. Instead, artificial landmarks which are easily identifiable and whose positions are known are used to find the position of the robot.

In literature, many types of artificial landmarks exist. Barcodes, infrared leds, colored cylinders are some of them. In this thesis, we used a new artificial landmark type which is *coded cylinder*. Section 5.1.1 gives information about the properties of the coded cylinders that we used. In Sections 5.1.2 and 5.1.3, methods to detect and recognize each landmark is explained.

### 5.1.1 Landmark Properties

For a landmark to be useful, it should be identifiable from different point of views and scales. Planar landmarks are usually affected from affine motion and seen different from different point of views. Our artificial landmark type and the technique we use to identify the landmarks are invariant to both affine transformation and scaling for a large range.

We use a cylinder block whose radii is 10 cm and height is 12.5 cm. To the bottom and top of the side of the cylinder, black stripes which have 1.5 cm width are sticked horizontally. These stripes are used to distinguish the landmark from other objects in the scene and they are called limiting stripes. Middle side of the cylinder block is used to recognize each landmark. To the middle side, black stripes are sticked horizontally according to the number of the landmark. These stripes are called *coding stripes.* See Figure 5.1. Stripes are sticked horizontally because robot do not move in the y-axis (axis which is perpendicular to the ground floor) and thus, width and distances of the stripes do not change while the robot moves.

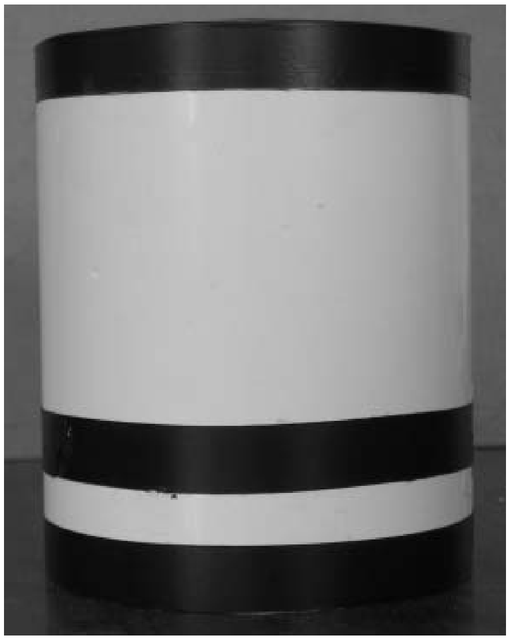


Figure 5.1: One of the artificial landmark whose number is 1

A 3-bit binary coding is done on the cylinders. That is, according to the number of the landmark, stripes are placed to the middle side of the cylinder such that topmost stripe is the most significant bit and the most bottom stripe is the least significant bit. For example, to represent number 4 a black stripe close to the top limiting stripe is sticked. Figure 5.2 shows the general properties of the landmark that we use.

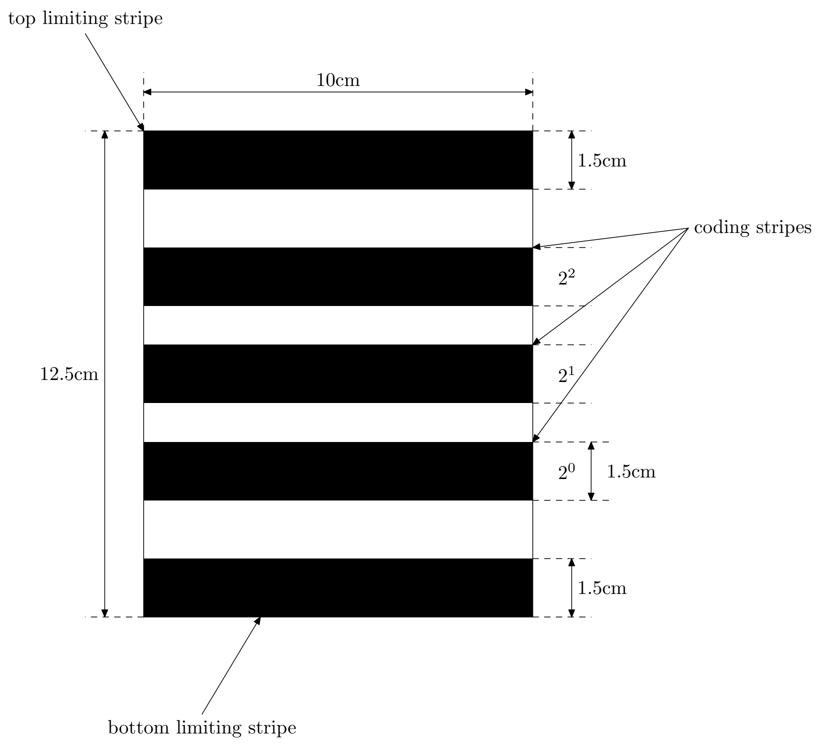


Figure 5.2: Landmark Structure and Properties

A cylinder block is chosen as the landmark because it is seen same from all point of views. In addition, because the robot does not move in the y-axis, there is no need to consider the top of the cylinder and this fact makes the cylinder block more convenient for indoor robot movements. It seems difficult to find the center point of the cylinder by looking from outside because of the smooth surface but, by using the algorithm that is explained in the next section, it is not a problem. And experiments show that the artificial landmark that we use is convenient for indoor robot localization. Sections 5.1.2 and 5.1.3 explain the detection and recognition of the coded cylinders that we use.

### 5.1.2 Detection of The Landmarks

As stated earlier, two limiting stripes are sticked to ends of the cylinders to distinguish the landmarks from the other objects in the scene. In detection, this limiting stripes are used. Color of the main body of the cylinders are white while stripes are black. So, there exist transitions from white to black and black to white in the cylinders. Since stripes are placed horizontally, these transitions occur as a line. By detecting these transition lines, detection of the landmarks can be reached.

To detect the transition lines, two image blocks, each has 12 pixel width and 4 pixel height are formed. One image block is for black to white transition and one block is for white to black transition. Thus, for the black to white transition, the image block is formed such that top half of the image block is white and bottom half of the image block is black. For the white to black transition, it is vice versa. Figure 5.3 shows the formed image blocks.

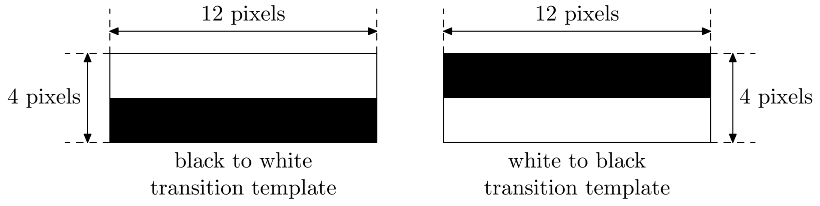

12 pixels
12 pixels
4 pixels
4 pixels
black to white transition template
white to black transition template

Figure 5.3: Image patches which are used to detect transitions

After image block are formed. Similarity of the regions of the image in which we search the landmark, with the image block that we formed should be found. Similarity criterion is the Normalized Sum of Square Difference(NSSD) which is defined as

$$R(x,y) = \frac{\sum\limits_{x',y'} (T(x',y') - I(x+x',y+y'))^2}{\sqrt{\sum\limits_{x',y'} T(x',y')^2 \sum\limits_{x',y'} I(x+x',y+y')^2}} \tag{5.1}$$

where $T(x,y)$ is the image block and $I(x,y)$ is the image in which we search for the image block, $x', y'$ is the coordinate of the pixel in the image block and $x, y$ is the coordinate of the pixel in the main image. This similarity criterion works as follows.

For each pixel in the main image, an image patch of the same size with the image block around the pixel is formed. With each pixel in the image block, pixel with same coordinate in the formed image patch from the main image is compared by taking difference. These differences are summed and divided to the normalizing factor which is given in the Equation 5.1. For a perfect match, the value of $R(x,y)$ is 0. However, for the most cases, zero value cannot be reached. For this thesis, threshold for the similarity is chosen to be 0.2 and values bigger than this value are eliminated.

After each pixel in the main image is assigned a value for the similarity with the image blocks, starting from the bottom of the image, for each pixel coordinate that has the similarity with the black to white transition image block which corresponds to the bottom limiting stripe and has a value smaller than the threshold, a search for the above pixels on the same column is initiated. This search is to find the closest white to black transition line. White to black transition line is searched similar to the black to white transition and similarity values smaller than the threshold is chosen as the line point. If a white to black transition point is found above the black to white transition point, one stripe is assumed to be found and above the white to black transition point a new search for the black to transition point is initiated. This process continues until, no white to black or black to white transition point is found anymore above the found ones. Figure 5.4 shows the transition points and search for the other transition points on the same vertical line.

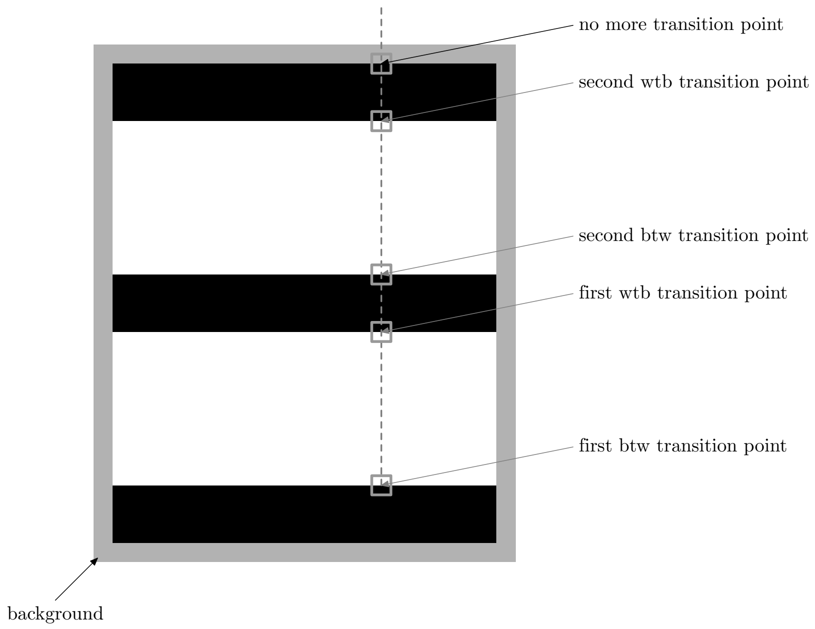

no more transition point
second wtb transition point
second btw transition point
first wtb transition point
first btw transition point
background

Figure 5.4: After a black to white transition is found, a white to black transition is searched above and vice versa. btw and wtb denote black to white and white to black respectively.

By counting the number of stripes and calculating the distance of each black to white transition point to the white to black transition point on the same vertical line, candidate landmark types are determined. Recognition of the landmark type will be explained in Section 5.1.3.

For each pixel in the main image, search is run and thus, many points may give similarity value smaller than the threshold. If there is a landmark in the scene, points in the landmark give high similarity and points are close to each other. After candidate landmark type is determined for each candidate point, points with the same landmark type are placed to a binary image in the same coordinates and on the formed binary image, connected component analysis is run.

Connected Component Analysis(CCA) is a labeling method to find the pixel groups which are called contours. In other words, CCA finds the contours in which pixels are neighbor to each other. Intel OpenCV library has functions to perform CCA and give information about the contours. After contours are labeled, area and orientation of each contour can be calculated.

Using CCA, contours in the binary image in which candidate landmark points are located are found and area and center points of the each contour is calculated. Contour having the biggest area is chosen and center point of it is accepted as the location of the landmark in the image.

There may be more than one landmark type for the same landmark in the scene. To decide on the correct type, if two locations assigned to the landmark types are close to each other, landmark type having the bigger contour area is chosen as the true landmark type for that landmark.

Above analysis gives course position of the landmark in the scene. However to find the exact position of the landmark, middle point of the cylinder should be found. In the cylinder, it has a smooth surface and there is no information which gives idea about the middle point of the cylinder. For this reason borders of the cylinder is found and midpoint between the borders are accepted as the center point.

To find the borders of the cylinder, edge detection could be solution but, generally, detected edges may not be distinguishable because of the other edges close to these edges. Instead, a method similar to the edge detection by using the similarity criterion is used to find the borders of the cylinder.

Coarse position of the landmark is known and found coordinate is in the landmark. In addition, height of the landmark in pixels can be found. Using these information, a thin slice around the found position which has a width of 4 and height which is equal to the found height of the landmark is formed. Using NSSD, around the found position this slice is searched along a line horizontally and NSSD for each pixel is calculated. On the borders, the biggest difference between two adjacent values of the NSSD occurs and this point is accepted as the border point. This operation is performed for both borders and midpoint of the two border point is chosen as the center point of the cylinder. Figure 5.5 shows the process of finding the center point and the related terms.

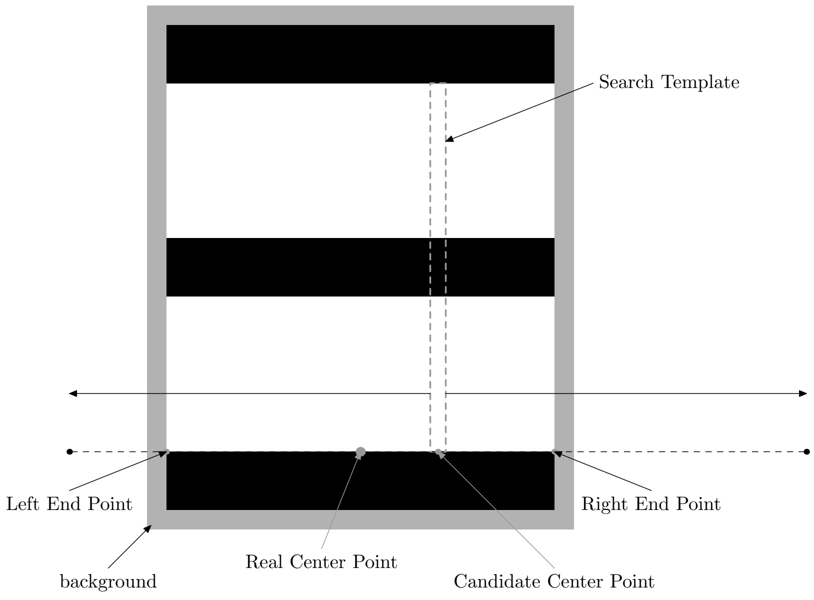


Figure 5.5: Process of Finding The Center Point of The Landmark

In this thesis, positions of the landmarks with respect to the camera system should be calculated and for this reason, image points of the landmark at both left and right camera should be found. Above detection algorithm is run on the reference image which is grabbed from the right camera. To find the corresponding point on the left camera, a similarity search is run on the pixels which are in the vicinity of the epipolar line on the left image. As stated in the Section 4.2.2 cameras are placed parallel and thus, epipolar line is the image row which is has the same $y$ value with the found coordinate of the landmark. An image block which has the same width and height with the found landmark is found and NSSD for each pixel in the vicinity of the epipolar line is calculated. Best match which has the smallest NSSD value is chosen as the match. Generally, search result does not give the center point of the landmark in the left image and because of that, search to find the borders and consequently the center point of the landmark is also run for the left image.

After two points in the left and right camera are found which corresponds to the landmark, using the triangulation method explained in the Section 3.2.3 is applied and position of the landmark in the camera coordinate system is found.

### 5.1.3 Recognition of The Landmarks

As explained before, after candidate points are found in the image, according to the number of stripes and distances of each stripe to another determines the type of the landmark. The main aim in recognizing the landmark is to find the 3-bit binary code representing the landmark.

Each candidate landmark point has the information about the number of stripes above the point and distances of stripes to each other using this information type of the landmark can be determined.

Table 5.1 shows the type of the landmark according to the number of the stripes and comparison of the distances. In this table, distances play a key role in determining the type when number of stripes are the same. There may be maximum four distance terms. If no stripe is found, there is one distance term $d_1$ which shows the distance between two limiting stripes. If one stripe is found there are two distance terms $d_1$ showing the distance between bottom limiting stripe and the found coding stripe and $d_2$ showing the distance between the top limiting stripe and the found coding stripe. If two coding stripe is found, there are three distance terms which are denoted similarly. If there are three coding stripes, although there are four distance terms, they are not important because there is only one combination and this case represents the number 7. Figure 5.6 shows the naming of each distance term for different cases.

Comparison of the distance values in Table 5.1 is pixelwise. For the case, that the number of the coding stripes is 1, $d_1$ is not accepted as equal to the $d_2$ when number of pixels are same, because they may not be equal although they are very close to each other. Rather, $d_1$ is accepted to equal to $d_2$ when they are close to each other.

Table 5.1: Recognition Table showing the type of the landmark according to the number of coding stripes and distance combinations

| **# of stripes** | **Distance Comparison** | **Landmark #** |
|---|---|---|
| 0 | - | 0 |
| 1 | $d_1 < d_2$ | 1 |
|  | $d_1 \cong d_2$ | 2 |
|  | $d_1 > d_2$ | 4 |
| 2 | $d_1 < d_2 \wedge d_3 < d_2$ | 5 |
|  | $d_2 < d_1 \wedge d_3 < d_1$ | 6 |
|  | $d_1 < d_3 \wedge d_2 < d_3$ | 3 |
| 3 | - | 7 |

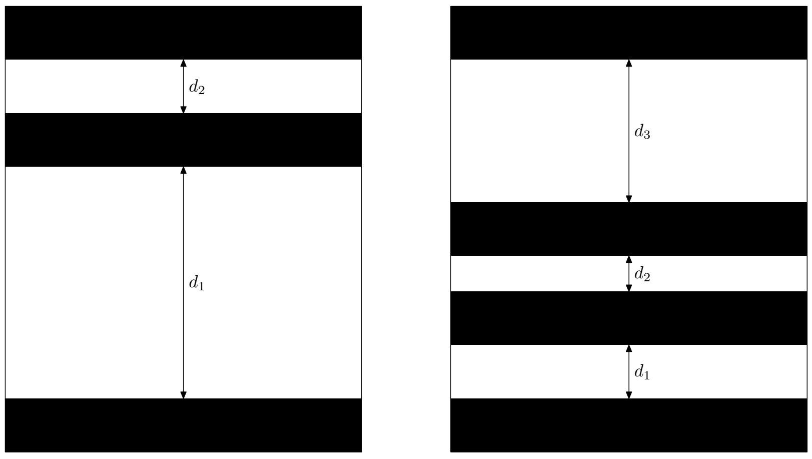


Figure 5.6: Distance names are given from bottom to top

## 5.2 Natural Landmarks

In this section, natural landmark detection and matching methods that are used in this thesis will be described. Properties of the selected landmarks are explained in Section 5.2.1. In Sections 5.2.2 and 5.2.3, the landmark detection technique we used in the thesis and correlation based matching will be described.

### 5.2.1 Landmark Properties

Natural landmarks are features which are determined by the system and detected according to some criteria. This requires reliable, repeatable and measurable features. In this thesis, discrete point features are used as natural landmarks which corresponds to some corners in the images taken from the cameras. Corners and detection will be described in Section 5.2.2.

Since the robot moves, features should be seen for a long time with the same identifiability. These features are natural landmarks for the robot to aid it for finding its location. Thus, reliability and repeatability is essential properties for these features.

To simplify the calculations and because of the restrictions of the algorithm that we used, the features are assumed to be stationary. Image patches are used to represent these features and these image patches are used to match and track the features. In this thesis, $7 \times 7$ patches are used. These dimensions are commonly used and give good results for our system. Smaller patches increases the probability of mismatches, while larger patches increase the computational time and introduce offset from the actual position of the feature.

Number of features to be processed in the system affects the performance. Few features decreases the computational time but erroneous measurements directly affect the position estimates. Too many features increases the computational time quadratically due to the covariance calculations which will be described in the next chapter. For this reason a moderate number of features are used and parameters are selected to establish this.

### 5.2.2 Detection of Landmarks

In this thesis, strong corners in the images are selected to be image features. By finding corresponding image points in the other image, depth information associated with these image features are found. For corner detection the algorithm introduced by Shi and Tomasi [33] is used. This algorithm is similar to Harris Corner Detection [13] but have some improvements on the cornerness.

The algorithm calculates an operator which is defined as

$$\mathtt{C} = \begin{bmatrix} \Sigma I_x I_x & \Sigma I_x I_y \\ \Sigma I_y I_y & \Sigma I_y I_y \end{bmatrix}$$

where $I_x$ and $I_x$ are horizontal and vertical gradients of the image $I$. Summation is done in the patch around the point of interest. If the *smaller* one of two eigenvalues $\lambda_1$ and $\lambda_2$ is large, this point is marked as corner. If only one eigenvalue is large than this point is most probably an edge or even smooth surface. Figure 5.7 shows some image patches. First and second image pathces corresponds to edges and one of the eigenvalues of `C` operator for these image patches is zero. Third image patch is a corner and at both direction it shows large gradient and both eigenvalues are large. Fourth image patch is smooth surface and both of the eigenvalues are zero. For all image pixels, this operator is calculated and smaller eigenvalues are stored. A threshold value is chosen as $\alpha \max(eigenvalues)$ and stored eigenvalues whose values are smaller than this value are eliminated. $\alpha$ is the quality level and it is a parameter to determine number and quality of corners in the image. In addition, if euclidian distance between two corner coordinates is smaller than a threshold value, the weaker corner is eliminated. In this thesis, threshold distance is chosen as 10. This threshold works well for our system. A large distance, reduces the number of corners while small distances increases the probability of mismatches. By this way, well spreaded strong corners are found in overall image. A final step for our system is to eliminate corners which are too close to the borders of the image because these corners become most probably unseen after a few steps and thus are not repeatable. OpenCV library described in Section 4.1.1 has a function which is an implementation of the algorithm and in this thesis, this function is used.

This feature detection algorithm is applied to both left and right images at each step. Matching and tracking is performed on the detected corners with region correlation method that will be described in the next section. Rather than searching for image patches in all image for correlation, search is performed among only detected corners. Because, strong and reliable features have strong corners that do not vanish easily and a search among corners eliminates weak features in a few steps.

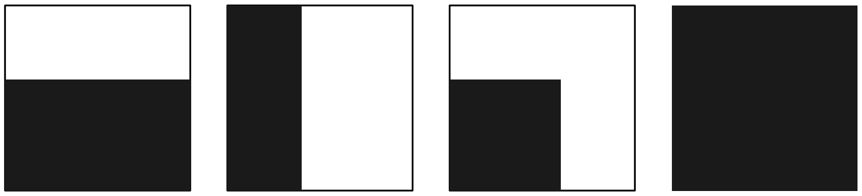


Figure 5.7: First and second image patches show large gradient in one direction but zero in the other and have zero eigenvalue. Third image patch has large gradient in both direction and both eigenvalues are large. Fourth image patch is smooth and both eigenvalues are zero.

### 5.2.3 Searching for and Matching Landmarks

In this thesis, each feature is identified by an image patch (also called template) around the image feature point. To search for and matching of a feature, this image patch is used. After corners are detected at each image, for each corner detected on the reference image, an image patch around the corner point is stored in the system. Image patch is a $7 \times 7$ block taken from the image such that feature point is in the center of the image block. After image patches of each corner is found, to find the corresponding pair of the corner point in the left image and consequitive frames, a similarity criterion is used. This criterion is the Normalized Cross Correlation(NCC) which is defined as

$$R(x,y) = \frac{\sum\limits_{x',y'} T(x',y') I(x+x', y+y')}{\sqrt{\sum\limits_{x',y'} T(x',y')^2 \sum\limits_{x',y'} I(x+x', y+y')^2}}$$

where $T$ is the image patch around the feature found and $I$ is the image in which the matching feature will be searched. To find the matching point, in the image where the feature will be searched, among the corners found in this image, image point of the feature is compared with the image block around each corner in this image. An image block with the same size with the image patch of the feature is formed and and NCC between this block and the image patch is calculated. For the perfect match, NCC gives the value 1. Usually, due to the noise, illumination changes and transformations caused by motion, perfect match cannot be reached. 0.85 is used in this thesis as threshold value for NCC and values below this threshold are eliminated.

As stated in the previous section, search and matching is performed on the corners found at each image. At initial feature detection, strong corners are found at both images, than for each corner in the right image corresponding corner point is searched in the left image around the epipolar line by normalized cross-correlation. (Right camera is chosen as reference camera). Corners in the vicinity of epipolar line are searched for the best fit and the one which has the highest correlation is chosen if it is also bigger than the threshold value. Then, 3D points for this points are calculated and added to the system. For the consequitive frames, to track a feature that is already added to the system, stored image patch of this feature is used. In this case, a search region is found and for each corners which lie in this region, correlation of the image patch around these corners with the stored image patch is calculated. Best match is chosen as the matched feature. Again to find the 3D point of this feature, search around the epipolar line in the left image is performed. Determination of search region will be described in Section 7.1.3.

## 5.3 Depth Tests

Accurate measurement of the position of the landmarks with respect to the robot is important for robot localization algorithms. Although, an uncertainty is defined to compensate the errors in measurements in these systems. True measurements both increase the consistency and reliability of these systems.

In this section, for both artificial and natural landmarks, depth tests are performed to see the response of the camera system for different types of landmarks. In addition, for artificial landmarks, minimum and maximum distances for detectability and recognizability are measured.

For testing, the robot vehicle on which stereo camera system is mounted, is placed to the ground floor. From that location, through the right camera's principal axis, depth of the landmarks are measured. For natural landmarks, corners on a test pattern are detected. By placing the pattern to the different places through the principal axis. Depth of the corners are measured for different distances. Test pattern is white bands on a black background. High contrast on the bands and the background increases the strength of the corners.

For artificial landmarks, a selected landmark is placed the different locations through the principal axis of the right camera and for each location depth is measured. In addition, all landmarks are tested at some positions to find the maximum and minimum distances for detectability and recognizability of the landmarks.

Table 5.2 shows the measurement values of the features for different distances.

Table 5.2: Depth Tests Results of the Artificial and Natural Landmarks.

| **Ground Truth (mm)** | **Artificial Landmarks (mm)** | **Natural Landmarks (mm)** |
|---|---|---|
| 500 | 514.2 | 515.6 |
| 1000 | 990.4 | 1039.9 |
| 1500 | 1451 | 1570.3 |
| 2000 | 1892.8 | 2074.6 |
| 2500 | 2290.2 | 2577.7 |
| 3000 | 2896.8 | 3167.8 |
| 3500 | - | 3676 |

Table 5.3: Detectability and Recognizability Ranges for the Artificial Landmarks

|  | **Detectability** | **Recognizability** |
|---|---|---|
| **Minimum**(mm) | 500 | 800 |
| **Maximum**(mm) | 3000 | 3000 |

From the measurement values in Table 5.2, it can be concluded that, measured depth values are near to the real ones. However, natural landmarks (corners) give better results than the artificial ones. It is mainly due to the matching errors. The center point of the coded cylinders cannot be found exactly in the left and right images and thus measurement errors occur. For corners, finding the corresponding pair is more successfully performed and this affects the accuracy of the depth measurements. "-" for the value of the depth value of the artificial landmark at 350cm shows that a measurement cannot be taken at this distance because of the limitations which are given below. Although there exist a depth value for the natural landmark at this distance, it does not corresponds to the same landmark. After 300cm, the corner of the interest become undetectable.

Minimum and maximum distances for the detectability and recognizability is given in Table 5.3.

Although it seems that ranges for the artificial landmarks that we used is low, for the experimental setup that we performed localization tests, these values are sufficient. In addition, it is possible to increase ranges by increasing the resolution of the grabbed images and increasing the dimensions of the coded cylinders.

# Chapter 6

# Robot Localization Using Artifical Landmarks

## 6.1 Robot Localization Algorithm

### 6.1.1 State Vector and its Covariance

Since position of the landmarks are known and there is no uncertainty about their positions, the only point of interest is the position of the robot in the world coordinate frame. As explained in Section 4.2.1, robot position is represented as $(x, z, \theta)^{\mathsf{T}}$ and state vector to keep the position information is chosen as this representation. Current estimate of the location of the robot is kept in the mean vector of the state vector and denoted as $\hat{\mathbf{x}}$.

$$\hat{\mathbf{x}} = \begin{pmatrix} x \\ z \\ \theta \end{pmatrix}$$

Covariance matrix of the state vector is $3 \times 3$ symmetric matrix and defined as

$$\mathsf{P} = cov[(\mathbf{x} - \hat{\mathbf{x}})(\mathbf{x} - \hat{\mathbf{x}})^{\mathsf{T}}]$$

where $\mathbf{x}$ is the real value of the position and $\hat{\mathbf{x}}$ is the estimate of the robot position. Covariance of the state vector gives information about the uncertainty about the estimated position of the robot. In other words, it gives information about how much the estimated position deviated from the real position. Dimensions of mean vector and covariance matrix are constant. At each step, values in this vector and matrix will be updated. $x$ and $z$ in the state vector are in millimeters while $\theta$ is in radians.

### 6.1.2 Predicting State and Covariance

In EKF, first of the two phases at each step is prediction. In prediction phase, state vector is predicted using the process model. Current robot position is predicted by using the vehicle model equations 4.1, 4.2 and 4.3. Predicted state is

$$\hat{\mathbf{x}}(k+1|k) = \mathbf{f}_v(\hat{\mathbf{x}}(k|k), \mathbf{u}(k))$$

where $\hat{\mathbf{x}}(k|k)$ is the updated position of the previous step and $\mathbf{u}(k)$ is the input vector.

Using the EKF covariance prediction equation 2.24, predicted covariance is

$$\mathtt{P}(k+1|k) = \frac{\partial \mathbf{f}_{\mathtt{v}}}{\partial \mathbf{x}} \mathtt{P}(k|k) \frac{\partial \mathbf{f}_{\mathtt{v}}}{\partial \mathbf{x}}^{\mathsf{T}} + \mathtt{Q}(k)$$

where $\frac{\partial \mathbf{f}_{\mathtt{v}}}{\partial \mathbf{x}}$ is the Jacobian matrix of the $\mathbf{f}_v$ with respect to state vector $\mathbf{x}$ at $\hat{\mathbf{x}}(k|k)$ and $\mathtt{Q}(k)$ is the process noise covariance. $\frac{\partial \mathbf{f}_{\mathtt{v}}}{\partial \mathbf{x}}$ is given as

$$\frac{\partial \mathbf{f}_{\mathtt{v}}}{\partial \mathbf{x}} = \begin{bmatrix} 1 & 0 & d\cos\theta \\ 0 & 1 & d\sin\theta \\ 0 & 0 & 1 \end{bmatrix}$$

### 6.1.3 Predicting Measurements and Innovation Covariance

Measurement device in our system is stereo camera system. The camera system finds the positions of the landmarks in camera coordinate system. Since robot position in the state vector is in world coordinate frame, relationship between found landmark positions, and the robot position in the world coordinate frame should be determined. Measurement function $\mathbf{h}_i$ for the $i^{th}$ landmark is

$$\mathbf{h}_i = \begin{pmatrix} h_{ix} \\ h_{iy} \\ h_{iz} \end{pmatrix} = \begin{pmatrix} (x_i - x)\cos\theta - (z_i - z)\sin\theta \\ y_i \\ (x_i - x)\sin\theta - (z_i - z)\cos\theta \end{pmatrix} \tag{6.1}$$

where $x_i$,$y_i$,$z_i$ are the position values of the $i^{th}$ landmark in the world coordinate frame. Since locations of the landmarks in the world coordinate frame are assumed to be known, these values are given to the system at the beginning.

Predicted measurement for each landmark is calculated using the above equation. Predicted measurement vector is denoted by $\hat{\mathbf{z}}$ and is equal to $\mathbf{h}(\hat{\mathbf{x}}(k+1|k))$ where

$$\mathbf{h} = \begin{bmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \vdots \\ \mathbf{h}_n \end{bmatrix}$$

Innovation Covariance determines the deviation from the predicted measurement values. Innovation covariance is calculated from the Equation 2.25. $\frac{\partial \mathtt{h}}{\partial \mathtt{x}}$ is the Jacobian matrix of $\mathbf{h}$ with respect to state vector $\mathbf{x}$ at $\hat{\mathbf{x}}(k+1|k)$.

Innovation Covariance and predicted measurement vector can be used to define a search region for the landmark detection. However, in this thesis, artificial landmarks that we used are needed to be searched in whole image, because image area representing each landmark is large and also a recognition phase is needed to distinguish each landmark. Thus, a search region will be useless in this part of the thesis. Uses of innovation covariance and predicted measurements in determining a search region will be explained in the next chapter.

Although, determining a search region is useless for this part of the thesis, determining a field of view can be useful to eliminate false matches. That is, since position of each landmark in the world coordinate frame is known and predicted measurement for each landmark is calculated, landmarks which remain out of the field of view are marked as unseen and if the measurements show that a landmark which is marked as unseen is on the view, this measurement is accepted as false and eliminated. This approach increases the robustness of the system and reduces the errors caused by false measurements.

### 6.1.4 Correcting the State and its Covariance

After measurements are taken, mean vector and state covariance are updated using the EKF correction equations 2.28 and 2.29.

Filter gain is

$$\mathtt{K}(k+1) = \mathtt{P}(k+1)\frac{\partial \mathtt{h}}{\partial \mathtt{x}}\mathtt{S}(k+1)^{-1}$$

State estimation and covariance are

$$\hat{\mathtt{x}}(k+1|k+1) = \hat{\mathtt{x}}(k+1|k) + \mathtt{K}(k+1)(\mathbf{z} - \mathbf{h}(\hat{\mathtt{x}}(k+1|k))) \tag{6.2}$$

$$\mathtt{P}(k+1|k+1) = \mathtt{P}(k+1|k) - \mathtt{KSK}^{\mathsf{T}} \tag{6.3}$$

At each step, at most three landmarks are seen and measurements to these seen landmarks can be taken. For the unseen landmarks, predicted measurements are given as measurement. By this way, contribution of the unseen landmarks becomes zero and system is updated using only measurements of the seen landmarks.

## 6.2 Experiments

### 6.2.1 Experimental Setup

Experiments are performed in our laboratory. General application area of robot localization with artificial landmarks is industry, however, performed experiments are convenient to adapt the environment to an industrial indoor environment.

Since measurements and estimates are in metric system, ground-truth measurements are needed to see the actual position of the robot and to compare results. For this reason, a grid is drawn on the floor. Regularly spaced rectangles each $50 \times 50$ millimeters occupy an area of $2.5 \times 2.5$ meters which is enough for characterization of experiments. The grid provides both visual and quantitative information about the position of the robot. Figure 6.1 shows the experimental setup.

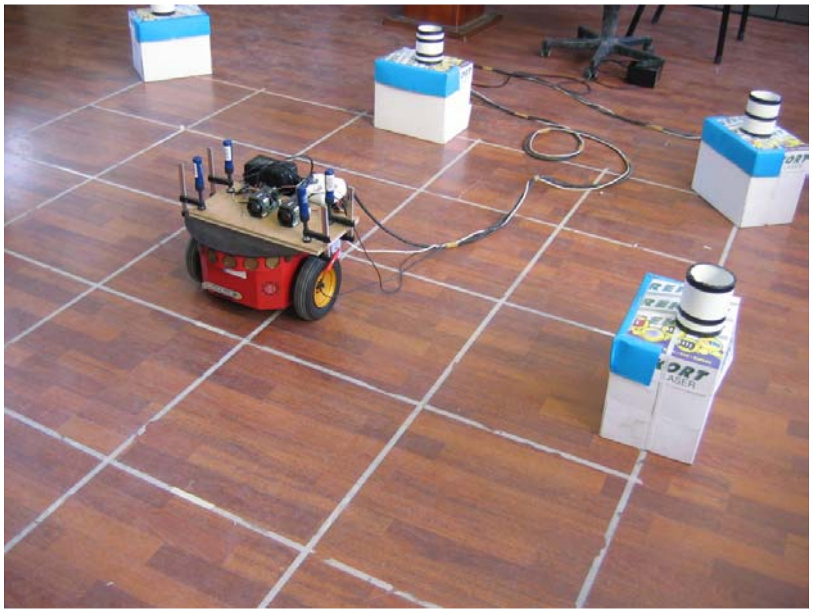


Figure 6.1: Experimental Setup for the Artificial Landmark Tests

As explained in the Chapter 5, there are eight landmarks which are cylinders on which black stripes are sticked to identify the landmarks. Location of each landmark in the world coordinate frame is assumed to be known. For this reason, landmarks are placed in the known coordinates of the grid. Four landmarks are placed to the outer corners of the grid area and four landmarks are placed to the midpoint of the two adjacent landmarks which are at the corners. Figure 6.2 shows general layout of the landmarks on the grid area.

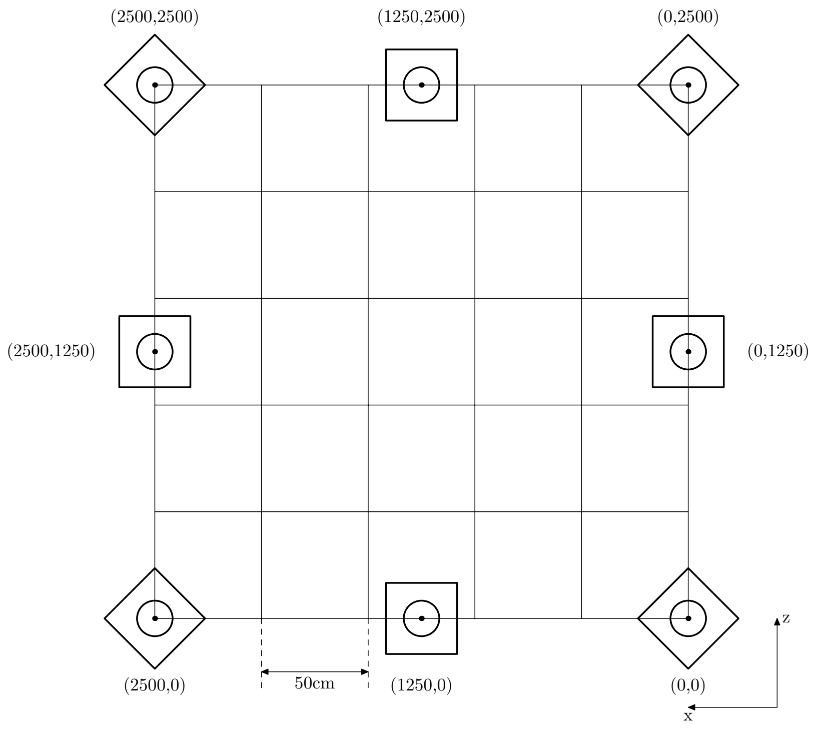


Figure 6.2: Placement of landmarks on the grid area

Each landmark is placed on a box of height 28.6 to bring the landmarks to the same height with the camera system. By this way, landmarks can be seen for a long time without going out of view caused by the height. Color of the boxes are white and to prevent potential errors due to this white color, a blue rectangular cartoon is sticked to the top of the seen sides of the box. To find the position of the robot, distances to the nearest grid point is measured. Since, center of the robot is chosen as the middle point of the two front wheels, this point is marked and measurements are taken according to this point. For orientation, we draw lines between the points where wheels of the robot touches the ground. Then, we measure the angle between this line and the line parallel to the grid line.

For the reasons stated in the section 4.2.1, motion of the robot is step-by-step. At each step, after robot stops, images are grabbed from cameras and processed to take measurements. In this part of the thesis, since each landmark is easily identifiable, step size can be chosen to be long.

### 6.2.2 Kidnapped Robot Test

Kidnapped Robot problem [9] differs from the robot localization problem in that the robot is told a different position from the actual position. Kidnapped robot problem is used to test the robot's ability to recover from big localization errors [10]. This test aims to test the response of the algorithm for the kidnapped robot problem. For this reason, robot is placed to the location (505,505) on the grid area but the coordinate (1260,1260) is given to the system as the initial position. Then the robot is given the command of 45 degrees of turn in steps of 5 degrees. During the turn, the robot sees only the landmark of number 2. Test results are given below. As seen in

Table 6.1: Test Results for the Kidnapped Robot Experiment

| Time Step | Predicted Pos. | | Corrected Pos. | | Odometry | |
|---|---|---|---|---|---|---|
| | x(mm) | z(mm) | x(mm) | z(mm) | x(mm) | z(mm) |
| 1 | 1260 | 1260 | 1323 | 1164 | 0 | 0 |
| 2 | 1323 | 1164 | 1323 | 1062 | 0 | 0 |
| 3 | 1323 | 1062 | 1280 | 987 | 0 | 0 |
| 4 | 1280 | 987 | 1203 | 915 | 0 | 0 |
| 5 | 1203 | 915 | 1114 | 885 | 1 | 0 |
| 6 | 1114 | 885 | 995 | 843 | 1 | 0 |
| 7 | 995 | 843 | 947 | 667 | 1 | 0 |
| 8 | 947 | 667 | 812 | 596 | 2 | 0 |
| 9 | 812 | 596 | 653 | 586 | 2 | 1 |

the above test results, step by step, the system finds the robots true position. Note that, odometry device in the robot system gives values with the assumption that its starting point is (0,0,0). So if the robot is given that it is in (1260,1260,0), its results are simply bias of this given initial point and it is not able to find the robots true position.

# Chapter 7

# Robot Localization Using Natural Landmarks

## 7.1 Robot Localization Algorithm

Robot localization algorithm in this part of the thesis uses natural landmarks and works similar to EKF-based SLAM algorithms. Although, robot's position is to be estimated, position distribution of features(landmarks) which robot measures distances with respect to, are propagated with robot's position to next steps. In SLAM algorithms by this way, mapping of the features in the world coordinate system is also performed. However, since besides localization, mapping is the aim in these algorithms, often, features are stored and recalculated at each step for re-association. For short-term localization problems, with the cost of increasing uncertainty, without storing features in the system, unseen features are removed from the system and remaining features with newly added ones are used for localization. We implemented the algorithm described in [3] by adapting it to our robot system and considering only robot localization problem rather than SLAM. Algorithm, uses EKF for state estimation and propagation of uncertainty. This section will describe our robot localization algorithm. For the rest of the chapter the term, "feature" will be used to represent the term, "natural landmark".

### 7.1.1 State Vector and its Covariance

Current estimate of the robot's position and currently seen features' locations are kept in the mean vector and denoted as $\hat{\mathbf{x}}$. Uncertainty in the estimate is kept in covariance and denoted as $\mathtt{P}$. $\hat{\mathbf{x}}$ and $P$ are defined as

$$\hat{\mathbf{x}} = \begin{bmatrix} \hat{\mathbf{x}}_v \\ \hat{\mathbf{y}}_1 \\ \hat{\mathbf{y}}_2 \\ \vdots \\ \hat{\mathbf{y}}_n \end{bmatrix} \quad , \quad \mathtt{P} = \begin{bmatrix} \mathtt{P}_{xx} & \mathtt{P}_{xy_1} & \mathtt{P}_{xy_2} & \dots & \mathtt{P}_{xy_n} \\ \mathtt{P}_{y_1x} & \mathtt{P}_{y1y_1} & \mathtt{P}_{y_1y_2} & \dots & \mathtt{P}_{y_1y_n} \\ \mathtt{P}_{y_2x} & \mathtt{P}_{y_2y_1} & \mathtt{P}_{y_2y_2} & \dots & \mathtt{P}_{y_2y_n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathtt{P}_{y_nx} & \mathtt{P}_{y_ny_1} & \mathtt{P}_{y_ny_2} & \dots & \mathtt{P}_{y_ny_n} \end{bmatrix}$$

where $\hat{\mathbf{x}}_v$ is the robot position vector, and $\hat{\mathbf{y}}_i$ is the $i^{th}$ feature's location vector

$$\hat{\mathbf{x}}_v = \begin{pmatrix} x \\ z \\ \theta \end{pmatrix} \quad , \quad \mathbf{y}_i = \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix}$$

and $\mathtt{P}_{xx}$,$\mathtt{P}_{xy_i}$,$\mathtt{P}_{y_ix}$,$\mathtt{P}_{y_iyj}$ are $3 \times 3$ are covariance matrices of the subscripted vectors.

$\hat{\mathbf{x}}$ is $3(n+1) \times 1$ vector, while $\mathtt{P}$ is $3(n+1) \times 3(n+1)$ symmetric matrix where n is the number of features in the field of view. Dimensions of both $\hat{\mathbf{x}}$ and $\mathtt{P}$ are dynamic, that is, as the features are added to or deleted from the system, dimensions change. All entities are in world coordinate frame and in millimeters.

### 7.1.2 Predicting State and its Covariance

Current robot position is predicted by using robot model equations 4.1,4.2 and 4.3. Since features in the scene are stationary, predicted feature locations are the same with the previous corrected locations. Predicted state $\hat{\mathbf{x}}(\mathbf{k+1|k})$ is

$$\hat{\mathbf{x}}(k+1|k) = \begin{bmatrix} \mathbf{f}_v(\hat{\mathbf{x}}(k|k), \mathbf{u}) \\ \hat{\mathbf{y}}_1(k|k) \\ \hat{\mathbf{y}}_2(k|k) \\ \vdots \\ \hat{\mathbf{y}}_n(k|k) \end{bmatrix} \tag{7.1}$$

where $\hat{\mathbf{x}}$ and $\mathbf{u}$ are defined in section 4.2.1.

State covariance is given as

$$\mathtt{P} = \begin{bmatrix} \frac{\partial \mathbf{f_v}}{\partial \mathbf{x_v}} \mathtt{P}_{xx}(k|k) \frac{\partial \mathbf{f_v}}{\partial \mathbf{x_v}}^{\mathsf{T}} + \mathtt{Q}_v(k) & \frac{\partial \mathbf{f_v}}{\partial \mathbf{x_v}} \mathtt{P}_{xy_1} & \frac{\partial \mathbf{f_v}}{\partial \mathbf{x_v}} \mathtt{P}_{xy_2} & \dots & \frac{\partial \mathbf{f_v}}{\partial \mathbf{x_v}} \mathtt{P}_{xy_n} \\ \mathtt{P}_{y_1x} \frac{\partial \mathbf{f_v}}{\partial \mathbf{x_v}}^{\mathsf{T}} & \mathtt{P}_{y1y_1} & \mathtt{P}_{y_1y_2} & \dots & \mathtt{P}_{y_1y_n} \\ \mathtt{P}_{y_2x} \frac{\partial \mathbf{f_v}}{\partial \mathbf{x_v}}^{\mathsf{T}} & \mathtt{P}_{y_2y_1} & \mathtt{P}_{y_2y_2} & \dots & \mathtt{P}_{y_2y_n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathtt{P}_{y_nx} \frac{\partial \mathbf{f_v}}{\partial \mathbf{x_v}}^{\mathsf{T}} & \mathtt{P}_{y_ny_1} & \mathtt{P}_{y_ny_2} & \dots & \mathtt{P}_{y_ny_n} \end{bmatrix} \tag{7.2}$$

where $\mathtt{Q}_v(k)$is the process noise covariance matrix that is defined in section 4.2.1 and $\frac{\partial \mathbf{f}_v}{\partial \mathbf{x}_v}$ is the Jacobian matrix of the $\mathbf{f}_v$ with respect to robot position vector $\mathbf{x}_v$ at $\hat{\mathbf{x}}_v(k|k)$ which is defined in Section 6.1.2

State Covariance is calculated by using the Equation 2.24. $\frac{\partial \mathtt{f}}{\partial \mathtt{x}}$ and $\mathtt{Q}(k)$ in this equation are

$$\frac{\partial \mathtt{f}}{\partial \mathtt{x}} = \begin{bmatrix} \frac{\partial \mathbf{f}_v}{\partial \mathbf{x}_v} & 0 & 0 & \dots & 0 \\ 0 & \mathtt{I} & 0 & \dots & 0 \\ 0 & 0 & \mathtt{I} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \mathtt{I} \end{bmatrix} \quad , \quad \mathtt{Q}(k) = \begin{bmatrix} \mathtt{Q}_v(k) & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix} .$$

Identity matrices in the Jacobian matrix and $\mathtt{0}$ terms in the process covariance matrix comes from the assumption that features in the scene do not move.

### 7.1.3 Predicting Measurements and Innovation Covariance

The camera system finds the positions of the features in camera coordinate system. Since robot position and feature locations in the state vector are in world coordinate frame, relationship between found feature positions, robot position and feature locations in the world coordinate frame should be determined. For $i^{th}$ feature, this relationship is kept in the measurement function $\mathbf{h}_i(\hat{\mathbf{x}}(k+1|k))$ which is as in Section 6.1.3.

To predict the measurement value at the predicted state, predicted state values are applied to the above equation for each feature. Predicted measurement vector is denoted by $\hat{\mathbf{z}}$ and is equal to $\mathbf{h}(\hat{\mathbf{x}}(k+1|k))$ where

$$\mathbf{h} = \begin{bmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \vdots \\ \mathbf{h}_n \end{bmatrix}$$

Innovation Covariance determines the deviation from the predicted measurement values. Innovation covariance is calculated using the Equation 2.25. $\frac{\partial \mathtt{h}}{\partial \mathtt{x}}$ is the Jacobian matrix of $\mathbf{h}$ with respect to state vector $\mathbf{x}$ at $\hat{\mathbf{x}}(k+1|k)$.

Predicted measurements and innovation covariance can be used to determine a search region for the feature. However, both predicted measurement values and innovation covariance are defined for the camera coordinate frame and in millimeters. Since to find position information of the features, projections of the features on the image planes are used. These values must be converted into image coordinates in pixels. Using projection equation $\mathbf{x} = P\mathbf{X}$ as explained in Section 3.1.2, predicted 3D points can be converted into image coordinates for both cameras.

For innovation covariance, first order propagation of uncertainty is used to find the covariance in the image coordinates. Using the internal camera matrix $\mathtt{K}$, relation of image coordinates to 3D points are written explicitly. For the right camera this relation is

$$u_R = \frac{f_{Rx}h_x + s_R h_y}{h_z} + u_{R0} \quad , \quad v_R = \frac{f_{Ry}h_y}{h_z} + v_{R0}.$$

Innovation Covariance for the image vector $\mathbf{u}_R = \begin{pmatrix} u_R \\ v_R \end{pmatrix}$, is computed from $\mathtt{U}_R = \frac{\partial \mathbf{u}_\mathrm{R}}{\partial \mathbf{h}} \mathtt{S}(k+1) \frac{\partial \mathbf{u}_\mathrm{R}}{\partial \mathbf{h}}^\mathsf{T}$. Here the value of the Jacobian is

$$\frac{\partial \mathbf{u}_\mathrm{R}}{\partial \mathbf{h}} = \begin{bmatrix} \frac{f_{Rx}}{h_z} & \frac{s}{h_z} & -\frac{f_{Rx}h_x + s_R h_y}{h_z^2} \\ 0 & \frac{f_{Ry}}{h_z} & -\frac{f_{Ry}h_y}{h_z^2} \end{bmatrix}$$

By applying above calculations for each camera, predicted image coordinates and image covariance is found. To find the search region, a number of standard deviations are specified. By this way, an ellipse is formed and image features are to be searched in this region. To simplify the operations, rather than finding an ellipse, taking only square roots of the diagonal elements of the image covariance, a rectangle is formed and image features are searched in this rectangle. Finding a search region, increases the computational efficiency and decreases the probability of mismatches.

### 7.1.4 Correcting the State Vector and its Covariance

After measurements for the features are taken, corrected(updated) state vector and its covariance is calculated using EKF correction equations. Gain is

$$\mathtt{K}(k+1) = \mathtt{P}(k+1)\frac{\partial \mathtt{h}}{\partial \mathtt{x}}\mathtt{S}(k+1)^{-1} \tag{7.3}$$

State estimate and its covariance is found by using the gain as

$$\begin{aligned} \hat{\mathtt{x}}(k+1|k+1) &= \hat{\mathtt{x}}(k+1|k) + \mathtt{K}(k+1)(\mathbf{z} - \mathbf{h}(\hat{\mathtt{x}}(k+1|k))) \quad (7.4) \\ \mathtt{P}(k+1|k+1) &= \mathtt{P}(k+1|k) - \mathtt{KSK}^\mathsf{T} \quad (7.5) \end{aligned}$$

When taking measurements, it is probable that some features cannot be seen and thus, there may not be any measurements for these features. In this case for this time step, predicted measurements are entered to the state vector update equation and by this way, contribution of these features to the state vector becomes zero. At next step, these unseen features are deleted. Deleting a feature will be explained later.

### 7.1.5 Adding a new feature

As stated earlier, state vector and its covariance matrix are dynamic. Their dimensions change as new features are added or deleted. At initialization and when number of features decreases up to a threshold number(typically 2), new features are added to the system. When adding a new feature, its initial position in the world coordinate frame and its initial covariance should be found. In addition, cross-covariances with other features and robot position should be added to the state covariance matrix.

For this purpose, using measurements taken from the camera system, initial position is found. Relationship between camera measurements of the feature and its counterpart in the world coordinates and robot position is similar to the equation 6.1 and is given as

$$\mathbf{y}_i = \begin{pmatrix} x_v + h_{ix}\cos\theta + h_{iz}\sin\theta \\ h_{iy} \\ z_v + h_{ix}\sin\theta + h_{iz}\cos\theta \end{pmatrix} \tag{7.6}$$

Adding this new feature to the state vector is simply concatenating this column vector to the end of state vector. To form the state covariance, Jacobians $\frac{\partial \mathbf{y_i}}{\partial \mathbf{x}}$ and $\frac{\partial \mathbf{y_i}}{\partial \mathbf{h}}$ are calculated. New state vector and its state covariance after a new feature added to the existing two features is as follows:

$$\mathbf{x}_{new} = \begin{pmatrix} \mathbf{x}_v \\ \mathbf{y}_1 \\ \mathbf{y}_2 \\ \mathbf{y}_i \end{pmatrix} \tag{7.7}$$

$$\mathtt{P}_{new} = \begin{bmatrix} \mathtt{P}_{xx} & \mathtt{P}_{xy_1} & \mathtt{P}_{xy_2} & \mathtt{P}_{xx}\frac{\partial \mathbf{y_i}}{\partial \mathbf{x}}^{\mathsf{T}} \\ \mathtt{P}_{y1x} & \mathtt{P}_{y_1y_1} & \mathtt{P}_{y_1y_2} & \mathtt{P}_{y1x}\frac{\partial \mathbf{y_i}}{\partial \mathbf{x}}^{\mathsf{T}} \\ \mathtt{P}_{y2x} & \mathtt{P}_{y_2y_1} & \mathtt{P}_{y_2y_2} & \mathtt{P}_{y2x}\frac{\partial \mathbf{y_i}}{\partial \mathbf{x}}^{\mathsf{T}} \\ \frac{\partial \mathbf{y_i}}{\partial \mathbf{x}}\mathtt{P}_{xx} & \frac{\partial \mathbf{y_i}}{\partial \mathbf{x}}\mathtt{P}_{xy_1} & \frac{\partial \mathbf{y_i}}{\partial \mathbf{x}}\mathtt{P}_{xy_2} & \frac{\partial \mathbf{y_i}}{\partial \mathbf{x}}\mathtt{P}_{xx}\frac{\partial \mathbf{y_i}}{\partial \mathbf{x}}^{\mathsf{T}} + \frac{\partial \mathbf{y_i}}{\partial \mathbf{h}}\mathtt{R}\frac{\partial \mathbf{y_i}}{\partial \mathbf{h}}^{\mathsf{T}} \end{bmatrix} \tag{7.8}$$

where R is the measurement noise covariance.

### 7.1.6 Deleting a Feature

If a feature in the state vector cannot be seen by the camera system, in other words, image feature points corresponding to feature cannot be found in the calculated search region. Predicted measurement is returned to update equation and feature is deleted.

Deleting is simply removing corresponding row and column from the state vector and the state covariance. Below, an example deletion operation is shown. First feature of three features is deleted from the state vector and its covariance:

$$\begin{pmatrix} \mathbf{x}_v \\ \mathbf{y}_1 \\ \mathbf{y}_2 \\ \mathbf{y}_3 \end{pmatrix} \rightarrow \begin{pmatrix} \mathbf{x}_v \\ \mathbf{y}_2 \\ \mathbf{y}_3 \end{pmatrix}$$

$$\begin{bmatrix} \mathrm{P}_{xx} & \mathrm{P}_{xy_1} & \mathrm{P}_{xy_2} & \mathrm{P}_{xy_3} \\ \mathrm{P}_{y_1x} & \mathrm{P}_{y_1y_1} & \mathrm{P}_{y_1y_2} & \mathrm{P}_{y_1y_3} \\ \mathrm{P}_{y_2x} & \mathrm{P}_{y_2y_1} & \mathrm{P}_{y_2y_2} & \mathrm{P}_{y_2y_3} \\ \mathrm{P}_{y_3x} & \mathrm{P}_{y_3y_1} & \mathrm{P}_{y_3y_2} & \mathrm{P}_{y_3y_3} \end{bmatrix} \rightarrow \begin{bmatrix} \mathrm{P}_{xx} & \mathrm{P}_{xy_2} & \mathrm{P}_{xy_3} \\ \mathrm{P}_{y_2x} & \mathrm{P}_{y_2y_2} & \mathrm{P}_{y_2y_3} \\ \mathrm{P}_{y_3x} & \mathrm{P}_{y_3y_2} & \mathrm{P}_{y_3y_3} \end{bmatrix}$$

## 7.2 Experiments

To test the algorithm, some experiments are performed. One of the experiments tests the system and the algorithm using specified setup and manual feature selection. The other experiment is with automatic feature selection and detection. Also, in this experiment, effect of the search region and feature deletion is tested.

### 7.2.1 Experimental Setup

Experiments are performed in our laboratory in which there are many features to be tracked. Since there is not any artificial landmark in this part of the thesis, existence of many features is convenient for the experiments. Layout of the laboratory is similar to a large room in a house and this makes the robot, a potential house tool like automatic vacuum cleaner.

Experimental setup is similar to the one which is described in Section 6.2.1. However, there is not any artificial landmark and camera system searches for the natural ones. Measurement method of the ground truth values are performed in the same way on the method that is described in Section 6.2.1.

To simplify and specify the test on feature selection, a white board is placed to the one edge of the grid area, and robot is moved through this board. Black objects are placed in front of this board to represent features. Since there is high contrast between objects and the white board, corners on the objects can be easily detected and added as features to the system. Figure 7.1 shows the experimental setup for the experiments performed to test the algorithm in this chapter.

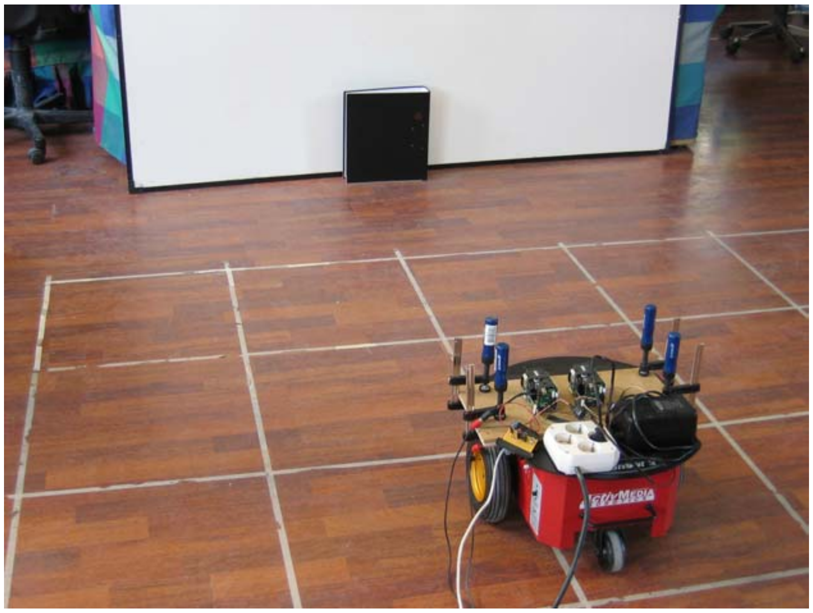


Figure 7.1: Experimental Setup for the Natural Landmark Tests

### 7.2.2 Experiments with Manual Feature Selection

In this part of the experiments, the algorithm is tested with controlled measurements. At each step, features are added and matched by the operator manually. At initialization, the operator selects some points which have nonuniform pattern around them. Corresponding points of the selected point on the left image is found by correlation method that is described in Section 5.2.3. After 3D position of the each point is calculated, these 3D positions are added to the state vector. According to the given motion input, position and measurement values are calculated. Finding the same feature points on the consequitive images is also done by the operator, manually.

Table 7.1: Experimental Results of the Manual Feature Selection for forward-backward Motion

| Time Step | Predicted Pos. (mm) | Corrected Pos. (mm) | Odometry (mm) |
|---|---|---|---|
| 1 | 50 | 39 | 46 |
| 2 | 88 | 90 | 95 |
| 3 | 140 | 137 | 143 |
| 4 | 187 | 175 | 191 |
| 5 | 225 | 214 | 241 |
| 6 | 264 | 263 | 291 |
| 7 | 313 | 315 | 341 |
| 8 | 365 | 380 | 393 |
| 9 | 430 | 425 | 441 |
| 10 | 475 | 473 | 490 |

Motion pattern for this experiment is forward motion of 500 mm with 50 mm at each step. In ideal case, in this type of motion, the robot vehicle is expected to stop at (0,500,0) position. However, mostly there is an error and ability of the algorithm to find this error compared to the odometry values is tested. Experimental results for this test is given in Table 7.1. These results show only the $z$ component of the robot position because the robot moves on the z direction and the error on this direction is the most significant compared to other components.

Starting from the (0,0,0) position after 10 steps of motion, ground truth $z$ component of the position is 475 mm. Table 7.2 shows the deviation from the ground-truth value. As seen in the above table, algorithm gives the most true value about the

Table 7.2: Deviation From the Ground-Truth Value

| Ground-truth (mm) | Predicted Pos. $\Delta z(mm)$ | Corrected Pos. $\Delta z(mm)$ | Odometry $\Delta z(mm)$ |
|---|---|---|---|
| 475 | 0 | 2 | 15 |

position of the robot provided that features are matched truly.

### 7.2.3 Experiments with Automatic Feature Selection

In this experiment, all feature addition, deletion and matching operations are performed automatically. At initialization step, strong corners that have quality levels higher than the threshold are added to the state vector after 3D positions are calculated. At each step, features at the state vector are searched around the predicted positions in the image. If the matched point is found, its corresponding point in the left image is searched. If found, 3D position is calculated and given to the system as measurement.

In the cases that a match point for the features in the state vector cannot be found or corresponding point of the match point cannot be found in the left image, calculated predicted measurement value of this feature is given to the system as the posterior measurement. By this way, contribution of the measurement is made zero. Then, this feature is deleted from the state vector and the state covariance. This is applied for the first two sets of tests. For the third set, features are not deleted right after they cannot be detected. Instead, a feature is deleted if it cannot be detected for a long time (for example 5 times). By this way, temporary lost of feature is prevented.

In addition, as stated earlier in Section 7.1.3, the predicted measurement and the innovation covariance are used to determine a search region in the images for the search of the features. Generally three standard deviations are used for the size of the search region. Effect of the five standard deviation is also tested.

At each step, strong corners are searched and the corners that are different from the existing ones and above the threshold are added to the system. This increases the accuracy of the estimation.

Two types of motion patterns are used for tests. One of them is forward-backward motion and the other is turn clockwise-counterclockwise motion.

Forward-Backward motion is such that robot moves forward 500 mm and moves backward 500 with steps of 50 mm. The robot is expected to stop at starting point but generally, it stops at a position different from the starting point. Both algorithm's and odometry's response to this error is tested. For the test, parameters are given in Table 7.3 and test results are given in Table 7.4.

Table 7.3: Test Parameters for the Forward-Backward Motion

| | |
|---|---|
| Quality Level | 0.40 |
| MinDistance | 10 |
| Patch Size | $7 \times 7$ |
| Correlation Threshold | 0.80 |
| Measurement Noise Cov | diag(9,100,625) |

In Table 7.4, Pos(3) denotes the test results for search region with three standard deviation. Pos(5) is the test results for search region with five standard deviations. In these two tests, lost features are deleted immediately. Pos (ND3) denotes the test results for the tests in which features are not deleted immediately and search region is with three standard deviations.

As seen in Table 7.4, in the results of Pos(3), after some steps, it gives erroneous results and the error propagates. This is due to the lost of all features, in some steps. In this case, it detects new features and these new features are located according to the estimated position of the robot and information related to the previous features are completely lost. In the results of Pos(5), since search region is big, features are not completely lost in any steps and therefore it gives close results to the ground-truth values. In Pos(ND3), since features are not deleted immediately, information about a feature is kept in next steps, even a feature is lost in one step. By this way, features are tracked for a long time.

In 100. step, odometry gives the value of 5, although the actual position is 183. This is because before the 100.step, the robot is prevented to move by holding the robot. During this prevention, wheels continue to turn and because of this, odometry gives this far result. The results in Pos(5) and Pos(ND3) are consistent with the ground-truth values.

The other motion pattern for tests is turn clockwise-counterclockwise motion such that the robot turns 40 degrees counterclockwise and 40 degrees clockwise with steps of 2 degrees. This motion is repeated two times. Test results are given in Table 7.5

As seen in Table 7.5, for all tests, the algorithm give close results to the ground-truth value.

Table 7.4: Test Results of Automatic Feature Selection for Forward-Backward Motion

| Time Step | Pos.(3) (mm) | Pos.(5) (mm) | Pos.(ND3) (mm) | Odometry (mm) | Ground-Truth (mm) |
|---|---|---|---|---|---|
| 10 | 460 | 523 | 523 | 501 | 478 |
| 20 | 7 | 8 | 8 | 9 | 6 |
| 30 | -560 | -535 | -590 | -496 | -483 |
| 40 | -109 | -69 | -59 | -27 | -28 |
| 50 | 405 | 489 | 481 | 478 | 460 |
| 60 | -55 | -44 | -30 | -17 | -16 |
| 70 | -622 | -540 | -556 | -530 | -508 |
| 80 | -115 | -45 | -65 | -41 | -39 |
| 90 | 387 | 486 | 474 | 463 | 441 |
| 100 | 48 | 158 | 119 | 5 | 183 |
| 110 | -363 | -291 | -290 | -484 | -290 |
| 120 | -123 | -26 | 2 | -13 | -53 |
| 130 | 323 | 438 | 437 | 463 | 410 |
| 140 | -148 | -61 | -103 | -23 | -63 |
| 150 | -652 | -591 | -635 | -533 | -547 |
| 160 | -239 | -15 | -114 | -45 | -76 |
| 170 | 235 | 441 | 387 | 461 | 408 |
| 180 | -256 | -66 | -106 | -25 | -51 |
| 190 | -803 | -712 | -658 | -501 | -558 |
| 200 | -367 | -159 | -165 | -60 | -89 |

Table 7.5: Test Results of Automatic Feature Selection for Turn Motion

| Time Step | Pos.(3) (deg) | Pos.(5) (deg) | Odometry (deg) | Ground-Truth (deg) |
|---|---|---|---|---|
| 20 | 20.59 | 22.50 | 17.68 | 20.81 |
| 40 | -15.96 | -14.76 | -17.57 | -16.62 |
| 30 | 4.40 | 4.69 | 4.48 | -3.36 |
| 40 | -29.75 | -31.34 | -33.57 | -31.98 |

# Chapter 8

# Conclusion

## 8.1 Summary of The Results

From many aspects, robot localization algorithms both using artificial and natural landmarks are successfully implemented. Some of the results for the overall system is as follows:

- For both natural and artificial landmarks, quality of the measurements is very important. High uncertainty and wrong results for the short-term localization for the artificial landmark case is due to the high error in depth measurements. Smooth surface of the cylinder and lack of the specific points which can be fixated on both left and the right camera images has caused the system to find erroneous corresponding pairs and therefore wrong depth values. Strong corners are point features and can be fixated on both images and therefore give true corresponding pairs for the calculation of the depth of the natural landmark and therefore, more accurate localization is established by using the natural landmarks. Consequently, position of the robot is found nearly same as the ground truth values for the natural landmarks, while odometry results are near to the ground truth values for the artificial landmark case.

- Data association is a problem for both localization and SLAM problems. Artificial landmarks are chosen such that they are easily recognizable. Our landmark type has proved to be easily recognizable and can be seen same from many different point of views. Therefore even after long runs, landmarks can be redetected. By this way, uncertainty of the robot position is kept in a range and it is prevented to increase without any bound.

- In the experiments it is seen that, even it can be found easily on left and right images, on consequitive frames, after some scaling, previously detected corners become undetectable. For long-time localization it is an handicap and prevents accurate localization. In addition affine transformations affect the detectability of the corners. Therefore, in experiments related to the robot localization using natural landmarks, turning motion of the robot causes the features to be undetectable even after small angles of turn.

- Stereo camera system is fixed on the robot vehicle and views the scene that is in front of the robot vehicle. For forward and backward motion, features can be tracked for a long time without going out of the view. However, for the turn motion, scene changes rapidly and features quickly go out of view. Therefore, features can be detected only for a few steps. In addition, as stated above, the corners as features are affected from the affine transformation. These two factors sum up when a turn motion is performed in the natural landmark case and cause the robot localization algorithm to fail.

- On the contrary to the artificial landmarks, information about a feature is lost after the feature is deleted. Feature deletion is performed when the feature go out of view or is not redetected. Thus, on the long-time motion, previously detected and deleted features cannot be redetected and uncertainty of the robot position increases without any bound as time goes on.

- In the experiments, using artificial landmarks, it is seen that the robot system can recover from the error and finds its position with some error even a wrong initial position is given. For the natural landmark case, since position of the landmarks are determined according to the position of the robot, a kidnapped robot test cannot be applied.

## 8.2 Future Work

In the light of experience gained, some future work about the robot system are as follows:

- Recognizability of the artificial landmarks can be combined with the accurate detection of the natural landmarks to form a more robust localization algorithm. By this way, long-time localization can be established with accurate values.

- Use of scale and affine invariant features as natural landmarks can increase the performance of the robot localization algorithm. Recent studies like use of SIFT features for SLAM[32] show the tendency on this subject. Use of reliable features is not only the subject of robot localization but also the subject of object recognition. By following general advances in object recognition, innovative solutions for the more robust features can be found.

- The camera system was fixed on the robot vehicle and views only the front scene of the robot. As stated in the conclusion part, this causes the features quickly go out of the view especially in the turn motion. To use active head and camera systems like in [3], will provide the system to track the features for a long time and improve the performance of the system.

- One of the handicaps of EKF based robot localization algorithms is that it works on unimodal distributions, especially for tracking features this can cause the features to be lost even they are in the scene. Trying different techniques like multi-hypothesis localization and Monte Carlo Localization can improve the performance. See [22] for these techniques.

# Appendix A

# Full Set of The Artificial Landmarks

In this thesis, for robot localization using artificial landmarks, 8 coded cylinders are used. Full set of the coded cylinders are given in Figure A.1. Each coded cylinder is identified by the coding stripes. Coding stripes form a 3-bit coding system from 0 to 7.

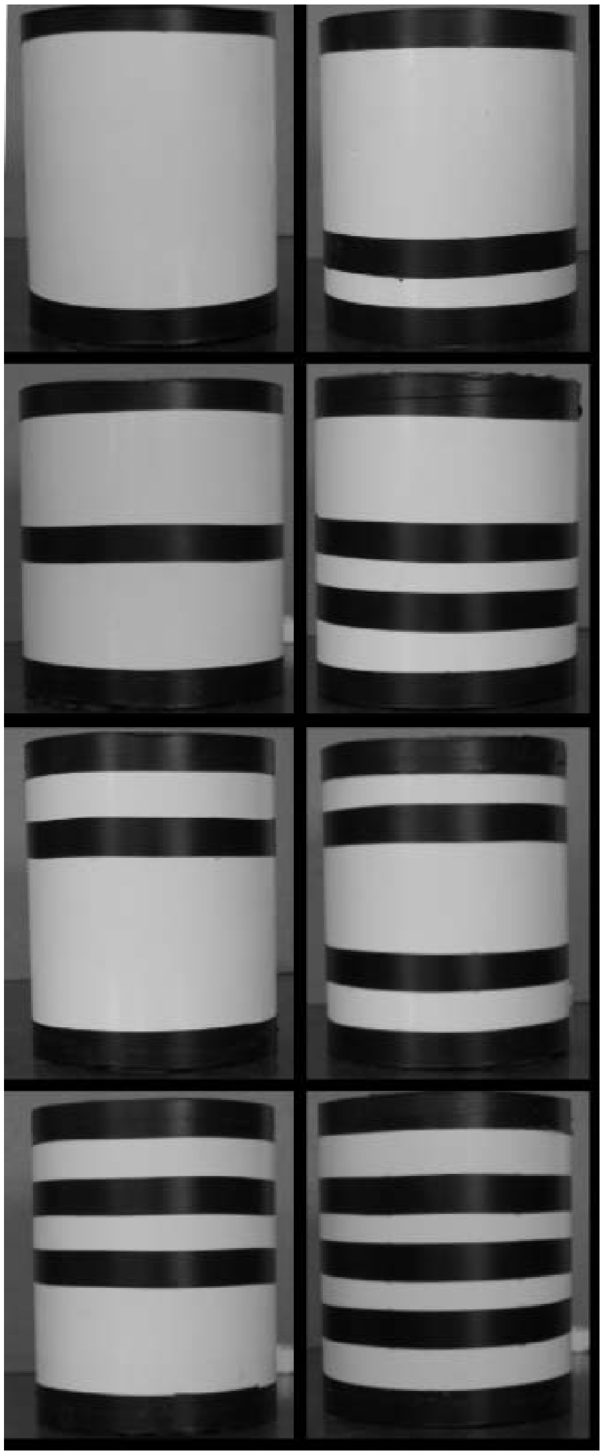


Figure A.1: Full set of landmarks (From top to bottom and left to bottom, in ascending order)

# References

[1] Yaakov Bar-Shalom, X.-Rong Li, and Thiagalingam Kirubarajan. *Estimation with Applications to Tracking and Navigation*. John Wiley & Sons, Inc., 2001.

[2] J.L. Crowley, F. Wallner, and B. Schiele. Position estimation using principal components of range data. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3121–3128, Leuven, Belgium, 1998.

[3] A. J. Davison. *Mobile Robot Navigation Using Active Vision*. PhD thesis, University of Oxford, 1998.

[4] A.J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *Proc. International Conference on Computer Vision*, Nice,France, oct 2003.

[5] Andrew J. Davison and Nobuyuki Kita. Sequential localisation and map-building for real-time computer vision and robotics. *Robotics and Autonomous Systems*, 36(4):171–183, 2001.

[6] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte carlo localization for mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1322–1328, May 1999.

[7] Gregory D.Hager, David Kriegman, Erliang Yeh, and Christopher Rasmussen. Image-based prediction of landmark features for mobile robot navigation. In *IEEE International Conference on Robotics and Automation*, 2002.

[8] M. Drumheller. Mobile robot localization using sonar. *IEEE Trans. Pattern Anal. Mach. Intell.*, 9(2):325–332, 1987.

[9] S. Engelson and D. McDermott. Error correction in mobile robot map learning. In *Proceedings of the 1992 IEEE International Conference on Robotics and Automation*, pages 2555–2560, 1992.

[10] Dieter Fox, Sebastian Thrun, Wolfram Burgard, and Frank Dellaert. Particle filters for mobile robot localization. In Arnaud Doucet, Nando de Freitas, and Neil Gordon, editors, *Sequential Monte Carlo Methods in Practice*, New York, 2001. Springer.

[11] Puneet Goel, Stergios I. Roumeliotis, and Gaurav S. Sukhatme. Robust localization using relative and absolute position estimates. In *IEEE/RSJ International Conference on Robotics and Intelligent Systems*, Kyongju, Korea, 1999.

[12] Javier Gonzalez, Anthony (Tony) Stentz, and Anibal Ollero. An iconic position estimator for a 2-d laser rangefinder. Technical Report CMU-RI-TR-92-04, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, May 1992.

[13] Chris Harris. Geometry from visual motion. pages 263–284, 1993.

[14] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[15] R. Hinkel and T. Knieriemen. Environment perception with laser radar in a fast moving robot. In *Proceedings of Symposium on Robot Control*, pages 68.1–68.7, Karlsruhe, Germany, 1988.

[16] I.Cox. Blanche:an experiment in guidance and navigation of an autonomous robot vehicle. *IEEE Transactions on Robotics and Automation*, 7(2):193–204, 1991.

[17] I.Cox and G.Wilfong. *Autonomous Robot Vehicles*. Springer Verlag, 1990.

[18] İlkay Ulusoy. *Active Stereo Vision:Depth Perception for Navigation, Environmental Map Formation and Object Recognition*. PhD thesis, Middle East Technical University, Turkey, 2003.

[19] j. Leonard, H. Durrant-Whyte, and I. Cox. Dynamic map building for an autonomous mobile robot. *International Journal of Robotics Research*, 11(4):89–96, 1992.

[20] Ramesh Jain, Rangachar Kasturi, and Brian G. Schunck. *Machine vision*. McGraw-Hill, Inc., 1995.

[21] Patric Jensfelt, David Austin, Olle Wijk, and Magnus Andersson. Feature based condensation for mobile robot localization. In *IEEE International Conference on Robotics and Automation*, 2000.

[22] Patrick Jensfelt. *Approaches to Mobile Robot Localization in Indoor Environments*. PhD thesis, Royal Institute of Technology, Sweden, 2001.

[23] Emil Kalman, Rudolph. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[24] A. Kosaka and A.C. Kak. Fast vision-guided mobile robot navigation using model-based reasoning and prediction of uncertainties. *Computer Vision, Graphics, and Image Processing-Image Understanding*, 56(3):271–329.

[25] P. R. Kumar and Pravin Varaiya. *Stochastic systems: estimation, identification and adaptive control*. Prentice-Hall, Inc., 1986.

[26] C. Lin and R. Tummala. Mobile robot navigation using artificial landmarks. *Journal of Robotic Systems*, 14(2):93–106, 1997.

[27] Peter S. Maybeck. *Stochastic Models, Estimation, and Control: Volume 1*. New York: Academic Press, Inc., 1979.

[28] Michael Montemerlo, Sebastian Thrun, Daphne Koller, and Ben Wegbreit. Fastslam: a factored solution to the simultaneous localization and mapping problem. In *Eighteenth national conference on Artificial intelligence*, pages 593–598. American Association for Artificial Intelligence, 2002.

[29] Mustafa Özuysal. Manual and auto calibration of stereo camera systems. Master's thesis, Middle East Technical University, 2004.

[30] Stergios I. Roumeliotis, Gaurav S. Sukhatme, and George A. Bekey. Fault detection and identification in a mobile robot using multiple-model estimation. In *IEEE International Conference on Robotics and Automation*, Leuven, Belgium, May 1998.

[31] D. Scharstein and A.J.Briggs. Real-time recognition of self-similar landmarks. *Image and Vision Computing*, 19:763–772, 2000.

[32] Stephen Se, David Lowe, and Jim Little. Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks. *International Journal of Robotics Research*, 21(8):735–758, August 2002.

[33] Jianbo Shi and Carlo Tomasi. Good features to track. In *1994 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94)*, pages 593 – 600, 1994.

[34] Greg Welch and Gary Bishop. An introduction to the kalman filter. Technical report, 1995.

[35] B. Zitová and J.Flusser. Landmark recognition using invariant features. *Pattern Recognition Letters*, 20(5):541–547, 1999.