A LOW COST STEREO BASED 3D SLAM FOR WEARABLE APPLICATIONS

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

BY

MUSTAFA YASİN ŞAKA

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

DECEMBER 2010

Approval of the thesis:

A LOW COST STEREO BASED 3D SLAM FOR WEARABLE APPLICATIONS

submitted by **MUSTAFA YASİN ŞAKA** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen _____
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. İsmet Erkmen _____
Head of Department, **Electrical and Electronics Engineering**

Assist. Prof. Dr. İlkay Ulusoy _____
Supervisor, **Electrical and Electronics Engineering Dept., METU**

Examining Committee Members:

Prof. Dr. Aydın Alatan _____
Electrical and Electronics Engineering Dept., METU

Assist. Prof. Dr. İlkay Ulusoy _____
Electrical and Electronics Engineering Dept., METU

Assist. Prof. Dr. Erol Şahin _____
Computer Engineering Dept., METU

Assist. Prof. Ece Güran Schmidt _____
Electrical and Electronics Engineering Dept., METU

Assist. Prof. Dr. Afşar Saranlı _____
Electrical and Electronics Engineering Dept., METU

Date: 16.12.2010

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name : Mustafa Yasin ŞAKA

Signature :

ABSTRACT

A LOW COST STEREO BASED 3D SLAM FOR WEARABLE APPLICATIONS

Şaka, Mustafa Yasin
M. S., Department of Electrical and Electronics Engineering
Supervisor: Assist. Prof. Dr. İlkay ULUSOY

December 2010, 141 pages

A wearable robot should know its environment and its location in order to help its operator. Wearable robots are becoming more feasible with the development of more powerful and smaller computing devices and cameras. The main aim of this research is to build a wearable robot with a low cost stereo camera system which explores a room sized unknown environment online and automatically. To achieve 3D localization and map building for the wearable robot, a consistent visual-SLAM algorithm is implemented by using point features in the environment and Extended Kalman Filter for state estimation.

The whole system includes camera models and calibration, feature extraction, depth measurement and Extended Kalman Filter algorithm. Moreover, a map management algorithm is developed. This algorithm keeps the number of features spatially uniform in the scene and adds new features when feature number decreases in a frame. Furthermore, a user-interface is presented so that the location of the camera,

the features and the constructed map are visualized online. Most importantly, the system is conducted by a low-cost stereo system.

Keywords: Wearable Robot, Robot Localization, Map Building, 3D SLAM

ÖZ

GİYİLEBİLİR UYGULAMALAR İÇİN DÜŞÜK MALİYETLİ KAMERALAR İLE ÜÇ BOYUTLU EŞ ZAMANLI KONUMLANDIRMA VE HARİTALAMA

Şaka, Mustafa Yasin
Yüksek Lisans, Elektrik Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Yard. Doç. Dr. İlkay ULUSOY

Aralık 2010, 141 sayfa

Giyilebilir bir robot onu kullanana yardım etmek için dış dünyayı ve lokalizasyonunu bilmek durumundadır. Daha güçlü işlemcilerin ve daha ufak kameraların üretimindeki gelişmeler giyilebilir robotları daha cazip hale getirmiştir. Bu araştırmanın ana amacı oda boyutunda bilinmeyen bir ortamı otomatik ve çevirim içi olarak keşfeden düşük maliyetli bir stereo kamera sistemi ile oluşturulmuş bir giyilebilir robot oluşturmaktadır. Giyilebilir bir robotta, üç boyutlu lokalizasyon ve harita çıkartımı sağlamak için, çevrede bulunan nokta işaretlerini kullanan ve durum kestirimi için genişletilmiş Kalman filtresini (EKF – Extended Kalman Filter) kullanan tutarlı görsel eş zamanlı harita çıkarımı ve yer kestirimi (SLAM – Self Localization and Mapping) algoritması yapılmıştır.

Tüm sistem, kamera modellerini ve kalibrasyonunu, derinlik ölçümünü ve genişletilmiş Kalman filtresini içermektedir. Buna ek olarak harita yönetim algoritması geliştirilmiştir. Bu algoritma çevredeki işaretlerin dengeli dağılımını ve

çevrede işaret sayısı azalınca yeni işaretler eklemeyi sağlamaktadır. Ayrıca kameranın yerini, işaretleri ve çıkartılan haritayı canlı gösteren bir kullanıcı arayüzü oluşturulmuştur. En önemlisi, bu sistem düşük maliyetli bir stereo sistem ile çalışmaktadır.

Anahtar Kelimeler: Giyilebilir Robot, Robot Lokalizasyonu, Harita Çıkarma, 3 boyutlu SLAM

To My Family

ACKNOWLEDGEMENTS

I would like to thank to my supervisor Dr. İlkay Ulusoy for her guidance, advice, criticism, encouragement and insight throughout the completion of the thesis.

I am also grateful to TAI Inc. for the facilities that made my work easier.

I also thank to my father Ziya Şaka and mother Hediye Şaka to help me in my thesis.

I also thank to TUBİTAK BIDEB for its financial support during the preparation of the thesis.

I also thank to Erol Oğuz to provide convenience during the completion of the thesis.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

1.1 Motivation and Problem Definition

A wearable robot gives opportunity to sense the world from a first-person's perspective. The wearable robot moves and observes the scene attended by the wearer. A wearable robot can both observe the actions of a wearer and the state of the environment and help to wearer in various domains. It can help the wearer to keep the track of tools and materials. Also it can provide warnings for dangerous situations or objects and it can help with a work plan. Moreover a wearable robot can be helpful with a remote human expert. A remote expert can help the wearer with information about environment.

To help the wearer, a wearable robot should know its place in the environment with respect to the objects around and build a consistent map. Using only visual sensors is very appealing since wearer and remote expert directly use the measurements of the objects. For a wearable robot, simultaneous localization and mapping (SLAM) is used in order to know its place with respect to the environment. SLAM is the task for estimation of both motion and structure from sensor observations in an unknown environment. SLAM is the process by which a mobile robot answers two primary questions "Where am I?" and "What is the structure of the environment?" The robot requires the answers continuously because navigation, building map and other decisions depend on them. Robot sensors provide the information for SLAM algorithm. In this thesis a stereo low cost camera system is used as the only sensor,

pose and placement of the camera system is calculated by using the geometry in nature.

1.2 Previous Studies

For a mobile robot, it is impossible to move without knowing its environment so it has to know its location and move accordingly. In order to localize itself, robot has to know the map of the environment. Sometimes maps are pre-known but generally robot has to build its own map in an unknown environment. Davison et al. [12] built a wearable robot which builds a consistent map in indoor environments by using Visual Mono SLAM and using an IEEE-1394 camera system. This wearable robot provides the opportunity to sense the world from a first-person perspective. The camera sensor moves with the wearer and camera observes the places attended by the wearer. Moreover system works in workspace localization with several cubic meters and robot helps the wearer in his/her tasks. Also there are other usage areas of wearable robots. In [19], hand gesture recognition is used for keyboard and mouse based input. Wearable robot estimates the direction of hand in real time. As a result, camera system estimates the focus of user attention and control redirections of gaze of a wearable active camera. Moreover in the wearable domain, hand gesture recognition is a natural replacement for keyboard and mouse-based input. In [30], for example, a hat-mounted camera is used for a sign language recognition task. Also in [31], a bare hand is used as a cursor-and-click device for interacting with menus displayed on a head mounted display. Additionally, pointing gestures are the main form of non-verbal communication, presenting a major complement to speech in human to human communication [32]. In [18], to improve the localization of the wearable robots, known planar objects are used with point features. This is done by using SIFT [27] features.

In this thesis, SLAM process works in 3D environment. 3D SLAM is achieved also in [2], [3], [11], [16], [29]. In 2D SLAM state of the system is generally consists of

6 variables [6] but in 3D SLAM, this number increased to 13 [2], [29]. SLAM Process can be achieved by using stereo camera systems [3], [11], [13], [14]. In [3], SLAM system is built by using stereo active cameras. Active cameras follow features in defined ranges. Moreover in this paper some remarkable implementations used in this thesis are mentioned. Some of them are listed as follows: In order to achieve long-term stability of maps, the image patch intensities are saved and never changed in order to not give a change to drift in patches. Moreover frames are searched only when there is a need for new features. In [13], by using stereo camera system, 3D SLAM system is achieved. Also by using a mono camera system SLAM can be achieved [2], [16], [23], [11], [22]. Scale observability is the main problem for monocular approaches in SLAM process. In order to handle this, pre-known features may be used [2] or features should be initialized by using consecutive frames [15]. Additionally, EKF based solutions are generally used in SLAM algorithms with visual sensors [2]. However other methods like particle filter approaches give good and robust results [22]. In SLAM algorithms generally point features are used. However in order to improve the robustness of the algorithms features like SIFT features [27] or fast-corner detectors [16] can be used. Also using both mono camera advantages with stereo camera advantages are more effective then only using one of them. Using the advantages of mono-vision (bearing-only, with infinity range but no 3D instant information) with stereo vision (3D information up to a limited range) gives robust and accurate estimations [17], [14].

Visual SLAM algorithms are used in indoor environments [3] [2] also in outdoor environment [13] [11]. Generally in outdoor environment, SLAM algorithms need good mapping strategies because of the large number of features. As features are added to the state vector of SLAM algorithm the computational cost increases. Because of the limited time for online process, the feature number should be limited up to some point. As a result, SLAM systems generally work in indoor environments where the working environment does not exceed a few cubic meters. In order to handle this problem Hierarchical SLAM is proposed [29]. Hierarchical

SLAM builds independent local submaps and a higher map which coordinates these submaps. When the current map reaches the maximum number of features, it is freezed and a new local map is initialized to handle the limited size of the SLAM systems. Only the feature positions and their distances passed to the new local map to maintain statistical independence between maps. No information is transferred from the previous map to the new one. By using Hierarchical Map technique, loop closure is achieved near real time by using a stereo camera system [11]. For now, large-scale real-time visual mapping of outdoors, buildings are unfeasible because the computational complexity of the SLAM algorithms grows with map size. So real-time working is limited by some map size [12].

SLAM system can be used for many purposes. For example by using Visual Mono SLAM system a visual compass is presented. It is in real time and copes with the challenging conditions: hand-held camera, varying natural outdoor illumination, low cost camera and people moving in the scene. [23] Also by using Mono SLAM system, a Humanoid Robot close loops and builds a consistent map in an indoor environment [24].

1.3 Scope of Thesis

The main purpose of the thesis is to build a consistent Visual Stereo SLAM Algorithm by using a stereo camera system which consists of two low cost webcams and this camera system should be moveable with hand, robot etc. System input should be only visual and a consistent map and self localization should be achieved by observing the environment.

1.4 Thesis Contribution

The main contribution of this thesis is to construct a consistent SLAM algorithm based on Extended Kalman Filter by using a low cost stereo system. The principal contributions are listed as follows:

• Stereo SLAM Algorithm is presented with a low-cost camera system which is built by two webcams with narrow-angle. These cameras can be bought from any store. Stereo SLAM Algorithm is applicable to any stereo camera pair if the calibration parameters of the camera system are known. These parameters can easily be calculated by Camera Calibration Procedure [4].

• To work with Stereo System there is no need for a complicated hardware but only two cameras and a computer with 2 USB inputs are sufficient.

1.5 Implementations

In order to build the Stereo SLAM Algorithm with low-cost cameras, the tasks listed below are implemented.

• An online vision-based SLAM system is implemented. SLAM algorithm acquires frames and processes them simultaneously then it constructs a map and localizes robot on this map.

• A feature detection algorithm is implemented. By using a stereo system, the distances of the features are calculated with the help of triangulation. So there is no need for pre-known features to start the algorithm.

• A map management strategy is implemented which enables SLAM to achieve long-term runs by adding good features and deleting bad features.

• A Matlab user interface which shows the camera state and the current frame simultaneously is implemented.

• A movable stereo system is implemented. It is easy to move the system and it can be carried by one hand easily. Also it can be mounted to any robot.

• Experiments on Stereo SLAM Algorithm are conducted, analyses are made and results are presented.

1.6 Content of the Thesis

The content of the thesis can be divided to three main parts. These are initialization of the system, feature search algorithm and SLAM process. In initialization of the system, firstly the camera parameters are initialized which are calculated from camera calibration procedure. Then initialization of state vector, covariance matrix and cameras are performed. The camera calibration procedure and the initialization of the cameras are presented in chapter 4.

Secondly, new features to start the SLAM algorithm are initialized by Feature Search Algorithm. Feature Search Algorithm tries to keep feature numbers around a predetermined value. The features are added or deleted from the state vector. Features are found by the help of the Harris Corner Detector which is presented in section 2.6. Then features are matched by their stereo pairs and their distances are found by triangulation process which is discussed in section 2.5. These features are added to the state vector. The addition of the features to the state vector is described in section 3.7.

The details of SLAM process is hold in chapter 3. SLAM process can be divided into three main parts; prediction, matching and update steps. In the prediction

phase, the camera pose is estimated by using the previous velocity and acceleration estimations of the camera system. Furthermore, potential feature locations are predicted by using the previous velocities and accelerations of the camera system. In the matching phase, the feature matching process is applied. In the update phase, the state vector and covariance of the camera system are updated with the help of the prediction and estimation results.

Finally, basic information about camera model and lens distortion used in the thesis is presented in chapter 2. Additionally, the Normalized Cross Correction which is used for matching process between frames and the quaternions used for camera pose calculations are presented in chapter 2.

1.7 Thesis outline

Chapter 1: Explanation of the SLAM problem and outline of the thesis is given.

Chapter 2: Pinhole camera model, lens distortion model applied to pinhole camera model, two camera geometry, triangulation, image processing which includes Harris Corner Detector and matching process of features are presented. Additionally a brief introduction to quaternions is given.

Chapter 3: Mathematical details of SLAM algorithm are explained in this section. Additionally a brief introduction to Extended Kalman Filter (EKF) Algorithm is given.

Chapter 4: System components and its properties, general process including calibration of cameras, feature initialization algorithms, GUI, SLAM algorithms used in this thesis are listed and explained in this section.

Chapter 5: Experiments and results are given in this section.

Chapter 6: A brief summary about the process, conclusion and possible future works are given.

CHAPTER 2

CAMERA AND IMAGE PROCESSING

In this chapter basic information about camera model and image processing used in this thesis will be presented. Firstly in Section 2.1 Pinhole camera model will be explained. In section 2.2 and 2.3 a simple lens distortion model combined with pinhole camera model will be discussed. In section 2.4 and 2.5 two camera geometry and triangulation which is used to calculate the distances of features in the environment will be discussed. In 2.6 a brief introduction to features, in 2.7 the basics of Harris Corner Detector and in 2.8 feature matching concepts will be discussed. Finally in section 2.9 a brief introduction to quaternions is given.

2.1 Pinhole Camera [9]

An idealized pinhole camera C is composed of a center of projection $C \in R^3$, a focal length $f \in R^+$ and an orientation matrix R. The center and the orientation of camera are described with respect to a world coordinate system on R^3.

A point expressed in the world coordinate system as $X= (X_0 ,Y_0, Z_0)$ can be expressed in the camera coordinate system C as

$$\mathbf{X} = \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = R\left(\begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \end{bmatrix} - C\right) \tag{2-1}$$

The purpose of the camera is to capture a two-dimensional image of a three dimensional scene S, that is a collection of points in R^3. This image is produced by perspective projection as follows: Each camera C has an associated image plane P located in the camera coordinate system at Z_c=f. The pinhole camera system is shown in figure 2-1. Two dimensional plane is referred as the pinhole plane. The pinhole is the origin of the camera coordinate system which is also known as optical center of camera. The coordinate system is left handed which means X_C points side, Y_c points up and Z_c points in view direction. X_C and Y_C build the pinhole plane and Z_C is called as principal axis. The image plane is parallel to the pinhole plane and located at distance f. The focal length f, is the distance between the image plane and the pinhole plane. A scene point X = (X_C Y_C Z_C) is projected onto the image plane P at point (u_i, v_i) by the perspective projection equations.

$$u_i = f\frac{X_c}{Z_c} \quad v_i = f\frac{Y_c}{Z_c} \tag{2-2}$$

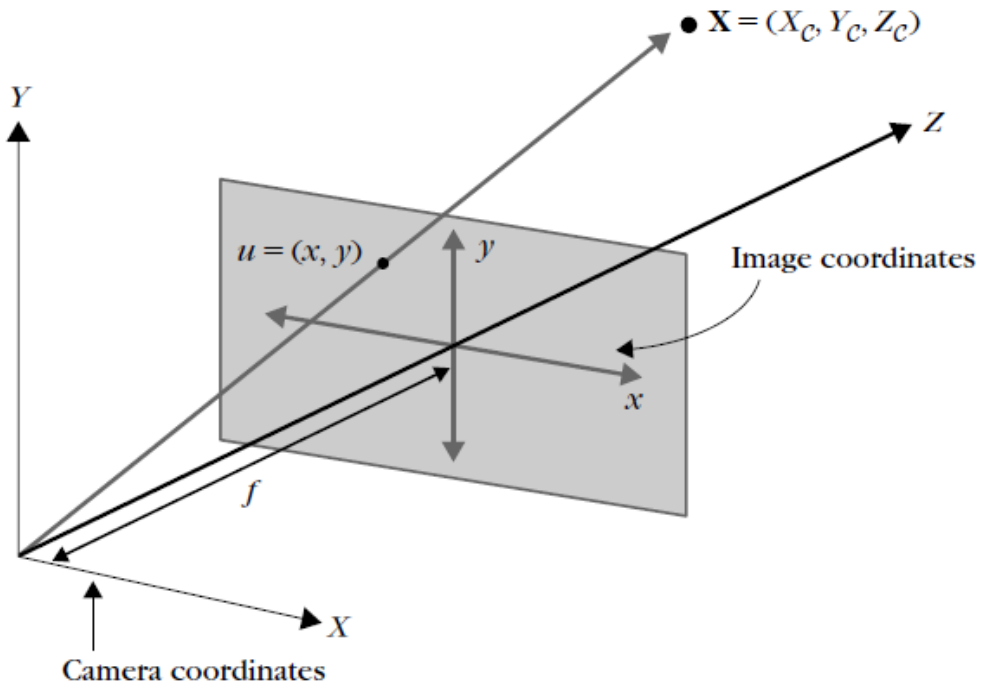

Figure 2-1 Pinhole Camera

2.2 Adaptation of the Basic Model [8]

Today most of the cameras use photographic lenses instead of a pinhole. We have to model lens imperfections to use the basic pinhole camera model. In pinhole camera model, every point $P_i = (x_i, y_i, z_i)^T$, $z_i > 0$ can be projected on the image plane but in reality the cameras have limited size of view since they have limited image sensor area where the points are projected on. The boundaries of size of view are given below.

$$\begin{pmatrix} \alpha_u \\ \alpha_v \end{pmatrix} = 2 \begin{pmatrix} \arctan\left(\frac{width \cdot d_u}{2f} \right) \\ \arctan\left(\frac{height \cdot d_v}{2f} \right) \end{pmatrix} \tag{2-3}$$

where f is the focal length. The parameters *width* and *height* state the image resolution in pixels, d_u and d_v give the physical width and height of a pixel on image sensor. Limitation in the field of view also affects the size and resolution of the created image. Since the origin of images in computer applications is located in the top left corner of the image, this fact is considered in the model by shifting the origin of the image coordinate system. The resulting coordinates of the pixels $(u_i, v_i)^T$ are given as follows:

$$\begin{pmatrix} u_i \\ v_i \end{pmatrix} = \begin{pmatrix} \lfloor ku_i + o_u + 0.5 \rfloor \\ \lfloor lv_i + o_v + 0.5 \rfloor \end{pmatrix} = \begin{pmatrix} \left\lfloor kf \frac{x_i}{z_i} + o_u + 0.5 \right\rfloor \\ \left\lfloor lf \frac{y_i}{z_i} + o_v + 0.5 \right\rfloor \end{pmatrix} \tag{2-4}$$

where o_u and o_v give the displacement of the upper left corner of the image from its center in pixel units. If a camera resolution is given as width × height then o_u and o_v are calculated as follows:

$$o_u = \frac{width}{2} \qquad o_v = \frac{height}{2} \tag{2-5}$$

The dimensions of pixels are denoted as du×dv where du and dv are measured in meters. So equation 2-4 becomes

$$\begin{pmatrix} u_i \\ v_i \end{pmatrix} = \begin{pmatrix} \left\lfloor \frac{f}{d_u} \frac{x_i}{z_i} + o_u + 0.5 \right\rfloor \\ \left\lfloor \frac{f}{d_v} \frac{y_i}{z_i} + o_v + 0.5 \right\rfloor \end{pmatrix} = \begin{pmatrix} \left\lfloor f_u \frac{x_i}{z_i} + o_u + 0.5 \right\rfloor \\ \left\lfloor f_v \frac{y_i}{z_i} + o_v + 0.5 \right\rfloor \end{pmatrix} \tag{2-6}$$

Parameters k and l are the scale values to map the distance obtained by $f\frac{x_i}{z_i}$ and $f\frac{y_i}{z_i}$ to the corresponding pixels and they are replaced by 1/du and 1/dv. f_u and f_v are the focal length f in pixel units. f_u and f_v are obtained by camera calibration procedure which is described in section 4.2.

2.3 A Simple Distortion Model [8]

In equation 2-6, the center of the image sensor is assumed to be placed ideally. In reality, this property does not hold. If the displacement of the center of the image sensor is known, the equation 2-6 becomes

$$\begin{pmatrix} u_{d,i} \\ v_{d,i} \end{pmatrix} = \begin{pmatrix} \left\lfloor f_u \frac{x_i}{z_i} + u_0 \right\rfloor \\ \left\lfloor f_v \frac{y_i}{z_i} + v_0 \right\rfloor \end{pmatrix} \tag{2-7}$$

where u_0 and v_0 denote the image coordinate of the image center R. $(u_{d,i}, v_{d,i})^T$ is called distorted image coordinates since we still do not model the lens distortion even we calculate them according to actual image center.

Distortions occur due to lens system of the camera. The distortion model gives how to calculate undistorted image coordinates $(u_{0,i}, v_{0,i})^T$ from distorted image coordinates $(u_{d,i}, v_{d,i})^T$ for a given point. In this thesis a simplified "Brown-Conrady model" is used [4]. The model has two kinds of distortions; radial distortion and tangential distortion. Because of the shape of the lens, radial distortions occur which cause the straight lines to be projected in a curved fashion. Tangentials occur because of imperfections in the centering of the camera lens. "Brown-Conrady

model" uses 5 distortion coefficients. Three of them (k_1, k_2, k_3) belong to radial distortion; two of them (p_1, p_2) belong to tangential distortion.

In order to model the distortion, it is recommended to take the distortion coefficients zero beyond 4th order [4]. In radial distortion model, k_1, k_2 and k_3 give 2nd, 4th and 6th distortion coefficients respectively. So k_3 is taken as 0. Moreover tangential distortions are dominated by radial distortions. So we can assume $p_1=p_2=0$. So distortion model is simplified to

$$\begin{pmatrix} u_{u,i} \\ v_{u,i} \end{pmatrix} = \begin{pmatrix} (u_{d,i} - u_0)(1 + k_1 r^2 + k_2 r^4) \\ (v_{d,i} - v_0)(1 + k_1 r^2 + k_2 r^4) \end{pmatrix} + \begin{pmatrix} u_0 \\ v_0 \end{pmatrix} \tag{2-8}$$

where

$$r = \sqrt{\left(\frac{1}{f_u}(u_{d,i} - u_0)\right)^2 + \left(\frac{1}{f_v}(v_{d,i} - v_0)\right)^2} \tag{2-9}$$

Distorted coordinates of a point are calculated from undistorted coordinates by the following formula:

$$\begin{pmatrix} u_{d,i} \\ v_{d,i} \end{pmatrix} = h_d \begin{pmatrix} u_{u,i} \\ v_{u,i} \end{pmatrix} = \begin{pmatrix} u_0 + \dfrac{(u_{u,i} - u_0)}{(1 + k_1 r_d^2 + k_2 r_d^4)} \\ v_0 + \dfrac{(v_{u,i} - v_0)}{(1 + k_1 r_d^2 + k_2 r_d^4)} \end{pmatrix} \tag{2-10}$$

with

$$r_u = r_d (1 + k_1 r_d^2 + k_2 r_d^4) \tag{2-11}$$

$$r_u = \sqrt{(d_u(u_{u,i} - u_0))^2 + (d_v(v_{u,i} - v_0))^2} \tag{2-12}$$

r_d in equation (2-11) is calculated by Newton-Raphson iteration, where a couple of iterations is enough to calculate and given below.

$$r_{dn+1} = r_{dn} - \frac{g(r_{dn})}{g'(r_{dn})} = r_{dn} - \frac{r_{dn} + k_1 r_{d_n}^{\ 3} + k_2 r_{d_n}^{\ 5} - r_u}{1 + k_1 r_{d_n}^{\ 2} + k_2 r_{d_n}^{\ 4}}, \quad r_{d0} = r_u \tag{2-13}$$

2.4 Two Camera Geometry [9]

In this section the image relationship resulting from the same static scene being imaged by two cameras C and C' will be discussed. These cameras can be two physically separate cameras or a single camera moving at different points in time. Let the scene coordinates of a point P in the C coordinate system be (x_i, y_i, z_i) and in C' coordinate system be (x'_i, y'_i, z'_i). The corresponding image coordinates are given as $u=(u_i, v_i)$, $u'=(u'_i, v'_i)$. The points u and u' are said to be corresponding points and the pair (u,u') is called a point correspondence. We assume that two cameras are related by a rigid motion, which means that C' coordinate system can be expressed as a rotation $\hat{R}$ of the C coordinate system followed by a translation $[t_x, t_y, t_z]^T$ that is

$$\begin{vmatrix} x'_i \\ y'_i \\ z'_i \end{vmatrix} = \hat{R} \begin{vmatrix} X \\ Y \\ Z \end{vmatrix} + \begin{vmatrix} t_x \\ t_y \\ t_z \end{vmatrix} \tag{2-14}$$

$\hat{R}$ is given as

$$\hat{R} = \begin{pmatrix} \cos\alpha\cos\gamma + \sin\alpha\sin\beta\sin\gamma & \cos\beta\sin\gamma & -\sin\alpha\cos\gamma + \cos\alpha\sin\beta\sin\gamma \\ -\cos\alpha\sin\gamma + \sin\alpha\sin\beta\cos\gamma & \cos\beta\cos\gamma & \sin\alpha\sin\gamma + \cos\alpha\sin\beta\cos\gamma \\ \sin\alpha\cos\beta & -\sin\beta & \cos\alpha\cos\beta \end{pmatrix} \quad (2\text{-}15)$$

where α, β and γ are rotation angles around X- , Y- and Z- axes. Axes and transformation matrix elements will be calculated by calibration procedure in section 4.2.

2.5 Triangulation [8]

Triangulation method is used to obtain 3D information from camera images. In order to find 3D information, at least 2 images taken from different positions are necessary. Assume that there are two calibrated identical cameras which observe the same scene. The cameras are positioned such that their pinhole planes are coplanar and their optical centers are apart by a known distance b, also referred as baseline in stereo vision context. Moreover cameras are oriented in the same direction. A point $P_i = (x_i, y_i, z_i)^T$ is projected on cameras at pixel points $\varphi_i^l = (u_i^l, v_i^l)^T$ and $\varphi_i^r = (u_i^r, v_i^r)^T$ (l:left, r:right) as depicted in Figure 2-2. It is assumed that $v_i^l = v_i^r$ holds. By these two projections, the depth of P_i can be calculated. According to the intercept theorem, the following relation is obtained:

$$\frac{z_i - f}{z_i} = \frac{\overline{Q_i^l Q_i^r}}{b} = \frac{b - (u_i^l - u_i^r)}{b} \quad (2\text{-}16)$$

with $f = f^l = f^r$. From (2-16)

$$\frac{b-(u_i^l-u_i^r)}{z_i-f}=\frac{b}{z_i} \Rightarrow \quad z_i=\frac{bf}{u_i^l-u_i^r}=\frac{bf}{\Delta u_i} \tag{2-17}$$

with $\Delta u_i = u_i^l - u_i^r$, baseline b.

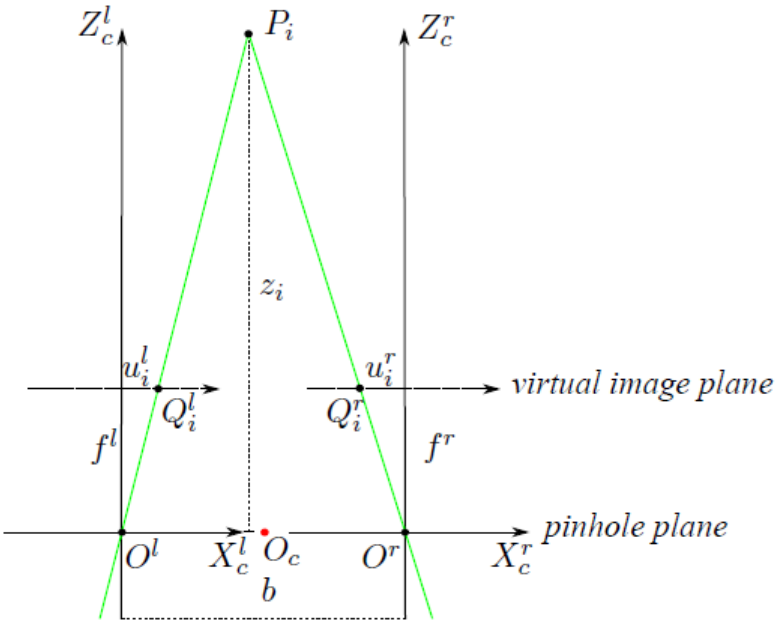

Figure 2-2 Triangulation

2.6 Harris Corner Detector [1]

In order to find 3D information from the scene with triangulation by using the camera position and orientation, correspondences between two projections $\varphi_i^l = (u_i^l, v_i^l)^T$ and $\varphi_i^r = (u_i^r, v_i^r)^T$ of a point P_i should be established. In remaining subchapters how these correspondences are established will be discussed.

In this thesis the point features are used and to find these features Harris Corner Detector is used. Harris Corner Detector is an isotropic detector which uses a weighted sum of squared distances (SSD_w) on a Gaussian smoothed image. The

SSD_w over an image patch p with dimensions (p_w+1 × p_h+1) is compared to a patch of the same size shifted by (x, y) in direction of the u and v axes of the image and its formula is given as

$$SSD_w(x,y) = \sum_{u=0}^{p_w} \sum_{v=0}^{p_h} w(u,v)(I(u,v) - (I(u+x, v+y))^2 \tag{2-18}$$

where I(u,v) is the gray-scale value at image position (u,v) and w(u,v) is the corresponding weight. The Taylor expansion of I(u+x,v+y) is calculated by

$$I(u+x, v+y) \approx I(u,v) + x\frac{\partial I}{\partial u} + y\frac{\partial I}{\partial v} \tag{2-19}$$

where $\partial I / \partial u$ and $\partial I / \partial v$ are partial derivatives. The partial derivatives are calculated as

$$\frac{\partial I}{\partial u} = I \otimes \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \tag{2-20}$$

$$\frac{\partial I}{\partial v} = I \otimes \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} \tag{2-21}$$

where $I \otimes k$ indicates the convolution of the image with the kernel k. So we can write equation 2-18 as

$$
\begin{aligned}
SSD_w(x,y) &\approx \sum_{u=0}^{p_w}\sum_{v=0}^{p_h} w(u,v)\left(x\frac{\partial I}{\partial u}+y\frac{\partial I}{\partial v}\right)^2 \\
&= \begin{pmatrix} x & y \end{pmatrix} \sum_{u=0}^{p_w}\sum_{v=0}^{p_h} w(u,v) \begin{bmatrix} \left(\frac{\partial I}{\partial u}\right)^2 & \frac{\partial I}{\partial u}\frac{\partial I}{\partial v} \\ \frac{\partial I}{\partial u}\frac{\partial I}{\partial v} & \left(\frac{\partial I}{\partial u}\right)^2 \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \\
&= \begin{pmatrix} x & y \end{pmatrix} H \begin{pmatrix} x \\ y \end{pmatrix}
\end{aligned}
\tag{2-22}
$$

where H is called the Harris matrix. If two eigenvalues of H; λ_1, λ_2 are greater than zero, a corner is detected. To avoid computation of the eigenvalues, the following interest measure R is used.

$$
R = \lambda_1\lambda_2 - \kappa(\lambda_1+\lambda_2)^2 = \det(H) - \kappa\, trace(H)^2 \tag{2-23}
$$

where K is constant and has to be determined empirically. If R>>0 is observed then a corner is detected. In order to find the best features in the frame, locally detected best corners are sorted with respect to their interest measure and the corners with highest score are taken.

2.7 Feature matching [8]

After finding the interest points in an image, a comparison should be perfomed in order to establish a correspondence between two images taken from two cameras or taken from one camera in preceding frames. Feature matching is used for two reasons: Firstly to find a stereo pair of features taken from left and right camera. Secondly to measure the distance traveled by stereo system we should find the pair of a feature in the new frame of the left camera. For this purpose, a comparison function C is needed which gets two patches and indicates whether these images

similar or not. In this thesis 11x11 image patches are used because it is neither large to increase the computational time nor small to lead wrong matches. In this study, firstly normalized Sum of Squared differences (SSD) is used. The SSD is given as

$$R(x,y)=\sum_{x',y'}\frac{(T(x',y')-I(x+x',y+y'))^2}{\sqrt{\sum_{x',y'}T(x',y')^2\sum_{x',y'}I(x+x',y+y')^2}} \tag{2-24}$$

SSD has problems in patches with different illumination. For example two patches are given as $I'(u,v)=I(u,v)+k \quad \forall u,v \quad k\in Z$. In this situation two patches indicate the same image in different illumination condition but C_{ssd} indicates a difference between I and I′. Moreover C_{ssd} changes with different patch size and also it is necessary to specify a threshold for different size and environments. In order to tackle with these problems, Normalized Cross-Correlation (NCC) is used for comparison function. NCC is given as

$$C_{NCC}(I,I')=\frac{1}{n}\sum_{u}\sum_{v}\frac{(I(u,v)-\bar{I})\,(I'(u,v)-\overline{I'})}{\sigma_I\sigma_{I'}} \tag{2-25}$$

where $\bar{I},\bar{I}'$ are the mean values of given patches, n is the number of pixels in the image patches, σ_I and $\sigma_{I'}$ are the standard deviations of the patches I and I′. The range values of NCC is given as C_{NCC} (I, I′) → [-1, 1] where 1 gives the perfect patch.

2.8 Quaternions [20]

Quaternions are used to model the camera pose. The camera pose consists of 3D position (x, y, z) and quaternion with 4 parameters (q_o q_x q_y q_z) which are used to

represent orientation. Any 3D rotation can be described by a single rotation around an approximately placed axis. In a quaternion, a unit vector u representing the axis of this rotation and its angular magnitude θ are stated as follows:

$$\begin{pmatrix} q_0 \\ q_x \\ q_y \\ q_z \end{pmatrix} = \begin{pmatrix} \cos \theta/2 \\ u_x \sin \theta/2 \\ u_y \sin \theta/2 \\ u_z \sin \theta/2 \end{pmatrix} \tag{2-26}$$

The properties of quaternions which are used in the thesis are summarized below. The magnitude of a quaternion is equal to 1, that is

$$q_0^2 + q_x^2 + q_y^2 + q_z^2 = 1 \tag{2-27}$$

Quaternion q has a conjugate $\overline{q}$, where $\overline{q}$ has a rotation around the same axis but with negative magnitude.

$$q = \begin{pmatrix} q_0 \\ q_x \\ q_y \\ q_z \end{pmatrix}, \quad \overline{q} = \begin{pmatrix} q_0 \\ -q_x \\ -q_y \\ -q_z \end{pmatrix} \tag{2-28}$$

Rotation R which gives the same rotation with q is defined as

$$R_v = q \times v \times \overline{q} \tag{2-29}$$

where v is an arbitrary 3x1 column vector. From this equation R is calculated as

$$R = \begin{bmatrix} q_0^2 + q_x^2 - q_y^2 - q_z^2 & 2(q_x q_y - q_0 q_z) & 2(q_x q_z + q_0 q_y) \\ 2(q_x q_y + q_0 q_z) & q_0^2 - q_x^2 + q_y^2 - q_z^2 & 2(q_y q_z - q_0 q_x) \\ 2(q_x q_z - q_0 q_y) & 2(q_y q_z + q_0 q_x) & q_0^2 - q_x^2 + q_y^2 + q_z^2 \end{bmatrix} \tag{2-30}$$

If q_1 represents the rotation R^{AB} and q_2 represents the rotation R^{BC}, then the composite rotation is represented by product of the two quaternions.

$$R^{AC} = R^{AB} R^{BC} \tag{2-31}$$

$$q_3 = q_1 \times q_2 = \begin{pmatrix} q_{10} q_{20} - \left(q_{1x} q_{2x} + q_{1y} q_{2y} + q_{1z} q_{2z} \right) \\ q_{10} \begin{pmatrix} q_{2x} \\ q_{2y} \\ q_{2z} \end{pmatrix} + q_{20} \begin{pmatrix} q_{1x} \\ q_{1y} \\ q_{1z} \end{pmatrix} + \begin{pmatrix} q_{1y} q_{2z} - q_{2y} q_{1z} \\ q_{1z} q_{2x} - q_{2z} q_{1x} \\ q_{1x} q_{2y} - q_{2x} q_{1y} \end{pmatrix} \end{pmatrix} \tag{2-32}$$

CHAPTER 3

SLAM ALGORITHM

After the basic information about camera model and image processing, in this chapter the details of SLAM algorithm will be explained.

Firstly an introduction to Extended Kalman Filter will be given in Section 3.1. Then we will explain the state representation in Visual SLAM algorithm. In section 3.3 prediction function is explained. By prediction function, camera pose prediction by using velocities and accelerations will be estimated. In section 3.4 the measurement prediction will be presented. Measurement function predicts feature locations in the acquired frame by the movement of camera. In section 3.5, actual matching algorithm between preceding frames will be discussed. The update of the state by using prediction and measurement function will be presented in section 3.6. In section 3.7, initialization of the features will be discussed. Finally in 3.8, deletion of the features is explained.

Visual Stereo SLAM builds a consistent 3D map of the environment by using two low cost cameras. The pose of the camera is estimated in a static environment where there should not be any movement like walking people etc. Repeated observation of the features and comparison of the projection locations with the expected locations contribute to both camera pose and point feature estimation. The underlying method is triangulation. Initial depth information is formed by observing the point features by two cameras. After initialization of the features, triangulation

process is handled by EKF. Since point features are connected by covariance matrix in EKF, one measurement of a feature might improve the localization estimation of the features which are not observed at that frame. This process will be presented in this chapter.

3.1 Extended Kalman Filter Application [6]

In this chapter a brief introduction to Extended Kalman Filter (EKF) will be presented. EKF algorithm is the underlying mechanism to estimate pose and feature positions in Visual Stereo SLAM.

The Extended Kalman Filter is an extension of the Kalman Filter used to model non-linear systems. The Kalman Filter is a well known and widely used recursive Gaussian Filter to estimate the state of continuous linear systems under uncertainty. The state vector $\mathbf{x}_t$ of a system is modeled by a multivariate Gaussian distribution with mean μ_t and covariance $\sum_t$, at time t. In Stereo SLAM, the state consists of 3D position of camera in world coordinate system, its orientation, velocities and the estimated positions of all observed features. The system will be observed at discrete points in time, where the current time is represented as t, and the previous steps are represented as t-1, t-2 … In each time, state may be influenced by a set of actions denoted by a_t. Moreover in each time step, there are sensors with measurement functions *h* which get noisy measurements z_t at time t. The step of EKF algorithm consists of two phases: Prediction and update. In prediction phase, the state that the system is transferred to after the execution of the actions a_t is predicted. This prediction is made by the transition function $g(a_t, \mu_{t-1})$. Once the predicted state $(\overline{\mu_t}, \overline{\Sigma_t})$ is calculated, the measurement function $h(\overline{\mu_t})$ has to be built. After the measurement z_t is taken, the measured state of the system is compared with the predicted state. Both the prediction and measurement influence the new estimated state $(\mu_t, \sum_t)$. Algorithm 3.1 shows the whole process briefly. We can observe

measurement z_t and the actions a_t, but the actual state $\mathbf{x}_t$ cannot be observed but estimated only by z_t and a_t. This part is shown in Figure 3-1.

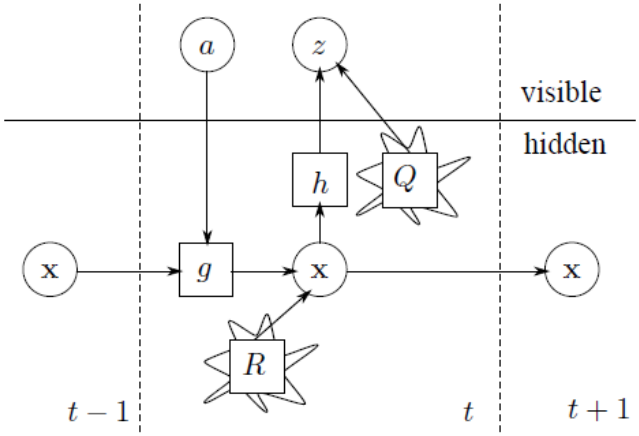

Figure 3-1 EKF Sequence

Algorithm 3.1: Extended Kalman Filter (μ_{t-1},$\sum_{t-1}$,a_t,z_t)

1. $\bar{\mu}_t = g(a_t, \mu_{t-1})$
2. $\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + R$
3. $K_t = \Sigma_t H_t^T \left(H_t \bar{\Sigma}_t H_t^T + Q_t\right)^{-1}$
4. $\mu_t = \bar{\mu}_t + K_t (z_t - h(\bar{\mu}_t))$
5. $\Sigma_t = (I - K_t H_t) \bar{\Sigma}_t$

Since Kalman Filter is only used for linear systems it is not directly applicable to non-linear systems. The EKF uses linearization to enable estimation of non-linear systems. In non-linear systems the state transition cannot be expressed by a matrix like in Kalman Filter but is expressed by a function g. To linearize g, first order Taylor expansion is used which creates a linear approximation of g depending on the properties of the first order derivative of g. Taylor expansion approximates a function from a single point by using the derivatives at this point. For Gaussian

functions it is reasonable to use the point of the largest likelihood for this point which is the mean μ_{t-1}. The linearization for g is therefore given by

$$\begin{aligned} g(a_t, x_{t-1}) &\approx g(a_t, \mu_{t-1}) + g'(a_t, \mu_{t-1})\,(x_{t-1} - \mu_{t-1}) \\ &= g(a_t, \mu_{t-1}) + G_t(x_{t-1} - \mu_{t-1}) \end{aligned} \tag{3-1}$$

where $\mathbf{x}_t$ denotes the actual state at time t-1 and G_t is the Jacobian of g (a_t, μ_{t-1}). The Jacobian G_t plays an important role in the estimation of the covariance (Algorithm 3.1. line 2). To model noise in the transition function a Gaussian random variable with mean 0 and covariance R_t is incorporated in the state prediction. Same linearization method is applied to measurement function h resulting in its Jacobian H_t (Algorithm 3.1. line 3). The Kalman Gain is calculated in same line. The Kalman gain is a weight which shows how strong the actual measurement z_t can influence the predicted mean $\overline{\mu_t}$, resulting in μ_t (line 4). Finally Kalman gain and the Jacobian of the measurement function are used to update the estimation of the covariance (line 5).

3.2 State Representation [2]

The state vector $\mathbf{x}_t$ is composed of two parts. The first part is composed of camera state and its size is fixed. The second part contains features and their estimated positions which build the map. This part is initially empty and grows or shrinks by adding or deleting the features.

In the following chapters two coordinate systems; the world coordinate system W and camera coordinate systems C will be considered. These coordinate systems will be denoted with superscripts to indicate the coordinate system of the variable.

Moreover Visual Stereo Slam uses right-hand coordinate system for both coordinates.

3.2.1 Camera Representation

The first part of the state vector, camera state is represented by a 13 dimensional vector $\mathbf{x}_v$ which is the combination of location $\mathbf{r}^{WC}$, quaternion defining orientation $\mathbf{q}^{WC}$, velocity $\mathbf{v}^{W}$, and angular velocity w^{C}.

$$\mathbf{x}_v = \begin{bmatrix} \mathbf{r}^{WC} \\ \mathbf{q}^{WC} \\ \mathbf{v}^{W} \\ w^{C} \end{bmatrix} \tag{3-2}$$

$\mathbf{r}^{WC} = (x_c\ y_c\ z_c)^T$ denotes the 3D position of the camera optical center in the world coordinate system, $\mathbf{q}^{WC}$ is the unit quaternion specifying the camera orientation relative to the world frame, $\mathbf{v}^{W}$ encodes the linear velocities of the camera along the coordinate axes of W and w^{C} indicates the angular velocities relative to the camera coordinate system C. Initial camera position is in the origin of the world coordinate system ($\mathbf{r}^{WC} = (0\ 0\ 0)^T$) , looking into the direction of the positive Z_w-axis ($\mathbf{q}^{WC} = (1\ 0\ 0\ 0)^T$) and the camera is assumed to be stationary ($\mathbf{v}^{W} = (0\ 0\ 0)^T$, $w^{C} = (0\ 0\ 0)^T$).

The 13 x 13 covariance matrix **P** is initialized as follows:

$$P = \begin{bmatrix} 0 & \cdots & 0 & 0 \\ \vdots & \ddots & 0 & 0 \\ 0 & \cdots & 0 & 0 \\ 0 & \cdots & 0 & V \end{bmatrix} \quad , V = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.5 \end{bmatrix} \tag{3-3}$$

3.2.2 Feature Representation [15]

A feature $\mathbf{y}_i$ is defined by vector:

$$\mathbf{y}_i = \begin{bmatrix} x_{c,i} & y_{c,i} & z_{c,i} & \theta_i & \varphi_i & p_i \end{bmatrix}^T \tag{3-4}$$

$x_{c,i}$, $y_{c,i}$, $z_{c,i}$ specify the 3D position of the camera's optical center at the first observation of feature $\mathbf{y}_i$. θ_i and φ_i represent the azimuth and the elevation of the feature in reference to the camera coordinate system and p_i is the inverse depth of $\mathbf{y}_i$.

The 3D point modeled by equation (3-4) is given by

$$\mathbf{x}_i = \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} = \begin{bmatrix} x_{c,i} \\ y_{c,i} \\ z_{c,i} \end{bmatrix} + \frac{1}{p_i} m(\theta_i, \phi_i) \tag{3-5}$$

$$m(\theta_i, \phi_i) = (\sin\theta_i \cos\phi_i \quad -\sin\phi_i \quad \cos\theta_i \cos\phi_i)^T \tag{3-6}$$

Function $m(\theta_i, \phi_i)$ yields a unit vector pointing from the camera's optical center to feature $\mathbf{y}_i$. Multiplying this vector with the depth d_i

$$d_i = \frac{1}{p_i} \tag{3-7}$$

and adding it to the position of the first observation $(x_{c,i}, y_{c,i}, z_{c,i})^T$ results in current 3D position.

The state vector **x** is composed of the camera state and the map features:

$$\mathbf{x} = (\mathbf{x}_v{}^T, \mathbf{y}_1{}^T, \mathbf{y}_2{}^T, ... \mathbf{y}_n{}^T)^T \tag{3-8}$$

3.3 State Prediction [8]

Since we use a freely moving camera and there are no given explicit commands, state transition function only depends on μ_{t-1} and is defined as

$$g_v(\mu_{t-1}) = \begin{bmatrix} \mathbf{r}_t{}^{WC} \\ \mathbf{q}_t{}^{WC} \\ \mathbf{v}_t{}^{W} \\ w_t{}^{C} \end{bmatrix} = \begin{bmatrix} \mathbf{r}_{t-1}^{WC} + \mathbf{v}_{t-1}^{W} \Delta t \\ \mathbf{q}_{t-1}^{WC} \times quat\,(w_{t-1}^{C} \Delta t) \\ \mathbf{v}_{t-1}^{W} \\ w_{t-1}^{C} \end{bmatrix} \tag{3-9}$$

where Δt is the time difference between t and t-1 and $quat\,(w_{t-1}^{C} \Delta t)$ is the quaternion representing the rotation of $w_{t-1}^{C} \Delta t$. After two operations, quaternion

from the given angular velocities will be computed. Firstly the angle-axis α=< **a**, α> representation of the rotation is calculated. For given angular velocity $w_t^C = (w_{t,X}^C \; w_{t,Y}^C \; w_{t,Z}^C)^T$ and Δt the equivalent angle-axis representation is expressed by

$$a = \langle \mathrm{a}, \alpha \rangle = < \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix}, \alpha > \; = \; < \begin{pmatrix} \frac{w_{t,X}^C \Delta t}{\left\| w_t^C \Delta t \right\|} \\ \frac{w_{t,Y}^C \Delta t}{\left\| w_t^C \Delta t \right\|} \\ \frac{w_{t,Z}^C \Delta t}{\left\| w_t^C \Delta t \right\|} \end{pmatrix}, \left\| w_t^C \Delta t \right\| > \tag{3-10}$$

where $w_{t,\gamma}^C$, γ ∈ {X,Y,Z} refers to the angular velocity around the indicated coordinate axis. The result is transferred to the quaternion **q**:

$$q = \left(\cos\frac{\alpha}{2} \quad \frac{a_x}{\|\mathrm{a}\|}\sin\frac{\alpha}{2} \quad \frac{a_y}{\|\mathrm{a}\|}\sin\frac{\alpha}{2} \quad \frac{a_z}{\|\mathrm{a}\|}\sin\frac{\alpha}{2} \right)^T \tag{3-11}$$

Since there is no information about linear and angular velocity, they are predicted to be the same in $\overline{\mu_t}$ as in μ_{t-1}. Since all features are static, their estimations are not changed by the transition function. So state transition function is represented as

$$g(\mu_{t-1}) = \begin{bmatrix} g_v(\mu_{t-1}) \\ \mathbf{0} \end{bmatrix} \tag{3-12}$$

where **0** denotes the 0 vector and dimension of this vector is dim(**0**) = n_{dim} – 13 given dim (μ_{t-1}) = n_{dim}.

To complete the prediction step of the EKF, we should consider the Jacobian G_t of the transition function $g(\mu_{t-1})$:

$$G_t = \begin{bmatrix} F_t & 0 \\ 0 & 0 \end{bmatrix}, \dim(F_t) = 13 \times 13, \dim(G_t) = n_{\dim} \times n_{\dim} \tag{3-13}$$

F_t is the Jacobian of $g_v(\mu_{t-1})$ given in (3-12). The structure of F_t is

$$F_t = \begin{bmatrix} \frac{\partial \mathbf{r}_t^{WC}}{\partial \mathbf{r}_{t-1}^{WC}} & 0 & \frac{\partial \mathbf{r}_t^{WC}}{\partial \mathbf{v}_{t-1}^{W}} & 0 \\ 0 & \frac{\partial \mathbf{q}_t^{WC}}{\partial \mathbf{q}_{t-1}^{WC}} & 0 & \frac{\partial \mathbf{q}_t^{WC}}{\partial w_{t-1}^{C}} \\ 0 & 0 & \frac{\partial \mathbf{v}_t^{W}}{\partial \mathbf{v}_{t-1}^{W}} & 0 \\ 0 & 0 & 0 & \frac{\partial w_t^{W}}{\partial w_{t-1}^{W}} \end{bmatrix} \tag{3-14}$$

0 is inserted in place of Jacobian submatrices where no influence is present. 4 of the 6 non-zero Jacobians are calculated easily as follows

$$\frac{\partial \mathbf{r}_t^{WC}}{\partial \mathbf{r}_{t-1}^{WC}} = \frac{\partial \mathbf{v}_t^{W}}{\partial \mathbf{v}_{t-1}^{W}} = \frac{\partial w_t^{W}}{\partial w_{t-1}^{W}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3-15}$$

$$\frac{\partial \mathbf{r}_t^{WC}}{\partial \mathbf{v}_{t-1}^{W}} = \Delta t \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3-16}$$

To distinguish the real and the three imaginary parts, a quaternion **q** will be denoted as $\mathbf{q} = (q_{t,r}\ q_{t,i}\ q_{t,j}\ q_{t,k})^T$. The quaternion **a** refers to the quaternion representing the rotation given by $w_{t-1}^{C}\Delta t$, thus a_{t-1} = quat ($w_{t-1}^{C}\Delta t$). Using these notations the 2 remaining Jacobians $\frac{\partial \mathbf{q}_t^{WC}}{\partial \mathbf{q}_{t-1}^{WC}}$ and $\frac{\partial \mathbf{q}_t^{WC}}{\partial w_{t-1}^{C}}$ are as follows

$$\frac{\partial \mathbf{q}_t^{WC}}{\partial \mathbf{q}_{t-1}^{WC}} = \begin{bmatrix} a_{t-1,r} & -a_{t-1,i} & -a_{t-1,j} & -a_{t-1,k} \\ a_{t-1,i} & a_{t-1,r} & -a_{t-1,k} & a_{t-1,j} \\ a_{t-1,j} & -a_{t-1,k} & a_{t-1,r} & a_{t-1,i} \\ a_{t-1,k} & a_{t-1,j} & -a_{t-1,i} & a_{t-1,r} \end{bmatrix} \tag{3-17}$$

$$\frac{\partial \mathbf{q}_t^{WC}}{\partial w_{t-1}^{C}} = \frac{\partial \mathbf{q}_t^{WC}}{\partial quat(w_{t-1}^{C}\Delta t)} \frac{\partial quat(w_{t-1}^{C}\Delta t)}{\partial w_{t-1}^{C}\Delta t} \tag{3-18}$$

$$\frac{\partial \mathbf{q}_t^{WC}}{\partial w_{t-1}^{C}} = \begin{bmatrix} q_{t-1,r} & -q_{t-1,i} & -q_{t-1,j} & -q_{t-1,k} \\ q_{t-1,i} & q_{t-1,r} & -q_{t-1,k} & q_{t-1,j} \\ q_{t-1,j} & q_{t-1,k} & q_{t-1,r} & q_{t-1,i} \\ q_{t-1,k} & -q_{t-1,j} & -q_{t-1,i} & q_{t-1,r} \end{bmatrix} \begin{bmatrix} \frac{\partial a_{t-1,r}}{\partial w_{t-1,X}^{C}\Delta t} & \frac{\partial a_{t-1,r}}{\partial w_{t-1,Y}^{C}\Delta t} & \frac{\partial a_{t-1,r}}{\partial w_{t-1,Z}^{C}\Delta t} \\ \frac{\partial a_{t-1,i}}{\partial w_{t-1,X}^{C}\Delta t} & \frac{\partial a_{t-1,i}}{\partial w_{t-1,Y}^{C}\Delta t} & \frac{\partial a_{t-1,i}}{\partial w_{t-1,Y}^{C}\Delta t} \\ \frac{\partial a_{t-1,j}}{\partial w_{t-1,X}^{C}\Delta t} & \frac{\partial a_{t-1,j}}{\partial w_{t-1,Y}^{C}\Delta t} & \frac{\partial a_{t-1,j}}{\partial w_{t-1,Z}^{C}\Delta t} \\ \frac{\partial a_{t-1,k}}{\partial w_{t-1,X}^{C}\Delta t} & \frac{\partial a_{t-1,k}}{\partial w_{t-1,Y}^{C}\Delta t} & \frac{\partial a_{t-1,k}}{\partial w_{t-1,Z}^{C}\Delta t} \end{bmatrix} \tag{3-19}$$

To solve $\frac{\partial quat(w_{t-1}^{C}\Delta t)}{\partial w_{t-1}^{C}\Delta t}$, firstly the partial derivatives of the real part of the quaternion should be considered. These are the partial derivatives in the first row:

$$\frac{\partial a_{t-1,r}}{\partial w_{t-1,\gamma}^{C}\Delta t} = -\sin\left(\left\|w_{t-1}^{C}\right\|\frac{\Delta t}{2}\right)\frac{w_{t-1,\gamma}^{C}}{\left\|w_{t-1}^{C}\right\|}\frac{\Delta t}{2}, \quad \gamma \in \{X,Y,Z\} \tag{3-20}$$

The rest of the partial derivatives are imaginary part of the quaternion **a** derived by its corresponding angular velocity. The rest of matrix consists of a 3x3 submatrix and its main diagonal partial derivatives are calculated as

$$\frac{\partial a_{t-1,n}}{\partial w_{t-1,\gamma}^{C}\Delta t} = \cos\left(\left\|w_{t-1}^{C}\right\|\frac{\Delta t}{2}\right)\frac{{w_{t-1,\gamma}^{C}}^{2}}{\left\|w_{t-1}^{C}\right\|^{2}}\frac{\Delta t}{2} + \sin\left(\left\|w_{t-1}^{C}\right\|\frac{\Delta t}{2}\right)\frac{1}{\left\|w_{t-1}^{C}\right\|}\left(1-\frac{{w_{t-1,\gamma}^{C}}^{2}}{\left\|w_{t-1}^{C}\right\|^{2}}\right) \tag{3-21}$$

where (η,γ) ∈ { (i,X), (j,Y), (k,Z) }. The rest of the partial derivatives are

$$\frac{\partial a_{t-1,n}}{\partial w_{t-1,\gamma}^{C}\Delta t} = \frac{w_{t-1,\zeta}^{C}w_{t-1,\gamma}^{C}}{\left\|w_{t-1}^{C}\right\|^{2}}\left(\cos\left(\left\|w_{t-1}^{C}\right\|\frac{\Delta t}{2}\right)\frac{\Delta t}{2} - \left(\frac{1}{\left\|w_{t-1}^{C}\right\|}\sin\left(\left\|w_{t-1}^{C}\right\|\frac{\Delta t}{2}\right)\right)\right) \tag{3-22}$$

with (η,γ,ζ) ∈ {(i,Y,X), (i,Z,X), (j,X,Y), (j,Z,Y), (k,X,Z), (k,Y,Z)}.

To calculate the prediction of the covariance $\sum_t$, the additional noise R_t is calculated.

$$R_t = \begin{bmatrix} R_t' & 0 \\ 0 & 0 \end{bmatrix}, \dim(R_t') = 13x13, \dim(R_t) = n_{\dim} \times n_{\dim} \tag{3-23}$$

R_t' is given as

$$R_t' = \widetilde{F}_t V_{\max,t} \widetilde{F}_t^T = \begin{bmatrix} \frac{\partial \mathbf{r}_t^{WC}}{\partial \mathbf{v}_{t-1}^W} & 0 \\ 0 & \frac{\partial \mathbf{q}_t^{WC}}{\partial w_{t-1}^C} \\ \frac{\partial \mathbf{v}_t^W}{\partial \mathbf{v}_{t-1}^W} & 0 \\ 0 & \frac{\partial w_t^W}{\partial w_{t-1}^W} \end{bmatrix} V_{\max,t} \begin{bmatrix} \frac{\partial \mathbf{r}_t^{WC}}{\partial \mathbf{v}_{t-1}^W} & 0 \\ 0 & \frac{\partial \mathbf{q}_t^{WC}}{\partial w_{t-1}^C} \\ \frac{\partial \mathbf{v}_t^W}{\partial \mathbf{v}_{t-1}^W} & 0 \\ 0 & \frac{\partial w_t^W}{\partial w_{t-1}^W} \end{bmatrix}^T \tag{3-24}$$

with

$$\dim(\widetilde{F}_t) = 13 \times 6, \dim(\mathbf{V}_{\max,t}) = 6 \times 6 \tag{3-25}$$

$\widetilde{F}_t$ is composed of the columns 8-13 of F_t given in (3-14). If the maximal linear velocity is denoted by $v_{\max}^W$ and maximal angular velocity as $w_{\max}^C$, matrix $V_{max,t}$ is defined as

$$\mathbf{V}_{\max,t} = \begin{pmatrix} V_{\max,t} & 0 \\ 0 & \Omega_{\max,t} \end{pmatrix} \tag{3-26}$$

where $V_{max,t}$ and $\Omega_{max,t}$ is

$$V_{\max,t} = (v_{\max}^{W}\Delta t)^2 \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \Omega_{\max,t} = (\mathrm{w}_{\max}^{\mathrm{C}}\Delta \mathrm{t})^2 \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3\text{-}27)$$

3.4 Measurement Function

In this section, in the first part measurement function of single camera will be discussed. This function is used in structure from motion of the SLAM process. In the second part measurement function used for calculating feature depths by using stereo camera will be discussed.

3.4.1 Measurement Function with Single Camera [8]

In this section the measurement function h and its Jacobian H_t will be represented. In EKF Algorithm given in algorithm 3.1, H_t influences the Kalman Gain K_t. The difference between the actual measurement z_t and the measurement prediction $h(\overline{\mu_t})$ is used to correct predicted pose according to received sensor data.

According to the pinhole camera model, an observed point on the image sensor defines a directional vector $\mathbf{h}^{C}= (h_x\ h_y\ h_z)^T$ in the camera coordinate system C. The directional vector for a point y_i in inverse depth coding is given as below

$$\mathbf{h}_i^C = \mathbf{h}_{p,i}^C = \mathbf{R}^{CW}\left(p_i\left(\begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} - \mathbf{r}^{WC}\right) + m(\theta,\phi_i)\right) \tag{3-28}$$

where $\mathbf{R}^{CW}$ denotes the position of the camera optical center in world coordinate system. $m(\theta_i, \varphi_i)$ is defined in (3.6). The visualization of inverse depth coding is given in Figure 3-2. [15]

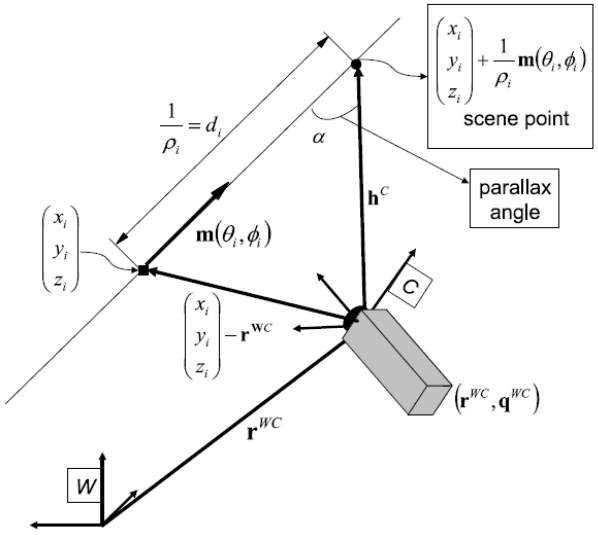

Figure 3-2 Feature Parameterization and measurement equation

The rotation is coded in the rotation matrix $\mathbf{R}^{wc}$ depending on the camera orientation $\mathbf{q}$. A unit quaternion $\mathbf{q} = (q_r\ q_i\ q_j\ q_k)$ can be converted by function q2r($\mathbf{q}$) into a rotation matrix $\mathbf{R}$ as given below [20].

$$R = q2r(q) = \begin{bmatrix} q_r^2 + q_i^2 - q_j^2 - q_k^2 & -2q_rq_k + 2q_iq_j & 2q_rq_j + 2q_iq_k \\ 2q_rq_k + 2q_iq_j & q_r^2 - q_i^2 + q_j^2 - q_k^2 & -2q_rq_i + 2q_jq_k \\ -2q_rq_j + 2q_iq_k & 2q_rq_i + 2q_jq_k & q_r^2 - q_i^2 - q_j^2 + q_k^2 \end{bmatrix} \tag{3-29}$$

For a given point $\mathbf{f_i}$, $\mathbf{f} \in \{x, y\}$ and its associated directional vector $\mathbf{h}_i^C$, expected projection of undistorted image coordinates $(u_{u,i}, v_{u,i})^T$ is given by

$$h_p(\mathbf{f}_i) = \begin{bmatrix} u_{u,i} \\ v_{u,i} \end{bmatrix} = \begin{bmatrix} u_0 - \frac{f}{d_u}\frac{h_{x,i}}{h_{z,i}} \\ v_o - \frac{f}{d_v}\frac{h_{y,i}}{h_{z,i}} \end{bmatrix} \tag{3-30}$$

where (u_o, v_o) is the principal point, $d_u \times d_v$ is the physical size of the pixel and f is the focal length of the camera.

Undistorted image coordinate $(u_{u,i}, v_{u,i})^T$ needs to be distorted to be comparable to the actual received image. The distortion of the image coordinate $(u_{u,i}, v_{u,i})^T$ can be found according to the formula given below and by the formula distorted image coordinates $(u_{d,i}, v_{d,i})^T$ are found.

$$\begin{bmatrix} u_{d,i} \\ v_{d,i} \end{bmatrix} = h_d \begin{bmatrix} u_{u,i} \\ v_{u,i} \end{bmatrix} = \begin{bmatrix} u_0 + \frac{u_{u,i} - u_0}{1 + k_1 r_d^2 + k_2 r_d^4} \\ v_{u,i} + \frac{v_{u,i} - u_0}{1 + k_1 r_d^2 + k_2 r_d^4} \end{bmatrix} \tag{3-31}$$

After these steps, distorted image coordinate measurements are checked whether they are inside or outside in the next frame (i.e. $0 \leq u_{d,i} \leq$ width and $0 \leq v_{d,i} \leq$ height have to be satisfied). All expected measurements inside the image compose $h(\overline{\mu}_t)$.

After calculating h($\overline{\mu}_t$), the Jacobian H_t of h($\overline{\mu}_t$) will be considered. In the remaining of this subsection, H_t' will be considered where H_t' will build H_t. The dimension of H_t' is the number of features expected to be inside the next frame. If we represent the features $F_t=\{f_{t,1}, f_{t,2}, \ldots f_{t,n}\}$ with $n_{dim} = 13 + \sum f_{t,i}\ \dim(f_{t,i})$, $f_{t,i} \in F_t$ and the subset that contains the features which are expected to be in the next frame: $M_t \subseteq F_t$, $\dim(M_t)=m_{dim}$ then with given values H_t' dimension will be $2m_{dim} \times n_{dim}$ and H_t' has the following structure.

$$H_t' = \begin{bmatrix} \dfrac{\partial h(\mathbf{g}_{t,1})}{\partial \overline{\mu}_t} \\ \dfrac{\partial h(\mathbf{g}_{t,2})}{\partial \overline{\mu}_t} \\ \vdots \\ \dfrac{\partial h(\mathbf{g}_{t,m_{\dim}})}{\partial \overline{\mu}_t} \end{bmatrix}, \quad \dim\left(\frac{\partial h(\mathbf{g}_{t,i})}{\partial \overline{\mu}_t}\right) = 2 \times n_{\dim}, g_{t,i} \in M_t \tag{3-32}$$

where

$$\frac{\partial h(\mathbf{f}_{t,i})}{\partial \overline{\mu}_t} = \left(\frac{\partial h(\mathbf{f}_{t,i})}{\partial \mathbf{x}_{t,v}} \quad \underbrace{0 \cdots 0}_{\sum_{k<i} \dim(\mathbf{f}_{t,k})} \quad \frac{\partial h(\mathbf{f}_{t,i})}{\partial \mathbf{f}_{t,i}} \quad \underbrace{0 \cdots 0}_{\sum_{i<l<n} \dim(\mathbf{f}_{t,l})} \right) \tag{3-33}$$

where

$$\dim\left(\frac{\partial h(\mathbf{f}_{t,i})}{\partial \mathbf{x}_{t,v}}\right) = 2 \times 13, \dim\left(\frac{\partial h(\mathbf{f}_{t,i})}{\partial \mathbf{f}_{t,i}}\right) = 2 \times \dim(\mathbf{f}_{t,i}) \tag{3-34}$$

Now $\frac{\partial h(\mathbf{f}_{t,i})}{\partial \mathbf{x}_{t,v}}$ and $\frac{\partial h(\mathbf{f}_{t,i})}{\partial \mathbf{f}_{t,i}}$ will be analyzed. To simplify notations t will be omitted.

Firstly we will deal with $\frac{\partial h(\mathbf{f}_t)}{\partial \mathbf{x}_v}$. This Jacobian shows how the camera state $\mathbf{x}_v$ influences the outcome of the measurement function $h(f_i)$ for a given feature. The dimension of this Jacobian is 2 × 13 because the dimension of the camera state is 13. $\frac{\partial h(\mathbf{f}_t)}{\partial \mathbf{x}_v}$ is partitioned as

$$\frac{\partial h(\mathbf{f}_i)}{\partial \mathbf{x}_v} = \begin{pmatrix} \frac{\partial h(\mathbf{f}_i)}{\partial \mathbf{r}^{WC}} & \frac{\partial h(\mathbf{f}_i)}{\partial \mathbf{q}^{WC}} & 0 \end{pmatrix} \tag{3-35}$$

where dim $\left(\frac{\partial h(\mathbf{f}_i)}{\partial \mathbf{r}^{WC}}\right)$ = 2 × 3 and dim $\left(\frac{\partial h(\mathbf{f}_i)}{\partial \mathbf{q}^{WC}}\right)$ = 2 × 4. Since $h(\mathbf{f}_i)$ does not depend on $\mathbf{v}^W$ and w^C of camera state $\mathbf{x}_v$, their values are 0 and build 2 × 6 zero matrix. The rest of the Jacobian parts can be expressed as

$$\frac{\partial h(\mathbf{f}_i)}{\partial \mathbf{r}^{WC}} = \frac{\partial h(\mathbf{f}_i)}{\partial \mathbf{h}_i^C} \frac{\partial \mathbf{h}_i^C}{\partial \mathbf{r}^{WC}} \tag{3-36}$$

$$\frac{\partial h(\mathbf{f}_i)}{\partial \mathbf{q}^{WC}} = \frac{\partial h(\mathbf{f}_i)}{\partial \mathbf{h}_i^C} \frac{\partial \mathbf{h}_i^C}{\partial \mathbf{q}^{WC}} \tag{3-37}$$

$$\frac{\partial h(\mathbf{f}_i)}{\partial \mathbf{f}_i} = \frac{\partial h(\mathbf{f}_i)}{\partial \mathbf{h}_i^C} \frac{\partial \mathbf{h}_i^C}{\partial \mathbf{f}_i} \tag{3-38}$$

Firstly we will examine $h(\mathbf{f}_i)$ with respect to the direction vector $\partial\mathbf{h}_i^C$ of $\mathbf{f}_i$ in the camera coordinate. We define the measurement function for undistorted image coordinates $(u_{u,i},v_{u,i})^T$ in equation (3-30) but the final result of the measurement estimation is disturbed by $h_d(u_{u,i},v_{u,i})^T$. This fact effects the Jacobian and results in

$$\frac{\partial h(\mathbf{f}_i)}{\partial \mathbf{h}_i^C} = \frac{\partial h(\mathbf{f}_i)}{\partial h_p(\mathbf{f}_i)} \frac{\partial h_p(\mathbf{f}_i)}{\partial \mathbf{h}_i^C} \tag{3-39}$$

The first part $\frac{\partial h(\mathbf{f}_i)}{\partial h_p(\mathbf{f}_i)}$ can be seen as $\frac{\partial h(h_d)}{\partial (u_{u,i}, v_{u,i})}$ which contains the derivatives of the distorted image coordinates $(u_{d,i},v_{d,i})^T$. The result is given by inversion of Jacobian $\frac{\partial h_u}{\partial (u_{d,i}, v_{d,i})}$

$$\frac{\partial h_u}{\partial (u_{d,i}, v_{d,i})} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \tag{3-40}$$

where

$$\begin{aligned} a &= (1 + k_1 r_d^2 + k_2 r_d^4) + (k_1 + 2k_2 r_d^2) 2((u_{d,i} - u_0) d_u)^2 \\ b &= (u_{d,i} - u_0)(k_1 + 2k_2 r_d^2)(2(v_{d,i} - v_0) d_v^2) \\ c &= (v_{d,i} - v_0)(k_1 + 2k_2 r_d^2)(2(u_{d,i} - u_0) d_u^2) \\ d &= (1 + k_1 r_d^2 + k_2 r_d^4) + (k_1 + 2k_2 r_d^2) 2((v_{d,i} - v_0) d_v)^2 \end{aligned} \tag{3-41}$$

with r_d defined in equation (2-11). From (3-30) the second part of the Jacobian $\frac{\partial h_p(\mathbf{f}_i)}{\partial \mathbf{h}_i^C}$ is calculated as

$$\frac{\partial h_p(\mathbf{f}_i)}{\partial \mathbf{h}_i^C} = \begin{bmatrix} -\frac{f}{d_u}\frac{1}{h_{z,i}} & 0 & \frac{f}{d_u}\frac{h_{x,i}}{h_{z,i}^2} \\ 0 & -\frac{f}{d_v}\frac{1}{h_{z,i}} & \frac{f}{d_v}\frac{h_{y,i}}{h_{z,i}^2} \end{bmatrix} \tag{3-42}$$

To calculate remaining Jacobians $\frac{\partial \mathbf{h}_i^C}{\partial \mathbf{r}^{WC}}$, $\frac{\partial \mathbf{h}_i^C}{\partial \mathbf{q}^{WC}}$ and $\frac{\partial \mathbf{h}_i^C}{\partial \mathbf{f}_i}$, we will calculate the partial derivatives of equation (3-28). The dependency of the directional vector in the camera coordinate system $\mathbf{h}_i^C$ with respect to the position of the camera's optical center in the world coordinate system given below will be computed.

$$\frac{\partial \mathbf{h}_{p,i}^C}{\partial \mathbf{r}^{WC}} = -\rho_i \mathbf{R}^{CW} \tag{3-43}$$

Now we will calculate the Jacobian of the directional vector with respect to the rotation of the camera $\mathbf{q}^{WC}$. The rotation $\mathbf{q}^{WC}$ influences $\mathbf{h_i}$ via rotation matrix $\mathbf{R}^{CW}$. $\mathbf{R}^{CW}$ is constructed by function q2r($\bar{\mathbf{q}}^{WC}$) so Jacobian matrix $\frac{\partial \mathbf{h}_i^C}{\partial \mathbf{q}^{WC}}$ becomes

$$\frac{\partial \mathbf{h}_i^C}{\partial \mathbf{q}^{WC}} = \frac{\partial \mathbf{h}_i^C}{\partial \bar{\mathbf{q}}^{WC}} \frac{\partial \bar{\mathbf{q}}^{WC}}{\partial \mathbf{q}^{WC}} \tag{3-44}$$

To calculate $\frac{\partial \mathbf{h}_i^C}{\partial \mathbf{q}^{WC}}$ the construction of $\mathbf{R}^{CW}$ by q2r($\bar{\mathbf{q}}^{WC}$) given by formula (3-29) and its partial derivatives with respect to real and imaginary parts of quaternion $\bar{\mathbf{q}}^{WC}$ needs to be calculated. This result in

$$\frac{\partial \mathbf{h}_i^C}{\partial \bar{\mathbf{q}}^{WC}} = \left[\frac{\partial q2r(\bar{\mathbf{q}}^{WC})}{\partial \bar{q}_r^{WC}} \mathbf{d}_i \quad \frac{\partial q2r(\bar{\mathbf{q}}^{WC})}{\partial \bar{q}_i^{WC}} \mathbf{d}_i \quad \frac{\partial q2r(\bar{\mathbf{q}}^{WC})}{\partial \bar{q}_j^{WC}} \mathbf{d}_i \frac{\partial q2r(\bar{\mathbf{q}}^{WC})}{\partial \bar{q}_k^{WC}} \mathbf{d}_i \right] \quad (3\text{-}45)$$

where $\mathbf{d}_i$ denotes the direction vector $\mathbf{h}_i$ before it is rotated into the camera frame. $\mathbf{d}_i$ is given as

$$\mathbf{d}_{\rho,i} = \rho_i \left(\begin{pmatrix} x_{c,i} \\ y_{c,i} \\ z_{c,i} \end{pmatrix} - \mathbf{r}^{WC} \right) + m(\theta_i, \phi_i) \quad (3\text{-}46)$$

The partial derivatives of q2r($\bar{\mathbf{q}}^{WC}$) with respect to the different parts of quaternion $\bar{\mathbf{q}}^{WC}$ are given as

$$\frac{\partial q2r(\bar{\mathbf{q}}^{WC})}{\partial \bar{q}_r^{WC}} = 2 \begin{pmatrix} \bar{q}_r^{WC} & -\bar{q}_k^{WC} & \bar{q}_j^{WC} \\ \bar{q}_k^{WC} & \bar{q}_r^{WC} & -\bar{q}_i^{WC} \\ -\bar{q}_j^{WC} & \bar{q}_i^{WC} & \bar{q}_r^{WC} \end{pmatrix} \quad (3\text{-}47)$$

$$\frac{\partial q2r(\bar{\mathbf{q}}^{WC})}{\partial \bar{q}_i^{WC}} = 2 \begin{pmatrix} \bar{q}_i^{WC} & \bar{q}_j^{WC} & \bar{q}_k^{WC} \\ \bar{q}_j^{WC} & -\bar{q}_i^{WC} & -\bar{q}_r^{WC} \\ \bar{q}_k^{WC} & \bar{q}_r^{WC} & -\bar{q}_i^{WC} \end{pmatrix} \quad (3\text{-}48)$$

$$\frac{\partial q2r(\bar{\mathbf{q}}^{WC})}{\partial \bar{q}_j^{WC}} = 2\begin{pmatrix} -\bar{q}_j^{WC} & \bar{q}_i^{WC} & \bar{q}_r^{WC} \\ \bar{q}_i^{WC} & \bar{q}_j^{WC} & \bar{q}_k^{WC} \\ -\bar{q}_r^{WC} & \bar{q}_k^{WC} & -\bar{q}_j^{WC} \end{pmatrix} \tag{3-49}$$

$$\frac{\partial q2r(\bar{\mathbf{q}}^{WC})}{\partial \bar{q}_k^{WC}} = 2\begin{pmatrix} -\bar{q}_k^{WC} & -\bar{q}_r^{WC} & \bar{q}_i^{WC} \\ \bar{q}_r^{WC} & -\bar{q}_k^{WC} & \bar{q}_j^{WC} \\ \bar{q}_i^{WC} & \bar{q}_j^{WC} & \bar{q}_k^{WC} \end{pmatrix} \tag{3-50}$$

and $\frac{\partial \bar{\mathbf{q}}^{WC}}{\partial \mathbf{q}^{WC}}$ is given as

$$\frac{\partial \bar{\mathbf{q}}^{WC}}{\partial \mathbf{q}^{WC}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \tag{3-51}$$

Finally we will calculate $\frac{\partial \mathbf{h}_{\rho,i}^{C}}{\partial \mathbf{f}_i}$ where this is the Jacobian of the directional vector $\mathbf{h}_i^C$ with respect to the estimation of point f_i.

$$\frac{\partial \mathbf{h}_{\rho,i}^{C}}{\partial \mathbf{f}_i} = \begin{pmatrix} \frac{\partial \mathbf{h}_{\rho,\mathbf{i}}^{C}}{\partial x_{c,i}, y_{c,i}, z_{c,i}} & \frac{\partial \mathbf{h}_{\rho,\mathbf{i}}^{C}}{\partial \theta_i} & \frac{\partial \mathbf{h}_{\rho,\mathbf{i}}^{C}}{\partial \phi_i} & \frac{\partial \mathbf{h}_{\rho,\mathbf{i}}^{C}}{\partial \rho_i} \end{pmatrix}, \quad \dim\left(\frac{\partial \mathbf{h}_{\rho,i}^{C}}{\partial \mathbf{f}_i}\right) = 3\times 6 \tag{3-52}$$

with

$$\frac{\partial \mathbf{h}_{\rho,\mathbf{i}}^{C}}{\partial x_{c,i}, y_{c,i}, z_{c,i}} = \rho_i \mathbf{R}^{CW} \tag{3-53}$$

$$\frac{\partial \mathbf{h}_{\rho,\mathbf{i}}^{C}}{\partial \theta_i} = \mathbf{R}^{CW} \begin{pmatrix} \cos\theta_i \cos\phi_i \\ 0 \\ -\sin\theta_i \cos\phi_i \end{pmatrix} \tag{3-54}$$

$$\frac{\partial \mathbf{h}_{\rho,\mathbf{i}}^{C}}{\partial \phi_i} = \mathbf{R}^{CW} \begin{pmatrix} -\sin\theta_i \sin\phi_i \\ -\cos\phi_i \\ -\cos\theta_i \sin\phi_i \end{pmatrix} \tag{3-55}$$

$$\frac{\partial \mathbf{h}_{\rho,\mathbf{i}}^{C}}{\partial \rho_i} = \mathbf{R}^{CW} \left(\begin{pmatrix} x_{c,i} \\ y_{c,i} \\ z_{c,i} \end{pmatrix} - \mathbf{r}^{WC} \right) \tag{3-56}$$

3.4.2 Measurement Function with Stereo Camera System

In this section depth calculation of features by using stereo camera system will be presented. This calculation is used for initializing the features and estimating the depth of the features. To calculate the position of a feature, we have to know intrinsic and extrinsic parameters of the cameras which are given at table 4-1 and 4-2 respectively. Now we will calculate the depth of the features which is observed from left and right cameras with given pixel points:

$$x\_l1 = \begin{bmatrix} xl1 & xl2 \end{bmatrix} \qquad x\_r1 = \begin{bmatrix} xr1 & xr2 \end{bmatrix} \tag{3-57}$$

Firstly, normalized pixel coordinates of these two points are calculated by using the formula given below.

$$x\_d_1 = \frac{x1\ - cc_1}{f_1} \qquad x\_d_2 = \frac{x2\ - cc_2}{f_2} \tag{3-58}$$

Where f_1, f_2 is the focal length of camera in pixels, cc_1 and cc_2 is the principal point coordinates and x_1, x_2 are the feature locations in one of the camera. After this step we calculate normalized undistorted coordinates of feature positions by using formula given below.

$$x\_u1 = \frac{x\_d_1}{1 + (x\_d_1)^2 k_1 + (x\_d_1)^4 k_2} \; x\_u2 = \frac{x\_d_2}{1 + (x\_d_2)^2 k_1 + (x\_d_2)^4 k_2} \tag{3-59}$$

k1 and k2 are the radial distortion coefficients of the camera and x_u pair is the normalized undistorted projection of the feature locations. Now we extend the normalized undistorted projections in homogenous coordinates.

$$x\_ul = \begin{bmatrix} x\_u1 \\ x\_u2 \\ 1 \end{bmatrix} \quad x\_vr = \begin{bmatrix} x\_ur1 \\ x\_ur2 \\ 1 \end{bmatrix} \tag{3-60}$$

By using these values, the normalized depths of feature positions are calculated.

$$\begin{aligned} dot\_xl &= x\_ul \bullet x\_ul \\ dot\_xr &= x\_ur \bullet x\_ur \end{aligned} \tag{3-61}$$

By using the known baseline distance between cameras, the depth of features are calculated. Firstly undistorted normalized left coordinates are transfered to right coordinate system. R is given with formula (4-1).

$$u = R * xt \tag{3-62}$$

With the formula below, normalized distance of baseline is calculated.

$$D_{nor} = Xdel \times Xder - (u \bullet x\_ur)^2 \tag{3-63}$$

Distance between cameras obtained from camera calibration procedure is given as

$$T = \begin{bmatrix} -35.6698 & 0.2846 & -3.2438 \end{bmatrix} \tag{3-64}$$

Now we apply triangulation procedure to find the distance of feature's depth.

$$dist = (u \bullet T) \times (x\_ur \bullet T) - dot\_xl * (u \bullet T) \tag{3-65}$$

Now the depth of feature is calculated with respect to the left camera

$$x_{dist} = x\_ul\, x(\frac{dist}{D_{nor}}) \tag{3-66}$$

Calculated depth is used for depth measurement in initialization of cameras.

3.5 Feature Matching [8]

After discussion of the measurement function h($\overline{\mu_t}$) which is used in order to calculate expected measurements, the actual matching of features and the construction of H_t from H_t' will be discussed. The matching of features from image patches is used to characterize the appearance of the image. To decrease the

computational effort, EKF algorithm is used. This idea is used as follows: Instead of searching every possible location to match the features one by one, the area around the expected position $(\overline{u_{d,i}}, \overline{v_{d,i}})^T$ is searched which is obtained from measurement function h($\overline{\mu_t}$). The search size depends on the covariance of EKF algorithm. If the 3D position of a feature is well known, its search region will be small or vice versa. When feature f_i is successfully detected in its specific region, the actual measured position $(u_{d,i}\, v_{d,i})^T$ in the image will be part of the measurement z_t. If the feature could not be matched in its search region, it will not contribute to the correction of the pose estimation. Now this idea will be explained in detail.

To find the uncertainty of the estimated position and search area of feature $\mathbf{f}_i$, innovation covariance S_t is calculated which is defined as

$$S_t = H_t' \overline{\Sigma}_t H_t'^T + Q_t' \tag{3-67}$$

where $\overline{\Sigma}_t$ is the predicted covariance. H_t' is calculated in last section and Q_t' is the matrix to define the sensor noise in pixels. The dimensions of related matrices are given below

$$\dim(S_t) = 2m'_{\dim} \times 2m'_{\dim}, \quad \dim(H_t') = 2m'_{\dim} \times n_{\dim}$$
$$\dim(\overline{\Sigma}_t) = n_{\dim} \times n_{\dim}, \quad \dim(Q_t') = 2m'_{\dim} \times 2m'_{\dim}$$

where $n_{\dim}$ denotes the dimension of the current state $\mathbf{x}_t$ and $m'_{\dim}$ is the number of features predicted to be on the image sensor at time t. From given matrices only Q_t' is not discussed and this matrix is given as

$$Q_t' = \sigma_R^2 \, I \tag{3-68}$$

I is the identity matrix with dimensions $2m'_{\text{dim}} \times 2m'_{\text{dim}}$ and σ_R^2 is the squared standard deviation of image noise. This value is taken as $\sigma_R = 1$ in the algorithm.

In S_t matrix, the uncertainties of image coordinates are in the main diagonal. From these values the boundaries of the search region ($b_{u,i}$, $b_{v,i}$) for each region is given as

$$\begin{pmatrix} b_{u,i} \\ b_{v,i} \end{pmatrix} = 2 \begin{pmatrix} \sqrt{s_{t,(2i-1,2i-1)}} \\ \sqrt{s_{t,(2i,2i)}} \end{pmatrix} \tag{3-69}$$

where $s_{t(i,j)}$ denotes the element of matrix S_t at position (i,j). Search region is found by these values. Search region is taken as rectangle and middle of this rectangle is $(\bar{u}_{d,i}, \bar{v}_{d,i})^T$ and its upper left corner is $(\bar{u}_{d,i} - b_{u,i}, \bar{v}_{d,i} - b_{v,i})^T$ and lower right corner is $(\bar{u}_{d,i} + b_{u,i}, \bar{v}_{d,i} + b_{v,i})^T$.

If the feature can be matched inside this region, it is considered to be successfully matched. After a successful matching, the feature will contribute to the correction of the estimates of $\bar{\mu}_t$ and covariance $\bar{\Sigma}_t$. Otherwise the predicted measurement for $\mathbf{g}_i$ will be removed from $h(\bar{\mu}_t)$, and rows 2i-1 and 2i are deleted from matrix H'_t. The final H'_t is H_t of the predicted measurement $h(\bar{\mu}_t)$.

3.6 Update Step [8]

In EKF Algorithm Kalman Gain K_t is given as

$$K_t = \bar{\Sigma}_t H_t^T (H_t \bar{\Sigma}_t H_t^T + Q_t)^{-1} \tag{3-70}$$

with dimensions

$$\dim(K_t) = n_{\dim} \times 2m_{\dim}, \quad \dim(H_t) = 2m_{\dim} \times n_{\dim}$$
$$\dim(\Sigma_t) = n_{\dim} \times n_{\dim}, \quad \dim(Q_t) = 2m_{\dim} \times 2m_{\dim}$$

where n_{dim} is the dimension of the current state vector and m_{dim} is the number of matched features.

All given matrices are discussed before except Q_t matrix. Q_t matrix is similar to Q'_t apart from its dimensions so that Kalman Gain can be calculated with given matrices.

After finding μ_t and $\sum_t$ in EKF algorithm (Algorithm 3.1 line 4 and 5), a further process is needed. Since we use quaternions each time step, they need to be normalized. $\mathbf{q}^{WC}$ is normalized as

$$norm(q) = \left(\frac{q_r}{\sqrt{q_r^2+q_i^2+q_j^2+q_k^2}} \frac{q_i}{\sqrt{q_r^2+q_i^2+q_j^2+q_k^2}} \frac{q_j}{\sqrt{q_r^2+q_i^2+q_j^2+q_k^2}} \frac{q_k}{\sqrt{q_r^2+q_i^2+q_j^2+q_k^2}} \right) \quad (3\text{-}71)$$

The mean with normalized quaternion will be referred as $\mu_{t,new}$.

After finding μ_t, now $\sum_t$ will be adapted. The adapted covariance matrix is

$$\Sigma_t' = \begin{pmatrix} 1_a & 0 & 0 \\ 0 & \frac{\partial \text{norm}(\mathbf{q}^{\text{WC}})}{\partial \mathbf{q}^{\text{WC}}} & 0 \\ 0 & 0 & 1_b \end{pmatrix} \Sigma_t \begin{pmatrix} 1_a & 0 & 0 \\ 0 & \frac{\partial \text{norm}(\mathbf{q}^{\text{WC}})}{\partial \mathbf{q}^{\text{WC}}} & 0 \\ 0 & 0 & 1_b \end{pmatrix}^T \tag{3-72}$$

For matrices I_a and I_b, the dimensions are given as dim (I_a) = 3×3, dim (I_b)=(n_{dim}-7) × (n_{dim}-7). The Jacobian $\frac{\partial \text{norm}(\mathbf{q}^{WC})}{\partial \mathbf{q}^{WC}}$ is determined as

$$\frac{\partial \text{norm}(\mathbf{q})}{\partial \mathbf{q}} = (q_r^2 + q_i^2 + q_j^2 + q_z^2)^{-\frac{3}{2}} \mathbf{Q} \tag{3-73}$$

with

$$\mathbf{Q} = \begin{pmatrix} q_i^2 + q_j^2 + q_z^2 & -q_r q_i & -q_r q_j & -q_r q_k \\ -q_r q_i & q_r^2 + q_j^2 + q_k^2 & -q_i q_j & -q_i q_k \\ -q_r q_j & -q_i q_j & q_r^2 + q_i^2 + q_k^2 & -q_j q_k \\ -q_r q_k & -q_i q_k & -q_j q_k & q_r^2 + q_i^2 + q_j^2 \end{pmatrix} \tag{3-74}$$

Calculated mean and covariance is used as previous estimation in the next EKF step.

3.7 Feature Initialization [8]

A new feature $\mathbf{y}_i$ will be initialized by function

$$\mathbf{y}_i = (x_{c,i} \;\; y_{c,i} \;\; z_{c,i} \;\; \theta_i \;\; \phi_i \;\; \rho_i)^T \tag{3-75}$$

The parameters $x_{c,i} \;\; y_{c,i} \;\; z_{c,i}$ specify the camera center location at first observation of the feature. The variables θ_i and φ_i are the azimuth and the elevation of the feature with respect to the camera center whereas ρ_i is the inverse depth of the feature. Initialization of $x_{c,i}$, $y_{c,i}$ and $z_{c,i}$ is the placement of the camera system at first observation and given as follows

$$\left(x_{c,i} \;\; y_{c,i} \;\; z_{c,i}\right)^T = \mathbf{r}^{WC} \tag{3-76}$$

To calculate azimuth θ_i and elevation φ_i, firstly undistorted image coordinates $(u_{u,i},v_{u,i})^T$ are calculated via $h_u(u_{d,i},v_{d,i})^T$ (2-8). Then we calculate the directional vector $\mathbf{h}^W$ pointing from the cameras optical center towards the features' location in the world coordinate frame W. $\mathbf{h}^W$ is as follows

$$\mathbf{h}^W = \begin{pmatrix} h_x^W \\ h_y^W \\ h_z^W \end{pmatrix} = q2r(\mathbf{q}^{WC}) \begin{pmatrix} (u_0 - u_{u,i}) \dfrac{f}{d_u} \\ (v_0 - v_{u,i}) \dfrac{f}{d_v} \\ 1 \end{pmatrix} \tag{3-77}$$

where $q2r(\mathbf{q}^{WC})$ is the rotational matrix constructed from quaternion $\mathbf{q}^{WC}$ (3.29). From the directional vector $\mathbf{h}^W$, the azimuth θ_i and the elevation φ_i can be deduced, since it holds

$$\begin{pmatrix}\theta_i\\ \phi_i\end{pmatrix}=\begin{pmatrix}\arctan(h_x^W, h_y^W)\\ \arctan\left(-h_y^W, \sqrt{(h_x^W)^2+(h_z^W)^2}\right)\end{pmatrix} \tag{3-78}$$

After calculation of azimuth and elevation, only inverse depth p_i is left. This value is calculated from triangulation and details are given in section 3.4.2.

We find all values of (3-65) so that feature y_i can be placed into the state mean μ_t. After expanding μ_t we will expand the covariance $\sum_t$. The addition of feature y_i to covariance $\sum_{t,old}$ is given by

$$\Sigma_t = J\begin{pmatrix}\Sigma_{t,old} & 0 & 0\\ 0 & Q_I & 0\\ 0 & 0 & \sigma_\rho^2\end{pmatrix}J^T \tag{3-79}$$

where $\sum_{t,old}$ is the covariance before addition of the feature, $\sum_t$ is the covariance after addition of the feature. The dimensions are given as $\dim(\sum_t) = n_{dim} + 6 \times n_{dim} + 6$, $\dim(\sum_{t,old}) = n_{dim} \times n_{dim}$, $\dim(\sum_t) = n_{dim} + 6 \times n_{dim} + 3$, $\dim(Q_I)=2\times 2$. Matrix Q_I is a 2×2 matrix, containing the variance of the image measurement noise and is constructed similar to matrix Q_t' as in equation (3-58). Matrix J is constructed by a $n_{dim} \times n_{dim}$ identity matrix and the partial derivatives of function y. Matrix J has the structure given below.

$$J=\left(\begin{array}{cccc|cc} & & I & & & 0\\ \hline \frac{\partial y}{\partial \mathbf{r}^{WC}} & \frac{\partial y}{\partial \mathbf{q}^{WC}} & 0\ldots 0 & & \frac{\partial y}{\partial (u_d, v_d)} & \frac{\partial y}{\partial \rho}\end{array}\right) \tag{3-80}$$

where I is $n_{dim} \times n_{dim}$ identity matrix $\frac{\partial y}{\partial \mathbf{r}^{WC}}$ is given as

$$\frac{\partial y}{\partial \mathbf{r}^{WC}} = \begin{pmatrix} I \\ 0 \end{pmatrix}, \dim(\mathrm{I}) = 3 \times 3, \ \dim(0) = 3 \times 3 \tag{3-81}$$

and $\frac{\partial y}{\partial \mathbf{q}^{WC}}$ is given as

$$\frac{\partial y}{\partial \mathbf{q}^{WC}} = \begin{pmatrix} 0 & 0 & 0 & \frac{\partial \theta_i}{\partial \mathbf{q}^{WC}} & \frac{\partial \phi_i}{\partial \mathbf{q}^{WC}} & 0 \end{pmatrix}^T, \quad \dim\left(\frac{\partial y}{\partial \mathbf{q}^{WC}}\right) = 6 \times 4 \tag{3-82}$$

$$\frac{\partial \theta_i}{\partial \mathbf{q}^{WC}} = \frac{\partial \theta_i}{\partial \mathbf{h}^{W}} \frac{\partial \mathbf{h}^{W}}{\partial \mathbf{q}^{WC}} \tag{3-83}$$

$$\frac{\partial \phi_i}{\partial \mathbf{q}^{WC}} = \frac{\partial \phi_i}{\partial \mathbf{h}^{W}} \frac{\partial \mathbf{h}^{W}}{\partial \mathbf{q}^{WC}} \tag{3-84}$$

$\frac{\partial \mathbf{h}^{W}}{\partial \mathbf{q}^{WC}}$ denotes the Jacobian of directional vector $\mathbf{h}^{W}$ from camera to feature in world coordinates with respect to the orientation of the camera. The resulting Jacobians are

$$\frac{\partial \theta_i}{\partial \mathbf{h}^W} = \begin{pmatrix} \dfrac{h_z^W}{(h_x^W)^2 + (h_z^W)^2} \\ 0 \\ \dfrac{-h_x^W}{(h_x^W)^2 + (h_z^W)^2} \end{pmatrix} \tag{3-85}$$

$$\frac{\partial \phi_i}{\partial \mathbf{h}^W} = \begin{pmatrix} \dfrac{h_x^W h_y^W}{((h_x^W)^2 + (h_y^W)^2 + (h_z^W)^2)\sqrt{(h_x^W)^2 + (h_z^W)^2}} \\ \dfrac{\sqrt{(h_x^W)^2 + (h_z^W)^2}}{(h_x^W)^2 + (h_y^W)^2 + (h_z^W)^2} \\ \dfrac{h_y^W h_z^W}{((h_x^W)^2 + (h_y^W)^2 + (h_z^W)^2)\sqrt{(h_x^W)^2 + (h_z^W)^2}} \end{pmatrix} \tag{3-86}$$

$$\frac{\partial \mathbf{h}^W}{\partial \mathbf{q}^{WC}} = \begin{pmatrix} \dfrac{\partial q2r(\mathbf{q}^{WC})}{\partial q_r^{WC}}\mathbf{h}^W & \dfrac{\partial q2r(\mathbf{q}^{WC})}{\partial q_i^{WC}}\mathbf{h}^W & \dfrac{\partial q2r(\mathbf{q}^{WC})}{\partial q_j^{WC}}\mathbf{h}^W & \dfrac{\partial q2r(\mathbf{q}^{WC})}{\partial q_k^{WC}}\mathbf{h}^W \end{pmatrix} \tag{3-87}$$

Jacobians $\frac{\partial q2r(\mathbf{q}^{WC})}{\partial q_r^{WC}}, \frac{\partial q2r(\mathbf{q}^{WC})}{\partial q_i^{WC}}, \frac{\partial q2r(\mathbf{q}^{WC})}{\partial q_j^{WC}}$ and $\frac{\partial q2r(\mathbf{q}^{WC})}{\partial q_k^{WC}}$ are calculated according to (3-47) to (3-50).

3.8 Deleting a Feature [3]

In Visual Stereo SLAM, an increase in the number of features causes the state matrix X_t to grow and decrease the performance of EKF. In order to handle this situation, feature deletion mechanism is constructed. Some features are not good enough to have contribution to the state estimation. These features might be shadows or corners that do not match in the preceding frames. The EKF algorithm tries to match these features resulting in increase of the process time. Therefore these features should be deleted.

To delete a feature we simply remove it from mean and covariance. For example if the second feature is deleted, the deleted parts from state vector and covariance matrix are shown by lines below.

$$\begin{pmatrix} x_v \\ y_1 \\ \underline{y_2} \\ y_3 \end{pmatrix}, \begin{pmatrix} P_{xx} & P_{xy1} & \underline{P_{xy2}} & P_{xy3} \\ P_{y1x} & P_{y1y1} & \underline{P_{y1y2}} & P_{y1y3} \\ \underline{P_{y2x}} & \underline{P_{y2y1}} & \underline{P_{y2y2}} & \underline{P_{y2y3}} \\ P_{y3x} & P_{y3y1} & \underline{P_{y3y2}} & P_{y3y3} \end{pmatrix} \tag{3-88}$$

CHAPTER 4

IMPLEMENTATION

4.1 System Components

The system consists of two main components

• The camera system

• The processing unit

The camera system is mounted on a wooden structure and cameras are connected to computer by USB 2.0 cable.

4.1.1 Camera system

The camera system consists of two PHILIPS SPC530NC cameras. These cameras are low cost webcams which can be purchased from any store (The cost of one camera is 50 TL). They have a maximum 640×480 resolution. In order to improve performance 320×240 option is used. Moreover, they have VGA CMOS sensor and maximum frame per second is 30. The viewing angle of the cameras is 50 degrees. Frames are captured in RGB format.

The left camera is the reference camera in stereo system. Its focus is the origin of the camera coordinate frame. The cameras are fixed with a wooden structure which

provides them to stay together so that the calibration parameters are calculated once and reused. The rectangular shape of PHILIPS SPC530NC is an advantage in this case because the cameras are easily stuck together with this structure. The camera system with the wooden structure is given in Figure 4-1.

Figure 4-1 Camera System

4.1.2 The processing unit

The processing unit is a notebook which has an Intel Core 2 Duo 240 GHz processor with 3.00 GB RAM which is sufficient for image processing and EKF tasks. The operating system is Windows Vista. The programs in this thesis are written in MATLAB 2010a with m files. Moreover, some large loop functions are written with Lcc compiler in order to decrease the process time. Lcc compiler is a C/C++ compiler which is in Matlab environment. These files are written in C language and these files are converted to mex files. Mex files enable the C files to

be compiled and run in Matlab environment. These are done by writing a gateaway function to C files and compiling them with mex function in Matlab environment.

4.2 Camera Calibration

f_u and f_v which are mentioned in section 2.2, m_x and m_y ; the number of pixels per x and y unit of image coordinates, the coordinates of the principal point of the image and parameters of distortion models are called intrinsic parameters of camera and should be calculated by a camera calibration process. Moreover, extrinsic parameters of a camera should be calculated which are the translation coefficients [tx, ty, tz] and rotational parameters [α, β, γ]. In this thesis, Caltech Camera Calibration Toolbox [4] for Matlab is used. For the calibration procedure, a pattern which looks like a chessboard is photographed from various distances and angles for each of the cameras. These photographs can be seen in Figure 4-2.

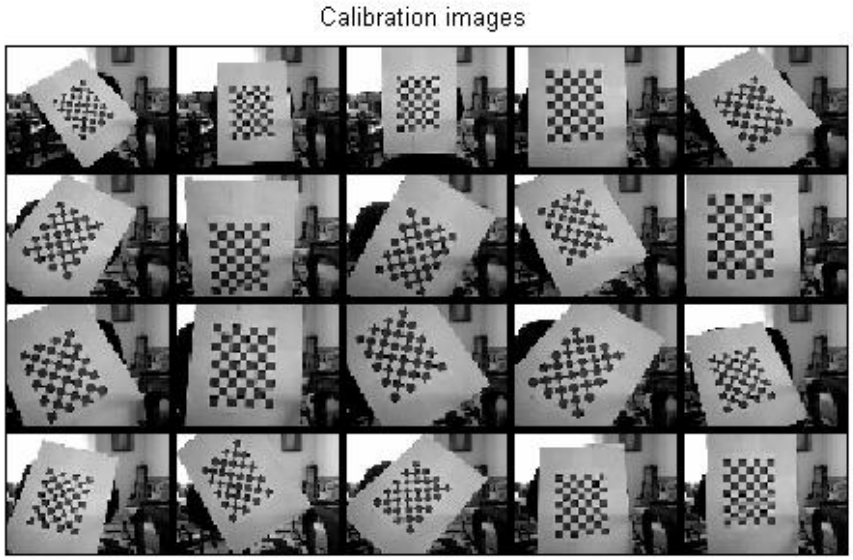

Figure 4-2 Calibration Images

After loading the photographs, the outermost corners of the patterns are selected manually for every frame one-by-one. After selecting corners, Camera Calibration Tool automatically finds the rest of the corners in the pattern as in Figure 4-3.

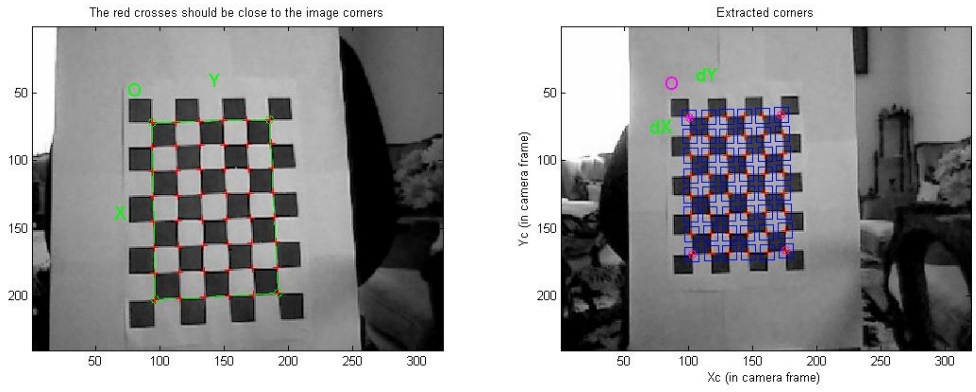

Figure 4-3 Extracting Corners

Then, calibration process is started and this process gives the intrinsic parameters of the cameras. By using these values, corners are recalculated automatically and intrinsic parameters are recalculated by smaller uncertainties. This process is applied both for left and right cameras. To calculate the extrinsic parameters for right and left cameras, stereo calibration option of calibration tool is used. Stereo calibration gives the extrinsic parameters of the cameras. After stereo calibration, relative positions of the patterns with respect to the cameras can be seen graphically in Figure 4-4.

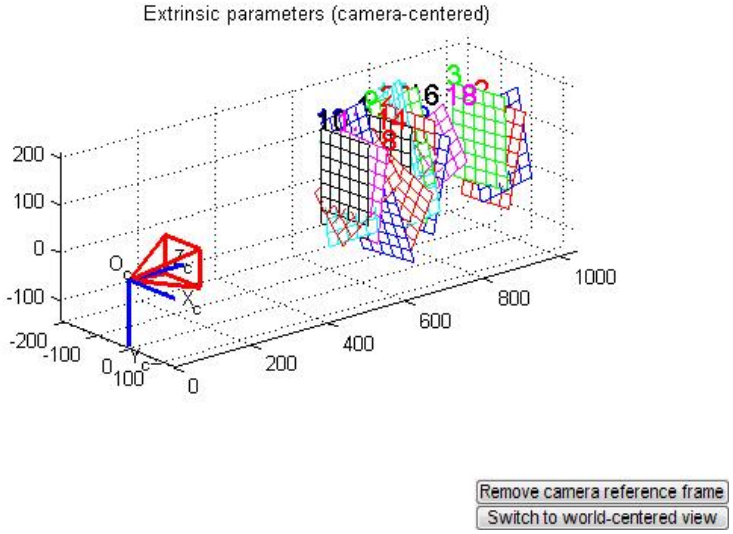

Figure 4-4 Positions of Patterns with Respect to Camera

Calibration process gives the extrinsic and intrinsic parameters as follows.

Table 4-1 Intrinsic Parameters

Intrinsic Parameters		
	Left Camera	**Right Camera**
$\mathbf{f_u}$ (pixels)	411.44455	409.98285
$\mathbf{f_v}$ (pixels)	411.48868	410.30331
$\mathbf{u_0}$ (pixels)	154.84445	167.80740
$\mathbf{v_0}$ (pixels)	128.01049	148.52453
s (degrees)	90	90
kc_1	-0.24774	-0.22438
kc_2	1.64851	1.19661
kc_3	0.00306	0.00525
kc_4	-0.00417	-0.00266
kc_5	0.00000	0.00000

Table 4-2 Extrinsic Parameters

Extrinsic Parameters		
	Left Camera	**Right Camera**
$\boldsymbol{\alpha}$ (radians)	0	0.05454
$\boldsymbol{\beta}$ (radians)	0	-0.01412
$\boldsymbol{\gamma}$ (radians)	0	0.00876
$\mathbf{t_x}$ (mm)	0	-36.31831
$\mathbf{t_y}$ (mm)	0	0.28972
$\mathbf{t_z}$ (mm)	0	-3.30263

By using extrinsic parameters, rotation matrix between cameras is calculated from equation (2-15).

$$\hat{R} = \begin{pmatrix} 0.99986 & -0.00914 & -0.01387 \\ 0.00837 & 0.99847 & -0.05457 \\ 0.01435 & 0.05444 & 0.99841 \end{pmatrix} \quad (4\text{-}1)$$

4.3 Steps of Stereo SLAM Algorithm

Steps of the Stereo SLAM Algorithm are given in figure 4-5 and in this chapter each step is analyzed in detail.

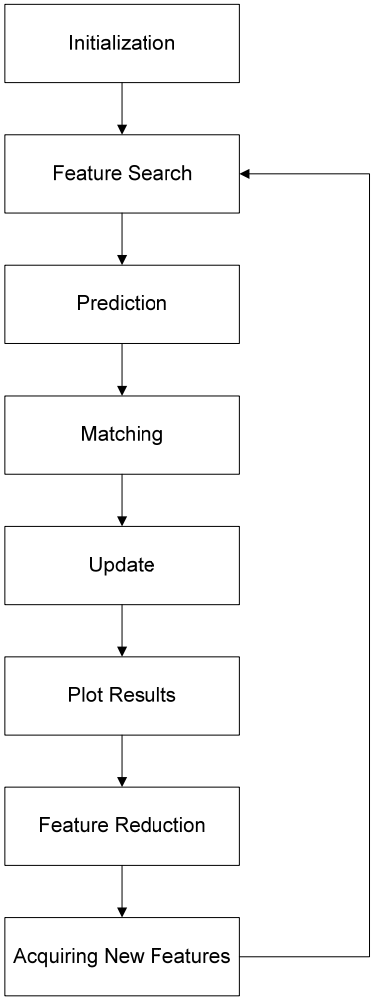

Figure 4-5 General Process

Firstly, necessary parameters for SLAM process are initialized. This includes initial state vector and covariance matrix which are described in section 3.2.1, camera parameters which are obtained from camera calibration matrix, covariance matrix noises and feature number. Secondly, the cameras are initialized. There are two reasons for this step. Firstly, we need some amount of code to use cameras in Matlab. The other reason is that the initial frames taken from cameras are blurry and dark so we have to wait to acquire good frames.

After taking the frames, good features are searched. Initially, the whole pixel points in a frame are searched and found features are added to the state vector and covariance matrix. Also their patches are saved. The details are given in section 4.5.

Now the process enters to a cycle where calculated state vector and covariance matrix are the start of new cycle's old state vector and covariance matrix. This cycle is started by "feature search" step.

Since initialized features will not be observable by time with movement, new features are needed. The details are similar to initialize new feature and will be presented in section 4.5.

After the features are acquired, new position of the camera is predicted. This calculation is explained in detail in section 3.3. After prediction, positions of the features are predicted by using the movement estimation of the camera. The features are matched between consecutive frames so that the displacement can be estimated. The calculations are given in 3.5.

After prediction and matching process, new state vector and covariance matrix are estimated so new variables are ready for next state. SLAM process is composed of prediction, matching and update steps. Results of the Stereo SLAM Algorithm are displayed in plot results part.

In some cases, bad features are observed which increase the process time and have no correction on state vector and covariance matrices. These features are deleted in "feature reduction" step. Finally, the new frames are acquired from cameras for the next cycle.

4.4 Initialization

Initialization step of Stereo SLAM Algorithm consists of three phases. In the first phase, the parameters used for the system are initialized. Then the camera system hardware is set up and the first pair of stereo images is acquired. By using the stereo pair images initial features are initialized. Initialization of the features is similar to feature search step and this will be explained together in "Feature search".

4.4.1 Initialization of Parameters

Firstly the parameters which are necessary for Stereo SLAM Algorithm process are initialized. These parameters are listed below.

a) Initial State Vector: Initial State Vector contains position, orientation, velocity and angular velocity of the camera system, features and explained in section 3.2.1. Since there is no feature at this section, initial value of this parameter is

$$\mathbf{x}_v = [0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0] \tag{4-2}$$

where its initial position is [0, 0, 0], initial angles are [0, 0, 0], initial velocity is [0, 0, 0] and initial angular velocity is [0, 0, 0]. Since we use quaternion [0, 0, 0] angles are translated to quaternions which are [1, 0, 0, 0].

b) Initial Covariance: Since there is no feature in the state vector at the beginning of Stereo SLAM Algorithm, initial covariance value is a 13x13 matrix and its initial value is given in section 3.2.1 which is,

$$P = \begin{bmatrix} 0 & \dots & 0 & 0 \\ \vdots & \ddots & 0 & 0 \\ 0 & \cdots & 0 & 0 \\ 0 & \cdots & 0 & V \end{bmatrix} \quad , V = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.5 \end{bmatrix} \tag{4-3}$$

c) Camera Parameters: Since we use camera parameters in calculations, parameters calculated from Camera Calibration Toolbox given in section 4.2 are initialized.

d) Covariance Matrix Noises: In this part, movement and pixel noises are initialized and added to Covariance Matrices. Human hand which moves randomly is modeled as movement noise and predicted to stay in the limits of noise variables. Observations of pixels from cameras are noisy and this is modeled with pixel noise.

e) Minimum Feature Number: Minimum limit of feature number is initialized in the beginning and can be changed by user. In the experiments this number is taken as 15. The details will be handled in section 4.6.

4.4.2 Camera Initialization

In order to initialize the camera parameters, we need some amount of code in Matlab. Also for acquisition of the frames, it is necessary to wait for a couple of seconds since initial frames taken from cameras are blurry and dark. The following steps are processed.

a) Assigning of Resolution: Since 320x240 resolution is used for cameras we have to start the cameras with this property.

b) Colorspace Selection: Since cameras use RGB format this property is selected.

c) Pause Time: To acquire good frames mentioned above we pause the Matlab for 2 seconds.

4.4.3 Acquiring First Frame Pair

In Matlab connecting to cameras are handled by Image Acquisition Toolbox. The fastest way to take frame in this toolbox is with "getdata" function but when we use this function, more than one frame might be taken. To solve this problem the memory is cleared with "flushdata" function and only the newest frame is taken from the cameras in real time operation. In order to acquire first frame pair the following steps are done by Stereo SLAM Algorithm;

- During the 2 second pause time and in initialization of cameras, Matlab acquires some frames and these frames have to be deleted from memory. In order to do this "flushdata" function is used for two of the cameras.

- After "flushdata" is executed, new frames are waited from cameras. Since frames per second of the cameras are 30, algorithm waits maximum 33 ms for a new frame pair. So by using "flushdata" and "getdata" function we acquire a pair of frame with a maximum 33 ms time difference. By that way first frame pair is synchronized.

- Since we acquire frames in RGB format we have to transform it to grayscale format to use in Harris Corner Detector Algorithm. This is done by “rgbtogray” function.

- Since cameras give frame values in INT8 precision this value has to be changed to double precision to use in Harris Corner Detector. After this step frame values are ready to be used in initialization of new features.

4.5 Feature Search Algorithm

In this subsection since we nearly use the same algorithm for “feature search” and “initialization of features”, these steps will be explained together. Firstly we explain how a feature is initialized and then explain the difference between feature search and initialization of features.

In the beginning we have to initialize features in order to keep camera pose. Because of camera movement or features that are not repeatable or strong will reduce the feature number available in camera’s view. In order to handle this problem we have to initialize new features whenever feature number decreases below a threshold. Features are initialized in two ways. Initially since there is no feature, feature search algorithm will initialize features in each place of the frame taken at start. By the time when feature number decreases we add features one-by-one in each cycle. There are four major steps of feature algorithm which are shown in figure 4-6 and explained in detail.

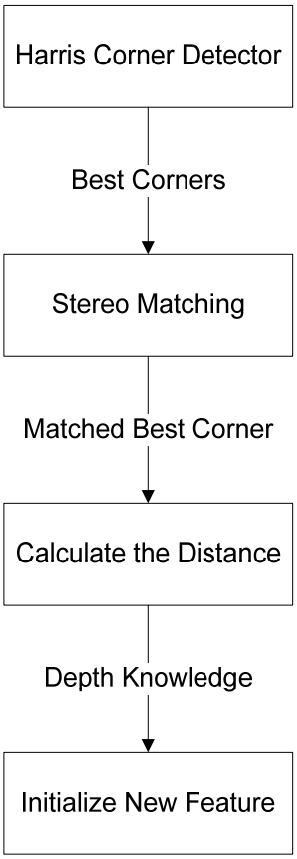

Figure 4-6 Feature Search Algorithm

4.5.1 Harris Corner Detector

When we get a frame in order to find repeatable and strong features we use Harris Corner Detector. Since we run the algorithm in different areas where illumination changes significantly, determining a threshold for Harris Corner Detector is not applicable. In order to handle this problem locally best corners are found in all over the frame but borders and these corners are sorted from best to worst according to the formula 2-23. Then a predetermined number of locally best corners are taken. This number is 30 but can be changed via code. Since we use 11×11 patch for features, 5 pixels at 4 of the borders of a frame there is no need for corner search. Since Stereo SLAM Algorithm's center is left camera, Harris Corner Detector is only applied to frame acquired from left camera.

4.5.2 Stereo Matching

After finding the corners P_t (t < number of best corners) in the frame taken from left camera, we try to match these points with their stereo pairs in the frame taken from right camera. Since cameras are so close to each other, instead of searching the stereo pairs of Pt = (u_t, v_t) in epipolar line we search them in a rectangular box around u_t and v_t. Rectangular box size is 15x15 and can be changed via code. The stereo pairs P't = (u'_t, v'_t) are found by NCC which is described in section 2.8.

4.5.3 Distribution of Features

To gain more information from environment, the features should be distributed in the frame. Assume that the features are grouped. In this situation, the information taken from the features depends only on some part of the image and this leads to calculation errors if these features are not reobservable by time. Moreover, instead of using potential features taken from other regions of the scene, we have to trust only a grouped feature. To handle this situation and to gain more information from images, a minimum threshold distance thr_{dist} is placed between features. This threshold is 20 pixels and can be changed via code. If features P_t = (u_t, v_t) is detected initially, minimum distance for a potential feature P_p = (u_p , v_p) given as

$$\sqrt{(u_t - u_p)^2 + (v_t - v_p)^2} >= thr_{dist} \tag{4-4}$$

should be held. If this property holds then we can use P_p as a feature. In other words potential feature should not be close to any other initialized feature less than this threshold.

4.5.4 Depth Calculation

Before using the obtained features we should calculate the depth of these features with respect to the camera's center. This is performed by triangulation which is described in section 2.7.

One trick for depth is that if we use close features to camera we have to search more pixel points than a far feature. Also this affects the processing time. In order to handle this situation a filter is added to the system. If depth is less than a threshold this feature is not initialized. This threshold is initialized as 2 m and can be changed via code.

4.5.5 Initialize New Feature

We know from section 3.7 that features are initialized by 6 parameters.

$$\mathbf{y}_i = \begin{bmatrix} x_{c,i} & y_{c,i} & z_{c,i} & \theta_i & \varphi_i & p_i \end{bmatrix}^T \tag{4-5}$$

where $x_{c,i}$, $y_{c,i}$, $z_{c,i}$ are placed by 3D camera position, θ_i and ϕ_i are the azimuth and elevation of the feature and p_i is the inverse depth of the feature.

By triangulation we calculate the depth of the features. When we initialize a new feature, the size of state vector and covariance matrix increase as described in section 3.7. Additionally the patch is saved in order to use and match it in preceding frames and save the place of the feature in state vector in order to match to its patch.

4.5.6 Difference between "Initialization of Features" at start and "Feature Search" step

When we initialize the system at start since there is no feature in scene we have to search the whole frame pixels. However, in some regions of the frames the number of features decreases and eventually becomes zero by the movement of the camera. Because of the limitation of time, we cannot search whole frame pixel points at every acquisition of a frame. To solve this problem, we check whether there is a decrease in feature number at each cycle and if there is, a new feature is initialized. To decrease the process time a frame is divided into 6 regions. When a decrease in number of features is detected, the number of features in whole region of the frame is counted. Feature is initialized in the region where there is the lowest number of features. Again the whole process is repeated as explained at start of this subchapter.

4.6 SLAM process

SLAM process consists of 3 main parts. These parts are prediction, matching and update parts. These parts are indeed steps of a Kalman Filter Algorithm. In section 3 all these parts and their calculations are given but in this section a brief summary will be given. In this step we only use left camera's frames because of time limitations but in section 4.11 some extended work by using stereo camera calculations are given.

4.6.1 Prediction step

In prediction step we calculate where the camera system will be after the movement calculated in the previous step. The detailed calculations are given in section 3.3. By using this predicted step we calculate where we will observe the features by the movement of the camera.

4.6.2 Matching

After predicting the positions of features, we search these predicting features in the acquired frames. In the predicted areas we try to match saved patches of features with the possible patch candidates. This step is the most time consuming part of Stereo SLAM Algorithm because all possible area is searched and each candidate patch is compared with patch of feature. These possible candidates are sorted by their NCC score which is described in section 2.7 and the NCC score of the best match is bigger than a threshold, this patch is selected as the new location of feature. This threshold should be selected carefully because if a lower value is selected, wrong matchings may occur. If this number is selected high, then matching may not occur. In this thesis this number is selected as 0.8. The matched points are taken as measurements and recorded for using in update step. Additionally same calculation technique is used for matching in EKF and matching of stereo pairs.

4.6.3 Update

In update phase Kalman Gain is calculated and state vector and covariance matrix are updated with respect to the calculations in prediction and matching phase. Updated state vector and covariance matrix are used for next EKF step's previous calculation. Also in this step normalization of quaternions are calculated. Details of this section are given in section 3.6.

In some cases, there are two calculated match candidates for a feature. When a wrong patch in search area of this feature is observed, the matched point oscillates between true and wrong match candidates. This fact leads to wrong measurement calculations in Stereo SLAM Algorithm. In update step, in order to prevent oscillations of features, if observed movement of a feature is more than expected

then its effect on Kalman Filter is set to be zero in order to prevent oscillation of the estimations. However, this procedure is not a solution all the time.

4.7 Plot Results

In plot results part, two main estimations are plotted one of which is feature positions on a frame and the other one is camera location with respect to the initial position of the camera. Observed feature locations are placed on acquired frame and plotted in the left window of user interface. Each camera location is recorded to memory. Final location and movement of the camera with respect to initial position are plotted in the right window of user interface.

4.7.1 SLAM Graphical Interface

In order to run the algorithm in Matlab environment, a user interface is used as in figure 4-7 and 4-8. When we push the "Start" button, the algorithm and cameras start. It is important to note that cameras should be directed to a rich feature environment at start. The left window in figure 4-8 shows the current frame in grayscale and red plus signs show the features tracked by the system. The right window shows the estimated position of the camera system with respect to the initial point (0, 0, 0).

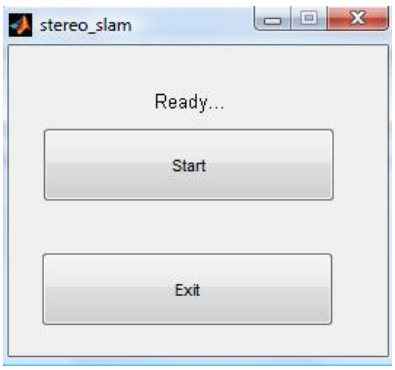

Figure 4-7 Buttons of the system

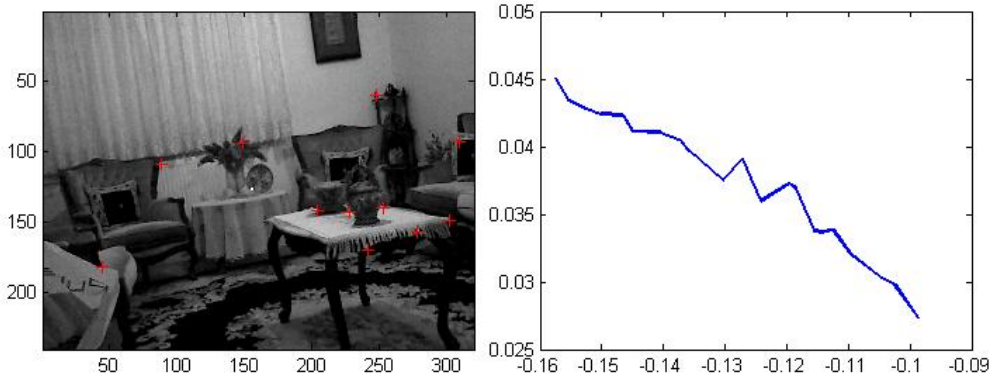

Figure 4-8 Online System Windows

4.8 Feature Deletion Algorithm

In some situations, the features are not reobservable and they should be deleted from the state vector and covariance. For example Harris Corner Detector may initialize a feature which is a shadow and from different angles this feature may not reobservable. To solve this problem, these features should be detected and instead of them new features should be initialized.

There is one criterion for a feature in order to delete it. They should not match for a long time consecutively. If there is match between features between consecutive frames, this helps the system to improve localization estimation. We can call these features healthy but if they do not match, we increase the number of unmatched counter assigned to the feature. If it exceeds a predetermined threshold then we delete this feature from state vector.

4.9 Acquiring Frames in Matlab

Since all calculation is done, we need frames for new cycle. Cameras always send their acquired frames to processor. Matlab saves these frames to memory. If more than one frame is saved then Stereo SLAM Algorithm uses the most recent frame and deletes the others to clear the memory. If no frame is received, Stereo SLAM Algorithm waits for a frame to be acquired. This frame is converted to grayscale and double precision. Stereo SLAM Algorithm only receives frames from right camera when algorithm needs new features. Whether a feature is initialized or not, frames of the right camera are deleted after the "feature search" step.

4.9.1 Synchronization of the Cameras

In this part the details of the acquisition of frames by Matlab are explained. When the cameras are initialized and started, they start to acquire frames and Matlab scores them in a buffer. In Stereo SLAM Algorithm, to calculate the depth of features, stereo image parts have to be acquired nearly in the same time. Also in EKF calculations, to estimate the localization, angle, velocities and feature positions the most recent frame has to be taken. In order to do these tasks two functions are used. These are "getdata" and "flushdata" functions. "getdata" can acquire any frame stored in the buffer. By using this function Stereo SLAM Algorithm acquires the most recent frame. "flushdata" function deletes all frames in the memory. All frames are deleted in each cycle because frames in the buffer lead to memory size errors.

Since Stereo SLAM Algorithm starts the cameras in different times, it deletes all frames available and waits for new frames to be acquired when it needs first stereo pair. Demonstration of this part is given in figure 4-9.

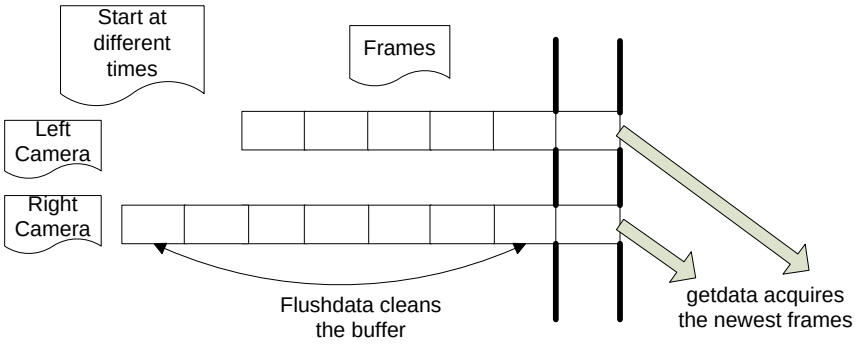

Figure 4-9 Image Acquisition at Start

During the Stereo SLAM process, if more than one frame is acquired then last acquired frames are taken by "getdata" function and other frames are deleted.

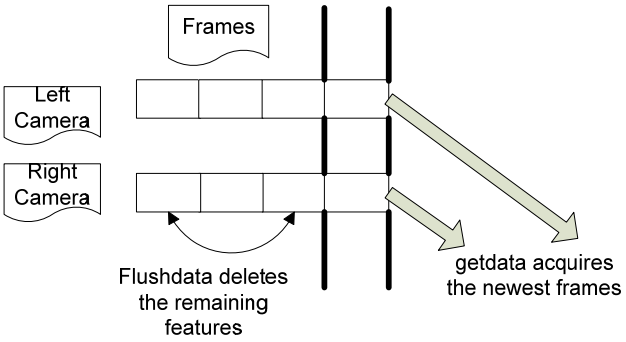

Figure 4-10 Image Acqusition During Process

Also if no feature is acquired, Stereo SLAM Algorithm waits for a frame to be acquired by Matlab.

4.10 Stereo Camera Usage and Single Camera Usage

In Stereo SLAM Algorithm, sometimes stereo image pairs are used and single image from left camera is used. Since depth of the features cannot be calculated by mono camera, stereo image pairs are used in feature initialization and feature search steps. The estimations of SLAM process are done by single camera usage. This is

because of time constraint of SLAM process. If stereo image pairs are used instead of single camera, the time cost of the calculations of SLAM process will be doubled. Additionally Stereo SLAM Algorithm processes with 10 fps and uses 320x240 image resolution for faster implementation than 640x480 resolution. 320x240 image resolution reduces time cost of process especially in feature searching and conversions of frames to grayscale and double precision.

4.11 Other Possible Stereo SLAM Implementations

In this section conceptual designs for Stereo SLAM Algorithm will be given during the design process and these designs were discussed but not implemented because of limited time usage. They can be implemented in future.

4.11.1 Near online version

In order to achieve online process, Stereo SLAM Algorithm only processes recent frames acquired by cameras with 10 frames per second. Nevertheless Stereo SLAM Algorithm can process all frames. In this situation Stereo SLAM Algorithm does not work real time but it is close to real time with an increasing delay. This process is done as follows: As given in section 4.8, Matlab records the frames in a buffer. If the oldest two frames are taken by Stereo SLAM Algorithm as the frames acquired, Stereo SLAM Algorithm uses each frame. However, since frames acquired faster than Stereo SLAM Algorithm process, frames can overload the memory in long use.

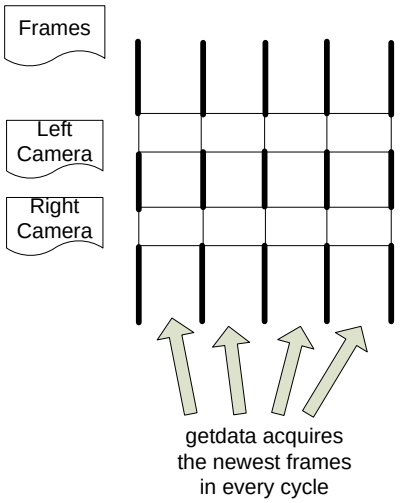

Figure 4-11 Near Online Usage

In this usage, frames received by right and left camera have to be taken by "getdata" in order to synchronize them. Otherwise, Stereo SLAM Algorithm may use frames with different time stamps which may results in wrong feature depth calculations or Stereo SLAM Algorithm may not find any stereo pair for initialization of features.

4.11.2 Adding Stereo Depth Calculations of Features

In this thesis prediction, the measurement and update parts are implemented by structure from motion principle and using only one camera. To improve these calculations, stereo depth calculations can be added to the Stereo SLAM Algorithm. For this purpose, the stereo depth of the observed features is calculated and these calculations are connected to measurement calculations in each cycle. This part needs 3 main calculations: Frame acquisition from right camera in each cycle, finding stereo pairs of features in each cycle (which increases the cost of matching process twice) and calculating the Kalman gain and covariance matrix of this measurement.

4.11.3 Twice Measurement Calculations

The measurement step can be applied to right camera separately from left camera. Since cameras are stuck together, when we calculate the placement of left camera we also calculate right camera. By using this property, after prediction step possible position of right camera and features can be predicted by right camera. After feature positions are estimated, Stereo SLAM Algorithm has two measurements instead of one. In update step these measurements are combined and new state can be estimated. However this means two times more implementation of whole process.

4.11.4 Combined Implementation of 4.10.1 and 4.10.2

Algorithms mentioned in 4.10.1 and 4.10.2 can be combined together with triple cost of process time.

4.11.5 640x480 resolution

To improve calculations of depth estimations of features, 640x480 resolution can be used since a 1x1 area in 320x240 resolution is displayed by 2x2 areas in 640x480 resolution which increases the sensibility of the measurements. On the other hand it also increases the computational time. It also means that Stereo SLAM Algorithm obtains better calculations in matching phase but an increased area is searched with respect to 320x240 resolution.

4.12 Feature number

In EKF algorithm, feature number must be limited because it is the most important part that increases the computational cost of the EKF algorithm. Moreover it should be as many as possible to get information from the scene. So that feature number should be a reasonable value in order to achieve a real time operation. In visual

SLAM, the feature number is taken as 15 for a frame but can be increased or decreased easily by the given code. When we initialize features, this number is taken into account but it is possible to observe more features then this threshold due to camera movement.

4.13 Decreasing the Process Time in Matlab

Since Matlab is a slow program and we need online process in Stereo SLAM algorithm, we have to decrease the process time of one cycle of Stereo SLAM system. There are several steps which are implemented in the code of Stereo SLAM system. Some of them are basic tricks of Matlab which can be found in help topics of this program but some of them are found during the implementation of the code. These are given as follows:

1. Preallocation of Memory: To decrease the process time of the codes Matlab variables are initialized before the process. Whenever we initialize a new variable Matlab allocates a memory for this variable. Whenever we increase the size of this variable Matlab again allocates a memory space for variable. In order Matlab to do this process once, variables are initialized first and then used in the Stereo SLAM algorithm.

2. Decreasing Repeated Calculations: Another simple trick is that repeated calculations are reduced to one. In Stereo SLAM algorithm some calculations are used repeatedly.

3. Decreasing the Plotting Times: In Stereo SLAM we can plot the results for every cycle, also we can decrease the process time by plotting the results once for up to five cycles(also we can give the results once in ten or more which is user dependent) .

4. No display at Command Window: Since showing an output on command window in Matlab is time consuming we don't give any outputs to command window of Matlab.

5. No Usage of Functions: A function has its input and output parameters. When we run a function in Matlab, entrance and exit of the variables is time consuming especially if these variables are large. Due to this fact, the functions are handled in main function with the help of cells.

6. Using mex Files: To improve the speed, large calculations are handled in C language in Matlab with the help of mex files.

7. Activating mex Files Once: Especially for matching process, large for loops occur during search of image patches. These for loops are inserted to mex files. Activating a mex file consumes minimum 25 μs process time. In first glance it seems a small number but if a feature is tracked in 10x10 area and 10 features are tracked totally, a process time of 25 ms is consumed only for activating mex files. Also several functions are combined in other parts of the system in order to decrease the processing time.

4.14 Usage of C Language in Matlab

Since Matlab is a slow program and we need a fast code to simultaneously receive images and process them, C codes are executed in Matlab codes. C codes written in Matlab are called mex files which stand for Matlab executable. Mex files are used in Matlab for two reasons.

• Calling large pre-existing C programs from Matlab without rewriting them as Matlab functions.

• Replacing performance-critical routines with C implementations.

We use second property of the mex files (Matlab is a high productivity environment and make it easy to program specific applications and eliminates low-level programming in complied languages). In order to increase the performance of the code, we make low-level programming. For example in Matlab there is no need for a variable to define it before, but in C we have to allocate a variable before the usage of the variable. Additionally when we compile a mex file we can use it like a Matlab m file.

A mex file composed of two parts

• A "gateaway Routine" that interfaces C and Matlab data

• A "Computational Routine" written in C that performs the computations we want to implement in the mex file.

A gateaway Routine is the entry point to the mex-file shared library. It is through this routine that Matlab accesses the rest of the routines in mex-files.

A sample gateaway routine is given below.

Void mexFunction(int nlhs, mxArray *plhs [ ], int nrhs, const mxArray *prhs [ ])
{ C code }
We take the input variables with prhs array and we define the output parameters with plhs array. Moreover we have to add the header file
include "mex.h"
at the start of the algorithm. For example if we define a function

X= myFunction (Y, Z);

prhs array contains two variables Y, Z and plhs is initially empty and after the run of the code it is assigned with the variable X. An example about a mex-file is given below.

```
#include "mex.h"
#include "math.h"
void numden(double *I1,double *num, double *den)
{
    double mean1,std1,a;
    int i,sizemtx;
    mean1=0;
    std1=0;
    sizemtx=121;
    for (i=0;i<sizemtx;i++)
    {
        mean1=I1[i]+mean1;
    }

    mean1=mean1/sizemtx;

    for (i=0;i<sizemtx;i++)
    { a=I1[i]-mean1;
        std1=a*a+std1;
    }
    std1=sqrt(std1/sizemtx);

    for (i=0;i<sizemtx;i++)
    { num[i]=(I1[i]-mean1);}
    den[0]=std1;


}
void mexFunction( int nlhs, mxArray *plhs[],
                  int nrhs, const mxArray *prhs[])
{


    double *I1,*num,*den;
    I1 = mxGetPr(prhs[0]);
    plhs[0] = mxCreateDoubleMatrix(121,1,mxREAL);
    plhs[1] = mxCreateDoubleMatrix(1,1,mxREAL);
    num = mxGetPr(plhs[0]);
    den = mxGetPr(plhs[1]);
```

```
    numden(I1,num,den);
}
```

At the start

include “mex.h”

is added because this library is needed in mex files

include “math.h”

is added since we use the math library in the code.

Then numden(...) function is given.

```
void mexFunction( int nlhs, mxArray *plhs[],
                  int nrhs, const mxArray *prhs[])
```

is written since we need in mex files.(gateaway routine)

Then we define the input and output parameters I1, num and den. Since I1 is an input parameter it is scalar with mxGetPr(prhs[O]) function. mxGetPr takes real datas from Matlab . By help of mxCreateDoubleMatrix we create arrays in mex files and we assign them to the output variables. After writing an executable C program in Matlab it is compiled with mex function in command window and it is ready for use.

CHAPTER 5

EXPERIMENTS AND RESULTS

Since all estimations of the Stereo SLAM Algorithm cannot be recorded and ground truth is not known in online mode, an offline simulation version is constructed in order to test the performance of the system. In this mode, Stereo SLAM Algorithm acquires frames from a previously recorded video. In these videos, camera system location and its orientation are known with respect to the initial pose in the frames. Moreover offline simulation version mode gives us an opportunity to test the system during the development time. By the help of this mode, several experiments have been carried out in a room sized environment to evaluate the system performance and these results are discussed in this section.

In order to operate the system, good features should be observed in the environment. For this reason, room is selected with many furniture, chair, tables, pictures etc. to contain good features. Moreover, in order to estimate the localization and map, reference system is needed. This is done by building a grid system with piece of papers on a table. Grids are built in 1cm squares. So by help of this grid system we can exactly know the pose of camera system with respect to the initial coordinates. Since we cannot fill the rooms with piece of papers, distances of features are measured with external ruler. A picture of the room is given in figure 5-1. To increase the possibility to find features some papers are added to the scene.

Figure 5-1 Room View

5.1 Explanation of the Signs in Windows

When the simulation mode is started, two windows and two buttons appear as in figure 4-7 as in the online mode. Last acquired frame is shown in the left window given in figure 5-2. On this window, features observed on the frame are shown by green plus signs. Around these signs, there are red circles. These circles show where the system searches the features. When a feature is matched in consecutive frames, circle's colour becomes red. In other case its colour becomes blue. In the right window localization of camera with respect to the initial point is plotted. The red points show the localization of features. Similar to left window, there are circles which surround these red points. They show the uncertainty in the localization of the features. The blue plus sign shows the current place of the camera system and the ellipsoid which surrounds this point shows the uncertainty in the placement of

the camera. The blue line gives the past directory of the camera system. The right window has grids which show the distances of features and the distance traveled by the camera. These grids can be configured by code.

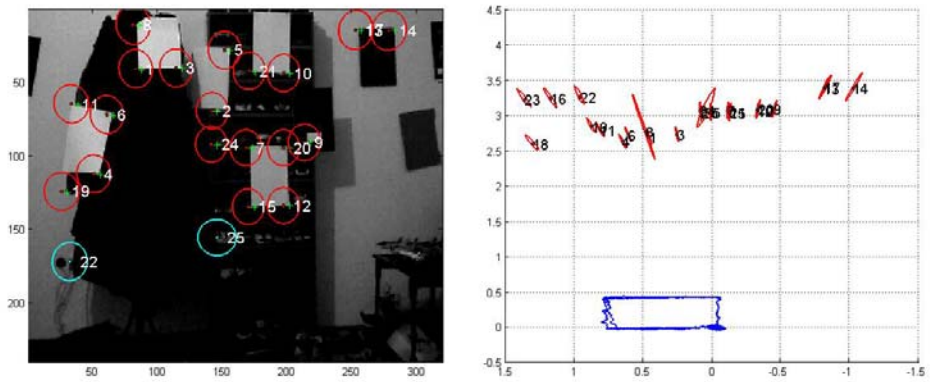

Figure 5-2 Offline System Windows

5.2 Depth Measurement Test

In this experiment, the real and measured distances of the features are compared. In order to do this, the depths of features are measured by feature search algorithm, real distances of the features to the camera are measured with a ruler. As expected if distances are calculated more accurately, the Stereo SLAM Algorithm gives better results. Therefore, to operate Stereo SLAM algorithm accurately, the measurements should be accurate enough and error range should not be high.

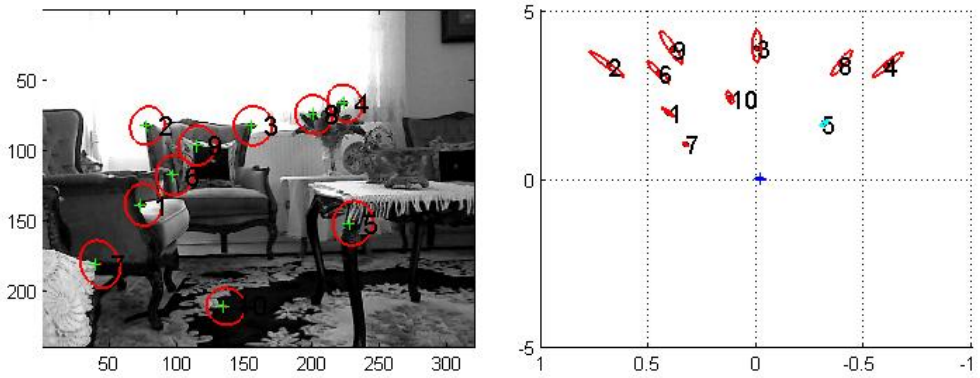

Figure 5-3 Depth Measurement Test Features

Depths of 10 random features given in the figure 5-3 are measured by the feature search algorithm. Also the real distances of these features are measured. Since center of camera system is located at the focus of left camera, the depths are calculated from the features to the left camera. The results are given in table 5-1.

Table 5-1 Real and Measured Depths

Feature No	**Real Depth(m)**	**Measured Depth (m)**	**Error (%)**
1	2.10	1.99	5.3
2	3.66	3.43	6.3
3	3.96	3.87	2.3
4	3.65	3.42	6.4
5	1.70	1.65	3.0
6	3.01	3.15	4.6
7	1.15	1.08	6.1
8	3.75	3.42	8.8
9	3.70	3.84	3.7
10	2.30	2.43	5.6

The results are good because the measured values are near to real values. Since we use 320x240 low cost cameras the errors are a little high. Moreover uncertainties in the results of the calibration lead to error in depth measurement. We think if we use a higher resolution camera and with better calibration parameters, the error percentage in the depth estimates will be better.

5.3 Right and left motion experiment

At this experiment we record videos for each of the cameras. The frames of the videos are taken such that each frame location is known with respect to the initial point. The camera system is moved to the +X direction at each step 0.01m and the state vector of the system is recorded and compared to real value at each frame. Total motion is 2m to the +X direction with respect to the initial position and we expect similar estimation from the results of the Stereo SLAM system. After moving 2m to the +X direction, camera turns back to its initial point and at the end the system is expected to give the initial values of the state vector. Moreover 2m path is selected such that the features in the initial frame dissappears totally and reobserved when camera turns back to its initial pose. Additionally in this experiment there are 399 steps. First 200 steps are to the +X direction and the rest are to the –X direction. Finally we convert the quaternions to ordinary Euler angles for easy follow. This is done by the formula given below [25]. To acquire frames in each 1cm, a small railway is built for camera system. In each step camera is moved 1cm to X direction on this railroad. The picture of the railroad is given in figure 5-4. Also in figure 5-5 signs on this way are given. First movement of the camera to the +X direction is signed with movement 1 and then turns back to its original position with movement 2. In figure 5 marks on this system are given. These signs are marked in each 1cm. Also in figure 5-6 the output of the system at 200th step and at figure 5-7 the output of the system at 399th step are given.

$$\begin{pmatrix}\phi\\ \theta\\ \varphi\end{pmatrix}=\begin{pmatrix}\arctan\left(\dfrac{2(q_0 q_1+q_2 q_3)}{1-2(q_1^2+q_2^2)}\right)\\ \arcsin\left(2(q_0 q_2-q_1 q_3)\right)\\ \arctan\left(\dfrac{2(q_0 q_2-q_1 q_3)}{1-2(q_2^2+q_3^2)}\right)\end{pmatrix} \tag{5-1}$$

where ϕ is the rotation about the X axis

θ is the rotation about the Y axis

φ is the rotation about the Z axis

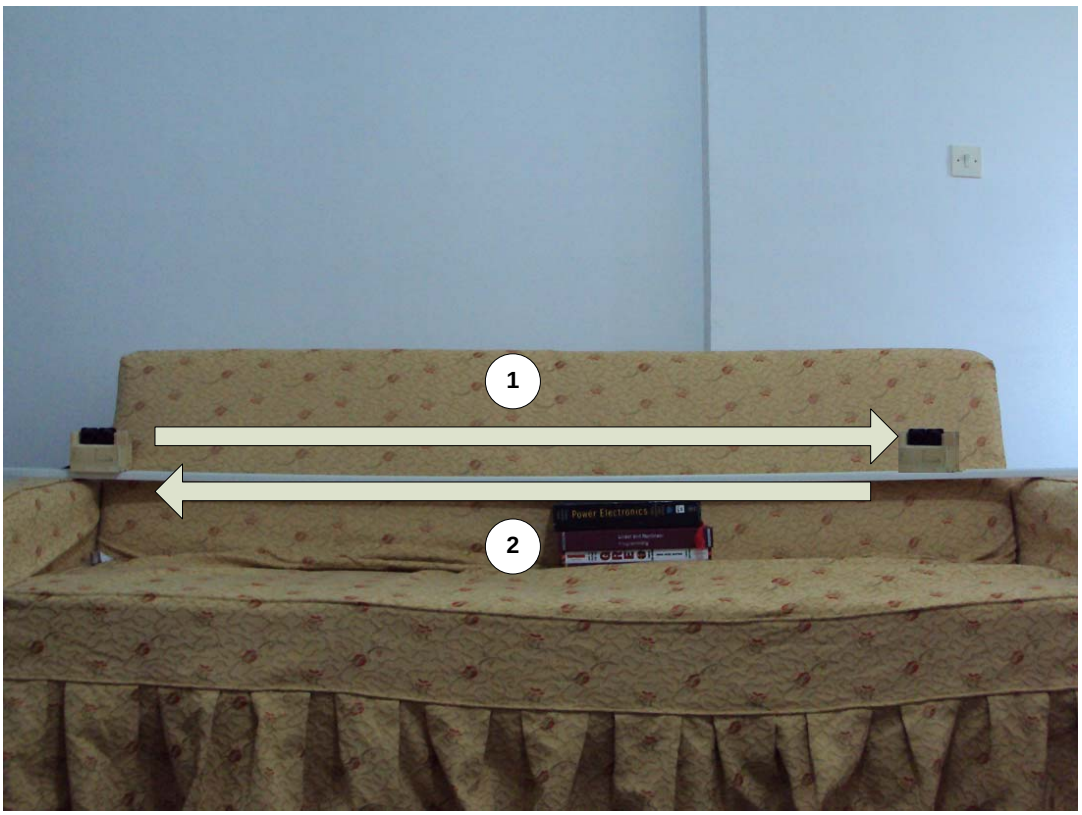

Figure 5-4 Railroad System

Figure 5-5 Signs on the system

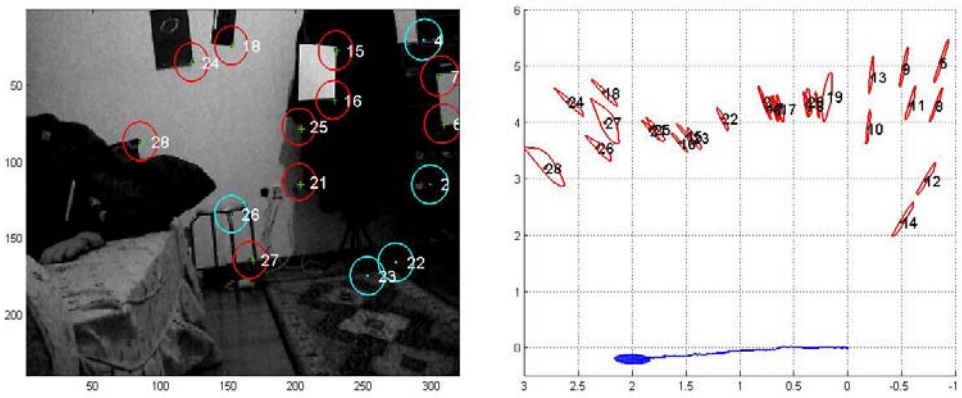

Figure 5-6 200.Step (Experiment 2)

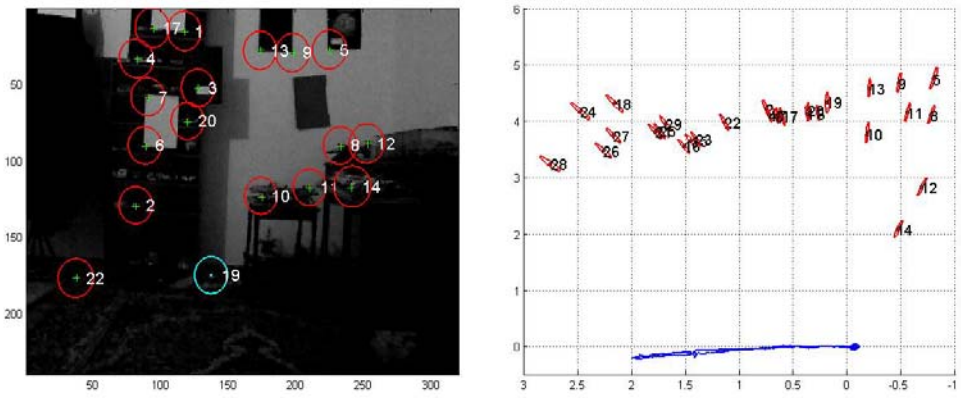

Figure 5-7 Last Step (Experiment 2)

In this experiment, the movement is only occurred in +X and –X direction. So firstly the movement in X direction will be analyzed. Estimated location and ground truth in X direction are given in figure 5-8. As seen from the graph, the estimation results are close to the real values. The difference between real and estimated values is given in figure 5-9. From this figure, maximum error in placement is 15.7cm at 93th step. Average error in X direction location is 6.1 cm. Since the locations of the features are estimated better as they are observed, in –X direction, location error is lower than in +X direction. Average location error in +X direction steps from 1 to 200 is 7.6cm and average error in –X direction steps from 201 to 399 is 4.7cm. This fact can be observed in other results of the experiment.

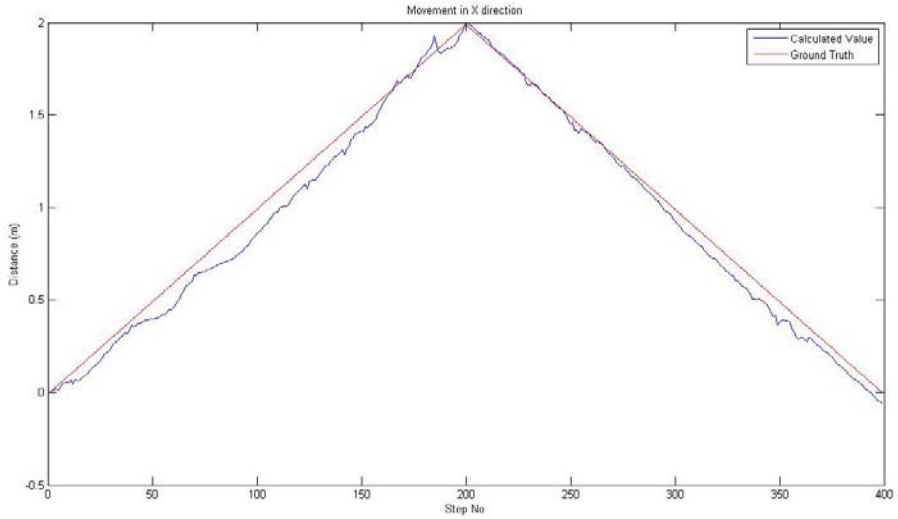

Figure 5-8 Movement in X direction (Experiment 2)

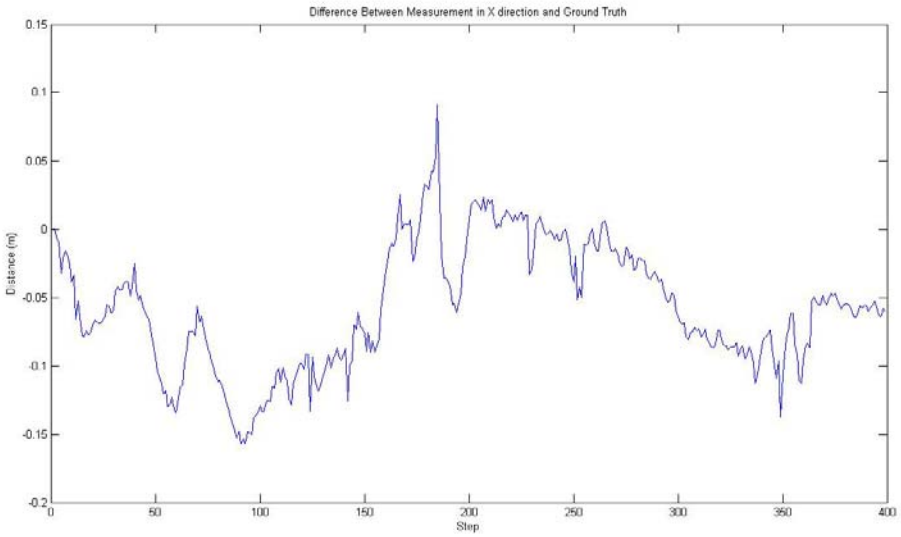

Figure 5-9 Difference between Real and Estimated Value (Experiment 2)

In figure 5-10 and 5-11 location estimation in Y and Z direction are given. No change in estimated values is expected in these directions. Nevertheless, the noise in position of the features and wrong estimations of feature locations lead to error in Y and Z direction location estimation. Maximum location estimation errors in X, Y and Z directions are given in table 5-2. Also from this table it is observed that mean

errors from step 1 to 200 are higher than mean error from 201 to 399 steps because of the same reason mentioned for X direction. At the last row of this table combined error in all directions is given.

Table 5-2 Errors in Localization (Experiment 2)

Direction	**Max Error (step) (cm)**	**Mean Error (cm)**	**Mean Error at 1:200 step (cm)**	**Mean Error at 201:399 step (cm)**
X	93. / 15.7	6.1	7.6	4.7
Y	142. /13.6	3.9	4.3	3.5
Z	197. /20.20	5.8	6.4	5.1
Total Error	197. /21.35	11.0	12.5	9.5

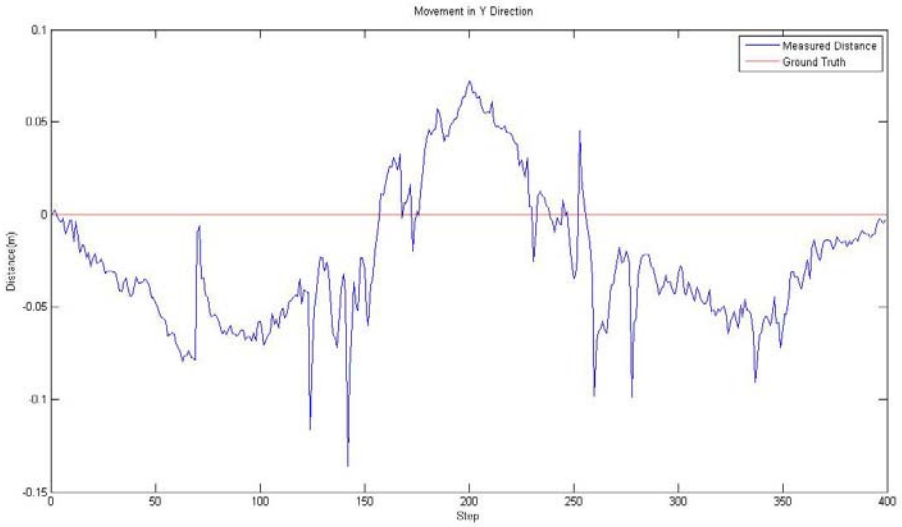

Figure 5-10 Movement in Y Direction (Experiment 2)

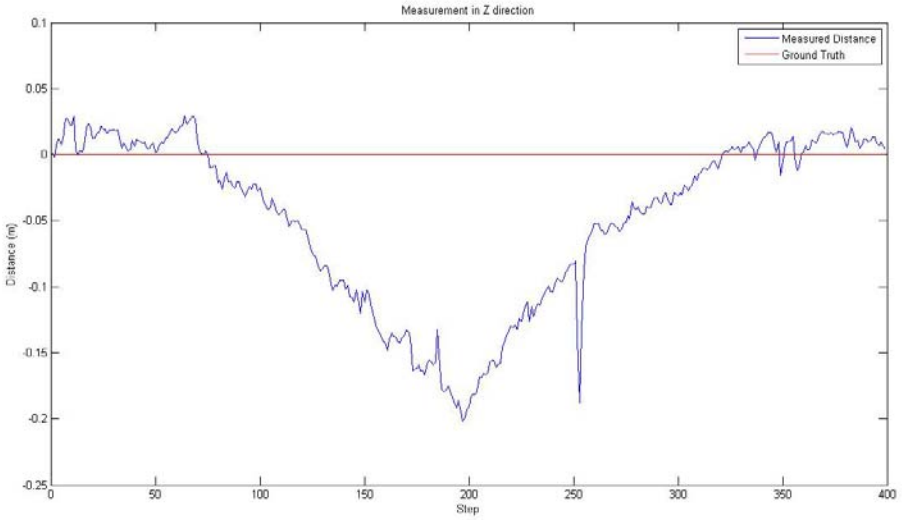

Figure 5-11 Movement in Z Direction (Experiment 2)

Since angular values of the camera system remain same with respect to the initial position of the camera, no angle change should be observed in any direction. However because of wrong placement of the camera, error in frame depth estimations, error in frame location calculations, briefly noise in estimations lead to change in angle values. In table 5-3 maximum and mean errors in angle estimations are given. Also in figure 5-12 to 5-14, real and the estimated angle values are given. As seen from the results, errors in angle estimations are small enough to process Stereo SLAM Algorithm.

Table 5-3 Errors in Angle Estimations (Experiment 2)

Direction	Max Error (step) (deg)	Mean Error (deg)	Mean Error at 1:200 step (deg)	Mean Error at 201:399 step (deg)
X	93. / 15.7	6.1	7.6	4.7
Y	142. /13.6	3.9	4.3	3.5
Z	197. /20.20	5.8	6.4	5.1
Total Error	197. /21.35	11.0	12.5	9.5

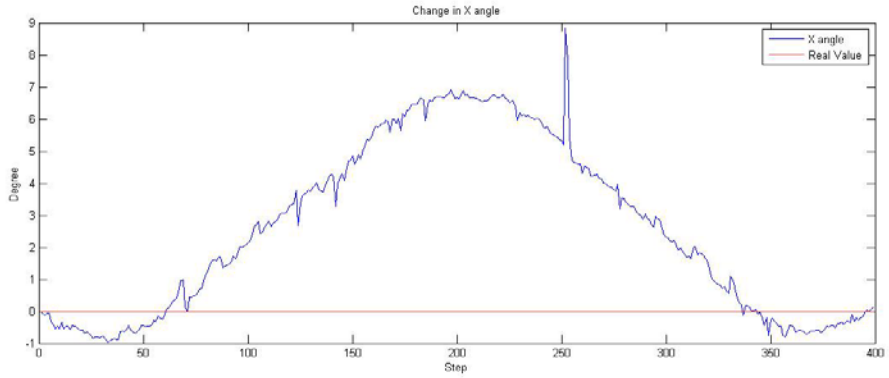

Figure 5-12 Estimated and Real X Angle (Experiment 2)

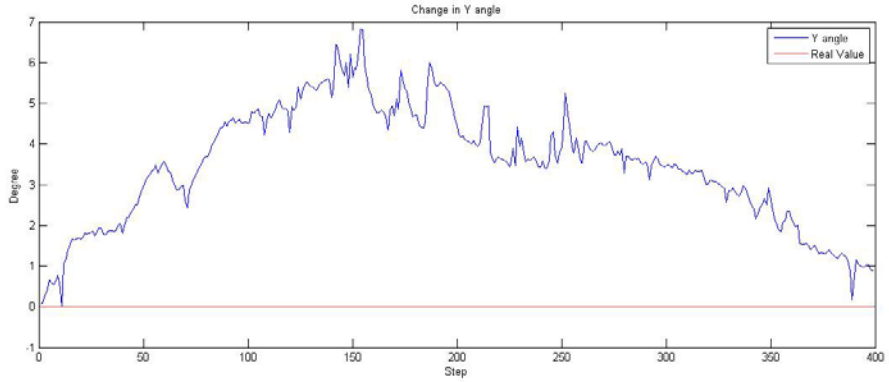

Figure 5-13 Estimated and Real Y Angle (Experiment 2)

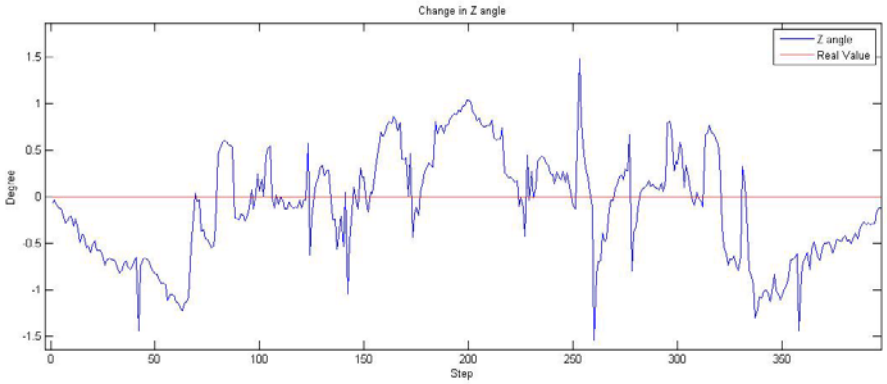

Figure 5-14 Estimated and Real Z Angle (Experiment 2)

Since only change in location occurs on X direction, velocity estimations are expected only in X direction. From step 1 to 200, 0.3m/s velocity and for the rest of the steps -0.3m/s velocity is expected in X direction. In table 5-4 maximum and mean errors are given. In figure 5-15 estimated and real velocity values in X direction are given. Estimated X velocity oscillates around 0.3m/s from step 1 to 200 and -0.3m/s from step 200 to 399.

Table 5-4 Errors in Velocity Estimations (Experiment 2)

Direction	Max Error (step) (m/s)	Mean Error (m/s)	Mean Error at 1:200 step (m/s)	Mean Error at 201:399 step (m/s)
X	12. / 0.68	0.118	0.144	0.092
Y	70. /1.05	0.092	0.098	0.086
Z	253. /1.21	0.084	0.083	0.085

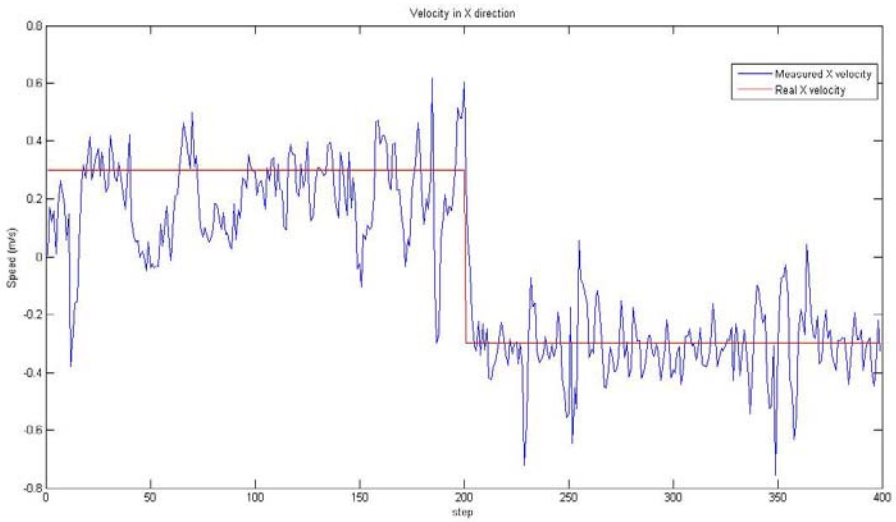

Figure 5-15 Estimated and Real Velocity in X direction (Experiment 2)

In figure 5-16 feature number in the state vector and observed features in the frame are given. In this experiment minimum feature number observed in a frame should be minimum 15. If feature number decreases below this value feature search algorithm adds a new feature to the state vector. When observed feature number decreases below 15, a new feature is initialized in the preceding step. Additionally when camera moves in –X direction more features are observed than +X direction. One of the other reasons for better location estimation is the result of this fact.

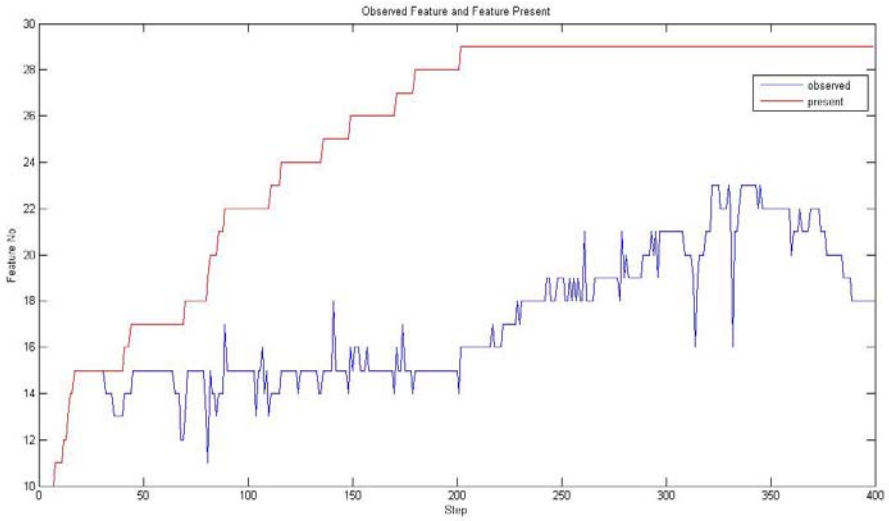

Figure 5-16 Observed Feature and Present Feature (Experiment 2)

Covariance values for X, Y and Z direction are given in figures 5-17, 5-18 and 5-19 respectively. Also mean values are calculated from these results. It is expected that mean errors for localizations are in the limits of 2 sigmas. From table 5-5 it is confirmed. As a result, covariance values give sufficient information for ground truth. Finally, since same features are observed after 200th step, there is a sharp decrease in covariance values.

Table 5-5 Comparison of Covariance Value and Mean Error (Experiment 2)

Direction	1:399 mean (cm)		1:200 mean (cm)		201:399 mean (cm)	
	2*sigma	error	2*sigma	error	2*sigma	error
X	12.0	6.1	13.6	7.6	6.7	4.7
Y	6.1	3.9	6.6	4.3	5.6	3.5
Z	5.2	5.8	6.1	6.4	4.3	5.1

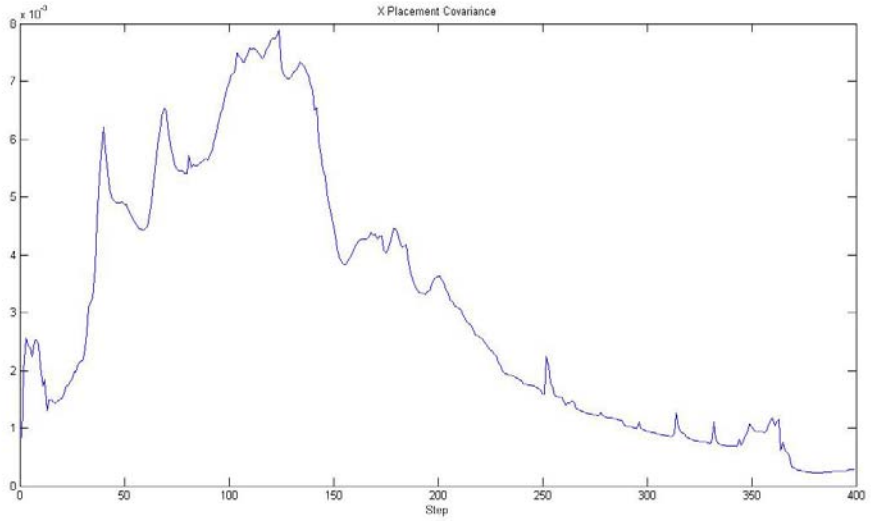

Figure 5-17 X Localization Covariance (Experiment 2)

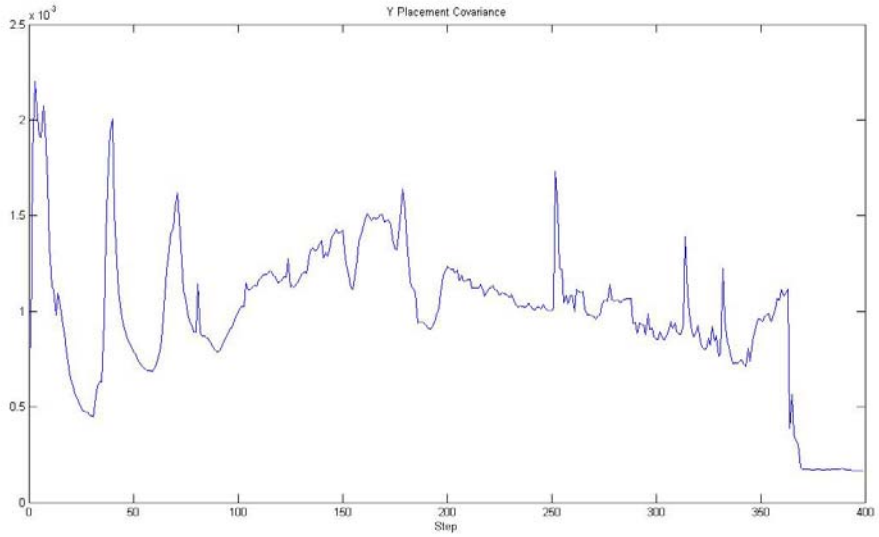

Figure 5-18 Y Localization Covariance (Experiment 2)

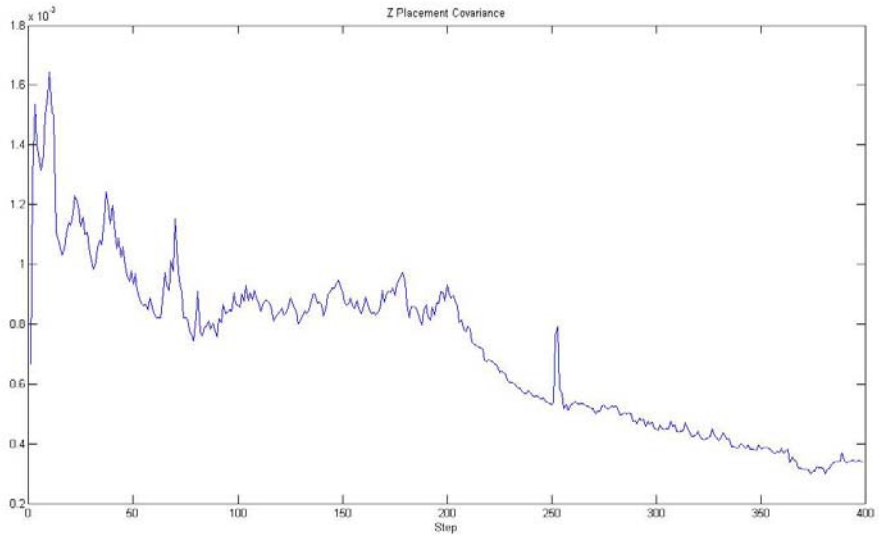

Figure 5-19 Z Localization Covariance (Experiment 2)

To summarize, Stereo SLAM Algorithm estimates close to real values with some error which can be guessed from the covariance matrices. Also no drift is observed from the results of localization estimation. At final step loop closure occurs within a small error (6.01 cm). Also the ground truth stays in 2 sigma range which can be observed from covariance matrices.

5.4 Loop Experiment (Experiment 3)

In Loop experiment, estimations of the Stereo SLAM Algorithm in a rectangular loop will be analyzed. The grid system which is built on a table is given in figure 5-20. Loop movement consists of four steps which are given as follows: 75 cm to the +X direction, 45 cm to the +Z direction, 75 cm to the -X direction and 45 cm to the -Z direction. These movements are signed as 1, 2, 3 and 4 in figure 5-20 respectively. Also closer look to grid system is given in figure 5-21. In each step camera is placed on lines and these lines are signed with step of the system. Frames are acquired every 1 cm and totally there are 240 steps in a loop. To observe the effects of reobserving same scene, cameras travel this loop two times. In each acquired frame angle of the camera is same with the initial position. No angle is expected in the angle estimations. Also output of the Stereo SLAM Algorithm at step 240 and step 480 are given at figure 5-22 and 5-23. As in experiment 2 in all figures real and estimated values are given.

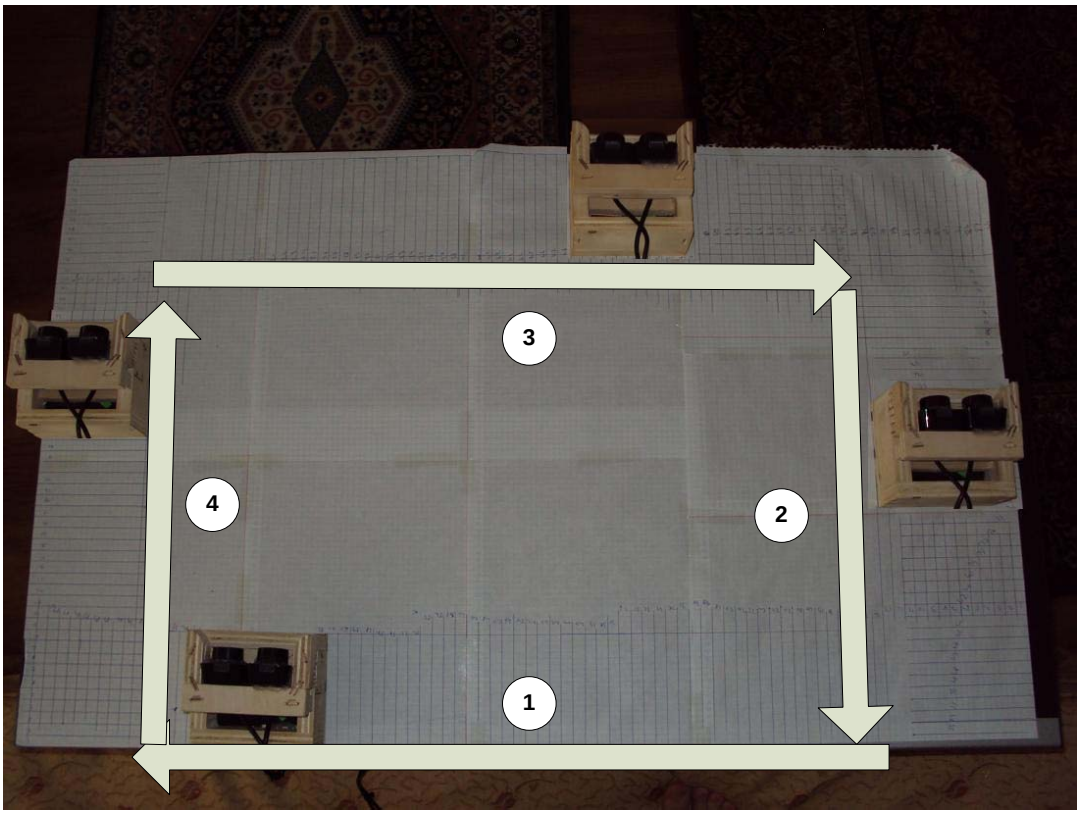

Figure 5-20 Grids and The Movement (Experiment 3)

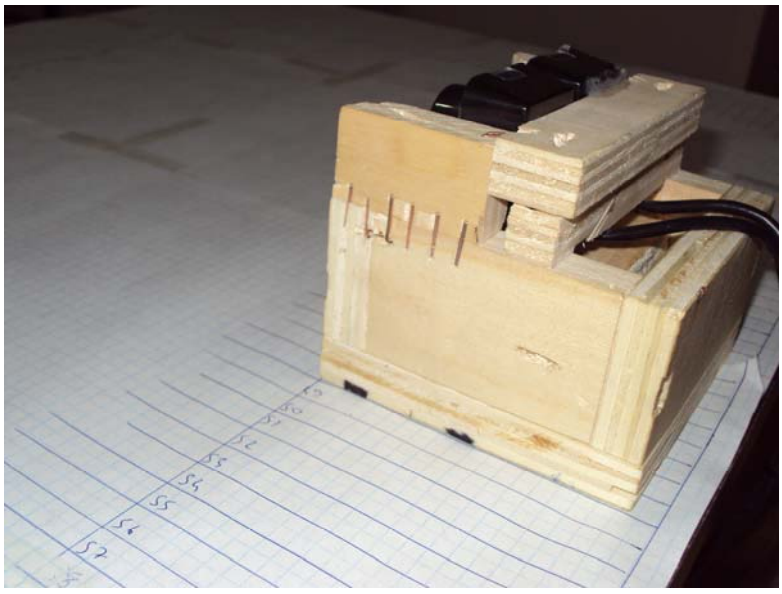

Figure 5-21 A Closer Look to Grids (Experiment 3)

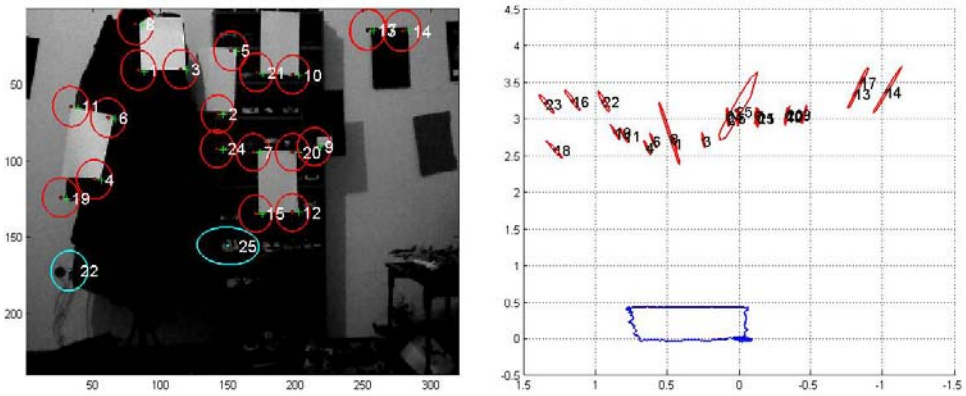

Figure 5-22 First Loop Closure of the Experiment (Experiment 3)

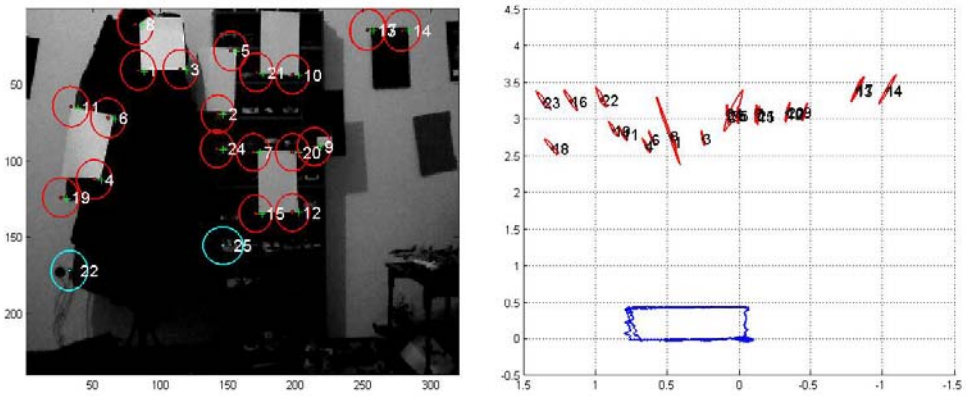

Figure 5-23 Second Loop Closure of the Experiment (Experiment 3)

Since we have successful matching of features in preceding frames and variance of camera location estimation is good, variance of feature search areas are generally constant and 10. This can be seen visually in figure 5-22 and 5-23. At the end of each successful loop closure "variance of search areas" are nearly same. Since noise is added to covariance matrix, this value cannot be below 10. Stereo SLAM

Algorithm searches the match candidates of features in these search areas. Some variance of features search areas are given in figure 5-24. It is observed that features covariances are around 11.

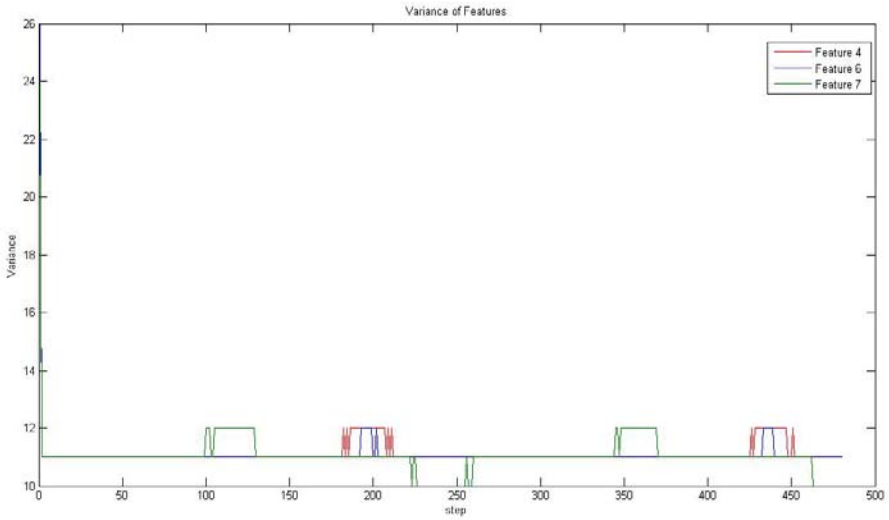

Figure 5-24 Variance of Search Area of Features (Experiment 3)

Now the localization output of the Stereo SLAM Algorithm will be analyzed. In this experiment movement occurs only on X and Z direction. Noise in estimation of the depth of the features and error in calculation of feature positions lead to observation of localization estimation noise in Y direction. In table 5-6 maximum errors, mean error, mean error in first loop, and mean error in second loop are given. Also in figures 5-25, 5-26, 5-27 the estimated and real values of localization in X, Y, Z directions are given. From table 5-6 it is observed that in second loop estimations are closer to ground truth because of better localization estimations of features.

Table 5-6 Errors in Localization (Experiment 3)

Direction	Max Error (step) (cm)	Mean Error (cm)	Mean Error at first loop (cm)	Mean Error at second loop (cm)
X	44. / 16.9	3.9	5.3	2.4
Y	195. /3.1	1.0	1.1	1.0
Z	86. /6.2	2.2	2.1	2.2

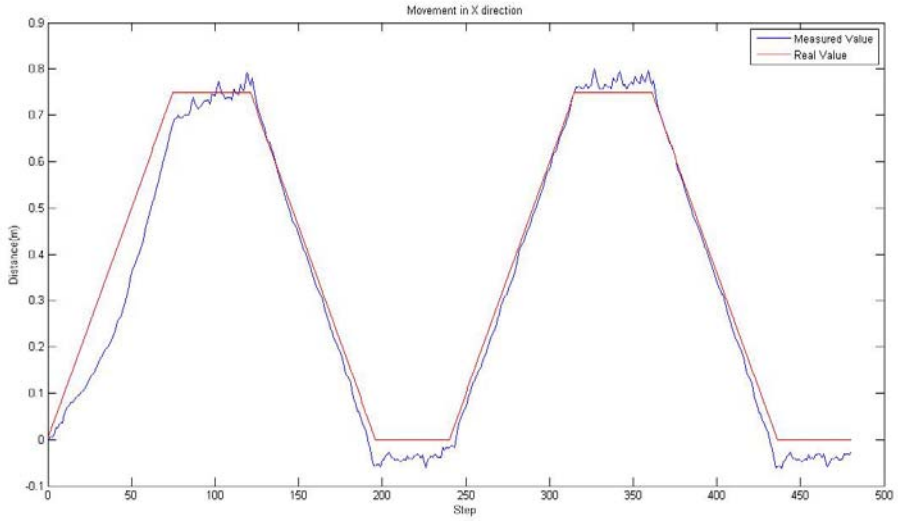

Figure 5-25 Movement in X direction (Experiment 3)

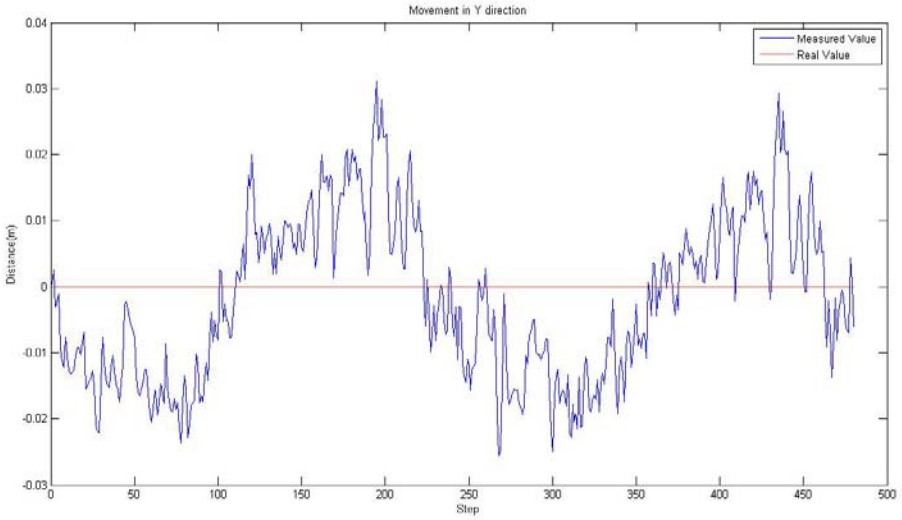

Figure 5-26 Movement in Y direction (Experiment 3)

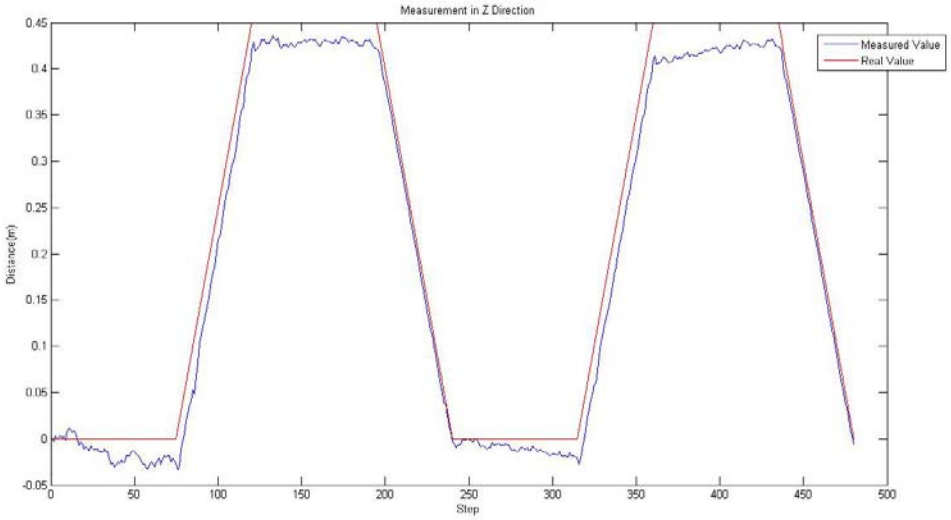

Figure 5-27 Movement in Z direction (Experiment 3)

In figure 5.28 and 5.29 output of X and Z directions are given together. From graphs it is seen that estimated values are close to ground truth and loop closure occurs at steps 240 and 480 within a few cm error.

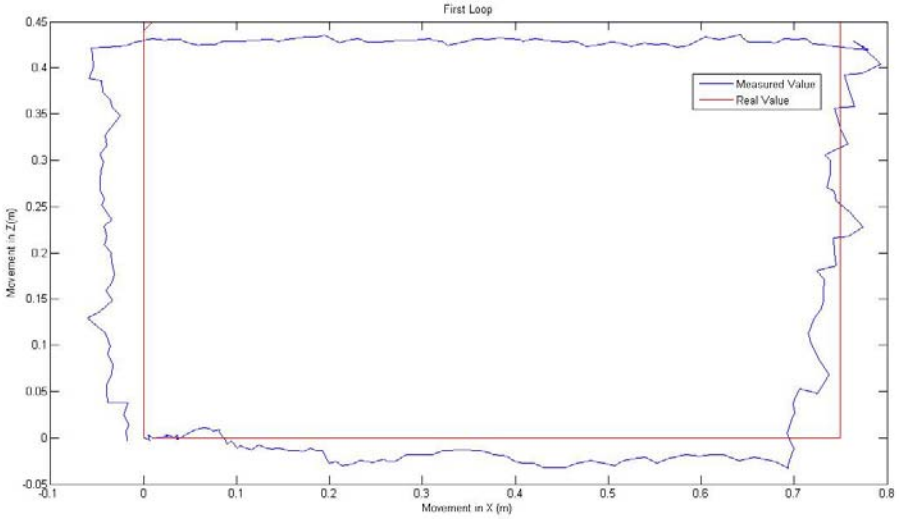

Figure 5-28 First Loop (Experiment 3)

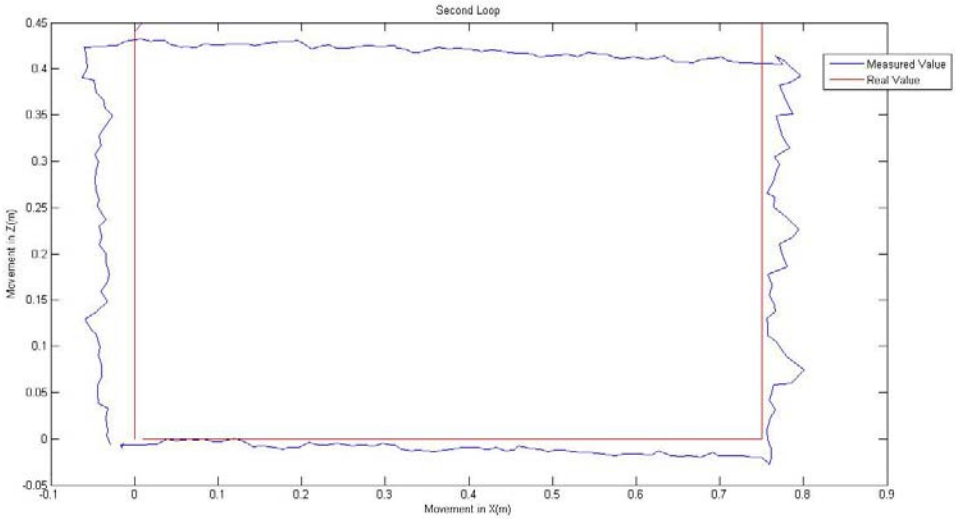

Figure 5-29 Second Loop (Experiment 3)

Secondly velocity estimation of the system will be given. When camera is moved in a direction, 0.3 m/s velocity at that direction should be observed and at Y direction no velocity estimation is expected. At table 5-7 maximum errors, mean error, mean error for first and second loop are given. In figure 5-30 to 5-32 estimated velocities with real values are given. From results we observe that Stereo SLAM Algorithm estimates velocities with some error and estimated values oscillate around the real values.

Table 5-7 Errors in Velocity (Experiment 3)

Direction	Max Error (step) (m/s)	Mean Error (m/s)	Mean Error at 1:200 step (m/s)	Mean Error at 201:399 step (m/s)
X	327. / 0.38	0.086	0.092	0.080
Y	271. /0.25	0.047	0.045	0.049
Z	76. /0.45	0.053	0.058	0.047

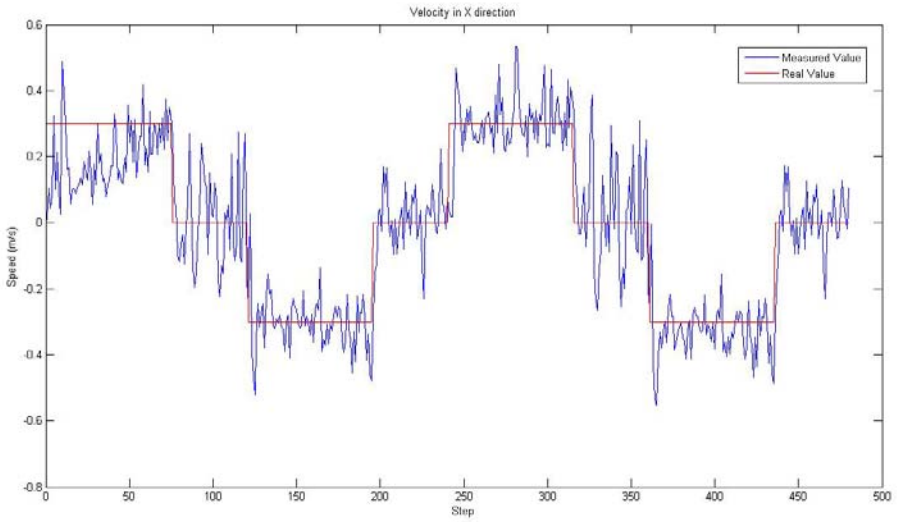

Figure 5-30 Estimated and Real Velocity in X direction (Experiment 3)

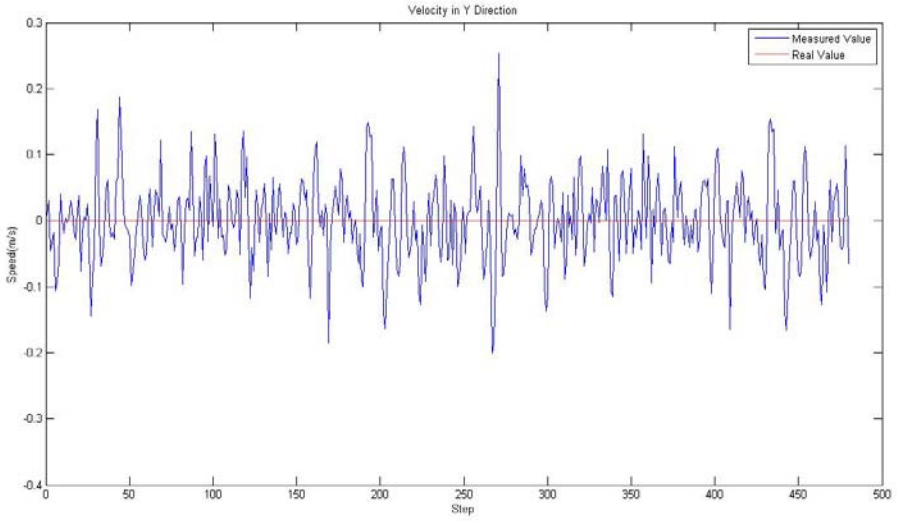

Figure 5-31 Estimated and Real Velocity in Y direction (Experiment 3)

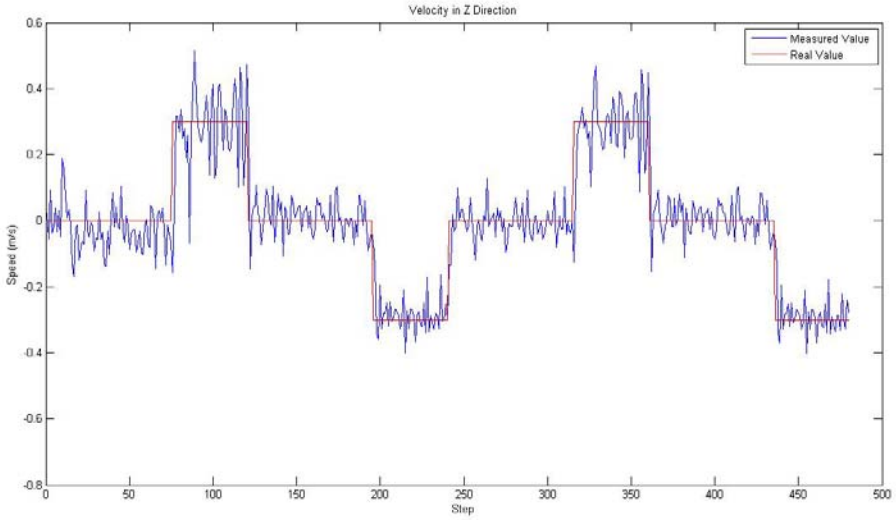

Figure 5-32 Estimated and Real Velocity in Z direction (Experiment 3)

Since camera pose remains same in all steps, the angle estimations of Stereo SLAM Algorithm are expected around zero. In figures 5-33 to 5-35 estimations of X, Y, Z angles are given. Also in table 5-8 max error, mean error values are given. From results, SLAM Algorithm estimates angle values around the expected value with small errors.

Table 5-8 Errors in Angle (Experiment 3)

Direction	Max Error (step) (deg)	Mean Error (deg)	Mean Error at 1:200 step (deg)	Mean Error at 201:399 step (deg)
X	311. / 0.67	0.28	0.24	0.31
Y	36. /3.58	1.03	1.38	0.69
Z	120. /0.89	0.19	0.20	0.17

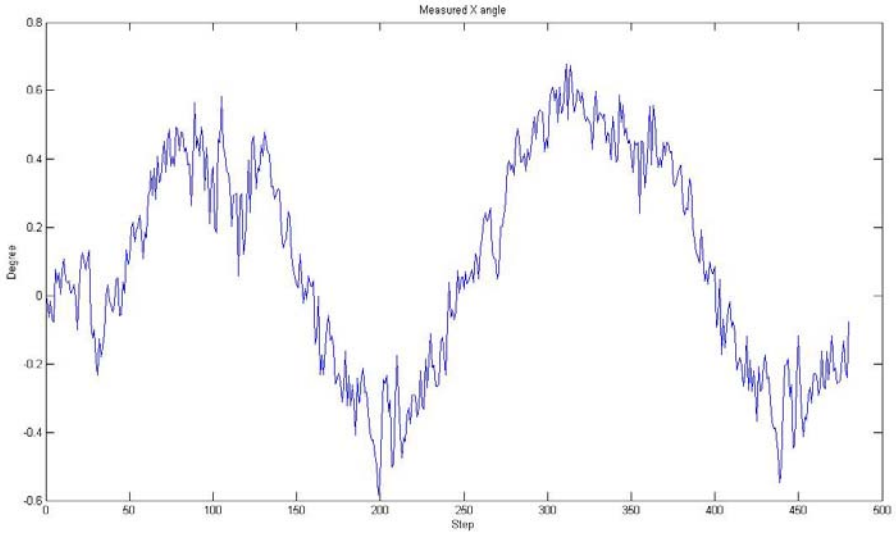

Figure 5-33 Estimated X Angle (Experiment 3)

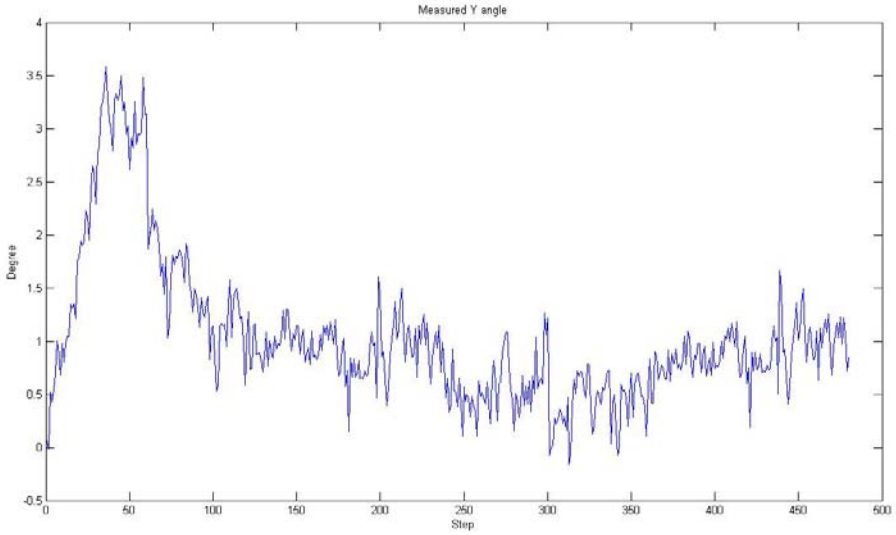

Figure 5-34 Estimated Y Angle (Experiment 3)

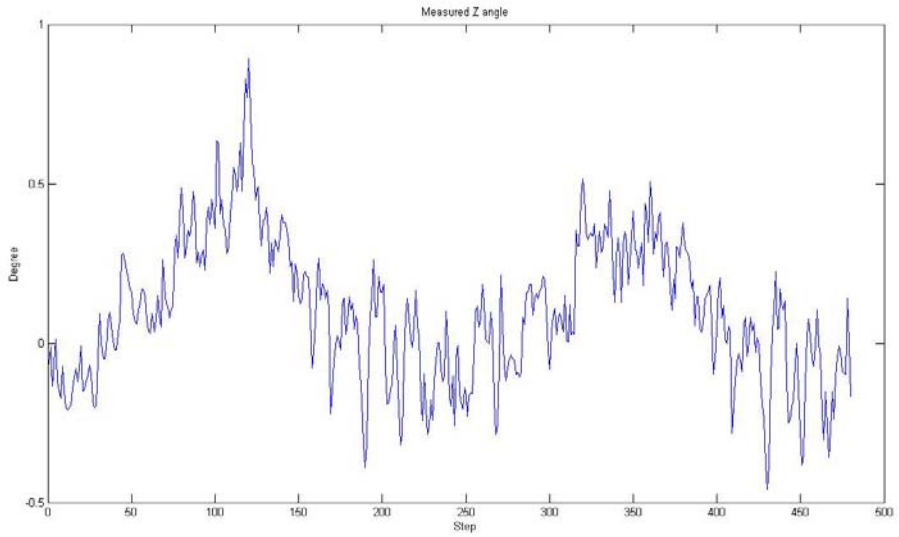

Figure 5-35 Estimated Z Angle (Experiment 3)

In figures 5-36 to 5-38 X, Y, Z covariance values of location are given. We expect mean errors for all directions to be in the limits of 2 sigmas. From table 5-9 it is confirmed. Additionally, by observing the same features, covariance values are decreased in second loop. So covariance values give opinion about real ground truth and this can be estimated within the range of 2 sigma value around the estimated value.

Table 5-9 Comparison of Covariance Value and Mean Error (Experiment 3)

Direction	1:399 mean (cm)		1:200 mean (cm)		201:399 mean (cm)	
	2*sigma	error	2*sigma	error	2*sigma	error
X	11.2	3.9	13.1	5.3	9.4	2.4
Y	9.8	1.0	9.8	1.1	9.1	1.0
Z	6.1	2.2	6.8	2.1	5.4	2.2

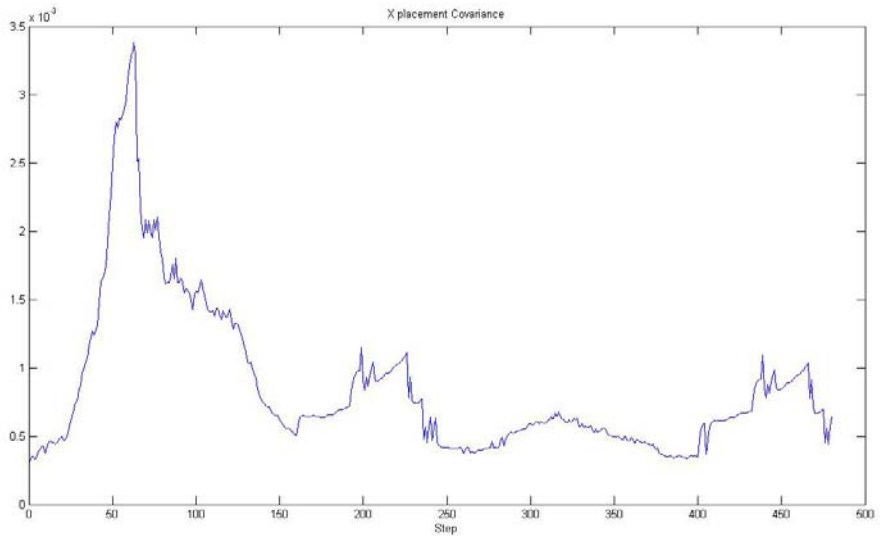

Figure 5-36 X Localization Covariance (Experiment 3)

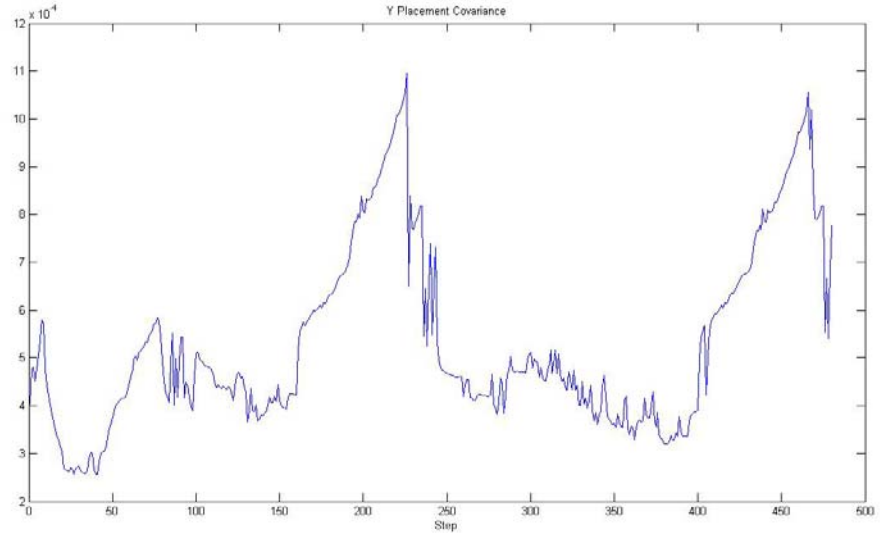

Figure 5-37 Y Localization Covariance (Experiment 3)

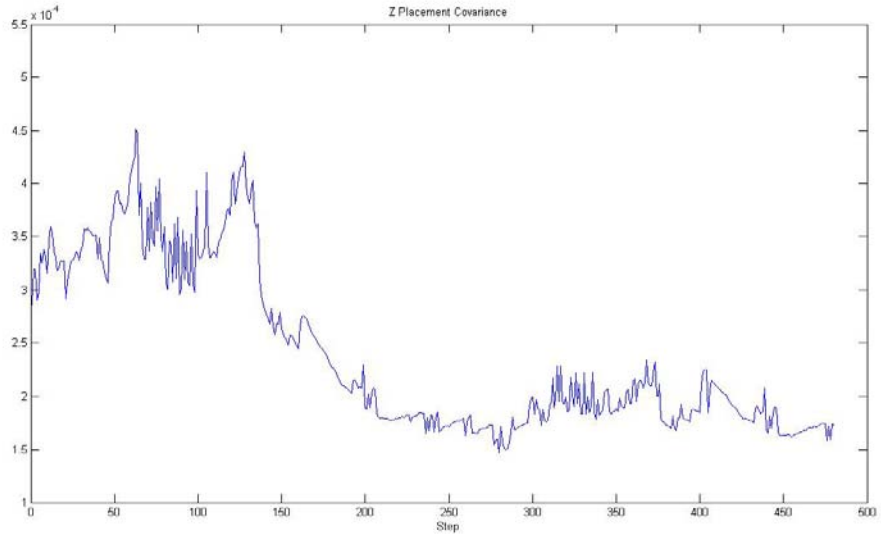

Figure 5-38 Z Localization Covariance (Experiment 3)

In this part, the search area and depth covariance of features will be investigated. From equation 3-57, there are four criteria that affect feature's search area and depth covariance. These are as follows: Covariance value of the camera system which indicates the quality of the location and angle estimation of the camera, covariance of the features which shows the quality of the estimation of feature positions, observation number of features; if a feature is matched between frames its covariance will decrease or vice versa, and finally noise in covariance matrix. This value is constant and not changed through the process. As a result, remaining values will affect the search area and covariance of the feature depth estimation. It is expected that, if a feature is observed in preceding frames its covariance value will decrease and because of the movement of camera and added noise, search area remains same. To give an example, depth variance of feature 1 and feature 2 are given in figure 5-39 and 5-40. From the graphs we observe that search area of features remains near 10 and covariance value of inverse depth is decreased by the time. In other words Stereo SLAM Algorithm calculates the depth of feature 1 at start between 1/ (0.367+0.05) = 2.39m and 1/ (0.367-0.05) = 3.15m with a 3.15m-2.39m=0.76m variance and by the observation of this feature this value reduces to (1/ (0.3282-0.005) = 3.0941m 1/ (0.3282+0.005) = 3.0012m) 9cm.

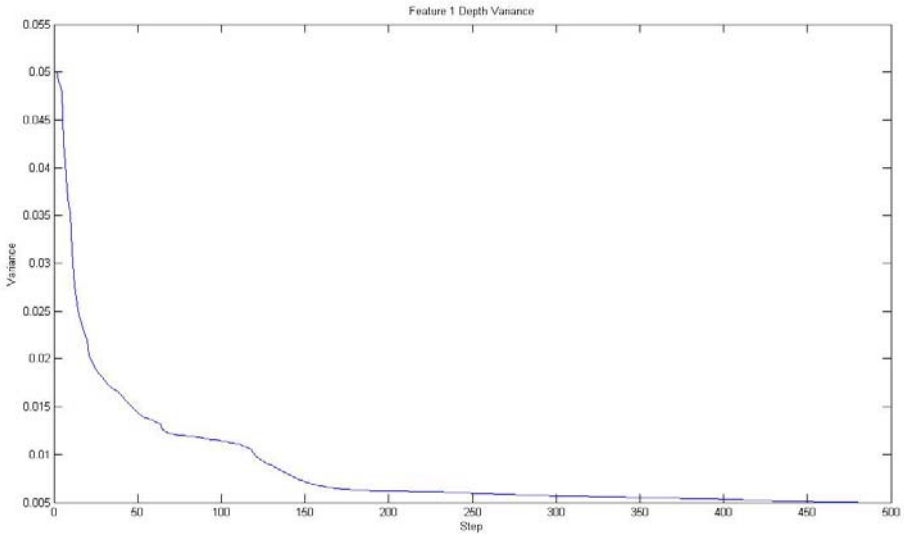

Figure 5-39 Feature 1 Depth Variance (Experiment 3)

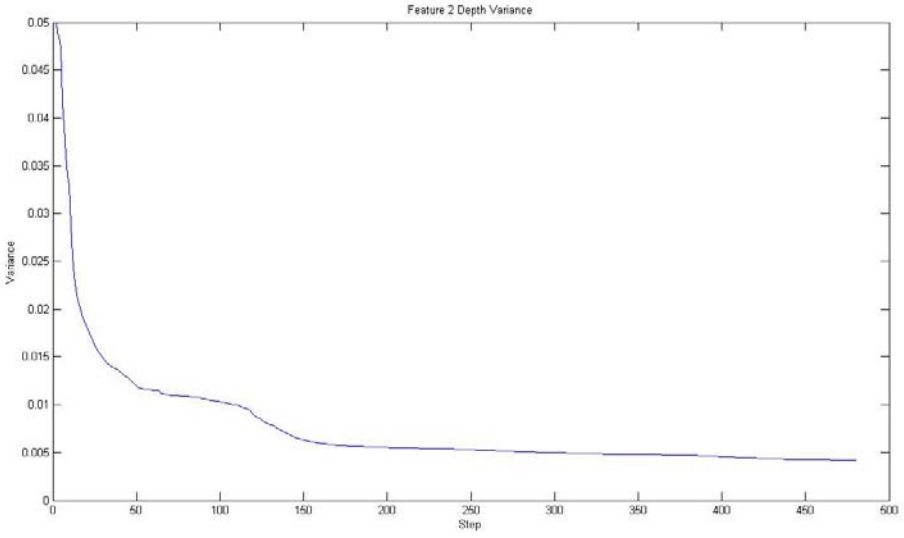

Figure 5-40 Feature 2 Depth Variance (Experiment 3)

If patch of a feature cannot be matched between preceding frames then its covariance will increase. For example feature 25 can not be matched. As a result, its

search area is increased and covariance of depth is not improved by estimations. Then because of the decrease in location covariance of the camera, search area is decreased but it is higher than search area of other features. Finally Stereo SLAM Algorithm starts to match it in preceding frames and its depth covariance starts to decrease. In figure 5-41 search area and variance of depth is given. Since variance is a small value with respect to search area, search area is divided to 1000 for easy follow. In figure 5-42 search area of figure 25 at step 197 is given. From the graph it is observed that, search area of feature 25 is bigger than other features. After matching of this feature at step 319, its search size is decreased and it has same value as other features and this is illustrated in figure 5-43.

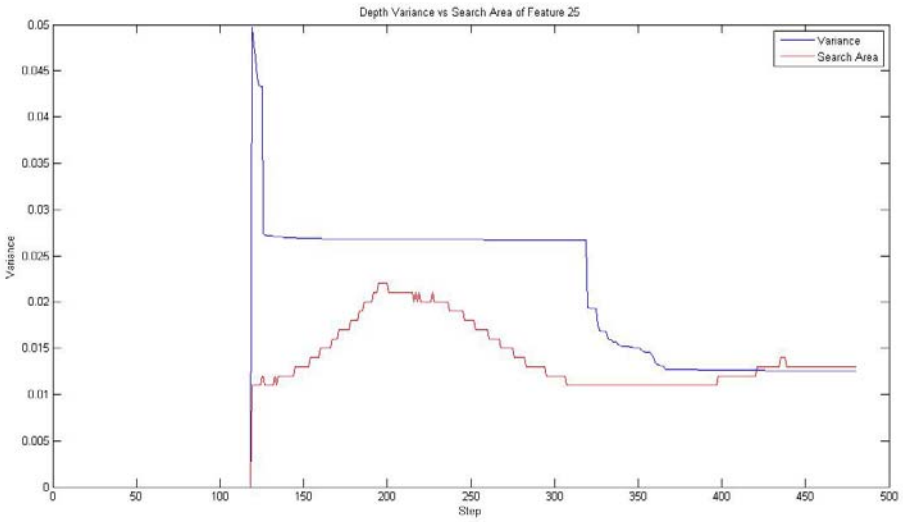

Figure 5-41 Depth Variance vs Search Area (Experiment 3)

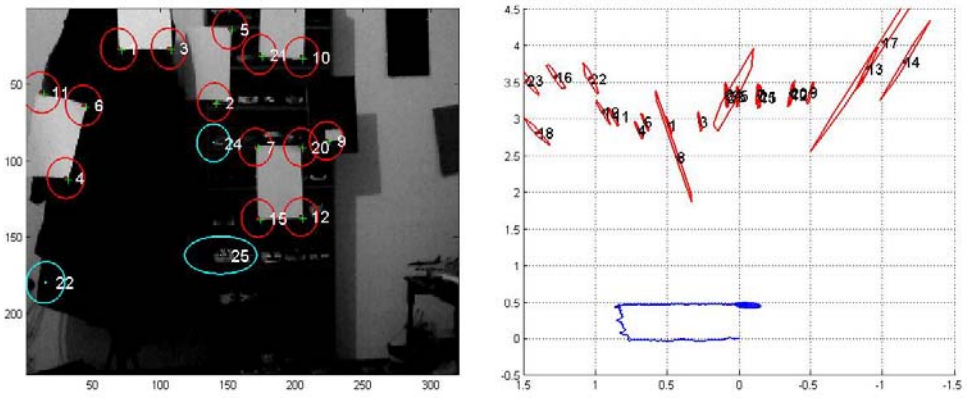

Figure 5-42 Step 197 Feature 25 Search Size (Experiment 3)

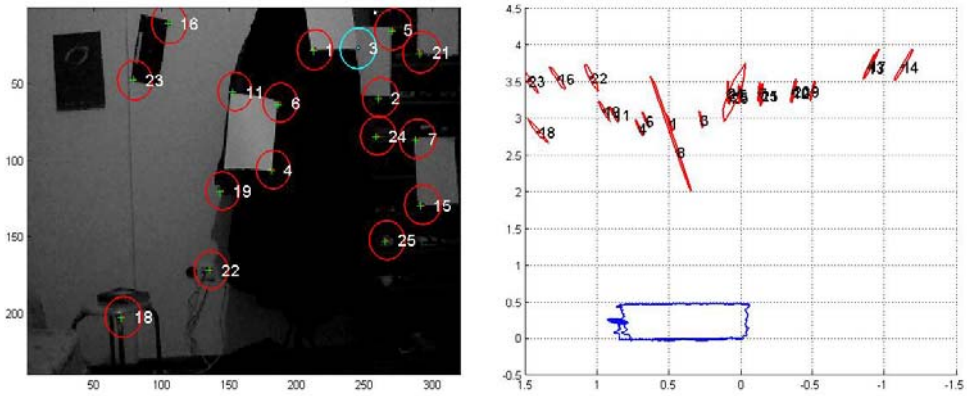

Figure 5-43 Step 319 Feature 25 Search Size (Experiment 3)

From the results we observe that Stereo SLAM Algorithm gives close to real values within same error and this error is within 2 sigma error range. Moreover even camera moves in a different path Stereo SLAM Algorithm closes the loop within a small error at 240 and 480 steps.

5.5 Angle Calculation Experiment (Experiment 4)

In this experiment angular changes in cameras pose are investigated. Frames are acquired on an arc. Angle of the arc is 50 degrees and radius of the arc is 81cm. Ground truth and estimations are given in figures. Also in this experiment camera moves on the arc 2 times to see the effects of re-observing same scenes. Additionally angular velocities are analyzed. In this experiment a grid system is used to change the angle with respect to the initial position. The grid system is shown in figure 5-44. Movement of the camera is given as follows: camera moves to middle of the table within 75 steps. Then with 74 steps camera finishes the arc. Camera turns back to its initial position.These are marked 1, 2 and 3 in figure 5-44 respectively. Totally there are 2 arcs with 298 steps. A closer look to grid system is given in figure 5-45. For every step, camera is placed on the line belongs to step written on grids. As in other experiments, estimated value and real values are given. First arc output of the Stereo SLAM algorithm is given at figure 5-46 and second arc output with first arc is given at figure 5-47.

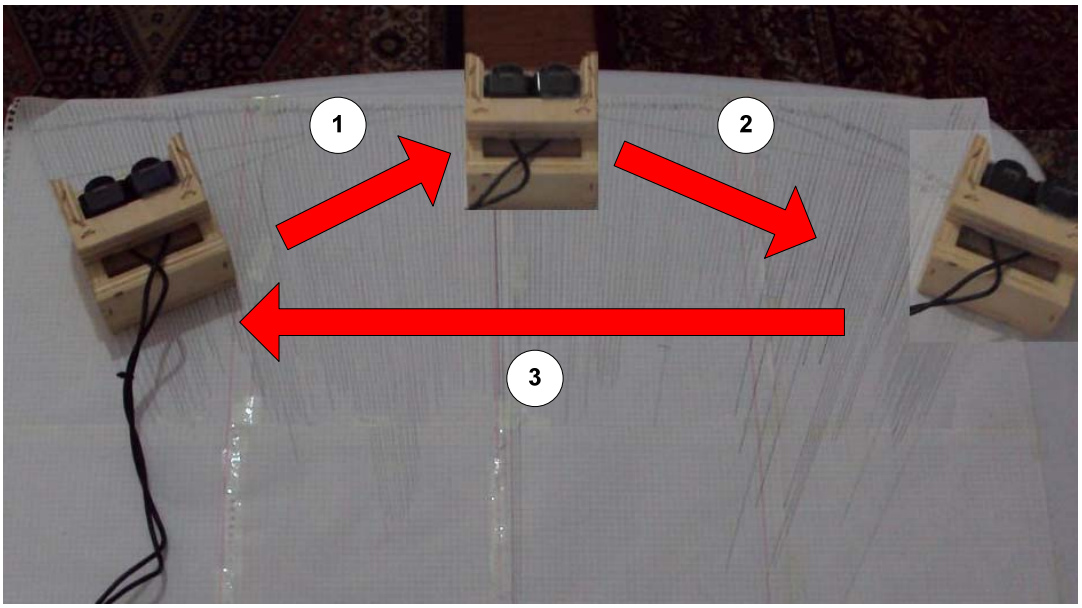

Figure 5-44 Grid System (Experiment 4)

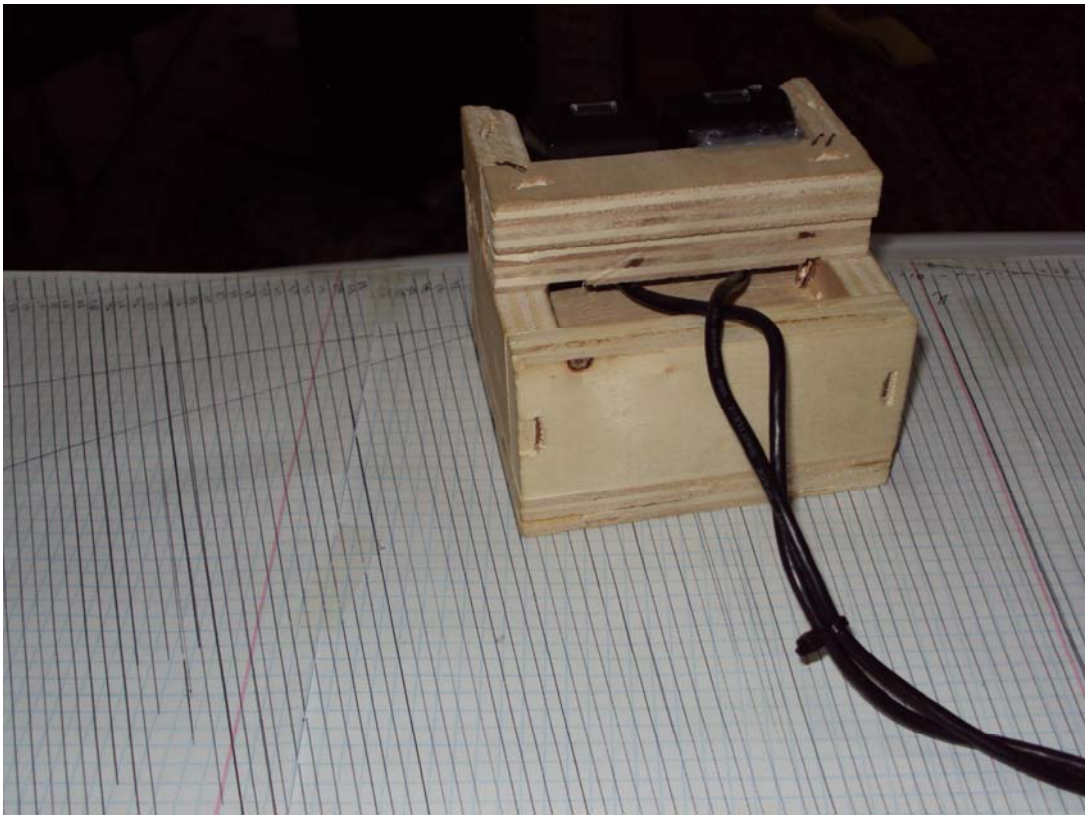

Figure 5-45 Detailed Picture of the Grid System (Experiment 4)

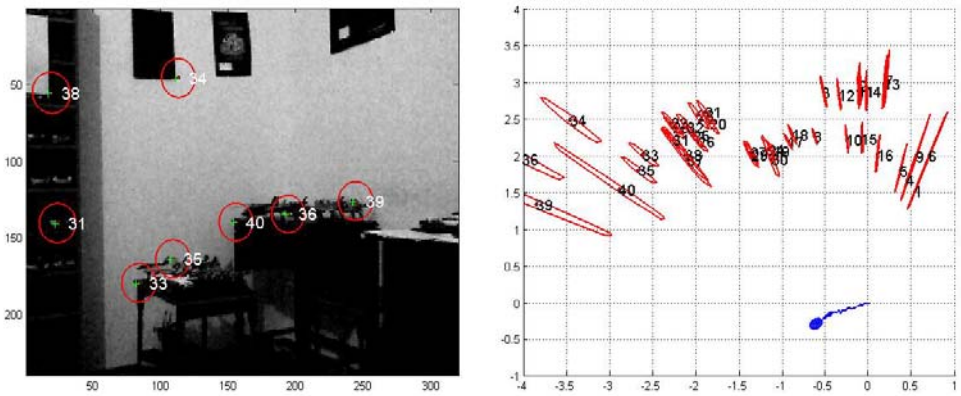

Figure 5-46 First Arc (Experiment 4)

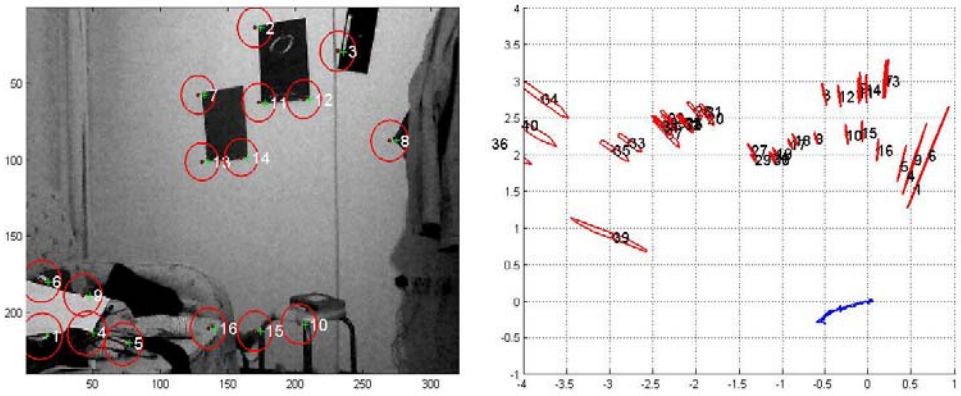

Figure 5-47 Second Arc (Experiment 4)

Firstly the location errors are analyzed. In figure 5-48 to 5-50 the estimated values and ground truth are given. In this experiment ground truth is changed on X and Z direction simultaneously and at Y direction no movement occurs. Maximum errors and mean errors on all directions are given at table 5-10. From results we see that Stereo SLAM Algorithm estimates the real value within a few cm errors with respect to ground truth.

Table 5-10 Errors in Localization (Experiment 4)

Direction	Max Error (step) (cm)	Mean Error (cm)	Mean Error at first arc (cm)	Mean Error at second arc (cm)
X	150. / 10.8	1.9	1.8	2.0
Y	150. /9.4	2.2	2.2	2.2
Z	178. /8.2	2.0	1.9	2.1

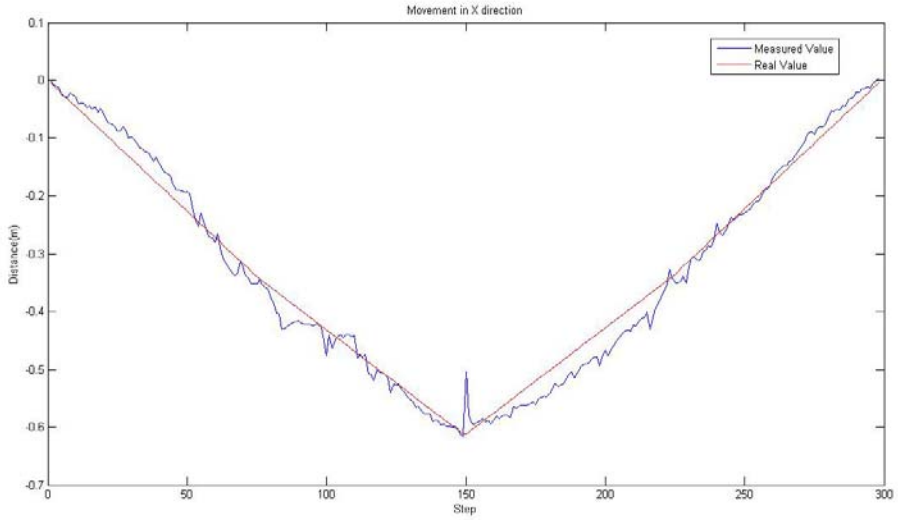

Figure 5-48 Movement in X direction (Experiment 4)

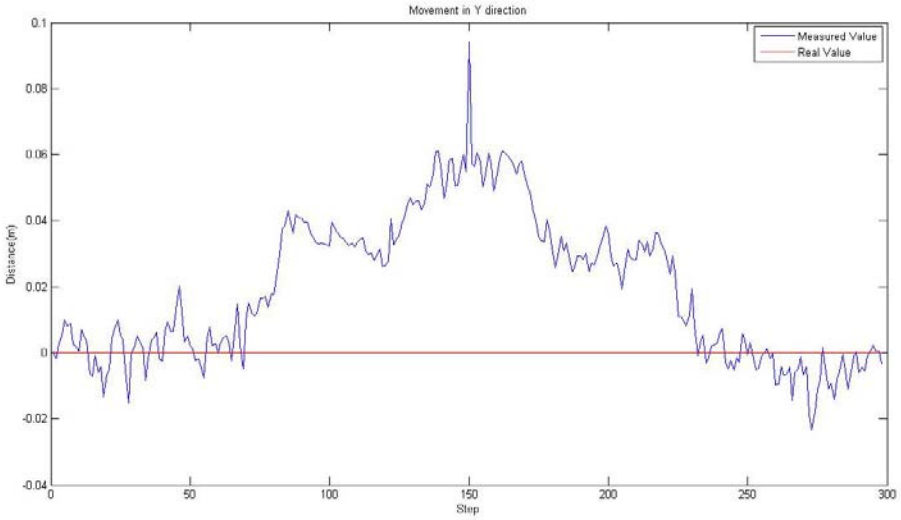

Figure 5-49 Movement in Y direction (Experiment 4)

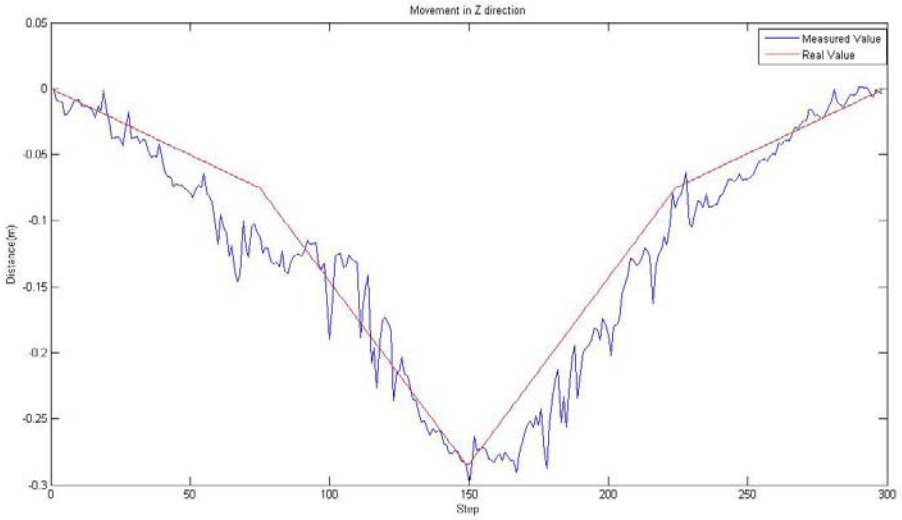

Figure 5-50 Movement in Z direction (Experiment 4)

Angular changes on X, Y, Z directions are given in figures 5-51 to 5-53 respectively. The main of aim of this experiment is to show that Stereo SLAM algorithm estimates angular changes within small errors with respect to the real values. In table 5-11 maximum error and mean errors are given for each angle. From results it is easily observed that stereo SLAM algorithm estimations are close to real values. Since errors are small with respect to the other experiments we don't observe a significant decrease in mean errors at second arc.

Table 5-11 Errors in Angle (Experiment 4)

Direction	Max Error (step) (deg)	Mean Error (deg)	Mean Error at first arc (deg)	Mean Error at second arc (deg)
X	121. / 3.79	1.93	2.00	1.87
Y	151. /-2.78	0.52	0.52	0.52
Z	121. /1.21	0.37	0.39	0.36

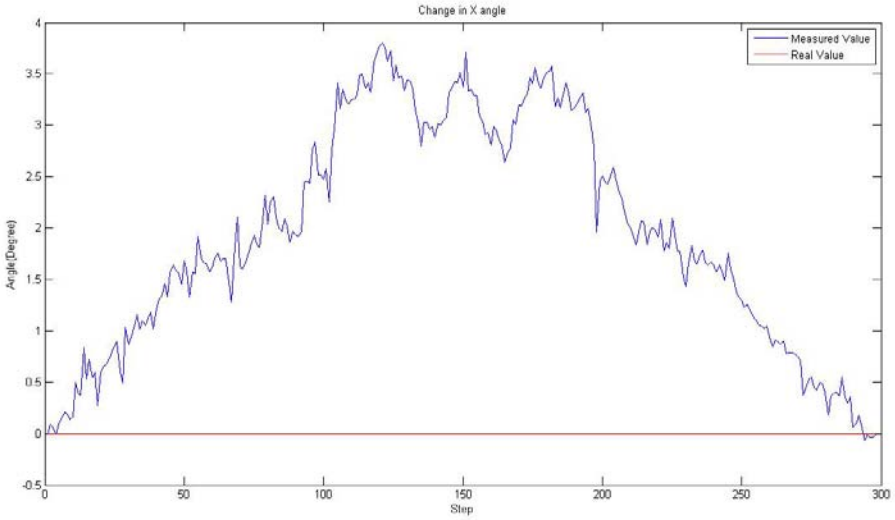

Figure 5-51 Estimated X Angle (Experiment 4)

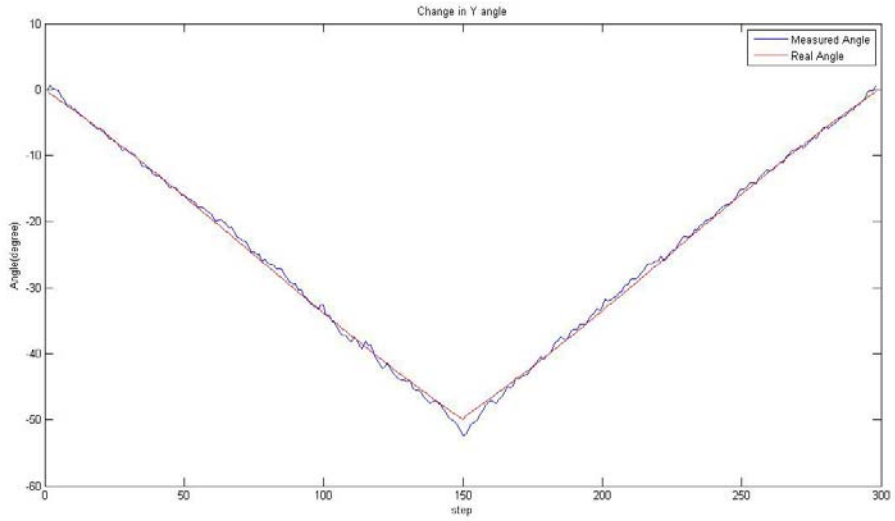

Figure 5-52 Estimated Y Angle (Experiment 4)

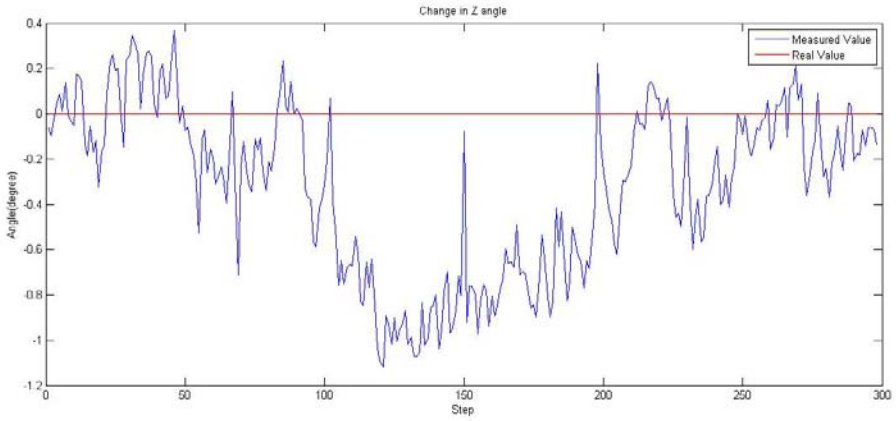

Figure 5-53 Estimated Z Angle (Experiment 4)

In figure 5-54 real and estimated values are given. From the graph it is observed that estimated value oscillates around the real value. Maximum error and mean error of Y angular velocity are given at table 5-11.

Table 5-12 Error in Y Angle Velocity (Experiment 4)

Direction	Max Error (step) (rad/s)	Mean Error (rad/s)	Mean Error at first arc (rad/s)	Mean Error at second arc (rad/s)
Y	132. /0.44	0.13	0.1361	0.1239

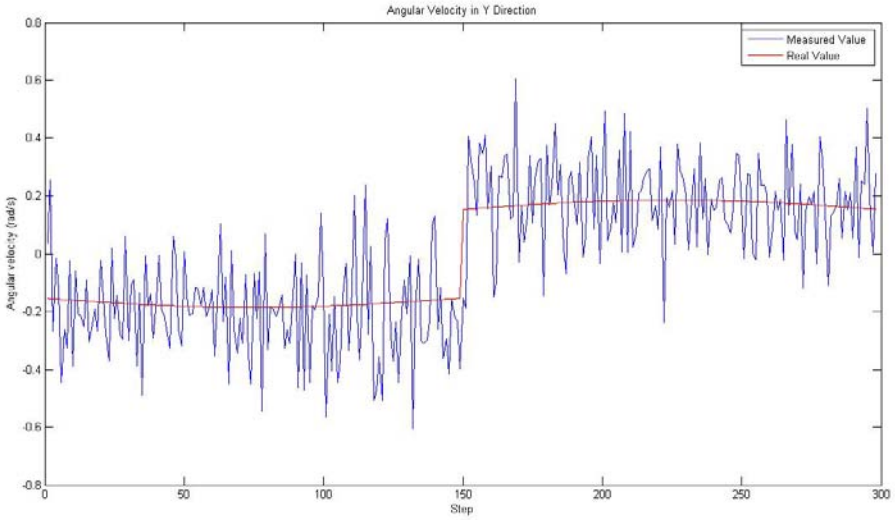

Figure 5-54 Estimated and Real Y Angular Velocity (Experiment 4)

Covariance values of X, Y, Z location are given at figure 5-55, 5-56, 5-57 respectively. We expect mean errors for location to be lower than 2 sigma value. At table 5-13 2 sigmas and mean errors are given. From results it is seen that covariance gives opinion about the ground truth and ground truth stays in the limits of 2 sigmas.

Table 5-13 Comparison of Covariance Value and Mean Error (Experiment 4)

Direction	Total (cm)		First Arc (cm)		Second Arc (cm)	
	2*sigma	error	2*sigma	error	2*sigma	error
X	4.34	1.9	4.67	1.8	4.02	2.0
Y	3.52	2.2	3.52	2.2	3.52	2.2
Z	3.65	2.0	3.97	1.9	3.34	2.1

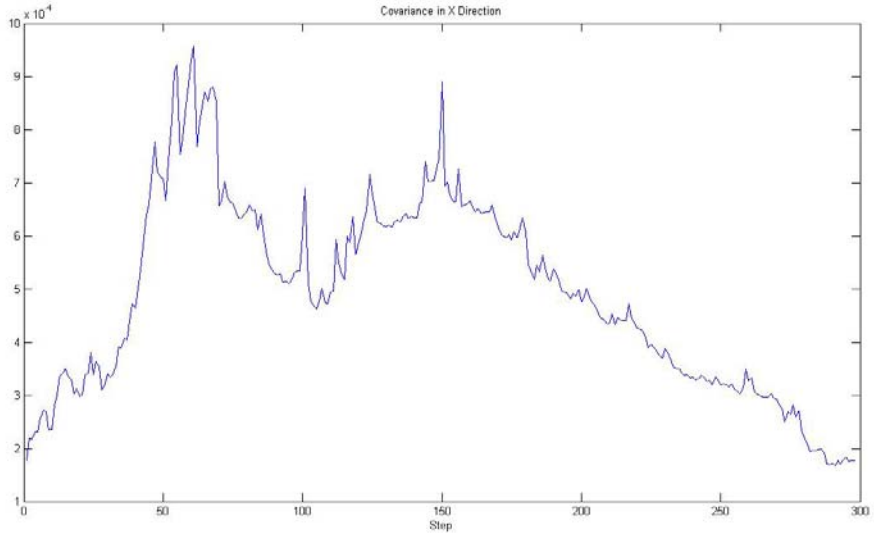

Figure 5-55 X Location Covariance (Experiment 4)

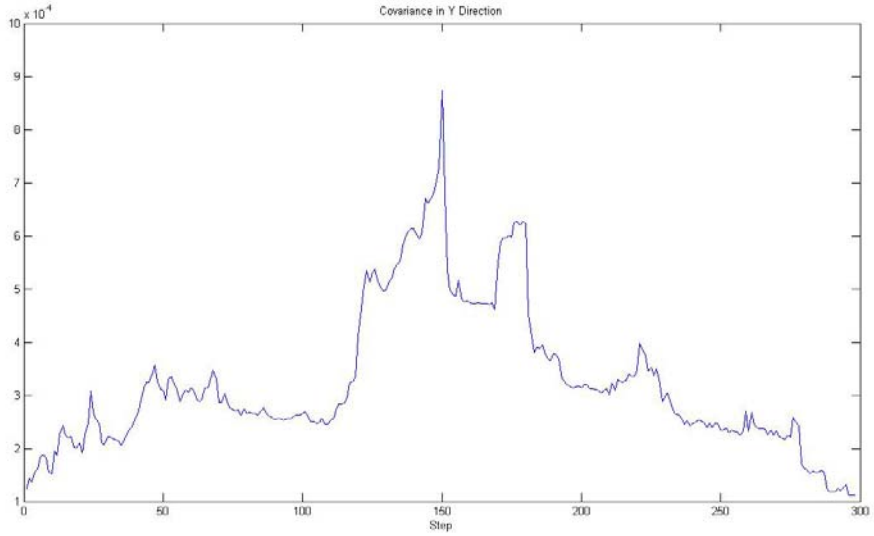

Figure 5-56 Y Location Covariance (Experiment 4)

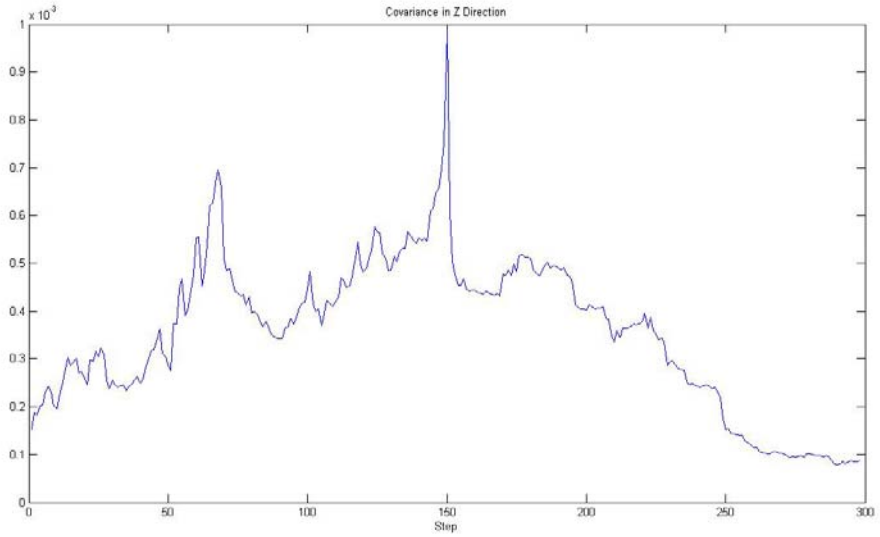

Figure 5-57 Z Location Covariance (Experiment 4)

From the results, even there is a change in angular direction Stereo Slam Algorithm estimates camera state near to ground truth with error which can be observed from the covariance values.

5.6 Random Motion Experiment (Experiment 5)

In this experiment the main is to show that if camera builds a loop Stereo SLAM Algorithm measures a closed loop output and no drift should be observed even in a random movement of the system. Several videos have recorded and Stereo SLAM Algorithm gives good results and estimates a closed loop for all of them. In this experiment one output of these videos is given. In this video movement in all directions and also angle changes occur. Since we have no ground truth, main aim is to observe driftless closed loop output of the algorithm. In observed video, camera system makes a 180 step movement and turns back to its initial position with the same frames. So the first loop consists of 359 steps and totally two loops with 717 steps. In this experiment we only investigate placement and angle output of the system. In figure 5-58 and 5-59 output of the system at 359th and 717th step is given.

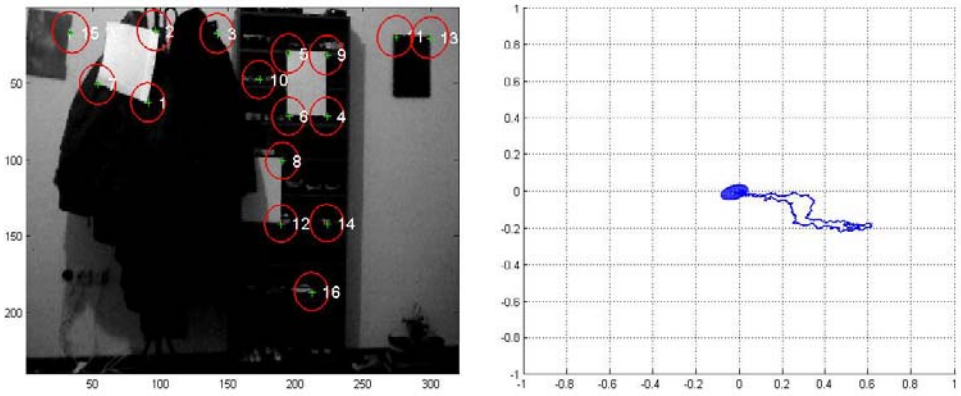

Figure 5-58 First Loop (Experiment 5)

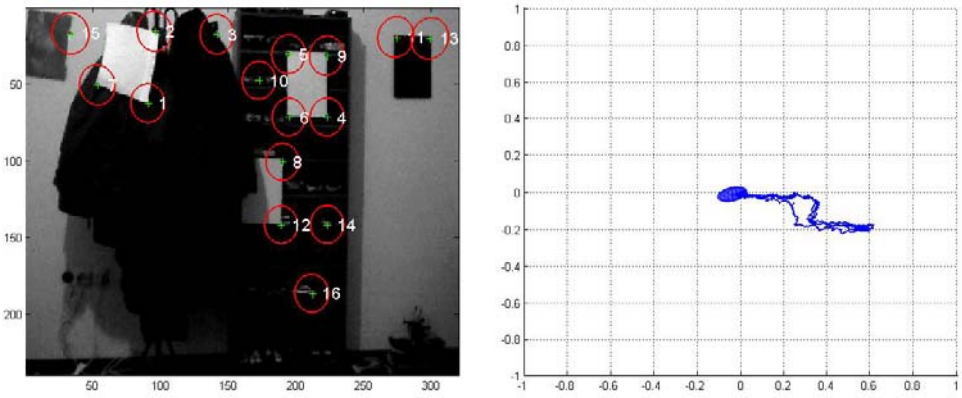

Figure 5-59 Second Loop (Experiment 5)

In figure 5-60, 5-61, 5-62 placements in X, Y and Z direction are given. We expect loop closure in 359th and 717th Steps. Also we should observe symmetry in the figures because we expect similar measurements from Stereo SLAM Algorithm. From results these properties can be observed. Also in first movement in steps 1 to 180 since feature position estimations are worse than future calculations, in preceding movement we observe better placement calculations. We observe it from the fact that in last three location calculations are similar to each other. Also in table 5.14 placement outputs and angle outputs are given to show loop closure within few cm.

Table 5-14 Output of Stereo SLAM Algorithm (Experiment 6)

Direction	First Loop Displacement(cm)	Second Loop Displacement(cm)	First Loop Angle(deg)	Second Loop Angle(deg)
X	-0,0458	4,0174	-0,3749	0,0008
Y	0,0771	1,0512	0,6186	0,7328
Z	-1,6107	1,3785	0,1105	-0,2128

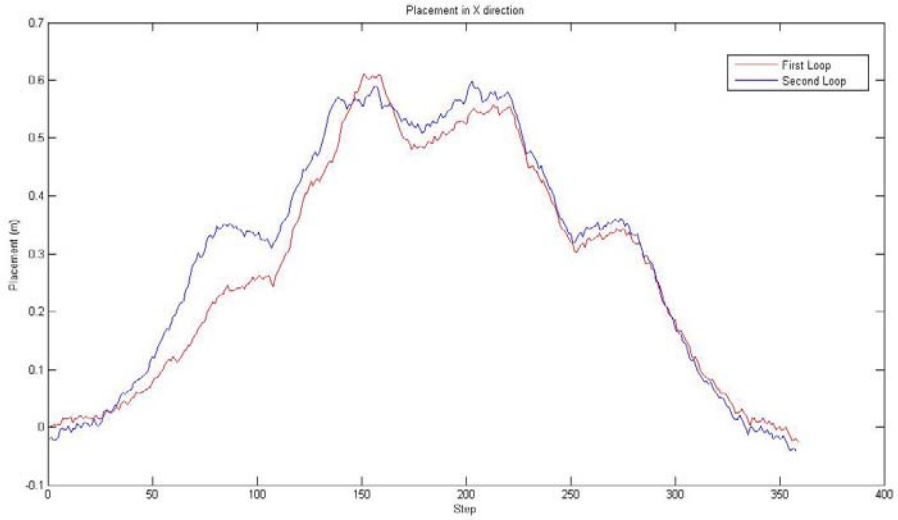

Figure 5-60 Movement on X Direction (Experiment 5)

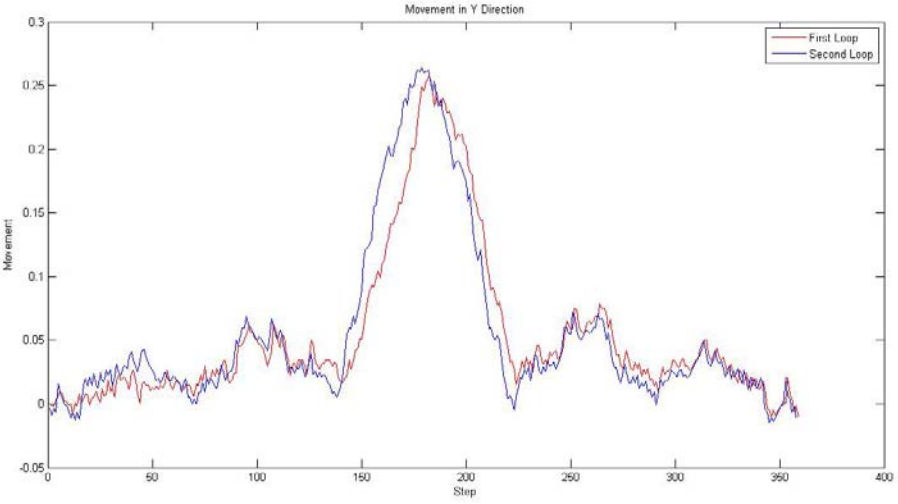

Figure 5-61 Movement on Y Direction (Experiment 5)

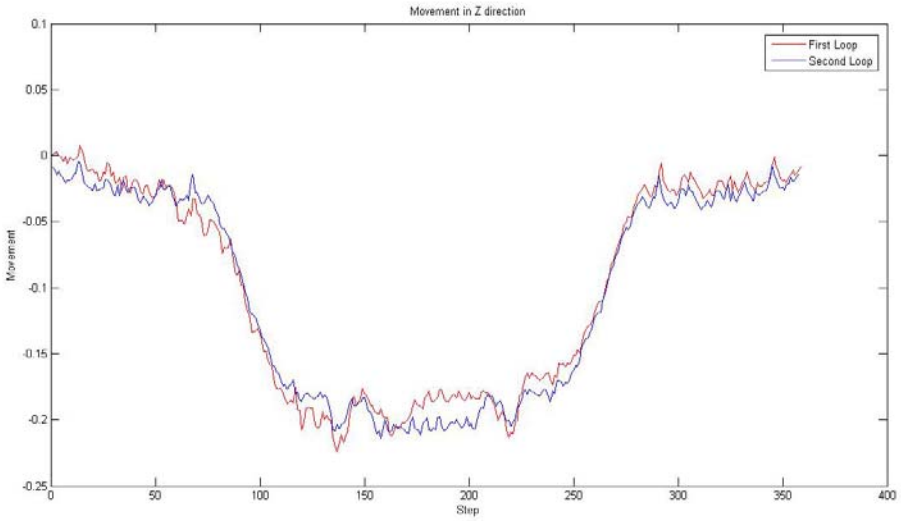

Figure 5-62 Movement on Z Direction (Experiment 5)

CHAPTER 6

SUMMARY, CONCLUSION AND FUTURE WORK

6.1 Summary and Conclusion

In this thesis, a consistent Simultaneous Localization and Mapping System is constructed by using Extended Kalman Filter. The Experiment results show that Stereo SLAM Algorithm estimates the camera location and landmark locations within small error ranges. Wrong distance estimations in the depth of the landmark locations increase the error in the estimation of the camera pose. This effect is reduced as much as possible since 15 features are observed. In order to handle error in distance estimations, some amount of error margin is assigned. If this margin is increased, error in camera pose estimation increases. If margin value is assigned to a higher value, more movement is estimated than actual movement. If this margin is decreased, correction on location of features cannot be handled by Stereo SLAM Algorithm because the system perceives location of the features precisely. Because of these facts, an optimum value should be selected. As calculated in experiment 1, the error in depth is approximately %5 of the actual values. This error rate is used for depth error margin. If this error is assigned as 0.5 and the initial depth estimation is assigned to 0.1 as in [2], much more movement is estimated than actual value and Stereo SLAM Algorithm cannot localize the camera system. To summarize, Stereo SLAM Algorithm decreases the error in estimation of localization of camera and features up to some point.

A feature search algorithm is implemented. In the first frame all pixel points are searched and up to 15 features are initialized. By the movement of the camera, some features may not be visible and feature number observed by the cameras decreases below 15. At this point new features are searched and added to the map. Additionally if there is a long time mismatch for a feature, this feature is deleted from state vector and covariance matrix.

Two modes of operation are defined. In online mode, Stereo SLAM Algorithm simultaneously takes frames, processes them and updates the state vector and covariance matrix. Offline simulation mode is built because the performance of the Stereo SLAM system should be analyzed. To operate the simulation mode, the frames are recorded and experiments are conducted with them.

By the time as the online mode development is improved, the process time for a frame is decreased to 10 frames in a second. This increases the matching possibility of features between frames because movement of the camera in the time interval between the frames is decreased and more steady measurements are taken. Since we use Matlab which is a slow program compared to other languages like C and C++, Stereo SLAM Algorithm runs up to 10 frames per second. But in my opinion if we implement Stereo SLAM Algorithm in C/C++ code (especially in OpenCV), the system can run with the frame rate of the cameras.

Depths of the features are estimated. In order to measure these depths, triangulation is used. In order to use triangulation, the intrinsic parameters of a camera like focal length, lens distortion etc. and extrinsic parameters of a camera which are related with the placement of right camera with respect to left camera have to be known. To know these parameters, camera calibration procedure is executed. These parameters are also necessary for finding the azimuth and elevation of the features with respect to the camera system. Error in depth estimation is affected by errors in calibration procedure. A camera with small error rates in the results of calibration parameters yield better depth estimations. Moreover, if a higher resolution option of cameras

like 640x480 is used, error in depth estimation will decrease and better camera pose location can be estimated with SLAM algorithm.

A graphical user interface is implemented which starts the algorithm, shows the last processed frame, feature locations and camera location with respect to initial point.

To match features in SLAM algorithm in the received frames, %95 of the confidence area (2 standard deviations) are searched. This procedure generally gives good results but since matching is not rotational invariant (because point features are not rotational invariant), features start to not match between preceding frames. Moreover when a feature is observed with different angles, there is a possibility for mismatch. The reason can be explained as follows: The patches of the features are saved and not updated by time. This may be solved by updating the image patches after each successful matching. However this may lead to drift in image patches. The image patch will be changed by movement, but in the end it may even not match with the first observed patch. Also sometimes wrong matches may occur. At these times, patches will be updated with wrong pixels. As a result, updating patches are dangerous and not implemented in this thesis.

Sometimes two patches in the search area of a feature have the highest score for matching process and Stereo SLAM Algorithm selects one patch for the current step and the other patch for the next step. This fact causes error in the estimation of locations. Even there is no movement in the camera system, Stereo SLAM Algorithm estimates some movement. To handle this problem, if there is no movement in the remaining of the features, the feature which oscillates is matched with one of the patches and is not updated. Also if oscillation is observed, effect of the feature on the state vector and the covariance matrix is reduced to zero.

In this thesis, low cost cameras are used. Low cost cameras can be obtained from any store with a low price starting from 10 $. Instead of using them, if IEEE 1394 camera is used the cost of the system will be minimum 200 $. Also since low cost

cameras produce frames in digital format there is no need for a device like an image grabber. Besides, cameras can be easily connected to any Windows platform with USB connection and also there is no need for extra drivers to use the cameras because webcam drivers of the Windows operating system are sufficient to work with them. Moreover these cameras are light weight and they can be easily moved by one hand. Their weight is only 90 grams. However these cameras also have negative sides. Their performance in low illumination significantly decreases and initialization of features becomes a hard task for Stereo SLAM Algorithm. Moreover low cost cameras have a narrow view angle (50 degrees in used camera) which affects the distribution of features.

By using stereo image pairs Stereo SLAM Algorithm estimates the depth of the features with respect to the camera system without any need for extra device or procedure. This is the most important benefit of using stereo image pairs. Also movement of cameras can be estimated by using stereo cameras but this property is not used in this thesis. Stereo Camera System also has negative sides. By using stereo image pairs two frames from different cameras have to be acquired. This means twice cost of process time with respect to a single frame. Additionally these frames should be acquired at the same time. As a result, by using mono camera in SLAM process the speed of the system is increased. By using features in the environment Stereo SLAM Algorithm estimates the state vector and covariance matrix. If the depths of the features are not known, using mono camera is useless because by using only one camera, depth of the features cannot be estimated. As a result in order to calculate the depth of features, stereo image pair is absolutely needed.

The core steps of the Stereo SLAM Algorithm consist of Kalman Filter steps. When state vector and covariance matrix are updated in Kalman Filter, noises are added to the covariance matrix since movements and estimations are noisy. These noises are given as follows: Firstly camera calibration parameters can not be calculated exactly by camera calibration procedure. So that errors in depth estimations are

inevitable. Also since sub-pixel calculations are not handled, the uncertainty in estimations increases. Additionally, stereo image pairs can only estimate depth up to a certain point and above this limit, error in depth estimations starts to increase. However, since camera system is used in a room environment, this fact is negligible. Also radial and tangential distortions of cameras affect the estimation of the state vector and covariance matrix. Since the camera is moved by hand and there is no commanded tasks for Stereo SLAM Algorithm, movement of cameras cannot be predictable and movement might be very noisy. As a result, output of the Stereo SLAM Algorithm has noisy estimations.

6.2 Future Work

These are some future works to extend the thesis.

To increase the frame rate processed by the system in one second, all code should be written and executed in C/C++ language. In this thesis approximately %70 of the code is written in C. Since some major tasks like acquisition and conversion of the images which are the most time consuming parts of the system are handled by Matlab and since it is a slow program with respect to the C, C++ etc. languages this affects the frame processed by SLAM system. In order to handle this major effect all code can be written in C/C++ with image processing code packs like OpenCV.

To increase the measurement accuracy of the depth and angle estimation of the features, 640x480 resolution option can be used instead of 320x240 resolution. Its side effect will be the increase in computation time but it will significantly increase the accuracy of the estimations. Moreover to increase the estimation accuracy, stereo depth calculations can be added to the estimations. Also two estimations can be applied by the system and combined together by using two cameras. Also two estimations and stereo estimations can be combined together. These processes are explained in detail in section 4.10.

In this thesis when cameras are operated, only the person on the computer can see the location of the camera and the frame processed. Instead of this, another remote expert should see the output of the camera system. In other words, state vector, feature positions and frames processed should be sent to another computer for distant viewing purposes.

Instead of using only point features, another type of features like SIFT [27] can be used in order to increase the stability of the system. Moreover a laser sensor, an accelerometer or gyro can be used to increase the stability of the system. It also increases the stability on Z angle changes.

REFERENCES

[1] Harris, C. and Stephens, M. A Combined Corner and Edge Detection. In Proceedings of The Fourth Alvey Vision Conference, pages 147–151. 1988.

[2]: Andrew J. Davison, Ian D. Reid, Nicholas D. Molton, O. Stasse "Mono SLAM: Real-Time Single Camera SLAM". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 29, No.6, June 2007

[3]: Andrew J. Davison, David W. Murray "Simultaneous Localization and Map Building Using Active Vision". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.24, No.7, July 2002

[4]: http://www.vision.caltech.edu/bouguetj/calib_doc/ last visited 29/08/2010

[5]: Adnan Kalay "An Implementation of Mono and Stereo SLAM System Utilizing Efficient Map Management Strategy" *Master Thesis*, Middle East Technical University, September 2008

[6]: S. Thrun, W. Burgard, D. Fox "Probabilistic Robotics". *Book*, The MIT Press, September 2005

[7]: J. Shi, C. Tomasi "Good Features to Track". *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR '94*, Pages 593-600, Seattle, WA, USA, 21-23 Jun 1994

[8]: Sven Albrecht "An Analysis of VISUAL MONO-SLAM" *Master Thesis*, Osnabruck University, October 2009

[9]: Hamid Aghajan, Andrea Cavallaro "Multi Camera Networks Principles and Applications". *Book*, Academic Press, 2009

[10] http://www.csse.uwa.edu.au/~pk/Research/MatlabFns/ last visited 29/08/2010

[11] L. Clemente, Andrew Davison, Ian Reid , J. Neira , J Tardos "Mapping Large Loops with a Single Hand Held Camera" *Robotics: Science and Systems* June, 2007.

[12] Andrew J. Davison, Waltero W. Mayol and David W.Murray "Real Time Localization and Mapping with Wearable Active Vision" *Prov IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, Tokyo, Japan, October 7-10, 2003

[13] L.M.Paz, P. Pinies, J.D, Tardos, J Neira "6 DOF SLAM with Stereo-in-hand" *International Conference on Intelligent Robots and Systems* 2007

[14] T. Lemaire, C. Berger, I. Jung and S. Lacroix "Vision Based SLAM: Stereo and Monocular Approaches"in International *Journal of Computer Vision* Page 343-364, 2007

[15] J.M.M. Montiel, J. Civera, Andrew J. Davison "Unified Inverse Depth Parametrization for Monocular SLAM" in *Proceedings of Robotics: Science and Systems,* Philidelphia, USA, August 2006

[16] P. Gemeiner, A. Davison, M. Vincze "Improving Localization Robustness in Monocular SLAM Using a High-Speed Camera" *Proceedings of Robotics: Science and Systems IV* Zurich, Switzerland, June 2008

[17] J. Solà, A. Monin, M. Devy "BiCamSLAM: Two times mono is more than stereo" *IEEE International Conference on Robotics and Automation, ICRA 2007*, 10-14 April 2007, Roma, Italy 2007

[18] R. O. Castle, D. J. Gawley, G. Klein, and D. W. Murray "Video-rate recognition and localization for wearable cameras" *Proc 18th British Machine Vision Conference*, Warwick, Sept 2007

[19] T.E.de Campos, W.W. Mayol-Cuevas and D.W. Murray "Directing the Attention of a Wearable Camera by Pointing Gestures" *Proc IEEE Brazilian*

Symposium on Computer Graphics and Image Processing (SIBGRAPI), October 2006

[20] Andrew J. Davison "Models and State Representation in Scene: Generalized Software for Real-Time SLAM" May 12, 2006

[21] P.H.S Torr "A Structure and Motion Toolkit in Matlab" Technical Report June 2002

[22] E. Eade, T. Drummond "Scalable Monocular SLAM" *IEEE Computer Converence on Computer Vision and Pattern Recognition* New York USA June 2006

[23] J. M. M. Montiel, A. Davison "A Visual Compass Based on SLAM" *In International Conference on Robotics and Automation*, 2006

[24] O. Stasse, A. Davison, R. Sellaouti, K. Yokoi "Real-time 3d SLAM for a humanoid robot considering pattern generator information" In *IEEE International Conference on Intelligent Robots and Systems*, 2006

[25] http://en.wikipedia.org/wiki/Conversion_between_quaternions_and_Euler_angles last visited 08/09/2010

[26] C. Han, Z. Xiang, J. Liu, Eryong Wu "Stereo Vision Based SLAM in Outdoor Environments" In IEEE *International Conference on Robotics and Biomimetics*, 2007

[27]Brown, M. and Lowe, D. "Invariant Features from Interest Point Groups". In *In British Machine Vision Conference*, pages 656–665. 2002.

[28] Civera, J., Davison, A.J. andMontiel, J.M.M. Inverse Depth Parametrization forMonocular SLAM. *IEEE Transactions on Robotics*, 24(5): pages 932–945, October 2008.

[29] M. Chli, A. J. Davison "Automatically and Efficiently Inferring the Hierarchical Structure of Visual Maps" In IEEE International Conference on *Robotics and Automation, 2009. ICRA '09.*

[30] T. Starner, J. Weaver, and A. Pentland. “Real-time American sign language recognition using desk and computer based video”. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1371–1375, 1998.

[31]T. Kurata, E. Okuma, M. Kourogi, and K. Sakaue. The hand mouse: GMM hand-color classification and mean shift tracking. In *Second International Workshop on Recognition, Analysis and Tracking of Faces and Gestures in Real-Time (in conjunction with ICCV)*, pages 119–124, Vancouiver, Canada, July 2001.

[32] V. Pavlovic, R. Sharma, and T. Huang. Visual interpretation of hand gestures for human-computer interaction: a review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:677–695, 1997.

[33] R. Castle “Simultaneous Recognition, Localization and Mapping for Wearable Visual Robots” Master Thesis Robotics Research Group Department of Engineering Science University of Oxford 2009