# MODELLING, SIMULATION AND TESTING OF ARTIFICIAL NEURAL NETWORK AUGMENTED KALMAN FILTER FOR INS/GPS AND MAGNETOMETER INTEGRATION

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

BY

DOĞAN YILDIZ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
MECHANICAL ENGINEERING

SEPTEMBER 2016

Approval of the thesis:

**MODELLING, SIMULATION AND TESTING OF ARTIFICIAL NEURAL NETWORK AUGMENTED KALMAN FILTER FOR INS/GPS AND MAGNETOMETER INTEGRATION**

submitted by **DOĞAN YILDIZ** in partial fulfillment of the requirements for the degree of **Master of Science in Mechanical Engineering Department, Middle East Technical University** by,

Prof. Dr. M. Gülbin Dural Ünver _____
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Raif Tuna Balkan _____
Head of Department, **Mechanical Engineering**

Assoc. Prof. Dr. E. İlhan Konukseven _____
Supervisor, **Mechanical Engineering Department, METU**

Dr. Volkan Nalbantoğlu _____
Co-supervisor, **Turkish Aerospace Industries, TAI**

**Examining Committee Members:**

Prof. Dr. Kemal Özgören
Mechanical Engineering Department, METU _____

Assoc. Prof. Dr. E. İlhan Konukseven
Mechanical Engineering Department, METU _____

Prof. Dr. Kemal Leblebicioğlu
Electrical and Electronics Engineering Department, METU _____

Assist. Prof. Dr. Selçuk Himmetoğlu
Mechanical Engineering Department, Hacettepe University _____

Assist. Prof. Dr. Kutluk Bilge Arıkan
Mechatronics Engineering Department, Atılım University _____

**Date:** _____

**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: DOĞAN YILDIZ

Signature :

# ABSTRACT

MODELLING, SIMULATION AND TESTING OF ARTIFICIAL NEURAL NETWORK AUGMENTED KALMAN FILTER FOR INS/GPS AND MAGNETOMETER INTEGRATION

Yıldız, Doğan

M.S., Department of Mechanical Engineering

Supervisor : Assoc. Prof. Dr. E. İlhan Konukseven

Co-Supervisor : Dr. Volkan Nalbantoğlu

September 2016, 197 pages

The objective of this thesis is to investigate a hybrid Artificial Intelligence/ Kalman Filter (AI/KF) system to determine 3D attitude, velocity and position of a vehicle in challenging GPS environment. In navigation problem, the aim is to determine the position and velocity of the host vehicle from initial conditions. By using Inertial Measurement Unit (IMU), it is possible to calculate position and velocity with an error. In other words, during the integration stage of the IMU measurement, errors will be accumulated throughout the time. In literature to eliminate the divergent characteristic of integral calculation, other sensor measurements are combined with navigation calculation process. The traditional complementary technique to calculate the vehicle position, velocity and attitude is integrated Inertial Navigation System (INS) and Global Positing System (GPS). The integrated INS/GPS shows greater accuracy with respect to standalone INS. To achieve this accuracy it is common to use Kalman Filter as an integration technique.

The Kalman Filter approach has been used widely as the standard optimal estimation technique. However because of the acquiring an accurate stochastic model and prior knowledge of the measurement error for the precision of the estimation KF has several shortcomings. Based on these shortcomings Artificial Intelligence (AI) based techniques are motivated.

In this thesis, navigation mechanization algorithm and sensors mathematical models were studied. Accelerometer, gyroscope, GPS and magnetometer were selected for the sensor fusion integration. With the help of accelerometer and gyroscope, position and velocity of the host vehicle were realized through integration process of mechanization algorithm. Also GPS and magnetometer measurements were used for position-velocity and heading determination independent from IMU respectively.

The design of sensor fusion algorithm is based on Extended Kalman Filter (EKF). The EKF linearized mathematical model relies on the error propagation model of the mechanization equations. As for the Neural Network structure Multilayer Perceptron Neural Network (MLPNN) were used to improve the integration results during GPS outages.

After modelling and simulating the results in simulation environment, real test data were used in AI/KF based prediction algorithm. The measurement results were logged in computer to be used in algorithm. The results show how AI/KF based algorithm is more accurate during GPS outages with respect to standalone Kalman Filter algorithm.

Keywords: Inertial Navigation System (INS), Global Positioning System (GPS), Extended Kalman Filter (EKF), Magnetometer, Artificial Neural Network(ANN), Multilayer Perceptron Neural Network (MLPNN)

# ÖZ

YAPAY SİNİR AĞLARI İLE GENİŞLETİLMİŞ KALMAN FİLTRESİNİN BÜTÜNLEŞTİRİLMİŞ ANS/KKS VE MANYETOMETRE İLE MODELLENMESİ, SİMÜLASYONU VE TEST EDİLMESİ

Yıldız, Doğan
Yüksek Lisans, Makina Mühendisliği Bölümü
Tez Yöneticisi : Doç. Dr. E. İlhan Konukseven
Ortak Tez Yöneticisi : Dr. Volkan Nalbantoğlu

Eylül 2016 , 197 sayfa

Bu tezde amaç küresel konumlama sisteminin devre dışı kaldığı durumlarda karma yapay zeka ve Kalman filtresini kullanarak 3 boyutlu yönelme açıları, hız ve pozisyon değerlerini belirlemektir. Seyrüsefer çözümlemelerinde amaç, aracın ilk durum bilgilerini kullanarak konum ve hız belirlemektir. Ataletsel ölçüm birimini kullanarak konum ve hız bilgisine hatalı olarak ulaşılabilir. Diğer bir söylemle, ataletsel ölçüm birimini kullanarak integral alma işlemi sırsında, hata değerleri zaman içerisinde artacaktır. Literatürde interal alma sırasında oluşan hata oranlarını önlemek için, diğer sensör ölçümlerinden yararlanılır. Geleneksel olarak konum,hız ve yönelim açılarını ölçmek için tümleştirilmiş ataletsel ölçüm birimi ve küresel konumlama sistemi kullanılır. Tek başına kullanılan ataletsel ölçüm birimine kıyasla tümleştirilmiş AÖB ve KKS daha doğru sonuçlar göstermektedir. Bu doğruluğu yakalamak için ise Kalman filtresi kullanılmaktadır.

Kalman filtresi yaygın bir şekilde kullanılan standart optimum kestirim tekniklerindendir. Fakat doğru bir stokastik modelin elde edilmesi ve ilk hata değerlerinin bilinmesinin getirdiği zorluklar Kalmam filtesinin belli başlı eksiklikleridir. Bu eksikliklere dayanarak yapay zeka tabanlı teknikler ön plana çıkmaktadır.

Bu tezde, seyrüsefer algoritması ve algılayıcı model algoritmaları gelistirildi. Algılayıcı tümleştirmede ivme ölçer, açıölçer, KKS ve manyetik alan ölçer kullanılmıştır. İvme ölçer ve açı ölçer yardımıyla pozisyon ve hız kestirimleri seyrüsefer algoritmaları kullanılarak elde edilmektedir. Ayrıca KKS ve manyetik alan ölçer kullanılarak bağımsız bir şekilde konum-hız ve yönelim açısı sırasıyla elde edilmektedir.

Algılayıcı tümleştirme algoritması genişletilmiş Kalman filtresine dayanmaktadır. Genişletilmiş Kalman filtresi doğrusallaştırılmış seyrüsefer hata modeline göre formülize edilmiştir. Yapay sinir ağları yapısı için ise küresel konumlandırma sisteminin devre dışı kalması durumunda çok katmanlı algılayıcı fonksiyon kullanılmıştır.

Modelleme ve simülasyon sonuçlarının elde edilmesinden sonra, gerçel test verileri tümleştirilmiş yapay sinir ağı ve Kalman filtresinde kullanılmıştır. Ölçüm verileri bilgisayar ortamında kaydedilip tümleştirilmiş yapıya beslenmiştir. Sonuçlar KKS'nin devre dışı kaldığı durumlarda, sadece Kalman filtresinin kullanıldığı durumlara kıyasla tümleştirilmiş yapay sinir ağı ve Kalman filtesi yapısının kullanılmasının nasıl daha iyi sonuç verdiğini göstermektedir.

Anahtar Kelimeler: Ataletsel Navigasyon Sistemi (ANS), Küresel Konumlama Sistemi (KKS), Genişletilmiş Kalman Filtresi, Manyetometre, Yapay Sinir Sistemi, Çok Katmanlı Algılayıcı Fonksiyon

*Dedicated to My Dear Family...*

# ACKNOWLEDGMENTS

First of all, I would like to express my sincere thanks to my supervisor Assoc. Prof. Dr. Erhan İlhan Konukseven for his complete guidance, advice and criticism throughout the MSc study.

I would also like to thank my co-advisor Dr. Volkan Nalbantoğlu for his valuable ideas throughout this study.

I would like to express my appreciation to Turkish Aerospace Industries Inc. for providing me a peaceful working environment and continuous support.

I also would like to thank my colleagues for their valuable support and encouragements throughout this study. Especially I would like to express my sincere thanks to Ersin Gönül for his valuable support and advices. Without his support this study would not be finished.

Finally, I would express my gratitude to my parents whose support and constant encouragement helped me throughout this study. My deepest appreciation is expressed to them for their love, understanding, and inspiration. Without their encouragement, I would not have been able to finish this work.

# TABLE OF CONTENTS

# LIST OF TABLES

TABLES

# LIST OF FIGURES

FIGURES

# NOMENCLATURE

## Symbols Scalar

| | |
|---|---|
| $a_x$ | Acceleration along x-axis of body |
| $a_y$ | Acceleration along y-axis of body |
| $a_z$ | Acceleration along z-axis of body |
| $c_{ij}$ | i, j element of direction cosine matrix |
| $f$ | Magnitude of specific force |
| $f_N$ | Magnitude of north component of specific force |
| $f_E$ | Magnitude of east component of specific force |
| $f_D$ | Magnitude of down component of specific force |
| $g$ | Magnitude of acceleration due to Earth's gravity |
| $h$ | Height above ground |
| $l$ | Longitude |
| $L$ | Latitude |
| $p$ | Roll rate |
| $q$ | Pitch rate |
| $r$ | Yaw rate |
| $R_0$ | Mean radius of curvature Earth |
| $v_N$ | Magnitude of north velocity |
| $v_E$ | Magnitude of east velocity |
| $v_D$ | Magnitude of down velocity |
| $\phi$ | Roll Euler Angle |
| $\theta$ | Pitch Euler Angle |
| $\psi$ | Yaw Euler Angle |
| $\omega_x$ | Roll rate of body wrt navigation frame |
| $\omega_y$ | Pitch rate of body wrt navigation frame |
| $\omega_z$ | Yaw rate of body wrt navigation frame |
| $\Omega$ | Turn rate of the Earth |

## Symbols Vector and Matrice

| | |
|---|---|
| $\boldsymbol{C}_b^i$ | DCM from body frame to inertial reference frame |
| $\boldsymbol{C}_b^e$ | DCM from body frame to Earth frame |
| $\boldsymbol{C}_b^n$ | DCM from body frame to navigation frame |
| $\boldsymbol{f}$ | Specific force vector |

| | |
|---|---|
| $\boldsymbol{g_l}$ | Local gravity vector |
| $\boldsymbol{q}$ | quaternion |
| $\boldsymbol{r}$ | Position vector |
| $\boldsymbol{v_i}$ | Velocity wrt inertial frame |
| $\boldsymbol{v_e}$ | Velocity wrt Earth frame |
| $\boldsymbol{\omega}_{ie}$ | Turn rate of the Earth wrt inertial frame |
| $\boldsymbol{\omega}_{ib}$ | Turn rate of the body wrt inertial frame |
| $\boldsymbol{\omega}_{eb}$ | Turn rate of the body wrt Earth frame |
| $\boldsymbol{\omega}_{in}$ | Turn rate of navigation frame wrt inertial frame |
| $\boldsymbol{\omega}_{en}$ | Turn rate of navigation frame wrt Earth frame |
| $\boldsymbol{\Omega}_{ib}$ | skew symmetric matrix of turn rate of body frame wrt inertial reference frame |
| $\boldsymbol{\Omega}_{eb}$ | skew symmetric matrix of turn rate of body frame wrt Earth frame |
| $\boldsymbol{\Omega}_{nb}$ | skew symmetric matrix of turn rate of body frame wrt navigation frame |

**Acronyms**

| | |
|---|---|
| AHRS | Attitude Heading Reference System |
| ANN | Artificial Neural Network |
| DCM | Direction Cosine Matrix |
| EKF | Extended Kalman Filter |
| GLONASS | Global Navigation Satellite System |
| GPS | Global Positioning System |
| INS | Inertial Navigation System |
| IMU | Inertial Measurement Unit |
| KF | Kalman Filter |
| MEMS | Microelectromechanical systems |
| MLPNN | Multilayer Perceptron Neural Network |
| NN | Neural Network |

# CHAPTER 1

# INTRODUCTION

## 1.1 Background and Problem Statement

Through out the history, navigation is an essential part about travelling and finding the way from one place to another. Especially in today's modern world navigation systems become more and more popular in everyday life including vehicle navigation, smart phones location determination. With the enhancements about navigation solution, more demands have been raised about the precision and accuracy of the determination of position, velocity and orientation of vehicle. To deal with this higher accuracy demand, people in this research area have developed higher cost and relatively high precision inertial sensor systems. In this case however higher cost has shown as an another problem. As a result, integrated navigation systems have been started to use for diminishing the higher cost values.

Inertial navigation with Global positioning system (GPS) is the most common type integrated system. The primary reasons for this integration are that INS/GPS system has complementary characteristics and with the developing technology MEMS based inertial measurement units (IMU) and GPS signals are affordable to all user. Whether INS/GPS integration shows great accuracy, there are still limitations in these sensor integration system.

GPS is the satellite based system that allows users with a GPS receiver to obtain position and velocity information worldwide. For last two decade with developing technology GPS system has been proven the effective tool to determine position and velocity. Despite these advantages, GPS need a direct line of sight with at least four

satellites. Especially in urban canyons, forests and highway tunnels/overpasses the degradation of measurements show itself and the strength of estimation accuracy diminishes severely.

Another important part of navigation is inertial measurement unit (IMU). Despite GPS, IMU is autonomous system which measures acceleration and orientation rates in 6 DOF manner. An inertial navigation system (INS) containing IMU integrates these rotation rates and accelerations in three orthogonal axis through navigation equations. Therefore the position,velocity and orientation matrix can be found as a result of this computation. As for the disadvantages of the IMU, sensor inherited noise values such as bias error, scale factor, misalignment error, etc. cause the rapid divergent in velocity and position calculation via integration. Because of this effect, to increase the accuracy of calculations, incorporated navigation computations have been designed.

When the two systems INS and GPS are compared, it can be seen clearly that these two systems reflect complementary characteristics.While INS is a self-contained autonomous system with good short term accuracy, GPS has good long term accuracy with limited errors. With the help of this complementary characteristic, their integration process overcomes individual drawbacks and provides more accurate and robust navigation solution. Of course to obtain this accuracy Kalman Filter also affects the results of estimation procedure. The real challenge for the sensor fusion of the INS/GPS system is the determination of accurate dynamic of stochastic error model for KF. Moreover KF requires linearized dynamic error model. However when the error model is linearized, the higher order terms are neglected. In this case non-linear terms must be considered carefully, because they directly influence the navigation solution.

## 1.2 Research Objectives

The objective of this thesis is that the development and testing of the artificial neural network augmented Kalman filter simulation model which is aimed to be used for land and air vehicles. Within the scope of thesis objective, Kalman filter behaviour during GPS outages are observed. The same procedure is realized with ANN augmented

system and the results are compared and the test results are presented. The aim is to improve position and velocity estimation accuracy with neural network structure.

To reach the main objective, development process covers the following stages:

1. Understanding the mechanization equations and their implementation to simulation model.
2. Design of the individual sensor models and observe their characteristics. Select correct error parameters for sensor models.
3. Analysis of Kalman filter algorithm and implementation of Kalman filter into sensor fusion structure.
4. Implement neural network module to Kalman filter sensor fusion structure.
5. Examine the simulation model with artificial and real test data. Validate the applicability of neural network into Kalman filter model during GPS outages.

### 1.3 Thesis Outline

**Chapter 1** presents the background information of the thesis study, explaining the general concept of INS/GPS integration and neural network augmented Kalman filter. Moreover, the thesis problem and objective are presented.

**Chapter 2** introduces the basic concepts of navigation and attitude computation. Both of these equations are the fundamental for strapdown inertial navigation systems. Also reference frame concepts and transformation matrices will be discussed.

**Chapter 3** covers the gyroscope and accelerometer technologies. The mathematical model of inertial sensor system will be explained.

**Chapter 4** explains the INS/GPS/Magnetometer integration with Kalman filter. Also sensor fusion concept and basis knowledge of Kalman filter will be discussed. For non-linear system Extended Kalman Filter will be explained.

**Chapter 5** presents the development of the neural network augmented Kalman filter

integration technique. The neural network basis, structure and parametrization will be explained in detail.

**Chapter 6** provides the simulation model and the test results of simulation. Also simulation model will be tested with experimental test data and the results will be discussed in this chapter.

**Chapter 7** presents the conclusion and some recommendations for future works.

# CHAPTER 2

# INERTIAL NAVIGATION EQUATION

## 2.1 Introduction

In inertial navigation system the basic calculation can be outlined as the double integration of specific force measurements to determine position of the vehicle. The first integration of the vehicle provides the velocity information, and the second integration yields the vehicle position with respect to initial conditions. Also projection of acceleration into desired reference frame can be achieved through angular orientations. To provide this knowledge angular velocities are integrated with respect to initial orientations. The fundamental concept is illustrated in Figure 2.1. An INS implements the concept of using a collection of accelerometer to sense

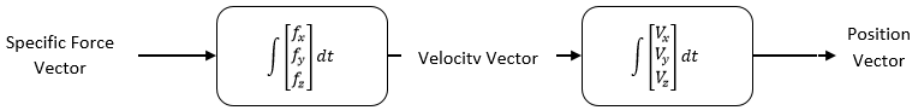



Figure 2.1: Fundamental Inertial Navigation Concept

specific force in each axis and gyroscope to sense angular velocity. The direction of accelerometers is determined using the angle or angular rate sense measurements instruments, gyroscopes. To apply the Figure 2.1 in the INS, the accelerometers would be specified for determination of total acceleration. In general total acceleration is composed of two fundamental parts: gravity acceleration created by gravity field and specific force acceleration produced by forces acting on the vehicle. Due to basic limitations of physics which are explained in chapter 3 accelerometers

can only be designed to measure specific force. Hence to know total acceleration in Figure 2.1, the gravity acceleration must be calculated. The gravity calculation is performed within INS by computing position [4]. The inertial navigation system concept is given in Figure 2.2

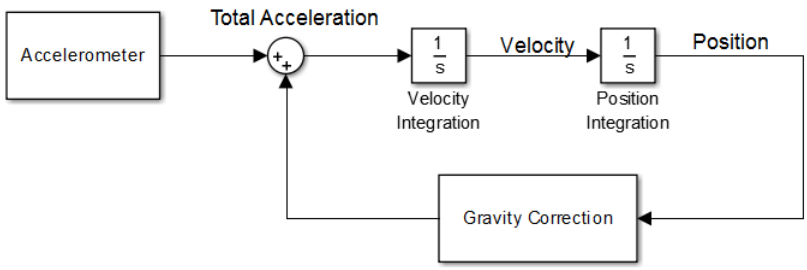
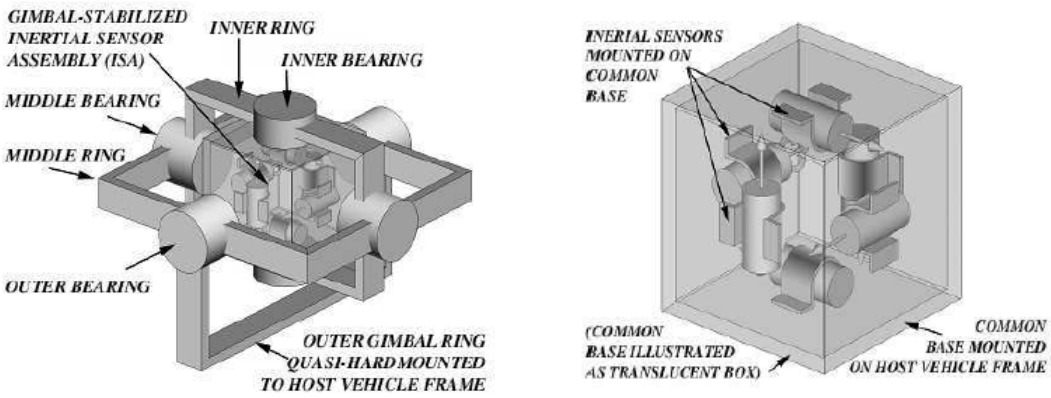



Figure 2.2: Inertial Navigation System Concept

Two types of INS mechanization implementations exist as shown in Figure 2.3. These are *Gimbaled Systems* and *Strapdown Systems*. Principally both systems are realized the same procedure. However the implementations are different. For gimbaled approach the accelerometers are mounted to a rigid structure that is mechanically coupled to user vehicle by set of concentric gimbals. The gimbals are connected to accelerometer mount and to the user vehicle by bearing assemblies that provide rotational freedom around bearing axes. Therefore sensors are isolated from the rotations of the outer system. The sensor stabilization is sustained by using the feedback from the gyroscopes. The advantage of this system is precise measurement. However gimbal systems are expensive and complicated. Also one important drawback is that when two rotations are aligned, gimbal lock phenomena occurs. One rotations can not be distinguished from other rotation.

In strapdown approach, inertial sensor platform is mounted directly within the INS chassis. That is so they are called "Strapdown" [4]. Therefore sensors follow the motion of the system. Because of these higher rotations are experienced by the sensors, high error results are observed. Nevertheless strapdown systems are preferred because of their small size, lower cost and weight. In this thesis strapdown system model is used. This chapter provides the background information about the inertial navigation system equations based on Newton's Second Law of motion,




(a) Gimbaled System (b) Strapdown System

Figure 2.3: Gimbaled and strapdown inertial measurement units [3]

which is necessary to understand mechanization algorithm. Firstly reference coordinate frame concept will be considered. After that reference frame transformations and mechanization equations will be obtained. Finally Earth gravity model and digital version of the mechanization algorithm will be explained.

## 2.2 Reference Coordinate Frames

The significant part of inertial navigation is the precise definition of Cartesian co-ordinate reference frames which are used in navigation equations. According to chosen reference frame, accelerometer and gyroscope readings must be converted. Generally, each frame is an orthogonal, right-handed, co-ordinate frame.

For navigation purposes, frames have an important role to navigate in the vicinity of the Earth. Inertial reference frame which is stationary according to fixed stars, Earth reference frame and local geographic frames are defined for terrestrial navigation purposes [1]. Some of these coordinate frames are shown in Figure 2.4 and the following coordinate frames are used in this study.

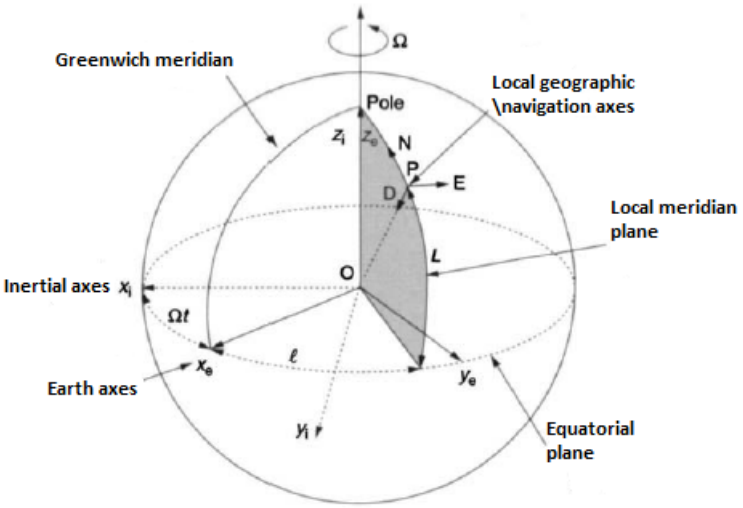



Figure 2.4: Frames of reference [1]

### 2.2.1 Inertial Frame

Earth centered inertial frame or inertial frame (i-frame) has its origin at the center of the earth and axes are non-rotating with respect to fixed stars. In Figure 2.4, $Ox_i, Oy_i, Oz_i$ are defined axes where $Oz_i$ is coincident with Earth's polar axis. For navigational purposes $x_i$ and $y_i$ lie in the equatorial plane.

### 2.2.2 Earth Frame

The Earth frame (e-frame) has its origin at the center of the Earth. Earth centered frame is defined by $Ox_e, Oy_e, Oz_e$. The axis $Ox_e$ lies along the intersection of plane of Greenwich meridian with Earth's equatorial frame and $y_e$ is 90°east of $x_e$. The Earth frame rotates with respect to inertial frame at rate of $\Omega = 7.292115 \times 10^{-5}$ rad/s.

### 2.2.3 Navigation Frame

Navigation frame (n-frame) is a local geographic frame. The origin of the n-frame is located on point P which is location of the navigation system (shown in Figure 2.4).

The axes of the navigation frame are aligned with north, east and down (local vertical) directions. The turn rate of the navigation frame with respect to Earth-fixed frame $\omega_{en}$ is determined by the motion of point P with respect to Earth. This situation is often stated as transport rate.

### 2.2.4 Wander Azimuth Frame

Wander azimuth frame (w-frame) is used for special conditions. W-frame is mainly selected for eliminating the singularities in navigation computation at the poles. Similar to n-frame, it is locally level. The rotation takes place through wander angle about local vertical in the vicinity of poles.

### 2.2.5 Body Frame

The body frame (b-frame) has its origin at the mass center of the host vehicle. The body orthogonal axes are aligned with the roll, pitch and yaw axes of the vehicle. The axes directions are figured in Figure 2.5. The axes directions are defined: roll

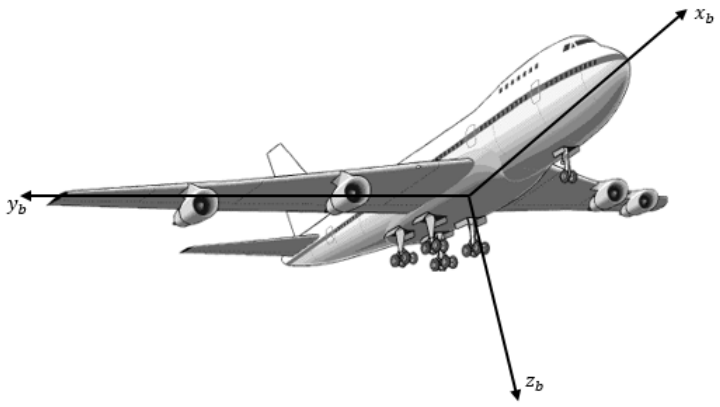



Figure 2.5: Body reference frame

axis along the longitudinal axis, pitch axis is directed 90°to the right and yaw axis is directed downward.

## 2.3 Basic Principles and Reference Frame Transformations

This section will review some of basic mathematical techniques which are encountered in navigational computations and derivations. This section also gives basic knowledge of various notations for vector transformations.

### 2.3.1 Vector Notation

In three-dimensional vector notation, a vector is represented with its three parameters that are resolved in selected frames. In this study, a vector is showed in bold lower-case letters with a superscript that indicates coordinate frame in which the components of the vector are given. As an example,

$$\mathbf{r}^m = \begin{bmatrix} x^m \\ y^m \\ z^m \end{bmatrix} \tag{2.1}$$

the superscript m represents the m-frame and the elements $(x^m, y^m, z^m)$ symbolize the coordinate components in m-frame.

### 2.3.2 Vector Coordinate Transformation

Vector transformation from one frame to another is encountered in navigation computation. To realize this transformation transformation matrices are used. For instance a transformation from m frame to n fame can be written as

$$\mathbf{r}^n = C_m^n \mathbf{r}^m \tag{2.2}$$

where $C_m^n$ represents the transformation matrix that transforms vector $\mathbf{r}$ from m-frame to n-frame. For the notation usage, the superscript of the vector that is to be transformed must match the subscript of the transformation matrix.

The inverse of the transformation matrix $C_m^n$ is the inverse of the procedure from m-frame to n-frame. Now the vector transformation from n-fame to m-frame is,

$$\mathbf{r}^m = (C_m^n)^{-1} \mathbf{r}^n = C_n^m \mathbf{r}^n \tag{2.3}$$

Also for all navigation frames are orthogonal frames of references. Therefore the inverse and the transpose of the transformation matrices are equal.

$$C_m^n = (C_n^m)^t = (C_n^m)^{-1} \tag{2.4}$$

The orthogonality property for transformation matrices can be checked by using its vectors whether they are mutually orthogonal or not.

$$C = \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix} \tag{2.5}$$

where

$$\mathbf{c}_1 = \begin{bmatrix} c_{11} \\ c_{21} \\ c_{31} \end{bmatrix}, \ \mathbf{c}_2 = \begin{bmatrix} c_{12} \\ c_{22} \\ c_{32} \end{bmatrix}, \ \mathbf{c}_3 = \begin{bmatrix} c_{13} \\ c_{23} \\ c_{33} \end{bmatrix} \tag{2.6}$$

then for matrix $C$ to be orthogonal the following must be true.

$$\mathbf{c}_1 \cdot \mathbf{c}_2 = 0, \ \mathbf{c}_1 \cdot \mathbf{c}_3 = 0, \ \mathbf{c}_2 \cdot \mathbf{c}_3 = 0 \tag{2.7}$$

### 2.3.3 Attitude Representation

In strapdown sensors, gyroscopes are used to measure attitude of vehicle according to reference coordinate frame. By using the measurements of turn rate provided by gyroscopes the attitude is updated through the time. As indicated before the co-ordinate frames are orthogonal and right-handed axes. Thus positive rotations about each axis are taken to be counter clockwise direction. This convention is also used in this study. The rotation order and angle quantity determine the rotation changes of body. To define the these rotations in other words the attitude of the body with respect to a co-ordinate reference frame, some mathematical representations can be used. Mostly three attitude representations are utilized, these are,

- Direction Cosine Matrix (DCM)
- Euler Angle
- Quaternion

In following pages these attitude representations are explained [1].

#### 2.3.3.1 Direction Cosine Matrix

Direction cosine matrix is represented by $3 \times 3$ matrix, the column represents the unit vectors projected along the reference axes. As an example, the DCM matrix from b-frame to n-frame is symbolized as $C_b^n$ and the component form can be explained from classical vector algebra, let's say $\mathbf{w}$ and $\mathbf{v}$ are two vectors,

$$\mathbf{w} \cdot \mathbf{v} = w\ v\ \cos(\phi)$$

where

$$w, v = \text{Magnitude of } \mathbf{w} \text{ and } \mathbf{v}.$$
$$\phi = \text{Angle between } \mathbf{w} \text{ and } \mathbf{v}.$$

Now, let's explain $\mathbf{v}$ in n-frame and b-frame, and expressions can be written as,

$$\begin{aligned} \mathbf{v} &= v_x^n\ u_x^n + v_y^n\ u_y^n + v_z^n\ u_z^n \\ \mathbf{v} &= v_x^b\ u_x^b + v_y^b\ u_y^b + v_z^b\ u_z^b \end{aligned} \tag{2.8}$$

An expression for $v_x^n$ in terms of $\mathbf{v}$ components in b-frame is obtained by taking the dot product of expression $\mathbf{v}$ in b-frame with $u_x^n$, therefore

$$v_x^n = v_x^b\ u_x^b \cdot u_x^n + v_y^b\ u_y^b \cdot u_x^n + v_z^b\ u_z^b \cdot u_x^n \tag{2.9}$$

Similar procedure can be followed for other components.

$$\begin{aligned} v_y^n &= v_x^b\ u_x^b \cdot u_y^n + v_y^b\ u_y^b \cdot u_y^n + v_z^b\ u_z^b \cdot u_y^n \\ v_z^n &= v_x^b\ u_x^b \cdot u_z^n + v_y^b\ u_y^b \cdot u_z^n + v_z^b\ u_z^b \cdot u_z^n \end{aligned} \tag{2.10}$$

In matrix form we can collect the terms in equations 2.9 and 2.10,

$$\begin{bmatrix} v_x^n \\ v_y^n \\ v_z^n \end{bmatrix} = \underbrace{\begin{bmatrix} u_x^b \cdot u_x^n & u_y^b \cdot u_x^n & u_z^b \cdot u_x^n \\ u_x^b \cdot u_y^n & u_y^b \cdot u_y^n & u_z^b \cdot u_y^n \\ u_x^b \cdot u_z^n & u_y^b \cdot u_z^n & u_z^b \cdot u_z^n \end{bmatrix}}_{C_b^n \text{ Direction Cosine Matrix}} \begin{bmatrix} v_x^b \\ v_y^b \\ v_z^b \end{bmatrix} \tag{2.11}$$

Replacing the elements of DCM we can simply write,

$$C_b^n = \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix} \tag{2.12}$$

Detailed expressions can be found in reference [4]. With the help of equation 2.12 we can easily convert a vector in b-frame to n-frame. By pre-multiplying the vector with direction cosine matrix,

$$\mathbf{r}^n = C_b^n \, \mathbf{r}^b \tag{2.13}$$

As for the rate of change of $C_b^n$ with time during the motion of vehicle, it can be given by following equation from Titterton and Weston [1],

$$\dot{C}_b^n = \lim_{\delta t \to 0} \frac{\delta C_b^n}{\delta t} = \lim_{\delta t \to 0} \frac{C_b^n(t + \delta t) - C_b^n(t)}{\delta t} \tag{2.14}$$

In this equation, $C_b^n(t)$ and $C_b^n(t + \delta t)$ represent the DCM at times $t$ and $t + \delta t$, respectively. It can be shown that $C_b^n(t + \delta t)$ can be written as the product of two matrices.

$$C_b^n(t + \delta t) = C_b^n(t) \; A(t) \tag{2.15}$$

Here $A(t)$ can be thought as a direction cosine matrix relating the b-frame at time $t$ to the b-frame at time $t + \delta t$, for $A(t)$

$$A(t) = [I + \delta\Psi] \tag{2.16}$$

where $I$ is identity and $\delta\Psi$

$$\delta\Psi = \begin{bmatrix} 0 & -\delta\psi & \delta\theta \\ \delta\phi & 0 & -\delta\phi \\ -\delta\theta & \delta\phi & 0 \end{bmatrix} \tag{2.17}$$

Notice that $\delta\psi$, $\delta\theta$ and $\delta\phi$ are the small rotations through which the b-frame has rotated over time interval $\delta t$ about its yaw, pitch and roll axis respectively. Substitute $C_b^n(t + \delta t)$ in equation 2.14,

$$\dot{C}_b^n = C_b^n \lim_{\delta t \to 0} \frac{\delta\Psi}{\delta t} \tag{2.18}$$

In equation 2.18, $\delta\Psi/\delta t$ is the skew symmetric matrix of the angular rate vector $\omega_{nb}^b = [\omega_x \; \omega_y \; \omega_z]^t$, which shows the turn rate of the b-frame with respect to n-frame expressed in b-frame. Therefore instead of equation 2.18,

$$\begin{aligned} \lim_{\delta t \to 0} \frac{\delta\Psi}{\delta t} &= \Omega_{nb}^b \\ \dot{C}_b^n &= C_b^n \; \Omega_{nb}^b \end{aligned} \tag{2.19}$$

where

$$\Omega_{nb}^{b} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \tag{2.20}$$

#### 2.3.3.2 Euler Angles

A classical method for attitude representation between two coordinate frames is realized by using Euler angle rotation sequence. Transformation between coordinate frames can be fulfilled with three successive rotations. Mostly used Euler transformation sequence is $\psi\theta\phi$ transformation. The rotation may be expressed as follows,

- Rotation with $\psi$ angle about reference z-axis
- Rotation with $\theta$ angle about reference y-axis
- Rotation with $\phi$ angle about reference x-axis

Here $\psi$, $\theta$ and $\phi$ are called as *Euler Rotation Angles*. The reason for choosing this type of representation is that the physical correspondence directly related with strapdown coordinate fame rotation in each axes.

The representation of the Euler angle can be stated as three different direction cosine matrices.

$$\text{Rotation about z-axis, } C_3 = \begin{bmatrix} \cos(\psi) & \sin(\psi) & 0 \\ -\sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.21}$$

$$\text{Rotation about y-axis, } C_2 = \begin{bmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \tag{2.22}$$

$$\text{Rotation about y-axis, } C_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & \sin(\phi) \\ 0 & -\sin(\phi) & \cos(\phi) \end{bmatrix} \tag{2.23}$$

Therefore a transformation from body to reference axes and the vice versa can be expressed as follows,

$$C_b^n = C_3\, C_2\, C_1 \tag{2.24}$$

$$C_n^b = (C_b^n)^t = C_1^t\, C_2^t\, C_3^t \tag{2.25}$$

$$C_b^n = \begin{bmatrix} \cos(\psi) & \sin(\psi) & 0 \\ -\sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & \sin(\phi) \\ 0 & -\sin(\phi) & \cos(\phi) \end{bmatrix}$$

By collecting the terms,

$$C_b^n = \begin{bmatrix} \cos(\theta)\cos(\psi) & -\cos(\phi)\sin(\psi) + \sin(\phi)\sin(\theta)\cos(\psi) & \sin(\phi)\sin(\psi) + \cos(\phi)\sin(\theta)\cos(\psi) \\ \cos(\theta)\sin(\psi) & \cos(\phi)\cos(\psi) + \sin(\phi)\sin(\theta)\sin(\psi) & -\sin(\phi)\cos(\psi) + \cos(\phi)\sin(\theta)\sin(\psi) \\ -\sin(\theta) & \sin(\phi)\cos(\theta) & \cos(\phi)\cos(\theta) \end{bmatrix} \tag{2.26}$$

The important part about the Euler angles is that how to propagate through time when vehicle is in motion. The relation between body rates and the Euler angle rates can be shown as in equation 2.27 and 2.28.

$$\begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + C_1 \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + C_1\, C_2 \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} \tag{2.27}$$

This equation can be written to find Euler angle rates as in follows.

$$\begin{aligned} \dot{\phi} &= (\omega_y \sin(\phi) + \omega_z * \cos(\phi)) \tan(\theta) \\ \dot{\theta} &= \omega_y \cos(\phi) - \omega_z \sin(\phi) \\ \dot{\psi} &= (\omega_y \sin(\phi) + \omega_z \cos(\phi)) \sec(\theta) \end{aligned} \tag{2.28}$$

Notice that the equation in 2.28 have a limitation. The uncertainty of $\dot{\phi}$ and $\dot{\psi}$ equations are observed when $\theta = \pm 90^\circ$.

#### 2.3.3.3 Quaternions

The four-parameter representation of rotation is called as quaternion. In this representation, the transformation between two coordinate frames can be expressed

by a single rotation about a vector $\boldsymbol{\mu}$ defined with respect to selected frame. As an example of quaternion

$$\mathbf{q} = \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} \cos(|\boldsymbol{\mu}|/2) \\ (\mu_x/|\boldsymbol{\mu}|)\sin(\mu/2) \\ (\mu_y/|\boldsymbol{\mu}|)\sin(\mu/2) \\ (\mu_z/|\boldsymbol{\mu}|)\sin(\mu/2) \end{bmatrix} \tag{2.29}$$

where $\mathbf{q}$ is quaternion and $\mu_x, \mu_y, \mu_z$ are the components of $\boldsymbol{\mu}$ and $|\boldsymbol{\mu}|$ is the magnitude of the $\boldsymbol{\mu}$.

In equation 2.29, quaternion is symbolized with four parameter $a, b, c$ and $d$. A quaternion is also expressed as a four parameter complex number with a real component $a$ and three imaginary parameter $b, c, d$.

$$\mathbf{q} = a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k} \tag{2.30}$$

From this expression by using the product of complex number property, the product of two quaternion can be expressed as,

$$\begin{aligned} \mathbf{i}\cdot\mathbf{i} = -1 \quad \mathbf{i}\cdot\mathbf{j} = \mathbf{k} \quad \mathbf{j}\cdot\mathbf{i} = -\mathbf{k} \\ \mathbf{j}\cdot\mathbf{j} = -1 \quad \mathbf{j}\cdot\mathbf{k} = \mathbf{i} \quad \mathbf{k}\cdot\mathbf{j} = -\mathbf{i} \\ \mathbf{k}\cdot\mathbf{k} = -1 \quad \mathbf{k}\cdot\mathbf{i} = \mathbf{j} \quad \mathbf{i}\cdot\mathbf{k} = -\mathbf{j} \end{aligned}$$

Therefore the two quaternion product such as $\mathbf{q} = a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k}$ and $\mathbf{p} = e + f\mathbf{i} + g\mathbf{j} + h\mathbf{k}$ can be formalized,

$$\begin{aligned} \mathbf{q}\cdot\mathbf{p} &= (a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k})(e + f\mathbf{i} + g\mathbf{j} + h\mathbf{k}) \\ &= ea - bf - cg - dh + (af + be + ch - dg)\mathbf{i} + \\ &\quad (ag + ce - bh + df)\mathbf{j} + (ah + de + bg - cf)\mathbf{k} \end{aligned} \tag{2.31}$$

Equation 2.31 can also be represented as matrix equation.

$$\mathbf{q}\cdot\mathbf{p} = \begin{bmatrix} a & -b & -c & -d \\ b & a & -d & c \\ c & d & a & -b \\ d & -c & b & a \end{bmatrix} \begin{bmatrix} e \\ f \\ g \\ h \end{bmatrix} \tag{2.32}$$

In the light of usage of quaternion main question is how quaternions are used in vector transformation? To answer this question define a vector quantity in b-frame $\mathbf{r}^b$ and in n-frame $\mathbf{r}^n$. Firstly $\mathbf{r}^b$ must be converted into quaternion equivalent.

$$\begin{aligned}\mathbf{r}^b &= x\mathbf{i} + y\mathbf{j} + z\mathbf{k} \\ \mathbf{r}^{b'} &= 0 + x\mathbf{i} + y\mathbf{j} + z\mathbf{k}\end{aligned} \tag{2.33}$$

Then by using quaternion $\mathbf{q}$, $\mathbf{r}^{n'}$ can be computed.

$$\mathbf{r}^{n'} = \mathbf{q}\mathbf{r}^{b'}\mathbf{q}^* \tag{2.34}$$

where $\mathbf{q}^* = a - b\mathbf{i} - c\mathbf{j} - d\mathbf{k}$, the complex conjugate of $\mathbf{q}$. Finally if the terms are collected

$$\begin{aligned}\mathbf{r}^{n'} =& 0 + [(a^2 + b^2 - c^2 - d^2)x + 2(bc - ad)y + 2(bd + ac)z]\mathbf{i} \\ &+ [2(bc + ad)x + (a^2 - b^2 + c^2 - d^2)y + 2(cd - ab)z]\mathbf{j} \\ &+ [2(bd - ac)x + 2(cd + ab)y + (a^2 - b^2 - c^2 + d^2)]\mathbf{k}\end{aligned} \tag{2.35}$$

Alternatively, $\mathbf{r}^{n'}$ may be expressed in matrix form,

$$\mathbf{r}^{n'} = C'\,\mathbf{r}^{b'} \tag{2.36}$$

where

$$C' = \begin{bmatrix} 0 & 0 \\ 0 & C \end{bmatrix} \tag{2.37}$$

$$C = \begin{bmatrix} (a^2 + b^2 - c^2 - d^2) & 2(bc - ad) & 2(bd + ac) \\ 2(bc + ad) & (a^2 - b^2 + c^2 - d^2) & 2(cd - ab) \\ 2(bd - ac) & 2(cd + ab) & (a^2 - b^2 - c^2 + d^2) \end{bmatrix} \tag{2.38}$$

Comparison with equation 2.13, equation 2.38 is equivalent to direction cosine matrix.

The other important part for the quaternion is the propagation of the quaternion with time. The following equation is used for this purposes,

$$\dot{\mathbf{q}} = 0.5\,\mathbf{q} \cdot \mathbf{p}^b_{nb} \tag{2.39}$$

In this equation $\mathbf{p}_{nb}^{b} = [0\ \omega_{nb}^{b}{}^{t}]^{t}$ and in matrix form equation 2.39 can be written as [5],

$$\dot{\mathbf{q}} = \begin{bmatrix} \dot{a} \\ \dot{b} \\ \dot{c} \\ \dot{d} \end{bmatrix} = 0.5 \begin{bmatrix} a & -b & -c & -d \\ b & a & -d & c \\ c & d & a & -b \\ d & -c & b & a \end{bmatrix} \begin{bmatrix} 0 \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{2.40}$$

Any equation form to determine the attitude of the strapdown navigation system can be utilized. In this thesis direction cosine matrix and Euler angle formulation will be used mainly.

## 2.4 Detailed INS Mechanization Equations

INS mechanization is the process of determining navigation states by using differential equation of motion and the raw inertial measurement data from IMU measurement.

Mechanization equations are usually expressed in local level frame, however different mechanization techniques are also available for purpose of usage [6]. In this section attention is focussed on local geographic navigation frame (n-frame) mechanisation. In Figure 2.6, the illustration of local geographic navigation frame mechanization is shown.

### 2.4.1 Velocity and Position Formulation

In strapdown navigation systems, it is required to calculate vehicle velocity with respect to Earth which is the ground velocity in inertial axes. Inertial velocity can be expressed in terms of ground velocity using Coriolis equation. Let $\mathbf{r}$ denotes the position vector of vehicle with respect to inertial frame. Then inertial velocity and ground velocity relation will be,

$$\underbrace{\left.\frac{d\mathbf{r}}{dt}\right|_i}_{\text{Inertial Velocity}} = \underbrace{\left.\frac{d\mathbf{r}}{dt}\right|_e}_{\text{Ground Velocity } \mathbf{v}_e} + \boldsymbol{\omega}_{ie} \times \mathbf{r} \tag{2.41}$$

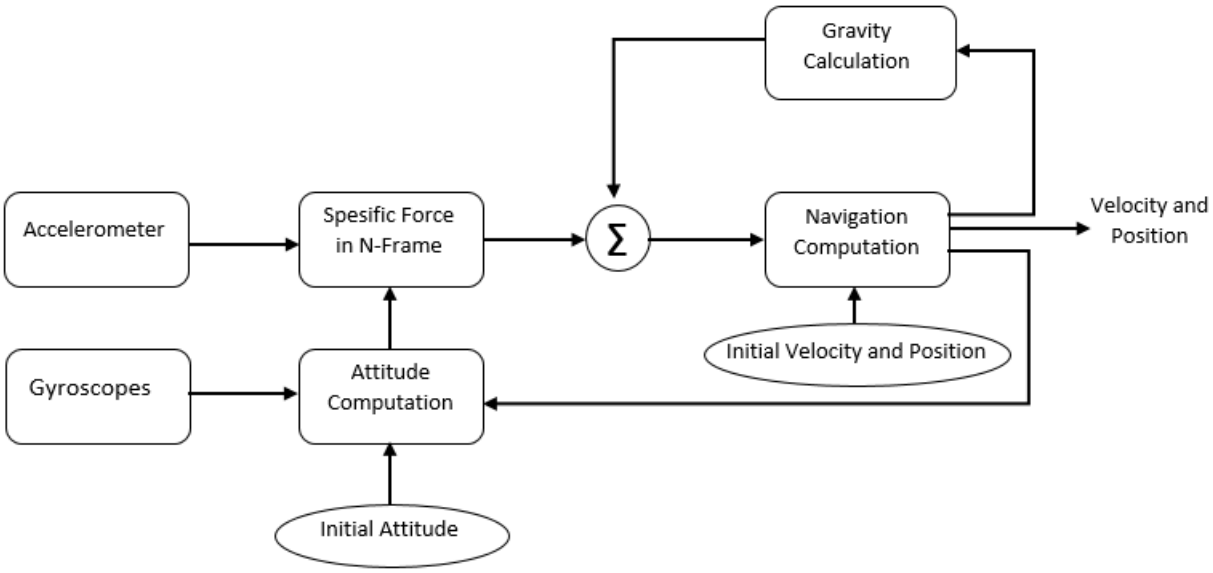



Figure 2.6: Local geographic navigation frame mechanization

Now differentiate this expression,

$$\left.\frac{d^2\mathbf{r}}{dt^2}\right|_i = \left.\frac{d\boldsymbol{v}_e}{dt}\right|_i + \left.\frac{d}{dt}[\boldsymbol{\omega}_{ie} \times \mathbf{r}]\right|_i \tag{2.42}$$

Applying the Coriolis equation to second term in equation 2.42

$$\left.\frac{d^2\mathbf{r}}{dt^2}\right|_i = \left.\frac{d\boldsymbol{v}_e}{dt}\right|_i + \boldsymbol{\omega}_{ie} \times \mathbf{v}_e + \boldsymbol{\omega}_{ie} \times [\boldsymbol{\omega}_{ie} \times \mathbf{r}] \tag{2.43}$$

It is assumed that the turn rate of the Earth is constant. Let rewrite the equation 2.43 by adding the gravity term because of that accelerometers will provide a measure of specific force only,

$$\left.\frac{d\mathbf{v}_e}{dt}\right|_i = \mathbf{f} - \boldsymbol{\omega}_{ie} \times \boldsymbol{v}_e - \boldsymbol{\omega}_{ie} \times [\boldsymbol{\omega}_{ie} \times \mathbf{r}] + \mathbf{g} \tag{2.44}$$

In this equation,

$\mathbf{f}$: represents the specific force acceleration of navigation system

$(\boldsymbol{\omega}_{ie} \times \boldsymbol{v}_e)$: is the Coriolis acceleration of the navigation system on rotating Earth system.

$\boldsymbol{\omega}_{ie} \times [\boldsymbol{\omega}_{ie} \times \mathbf{r}]$: is the centripetal acceleration on navigation system because of the Earth rotation.

The centripetal acceleration is not distinguishable from the gravitational acceleration. Therefore mostly the sum of centripetal acceleration and gravitational acceleration is used in equations. The sum of acceleration is called as local gravity vector. The local gravity vector is denoted as $g_l$. In Figure 2.7, the local gravitational vector, gravitational field intensity and centripetal acceleration are shown [1].

$$\boldsymbol{g}_l = \boldsymbol{g} - \boldsymbol{\omega}_{ie} \times [\boldsymbol{\omega}_{ie} \times \boldsymbol{r}] \tag{2.45}$$

Combining the equations 2.44 and 2.45 gives the following expression,

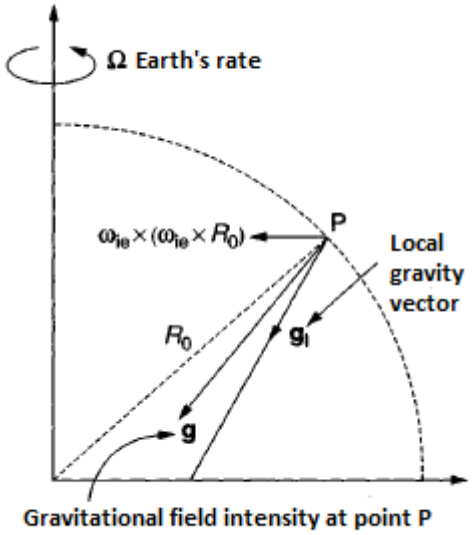



Figure 2.7: Components of gravitational field [1]

$$\left.\frac{d\boldsymbol{v}_e}{dt}\right|_i = \boldsymbol{f} - \boldsymbol{\omega}_{ei} \times \boldsymbol{v}_e + \boldsymbol{g}_l \tag{2.46}$$

Now, so far the mechanization with respect to inertial frame is computed. However the purpose is to express this mechanization in navigational frame. By using the inertial mechanization, local level frame mechanization can be written easily as in equation 2.47. In this mechanization, ground velocity is expressed in navigation coordinates to give $\boldsymbol{v}_e^n$. The rate of change of $\boldsymbol{v}_e^n$ with respect to navigation axes can be expressed in terms of rate of change in inertial axes,

$$\left.\frac{d\boldsymbol{v}_e}{dt}\right|_n = \left.\frac{d\boldsymbol{v}_e}{dt}\right|_i - [\boldsymbol{\omega}_{ie} + \boldsymbol{\omega}_{en}] \times \boldsymbol{v}_e \tag{2.47}$$

Substituting the equation 2.46,

$$\left.\frac{d\boldsymbol{v}_e}{dt}\right|_n = \boldsymbol{f} - [2\boldsymbol{\omega}_{ie} + \boldsymbol{\omega}_{en}] \times \boldsymbol{v}_e + \boldsymbol{g}_l \tag{2.48}$$

Finally if this equation is expressed in navigation frame,

$$\dot{\boldsymbol{v}}_e^n = C_b^n \boldsymbol{f}^b - [2\boldsymbol{\omega}_{ie}^n + \boldsymbol{\omega}_{en}^n] \times \boldsymbol{v}_e^n + \boldsymbol{g}_l^n \tag{2.49}$$

The measurements of specific force provided by the accelerometers are in body axes, as denoted by $\boldsymbol{f}^b$. To apply the equation 2.49 the accelerometer outputs must be convert to navigation frame. Therefore $C_b^n$ matrix is used. This matrix propagates as indicated in equation 2.19. In equation 2.19 $\Omega_{nb}^b$ is the skew symmetric matrix of $\boldsymbol{\omega}_{nb}^b$, the body rate with respect to navigation frame. The derivation of $\boldsymbol{\omega}_{nb}^b$ is,

$$\boldsymbol{\omega}_{nb}^b = \boldsymbol{\omega}_{ib}^b - C_n^b[\boldsymbol{\omega}_{ie}^n + \boldsymbol{\omega}_{en}^n] \tag{2.50}$$

In this representation,

$\boldsymbol{\omega}_{ib}^b$ is the measured body rates in body frame $\boldsymbol{\omega}_{ib}^b = [p\ q\ r]^t$.

$\boldsymbol{\omega}_{ie}^n$ is the Earth rate with respect to inertial frame in navigation frame.

$\boldsymbol{\omega}_{en}^n$ is the turn rate of navigation frame with respect to Earth frame in navigational frame.

Now let expand the terms in equation 2.49. Firstly $\boldsymbol{v}_e^n$ represents the velocity with respect to Earth in local geographic frame. The components of $\boldsymbol{v}_e^n$ are true north, east and local vertical velocity components

$$\boldsymbol{v}_e^n = \begin{bmatrix} v_n \\ v_e \\ v_d \end{bmatrix} \tag{2.51}$$

$\boldsymbol{f}^n = C_b^n \boldsymbol{f}^b$ is the specific force vector measured by accelerometer and resolved in local geographic frame

$$\boldsymbol{f}^n = \begin{bmatrix} f_n \\ f_e \\ f_d \end{bmatrix} \tag{2.52}$$

$\boldsymbol{\omega}_{ie}^n$ is the turn rate of the Earth in local geographic frame

$$\boldsymbol{\omega}_{ie}^n = \begin{bmatrix} \omega\cos(L) \\ 0 \\ -\Omega\sin(L) \end{bmatrix} \tag{2.53}$$

$\boldsymbol{\omega}_{en}^{n}$ symbolizes the turn rate of the local geographic frame with respect to the Earth-fixed frame. This quantity is also called as transport rate and expressed as,

$$\boldsymbol{\omega}_{en}^{n} = \begin{bmatrix} \dot{l}\cos(L) \\ -\dot{L} \\ -\dot{l}\sin(L) \end{bmatrix} \tag{2.54}$$

In equation 2.53 and 2.54, $L$ and $l$ represent latitude and longitude respectively. Moreover in equation 2.54 $\dot{l} = v_e/(R_0 + h)\cos(L)$ and $\dot{L} = v_n/(R_0 + h)$ where $h$ is height above the surface of Earth and $R_0$ is the mean radius of curvature of Earth. With the given parameters, equation 2.54 will be,

$$\boldsymbol{\omega}_{en}^{n} = \begin{bmatrix} \frac{v_e}{R_0+h} \\ -\frac{v_n}{R_0+h} \\ -\frac{v_e \tan(L)}{R_0+h} \end{bmatrix} \tag{2.55}$$

As a final term, $\boldsymbol{g}_{l}^{n}$ is the local gravity vector,

$$\boldsymbol{g}_{l}^{n} = \boldsymbol{g} - \boldsymbol{\omega}_{ie} \times (\boldsymbol{\omega}_{ie} \times \boldsymbol{R}) = \boldsymbol{g} - \frac{\Omega^2(R_0 + h)}{2} \begin{bmatrix} \sin(2L) \\ 0 \\ 1 + \cos(2L) \end{bmatrix} \tag{2.56}$$

The position determination near the surface of the Earth can be determined by calculating the latitude, longitude and height above the surface of the earth, and the following equations can be used for that purpose.

$$\dot{L} = \frac{v_n}{R_0 + h} \tag{2.57}$$

$$\dot{l} = \frac{v_e \sec(L)}{R_0 + h} \tag{2.58}$$

$$\dot{h} = -v_d \tag{2.59}$$

In velocity and position equations, it is assumed that Earth is perfectly spherical in shape. Additional modifications must be applied to navigation equation. The corrections over the surface of the Earth are referred at the following sections.

## 2.5 Earth Reference Model

### 2.5.1 Gravity Model

As known from other sections accelerometers give only specific forces which are caused by external forces. In order to get the precise estimates of true acceleration values it is needed to model the gravitational field in the vicinity of the Earth. Therefore precise positioning analysis can be sustained.

There are various models in literature to calculate gravity field variations [4]. In this study, the following expression for the variation of gravitational field will be utilized [1]. In this equation the variation is expressed with the latitude and height change above ground.

$$g(0) = 9.780318(1 + 5.3024 \times 10^{-3} \sin^2(L) - 5.9 \times 10^{-6} \sin^2(2L))\ m/s^2 \quad (2.60)$$

$$\frac{dg(0)}{dh} = -0.0000030877(1 - 1.39 \times 10^{-3} \sin^2(L))\ m/s^2/m \quad (2.61)$$

In most of the equations, it is sufficient to assume that the variation of gravity with altitude is as follows,

$$g(h) = \frac{g(0)}{(1 + h/R_0)^2} \quad (2.62)$$

### 2.5.2 Earth Shape Approximation

In order to apply the navigation equations, Earth shape model is assumed to be spherical. However for the accurate positioning spherical model is not sufficient to represent the accurate navigation. Due to flattening of the Earth at the poles, modelling the Earth as ellipsoid gives better result than spherical model. In ellipsoid model some parameters must be defined. These parameters are given in Table 2.1 The rates of change of latitude and longitude in equations 2.57 and 2.58 the mean radius of curvature of the Earth can be found as,

$$R_0 = (R_E\ R_N)^{1/2} \quad (2.63)$$

Table 2.1: WGS-84 Earth model [1]

| | | |
|---|---|---|
| Semi-major axis | $R$ | = 6378137.0 m |
| Semi-minor axis | $r = R(1-f)$ | = 6356752.3142 m |
| Flattening | $f = (R-r)/R$ | = 1/298.257223563 |
| Major eccentricity | $e = [f(2-f)]^{1/2}$ | = 0.0818191908426 |
| Earth's rate | $\Omega$ | = 7.292115e-5 rad/s |

where

$$R_N = \frac{R(1-e^2)}{(1-e^2\sin^2(L))^{3/2}} \tag{2.64}$$

$$R_E = \frac{R}{(1-e^2\sin^2(L))^{1/2}} \tag{2.65}$$

$R_N$ and $R_E$ are the meridian radius of curvature and a transverse radius of curvature, respectively.

Moreover, the distinction between geocentric and geodetic latitude must be indicated in here. Geocentric latitude is the angle between the equatorial plane and a line passing through the center of the Earth and surface location point. Geodetic latitude is the angle between equatorial plane and a line normal to the reference ellipsoid surface at desired point. In the discussion of navigation solutions geodetic latitude will be considered. In Figure 2.8, geocentric and geodetic latitude are shown.

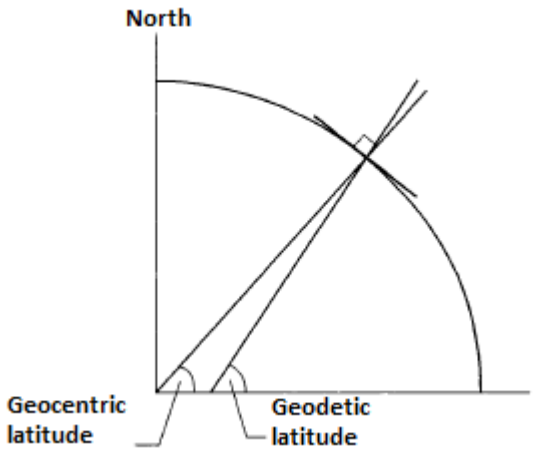



Figure 2.8: Geocentric and geodetic latitude [1]

## 2.6 Digital Integration Form of Navigation Equation

In other sections the analytic solutions of the navigation equations are solved. In this section discrete time implementation of the navigation equations will be discussed. The important processing tasks in real time implementations are the attitude computations and specific force resolution. In high frequency motions, these computations take essential computational part in modern processors.

In following sections these main processing parts attitude computation and specific force resolution with navigation equations will be presented.

### 2.6.1 Attitude computation

The conventional approach to attitude determination is to compute direction cosine matrix. With the help of direction cosine matrix, vehicle body reference frame can be related to reference co-ordinate frame using numerical integration technique. Theoretically it is possible to compute body attitude accurately in the presence of high frequency motion provided that the computational frequency is sufficiently high.

The implementation of attitude computation is based on the direction cosine algorithm in this study. In order to update $C_b^n$ from time $t_k$ to $t_{k+1}$, the following update algorithm form can be used [1]. Note that for the simplicity $C_b^n$ is shown as $C$ only.

$$C_{k+1} = C_k \; e^{(\int_{t_k}^{t_{k+1}} \Omega \, dt)} \tag{2.66}$$

As far as turn rate vector $\boldsymbol{\omega}$ remains fixed over the update interval, the following integration is true.

$$\int_{t_k}^{t_{k+1}} \Omega \, dt = [\boldsymbol{\sigma}\times] \tag{2.67}$$

Where

$$\boldsymbol{\sigma}\times = \begin{bmatrix} 0 & -\sigma_z & \sigma_y \\ \sigma_z & 0 & -\sigma_x \\ -\sigma_y & \sigma_x & 0 \end{bmatrix} \tag{2.68}$$

However calculating the updates of direction cosine matrix by using fixed $\boldsymbol{\omega}$ vector

over update interval is an accuracy limitation in orientation calculation. In general, $\boldsymbol{\omega}$ does not remain fixed in space, then following expression explains this situation.

$$\dot{\boldsymbol{\sigma}} = \boldsymbol{\omega} + \frac{1}{2}\boldsymbol{\sigma} \times \boldsymbol{\omega} + \frac{1}{\sigma^2}\left[1 - \frac{\sigma \sin(\sigma)}{2(1 - \cos(\sigma))}\right]\boldsymbol{\sigma} \times \boldsymbol{\sigma} \times \boldsymbol{\omega} \tag{2.69}$$

A simplified version of equation 2.69 can be written by expanding the cosine and sine terms with their series expansion forms and removing higher order terms,

$$\dot{\boldsymbol{\sigma}} = \boldsymbol{\omega} + \frac{1}{2}\boldsymbol{\sigma} \times \boldsymbol{\omega} + \frac{1}{12}\boldsymbol{\sigma} \times \boldsymbol{\sigma} \times \boldsymbol{\omega} \tag{2.70}$$

Also following algorithm is a simpler version of equation 2.70. The rotation of turn rate vector can be expressed in a much simpler form by calculation rotation difference between time $t_k$ and $t_{k+1}$. In reference [4] it is proposed as,

$$\boldsymbol{\delta\alpha}_{k+1} = \int_{t_k}^{t_{k+1}} \boldsymbol{\alpha} \times \boldsymbol{\omega}\, dt \tag{2.71}$$

Where turn rate vector integration and $\boldsymbol{\sigma}$ update terms can be written as,

$$\boldsymbol{\alpha} = \int_{t_k}^{t_{k+1}} \boldsymbol{\omega}\, dt \tag{2.72}$$

$$\boldsymbol{\sigma} = \boldsymbol{\alpha}_{k+1} + \boldsymbol{\delta\alpha}_{k+1} \tag{2.73}$$

After calculation of $\boldsymbol{\sigma}$, the propagation of transformation matrix which is equation 2.66 becomes,

$$\begin{aligned} C_{k+1} &= C_k\, e^{[\boldsymbol{\sigma}\times]} \\ &= C_k\, A_k \end{aligned} \tag{2.74}$$

Here $C_k$ represents the direction cosine matrix at time $k$ and the transformation matrix $A_k$ transforms a vector at body coordinates at time $k$ to body coordinates at time $k+1$. The variable $\boldsymbol{\sigma}$ is an angle vector with magnitude of $\sigma$ and will rotate the body orientation matrix at time $k$ to $k+1$. The components of $\boldsymbol{\sigma}$ are listed as $\sigma_x$, $\sigma_y$, $\sigma_z$ and the magnitude is

$$\sigma = \sqrt{(\sigma_x^2 + \sigma_y^2 + \sigma_z^2)} \tag{2.75}$$

In equation 2.74 if the exponential term is expanded, it can be used in numerical calculations easily.

$$A_k = I + [\boldsymbol{\sigma}\times] + \frac{[\boldsymbol{\sigma}\times]^2}{2!} + \frac{[\boldsymbol{\sigma}\times]^3}{3!} + \frac{[\boldsymbol{\sigma}\times]^4}{41} + \ \ldots \tag{2.76}$$

and if equation 2.68 is inserted in each $[\boldsymbol{\sigma}\times]$ term,

$$[\boldsymbol{\sigma}\times]^2 = \begin{bmatrix} -(\sigma_y^2+\sigma_z^2) & \sigma_x\sigma_y & \sigma_x\sigma_z \\ \sigma_x\sigma_y & -(\sigma_x^2+\sigma_z^2) & \sigma_y\sigma_z \\ \sigma_x\sigma_z & \sigma_y\sigma_z & -(\sigma_x^2+\sigma_y^2) \end{bmatrix} \tag{2.77}$$

$$[\boldsymbol{\sigma}\times]^3 = -(\sigma_x^2+\sigma_y^2+\sigma_z^2)\,[\boldsymbol{\sigma}\times] \tag{2.78}$$

$$[\boldsymbol{\sigma}\times]^4 = -(\sigma_x^2+\sigma_y^2+\sigma_z^2)\,[\boldsymbol{\sigma}\times]^2 \tag{2.79}$$

$$\vdots$$

Therefore equations 2.76 can be simplified as in equation 2.80 by using series expansion form,

$$A_k = I + \left[1 - \frac{\sigma^2}{3!} + \frac{\sigma^4}{5!} - \dots\right][\boldsymbol{\omega}\times] + \left[\frac{1}{2!} - \frac{\sigma^2}{4!} + \frac{\sigma^4}{6!} - \dots\right][\boldsymbol{\omega}\times]^2 \tag{2.80}$$

Which can be written as follows,

$$A_k = I + \frac{\sin(\sigma)}{\sigma}\,[\boldsymbol{\omega}\times] + \frac{(1-\cos(\sigma))}{\sigma^2}\,[\boldsymbol{\omega}\times]^2 \tag{2.81}$$

Therefore direction cosine matrix can be updated to find body orientation by using gyroscope readings. The recursive algorithms from equation 2.74 to 2.81 can be used for that purpose. To find detailed equation about attitude update reference [7] can be used.

### 2.6.2 Orthogonalization and Normalization

The rows of direction cosine matrix represent the projection of unit vectors which lie along the reference coordinate axis. However during the computations of strapdown attitude algorithm the orthogonality property may be corrupted. To eliminate this deterioration self-consistency checks should be realized. This process can be applied by two step, orthogonalization and normalization.[8]

**Orthogonalization:**

An orthogonal matrix can be described as the orthonormal behaviour between its rows. That is their dot product must be equal to zero. To sustain this dot product property, Gram-Schmidt algorithm can be applied to direction cosine matrix. Let the

direction cosine matrix be ,

$$\hat{C} = \begin{bmatrix} \boldsymbol{r}_1 \\ \boldsymbol{r}_2 \\ \boldsymbol{r}_3 \end{bmatrix} \tag{2.82}$$

where $\boldsymbol{r}_1$,$\boldsymbol{r}_2$, and $\boldsymbol{r}_3$ symbolize the rows of DCM. Then the orthogonalization of the DCM matrix is,

$$\hat{C}_{ort} = \begin{bmatrix} \boldsymbol{r}_1 \\ \boldsymbol{r}_2 - \frac{\boldsymbol{r}_2 \boldsymbol{r}_1^t}{\|\boldsymbol{r}_1\|^2} \boldsymbol{r}_1 \\ \boldsymbol{r}_3 - \frac{\boldsymbol{r}_3 \boldsymbol{r}_1^t}{\|\boldsymbol{r}_1\|^2} \boldsymbol{r}_1 - \frac{\boldsymbol{r}_3 \boldsymbol{r}_{ort,2}^t}{\|\boldsymbol{r}_{ort,2}\|^2} \boldsymbol{r}_{ort,2} \end{bmatrix} \tag{2.83}$$

**Normalization:**

In normalization process, the aim is making the magnitude of the rows of direction cosine matrix value unity. To realize the unity equation 2.84 can be used.

$$\hat{C}_{nor} = \begin{bmatrix} \frac{\boldsymbol{r}_{ort,1}}{\|\boldsymbol{r}_{ort,1}\|} \\ \frac{\boldsymbol{r}_{ort,2}}{\|\boldsymbol{r}_{ort,2}\|} \\ \frac{\boldsymbol{r}_{ort,3}}{\|\boldsymbol{r}_{ort,3}\|} \end{bmatrix} \tag{2.84}$$

With these corrections the deformations of the DCM as a result of discrete time calculations are eliminated. Therefore accurate solutions for orientation matrix can be obtained.

### 2.6.3 Acceleration transformation and Navigation Algorithm

The analytic solutions of the navigation equations are given in section 2.4.1. Remember the velocity and position formulation in integral form,

$$\boldsymbol{v}_e^n = \int_0^t \boldsymbol{f}^n \, dt - \int_0^t [2\boldsymbol{\omega}_{ie}^n + \boldsymbol{\omega}_{en}^n] \times \boldsymbol{v}_e^n \, dt + \int_0^t \boldsymbol{g}_l^n \, dt \tag{2.85}$$

Position formulation,

$$\boldsymbol{x}^n = \int_0^t \boldsymbol{v}^n \, dt \tag{2.86}$$

Numerically it is required to calculate the integral terms to find velocity and position. The first integral term in velocity equation 2.85 represents the updated velocity changes. Let represent the velocity increment in discrete form. Then first integral will be,

$$\boldsymbol{u}^n = \int_{t_k}^{t_{k+1}} \boldsymbol{f}^n \, dt \tag{2.87}$$

where $\boldsymbol{u}_n$ represents the change in velocity, resolved in navigation frame. In equation 2.87 the effect of changing the matrix $C_b^n$ is also added. Because measurements of acceleration in body axes, it must be transformed into navigation axes. In attitude computation section, the variation of attitude has been explained. If attitude update equations is used in equation 2.87,

$$\boldsymbol{u}^n = C_k \int_{t_k}^{t_{k+1}} A_k \boldsymbol{f}^b \, dt \tag{2.88}$$

Expression for $A_k$ can be found in 2.80. The important point in here, attitude update time must be high enough to take into account of vehicle dynamics. The velocity update equations then,

$$\boldsymbol{v}_{k+1}^n = \boldsymbol{v}_k^n + \boldsymbol{u}^n - [2\Omega_{ie} + \Omega_{en}]\delta t + \boldsymbol{g_l}\delta t \tag{2.89}$$

Where

$$\begin{aligned} \Omega_{ie} &= [\boldsymbol{\omega}_{ie}\times] \\ \Omega_{en} &= [\boldsymbol{\omega}_{en}\times] \\ \boldsymbol{v}_k^n &= \text{velocity at time } t_k \\ \delta t &= t_{k+1} - t_k \end{aligned}$$

Finally the position vector can be integrated. The integration choice is dependent on the application. Occasionally, rectangular or trapezoidal integration techniques are sufficient for navigation integration algorithm.

Rectangular integration:

$$\boldsymbol{x}_{k+1}^n = \boldsymbol{x}_k^n + \boldsymbol{v}_k^n \delta t \tag{2.90}$$

Trapezoidal integration:

$$\boldsymbol{x}_{k+1}^n = \boldsymbol{x}_k^n + (\frac{\boldsymbol{v}_k^n + \boldsymbol{v}_{k+1}^n}{2})\delta t \tag{2.91}$$

For higher demands in accuracy in navigation computation, a higher order integration techniques should be used. Such as Runge-Kutta method, Simpsons's rule,etc. For more comprehensive velocity and position update equations reference [9] can be utilized.

In this thesis, to build navigation equation algorithm equations between 2.41 and 2.59 are used for integration of acceleration values to obtain nort-east-down velocity

components and latitude, longitude an altitude position components. For discrete integration of attitude computation equations between 2.66 and 2.81 are utilized. And to sustain the orthogonalization property of transformation matrix equations between 2.82 and 2.84 are used. The corresponding simulink model is given in chapter 6.

# CHAPTER 3

# INERTIAL NAVIGATION AND AIDED SENSOR SYSTEMS

## 3.1 Introduction

This chapter provides the basis knowledge about the inertial navigation system components and some aided navigation devices in literature. By learning the structure of the navigation system, the nature and principle of navigation operation can be understood easily. In this chapter, firstly the inertial measurement unit is considered. Based on the inertial measurement unit structure, accelerometer and gyroscope technology will be covered. As for the aided navigation devices, GPS and magnetometer measurement devices will be discussed. Understanding of these systems is an important step for the INS/GPS integration process.

## 3.2 Inertial Measurement Unit

Inertial measurement unit is used to determine the state of the system in three dimensional space. IMU includes three axes gyroscope and three axes accelerometer which are orthogonally mounted on the platform.

The accelerometers and gyroscopes are used to sense the linear acceleration and angular rate change of the system, respectively. For the gyroscope, by integrating the angular velocity with the given initial conditions, angular orientation of the system can be obtained. For the accelerometer, by integrating the measurement once the velocity and at the second integration the position of the system can be obtained. Of course the initial values of the velocity and position values should be known.[2]

Inertial measurement units can be classified into two groups. These are called as gimbal systems and strapdown systems. These concepts are briefly explained in chapter 2. Based on these systems, especially strapdown application the primary concern is implementation of full inertial navigation system[1]. The relationship between INS system blocks are shown in Figure 3.1.

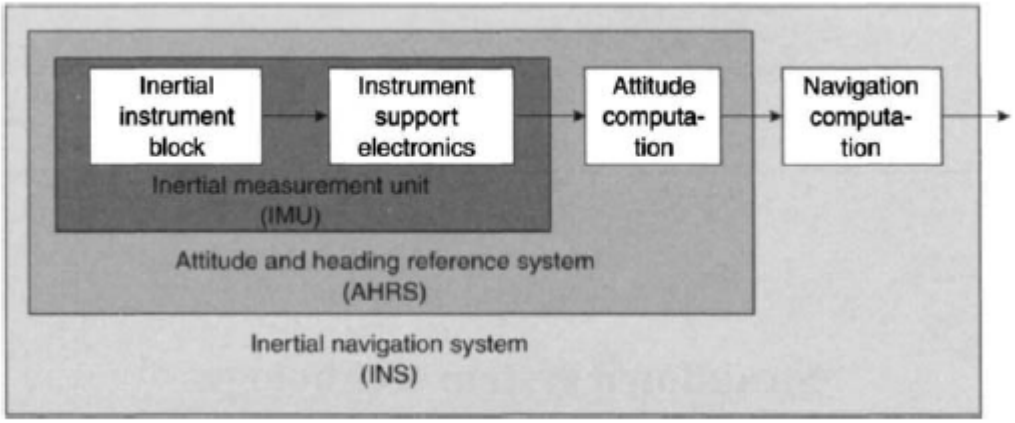



Figure 3.1: Strapdown inertial navigation system blocks [1]

As shown in figure, the instrument cluster and electronic blocks are parts of IMU. For attitude and heading information, IMU process is integrated with attitude equations solvers. The resulting system is called as attitude and heading reference system (AHRS). As a final computation process, if the navigation equations solvers are added to system, a full navigational capacity can be achieved and INS system is obtained.

## 3.3 Accelerometer and Gyroscope Technology

### 3.3.1 Accelerometer

Translational accelerations of a rigid body, resulting from the forces acting on vehicle are described by the Newton's second law of motion. However measuring total force to determine acceleration is not practical. On the other hand, within vehicle by using the small mass and measuring force on it, it is possible to determine acceleration. This small mass forms the main part of an instrument called

accelerometer. An accelerometer consists of a proof mass $m$, and a pair of springs as shown in Figure 3.2. The total force on accelerometer can be represented as,

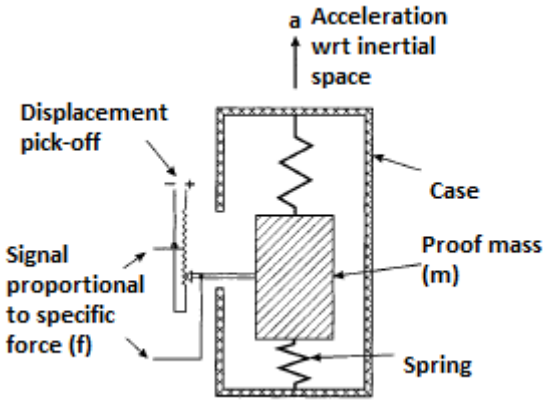



Figure 3.2: Accelerometer with proof mass [1]

$$F = ma = m(f + g) \tag{3.1}$$

Here $f$ is the acceleration from external force other then gravitational force and $g$ represents the gravitational acceleration. Under steady state conditions mass will be balanced by the spring tension. When there is an acceleration along the sensitive axis, proof mass displaces with respect to body. And the net extension will give a measure of the applied force. An accelerometer is insensitive to the gravitational acceleration. For example, an accelerometer which is falling freely within gravitational field, shows no extension on spring and gives result as zero. If the accelerometer stays as stationary, accelerometer will measure the force which stops from falling down. Hence, $a = 0$ and $f = -g$ in that case. It is obvious that the gravitational field is essential to enable navigation computation correctly.[1]

Mechanical based accelerometers mainly measure specific force in a manner analogous to the simple spring and mass system. Today various principles are developed to measure acceleration. Especially MEMS based accelerometers are commonly used in navigational technology.

### 3.3.2 Accelerometer Error Model

Accelerometers are subjected to errors which play an important role in accuracy of accelerometer. The major sources of error are listed in below [1].

**Fixed Bias:** It is defined as the output of the sensor when there is zero input. The size of bias error or displacement error is independent from of any motion. The unit of bias is usually expressed in milli-g or micro-g depending on the devices.

**Scale-factor errors:** Scale factor can be expressed as the deviation from least-squares straight line. This line relates the output signal to applied acceleration. There is no specific unit for scale factor error. Scale factor is stated as ratio parts per million or percentages of measured full scale quantity.

**Cross-Coupling Errors:** This error is resulting from accelerometer sensitivity to other normal axes. This kind of errors arise from manufacturing imperfections which cause the non-orthogonality in sensor axes. Cross-coupling error is often expressed as a percentage of applied acceleration.

**Vibro-Pendulous Errors:** When sensor is subjected to vibratory motion in pendulous accelerometer, the phasing between vibration and pendulum movements causes error in measurement. This error can be expressed in units of $g/g^2$.Besides this type of sensors, temperature dependent errors, in-run errors, and residual errors can cause the divergence in the output of accelerometer.

Generalised accelerometer error model for simulation purposes can be expressed as in below. It is assumed that the sensors are mounted with their sensitive axes aligned with principal axis of host vehicle [1].

$$\begin{bmatrix} \delta f_x \\ \delta f_y \\ \delta f_z \end{bmatrix} = B_A + B_V \begin{bmatrix} a_y a_z \\ a_z a_x \\ a_x a_y \end{bmatrix} + S_A \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} + M_A \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} + w_A \tag{3.2}$$

In this equation

$B_A$ is a three element fixed-bias

$B_V$ is a $3 \times 3$ matrix for vibro-pendulous error

$S_A$ is diagonal scale-factor matrix

$M_A$ is skew symmetric matrix for misalignment or cross coupling errors

$w_A$ is three-element vector white noise

The measurement of the specific force values can be written as follows,

$$\tilde{f}_x = f_x + \delta f_x \tag{3.3}$$

$$\tilde{f}_y = f_y + \delta f_y \tag{3.4}$$

$$\tilde{f}_z = f_z + \delta f_z \tag{3.5}$$

In thesis, after modelling these error, in simulation part these error values are sum up with each individual body axes acceleration values which are obtained from TAI helicopter simulator. Therefore the modelled raw accelerometer data can be obtained.

### 3.3.3 Gyroscope

To establish the motion of vehicle in three dimension, rotational motion and translational motion should be measured. Gyroscopes are sensors that measure angular rates with respect to inertial reference frame. If these angular rates are integrated, sensor measurements will give change in angular orientation.

Gyroscopes use the inertial property of high speed rotor spinning. A spinning rotor tends to maintain the direction of its axis by using its angular momentum. The phenomenon is called as gyroscopic inertia. This fundamental operation defines the direction of rotation which remains fixed in space. The fixed direction enables rotation to be determined. [1].

By using different construction techniques, the classes of gyroscope sensors are presented as below.

- Magneto-hydrodynamic rate sensor
- Vibratory gyroscopes

- Optical rate sensors including ring laser gyroscopes (RLGs) and fibre optic gyroscopes (FOGs)
- Micro-machined electromechanical systems (MEMS) gyroscopes

Because all such devices provide angular rotation, they are called as gyroscope. However notice that these devices do not depend on rotation body motion.

### 3.3.4 Gyroscope Error Model

The accuracy of the gyroscopes are limited because of the sensor errors. Some constructional imperfections cause the diverging the sensor outputs. The major sources of the errors in gyroscopes are listed below [1].

**Fixed Bias:** As in accelerometer, fixed bias corresponds to fixed value even in the absence of applied motion. Various effects cause this error, such as temperature gradients, residual torques, etc. It is usually expressed in units of degrees per hour $^\circ/h$.

**$g$-Dependent Bias:** This kind of bias is proportional to applied acceleration. Such errors come with rotor unbalance in spinning. The relationship between accelerometer and bias is expressed in units of $^\circ/h/g$.

**Anisoelastic Bias($g^2$-dependent bias):** This is another bias type which is proportional to the product of acceleration along orthogonal pair of axes. Such biases are caused by gyroscope rotor suspension structure. The anisoelastic bias has units of $^\circ/h/g^2$

**Anisoinertia Errors:** This kind of error is consequence of the inequalities in gyroscope moments of inertia about different axes. The biases are proportional to the product of angular rates applied to pairs of orthogonal axes. The unit is $^\circ/h/(rad/s)^2$

**Scale-Factor Errors:** This error is the ratio between the change in the output signal to a change in the input rate. Scale factor is expressed as a ratio of output to input rate in parts per million (ppm), or as percentage for some type of gyroscope models.

Scale factor is also represented as the deviations from fitted non-linear function which relates the output signal to applied angular rate.

**Cross-Coupling Errors:** This error results from gyroscope sensitivity to turn rates about axes normal to the input axes. Such errors come with the non-orthogonality of the sensor axes. The representation is parts per million or percentage in some gyroscope models.

**Angular Acceleration Sensitivity:** This error is also known as the gyroscopic inertial error. Due to inertia of the rotor, this kind of error is encountered. This error increases with increasing frequency of the motion.

Some or all of the described errors generally are encountered in gyroscope sensors. The generalized gyroscope sensor error model can be written as,

$$\begin{bmatrix} \delta\omega_x \\ \delta\omega_y \\ \delta\omega_z \end{bmatrix} = B_G + B_g \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} + B_{ae} \begin{bmatrix} a_y a_z \\ a_z a_x \\ a_x a_y \end{bmatrix} + B_{ai} \begin{bmatrix} \omega_y \omega_z \\ \omega_z \omega_z \\ \omega_x \omega_y \end{bmatrix} + S_G \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} + M_G \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} + w_G \tag{3.6}$$

In this equation,

$B_G$ is a three element residual fixed-bias

$B_g$ is a $3 \times 3$ matrix for $g$-dependent bias

$B_{ae}$ is $3 \times 3$ matrix for anisoelastic coefficients

$B_{ai}$ is $3 \times 3$ matrix for anisoineria coefficients

$S_G$ is a diagonal matrix representing scale-factor

$M_G$ is a $3 \times 3$ skew symmetric matrix represents cross-coupling errors

$w_A$ is three-element vector white noise

The measured rate of the gyroscope then can be written as,

$$\tilde{\omega}_x = \omega_x + \delta\omega_x \tag{3.7}$$

$$\tilde{\omega}_y = \omega_y + \delta\omega_y \tag{3.8}$$

$$\tilde{\omega}_z = \omega_z + \delta\omega_z \tag{3.9}$$

In thesis, after modelling these error, in simulation part these error values are sum up with each individual body axes angular rate values which are obtained from TAI helicopter simulator. Therefore the modelled raw angular rate data can be obtained. For accelerometer and gyroscope similar error model also can be found in reference [10].

## 3.4 Global Positioning System (GPS)

The global positioning system (GPS) was developed by US Department of Defense in 1970s to serve military requirements. It is based on a satellite network which includes at least 24 satellite. These satellites are orbiting around the world at the altitude of 26,560 km [2]. Global positioning system is a space-based radio navigation system. And this system is developed by Navstar GPS Joint Program Office. GPS gives the position and velocity values to user and all users around the world can access this information with GPS receiver [11].

The operation principle of GPS is to determine velocity and position by using radio signals which are broadcasting from satellites. These signals consist of both a pseudo-random noise code (PRN) and a navigation message [12]. The PRN code is used to calculate transmit time by the receiver. By using the transmit time and speed of light, the range (pseudo-range) between satellites and receiver can be computed. Also navigation message consists of the location of satellite. By using these knowledges from at least four satellite the position of the receiver can be determined. [2]

A master station in Colorado Springs watches the health status of the system using information from 12 monitoring stations around the globe. Therefore the accuracy of the satellites signals is ensured.

### 3.4.1 GPS Architecture

GPS structure is composed of a space segment, a control segment and a user segment.

**Space Segment:** Space segment of GPS structure includes a constellation of 24 satellites minimum. Today this number reaches to 32 satellites. These satellites are

orbiting the Earth nearly in circular orbits. GPS satellites occupy six orbits inclined at 55°to the equator. Each orbit includes four primary satellites which are distributed unevenly. The orbital period is about 12 hour [12]. The orbital design accommodate more than 30 satellites to ensure at least four satellite visible to user. To estimate the position and velocity of the user these satellites broadcast radio signals containing coded information. Each satellite transmits two carrier frequencies in L band, known as $L_1$ (1575.42MHz) and $L_2$ (1227.6MHz). Each of these signals is modulated by precise positioning service (PPS or P) and/or standard positioning service (SPS) also known as coarse/acquisition, C/A. The P-code and C/A are added to $L_1$ signal however only P-code is given in $L_2$ signal. As the name suggests, the PPS code is used for full precise navigation solution and it is only available for selected users. SPS signal is available to all users [2].

In order to determine the position and velocity, accurate timing is important. Because position is calculated by using the distance from each satellite and this distance from each satellite is computed by measuring the passing time between transmitting and receiving of the radio signal. The satellites have accurate atomic clock, however receivers have less accurate clocks. The errors in calculation of timing and distance are corrected by observing minimum of four satellites. Three for spatial coordinates and one for time.

**Control Segment:** The control segment is composed of the master control station, monitor station and ground antennas. The control segment maintains the orbital configuration of satellites and signal broadcast. Especially the corrections for ephemerides, almanac and other control parameters are sustained from control section. These corrections are sent to satellite constellation once per day. Satellite signals are tracked by six US Air Force and 11 national geospatial-intelligence agency (NGA) monitoring stations around the globe [12]. These unmanned stations are controlled by the master control station (MCS) and they observe the satellite integrity. By giving this information to MCS, the ephemerides and clock parameters are corrected [2].

**User Segment:** GPS receiver and receiver antenna are the user segment part of GPS. The radio signals from satellites are received by antenna and converted to electric

signals. Receivers process these signals to obtain position and velocity of the vehicle. The receivers decode the satellites signals and measure the transit time thereby determine the range between each satellite and the receiver. By using the navigation data receivers also determine the position of each satellite at the time signal were transmitted[13].

### 3.4.2 GPS Error Sources

The ranging measurement in GPS is affected by errors arising from variety of sources. To obtain an accurate solution these errors must be eliminated or minimized. Some of these GPS errors are given below [2],[14].

**Satellite Clock Error**

GPS system satellite clock drifts in time. The control segment of the GPS system adds correction parameter to navigation signal and upload the satellites. Therefore receiver can correct the clock error during position calculation.

**Receiver Clock Error**

As stated previously, receiver clocks are much less accurate than the satellite clocks and contain a bias value. This clock bias data is eliminated by using at least four satellite navigation signal. Thus the clock bias error can be estimated.

**Ionosphere Delay**

Ionosphere contains ionized gases and these ionized gas layer is changing the transit time of GPS signal. Satellite elevation also affects ionosphere delay. Signals from low elevation satellites pass a greater range distance through ionosphere those at higher elevations. Dual frequency GPS receivers which obtain both $L_1$ and $L_2$ signals are able to correct ionosphere delay more accurately. A single frequency receivers depend on Klobuchar model to calculate ionospheric delay.

**Tropospheric Delay**

Unlike the ionosphere, troposphere is composed of $N_2$ and $O_2$ and water vapor. Troposphere is electrically neutral but its reflective property causes a decreases in

speed relative to free space. Tropospheric delay has a dry and wet components. The wet components are difficult the model because of the water vapor content varies locally in troposphere. Despite of wet model, dry model is easy to model. Several models are used such as, Hopfield model and Chao model [2].

**Multipath Errors**

In urban environment the major source of error is multipath error. GPS signal follow different path to reach the receiver. These paths include direct line of sight and reflected signals from other objects. The indirectly arriving signals are delayed and have lower signal to noise ratio. Multipath error causes the distortion in receiving signal and position error can be 10 meters or more depends on the error value.

**Satellite Orbital Errors**

In ephemeris data the satellite position is send to receiver. However the actual position of the satellite can be different from the ephemeris data. The orbital errors are determined by the control segment and uploaded to the satellites for broadcast to the users as ephemeris data.

**Receiver Noise**

This error is a random noise error because of the electronics of a GPS receiver. With the effect of antenna circuitry, cables, thermal noises, signal quantization and sampling these errors cumulatively affect the measurement. Since it is a function of the signal to noise ratio, receiver noise varies with the elevation angle of a satellite.

### 3.4.3 GPS Satellite Orbits

The orbit of satellites is described with the Kepler's laws. Before going into details, it is important to have basic knowledge about Kepler's planetary motion. According The Kepler's law,

- A planet travels along an elliptical orbit with the sun standing at one of the foci of the ellipse.
- A line between the sun and a planet sweeps out equal areas in equal times.

- The square of the period of revolution of a planet is proportional to the cube of its mean distance from the sun.

Based on these laws, the satellite motion can be characterized by a fixed elliptical orbit with the Earth being at one of the foci. The orbit is specified by six parameters. These Keplerian parameters are listed below. And the illustration of these parameters are given in Figure 3.3

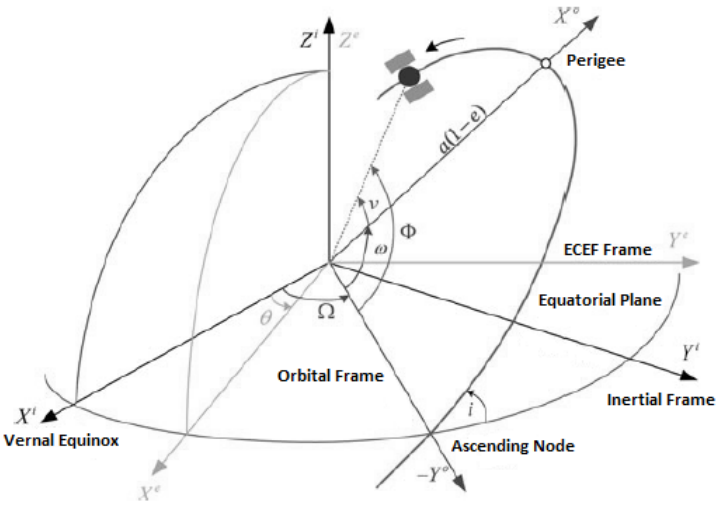



Figure 3.3: Elements of Keplerian Parameters and orbital frames [2]

1. *Eccentricity* [e]: A measure of deviation from perfect circle.
2. *Inclination* [i]: The angle of orbital plane relative to Earth's equatorial plane.
3. *Semimajor axis* [a]: It is one half of the longest diameter. Expressed as from center to the edge of ellipse.
4. *Right Ascension of the Ascending Node(RAAN)* [$\Omega$]: The ascending node is the point where the orbit crosses the equatorial plane of Earth while the satellite is moving in the positive z direction of the ECI and ECEF frames in other words from south to north.
5. *Argument of Perigee* [$\omega$]: The angle in the plane of the orbit between the ascending node and the point on the orbit closest to the center of Earth (perigee).

6. *True Anomaly* [$\nu$]: The angle between perigee and the satellite at a particular time. The sum of true anomaly and the argument of perigee is equal to another parameter called argument of latitude $\Phi$.

$$\Phi = \omega + \nu \tag{3.10}$$

The true anomaly specifies the location of satellite in orbit. The true anomaly does not vary at a constant rate over the orbit. Therefore instead of using true anomaly, mean anomaly is used which varies with constant rate. To define constant rate parameter, eccentric anomaly and the mean anomaly must be defined first.

(a) **Eccentric Anomaly:** [E]

The angle between the perigee and the projection of the satellite onto the circle of radius a. The relationship between true anomaly and the eccentric anomaly can be written as,

$$E = 2\tan^{-1}\left[\sqrt{\frac{1-e}{1+e}}\tan(0.5\nu)\right] \tag{3.11}$$

(b) **Mean Anomaly:** [M]

The angle between the perigee and an imaginary satellite that travels in a circular orbit with same focus and period as the actual satellite however the rate of change of position is constant. The mean anomaly can be calculated as,

$$M = E - e\sin(E) \tag{3.12}$$

In Figure 3.4 the symbolization of anomalies are shown.

### 3.4.4 Ephemeris Data Processing

In ideal conditions Keplerian motion describes the satellite position and velocity information. However because of the Earth non-uniform gravitational field and the perturbing forces such as, non-central gravitational force field, gravitational attraction of sun, moon and other planets, solar radiation pressure and atmospheric drag the GPS must be aware of these perturbations.

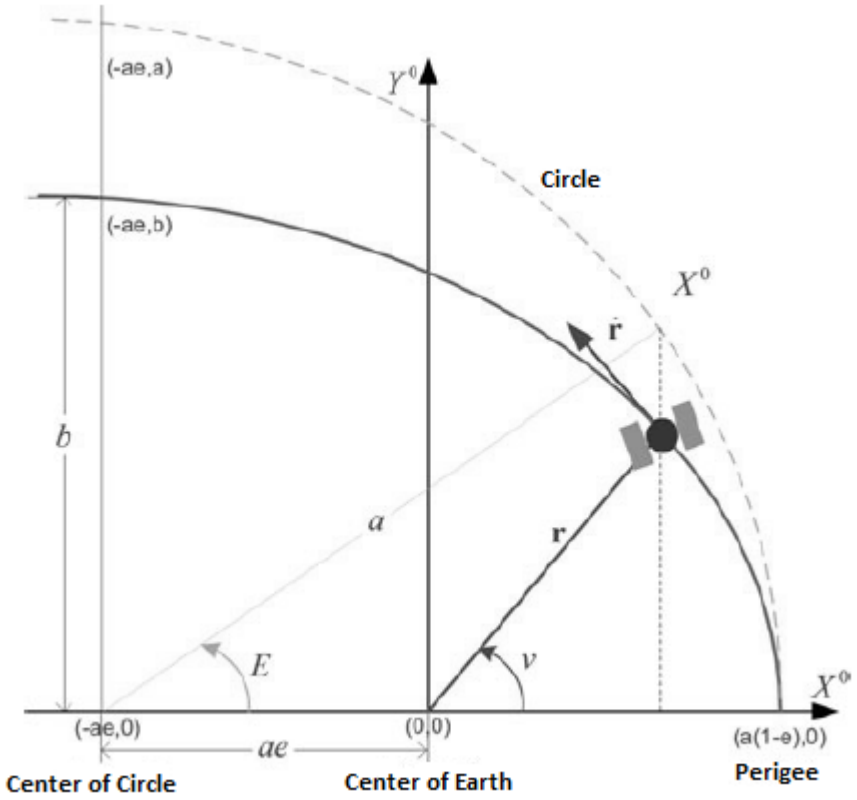



Figure 3.4: The eccentric anomaly and true anomaly[2]

Likewise Keplerian parameters, to explain the these perturbations on satellite 12 parameter are specified relative to reference epoch time. These parameters are known as *ephemerides*. The ephemerides data is broadcasting to the users. The broadcast ephemerides data is typically uploaded to satellites once per day. Satellites broadcast this message every 30s.

The ephemerides parameters are give in Table 3.1. The first six parameters describe elliptical orbit with mean motion and the satellite's motion as a function of time $t_{oe}$. The other parameters describe the deviation of the satellite's actual motion from the smooth elliptical orbit [15]. To be able to correct the measurement by GPS receiver, the position of the satellite must be known with the given corrections in Table 3.1. The position of a satellite is determined based on the orbital parameters in ephemeris data. These parameters are predicted by the master control station on the basis of measurement by monitoring stations. More details of these parameters can be found in IS-GPS-200H (2013).

Table 3.1: Ephemeris parameters of GPS navigation [2]

| | |
|---|---|
| $t_{oe}$ | Ephemeris reference time (sec) |
| $\sqrt{a}$ | Square root of the semimajor axis ($\sqrt{m}$) |
| $e$ | Eccentricity |
| $i_0$ | Inclination angle at the reference time (semicircles) |
| $\Omega_0$ | Longitude of the ascending node of the orbital plane (semicircles) |
| $\omega$ | Argument of perigee (semicircles) |
| $M_0$ | Mean anomaly at the reference time (semicircles) |
| $\Delta n$ | Correction to the computed mean motion (semicircles/sec) |
| $\dot{i}$ | Rate of change of the inclination angle (semicircles/sec) |
| $\dot{\Omega}$ | Rate of change of the RAAN with time (semicircles/sec) |
| $C_{uc}$,$C_{us}$ | Amplitudes of the cosine and sine harmonic correction for the computation of latitude (radians) |
| $C_{rc}$,$C_{rs}$ | Amplitudes of the cosine and sine harmonic correction terms for orbit radius (meters) |
| $C_{ic}$,$C_{is}$ | Amplitudes of the cosine and sine harmonic correction terms for inclination angles (radians) |

**Satellite Position Calculation:**

By using the parameters in Table 3.1, the following method is used to calculate position of satellite,

(**a**) Semi-major axes of the elliptical orbit from ephemeris data

$$a = (\sqrt{a})^2 \tag{3.13}$$

(**b**) Calculate the mean motion of the satellite

$$n_0 = \sqrt{\frac{\mu}{a^3}} \tag{3.14}$$

Where $\mu$ is Earth gravitational constant, the value is $3.986005 \times 10^{14}\ m^3/s^2$

(**c**) Find time $t_k$ which is time since the reference epoch $t_{oe}$ as specified in ephemeris.

$$t_k = t - t_{oe} \tag{3.15}$$

Where $t$ is the GPS system time at the time of transmission. Time $t_k$ should be corrected for the end of week crossover

$$\begin{aligned} &if(t_k > 302,400) \\ &\quad t_k = t_k - 604,800 \\ &elseif(t_k < -302,400) \\ &\quad t_k = t_k + 604,800 \\ &end \end{aligned} \tag{3.16}$$

(**d**) Adjust the mean motion by the correction $\Delta n$ specified in ephemeris

$$n = n_0 + \Delta n \tag{3.17}$$

(**e**) Compute the mean anomaly $M_k$ at time $t_k$

$$M_k = M_0 + nt_k \tag{3.18}$$

Where $M_0$ is the mean anomaly at the reference time.

(**f**) Calculate the eccentric anomaly $E_k$ by solving Kepler's law

$$E_k = M_k + e\sin(E_k) \tag{3.19}$$

where e is the eccentricity of orbit. Normally, the above equation is solved iteratively by setting an initial $E_k = M_k$.

(**g**) Calculation of true anomaly $\nu_k$

$$\nu_k = \tan^{-1}\left(\frac{\sqrt{1-e^2}\sin(E_k/(1-e\cos(E_k)))}{(\cos(E_k-e)/(1-e\cos(E_k)))}\right) \tag{3.20}$$

(**h**) Compute the argument of latitude $\Phi_k$

$$\Phi_k = \nu_k + \omega \tag{3.21}$$

(**i**) Calculation of the three harmonic perturbations. The argument of latitude correction is,

$$\delta u_k = C_{us}\sin(2\Phi_k) + C_{uc}\cos(2\Phi_k) \tag{3.22}$$

The radius correction is,

$$\delta r_k = C_{rs}\sin(2\Phi_k) + C_{rc}\cos(2\Phi_k) \tag{3.23}$$

The inclination correction is,

$$\delta i_k = C_{is}\sin(2\Phi_k) + C_{ic}\cos(2\Phi_k) \tag{3.24}$$

(**j**) Compute the corrected argument of latitude

$$u_k = \Phi_k + \delta u_k \tag{3.25}$$

(**k**) Compute the corrected radius,

$$r_k = a(1 - e\cos(E_k)) + \delta r_k \tag{3.26}$$

(**l**) Calculate the corrected inclination

$$i_k = i_0 + \delta i_k + \dot{i}\, t_k \tag{3.27}$$

(**m**) Calculate the position of satellite in orbital plane,

$$x'_k = r_k\cos(u_k) \tag{3.28}$$

$$y'_k = r_k\sin(u_k) \tag{3.29}$$

(**n**) Compute the corrected longitude of the ascending node,

$$\Omega_k = \Omega_0 + (\dot{\Omega} - \dot{\Omega}_e)t_k - \dot{\Omega}_e t_{oe} \tag{3.30}$$

where $\dot{\Omega}_e$ is the Earth rotation rate.

(**o**) Finally compute the position of satellite in e-frame

$$x_k = x'_k\cos(\Omega_k) - y'_k\cos(i_k)\sin(\omega_k) \tag{3.31}$$

$$y_k = x'_k\sin(\Omega_k) + y'_k\cos(i_k)\cos(\omega_k) \tag{3.32}$$

$$z_k = y'_k\sin(i_k) \tag{3.33}$$

Now the velocity of the satellite can be computed by taking the time derivative of the position. The steps are given below.

(**a**) Calculate the rate of change of eccentric anomaly

$$\dot{E}_k = \frac{n}{1 - e\cos(E_k)} \tag{3.34}$$

(**b**) Calculate the rate of change of argument of latitude

$$\dot{\Phi}_k = \frac{\sqrt{1-e^2}}{1 - e\cos(E_k)}\dot{E}_k \tag{3.35}$$

(**c**) Compute the rate of change of the corrected argument latitude

$$\dot{u}_k = (1 + 2C_{us}\cos(2\Phi_k) - 2C_{uc}\sin(2\Phi_k))\dot{\Phi}_k \tag{3.36}$$

(**d**) Find the rate of change of the corrected radius

$$\dot{r}_k = 2(C_{rs}\cos(2\Phi_k) - C_{rc}\sin(2\Phi_k))\dot{\Phi}_k + Ae\sin(E_k)\dot{E}_k \tag{3.37}$$

(**e**) Calculate the rate of change of satellite's position in its orbital plane

$$\dot{x}'_k = \dot{r}_k\cos(u_k) - r_k\sin(u_k)\dot{u}_k \tag{3.38}$$

$$\dot{y}'_k = \dot{r}_k\sin(u_k) + r_k\cos(u_k)\dot{u}_k \tag{3.39}$$

(**f**) Compute the rate of change of corrected inclination

$$\frac{di_k}{dt} = 2(C_{is}\cos(2\Phi_k) - C_{ic}\sin(2\Phi_k))\dot{\Phi}_k + \dot{i0} \tag{3.40}$$

(**g**) Calculate the rate of change of the corrected longitude of the ascending node

$$\dot{\Omega}_k = \dot{\Omega} - \dot{\Omega}_e \tag{3.41}$$

(**h**) Differentiate the equations 3.31-3.33 to obtain the velocity of the satellite in ECEF frame.

$$\dot{x}_k = \dot{x}'_k\cos(\Omega_k) - \dot{y}'_k\cos(i_k)\sin(\Omega_k) + y'_k\sin(i_k)\sin(\Omega_k)\frac{di_k}{dt} - y_k\dot{\Omega}_k \tag{3.42}$$

$$\dot{y}_k = \dot{x}'_k\sin(\Omega_k) + \dot{y}'_k\cos(i_k)\cos(\Omega_k) - y'_k\sin(i_k)\sin(\Omega_k)\frac{di_k}{dt} + x_k\dot{\Omega}_k \tag{3.43}$$

$$\dot{z}_k = y'_k\sin(i_k) + y_k\cos(i_k)\frac{di_k}{dt} \tag{3.44}$$

Foregoing equations need on-line ephemeris data to calculate satellite position calculation. Therefore in this thesis, for the modelling of satellite orbits in Matlab-simulink model reference [16] was used. The detailed explanation of satellite constellation and their parameters can be found in reference [16].

### 3.4.5 Receiver Position and Velocity Estimation

The main observables in GPS are pseudo-range and pseudo-range rates to estimate navigation data.

Pseudo-range are obtained by measuring the time difference between the propagation time of GPS signal and receiving time of receiver, then multiplying time difference with speed of light. In calculation of distance the bias error from receiver clock should be considered. This offset is a fourth unknown in addition to the positional components of latitude, longitude and height. Therefore at least four satellites measurements are required to determine these four unknowns [2].

**Position Estimation:**

To obtain de position information, firstly the satellite clock bias error, ionospheric errors and tropospheric errors should be eliminated. Then the corrected pseudo-range is,

$$\rho_c^m = r^m + c\delta t_r \tag{3.45}$$

Where

$\rho_m$ is the measured pseudo-range between $m$th satellite and the receiver in meters.

$r_m$ is the true range between receiver's antenna at time $t_r$ (receive time) and the satellites antenna at time $t_t$ (transmit time ) in meters.

$\delta t_r$ is the receiver's clock offset in seconds.

The geometric interpretation between $m$th satellite and receiver is,

$$r^m = \sqrt{(x - x^m)^2 + (y - y^m)^2 + (z - z^m)^2} = \|\boldsymbol{x} - \boldsymbol{x}^m\| \tag{3.46}$$

Where

$\boldsymbol{x} = [x, y, z]^t$ is the receiver position in ECEF

$\boldsymbol{x^m} = [x^m, y^m, z^m]^t$ is the position of the $m$th satellite in ECEF frame

Then equation 3.45 in vector form,

$$\rho_c^m = \|\boldsymbol{x} - \boldsymbol{x}^m\| + b_r \tag{3.47}$$

where $b_r = c\delta t_r$ is the error range from bias error of receiver in meters. By using Taylor series expansion, equation 3.47 can be linearized. Discarding the higher terms, linearized version of equation 3.47 can be written with best estimate of position.

$$\boldsymbol{x}_{est} = [x_{est}, y_{est}, z_{est}]^t \tag{3.48}$$

Then the linearization becomes,

$$\begin{aligned}\rho_c^m =& \sqrt{(x_{est} - x^m)^2 + (y_{est} - y^m)^2 + (z_{est} - z^m)^2} + \\ & \frac{(x_{est} - x^m)(x - x_{est}) + (y_{est} - y^m)(y - y_{est}) + (z_{est} - z^m)(z - z_{est})}{\sqrt{(x_{est} - x^m)^2 + (y_{est} - y^m)^2 + (z_{est} - z^m)^2}} + b_r\end{aligned} \tag{3.49}$$

Estimated range can be defined,

$$\rho_{c,est}^m = \sqrt{(x_{est} - x^m)^2 + (y_{est} - y^m)^2 + (z_{est} - z^m)^2} \tag{3.50}$$

Subtract the estimated equation 3.50 from 3.49,

$$\rho_c^m - \rho_{c,est}^m = \frac{(x_{est} - x^m)(x - x_{est}) + (y_{est} - y^m)(y - y_{est}) + (z_{est} - z^m)(z - z_{est})}{\sqrt{(x_{est} - x^m)^2 + (y_{est} - y^m)^2 + (z_{est} - z^m)^2}} + b_r - b_{r,est} \tag{3.51}$$

Equation 3.51 can be written more compactly. In matrix form equation can be written as in equation below.

$$\delta\rho_c^m = (L_{est}^m \delta x) + \delta b_r \tag{3.52}$$

Where

$$\begin{aligned}\delta\rho_c^m &= \rho_c^m - \rho_{c,est}^m \\ \delta b_r &= b_r - b_{r,est} \\ L_{est}^m &= \frac{[(x_{est} - x^m), (y_{est} - y^m), (z_{est} - z^m)]^t}{\sqrt{(x_{est} - x^m)^2 + (y_{est} - y^m)^2 + (z_{est} - z^m)^2}}\end{aligned}$$

Here, $L_{est}^m$ is the line of sight unit vector from the $m$th satellite to the receiver's position.

In general, the linearized pseudo-range measurements equations can be written as M satellites as below,

$$\delta\boldsymbol{\rho}_c = \begin{bmatrix} \delta\rho_c^1 \\ \delta\rho_c^2 \\ \vdots \\ \delta\rho_c^M \end{bmatrix}_{M\times 1} = \begin{bmatrix} (L_{est}^1)^t & 1 \\ (L_{est}^2)^t & 1 \\ \vdots & \vdots \\ (L_{est}^M)^t & 1 \end{bmatrix}_{M\times 4} \begin{bmatrix} \delta x \\ \delta b_r \end{bmatrix}_{4\times 1} \tag{3.53}$$

Finally, the compact version is,

$$\delta\boldsymbol{\rho}_c = G_{M\times 4} \begin{bmatrix} \delta x \\ \delta b_r \end{bmatrix}_{4\times 1} \tag{3.54}$$

where $G$ is the geometry matrix with $M \times 4$ dimensions which characterizes the relative geometry of the satellite and receiver. To solve for the unknowns

$$\begin{bmatrix} \delta\boldsymbol{x} \\ \delta b_r \end{bmatrix} = G^{-1}\delta\boldsymbol{\rho}_c \tag{3.55}$$

For the case of $M > 4$ the system is over-determined and the solution can be found by using the least squares criterion. The least square solution can be given as,

$$\begin{bmatrix} \delta\hat{\boldsymbol{x}} \\ \delta\hat{b}_r \end{bmatrix} = (G^tG)^{-1}G^t\delta\boldsymbol{\rho}_c \tag{3.56}$$

The improved states are,

$$\hat{\boldsymbol{x}} = \boldsymbol{x}_{est} + \delta\hat{x} \tag{3.57}$$

$$\hat{b}_r = b_{r,est} + \delta\hat{b}_r \tag{3.58}$$

The process must be sustained until pseudo-range measurements are obtained in desired error range.

**Velocity Estimation**

Pseudo-range rate can be written as,

$$\dot{\rho}^m = L_x^m(\nu_x - \nu_x^m) + {}_y^m(\nu_y - \nu_y^m) + l_z^m(\nu_z - \nu_z^m) + d_r \tag{3.59}$$

Where

$L^m = \dfrac{[(x-x^m),(y-y^m),(z-z^m)]^t}{\sqrt{(x-x^m)^2+(y-y^m)^2+(z-z^m)^2}} = [L_x^m, L_y^m, L_Z^m]^t$ True line of sight vector

$d_r = c\delta\dot{t}_r$ Receiver clock drift in meters/sec

Estimated pseudo-range rate will be,

$$\dot{\rho}^m_{est} = L^m_{x,est}(\nu_{x,est} - \nu^m_x) + {}^m_{y,est}(\nu_{y,est} - \nu^m_y) + l^m_{z,est}(\nu_{z,est} - \nu^m_z) + d_{r,est} \quad (3.60)$$

In equation 3.60 $\nu_{x,est}, \nu_{y,est}, \nu_{z,est}$ are the estimated velocity components of the receiver in ECEF frame. The error in the pseudo-range rate can be calculated as,

$$\dot{\rho}^m - \dot{\rho}^m_{est} = L^m_{x,est}(\nu_x - \nu_{x,est}) + L^m_{y,est}(\nu_y - \nu_{y,est}) + L^m_{z,est}(\nu_z - \nu_{z,est}) + d_r - d_{r,est} \quad (3.61)$$

In matrix form,

$$\delta\dot{\rho}^m = (L^m_{est})^t \delta\nu + \delta d_r \quad (3.62)$$

For M satellite, the linearized pseudo-range rate measurement can be written as,

$$\delta\dot{\rho} = \begin{bmatrix} \delta\dot{\rho}^1 \\ \delta\dot{\rho}^2 \\ \vdots \\ \delta\dot{\rho}^M \end{bmatrix}_{M\times 1} = \begin{bmatrix} (L^1_{est})^t & 1 \\ (L^2_{est})^t & 1 \\ \vdots & \vdots \\ (L^M_{est})^t & 1 \end{bmatrix}_{M\times 4} \begin{bmatrix} \delta\nu \\ \delta d_r \end{bmatrix}_{4\times 1} \quad (3.63)$$

Or in compact form the equation 3.64 can be written as,

$$\delta\dot{\rho}_{M\times 1} = G_{M\times 4} \begin{bmatrix} \delta\nu \\ \delta d_r \end{bmatrix}_{4\times 1} \quad (3.64)$$

To solve both position and velocity equation simultaneously, equation 3.53 and 3.63 can be combined in one model.

$$\begin{bmatrix} \delta\rho^1_c \\ \vdots \\ \delta\rho^M_c \\ \delta\dot{\rho}^1 \\ \vdots \\ \delta\dot{\rho}^M \end{bmatrix}_{'M\times 1} = \begin{bmatrix} (L^1_{est})^t & 1 & 0_{3\times 1} & 0 \\ \vdots & \vdots & \vdots & \vdots \\ (L^M_{est})^t & 1 & 0_{3\times 1} & 0 \\ 0_{3\times 1} & 0 & (L^1_{est})^t 1 & \\ \vdots & \vdots & \vdots & \vdots \\ 0_{3\times 1} & 0 & (L^M_{est})^t 1 & \end{bmatrix}_{2M\times 8} \begin{bmatrix} \delta x \\ \delta b_r \\ \delta v \\ \delta d_r \end{bmatrix}_{8\times 1} \quad (3.65)$$

In matrix form, rewrite the equation 3.65,

$$\delta z_{2M\times 1} = \tilde{G}_{2M\times 8} \delta S_{8\times 1} \quad (3.66)$$

By using least square techniques, in the case of more than 4 satellites equation 3.67 can be used.

$$\delta\hat{S} = (\tilde{G}^t\tilde{G})^{-1}\tilde{G}^t\delta z \quad (3.67)$$

Finally the improved estimates of the receiver's position and clock bias are,

$$\hat{x} = x_{est} + \delta\hat{x} \tag{3.68}$$

$$\hat{v} = v_{est} + \delta\hat{v} \tag{3.69}$$

$$\hat{b}_r = b_{r,est} + \delta\hat{b}_r \tag{3.70}$$

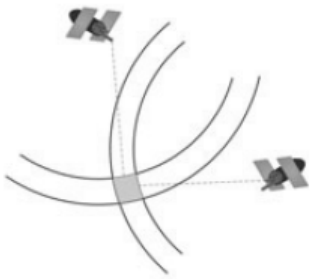
$$\hat{d} = d_{r,est} + \delta\hat{d}_r \tag{3.71}$$

Also, if a priori estimates have large errors then the least squares solution will be iterated until the change in the estimate is sufficiently small.

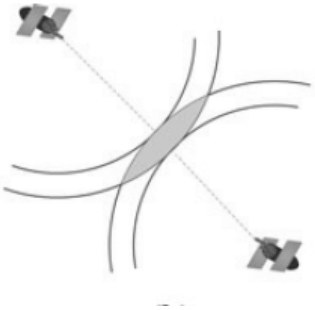
**Satellite Geometry and Dilution of Precision:**

Positions of the available satellites around the GPS receiver are important to determine the position estimation accurately and this effect is called *geometric dilution of precision (GDOP)* or simply dilution of precision (DOP) [2],[14]. The illustration of the concept is given below figure. Ideally the signals from the

(a) The ideal position



(b) Favorable geometry



(c) Poor geometry

Figure 3.5: The dilution of precision with range measurements in two dimensions. [2]

satellites should form the circles. Therefore the circles intersect at a point. However,

because of that there is always an error in range measurement, the uncertainty causes two concentric circles as shown in Figure 3.5b. The area between circles is the region of the uncertainty in the range. The area of this region depends on the relative geometry of two satellites with respect to user. In Figure 3.5b the satellites are located at right angles relative the user and the area of the intersection is smaller, thus DOP is lower, in other words the error is smaller in position determination. In Figure 3.5c, satellites are almost linear, so the area of intersection and the error are large.

Dilution of precision can be computed by the geometry matrix G [2]. The error covariance matrix can be written as,

$$E(\delta\hat{y}\delta\hat{y}^t) = (G^tG)^{-1}G^tRG(G^tG)^{-1} \tag{3.72}$$

$R$ is the covariance matrix of the pseudo-range measurements. Assuming that the measurement error is uncorrelated and has the same variance $\sigma^2$. Then it can be written,

$$R = \sigma^2 I \tag{3.73}$$

Therefore equation 3.72 will be,

$$E(\delta\hat{y}\delta\hat{y}^t) = \sigma^2(G^tG)^{-1} \tag{3.74}$$

By defining the $C = E(\delta\hat{y}\delta\hat{y}^t)$ and $H = (G^tG)^{-1}$, we can write the elements of $C = \sigma^2 H$ as,

$$\begin{aligned}
\sigma_x^2 &= \sigma^2 H_{11} \\
\sigma_y^2 &= \sigma^2 H_{22} \\
\sigma_z^2 &= \sigma^2 H_{33} \\
\sigma_b^2 &= \sigma^2 H_{44}
\end{aligned} \tag{3.75}$$

Various DOP parameters which show the role of user-satellite geometry can be defined. Some of these parameters are given below.

Position dilution of precision $(PDOP) = \sqrt{H_{11} + H_{22} + H_{33}}$

Time dilution of precision $(TDOP) = \sqrt{H_{44}}$

$$\text{Geometric dilution of precision } (GDOP) = \sqrt{H_{11} + H_{22} + H_{33} + H_{44}}$$

The satellite-user geometry improves with the increasing number of satellites. Therefore the positional accuracy increases.

## 3.5 Magnetometer

Using the Earth magnetic field is one of the oldest navigation techniques to determine the directional reference. Today by using magnetometers the direction of the path can be measured directly. Also by using the Earth magnetic field the attitude of the vehicle can be determined. In the absence of local magnetic interferences, magnetometer senses the Earth magnetic field intensity acting along its sensitive axes in body frame. Moreover in recent studies magnetometer was used to demonstrate the benefits of using the magnetic data during GPS signal outages [17].

The Earth magnetic field direction lies close to true north and magnetic north. However the angle between true north and magnetic north is not constant. It varies with the position on Earth and time. The direction of the Earth magnetic field is defined in terms of orientation with respect to true north. The orientation with respect to true north is called *magnetic declination* ($\gamma$) and the its angle with horizontal is called as *angle of dip* ($\delta$) [1]. The measured values in magnetometer can not be distinguished from that of Earth. Therefore this effect must be included in calculation. The illustration of Earth magnetic field with Earth coordinate is given in Figure 3.6.

### 3.5.1 Magnetic Measurements

Strapdown magnetometer can be used to measure heading of the vehicle. Three magnetometers which are oriented in orthogonal direction is used to measure $H_x, H_y$ and $H_z$ magnetic vector components. To obtain heading angle the compass orientation needs to be mathematically transformed in the navigation frame. The

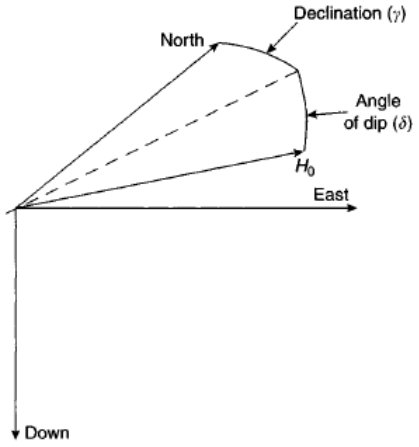



Figure 3.6: Components of Earth magnetic field [1]

rotational transformation is achieved by applying the rotation equations [1].

$$\begin{bmatrix} H_x \\ H_y \\ H_z \end{bmatrix} = C_n^b \begin{bmatrix} H_0 \cos(\delta) \cos(\gamma) \\ H_0 \cos(\delta) \cos(\gamma) \\ H_0 \sin(\delta) \end{bmatrix} \tag{3.76}$$

Where $\delta$ is the angle of dip and $\gamma$ is the angle of declination. Then heading can be found from equation 3.76.

$$\begin{bmatrix} \cos(\theta) & 0 & -\sin(theta) \\ 0 & 1 & 0 \\ \sin(theta) & 0 & \cos(theta) \end{bmatrix}^{-1} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & \sin(\phi) \\ 0 & -\sin(phi) & \cos(\theta) \end{bmatrix}^{-1} \begin{bmatrix} H_x \\ H_y \\ H_z \end{bmatrix} = \begin{bmatrix} \cos(\psi) & \sin(\psi) & 0 \\ -\sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} H_0 \cos(\delta) \cos(\gamma) \\ H_0 \cos(\delta) \cos(\gamma) \\ H_0 \sin(\delta) \end{bmatrix} \tag{3.77}$$

Rearranging the equation 3.77, the heading $\psi$ can be found as,

$$\psi = \tan^{-1}(H_y/H_x) - \gamma \tag{3.78}$$

### 3.5.2 Magnetometer Error Analysis

The main error of a magnetometer are the scale factor, sensor offsets, sensor non-orthogonality, misalignment and magnetic deviation. The last one magnetic deviation

depends on the surrounding magnetic field anomalies. These effects are called as soft iron and hard iron effects [18].

**Scale Factor:** Instrumentation errors are due to fabrication processes. They have a constant, specific parameters for each magnetometer. For each sensitivity axis this parameter can be different. The scale factor is shown as

$$S = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & s_z \end{bmatrix} \tag{3.79}$$

**Misalignment:** Due to fabrication issues, the axis of triad is not mounted perfectly aligned with the sensor axes. Non-orthogonal sensing elements and misalignment can be modeled with a matrix $M$.

$$M = \begin{bmatrix} m_{xx} & m_{xy} & m_{xz} \\ m_{yx} & m_{yy} & m_{yz} \\ m_{zx} & m_{zy} & m_{zz} \end{bmatrix} \tag{3.80}$$

**Sensor Offset:** The sensor offset introduces a bias. It is modelled as,

$$B_{so} = \begin{bmatrix} b_{so,x} \\ b_{so,y} \\ b_{so,z} \end{bmatrix} \tag{3.81}$$

**Hard and Soft Iron:** Due to the local deviations of the magnetic field which are caused by on-board hardware and surrounding materials the error source of soft iron and hard iron effect are observed.

The soft component of the magnetic deviation corresponds to induced magnetization. The source of this error is permeability of ferromagnetic compounds. This Error changes the intensity and the direction of magnetic field. Soft iron can be modelled by $3 \times 3$ matrix,

$$A_{si} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{12} & a_{22} & a_{23} \\ a_{13} & a_{32} & a_{33} \end{bmatrix} \tag{3.82}$$

The hard component of magnetic deviation corresponds to a permanent magnetic deviation. It results from permanent magnets and magnetic hysteresis. Hard iron effect can be represented as,

$$B_{hi} = \begin{bmatrix} b_{hi,x} \\ b_{hi,y} \\ b_{hi,z} \end{bmatrix} \tag{3.83}$$

By using these parameters the complete error model for magnetometers can be written as in the following equation [18].

$$\hat{H} = SM(A_{si}H - B_{hi}) - B_{so} + n \tag{3.84}$$

$$= SMA_{si}H - SMB_{hi} - B_{so} + n \tag{3.85}$$

$$= AH - b + n \tag{3.86}$$

where $H$ is the error free magnetic field and $\hat{H}$ is the readings from magnetometer. $n$ is Gaussian white noise. $A = SMA_{si}$ and $b = SMB_{hi} + B_{so}$.

In this this thesis, to create accelerometer, gyroscope, GPS and magnetometer models this chapter is utilized. For accelerometer and and gyroscope models equations 3.2 and 3.6 are used respectively. The related error parameters are obtained from reference [1]. For GPS to simulate satellite dynamics equations between 3.13 and 3.44 are used. Also for position and velocity estimation of receiver, equations between 3.45 and 3.67 are utilized. To measure the estimation accuracy of receiver position and velocity estimation, satellite dilution of precision parameters are calculated according to equations between 3.72 and 3.75. Finally magnetometer errors are modelled according to equations between 3.76 and 3.84.

# CHAPTER 4

# KALMAN FILTER AND INS/GPS/MAGNETOMETER INTEGRATION

## 4.1 Kalman Filter Introduction

In previous chapter it is stated that the errors in INS are affected by inertial sensors. Also the computational errors and initialization errors affect the results of INS computation. In order to improve INS results, at regular time intervals the estimation procedure should be applied to predict stochastic errors for compensation. If the elimination of errors are not realized, the whole positioning accuracy will jeopardize.

To eliminate the error sources various methods are available. Kalman Filtering (KF) and its variances, particle filter (PF) and artificial intelligence are some of them. In traditional, Kalman filter procedure is the most preferred one among other techniques. Especially the INS/GPS system integration via Kalman filter is the common sensor fusion technique in navigation computation area.

Generally Kalman filter is an algorithm to estimate noise in the error states of system optimally. Kalman filter is a recursive algorithm that provides an optimal least mean variance estimation of error states. By selecting the weighing criteria of measurement, KF estimates the current value of the state approximately. KF uses the following informations to estimate states [2],

1. Model and measurement information

2. Stochastic error characteristic of system noise and measurement noise

3. Initial condition values of the KF states.

In this section, Kalman filter for linear system will be considered. Before non-linear system, the algorithm which is applicable for linear system is presented. Then for non-linear system Extended Kalman filter will be discussed.

### 4.1.1 Discrete Time Kalman Filter

The dynamical behaviour of the linear system can be represented by first order differential equations in the form of,

$$\dot{\boldsymbol{x}} = F\boldsymbol{x} + G\boldsymbol{u} + \boldsymbol{w} \tag{4.1}$$

here, $\boldsymbol{x}$ is the states of the system, $\boldsymbol{u}$ is the deterministic input and $\boldsymbol{w}$ is the system noise. $F$ is the system matrix and $G$ is the system input matrix. $F$ and $G$ can be constant or time-varying matrices [1]. The noise vector $\boldsymbol{w}$ has zero mean and Gaussian distribution with power spectral density (PSD) $Q$. If the measurement matrix of the system is considered, this can be expressed as,

$$\boldsymbol{y} = H\boldsymbol{x} + \boldsymbol{\eta} \tag{4.2}$$

where $\boldsymbol{y}$ is the measurement vector and $H$ is measurement matrix. For the noise effect, $\boldsymbol{\eta}$ can be selected as measurement noise which has zero mean and normally distributed, with PSD $R$. As for discrete linear system, the discrete KF models can be described as follow [2],

$$\boldsymbol{x}_k = \Phi_{k,k-1}\boldsymbol{x}_{k-1} + G_{k-1}\underline{\mathrm{w}}_{k-1} \tag{4.3}$$

where

$\boldsymbol{x}_k$ is the state vector

$\Phi_{k,k-1}$ is the state transition matrix

$G_{k-1}$ is the noise distribution vector

$\boldsymbol{w}_{k-1}$ is the process noise vector

Discrete time linear measurement equation of the system is,

$$\boldsymbol{z}_k = H_k\boldsymbol{x}_k + \boldsymbol{\eta}_k \tag{4.4}$$

where

| | |
|---|---|
| $\boldsymbol{z}_k$ | is the measurement vector of the system output |
| $H_k$ | is the observation matrix |
| $\boldsymbol{\eta}_k$ | is the measurement noise vector |

The state transition matrix $\Phi$ represents the dynamic behaviour of the system. For the given dynamic coefficient matrix $F$ in continuous time system, the state transition matrix can be given as,

$$\Phi = exp(F\Delta t) \tag{4.5}$$

For the discrete time systems, by using Taylor series expansion $\Phi$ can be written as,

$$\Phi = (I + F\Delta t) \tag{4.6}$$

where $I$ is the identity matrix and $\delta t$ is sampling interval.

To apply Kalman filter the following assumptions must be considered.

1. Both system and measurement must be represented by linear models.

2. The system noise $\boldsymbol{w}_k$ and measurement noise $\boldsymbol{\eta}_k$ must be uncorrelated, zero mean, Gaussian noise with the given covariance matrices.

$$E[\boldsymbol{w}_k] = 0, \quad E[\boldsymbol{\eta}_k] = 0 \tag{4.7}$$

$$E[\boldsymbol{w}_k \boldsymbol{\eta}_j^t] = 0 \tag{4.8}$$

$$E[\boldsymbol{w}_k \boldsymbol{w}_j^t] = \begin{cases} Q_k & k = j \\ 0 & k \neq j \end{cases} \tag{4.9}$$

$$E[\boldsymbol{\eta}_k \boldsymbol{\eta}_j^t] = \begin{cases} R_k & k = j \\ 0 & k \neq j \end{cases} \tag{4.10}$$

   where $Q_k$ and $R_k$ represent the covariance matrices of the system noise and measurement noise respectively [2].

3. The initial system state vector $\boldsymbol{x}_0$ is uncorrelated to both process and measurement noise vectors.

$$E[\boldsymbol{x}_0 \boldsymbol{w}_k^t] = 0, \quad E[\boldsymbol{x}_0 \boldsymbol{\eta}_k^t] = 0 \tag{4.11}$$

4. The mean value of the initial state $\bar{\boldsymbol{x}}_0$ and its covariance matrix $P_0$ are known.

$$\bar{\boldsymbol{x}}_0 = E[\underline{[}x_0] \tag{4.12}$$

$$P_0 = E[(\boldsymbol{x}_0 - \bar{\boldsymbol{x}}_0)(\boldsymbol{x}_0 - \bar{\boldsymbol{x}}_0)^t] \tag{4.13}$$

### 4.1.2 Kalman Filter Algorithm Procedure

Kalman filter estimates the states of the system by using a recursive algorithm. The operation order of Kalman filter consists of two phase. The first phase is the prediction or time update. In this phase the model propagates from current state the next state with its covariance matrix between time interval $k-1$ and $k$. The second phase is correction or measurement update. In this phase measurements are used to update previous estimate, therefore improving the last estimate [2].

Let discuss Kalman filter stages in detail.

**Prediction Stage (Time Update)**

The estimation of system state $\boldsymbol{x}$ at time $k$ with the given information up to time $k-1$ is called prediction. The prediction states are shown as $\hat{\boldsymbol{x}}_k(-)$. Since system noise is zero mean the best prediction will be,

$$\hat{\boldsymbol{x}}_k(-) = \Phi_{k|k-1}\boldsymbol{x}_{k-1}(+) \tag{4.14}$$

where $\boldsymbol{x}_{k-1}(+)$ is the best estimate during last epoch. Kalman Filter also propagates the uncertainty which is called error covariance. It is represented by covariance matrix $P_k(-)$ and calculated by using following formula

$$P_k(-) = \Phi_{k|k-1}P_{k-1}(+)\Phi^t_{k|k-1} + G_{k-1}Q_{k-1}G^t_{k-1} \tag{4.15}$$

where $P_{k-1}(+)$ represents the best estimate of covariance in last epoch.

**Correction Stage (Measurement Update)**

The correction of the Kalman filter comes with the measurement part. After Kalman filter predicts its estimate, this value is corrected with measurement. Firstly the Kalman gain $K$ is computed based on measurement covariance $R_k$. The Kalman

gain is determined such that it minimizes the mean square error of estimate.

$$K_k = P_k(-)H_k^t[H_k P_k(-)H_k^t + R_k]^{-1} \tag{4.16}$$

As seen from equation 4.15 $K$ depends on both $P_k(-)$ and $R_k$. If the measurements are noisy ($R_k$ increases) or process noise is lower ($P_k(-)$ decreases) then K becomes relatively smaller. Otherwise, when more noise in process ($P_k(-)$ increases) or measurement is less noisy ($R_k$ decreases) $K$ becomes relatively larger. As a result, when $K$ is large it assigns more weight to measurement and when it is small it shows greater faith in prediction. For example, In the INS/GPS integration, K takes relatively larger values when the GPS is more accurate and less noisy. In that case the measurement covariance matrix becomes relatively small.

When new measurement arrives at time $t_k$ the difference between prediction and measurement $\boldsymbol{z}_k$ is weighted by Kalman gain.

$$\hat{\boldsymbol{x}}_k(+) = \hat{\boldsymbol{x}}_k(-) + K_k[\boldsymbol{z}_k - H_k\hat{\boldsymbol{x}}_k(-)] \tag{4.17}$$

where $H_k\hat{\boldsymbol{x}}_k(-)$ is the prediction observation and $\boldsymbol{z}_k - H_k\hat{\boldsymbol{x}}_k(-)$ is the innovation sequence.

In measurement update stage Kalman filter also update the uncertainty of its new prediction $\hat{\boldsymbol{x}}_k(+)$,

$$P_k(+) = [I - K_k H_k]P_k(-) \tag{4.18}$$

The expanded form or *Joseph* form of the equation 4.17.

$$P_k(+) = [I - K_k H_k]P_k(-)[I - K_k H_k]^t + K_k R_k K_k^t \tag{4.19}$$

which is numerically stable and gives correct answers even when the computation of K has an error. Figure 4.1 shows the flow diagram of the Kalman filter algorithm. In most Kalman Filter application update procedure is at lower rate than prediction stage. For example in INS/GPS integration KF prediction is at 100 Hz whereas KF update procedure may realize at 1 Hz.

### 4.1.3 Non-linear (Extended) Kalman Filter

Linear dynamic systems use standard Kalman filter to estimate the states. The same procedure can be extended to estimation of non-linear systems. For the non-linear

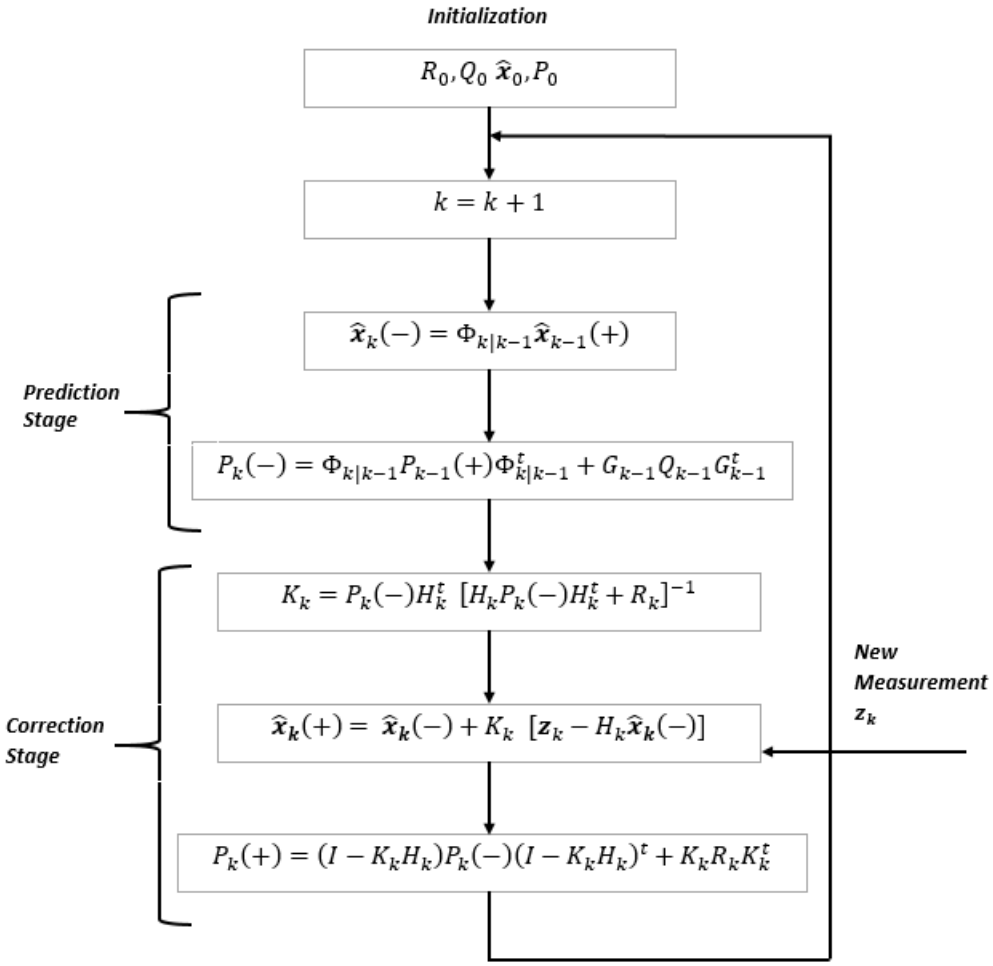



Figure 4.1: Kalman filter procedure

case, the Kalman filter is named as *Extended Kalman Filter*. For the non-linear systems, it can be written as,

$$\boldsymbol{x}_{k+1} = f_k(\boldsymbol{x}_k) + \boldsymbol{w}_k \tag{4.20}$$

$$\boldsymbol{y}_k = h_k(\boldsymbol{x}_k) + \boldsymbol{\eta}_k \tag{4.21}$$

where $\boldsymbol{x}_k$ is the state vector, $f_k(\boldsymbol{x}_k)$ is the non-linear state transition matrix and $\boldsymbol{w}_k$ is the process noise matrix. $\boldsymbol{y}_k$ is the measurement vector, $h_k(\boldsymbol{x}_k)$ is the non-linear measurement function and $\boldsymbol{\eta}_k$ is the measurement noise vector. Again $\boldsymbol{w}_k$ and $\boldsymbol{v}_k$ are zero mean Gaussian noises.

To apply the Kalman Filter, the basic idea is the linearization of the non-linear functions $f_k(\boldsymbol{x}_k)$ and $h_k(\boldsymbol{x}_k)$ around the recent estimate $\boldsymbol{x}_k$. Linearization of the

non-linear functions can be determined by calculating the Jacobian matrices [1].

$$F_k = \left.\frac{\partial f_k(\boldsymbol{x}_k)}{\partial \boldsymbol{x}_k}\right|_{\boldsymbol{x}=\boldsymbol{x}_k} \quad H_k = \left.\frac{\partial h_k(\boldsymbol{x}_k)}{\partial \boldsymbol{x}_k}\right|_{\boldsymbol{x}=\boldsymbol{x}_k} \tag{4.22}$$

After obtaining the linearized system matrices, standard Kalman filter procedure can be applied. The procedure is given in Table 4.1. In this study the extended Kalman filter procedure will be followed. In following section the linearized system model will be obtained.

### 4.1.4 Autocovariance Least-Squares Method

In Kalman filter procedure to find the optimal Kalman gain, system noise $Q$ and measurement noise $R$ covariance matrices must be determined. However the covariance matrices are not known in general. To estimate the covariance matrices *Autocovariance Least Square Method (ALS)* is proposed by Brian J. Odelson et al.[19]

According to Odelson [19], extracting the covariance information from collected data is more preferable and systematic approach. For linear time varying systems Ming Ge and Eric C. Kerrigan [20] offer ALS as in below. Before going into detail of the approach, consider the following discrete state-space model:

$$x_{k+1} = A_k x_k + G_k w_k \tag{4.23}$$

$$y_k = C_k x_k + v_k \tag{4.24}$$

where $A_k$ and $C_k$ are state-space matrices with $n \times n$ and $p \times n$ dimensions. $G_k$ noise matrices with $n \times r$ dimension. The noise parameters $w_k$ and $v_k$ are random Gaussian distributions with zero mean and unknown covariance matrices $Q$ and $R$. Because $Q$ and $R$ matrices are unknown, sub-optimal filter is used to estimate state sequence as in 4.25

$$\hat{x}_k = \hat{x}_{k|k-1} + L_k(y_k - \hat{y}_{k|k-1}) \quad k = 1, ..., M \tag{4.25}$$

Table 4.1: Extended Kalman Filter [2]

| |
|---|
| **State Space Model** |
| $$\boldsymbol{x}_{k+1} = f_k(\boldsymbol{x}_k) + \boldsymbol{w}_k$$ $$\boldsymbol{y}_k = h_k(\boldsymbol{x}_k) + \boldsymbol{\eta}_k$$ |
| **Initilization of Kalman Filter** |
| $$\hat{\boldsymbol{x}}_0 = E(\boldsymbol{x}_0))$$ $$P_0 = E((\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0)(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0)^t)$$ |
| **Calculate Jacobians** |
| $$F_k = \frac{\partial f_k(\boldsymbol{x}_k)}{\partial \boldsymbol{x}_k}\Big\vert_{\boldsymbol{x}=\boldsymbol{x}_k} \quad H_k = \frac{\partial h_k(\boldsymbol{x}_k)}{\partial \boldsymbol{x}_k}\Big\vert_{\boldsymbol{x}=\boldsymbol{x}_k}$$ |
| **Time Update** |
| $$\hat{\boldsymbol{x}}_k(-) = f_k(\boldsymbol{x}_{k-1}(+))$$ $$P_k(-) = F_k P_{k-1}(+) F_k^t + Q_{k-1}$$ |
| **Kalman Gain** |
| $$K_k = P_k(-) H_k^t [H_k P_k(-) H_k^t + R_k]' - 1$$ |
| **Measurement Update** |
| $$\hat{\boldsymbol{x}}_k(+) = \hat{\boldsymbol{x}}_k(-) + K_k[\boldsymbol{y}_k - H_k \hat{\boldsymbol{x}}_k(-)]$$ $$P_k(+) = [I - K_k H_k] P_k(-) [I - K_k H_k]^t + K_k R_k K_k^t$$ |

By using the one-step ahead predicted state and output, the state error term is defined as below:

$$\hat{x}_{k+1|k} = A_k \hat{x}_k \tag{4.26}$$

$$\hat{y}_{k|k-1} = C_k \hat{x}_{k|k-1} \tag{4.27}$$

$$\epsilon_k = x_k - \hat{x}_{k|k-1}, \quad k = 1, ..., M \tag{4.28}$$

Therefore estimation error propagation form can be written by using equation 4.26, 4.27 and 4.28.

$$\epsilon_{k+1} = (A_k - A_k L_k C_k)\epsilon_k + [G_k \ - A_k L_k] \begin{bmatrix} w_k \\ v_k \end{bmatrix} \tag{4.29}$$

where

$$\bar{A}_k = A_k - A_k L_k C_k$$
$$\bar{G}_k = [G_k \ - A_k L_k]$$
$$\bar{w}_k = [w_k \ v_k]^T$$

Now the innovation sequence can be defined as,

$$z_k = y_k - \hat{y}_{k|k-1} \tag{4.30}$$

Therefore

$$z_k = C_k \epsilon_k + v_k \tag{4.31}$$

The suboptimal filter data is correlated with each other for $k = 1, ..., M$. The similarity between collected data and its lagged version can be represented with auto-covariance matrix. The auto-covariance matrix with time lag $j = 0, 1, .., N-1$ can be written as,

$$\zeta_j((z_k)_{k=k_0}^{k_0+M+N}) = E[z_{k_0+j} z_{k_0}^T \quad ... \quad z_{k_0+M-N+j} z_{k_0+M-N}^T] \tag{4.32}$$

where $k_0 > 1$ to eliminate uncertainty of initial values. Based on equation 4.32, the auto-covariance matrix can be defined as,

$$\mathcal{R} = \begin{bmatrix} \zeta_0((z_k)_{k=k_0}^{k_0+M+N}) \\ \vdots \\ \zeta_{N-1}((z_k)_{k=k_0}^{k_0+M+N}) \end{bmatrix} \tag{4.33}$$

The calculation of auto-covariance matrix based on collected data,

$$\bar{\mathcal{R}}_i = \frac{1}{M-N+1}\begin{bmatrix} z_{k_0+i} & \cdots & z_{k_0+i+M-N} \\ z_{k_0+i+1} & \cdots & z_{k_0+i+M-N+1} \\ \vdots & \ddots & \vdots \\ z_{k_0+i+N-1} & \cdots & z_{k_0+i+M-1} \end{bmatrix} \times \begin{bmatrix} z_{k_0+i}^T \\ z_{k_0+i+1}^T \\ \vdots \\ z_{k_0+i+M-N}^T \end{bmatrix} \tag{4.34}$$

and

$$\bar{\mathcal{R}} = \begin{bmatrix} \bar{\mathcal{R}}_0 & \bar{\mathcal{R}}_1 & \cdots & \bar{\mathcal{R}}_{M-N} \end{bmatrix} \quad i = 0, ..., M-N \tag{4.35}$$

The detailed explanation of extracting auto-covariance matrix into least square problem to find $Q$ and $R$ matrices given in reference [19] and [20]. Here, the least square form of the equation will be given.

$$(\mathcal{R}_i)_s = (\bar{\Gamma}_i \otimes \Gamma_i)\mathcal{I}_{1,n}(P_{k_0-1})_s + (\bar{\Omega}_i \otimes \Omega_i)\mathcal{I}_{i+1,r}(Q)_s + ((\bar{\Phi}_i \otimes \Phi)\mathcal{I}_{i+1,p} + I_p \otimes \Psi_i)(R)_s \tag{4.36}$$

In matrix form the equation 4.36 can be rewritten in least square form $\mathcal{A}x = b$

$$\underbrace{\begin{bmatrix} \mathcal{A}_0 \\ \mathcal{A}_1 \\ \vdots \\ \mathcal{A}_{M-N} \end{bmatrix}}_{\mathcal{A}} \underbrace{\begin{bmatrix} \hat{P}_{k_0-1} \\ \hat{Q} \\ \hat{R} \end{bmatrix}}_{x} = \underbrace{\begin{bmatrix} \bar{b}_0 \\ \bar{b}_1 \\ \vdots \\ \bar{b}_{M-N} \end{bmatrix}}_{b} \tag{4.37}$$

where

$$\mathcal{A}_i = \begin{bmatrix} (\bar{\Gamma}_i \otimes \Gamma_i)\mathcal{I}_{1,n}\mathcal{D}_n & (\bar{\Omega}_i \otimes \Omega_i)\mathcal{I}_{i+1,r}\mathcal{D}_r & ((\bar{\Phi}_i \otimes \Phi)\mathcal{I}_{i+1,p} + I_p \otimes \Psi_i)\mathcal{D}_p \end{bmatrix}$$
$$\bar{b}_i = (\bar{\mathcal{R}}_i)_s$$

Then ALS estimate values can be found by using the least square solution,

$$x = (\mathcal{A}^t\mathcal{A})^{-1}\mathcal{A}^t b \tag{4.38}$$

The detailed expansion of the ALS terms in equation 4.37 is given in appendix A.

## 4.2 INS/GPS and Magnetometer Integration with Kalman Filter

INS/GPS and magnetometer integration is based on error state system model. Position, velocity and attitude errors are the states of linearized model. This means

that the non-linear position, velocity and attitude equations must be linerized about navigation solution. The derivation of linerized state space model is given in following section.

### 4.2.1 Linearization of Dynamic Error Model

In this part the error state space matrix will be constructed for navigation operations. The derivation of error equations are given as follows [13].

**Attitude Errors:**

The direction cosine matrix $C_b^n$ propagates according to following formula,

$$\dot{C}_b^n = C_b^n \Omega_{ib}^b - \Omega_{in}^n C_b^n \tag{4.39}$$

Where $\Omega_{ib}^b$ and $\Omega_{in}^n$ are the skew symmetric of absolute body rate and navigation frame rate. For the attitude errors, small angle perturbation is used to linearize attitude equations according to state space variables.

$$\delta\dot{\boldsymbol{\Psi}} = -\boldsymbol{\omega}_{in}^n \times \delta\boldsymbol{\Psi} + \delta\boldsymbol{\omega}_{in}^n - C_b^n \delta\boldsymbol{\omega}_{ib}^b \tag{4.40}$$

where $\delta\boldsymbol{\Psi}$ is the attitude error states. Also $\delta\boldsymbol{\omega}_{ib}^b$ is the gyroscope error and $\delta\boldsymbol{\omega}_{in}^n$ is error in angular rate of n-frame and composed of,

$$\delta\boldsymbol{\omega}_{in}^n = \delta\boldsymbol{\omega}_{ie}^n + \delta\boldsymbol{\omega}_{en}^n \tag{4.41}$$

Moreover the Earth rate error can be given as,

$$\delta\boldsymbol{\omega}_{ie}^n = (-\Omega\sin(L)\delta L \quad 0 \quad -\Omega\cos(L)\delta L)^t \tag{4.42}$$

As a final, transport rate $\delta\boldsymbol{\omega}_{en}^n$ can be obtained as,

$$\delta\boldsymbol{\omega}_{en}^n = \begin{bmatrix} \frac{\delta V_e}{(R_e+h)} - \frac{V_e}{(R_e+h)^2}\delta h \\ -\frac{\delta V_n}{(R_n+h)} + \frac{V_n}{(R_n+h)^2}\delta h \\ \frac{-\tan(L)}{(R_e+h)}\delta V_e + \frac{V_e\tan(L)}{(R_e+h)^2}\delta h - \frac{V_e}{(R_e+h)\cos^2(L)}\delta L \end{bmatrix} \tag{4.43}$$

If the states are collected in matrix form, attitude error can be written as,

$$\delta\dot{\boldsymbol{\Psi}} = [F_{11} \quad F_{12} \quad F_{13}] \begin{bmatrix} \delta\boldsymbol{\Psi} \\ \delta\boldsymbol{V} \\ \delta\boldsymbol{P} \end{bmatrix} - C_b^n \delta\boldsymbol{\omega}_{ib}^b \tag{4.44}$$

where $P$ and $V$ stand for position and velocity states respectively in equation 4.28.

**Velocity Errors:**

Velocity error model can be obtained from equation 2.49. By perturbing the velocity equation, the error model is obtained as below.

$$\delta\dot{\boldsymbol{V}} = -\boldsymbol{\Psi}C_b^n\boldsymbol{f}^b + C_b^n\delta\boldsymbol{f}^b - (2\boldsymbol{\omega}_{ie}^n + \boldsymbol{\omega}_{en}^n) \times \delta\boldsymbol{V} - (2\delta\boldsymbol{\omega}_{ie}^n + \delta\boldsymbol{\omega}_{en}^n) - \delta\boldsymbol{g} \quad (4.45)$$

In state space form the equation 4.29 will be,

$$\delta\dot{\boldsymbol{V}} = [F_{21} \quad F_{22} \quad F_{23}] \begin{bmatrix} \delta\boldsymbol{\Psi} \\ \delta\boldsymbol{V} \\ \delta\boldsymbol{P} \end{bmatrix} \quad (4.46)$$

**Position Errors:**

Similarly the position error model is obtained from equations 2.57 - 2.59 by perturbing the states.

$$\delta\dot{L} = \frac{1}{(R_n + h)}\delta v_n - \frac{v_n}{(R_n + h)^2}\delta h \quad (4.47)$$

$$\delta\dot{l} = \frac{1}{(R_e + h)\cos(L)}\delta v_e + \frac{v_e \tan(L)}{(R_e + h)\cos(L)}\delta L - \frac{v_e}{(R_e + h)^2\cos(L)}\delta h \quad (4.48)$$

$$\delta\dot{h} = -\delta v_d \quad (4.49)$$

In matrix form, it can be written as follows,

$$\delta\dot{\boldsymbol{P}} = [F_{31} \quad F_{32} \quad F_{33}] \begin{bmatrix} \delta\boldsymbol{\Psi} \\ \delta\boldsymbol{V} \\ \delta\boldsymbol{P} \end{bmatrix} \quad (4.50)$$

Now the state space model can be construct by combining the matrix expressions in equations 4.28, 4.30 and 4.34. The states of the model are,

$$\delta\boldsymbol{\Psi} = [\delta\phi \quad \delta\theta \quad \delta\psi]^t \quad (4.51)$$

$$\delta\boldsymbol{V} = [\delta v_n \quad \delta v_e \quad \delta v_d]^t \quad (4.52)$$

$$\delta\boldsymbol{P} = [\delta L \quad \delta l \quad \delta h]^t \quad (4.53)$$

In matrix form,

$$\begin{bmatrix} \delta\dot{\boldsymbol{\Psi}} \\ \delta\dot{\boldsymbol{V}} \\ \delta\dot{\boldsymbol{P}} \end{bmatrix} = \begin{bmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{bmatrix} \begin{bmatrix} \delta\boldsymbol{\Psi} \\ \delta\boldsymbol{V} \\ \delta\boldsymbol{P} \end{bmatrix} + \begin{bmatrix} -C_b^n & 0_{3\times3} \\ 0_{3\times3} & C_b^n \\ 0_{3\times3} & 0_{3\times3} \end{bmatrix} \begin{bmatrix} \delta\boldsymbol{\omega} \\ \delta\boldsymbol{f} \end{bmatrix} \quad (4.54)$$

where $\delta\boldsymbol{\omega} = [\delta\omega_x \quad \delta\omega_y \quad \delta\omega_z]$ is the gyroscope noise parameters and $\delta\boldsymbol{f} = [\delta f_x \quad \delta f_y \quad \delta f_z]$ is the accelerometer noise parameters. Components of the state matrix are given below.

$$F_{11} = \begin{bmatrix} 0 & -(\Omega\sin(L) + \frac{v_e}{R}\tan(L)) & \frac{v_n}{R} \\ \Omega\sin(L) + \frac{v_e}{R}\tan(L) & 0 & \Omega\cos(L) + \frac{v_e}{R} \\ -\frac{v_n}{R} & -(\Omega\cos(L) + \frac{v_e}{R}) & 0 \end{bmatrix} \tag{4.55}$$

$$F_{12} = \begin{bmatrix} 0 & \frac{1}{R} & 0 \\ -\frac{1}{R} & 0 & 0 \\ 0 & -\frac{\tan(L)}{R} & 0 \end{bmatrix} \tag{4.56}$$

$$F_{13} = \begin{bmatrix} -\Omega\sin(L) & 0 & -\frac{v_e}{R^2} \\ 0 & 0 & \frac{v_n}{R^2} \\ -\Omega\cos(L) - \frac{v_e}{R\cos^2(L)} & 0 & \frac{v_e tan(L)}{R^2} \end{bmatrix} \tag{4.57}$$

$$F_{21} = \begin{bmatrix} 0 & -f_d & f_e \\ f_d & 0 & -f_n \\ -f_e & f_n & 0 \end{bmatrix} \tag{4.58}$$

$$F_{22} = \begin{bmatrix} \frac{v_d}{R} & -2(\Omega\sin(L) + \frac{v_e}{R}\tan(L)) & \frac{v_n}{R} \\ 2\Omega\sin(L) + \frac{v_e}{R}\tan(L) & \frac{1}{R}(v_n\tan(L) + v_d) & 2\Omega\cos(L) + \frac{v_e}{R} \\ -2\frac{v_n}{R} & -2(\Omega\cos(L) + \frac{v_e}{R}) & 0 \end{bmatrix} \tag{4.59}$$

$$F_{23} = \begin{bmatrix} -v_e(2\Omega\cos(L) + \frac{v_e}{R\cos^2(L)}) & 0 & \frac{1}{R^2}(v_e^2\tan(L) - v_n v_d) \\ 2\Omega(v_n\cos(L) - v_d\sin(L)) + \frac{v_n v_e}{R\cos^2(L)} & 0 & -\frac{v_e}{R^2}(v_n\tan(L) + v_d) \\ 2\Omega v_e\sin(L) & 0 & \frac{1}{R^2}(v_n^2 + v_e^2) \end{bmatrix} \tag{4.60}$$

$$F_{31} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{4.61}$$

$$F_{32} = \begin{bmatrix} \frac{1}{R} & 0 & 0 \\ 0 & \frac{1}{R\cos(L)} & 0 \\ 0 & 0 & -1 \end{bmatrix} \tag{4.62}$$

$$F_{33} = \begin{bmatrix} 0 & 0 & -\frac{v_n}{R^2} \\ \frac{v_e\tan(L)}{R\cos(L)} & 0 & -\frac{v_e}{R^2\cos(L)} \\ 0 & 0 & 0 \end{bmatrix} \tag{4.63}$$

In derivation of error model higher order terms in linearization are neglected. If higher terms are desired to be used in linearization process reference [21] can be used. In this study the upper equation will be utilized.

### 4.2.2 Types of Integration

As indicated before, there are individually pros and cons of INS and GPS system. Firstly, INS system is autonomous navigation system. It has good short term accuracy and provides position, velocity and attitude information. In long term, INS errors grow without bound. Secondly GPS has good long term accuracy with limited errors. To get the position accurately GPS needs at least four satellites which is not always possible.

The complementary characteristic of the two systems overcomes their individual drawbacks and gives more accurate and robust solutions. The integration of these two system is mostly based on Kalman filter technique. Therefore GPS prevents the deviation from inertial measurement of the INS and INS provides navigation solution during the GPS signal outages. To obtain maximum advantages, robustness and accuracy from optimal filtering, there are different integration techniques for Kalman filter [2],[14]. Specifically these are,

- Loosely coupled Kalman filter
- Tightly coupled Kalman Filter
- Ultra-tightly coupled Kalman Filter

#### 4.2.2.1 Loosely Coupled INS/GPS integration

In loosely coupled integration GPS and INS work independently from each other. The difference between INS solution and GPS measurement is sustained to Kalman filter. Based on the error model, the INS error states are estimated. The INS solution is corrected for the estimated error states and integrated to navigation equations for the next epoch. The diagram for the loosely coupled system is shown in Figure 4.2.

This system is also called as *decentralized* system because there is separate filter used for GPS. The main problem in loosely coupled system is the instability of GPS.

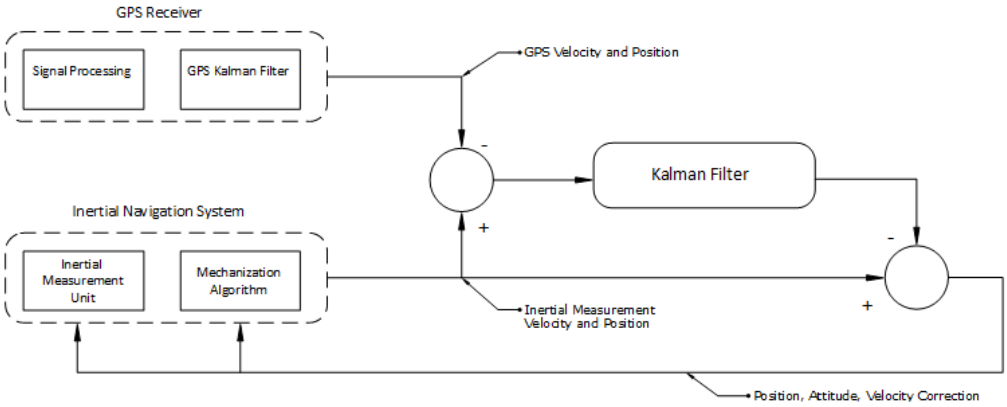



Figure 4.2: Block diagram for loosely coupled integration

When there are not sufficient satellites in line of sight, the KF filter assumption of uncorrelated measurement noise is put at risk. Therefore the system performance is reduced.

#### 4.2.2.2 Tightly Coupled INS/GPS integration

The difference of tightly coupled integration comes with pseudo-range and pseudo-range rate measurement. Kalman filter takes the difference of pseudo-range and pseudo-range rate measurement from GPS and the predicted values of them from INS. Then by finding the error states the navigation solution is corrected. In same cases pseudo-range is obtained GPS tracking code directly however pseudo-range rate can be calculated from GPS tracking code. This solution will be less robust [2]. The architecture is shown in Figure 4.3. This architecture is also called *centralized* integration because of single master filter.

This integration can eliminate the need for at least four satellite visibility. However tightly coupled system is more complex than loosely coupled system to implement. When compared with loosely coupled system, tightly coupled system is more accurate and robust but it comes with a extra states to Kalman filter to estimate and needs computationally extra space [2].

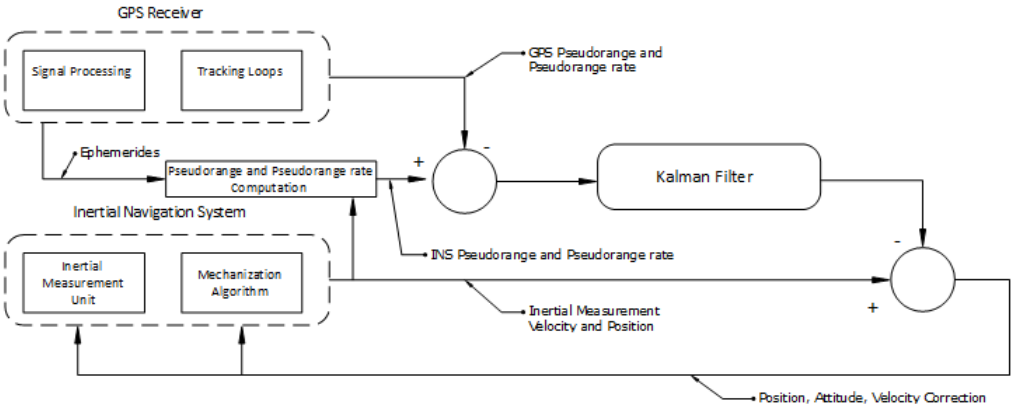



Figure 4.3: Block diagram for tightly coupled integration

#### 4.2.2.3 Ultra-Tight INS/GPS integration

The differences in this architecture is caused by deep integration. Specifically, the architecture of the GPS is different by means of integration of tracing loop. Also the information from INS is used as an integral part of GPS. Therefore the two systems are dependent on each other. This architecture directly affects the internal GPS hardware and complex to implement. The advantage of this system is the resistance of jamming. Moreover under lower signal to noise ratio and less than four GPS satellites this system works. The architecture of ultra-tight integration is shown in Figure 4.4.

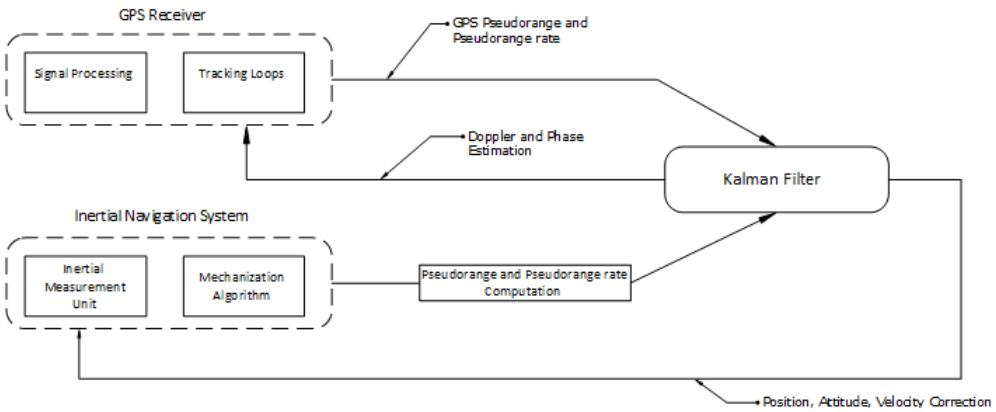



Figure 4.4: Block diagram for ultra-tight coupled integration

### 4.2.3 System and Measurement Model for Kalman Filter

In this thesis loosely coupled integration technique will be used. Therefore the system and measurement matrix are given according to loosely coupled structure.

**System Model**

Loosely coupled integration system system matrix can be written as,

$$\delta \dot{\boldsymbol{x}} = F\delta \boldsymbol{x} + Gw \tag{4.64}$$

Compare with equation 4.38, notice that the noise terms are replaced with unit variance white Gaussian noise $w$. All bias and white noise terms together with other source of error terms are included in mathematical model of Kalman filter in section 6. Now for the Kalman filter procedure, white noise terms are showed with states of Kalman filter. The $F$ state matrix is same with equation 4.38. The White noise term includes the followings[2].

$$G = \begin{bmatrix} \boldsymbol{\sigma}_{\Psi,1\times 3} & \boldsymbol{\sigma}_{V,1\times 3} & \boldsymbol{\sigma}_{P,1\times 3} \end{bmatrix}^t \tag{4.65}$$

The extended form of the general state space equation is,

$$\begin{bmatrix} \delta\dot{\boldsymbol{\Psi}} \\ \delta\dot{\boldsymbol{V}} \\ \delta\dot{\boldsymbol{P}} \end{bmatrix} = \begin{bmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{bmatrix} \begin{bmatrix} \delta\boldsymbol{\Psi} \\ \delta\boldsymbol{V} \\ \delta\boldsymbol{P} \end{bmatrix} + \begin{bmatrix} \boldsymbol{\sigma}_{\Psi,3\times 1} \\ \boldsymbol{\sigma}_{V,3\times 1} \\ \boldsymbol{\sigma}_{P,3\times 1} \end{bmatrix} w \tag{4.66}$$

where the white noise of attitude error is $\boldsymbol{\sigma}_{\Psi,3\times 1} = [\sigma_\phi \ \sigma_\theta \ \sigma_\psi]^t$, the white noise of velocity vector is $\boldsymbol{\sigma}_{V,3\times 1} = [\sigma_{v_n} \ \sigma_{v_e} \ \sigma_{v_d}]^t$ and finally the white noise of position vector is $\boldsymbol{\sigma}_{P,3\times 1} = [\sigma_L \ \sigma_l \ \sigma_h]^t$

**Measurement Model**

The measurement model for Kalman filter is shown below. Because the heading is also measured by magnetometer, as an observed state the $\psi$ value is also added. In matrix form the equation is,

$$\delta \boldsymbol{z} = H\delta \boldsymbol{x} + \boldsymbol{\nu} \tag{4.67}$$

The measurement vector $\delta \boldsymbol{z}$ consists the differences between INS measurements and

GPS/magnetometer measurements [2].

$$\delta \boldsymbol{z} = \begin{bmatrix} \psi_{INS} - \psi_{MAG} \\ v_{n,INS} - v_{n,GPS} \\ v_{e,INS} - v_{e,GPS} \\ v_{d,INS} - v_{d,GPS} \\ L_{INS} - L_{GPS} \\ l_{INS} - l_{GPS} \\ h_{INS} - h_{GPS} \end{bmatrix} \tag{4.68}$$

In equation 4.51, The term $H$ is the measurement matrix which relates observed states with the system matrix states. The term $\boldsymbol{\nu}$ is the measurement noise vector with a covariance matrix $R$. The measurement matrix and noise are simply written as,

$$\delta \boldsymbol{z} = H = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \delta \boldsymbol{\Psi} \\ \delta \boldsymbol{V} \\ \delta \boldsymbol{P} \end{bmatrix} + \begin{bmatrix} \nu_{\psi} \\ \nu_{v_n} \\ \nu_{v_e} \\ \nu_{v_d} \\ \nu_{L} \\ \nu_{l} \\ \nu_{h} \end{bmatrix} \tag{4.69}$$

To be used in Kalman filter equation the remaining important elements are covariance matrices. The initial values of these matrices can be obtained with auto-covariance least square method (ALS). The ALS method is given in section 4.1.4. To obtain best result for covariance matrices tuning strategy can also be used. Firstly, the state covariance matrix $P$ is a diagonal matrix and its elements are variances of states.

$$P = \begin{bmatrix} \sigma_{\phi}^2 & & & & & & & & \\ & \sigma_{\theta}^2 & & & & & & & \\ & & \sigma_{\psi}^2 & & & & \ddots & & \\ & & & \sigma_{v_n}^2 & & & & & \\ & & & & \sigma_{v_e}^2 & & & & \\ & & & & & \sigma_{v_d}^2 & & & \\ & & \ddots & & & & \sigma_{L}^2 & & \\ & & & & & & & \sigma_{l}^2 & \\ & & & & & & & & \sigma_{h}^2 \end{bmatrix} \tag{4.70}$$

As for the measurement covariance matrix $R$, it consists of the variances of observed states.

$$R = \begin{bmatrix} \sigma^2_{\psi,meas} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma^2_{v_n,meas} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma^2_{v_e,meas} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma^2_{v_d,meas} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma^2_{L,meas} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma^2_{l,meas} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \sigma^2_{h,meas} \end{bmatrix} \tag{4.71}$$

In this thesis, to build extended Kalman filter (EKF) general procedure which is given between equations 4.14 and 4.19 is applied to simulink model. In Kalman filter design the important parameters are system and measurement covariance matrices. To get an initial covariance matrices, auto-covariance least square method (ALS) is applied to simulink model. The ALS method procedure is used between equations 4.23 and 4.38. In EKF, another crucial elements is the linearized system model. In INS/GPS integration the linearized navigation error model is used and the linearized model is given in equations between 4.39 and 4.63.

# CHAPTER 5

# DESIGN OF ARTIFICIAL NEURAL NETWORK (ANN) AUGMENTED KALMAN FILTER

## 5.1 Introduction

In this chapter, the neural network structure and its application to navigation area will be discussed. In last twenty years, neural network shows its significance in vast majority of research areas. For example in robotics for artificial intelligence purposes, in pattern recognition for images, voices, handwriting, in control area to design controllers with neural network, etc. In this thesis neural network implementation to navigation systems will be presented. In the first part of this chapter, neural network basis, perceptron concept and learning types and procedure will be handled. In second part, the generalized neural network class, *Multilayer Perceptron* (MLP) will be studied to solve complex problems. Also famous error back propagation algorithm will be applied to MLPs to learn non-linear patterns. Finally, the union of Kalman filter navigation systems with artificial neural network and researches in this concept will be shown.

## 5.2 Neural Network Architecture and Learning Procedure

The idea behind the neural network is based on the working principle of human brain. Before going into detail of artificial neural network, it will be wise talking about the structure of human brain. Human brain has extremely complex structure to do non-linear parallel computing. To accomplish these computations, neurons are used by

central nervous system.

As in Figure 5.1, a neuron has a cell body which is called as *soma*. The signal to be transmitted to other neuron cell is generated in soma. The transmission is through the *axon*. The receiving signal part of the neuron is *dendrites*.[22]. A neuron has capability of making connections with large number of neurons by using its synaptic connections. At synaptic connections through chemical transmitters signal is transferred to other related neurons. Taking neuron as reference, for artificial neuron

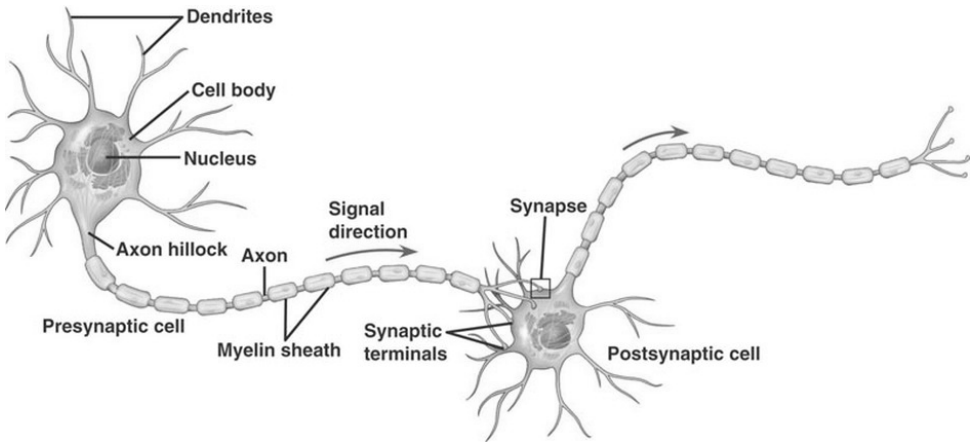



Figure 5.1: Typical neuron

network a model is proposed. Generally a *neural network* is a design to model the brain performance for particular task. In Simon Haykin's textbook, a definition for neural network is given as;

*A neural network is highly parallel distributed processor which has a natural tendency for storing experienced knowledge and making it available for use*

Therefore neural network resembles the brain in two ways[22];

- Knowledge is obtained from environment through learning process.
- Neurons connection strengths (synaptic weights in neural applications) are used to store the knowledge.

In following sections, neural network configuration is studied with details. For the further knowledge about neural networks, the reader can use reference [22].

### 5.2.1 Neuron Models and Network Structures

The artificial neuron model has four important parts. These are shown in Figure 5.2. The definitions of these parts are,

**Synapses:** They are used to symbolize the strength of neuron link. It is shown as $\omega_{kj}$. The meaning of first subscript is the neuron which gives output $y_k$. The second subscript is the input part neuron $x_j$. The value for weights can be positive or negative.

**Summation Point** It is used to sum weighted input values at junction point. The bias value is also added if the activation function input is increased or decreased.

**Bias:** External applied bias value is shown as $b_k$. Bias is used to increase or decrease the input of the activation function $v_k$.

**Activation Function:** To limit the amplitude of the output signal of neuron in desired range activations functions are used. Typically the interval is [0,1] or [-1,1].

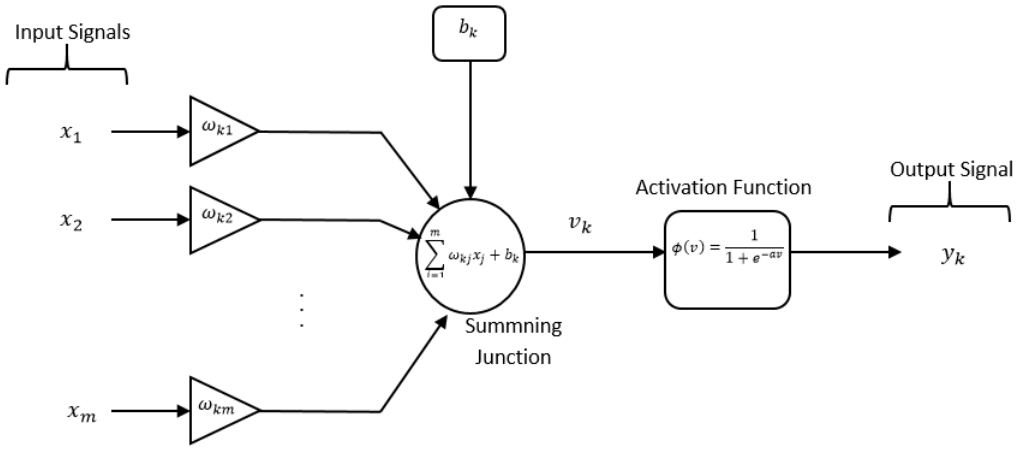



Figure 5.2: Nonlinear neuron model.

Mathematically, the neuron model can be formalized as in below. The input signals are $x_1, x_2, ..., x_m$ and synaptic weight values are $\omega_{k1}, \omega_{k2}, ..., \omega_{km}$ of neuron k.

$$u_k = \sum_{j=1}^{m} \omega_{kj} x_j \tag{5.1}$$

Here $u_k$ is the linear combiner output of the neuron k. Then the output $y_k$ can be written as,

$$y_k = \phi(u_k + b_k) \tag{5.2}$$

where $b_k$ is the bias value and $\phi(.)$ is the activation function. Also the summation of $u_k + b_k$ is called as *induced local field* or *activation potential*. The induced local field is shown as $v_k$. For the convenience if the bias term is included into equation 5.1, it can be rewritten as,

$$v_k = \sum_{j=0}^{m} \omega_{kj} x_j \tag{5.3}$$

and

$$y_k = \phi(v_k) \tag{5.4}$$

It can be thought as a new synapses added to summation term and its input value is 1 with weight $\omega_{k0} = b_k$.

**Activation Function Types**

In equation 5.2 the activation function is shown. The activation function gives the output of the neuron in terms of induced local field. The types of activation function in neuron design are listed below [22].

- **Threshold Function:** In this type activation function, the output takes 1 if induced local field value is greater than or equal to 0. Otherwise the output value of neuron takes 0 value. This type of functions are also called as *Heaviside Function*. Shown in Figure 5.3.

$$\phi(v) = \begin{cases} 1 & \text{if } v_k \geq 0 \\ 0 & \text{if } v_k < 0 \end{cases} \tag{5.5}$$

- **Piecewise-Linear Function:** Piecewise-linear function can be written as in equation 5.6. This type of function gives linear output in its linear region. This linear region can be modified with a constant to give amplification in its output.

$$\phi(v) = \begin{cases} 1 & \text{if } v_k \geq 0.5 \\ v & \text{if } -0.5 < v_k < 0.5 \\ 0 & \text{if } v_k \leq -0.5 \end{cases} \tag{5.6}$$

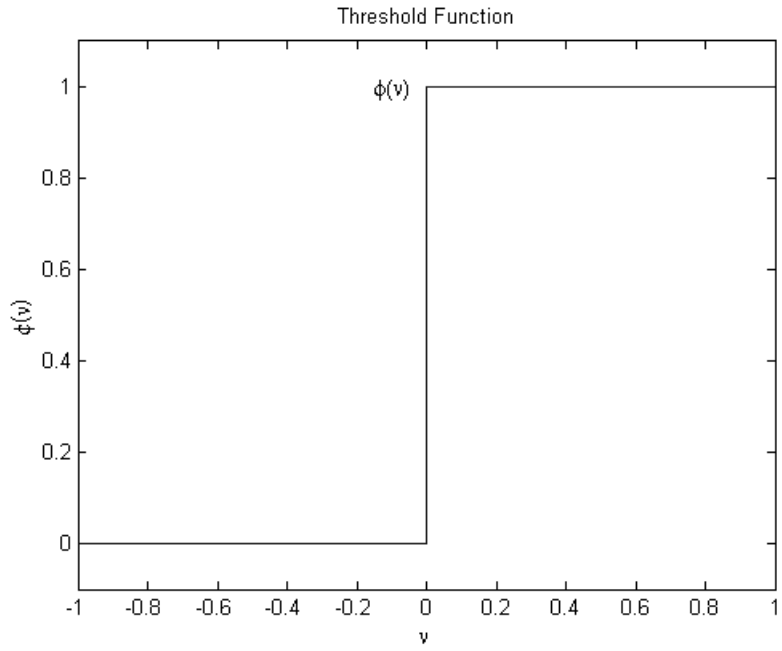



Figure 5.3: Threshold function.

- **Sigmoid Function:** In neural network design mostly the sigmoid functions are used. The reason is that sigmoid functions are differentiable functions. As an example a sigmoid function is given in 5.7.

$$\phi(v) = \frac{1}{1 + e^{-av}} \tag{5.7}$$

  Notice that by changing the $a$ value the slope characteristic function can be changed. If slope approaches its limits at infinity, the function will be continuous threshold function. In Figure 5.3, sigmoid function characteristic can be observed.

In activation functions the bounds are given as [0,1]. However for the desired range in neural network operations these values can be changes to [-1,1] in a same manner. For sigmoid functions, hyperbolic tangent function can be used.

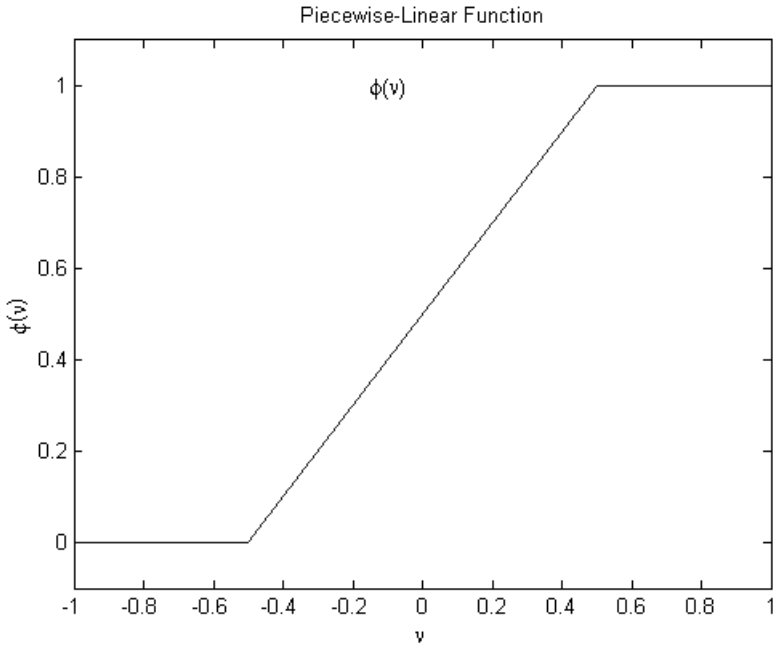



Figure 5.4: Piecewise-linear function.

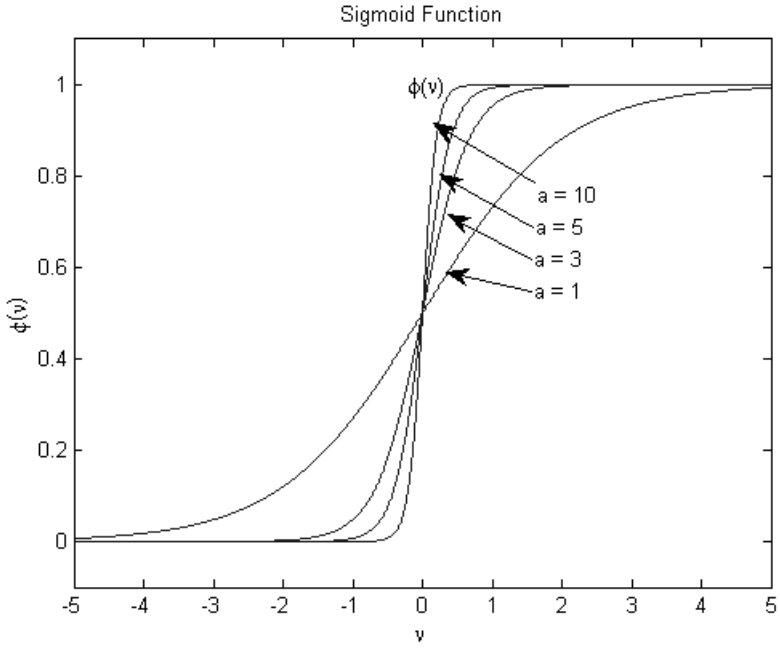



Figure 5.5: Sigmoid function.

$$\phi(v) = \tanh(av) \tag{5.8}$$

Hyperbolic tangent function is also differentiable. By changing the $a$ value the slope characteristic can be changed.

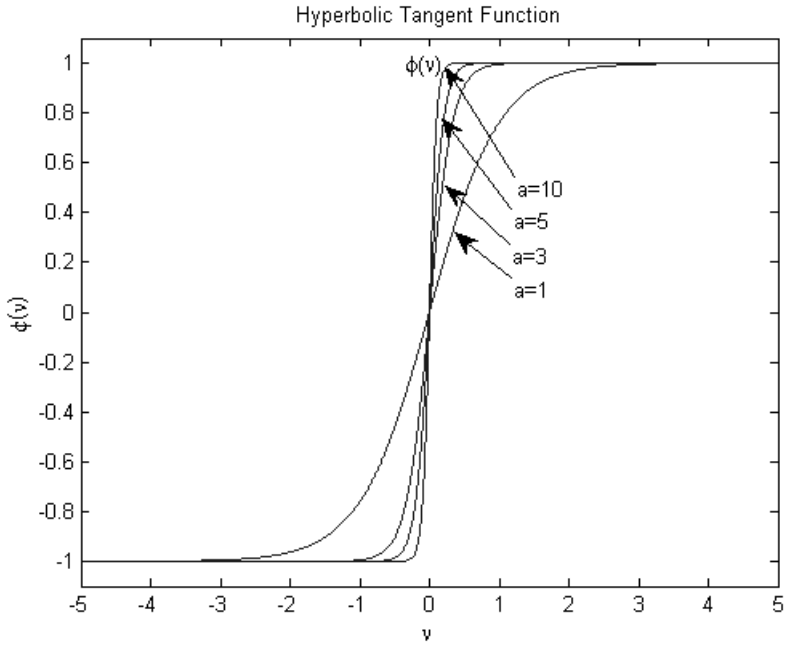



Figure 5.6: Hyperbolic tangent function.

### Network Structures

There are three different types of network architectures in literature.

- **Single-Layer Neural Networks:** The simplest form of the neural network structure is single layer neural network. It consists of input layer and output layer. A illustrated in Figure 5.7, single layer refers to output layer not input layer. Because all computations are realized on output layer.

- **Multi-Layer Neural Networks:** By adding a hidden layer to single layer neural network architecture, multilayer neural networks can be obtained. The hidden neurons make possible to recognize higher order classification among the given input. One hidden layer neural network is shown in Figure 5.8. The neural network is fully connected, in other words all nodes in one layer are

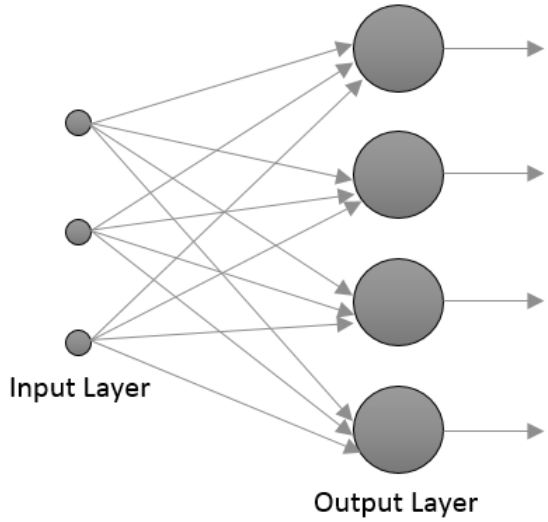



Figure 5.7: Single layer of network.

connected to another layer nodes. If some of connections are missing then system is called as partially connected.

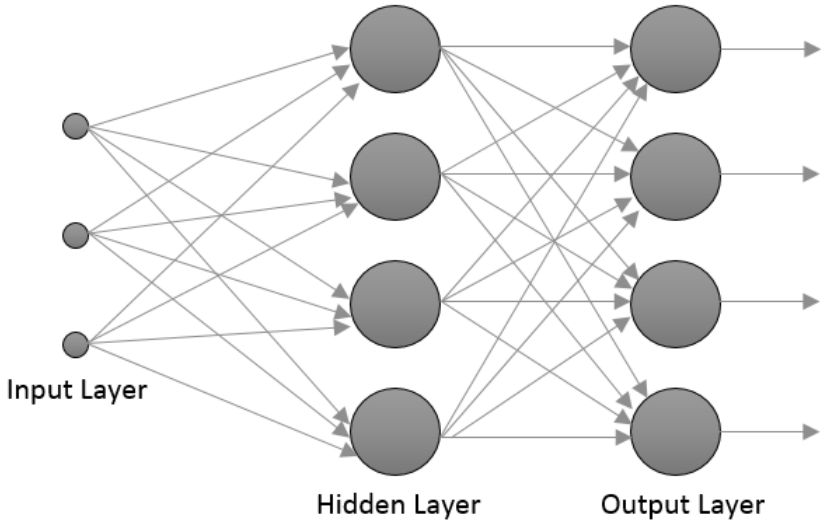



Figure 5.8: Multilayer neural network with one hidden layer.

- **Recurrent Neural Networks:** The important characteristic of recurrent neural network is feedback of its output layer to input layer. The recurrent neural network can be single layer or multilayer. The example of recurrent neural network with hidden layer is shown in Figure 5.9. The feedback increases the

learning capability of neural network. The unit delay elements sustain this non-linear feedback mechanism.

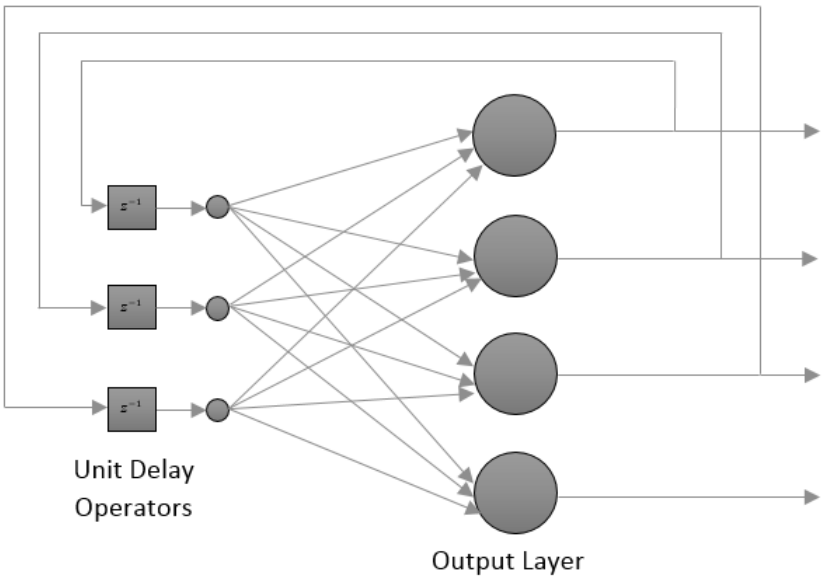



Figure 5.9: Recurrent Network with one hidden neuron.

### 5.2.2 Learning Rules and Paradigms

The crucial feature of neural network is the learning process with the help of learning algorithms. A neural network learns about its environment by changing free parameters of neurons which are synaptic weights and bias values. At each iteration process of learning algorithms neural network adapts its environment.

Learning procedure follows these three steps:

- Simulation of neural network in desired environment.
- Adapt the synaptic weights and biases through changing environments conditions.
- By using steady state weight parameters respond the environment inputs.

There are several learning algorithms for different network structures. Five basic

learning algorithms are: error-correction learning, memory based learning, Boltzmann learning, competitive learning, and Hebbian learning. In this study mainly error-correction learning will be discussed. For the details of other algorithms reference [22] can be used.

**Error-Correction Learning**

Error correction learning can be explained by using Figure 5.10. At each time step $n$ synaptic weights $\omega$ are updated. The output signal $y_k(n)$ is compared with desired output value $d_k(n)$. The produced error signal $e_k(n)$ is used to update weights.

$$e_k(n) = d_k(n) - y_k(n) \tag{5.9}$$

By using the error signal, output signal can get closer value to desired output. This

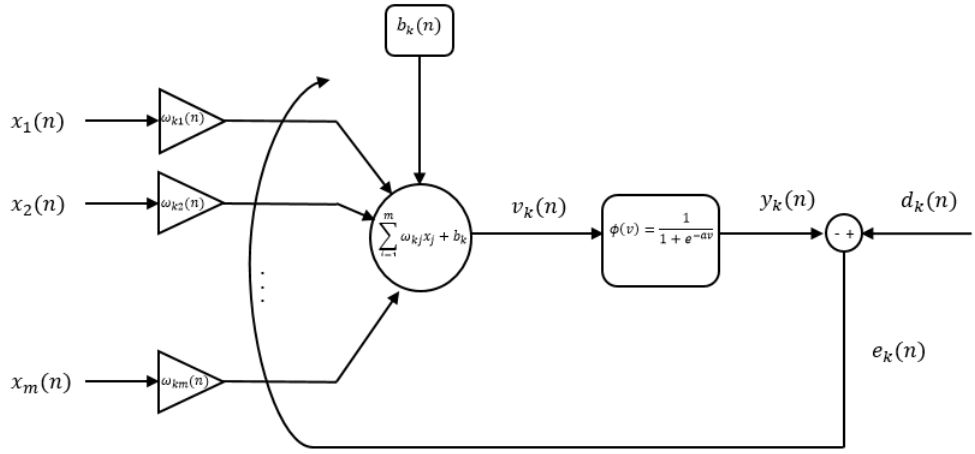



Figure 5.10: Error-correction learning.

objective is realized by minimizing the *instantaneous error energy cost function* at each step of iteration. The error cost function can be expressed as,

$$\mathcal{E}(n) = \frac{1}{2}e_k^2(n) \tag{5.10}$$

By step by step weight adjustment, error cost function reaches it steady state value and synaptic weights are stabilized. At this stage learning process must be ended.

The important question is how $\omega_{kj}$ is determined? By using delta-rule weights can be updated. According to delta-rule at iteration step $n$,

$$\delta\omega_{kj}(n) = \eta e_k(n)x_j(n) \tag{5.11}$$

Therefore the adjustment $\delta\omega_{kj}(n)$ is proportional to learning rate $\eta$, error signal $e_k(n)$, input signal $x_j(n)$ at weight branch $\omega_{kj}$. Same procedure is applied to other weights branch of neuron in same manner. Finally the adjustment value is added to current weight value for next iteration.

$$\omega_{kj}(n+1) = \omega_{kj}(n) + \delta\omega_{kj}(n) \tag{5.12}$$

The important parameter in equation 5.11 is $\eta$, learning rate parameter. By selecting proper learning rate parameter the performance of network can be increased.

**Learning Paradigms**

There are two learning paradigms. The first one is *Supervised Learning* and the second one is *Unsupervised Learning*.

1. **Supervised Learning:** Supervised learning is also known as learning with teacher. Because during the learning process input-output relation is introduced to system.

   The error-correction learning is a supervised learning example. During the supervised learning the error function or sum of error during training is the cost function of free parameters of the network system. In other words, the error cost function is a multidimensional function of synaptic weights. To find the solution on this error surface, the minimum points -local minimum or global minimum- must be detected. Thus in supervised learning the gradient of the error surface must be traced to find the minimum point in the direction of steepest descent and cost function can be minimized in its minimum location.

2. **Unsupervised learning:** In unsupervised learning there is no teacher to learn the environment contrary to supervised learning. By measuring the reaction of environment the free parameters are tuned. Therefore there is no target outputs to learn. As an example the competitive learning can be given. The input patterns are categorized in classes during learning process[22].

### 5.2.3 Perceptrons and Optimization with Least Mean Square (LMS) Algorithm

In other sections the neuron model was summarized and learning algorithms were studied. Now the simplest form of neural network, *perceptron* and numerical minimization tools for error cost function will be discussed. Therefore weights can be tuned to minimize error function in multi dimensional weight space.

**Perceptron:**

Perceptron is a neuron model with a non-linear signum function to classify the linearly separable parameters in hyperspace. By training the perceptron, learning algorithm converges to a surface boundary which is also a hyperplane. Of course if the perceptron is wanted to work properly the inputs must be linearly separable. For a single perceptron it is sufficient to identify two classes. However to classify more pattern, the neuron number must be increased. For the simplicity single neuron structure will be shown here.

For a single perceptron, the weights $\omega_1, \omega_2, ..., \omega_m$, the inputs $x_1, x_2, ..., x_m$ and fixed bias $b$ are indicated below in Figure 5.11. Then the perceptron formula will be written

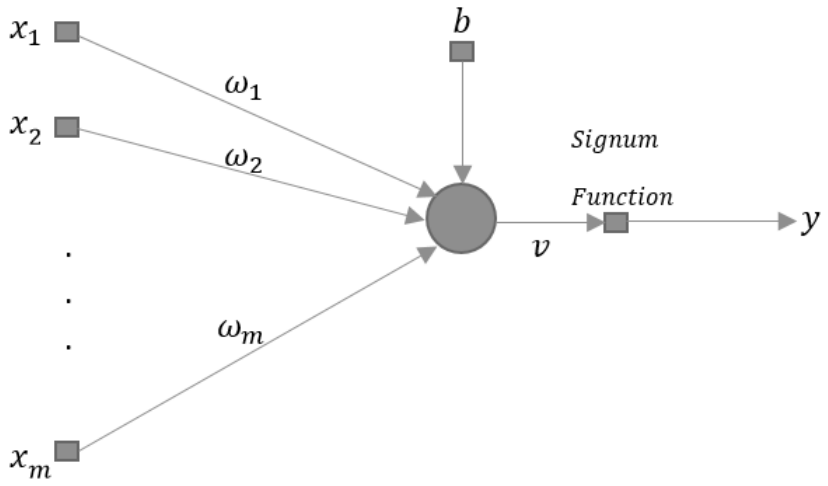



Figure 5.11: Single perceptron.

as,

$$v = \sum_{j=1}^{m} \omega_j x_j + b \tag{5.13}$$

As indicated before single perceptron can be used separate only two class. Let's say these classes $L_1$ and $L_2$. If the perceptron output is 1 then pattern is in $L_1$ class. If the output is -1 then pattern is classified in $L_2$ class. To examine the behaviour of single perceptron with error cost function, optimizations methods must be known. The optimization techniques in here can be applied to neural network to update the neuron's weights. Therefore error cost function can be minimized. For the simplicity perceptron operates in linear mode. However same principle can be applied to non-linear activation functions as well.

**Optimization Techniques**

Linear operation neuron model gives induced local field as output. Therefore at time $n$,

$$y(n) = v(n) = \sum_{k=1}^{m} \omega_k(n) x_k(n) \tag{5.14}$$

For the computation convenience, the matrix form can be written as,

$$y(n) = \boldsymbol{x}^t(n)\boldsymbol{\omega}(n) \tag{5.15}$$

where

$$\boldsymbol{\omega}(n) = [\omega_1(n)\ \omega_2(n)\ ...\ \omega_m(n)]^t$$

Now the output must be compared with the desired output $d(n)$ and the error $e(n)$ can be obtained at time $n$,

$$e(n) = d(n) - y(n) \tag{5.16}$$

Now the error cost function in equation 5.10 is critical. Let's think about the continuously differentiable $\mathcal{E}(\boldsymbol{\omega})$ function for the unknown weight parameters. For the optimal solution, the equation in equation 5.15 must be satisfied.

$$\nabla\mathcal{E}(\boldsymbol{\omega}^*) = 0 \tag{5.17}$$

where $\boldsymbol{\omega}^*$ is optimal solution. And gradient operator is,

$$\nabla\mathcal{E}(\boldsymbol{\omega}) = \left[\frac{\partial\mathcal{E}}{\partial\omega_1} \quad \frac{\partial\mathcal{E}}{\partial\omega_2} \quad ... \quad \frac{\partial\mathcal{E}}{\partial\omega_m}\right]^t \tag{5.18}$$

Staring with an initial condition $\boldsymbol{\omega}(0)$, at each iteration it is expected that weight values converge the optimum solution $\boldsymbol{\omega}^*$.

**Steepest Descent Method:** On the error surface if the direction opposite the gradient vector is selected, at every iteration weight values converge to optimum solution. Mathematically,

$$\boldsymbol{g} = \nabla \mathcal{E}(\boldsymbol{\omega}) \tag{5.19}$$

At each step $n$,

$$\boldsymbol{\omega}(n+1) = \boldsymbol{\omega}(n) - \eta \boldsymbol{g}(n) \tag{5.20}$$

where $\eta$ is the learning rate parameter and $\boldsymbol{g}(n)$ is the gradient vector at point $\boldsymbol{\omega}(n)$. Then correction term will be,

$$\begin{aligned} \Delta\boldsymbol{\omega}(n) &= \boldsymbol{w}(n+1) - \boldsymbol{\omega}(n) \\ &= -\eta \boldsymbol{g}(n) \end{aligned} \tag{5.21}$$

Notice that the process is actually error-correction learning rule as described in equation 5.11. To apply steepest descent algorithm, first order Taylor series expansion should be applied to error cost function around $\boldsymbol{\omega}(n)$ to approximate the $\mathcal{E}(\boldsymbol{\omega}(n+1))$.

$$\mathcal{E}(\boldsymbol{\omega}(n+1)) \approx \mathcal{E}(\boldsymbol{\omega}(n)) + \boldsymbol{g}^t(n)\Delta\boldsymbol{\omega}(n) \tag{5.22}$$

Then substitute equation 5.21 into equation 5.22.

$$\begin{aligned} \mathcal{E}(\boldsymbol{\omega}(n+1)) &\approx \mathcal{E}(\boldsymbol{\omega}(n)) - \eta \boldsymbol{g}^t(n)\boldsymbol{g}(n) \\ &\approx \mathcal{E}(\boldsymbol{\omega}(n)) - \eta \parallel \boldsymbol{g}(n) \parallel^2 \end{aligned} \tag{5.23}$$

From one iteration to another according to equation 5.23 the error cost function will decrease if positive $\eta$ is selected. Before selecting the $\eta$ value, the followings should be considered.

- For small $\eta$, learning procedure will be slow and searching path on error surface will be smooth.
- For large $\eta$ values, the trajectory on the error surface will be oscillatory path.
- When $\eta$ gets larger values after certain point, minimization process diverges.

**Least Mean Square Algorithm (LMS):** If error cost function is selected as *instantaneous error function*, least mean square algorithm can be applied.

$$\mathcal{E}(\boldsymbol{\omega}) = \frac{1}{2}e^2(n) \tag{5.24}$$

Differentiate error cost function $\mathcal{E}(\boldsymbol{\omega})$ with respect to $\boldsymbol{\omega}$ and by using the equation 5.16, it can be shown that,

$$\frac{\partial \mathcal{E}(\boldsymbol{\omega})}{\partial \boldsymbol{\omega}} = e(n)\frac{\partial e(n)}{\partial \boldsymbol{\omega}} \tag{5.25}$$

Remember that the neurons should be linear neuron type.

$$e(n) = d(n) - \boldsymbol{x}^t(n)\boldsymbol{\omega}(n) \tag{5.26}$$

Differentiate equation 5.26 with respect to $\boldsymbol{\omega}$,

$$\frac{\partial e(n)}{\partial \boldsymbol{\omega}(n)} = -\boldsymbol{x}(n) \tag{5.27}$$

Combining both equation 5.25 and 5.27.

$$\begin{aligned} \frac{\partial \mathcal{E}(\boldsymbol{\omega})}{\partial \boldsymbol{\omega}} &= -x(n)e(n) \\ \boldsymbol{g}(n) &= -x(n)e(n) \end{aligned} \tag{5.28}$$

Finally put this expression into 5.20, LMS algorithm is obtained.

$$\boldsymbol{\omega}(n+1) = \boldsymbol{\omega}(n) + \eta \boldsymbol{x}(n)e(n) \tag{5.29}$$

Initialization of the LMS algorithm can be made by setting zero values into weight vector. The expansion of LMS algorithm will be used in multilayer perceptron section. During the back propagation algorithm of the multilayer neural system, LMS algorithm will be useful.

## 5.3 Multilayer Perceptron (MLP)

In previous section the basics of perceptron have been studied. Classification of linearly separable classes is made by using single layer perceptron. However in complex non-linear cases single layer perceptron is not sufficient. Therefore multilayer perceptron (MLP) concept has been developed to solve complex problem by using *error back propagation algorithm*. Noticeable characteristic of multilayer

perceptron is that it consists of *hidden layers* for computational nodes besides input and output layers. The input signals flow from input layer to output layer through one or more hidden layers.

Basically back propagation algorithm is a generalized version of LMS algorithm. In back propagation algorithm two signal flows should be considered. First signal flow is the *forward pass*. During forward pass, all weights between connection of neurons are fixed and input signals affect layers through these weights layer by layer. At output layer, an output signal is produced as the reaction of neural network. Second flow is *backward pass*. In this time, the difference between actual output and observed output is subtracted. The obtained error signal is feed to the system in backward direction. In that way, the synaptic weight are updated to minimize the error difference. Hence the process is called back propagation algorithm.

In following sections, the multilayer connections and mathematics behind back propagation algorithm will be studied. And some useful techniques will be discussed to design multilayer perceptron.

### 5.3.1 Multilayer Perceptron Concept and Notation

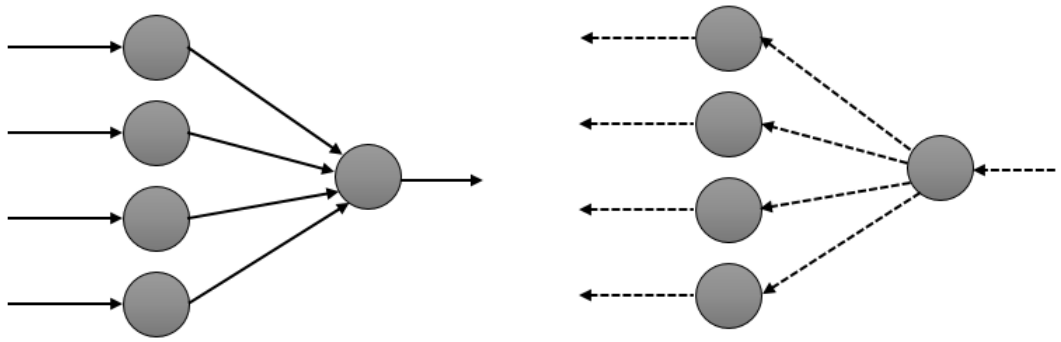
Multilayer architecture is shown in Figure 5.8. The network in here is fully connected and has one hidden neuron layer. The hidden layer number might be increased according to complexity of the problem. There are three important characteristics of multilayer perceptron to distinguish from single layer perceptron.

- Nonlinear activation function.
- One or more hidden layers.
- High degree of connectivity.

Among these characteristics, to accomplish the learning and extracting meaningful information from inputs hidden layers have a crucial role. During multilayer process, two types of signals are emerged.

1. **Function Signals:** Function signals come to input layer first and propagate in forward direction until the end of output layer. The function signals are also called input signals. However the reason why they are called as function signal is that for each neuron which the signal passes the signal is calculated as a function of synaptic weights and the input to the related neuron. In Figure 5.12a shows function signals.

2. **Error Signals:** At the output layer of network error signal is produced by extracting output from desired output value. And error signal is feed backwards layer by layer through network. These backward signals are called error signals, shown in 5.12b.



(a) Function signals. (b) Error signals.

Figure 5.12: The direction of two signals:forward and backward propagation.

Before getting into mathematical aspect of back propagation algorithm, to ease with the mathematical burden, notations through mathematical equations are listed below,

- $i$,$j$,$k$ indices show different neurons in network. Neuron $j$ lies right of neuron $i$ layer and neuron $k$ lies right of neuron $j$ layer. It is assumed that neuron $j$ is hidden neuron.

- $n$ represents time step for each iteration.

- $\mathcal{E}(n)$ refers to instantaneous error function which is sum of error squares at iteration $n$.

- $e_j(n)$ shows the error signal at the output of neurons $j$ at iteration $n$.
- $d_j(n)$ shows desired response for neuron $j$.
- $y_j(n)$ shows the function signal at the output of neuron $j$ at iteration $n$.
- $\omega_{ji}(n)$ refers to synaptic weight connection from output of neuron $i$ to input of neuron $j$ at iteration $n$. The correction for synaptic weight is shown as $\Delta\omega_{ji}(n)$.
- $v_j(n)$ is induced local field of neuron j at time step $n$.
- $\phi_j(n)$ is activation function of neuron $j$.
- $b_j$ is the bias value of neuron $j$. It is also taken into account by a synapse weight $\omega_{j0} = b_j$ and fixed input is 1.
- $x_i(n)$ is the $i$th element of input vector.
- $o_k(n)$ is the overall output vector.
- $\eta$ is the learning rate parameter.
- $m_l$ is the size in layer $l$ of multilayer perceptron network where $l = 0, 1, .., L$ and $L$ is the depth of network. For example, $m_0$ is the input layer and $m_1$ is the first hidden layer and $m_L$ is the output layer.

### 5.3.2 Back Propagation Algorithm

In previous sections instantaneous energy function has been presented. Briefly, the instantaneous energy function $\mathcal{E}(n)$ is obtained by summing the $\frac{1}{2}e_j^2(n)$ over all output neurons in output layer. Therefore it can be written,

$$e_j(n) = d_j(n) - y_j(n) \tag{5.30}$$

$$\mathcal{E}(n) = \frac{1}{2}\sum_{j\in C} e_j^2(n) \tag{5.31}$$

where $C$ symbolizes the entire output neuron. The objective of learning process is minimizing the error function by changing the weight parameters. To accomplish the minimization similar method to LMS algorithm will be used. The weights of neural network will be updated pattern-by-pattern base for one epoch in other words at the

end of presentation of all training data, weights will be updated to theirs new value for minimization.

In calculation of weight update process, there are two cases. First case is that neuron $j$ is in output layer. This case is simple because of the known desired output value. Second case is that neuron $j$ is in hidden layer. This case is little complicated than output neuron case. Even though there is no direct contribution to error in hidden layer, they share the responsibility for error in output layer.

**Case 1: Neuron $j$ is output node**

Consider the neuron $j$ in output layer, then its induced local field will be expressed as in below,

$$v_j(n) = \sum_{i=0}^{m} \omega_{ji}(n) y_i(n) \tag{5.32}$$

where $m$ is the total number of inputs.Notice that bias also included with $\omega_{j0}$ and $y_0 = 1$. Therefore the output $y_j(n)$ is,

$$y_j(n) = \phi_j(v_j(n)) \tag{5.33}$$

In a similar way with LMS algorithm the synaptic weight correction $\Delta\omega_{ji}(n)$ is proportional to $\partial\mathcal{E}(n)/\partial\omega_{ji}(n)$. To find $\partial\mathcal{E}(n)/\partial\omega_{ji}(n)$, chain rule can be utilized.

$$\frac{\partial\mathcal{E}(n)}{\partial\omega_{ji}(n)} = \frac{\partial\mathcal{E}(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} \frac{\partial v_j(n)}{\partial\omega_{ji}(n)} \tag{5.34}$$

To calculate the equation 5.34, all partial values must be calculated. From equation 5.31,

$$\frac{\partial\mathcal{E}(n)}{\partial e_j(n)} = e_j(n) \tag{5.35}$$

From equation 5.30, differentiating both sides,

$$\frac{\partial e_j(n)}{\partial y_j(n)} = -1 \tag{5.36}$$

From equation 5.33,

$$\frac{\partial y_j(n)}{\partial v_j(n)} = \phi_j^{'}(v_j(n)) \tag{5.37}$$

Finally from equation 5.32,

$$\frac{\partial v_j(n)}{\partial\omega_{ji}(n)} = y_i(n) \tag{5.38}$$

Combine all expressions and put into equation 5.34.

$$\frac{\partial \mathcal{E}(n)}{\partial \omega_{ji}(n)} = -e_j(n)\ \phi_j^{'}(v_j(n))\ y_i(n) \tag{5.39}$$

Now the correction $\Delta\omega_{ji}(n)$ can be expressed as,

$$\Delta\omega_{ji}(n) = -\eta \frac{\partial \mathcal{E}(n)}{\partial \omega_{ji}(n)} \tag{5.40}$$

where $\eta$ is learning parameter of the back propagation algorithm. If equation 5.40 can be rewritten,

$$\Delta\omega_{ji}(n) = \eta\ \delta_j(n)\ y_i(n) \tag{5.41}$$

where $\delta_j(n)$ is local gradient and defined by,

$$\begin{aligned} \delta_j(n) &= -\frac{\partial \mathcal{E}(n)}{\partial v_j(n)} \\ &= -\frac{\partial \mathcal{E}(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} \\ &= e_j(n)\phi_j^{'}(v_j(n)) \end{aligned} \tag{5.42}$$

**Case 2: Neuron $j$ is hidden node**

As shown in Figure 5.13, the problem in hidden layer neuron is that there is no desired output value to compare with output of related hidden neuron. However to apply the back propagation algorithm to hidden layer, the error signal from all connected neuron to hidden neurons are taken to compute weight correction. In output neuron local gradient has been shown. To apply local gradient to hidden neuron, it should be rewritten as,

$$\begin{aligned} \delta_j(n) &= -\frac{\partial \mathcal{E}(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} \\ &= -\frac{\partial \mathcal{E}(n)}{\partial y_j(n)} \phi_j^{'}(v_j(n)) \end{aligned} \tag{5.43}$$

To calculate partial derivative of the error $\partial \mathcal{E}(n)/\partial y_j(n)$ from Figure 5.13, it can be written,

$$\mathcal{E}(n) = \frac{1}{2} \sum_{k \in C} e_k^2(n) \tag{5.44}$$

Now, differentiate the equation 5.44 with respect to $y_j(n)$. One important notation should be taken into consideration. Contrary to case 1, $j$ neuron is hidden neuron in

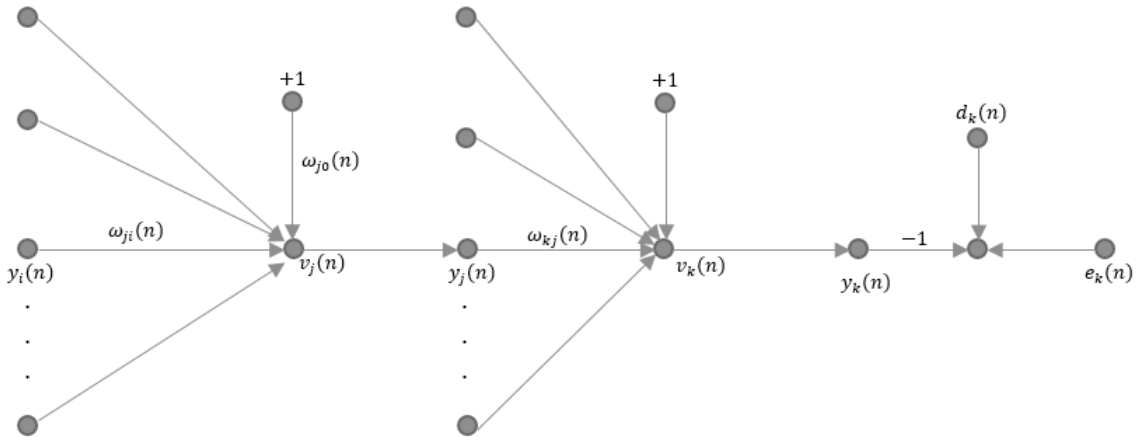



Figure 5.13: Hidden neuron $j$ is connected to output neuron $k$.

case 2.

$$\frac{\partial \mathcal{E}(n)}{\partial y_j(n)} = \sum_k e_k(n) \frac{\partial e_k(n)}{\partial y_j(n)} \tag{5.45}$$

By using the chain rule,

$$\frac{\partial \mathcal{E}(n)}{\partial y_j(n)} = \sum_k e_k(n) \frac{\partial e_k(n)}{\partial v_k(n)} \frac{\partial v_k(n)}{\partial y_j(n)} \tag{5.46}$$

From figure the difference between desired output and output of neuron $k$ is,

$$\begin{aligned} e_k(n) &= d_k(n) - y_k(n) \\ &= d_k(n) - \phi_k(v_k(n)) \end{aligned} \tag{5.47}$$

Therefore the partial derivative in equation 5.46 will be,

$$\frac{\partial e_k(n)}{\partial v_k(n)} = -\phi_k{}'(v_k(n)) \tag{5.48}$$

For the second partial derivative in equation 5.46, induced local field of neuron $k$ can be used.

$$v_k(n) = \sum_{j=0}^{m} \omega_{kj}(n) y_j(n) \tag{5.49}$$

where $m$ is the total number of input to neuron $k$. Differentiating equation 5.49 with respect to $y_j(n)$ gives,

$$\frac{\partial v_k(n)}{\partial y_j(n)} = \omega_{kj}(n) \tag{5.50}$$

Now equation 5.46 can be rewritten as below,

$$\frac{\partial \mathcal{E}(n)}{\partial y_j(n)} = -\sum_k e_k(n)\ \phi_k'{}'(v_k(n))\ \omega_{kj}(n)$$
$$= -\sum_k \delta_k(n)\ \omega_{kj}(n) \tag{5.51}$$

Finally, local gradient $\delta_j(n)$ can be found by using equation 5.51 in 5.43.

$$\delta_j(n) = \phi_j^{'}(v_j(n)) \sum_k \delta_k(n)\ \omega_{kj}(n) \tag{5.52}$$

The summation part of the equation 5.52 is an important quantity for back propagation algorithm in hidden layer. The first term $\delta_k(n)$ in summation shows the contribution of information $e_k(n)$ which belongs to next hidden layer or output layer error term. The second term $\omega_{kj}(n)$ shows the contribution level of error knowledge that comes from next layer neurons.

Generally, back propagation algorithm can be summarized as below,

$$\Delta\omega_{ji}(n) = \eta\ \delta_j(n)\ y_i(n) \tag{5.53}$$

To apply equation 5.53 two scenarios should be considered:

1. If back propagation algorithm is applied to output neuron, local gradient $\delta_j(n)$ equals to product of $\phi_j^{'}(v_j(n))$ and $e_j(n)$.

2. If back propagation algorithm is applied to hidden neuron, $\delta_j(n)$ equals to product of $\phi_j^{'}(v_j(n))$ and weighted sum of $\delta_k(n)$ which is computed for next hidden layer or output layer.

This back propagation of error is computed through each layer until all synaptic weights are updated. Also input pattern is fixed during each forward and backward pass operation.

In backward operation the computation of derivative of activation function is required. For the derivative computation, activation function must be differentiable. Therefore mostly sigmoidal functions are used as activation functions. Two types of sigmoidal functions are given below. These are logistic functions and hyperbolic tangent function.

**Logistic Functions:**

The form of logistic functions is,

$$\phi_j(v_j(n)) = \frac{1}{1+e^{-av_j(n)}} \quad a > 0 \text{ and } -\infty < v_j(n) < \infty \tag{5.54}$$

The range of output of logistic function is $0 \leq y_j \leq 1$. If logistic function is differentiated,

$$\phi_j^{'}(v_j(n)) = \frac{ae^{-av_j(n)}}{(1+e^{-av_j(n)})^2} \tag{5.55}$$

By manipulating equation 5.55, it can be written as

$$\phi_j^{'}(v_j(n)) = ay_j(n)(1-y_j(n)) \tag{5.56}$$

If neuron $j$ is located at the output layer, local gradient is

$$\begin{aligned}\delta_j(n) &= e_j(n)\phi_j^{'}(v_j(n)) \\ &= a(d_j(n)-y_j(n))y_j(n)(1-y_j(n))\end{aligned} \tag{5.57}$$

If neuron j is in hidden layer, then local gradient will be,

$$\begin{aligned}\delta_j(n) &= \phi_j^{'}(v_j(n))\sum_k \delta_k(n)\omega_{kj}(n) \\ &= ay_j(n)(1-y_j(n))\sum_k \delta_k(n)\omega_{kj}(n)\end{aligned} \tag{5.58}$$

**Hyperbolic Tangent Functions:**

The form of hyperbolic tangent function is,

$$\phi_j(v_j(n)) = a\tanh(bv_j(n)) \quad a,b>0 \tag{5.59}$$

The derivative of the hyperbolic tangent function with respect to $v_j(n)$ is,

$$\begin{aligned}\phi_j^{'}(v_j(n)) &= ab\ sech^2(bv_j(n)) \\ &= ab(1-\tanh^2(bv_j(n))) \\ &= \frac{b}{a}(a-y_j(n))(a+y_j(n))\end{aligned} \tag{5.60}$$

If hyperbolic activation function is in output layer, local gradient is,

$$\begin{aligned}\delta_j(n) &= e_j(n)\phi_j^{'}(v_j(n)) \\ &= \frac{b}{a}(d_j(n)-y_j(n))(a-y_j(n))(a+y_j(n))\end{aligned} \tag{5.61}$$

If hyperbolic activation function is in hidden neuron, then local gradient is,

$$\begin{aligned}\delta_j(n) &= \phi_j'(v_j(n)) \sum_k \delta_k(n)\omega_{kj}(n) \\ &= \frac{b}{a}(a - y_j(n))(a + y_j(n)) \sum_k \delta_k(n)\omega_{kj}(n)\end{aligned} \tag{5.62}$$

### 5.3.3 Sequential and Batch Learning

In learning process, training set presentation to multilayer perceptron is an important feature to accomplish successful learning. The learning process is realized in each epoch. Epoch is the one complete presentation of training set. Until all synaptic weights reach a certain value and average error function takes its minimum value the learning process continues epoch by epoch base. One good recommendation about presentation of training set is randomized presentation of set. This randomization makes the weight search stochastic and avoids the limit cycles during learning [22].

For a selected training set, there are two ways of back propagation algorithm to progress.

1. Sequential Mode: This process also is called as *online mode*. After presenting each training example, weights are updated. For example, among $N$ training examples for an epoch, first training example is given to back propagation algorithm. After first forward and backward pass the synaptic weights are updated. Then second training example given to system in a same manner. Until last training example in epoch process is repeated.

2. Batch Mode: Different from sequential mode, weight update is fulfilled after presentation of all training example in other word after one epoch. Also the cost function is defined as,

$$\mathcal{E}_{av} = \frac{1}{2N} \sum_{n=1}^{N} \sum_{j \in C} e_j(n)^2 \tag{5.63}$$

   where inner summation is for all output layer error and outer summation is for entire training set. According to $\mathcal{E}_{av}$, the weight update will be

$$\Delta\omega_{ji} = -\eta \frac{\partial \mathcal{E}_{av}}{\partial \omega_{ji}} \tag{5.64}$$

$$\Delta\omega_{ji} = -\frac{\eta}{N}\sum_{n=1}^{N} e_j(n)\frac{\partial e_j(n)}{\partial \omega_{ji}} \tag{5.65}$$

In batch mode, it should be taken into consideration that weight adjustment $\Delta\omega_{ji}$ is applied after entire training set has been submitted.

Among these presentation methods, sequential method is preferred. Because sequential mode is easy to implement and has effective solutions for complex problems. Also sequential mode needs less local storage because of on-line operation advantage. In case of data set including several copies of training example, sequential mode provides good result with respect to batch learning. However for accurate estimate of gradient vector batch mode learning can be used.

### 5.3.4 Stopping of Learning Process

In literature there is no special stopping criteria for back propagation algorithm. However by using local minimum or global minimum of error surface, some clues about stopping criteria can be used [22]. Firstly, gradient vector can be utilized to stop the learning if gradient vector $g(\boldsymbol{\omega})$ of the error surface is zero with respect to weight vector $\boldsymbol{\omega}$. However this criteria needs the computation of gradient vector $g(\boldsymbol{\omega})$ of error surface. Second criteria is using the average error cost function $\mathcal{E}_{av}$. In this case back propagation algorithm is considered to have minimum value when rate of change in average error function per epoch is sufficiently small. The disadvantage of this criteria is that the termination of learning process may occur in early stages such that learning may not accomplish properly.

Another useful performance criteria is generalization performance. The learning process should be stopped after generalization criteria is reached. More detailed explanation can be found in reference [22] for generalization performance.

### 5.3.5 Techniques for Performance of Back Propagation Algorithm

There are some methods to improve back propagation algorithm performance. In following paragraphs these methods are listed.

1. **Sequential and bath mode:** Compared with bath mode sequential mode is much faster and sequential mode is easy to implement. Especially for complex, large and redundant training samples sequential mode should be preferred.

2. **Randomizing the information:** To make learning process successful information content should be larger for back propagation algorithm. If training set has selected such that training error will be larger, back propagation algorithm performance will be higher. In sequential mode the best way to achieve this is randomization of training data set at each epoch.

3. **Activation Function:** When selected sigmoid activation function for neural network is selected, mostly antisymmetric functions are preferred instead of non-symmetric functions. The reason is that with antisymmetric activation functions back propagation algorithm shows faster learning process.

   The antisymmetric conditions is,

   $$\phi(-v) = -\phi(v) \tag{5.66}$$

   As an example hyperbolic tangent function can be given.

   $$\phi(v) = a\tanh(bv) \tag{5.67}$$

4. **Target Values:** The target values $d_j$ should be chosen within the range of activation function. Otherwise the weight tends to go infinity during back propagation algorithm. Learning process will be slow and neurons will saturate.

5. **Normalizing input:** Before training process is realized, training data set should have a mean value which is close to zero. Also training examples should be uncorrelated and their covariances should approximately equal. Therefore synaptic weights in neural network learn training set at the same speed.

6. **Initialization:** To initialize the synaptic weights both larger and small initialization values should be avoided. For larger initial values, neurons will be saturated and learning process will be slow because of that local gradient takes small values to reach local minimum. For small values back propagation

algorithm will operate around origin and because of that origin is a saddle point on error surface learning process will be slow. Therefore the suitable initialization values are between these two regions.

7. **Learning rates:** Smaller learning rate $\eta$ makes smaller changes in searching synaptic weights to minimize error cost function. Therefore slow learning will occur during training. On the other hand, if learning rate is selected high, searching of minimal weights values will be oscillatory and learning process becomes unstable. Therefore large and small learning rate parameters should be avoided.

   Moreover in multilayer perceptron design, the last layer towards to output layer should take smaller learning rate parameter than front neuron layers. The reason for that at output layer local gradient takes larger values than front end of neural network.

## 5.4 Artificial Neural Network Augmented Kalman Filter

As mentioned in previous sections, INS/GPS integration system with other sensors such as barometers, magnetometers, etc. are highly popular in navigation systems. To fuse these sensors mostly Kalman filter is utilized. However INS/GPS integration systems show great error in determination of positions during GPS outages. Also Kalman filter needs accurate stochastic model of navigation system and measurement model of all sensors used in sensor fusion. Because of that complications in navigation system, studies focus on *Artificial Intelligence* systems to find feasible solution.[23]

There are several AI based navigation studies in literature which use artificial neural networks. Kaygısız et al.[24] introduced a position estimation ANN based GPS/INS systems. ANN estimates the position variables based on GPS/INS system behaviour. However velocity and attitude variables was not included into ANN estimation variables. Noureldin et al.[25] introduced radial basis function (RBF) based neural network to estimate position in INS/GPS system during GPS signal lost. The method also uses wavelet analysis to compare INS and GPS outputs at different resolution

levels before ANN training. Naser El-Sheimy et al.[26] introduced two ANN based INS/DGPS integration architectures by using multilayer perceptron with gradient based learning algorithm. First architecture was used for position update (PUA) and the second one is position and velocity update architecture (PVUA). Semeniuk et al.[27] proposed RBF neural network to achieve position accuracy during GPS outages. Position and velocity knowledge were used as input to neural network. At the output the position accuracy was determined. The modelled navigation system operates as a Kalman filter during GPS outages. Chiang et al.[28] introduced cascade-correlation networks (CCN) for low-cost MEMS inertial sensor with GPS navigation system. Also CCN system structure was compared with multilayer perceptron neural network (MLPNN) as an alternative scheme. According to results Kalman filter showed better positioning accuracy during 5 to 10 s of GPS signal outage period. However because of the noisy measurement accuracy was lost in time. By using the CCN system this accuracy lost was recovered. And both MLNN and CCN system showed similar performance in same scenario. Chanthalansy [29] showed 2D positioning solution for low-cost INS/GPS systems with hybrid AI/KF tightly coupled architecture. The method proposed that augmented Kalman filter with RBFNN to improve integration process of navigation algorithm when GPS signal was lost. According to results the performance of hybrid system is proved to be more effective in reducing position errors during 60 seconds GPS outages. A. Noureldin, A. El-Shafie and M. Bayoumi [30] introduced new method for AI based INS/GPS structure. According to this study, all ANN based navigation systems are based on INS error at a certain time to train neural network and past values of INS error values were not considered. Therefore in this study input-delay neural network was proposed to model both INS position and velocity errors based on current and some past values of INS position and velocity samples. This study showed that proposed method is more efficient in long GPS outages. Malleswaran et al. [31] compares higher order neural networks (HONNs) with feed forward neural networks (FFNNs). The study showed that the performance of HONNs provides satisfactory results during GPS signal lost with respect to FFNNs. Another interesting study in this field is Zhang et al. [23]. In this research INS/GPS/magnetometer sensor fusion was used. As for intelligence navigation neural network structure with wavelet de-noising technology was utilized. To prevent INS system degradation during GPS

outages feedforward neural network was used to sustain position and velocity errors to Kalman filter. With wavelet multi resolution analysis noise was removed from training signal and neural network approximation was improved. Chen et al. [32] introduced strong tracking Kalman filter (STKF) with wavelet neural network (WNN) for INS error compensation during GPS outages. When GPS signal is blocked, WNN/STKF algorithm eliminates the INS error and position estimation will be provided by using wavelet neural network.

These are some interesting and enlightening researches in intelligence navigation area. Within the light of this perspective, in this thesis multilayer perceptron neural network structure (MLPNN) with back propagation algorithm was used to improve both position and velocity error estimation when GPS was blocked. The detailed explanation of system structure was given in following section.

### 5.4.1 Multilayer Perceptron Neural Network with Kalman Filter in INS/GPS Navigation

In this thesis to compensate the error growing during GPS outages of INS/GPS/magnetometer sensor fusion system, multilayer perceptron neural network (MLPNN) was chosen. The reason why MLPNN was used is that MLPNN is easy to implement with back propagation algorithm. Also on-line learning procedure was used in back propagation algorithm because of the computation speed and needing less memory storage.

The INS/GPS/magnetometer navigation with MLPNN system structure is given in Figure 5.14. The mathematical model of accelerometer, gyroscope, GPS and magnetometer are based on the equations in chapter 3. INS system provides attitude, velocity and position information. Also GPS provides velocity and positions information and magnetometer provides heading angle. All collected informations are given to Kalman filter. Finally obtained state errors from Kalman filter are feed into INS system to correct heading, velocity and position values. When GPS is functional, NN will be in training stage. All collected velocity and position data are given as input to the NN system and corresponding velocity and position error values are given as desired output to the NN system. NN is trained with collected

data. The mathematical relation between each position and velocity and position and velocity error can be shown as below.

$$\delta V_n = f_{V_n}(V_n, \omega_{ji}, \omega_{kj}) \tag{5.68}$$

$$\delta V_e = f_{V_e}(V_e, \omega_{ji}, \omega_{kj}) \tag{5.69}$$

$$\delta V_d = f_{V_d}(V_d, \omega_{ji}, \omega_{kj}) \tag{5.70}$$

$$\delta L = f_L(L, \omega_{ji}, \omega_{kj}) \tag{5.71}$$

$$\delta l = f_l(l, \omega_{ji}, \omega_{kj}) \tag{5.72}$$

$$\delta h = f_h(h, \omega_{ji}, \omega_{kj}) \tag{5.73}$$

Where $\omega_{ji}$ and $\omega kj$ symbolize that the input layer and hidden layer weights respectively. Therefore totally six MLPNN structure are constructed to learn error characteristics of INS/GPS navigation system and update the MLPNN weights during GPS online.

As shown in Figure 5.15, when GPS signal is lost, NN is used instead of GPS model. In this way velocity and position error parameters are given to Kalman filter to generate state error values. Therefore the deviation of velocity and position estimation is eliminated.

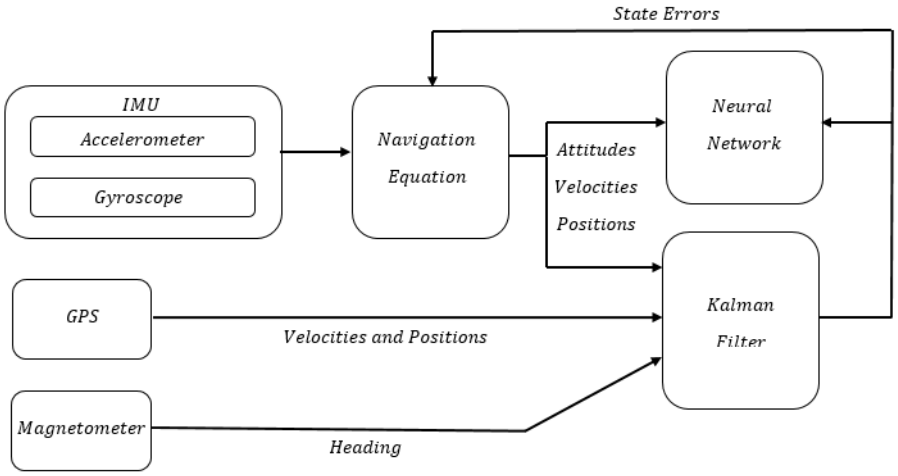



Figure 5.14: Neural network training when GPS signal is available

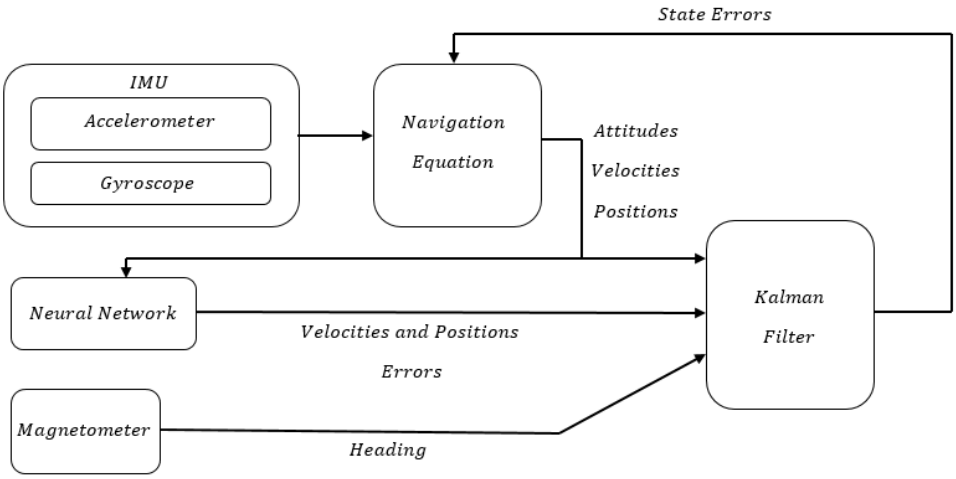



Figure 5.15: Neural network during GPS outages

The Matlab simulink scheme of the system model is given in appendix C. For further detailed explanation of the simulink structure and mathematical model parameters of the system refer to chapter 6.

In this thesis, for artificial neural network application multilayer perceptron neural network (MLPNN) is selected because of its esay of use and computationally its convenience for online learning. For each north, east and down velocity and each latitude, longitude and height position parameters MLPNN structure with one hidden layer is constituted. The important algorithm in MLPNN is back propagation algorithm and in simulink part the equations between 5.30 and 5.53 are used for back propagation learning algorithm. As for the sigmoid function in both layer tangent hyperbolic function given in 5.8 is selected because of its non-linear characteristic. Also MLPNN is applied to simulink model with sequential mode because the computational performance is high for online application in sequential learning.

# CHAPTER 6

# SIMULINK MODEL AND TEST RESULTS

## 6.1 Introduction

In this chapter, the detailed explanation of simulink architecture which consists of sensor models, mechanization algorithm, Kalman filter algorithm and neural network model will be given. Firstly the role of each subsystems in simulink model will be discussed. To analyse the Kalman filter performance two path results which are obtained from TAI helicopter simulator will be shown. The first path is simple circular path and the second path is relatively a complex one when compared the former. Secondly, instead of the simulator flight data, land vehicle test data and quadrocopter flight data are feed to the system separately. Both land path and flight path data from its accelerometer, gyroscope, magnetometer and GPS were collected and tested on simulink model. In this case in addition to navigation model neural network structure is also added. For certain time intervals the GPS signal is jammed and the neural network performance results are compared with the GPS signal lost case without neural network.

## 6.2 Simulink Model Test with Helicopter Simulator Flight Data

The navigation model mainly consists of 6 subsystems. These are,

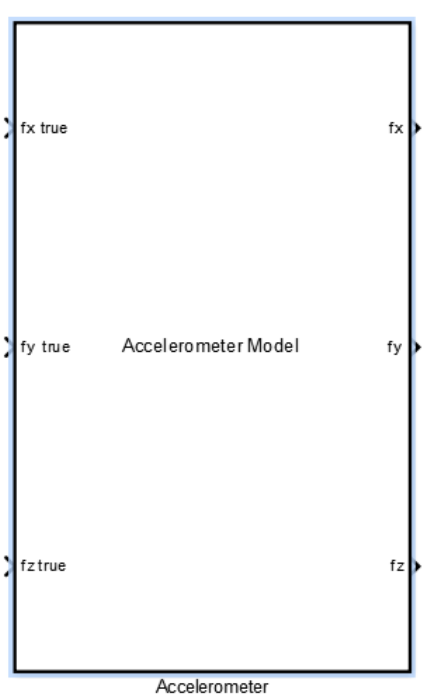
- Accelerometer model
- Gyroscope model

- Magnetometer model
- GPS model
- Navigation mechanization algorithm
- Kalman filter algorithm

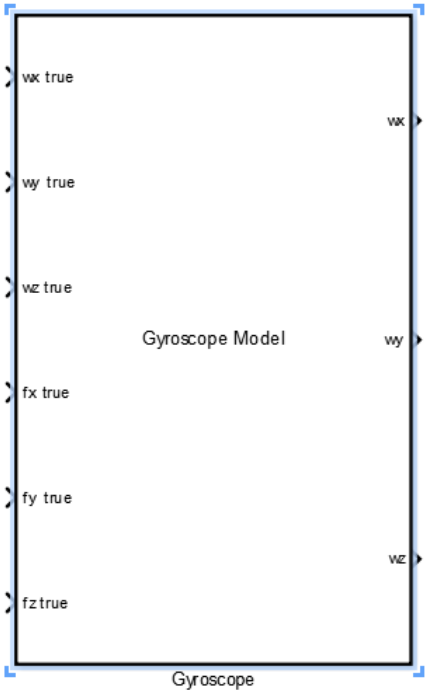
The brief explanation of these subsystems are given below.

**Accelerometer Model:**

The mathematical representation of an accelerometer error model was given in equation 3.2 in chapter 3. Equation 3.2 is applied to accelerometer model which is shown in Figure 6.1a. The input parameters of the accelerometer model are the helicopter body axis acceleration values without noise. By adding the error sources, acceleration values with noise are obtained in each body axis.Accelerometer error values given in Table 6.1.[1].




(a) Accelerometer simulink model




(b) Gyroscope simulink model

Table 6.1: Accelerometer error sources

| Error source | $1\sigma$ values |
|---|---|
| Accelerometer fixed biases $B_{ax}$, $B_{ay}$, $B_{az}$ | 10 milli-g |
| Accelerometer scale factors $S_{ax}$, $S_{ay}$, $S_{az}$ | 0.3 % |
| Accelerometer cross-coupling $M_{Axz}$, $M_{Axy}$, $M_{Ayz}$ | 0.1 % |
| Accelerometer white noise $\omega_A$ | 0.01 g |

**Gyroscope Model:**

The mathematical error characteristic of gyroscope was given in equation 3.6 in chapter 3. The equation 3.6 is applied to gyroscope model to obtain error values. The input parameters of gyroscope are body angular rates and acceleration values of helicopter simulator. Acceleration values are also included in gyroscope model to simulate coupling effect. For output values body axis angular rates with noise are extracted. The model figure is given in Figure 6.1b. Gyroscope error sources are given in Table 6.2.[1].

**Magnetometer Model:**

Magnetometer model gives the magnetic vector components in each body axis according to coordinate values on Earth. To obtain these magnetic vector values magnetometer models needs *World Magnetic Model*(WMM). World magnetic model gives the total magnetic intensity value of given location with inclination ad declination values. By giving these values as input to magnetometer model with transformation matrix, magnetometer model gives magnetic vector values in body axis. The mathematical calculation of magnetic vector in body axis was given in 3.76. The magnetometer model representation is given in Figure 6.2. After obtaining magnetic vector, by using equation 3.78 heading angle is obtained.

Table 6.2: Gyroscope error sources

| Error source | $1\sigma$ values |
|---|---|
| Gyroscope fixed biases $B_{gx}$, $B_{gy}$, $B_{gz}$ | 50 °/h |
| Gyroscope scale factors $S_{gx}$, $S_{gy}$, $S_{gz}$ | 0.05 % |
| Gyroscope cross-coupling $M_{gxz}$, $M_{gxy}$, $M_{gyz}$ | 0.1 % |
| Gyroscope g-dependent bias $B_{gxx}$, $B_{gxz}$, $B_{gyx}$, $B_{gyz}$, $B_{gzx}$, $B_{gzz}$ | 5 °/h/g |
| Gyroscope anisoelastic bias $B_{axx}$, $B_{ayx}$, $B_{azx}$ | 0.5 °/h/$g^2$ |
| Gyroscope white noise $\omega_G$ | 0.004 °/s |

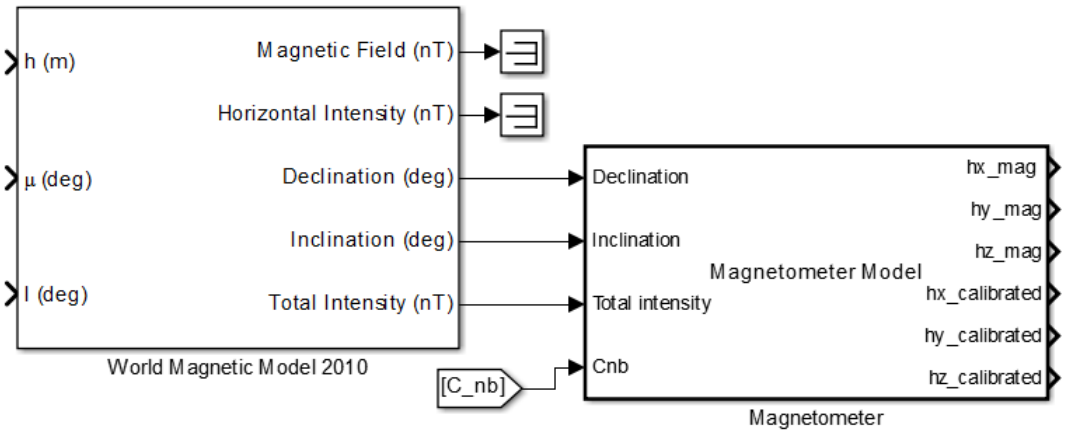



Figure 6.2: World magnetic model and magnetometer simulink model

**GPS Model:**

GPS model simulates the GPS satellite dynamics. Based on satellite pseudorange and pseudorange rate calculation receiver position and velocity values in NED frame are computed by using least square estimation technique. GPS model includes the following subsystems,

1. Satellite dynamics block:

   In this block the equations are obtained from section 3.4.4. Equations between 3.31 and 3.33 are for position of satellite and equations between 3.42 and 3.44 are for satellite velocity calculations.

2. Satellite true range computation block:

   To determine range between receiver and each satellite, equation 3.46 is used.

3. Satellite visibility calculation block:

   The visibility calculation is based on given direct line of sight angle. Within this angle satellite is counted as visible.

4. Pseudorange and pseudorange rate calculation block:

   Pseudorange and pseudorange rate calculations are given in section 3.4.5. Pseudo range and range rate are given in equation 3.53 and 3.63 respectively.

5. Receiver position and velocity computation block:

   For receiver position and velocity estimation, equation 3.65 is used.

6. Dilution of precision computation block (GDOP,HDOP,VDOP and TDOP):

   The dilution of precision parameters calculation is given in equation 3.75 for this block.

GPS model receives latitude, longitude, altitude and NED velocity values as input from simulator. By converting input values into ECEF coordinates and adding the satellite dynamics and white noise, receiver positions (Latitude,longitude,altitude) and velocities (NED velocity components) are calculated as output. The simulink model figure of GPS is given in Figure 6.3.

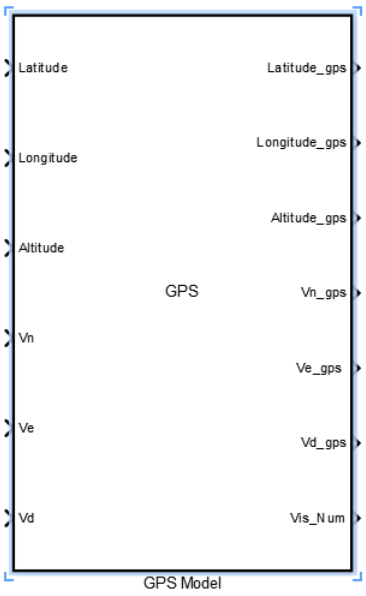



Figure 6.3: GPS simulink model

**Navigation Mechanization Algorithm:**

Navigation block computes NED velocities, latitude, longitude and altitude positions, attitude parameters and transformation matrix from body frame to NED frame by using accelerometer and gyroscope output values. Navigation block includes following subsystem,

1. WGS-84 Earth model block:

   For this block Earth error shape model given in Table 2.1. Equations between 2.63 and 2.65 are used.

2. Gravity model block:

   For gravity model section 2.5.1 can be used. The equations 2.60 and 2.61 are utilized.

3. Attitude computation block:

   Attitude computation equations for this block are given in section 2.6.1. Equations between 2.66 and 2.81 are applied to this block. This equation set is a recursive algorithm. Therefore it must be updated with each gyroscope readings in time.

4. Orthogonalization and normalization block:

   To preserve the orthogonality property the equations between 2.82 and 2.84 are used in this block.

5. Velocity computation block:

   This block consists of the acceleration terms. The equation 2.49 is used for this block.

6. Position computation block:

   Position calculation in latitude, longitude and height is given between equation 2.57 and 2.59 for this block.

Model representation is given in Figure 6.4.

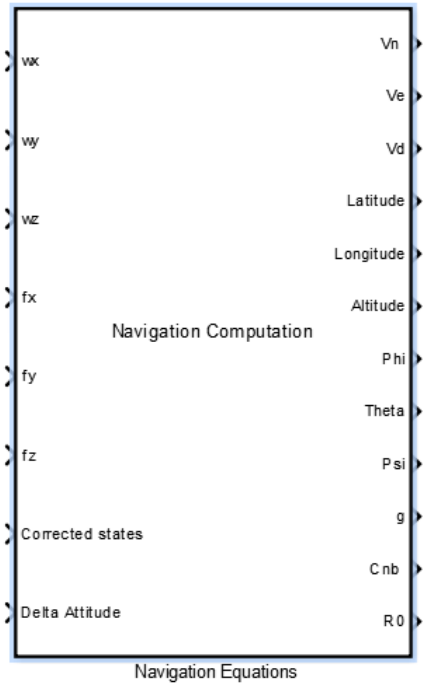



Figure 6.4: Navigation mechanization algorithm simulink subsystem block

**Kalman Filter Algorithm:**

Kalman filter is used to estimate error values of observed error states. In Kalman filter algorithm there are two important steps, prediction and update. In prediction linearized matrix model of non-linear navigation equations is used to predict error

values in time. When observed error state vector is come to Kalman filter, update procedure estimates the error values based on observation. The detailed explanation of Kalman procedure was given in chapter 4. After estimated error states are found,

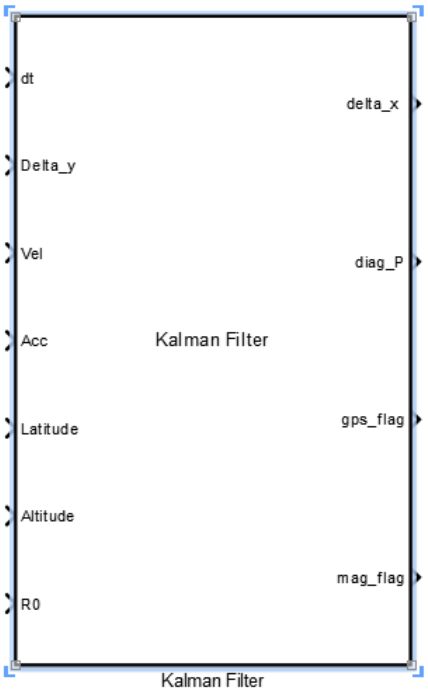



Figure 6.5: Kalman filter algorithm simulink subsystem block

heading, NED velocity and position values are corrected with this error vector. Kalman filter simulink subsytem model is represented in Figure 6.5.

Total simulink model figure with all subsystems (with sensor models and Kalman filter) is given in Figure C.2 in appendix C.

Now as indicated before two helicopter simulator test have been realized to verify the navigation and the Kalman filter algorithm. During these helicopter simulation test TAI designed utility type helicopter mathematical model is used. In simulator environment two different paths were obtained.These two test results are given under two parts,

- First one of these paths is called as *Pirouette Manoeuvre* according to ADS-33E-PRF helicopter handling qualities specifications. This path is obtained by rotating helicopter around a circle path keeping the nose of the helicopter at the center of the circle. In Figure 6.6 the pirouette manoeuvre can be seen. After

that simulator test results and simulink navigation model test results have been compared.

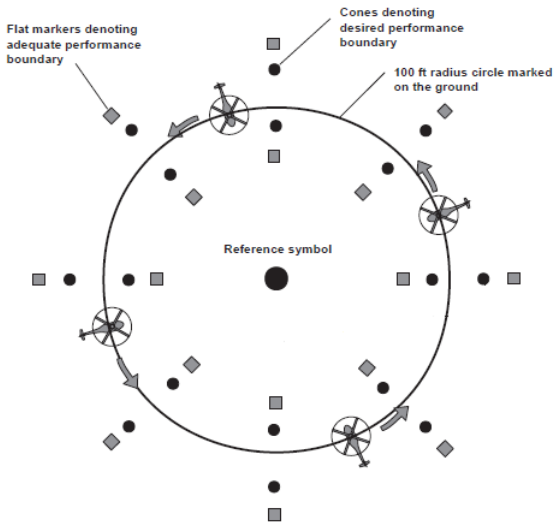



Figure 6.6: Course for pirouette manoeuvre

- As in first test the similar procedure has been followed. However different from the first test free flight has been chosen for test. At the end of test when it is compared to former a more complex trajectory was obtained. The simulator and simulink test results have been compared.

### 6.2.1 Simulator Circular Flight Path Test Results

In following figures, the circular path results are given. These figures are covered the srandalone INS solution and EKF solutions on circular trajectory. for simulation purposes the accelerometer and gyroscope true values are sum up with error models results coming from accelerometer and gyroscope error models. Therefore the raw sensor readings are obtained. As for the results figures 6.7 and 6.8 show the accelerations and body angular rates in body frame. Figures between 6.9 and 6.13 show the standalone INS solutions. Figures between 6.14 and 6.18 show the INS solution with EKF. These figures include velocity, position, attitude and latitude-longitude and three dimensional trajectory observations respectively.

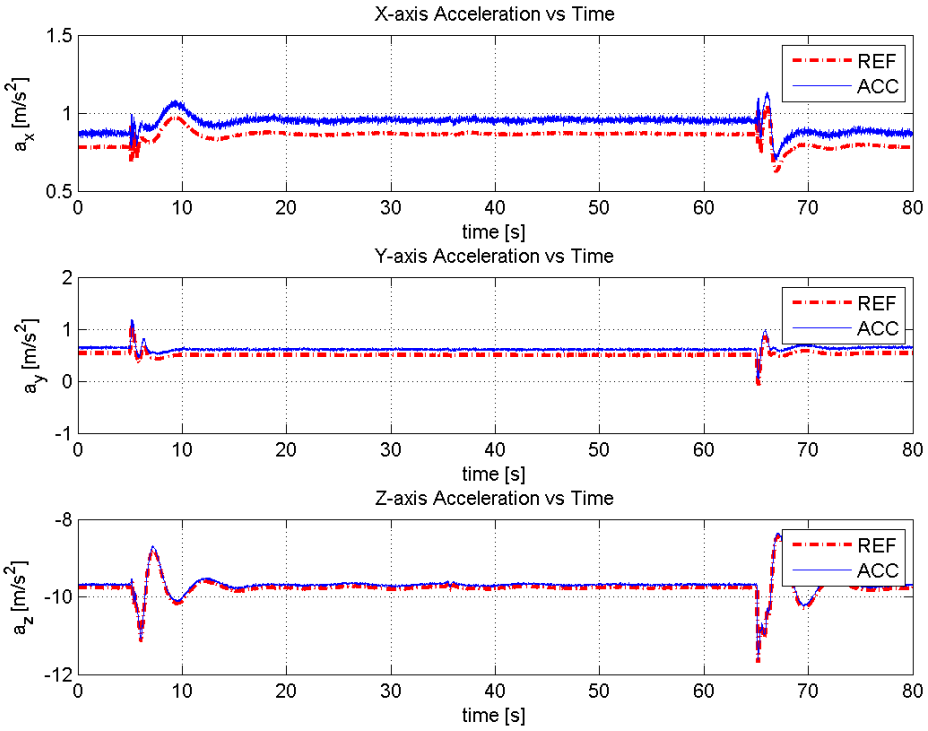



Figure 6.7: Three axis acceleration results for circle trajectory

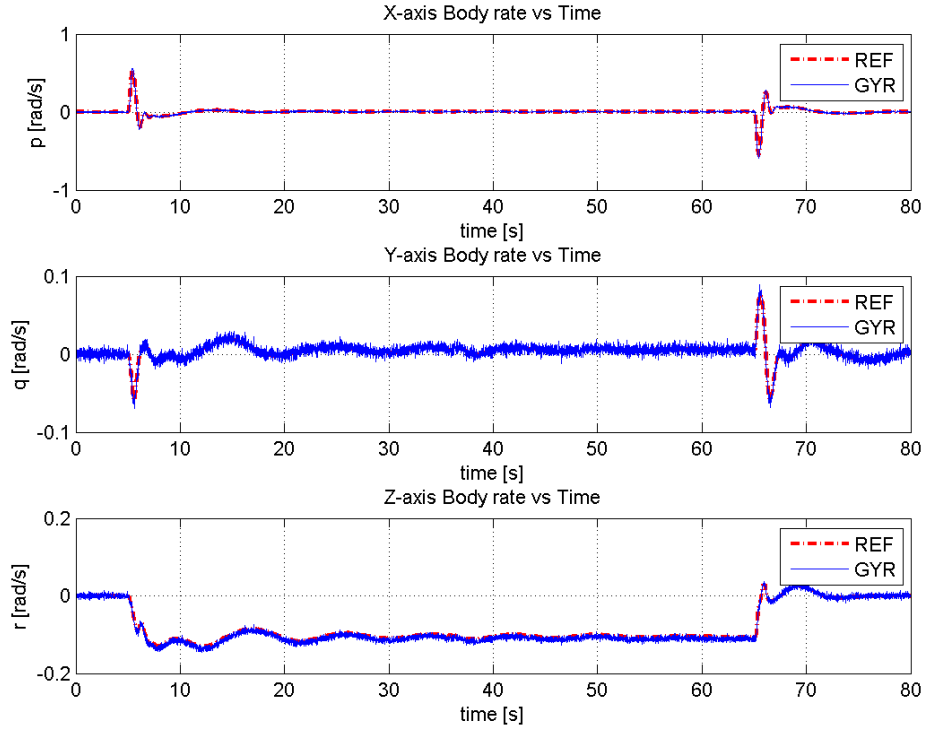



Figure 6.8: Three axis body angular rate results for circle trajectory

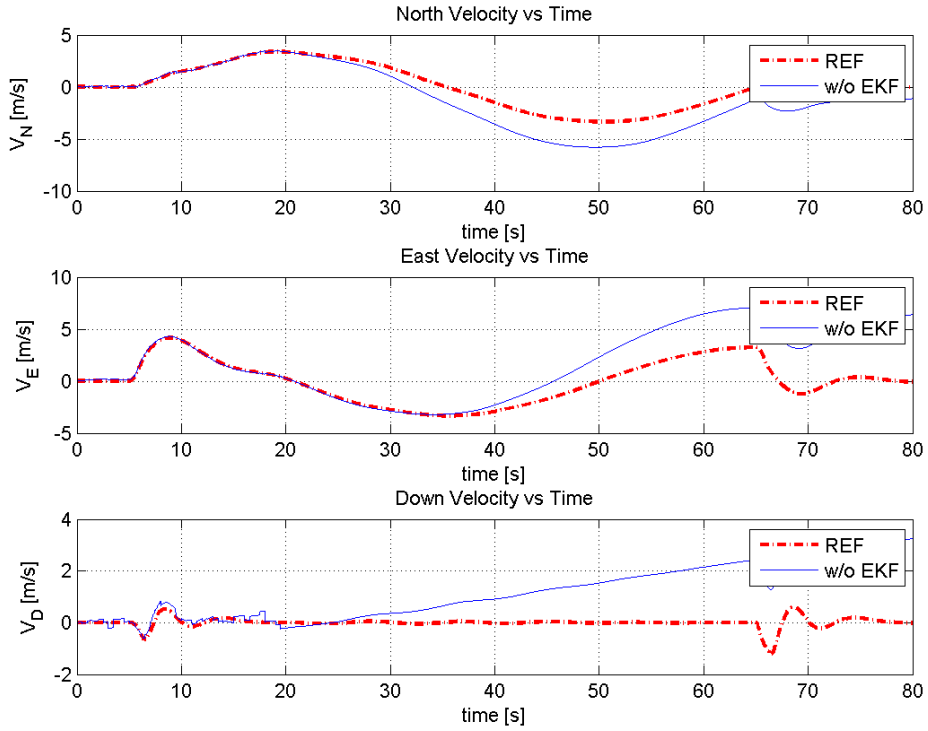



Figure 6.9: Standalone INS velocity results for circle trajectory

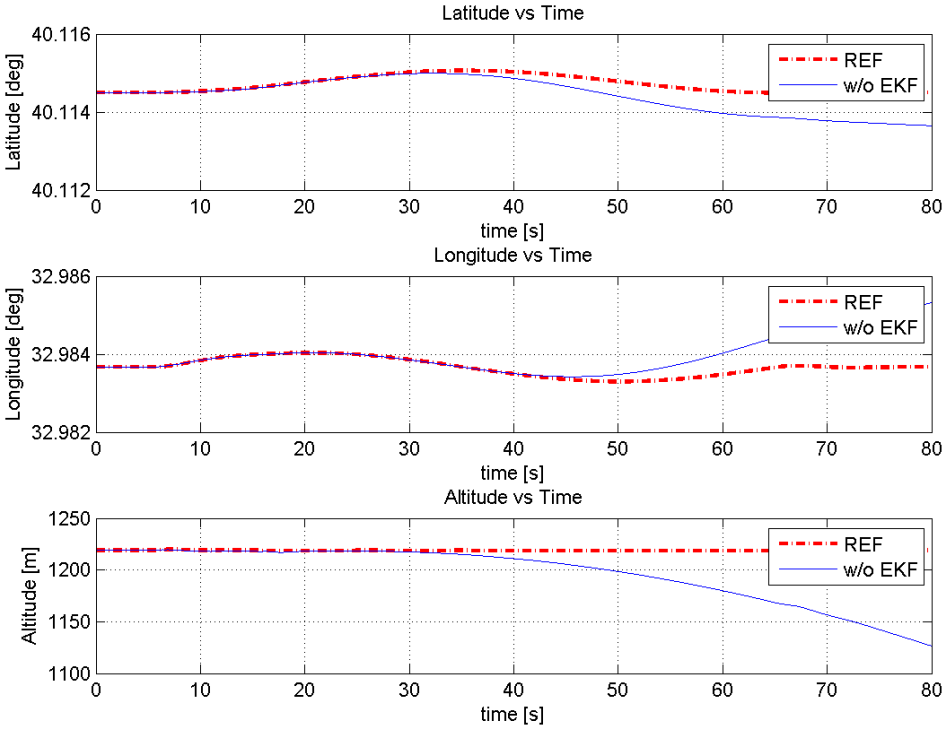



Figure 6.10: Standalone INS position results for circle trajectory

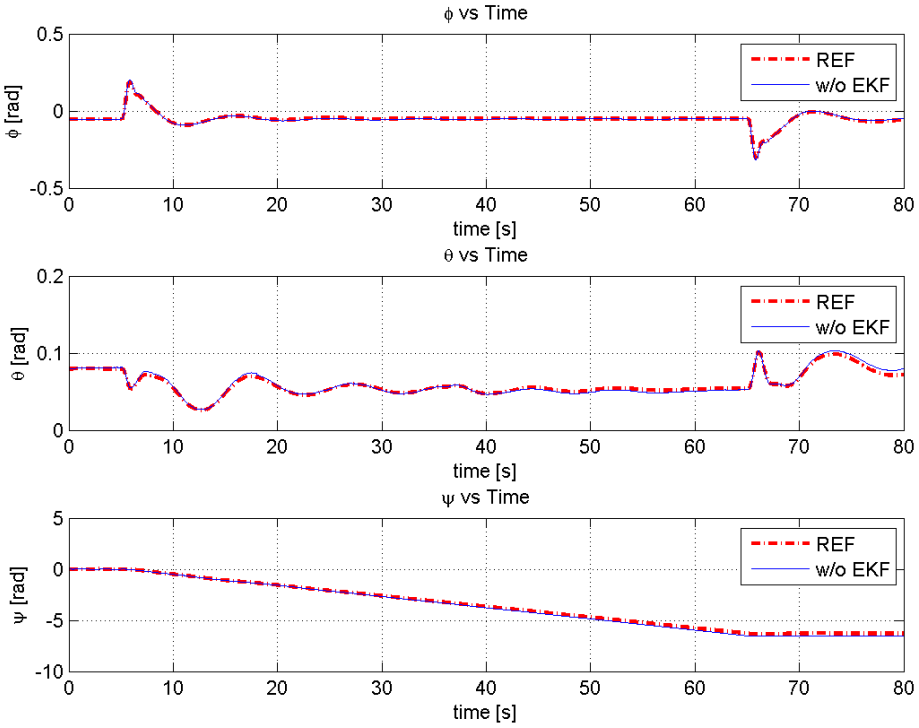



Figure 6.11: Standalone INS attitude results for circle trajectory

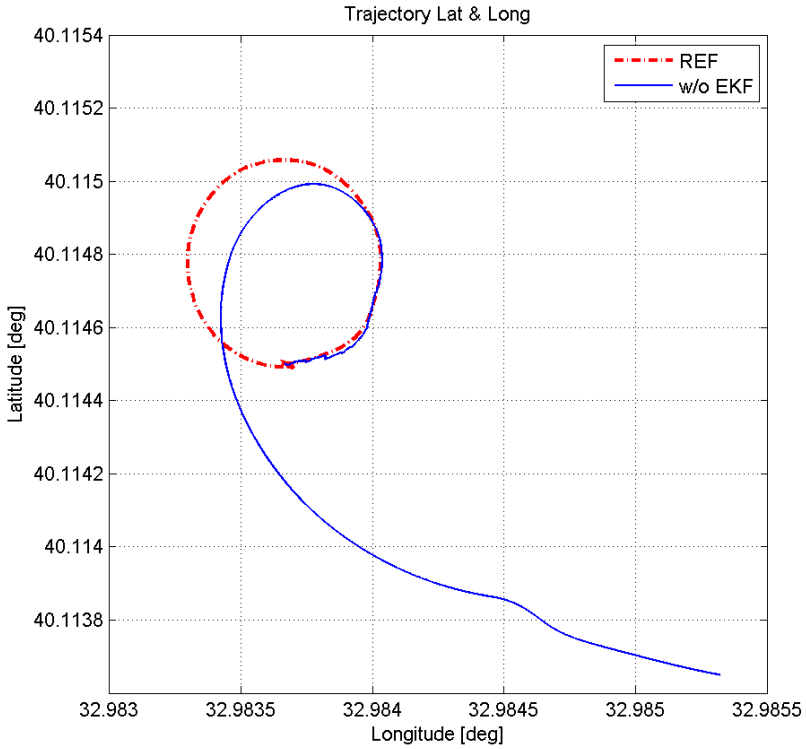



Figure 6.12: Standalone INS latitude and longitude for circle trajectory

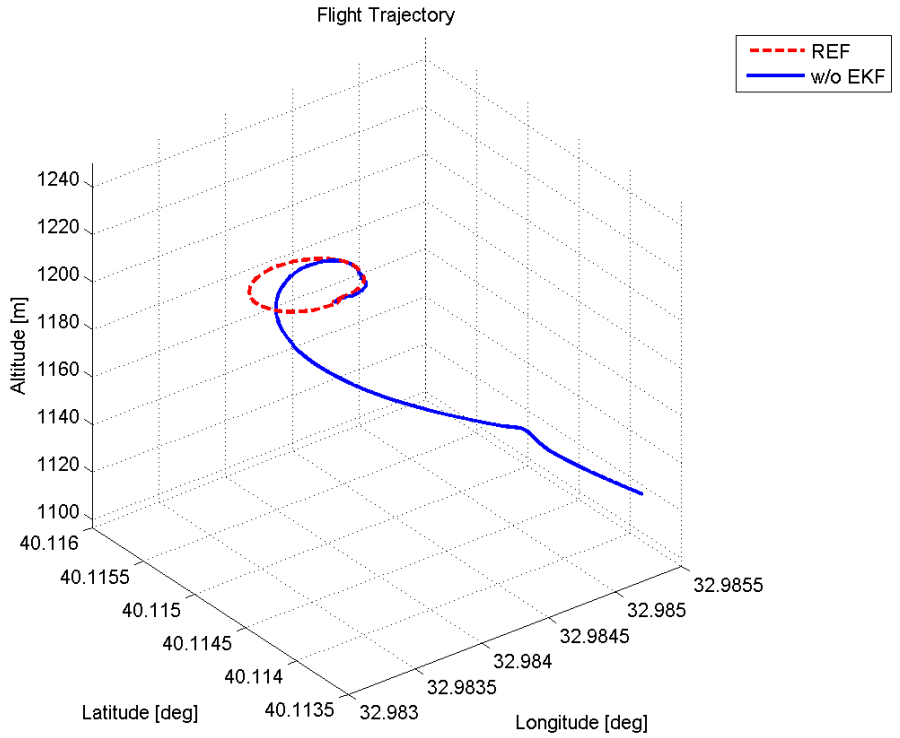



Figure 6.13: Standalone INS 3D flight for circle trajectory

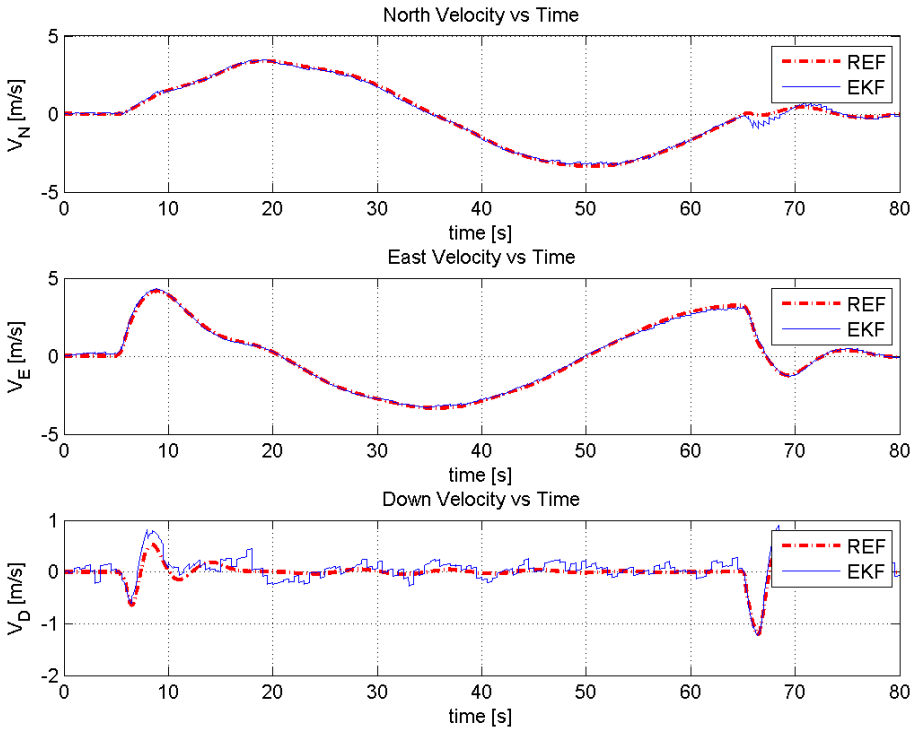



Figure 6.14: EKF velocity results for circle trajectory

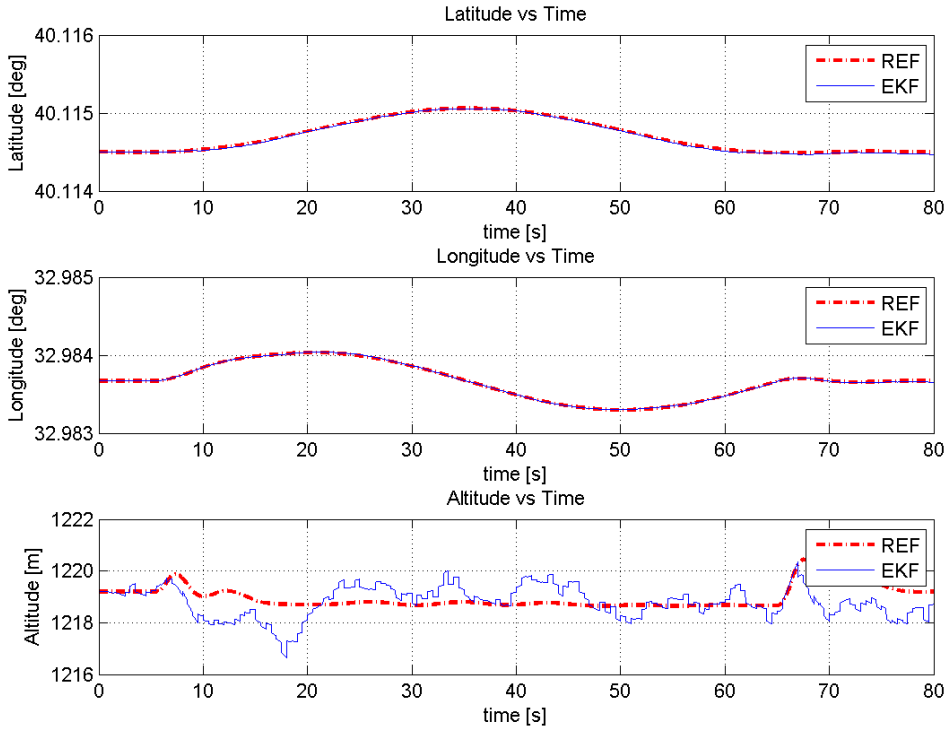



Figure 6.15: EKF position results for circle trajectory

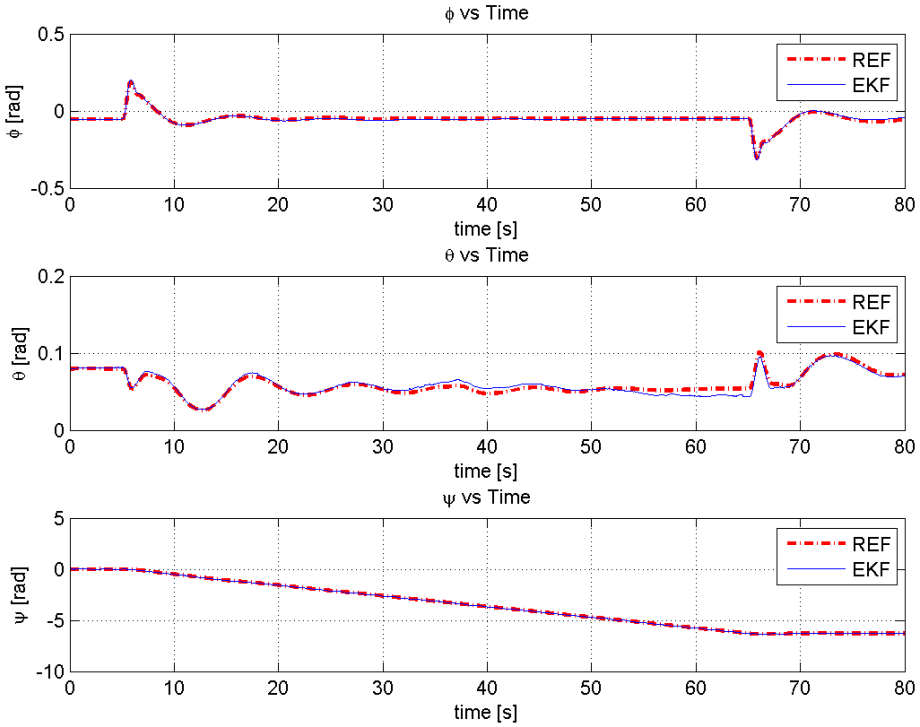



Figure 6.16: EKF attitude results for circle trajectory

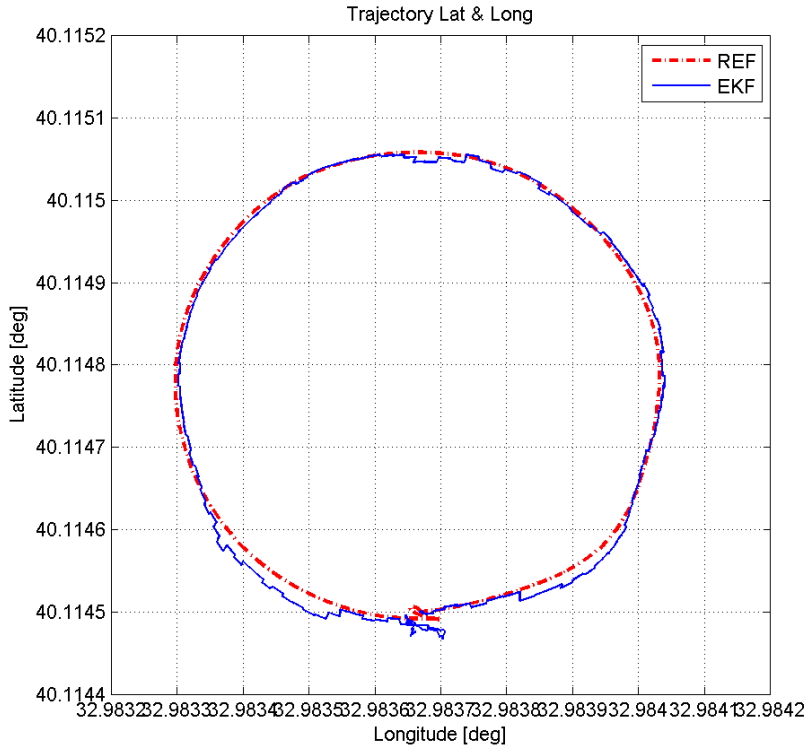



Figure 6.17: EKF latitude and longitude for circle trajectory

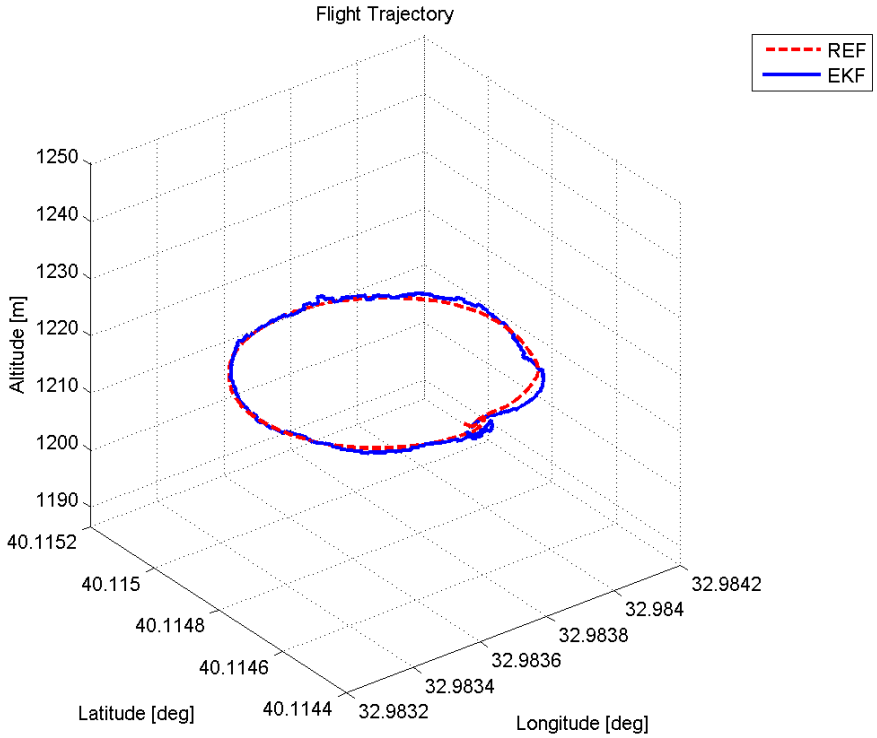



Figure 6.18: EKF 3D flight for circle trajectory

As it can be seen from the figures between 6.9 and 6.13, without EKF the INS solutions diverge because of the accumulation of the errors. Especially in altitude the divergence characteristic can be seen clearly. And in Figure 6.12 latitude and longitude are moving away from the reference solution. Therefore integration of Kalman filter is necessary with GPS velocity and position information. When EKF solutions are examined,it can be seen that position values are close to reference. The figures between 6.14 and 6.18 shows EKF estimation performance.

Moreover the estimation error values of EKF for velocities, positions and attitudes are given between figure 6.19 and 6.21.

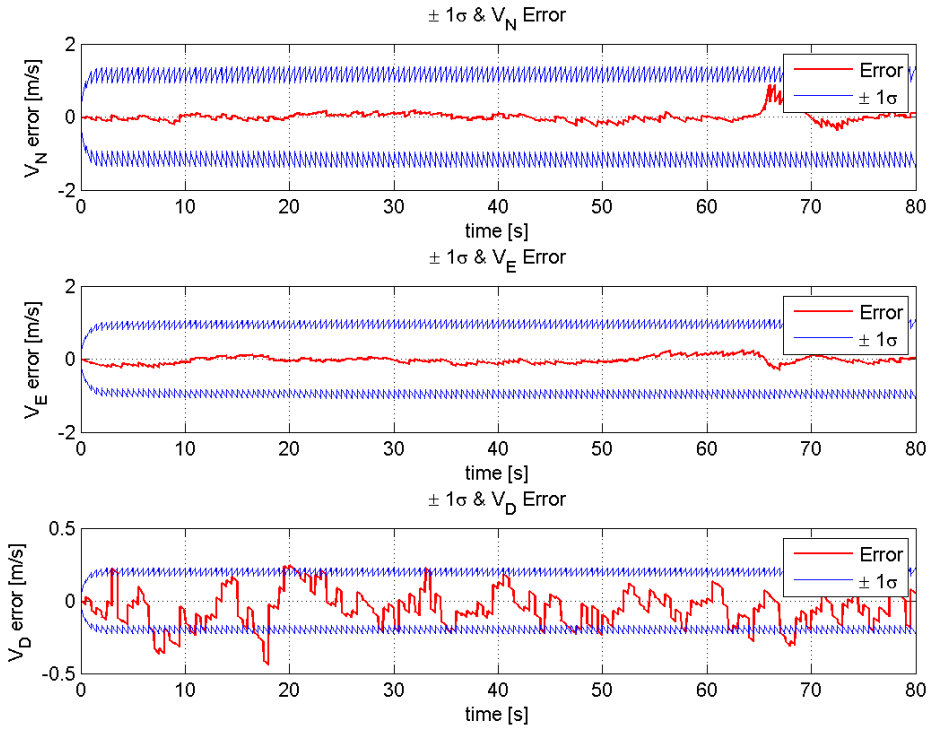



Figure 6.19: EKF velocity estimation error between $\pm 1\sigma$ bound for circle trajectory

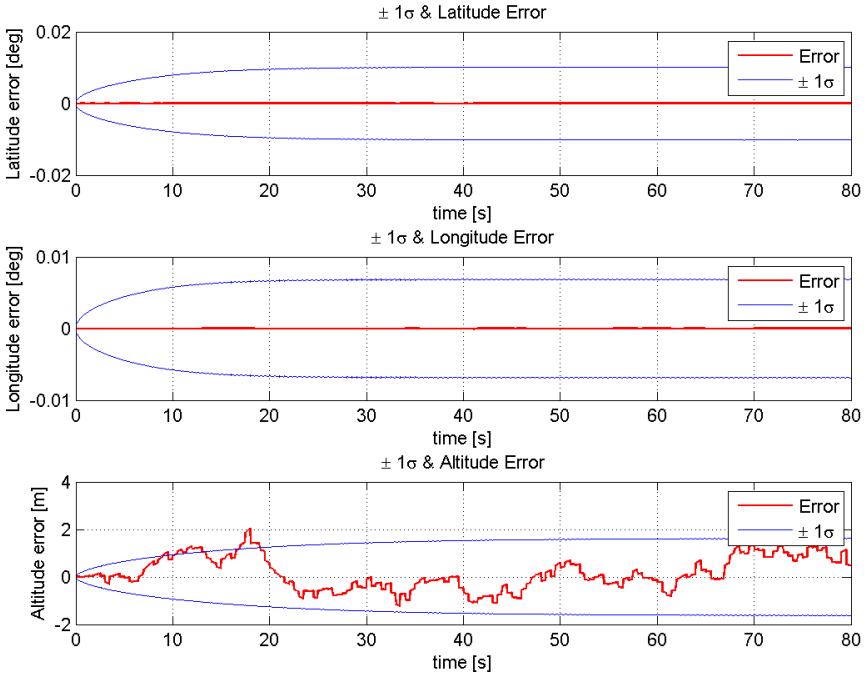



Figure 6.20: EKF position estimation error between $\pm 1\sigma$ bound for circle trajectory

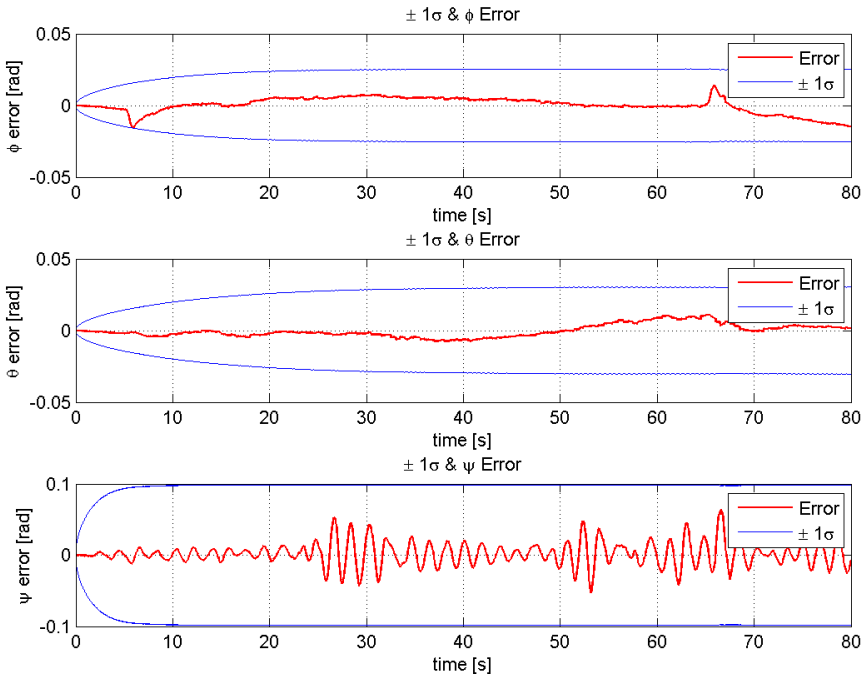



Figure 6.21: EKF attitude estimation error between $\pm 1\sigma$ bound for circle trajectory

Table 6.3: INS/GPS model RMSE results for circular trajectory

| Positions | Latitude [deg] | Longitude [deg] | Altitude [m] |
|---|---|---|---|
| RMSE values | 1.2493e-05 | 6.5388e-06 | 0.6992 |

In Table 6.3, RMSE result between INS/GPS simulink model and reference data from TAI helicopter simulator is given for latitude, longitude and height. As it can be shown in table, the results are almost same. Therefore it can be said that Kalman filter correctly estimates the trajectory.

### 6.2.2 Simulator Complex Flight Path Test Results

In this analysis the same navigation structure is tested with different and more complex trajectory data than previous section. Figures 6.22 and 6.23 show the accelerations and body angular rates in body frame. Figures between 6.24 and 6.28 show the standalone INS solutions. Figures between 6.29 and 6.33 show the INS solution with EKF. These figures include velocity, position, attitude and latitude-longitude and three dimensional trajectory observations respectively.

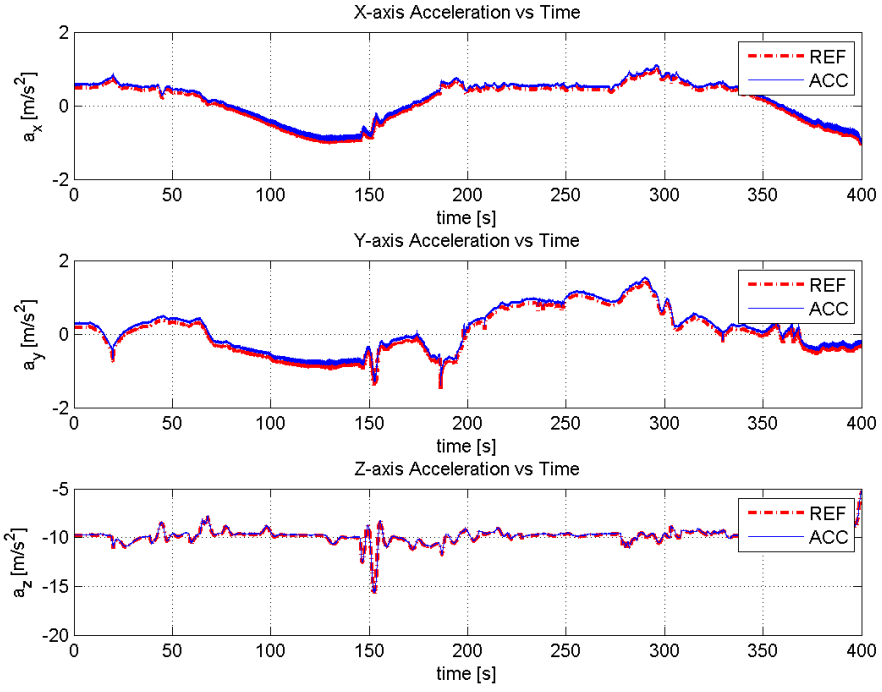



Figure 6.22: Three axis acceleration results for complex trajectory

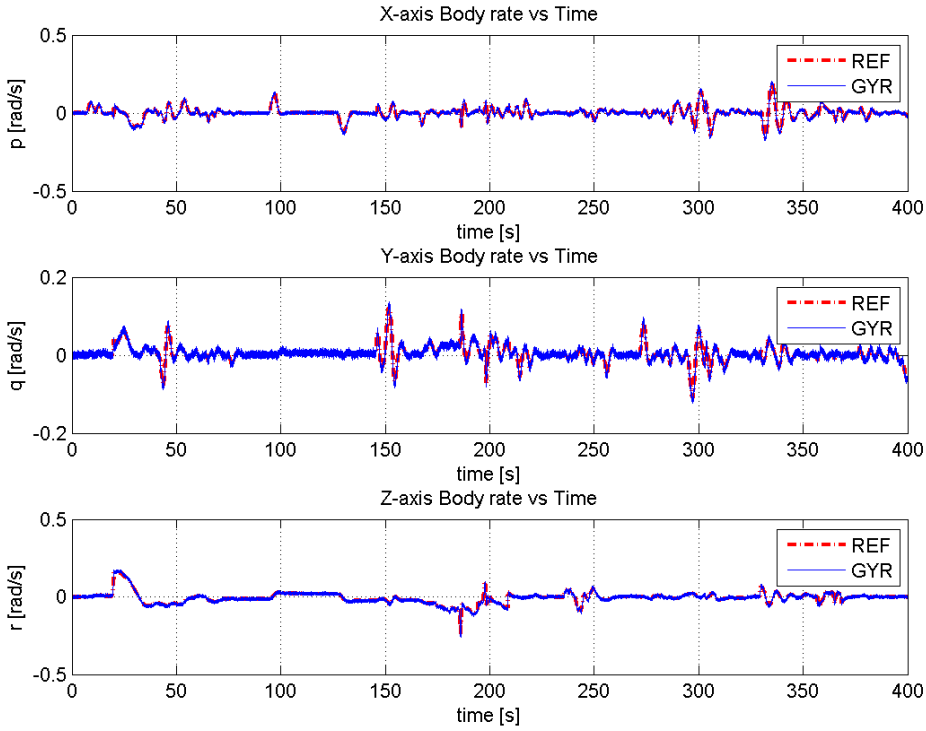



Figure 6.23: Three axis body angular rate results for complex trajectory

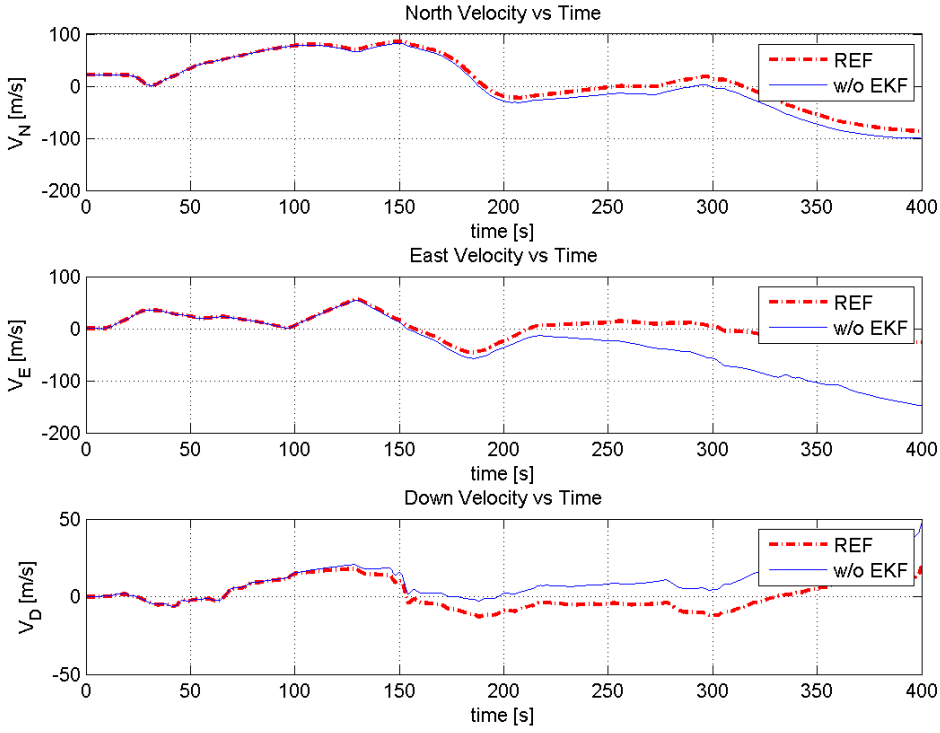



Figure 6.24: Standalone INS velocity results for complex trajectory

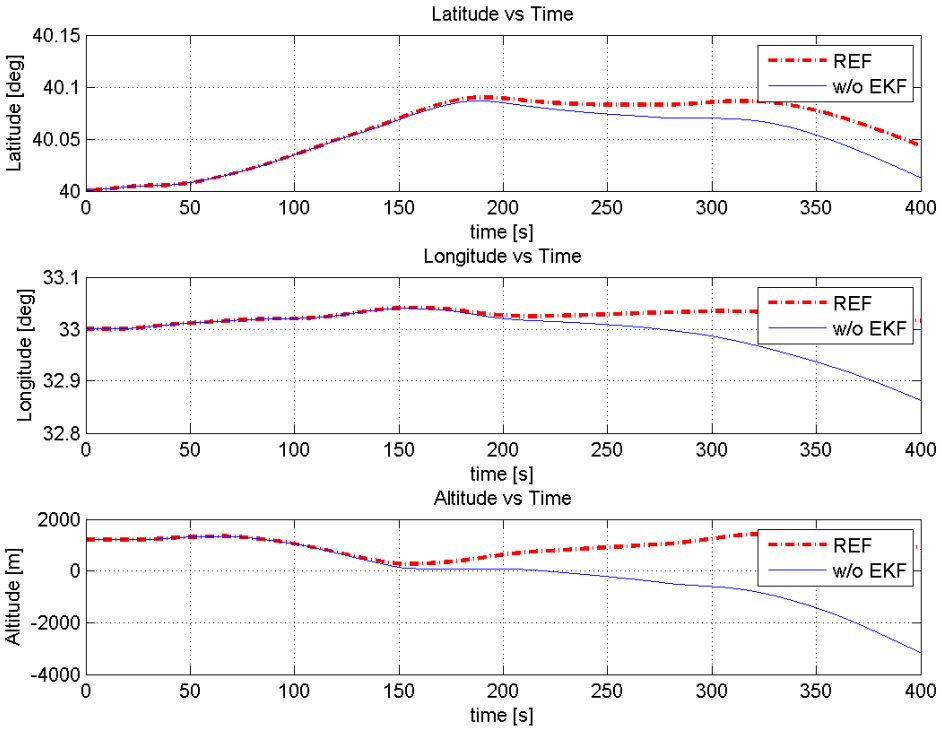



Figure 6.25: Standalone INS position results for complex trajectory

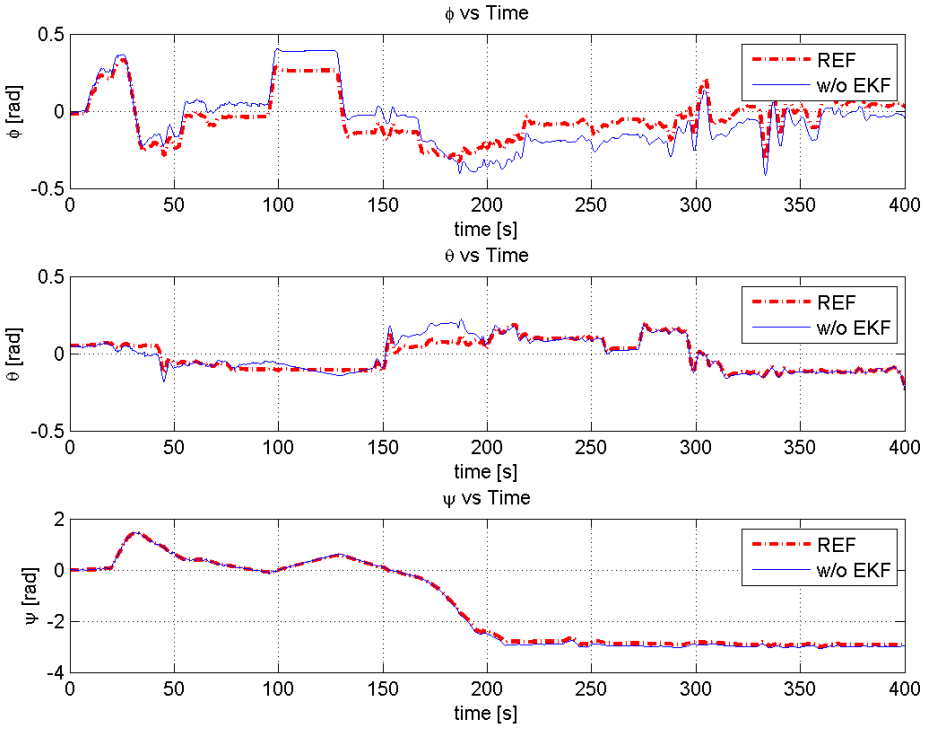



Figure 6.26: Standalone INS attitude results for complex trajectory

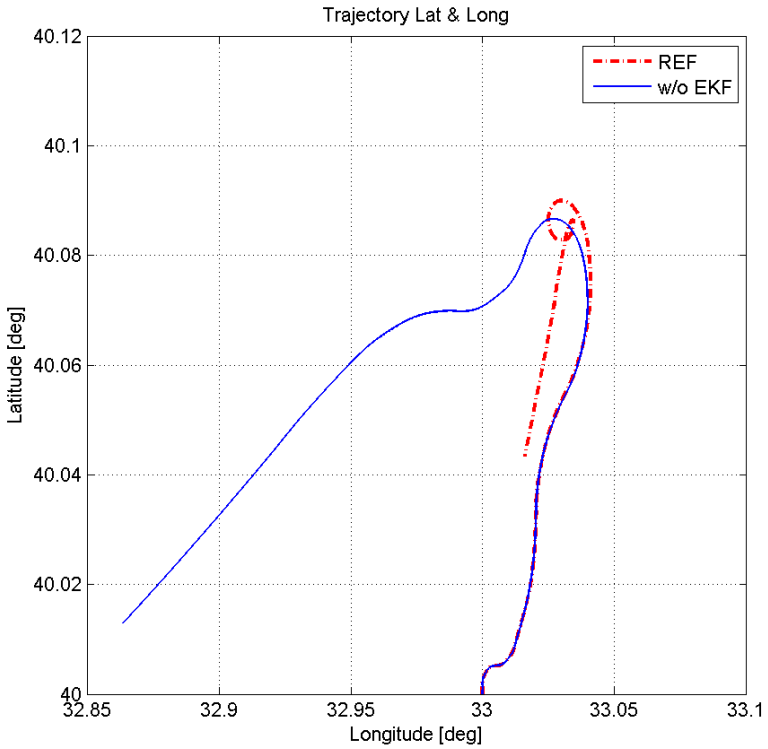



Figure 6.27: Standalone INS latitude and longitude for complex trajectory

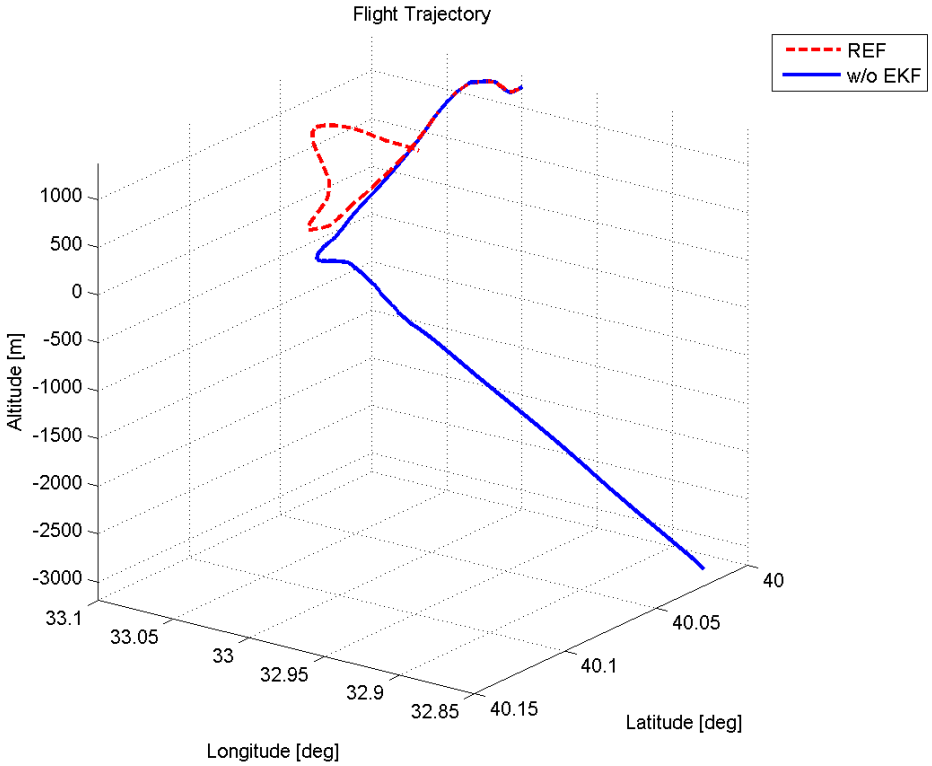



Figure 6.28: Standalone INS 3D flight for complex trajectory

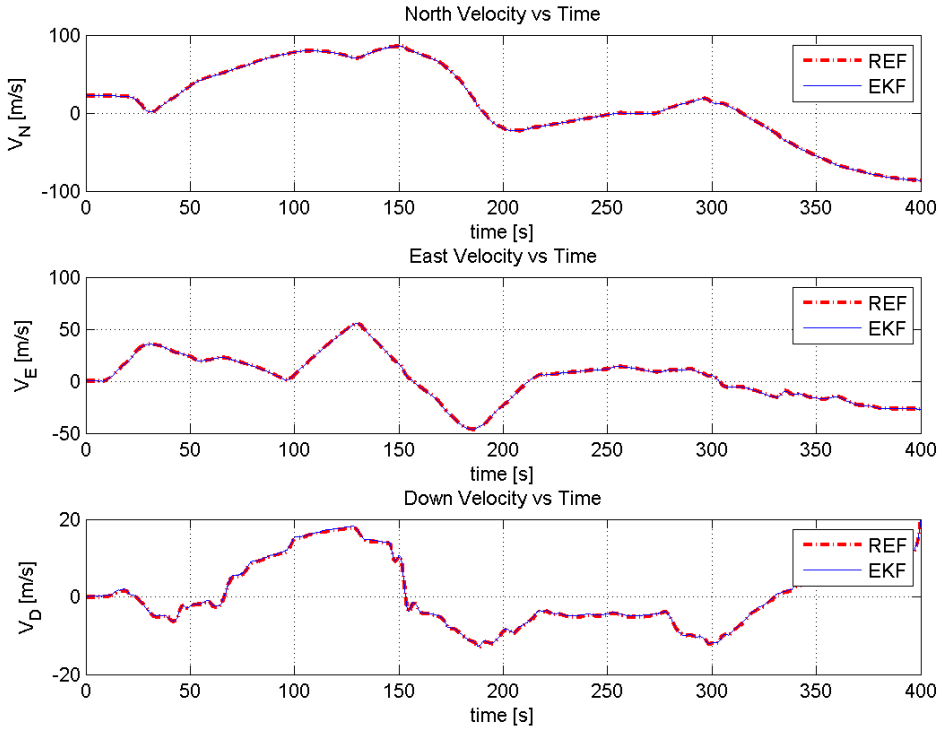



Figure 6.29: EKF velocity results for complex trajectory

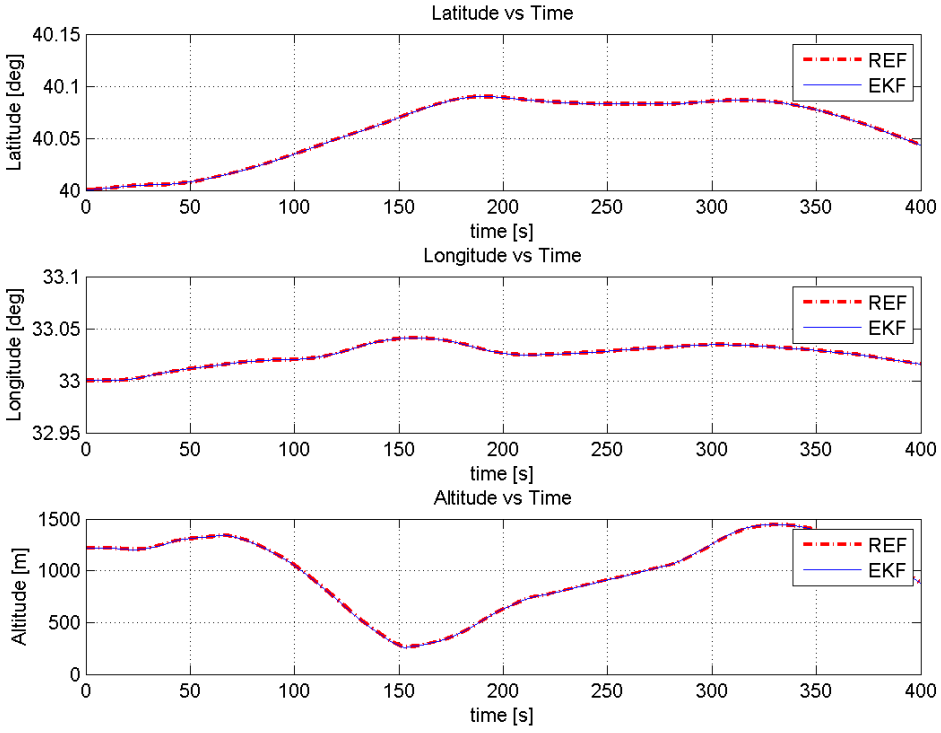



Figure 6.30: EKF position results for complex trajectory

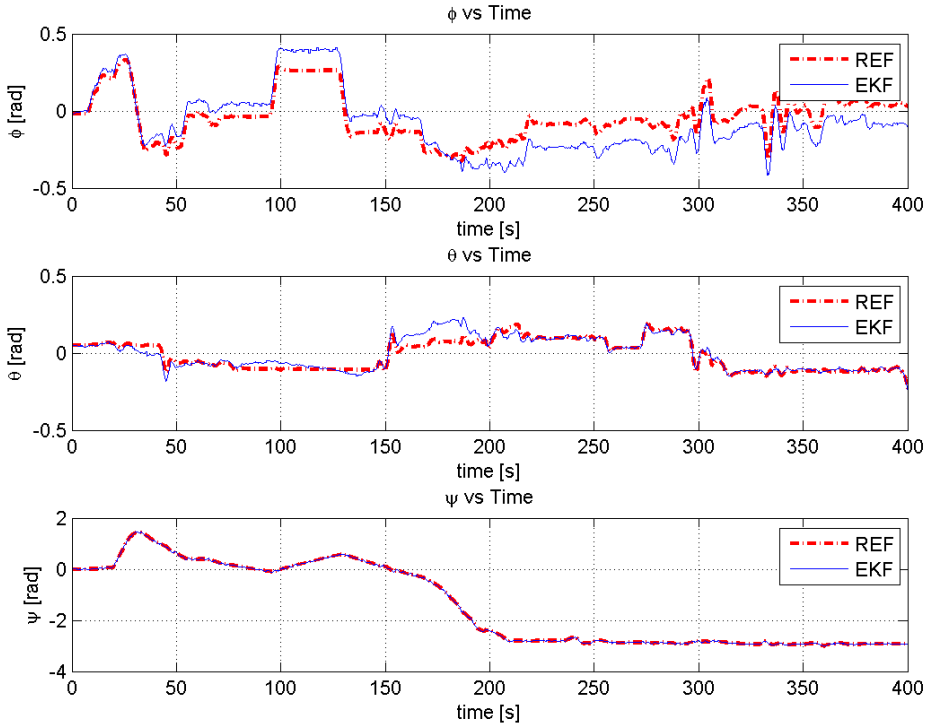



Figure 6.31: EKF attitude results for complex trajectory

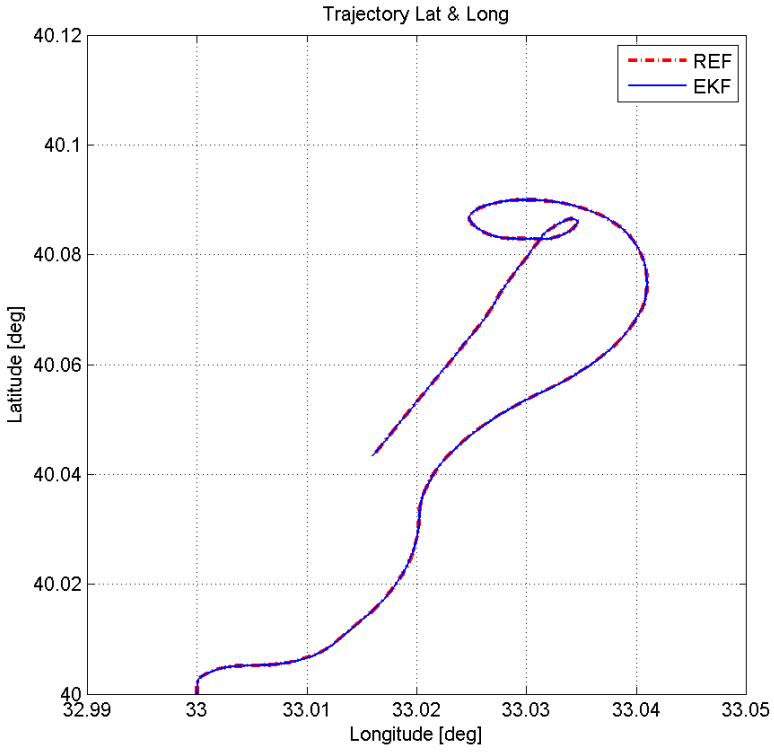



Figure 6.32: EKF latitude and longitude for complex trajectory

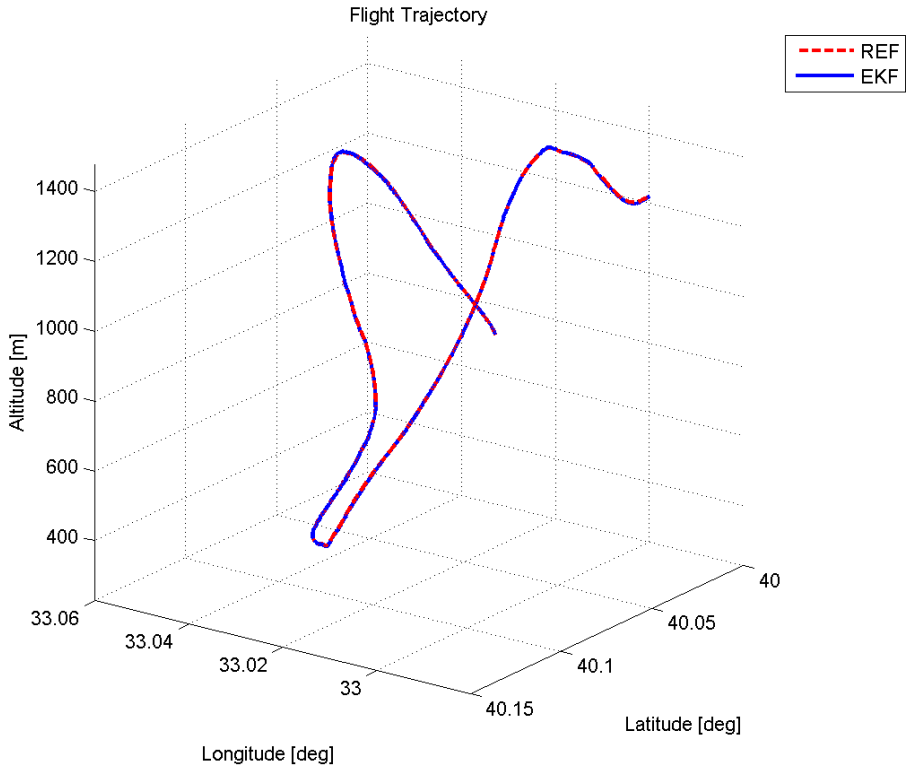



Figure 6.33: EKF 3D flight for complex trajectory

Again EKF results are clearly better than standalone INS solution. When the Figure 6.28 and 6.33 are compared, EKF performance can be seen easily. Especially the improvement in north, east, down velocities and latitude, longitude and height values are sustained with EKF integration. Without EKF accumulation of inserted error values causes low performance in navigation solutions. According to these results integration of INS and GPS with EKF is essential to obtain better navigation solutions. To get more insight about the error estimation, the estimation results are shared between figures 6.34 and 6.36.

Table 6.4: INS/GPS model RMSE results for complex trajectory

| Positions | Latitude [deg] | Longitude [deg] | Altitude [m] |
|---|---|---|---|
| RMSE values | 2.3060e-05 | 2.7733e-05 | 1.0951 |

In Table 6.4, RMSE results for complex trajectory is given. As in circular trajectory case, the INS/GPS models gives similar results with TAI helicopter simulator data.

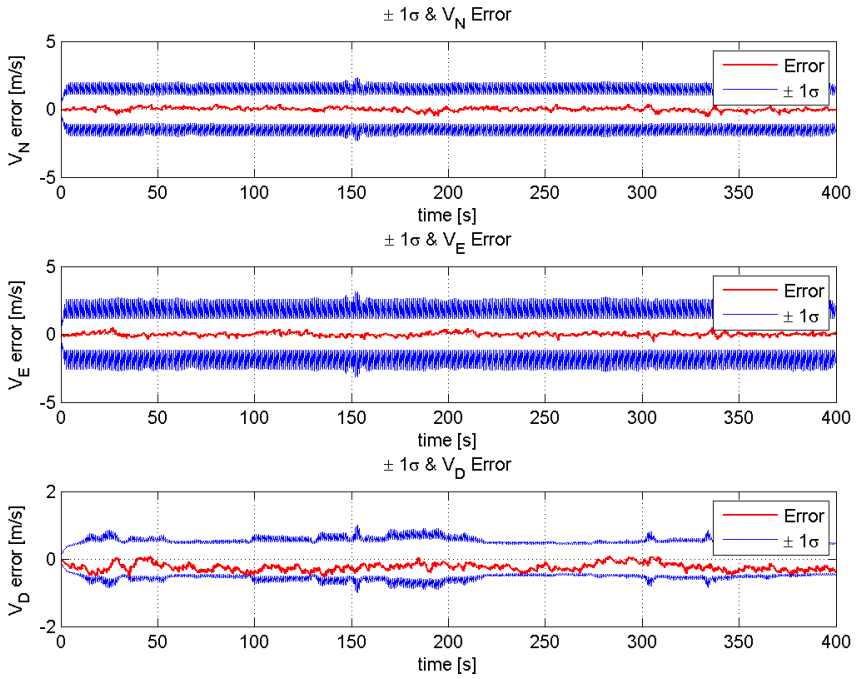



Figure 6.34: EKF velocity estimation error between $\pm 1\sigma$ bound for complex trajectory

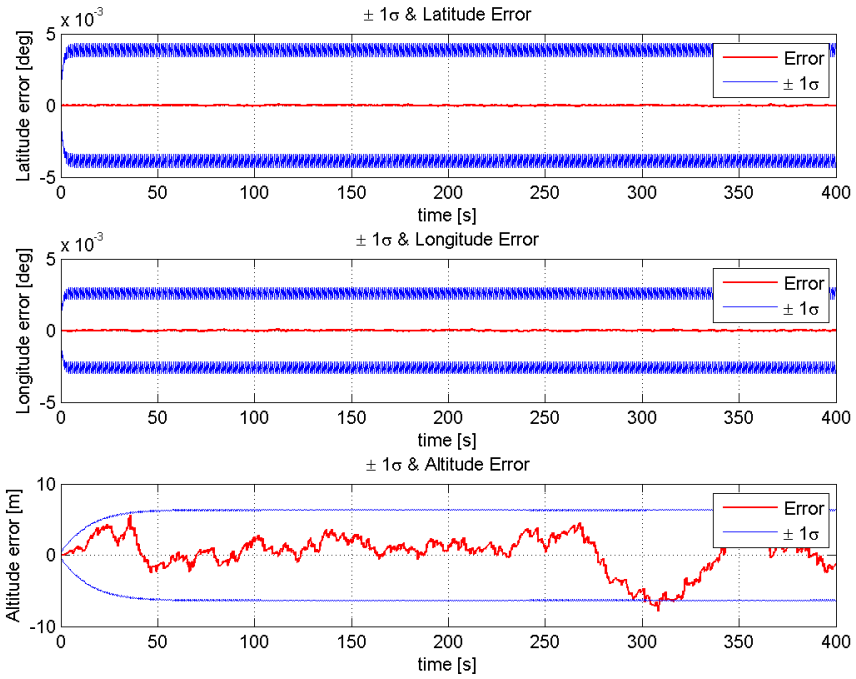



Figure 6.35: EKF position estimation error between $\pm 1\sigma$ bound for complex trajectory

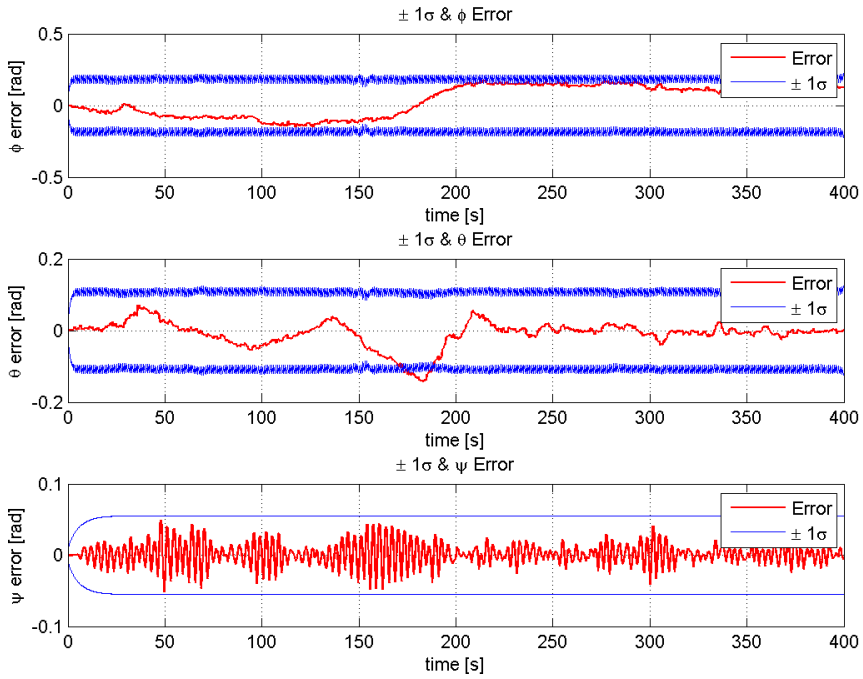



Figure 6.36: EKF attitude estimation error between $\pm 1\sigma$ bound for complex trajectory

## 6.3 Simulink Model Test with Experimental Land Vehicle Data and Quadrocopter Flight Data

In this section, the simulink navigation model will be tested with experimental field data. All navigation and Kalman filter structures are used as in previous chapter. However in this time the sensor values come from the sensor modules on land vehicle and quadcopter. In following sections, firstly hardware structure and the quadcopter model will be presented. After that Multilayer perceptron neural network (MLPNN) model and its usage in navigation model will be given. Finally the MLPNN performance results will be shown by comparing without MLPNN cases for both land test and quadcopter test respectively.

### 6.3.1 Hardware Structure

In simulator tests the INS model has been tested with sensor models. To simulate the INS system with real sensor data, quadcopter platform has been utilized. The

quadcopter platform can be seen in Figure 6.85. In this figure quadcopter plotform, quadcopter controller and the interface program Mission Planner on PC can be seen.

Figure 6.37: Quadcopter, controller and mission planner program

The hardware on quadcopter can be listed as below. The detailed information about autopilot and GPS are given in appendix B.

1. RCtimer 360KV brushless motor
2. Pixhawk autopilot with custom software
3. Neo M8N gps receiver with 3 axis compass
4. Laserlight2 Laser range finder (40m)
5. Raspberry Pi2 with camera module
6. Wireless ethernet module
7. 3 axis gimbal with GoPro4
8. 2x 4S5500mAh batteries
9. 600mW 5.8Ghz video transmitter
10. 500mW 2.4Ghz Rf Transceiver

The hardware structure on quadcopter platform can be seen in Figure 6.38. Accelerometer, gyroscope and magnetometer are included in Pixhawk autopilot module. Detailed information about sensors can be seen in appendix B.

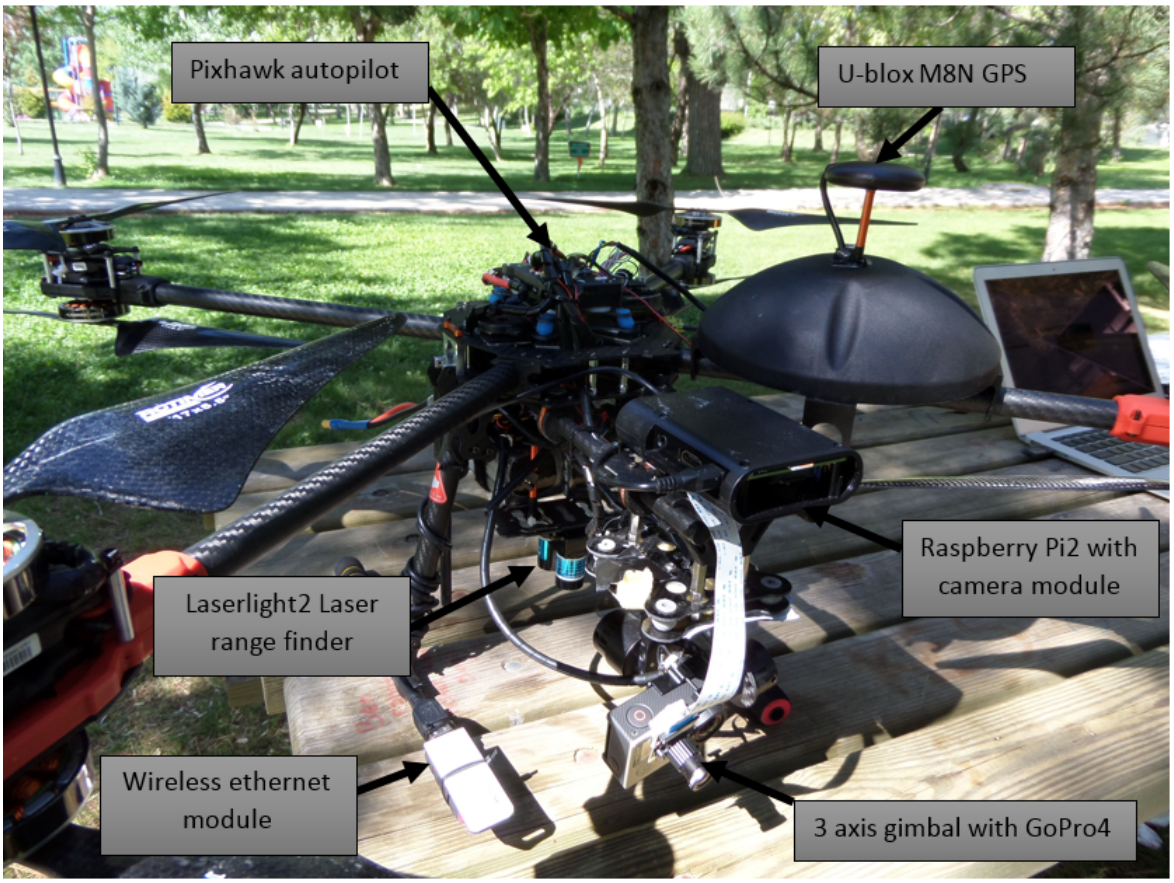


Figure 6.38: Quadcopter hardware stucture

### 6.3.2 Simulink Model Structure with ANN

In circular and complicated trajectory results the navigation algorithm performance with Kalman filter implementation has been tested. In this section in addition to navigation and Kalman algorithm, artificial neural network implementation is also tested with experimental data. The simulink model with multi layer perceptron neural network (MLPNN) can be seen in appendix C in Figure C.3.

As indicated before for neural network structure multi layer perceptron (MLPNN) has been chosen. The MLPNN includes two section. These are learning section and prediction section. In learning section 25 second non-sliding window data is buffered. The data includes $V - \delta V$ and $P - \delta P$ navigation solutions during GPS online. For each $V - \delta V$ and $P - \delta P$ section approximately 30 neurons are used in every hidden layer. As for learning rate, small and large learning rate parameters are avoided. Because small learning rates make learning process slow and large learning

rates make the convergence harder even impossible during learning session. When GPS signal is interrupted prediction section comes online and predicts the error states of Kalman filter for the given velocity and altitude values. The MLPNN simulink model is given in appendix C in Figure C.3. Also training and prediction sections of neural network in time line are shown in Figure 6.39.

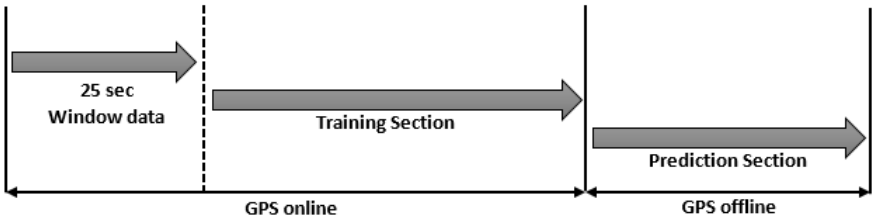



Figure 6.39: MLPNN training and prediction sequences in time line

### 6.3.3 Experimental Land Vehicle Test Results

Experimental land vehicle test is realized to test Pixhawk and GPS receiver performance on road test. To realize this experiment Pixhawk and GPS receiver are fixed to car as seen form Figure 6.40.

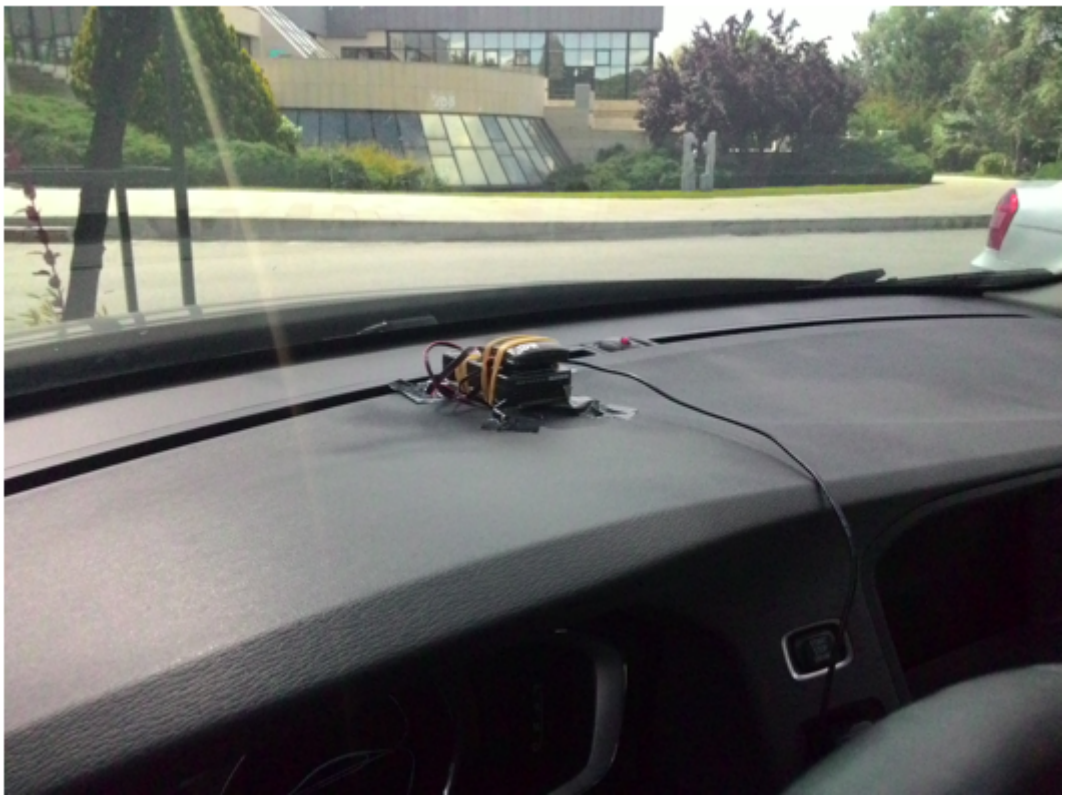


Figure 6.40: Land vehicle test setup

Two important test are covered in road test. Firstly the simulink EKF model is compared with Pixhawk reference data. The results are given in following sections. Secondly, MLPNN performance is tested for GPS signal lost. The road test was fulfilled in Middle East Technical University.

### Experimental Road Test Results with EKF

The experimental data set results are compared with the simulink navigation mechanization and extended Kalman filter algorithm. The figures include acceleration, angular rates, velocity, position and attitude results respectively. Also 2D latitude and longitude trajectory is shown on Google map. Figures 6.41 and 6.42 show the raw acceleration and body rates from accelerometer and gyroscope. Figures between 6.43 and 6.47 show the velocity, position and attitude EKF results. Latitude and longitude trajectory result and 3D trajectory solution are given in Figure 6.48 and 6.50. Also latitude and longitude trajectory on Google map can be shown in Figure 6.49. Estimation error results for velocity, position and attitude estimation are given in figures between 6.51 and 6.53. The RMSE results for road test is given in Table 6.5.

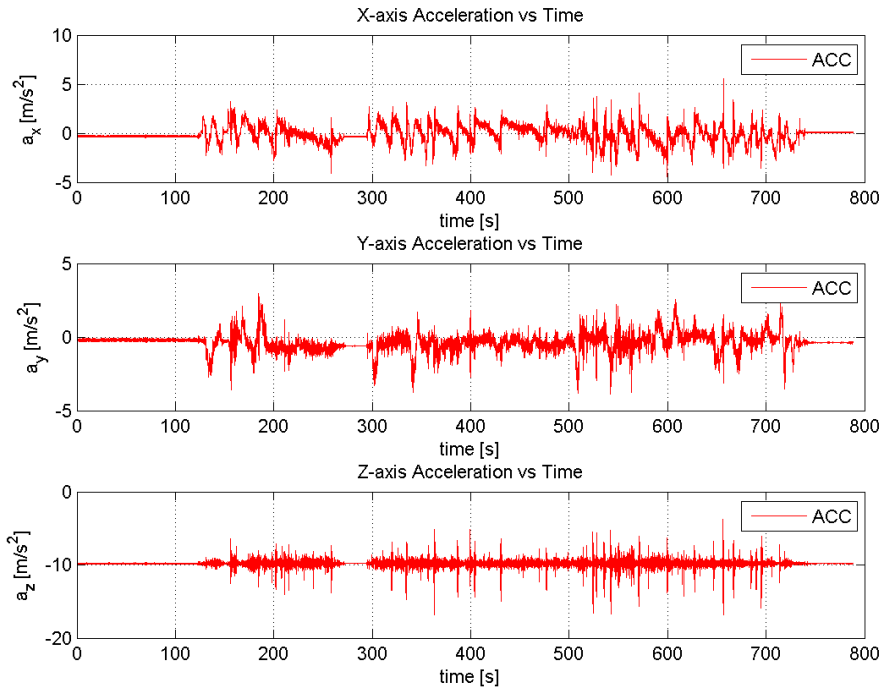



Figure 6.41: Three axis acceleration results for road test data

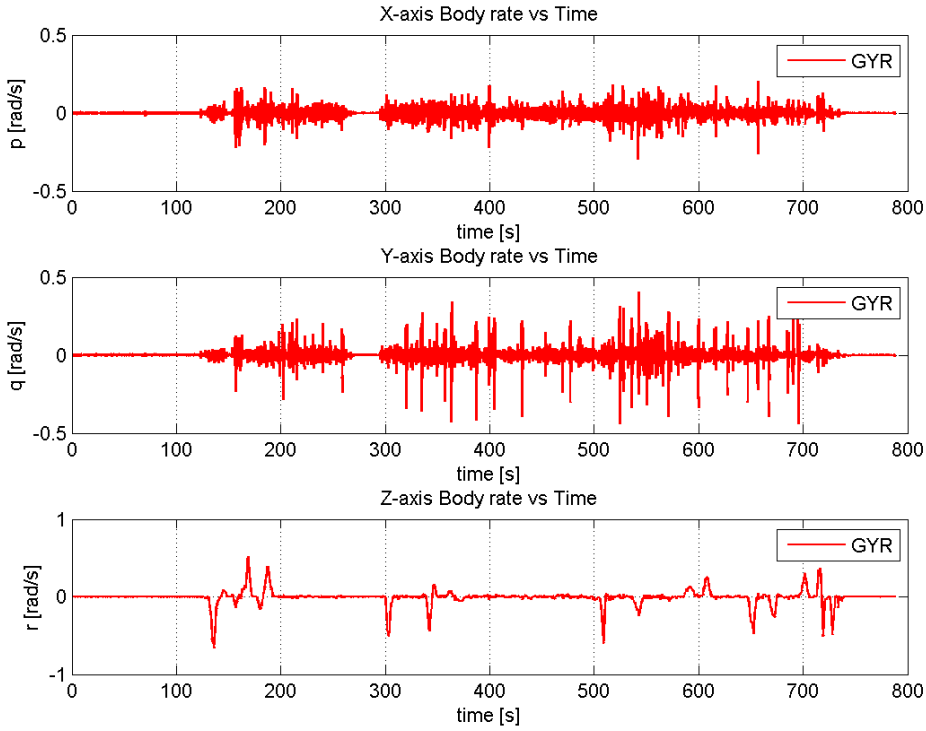



Figure 6.42: Three axis body angular rate results for road test data

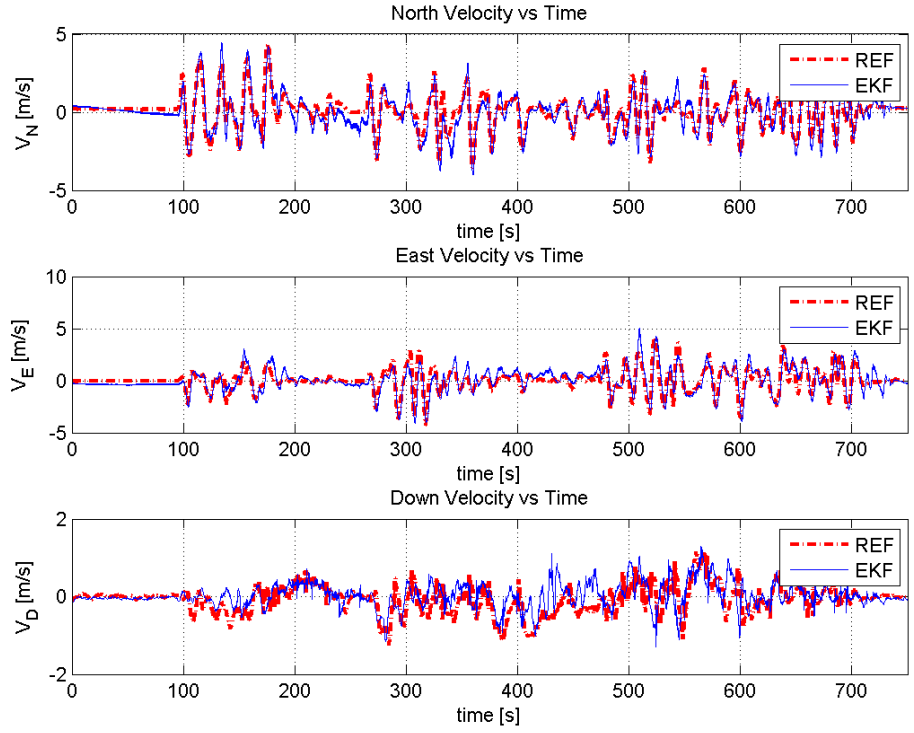



Figure 6.43: EKF velocity results for road test data

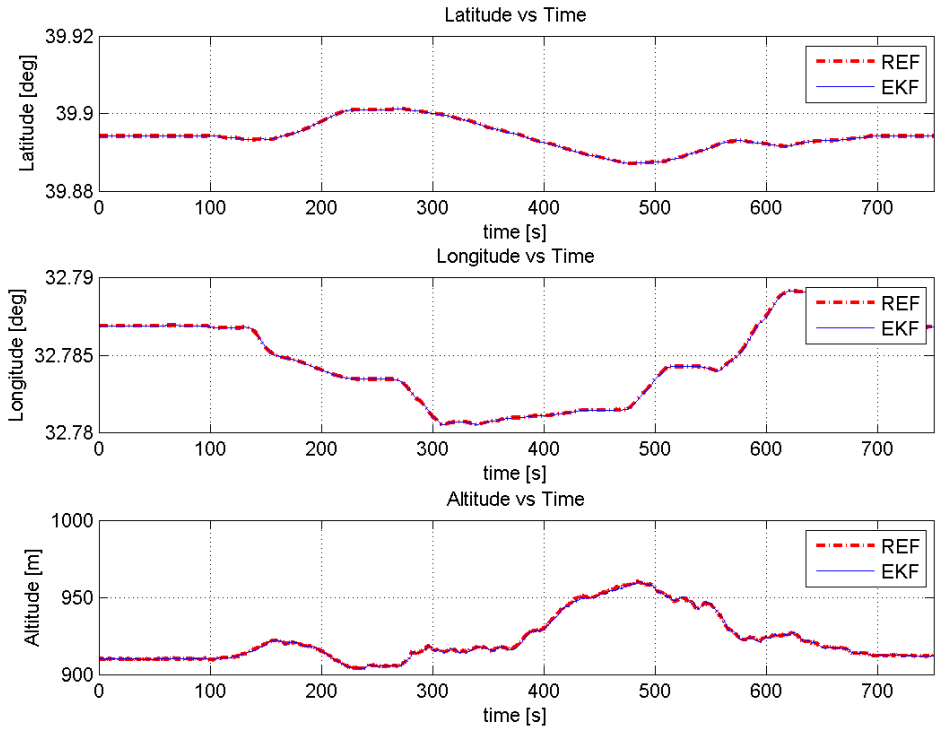



Figure 6.44: EKF position results for road test data

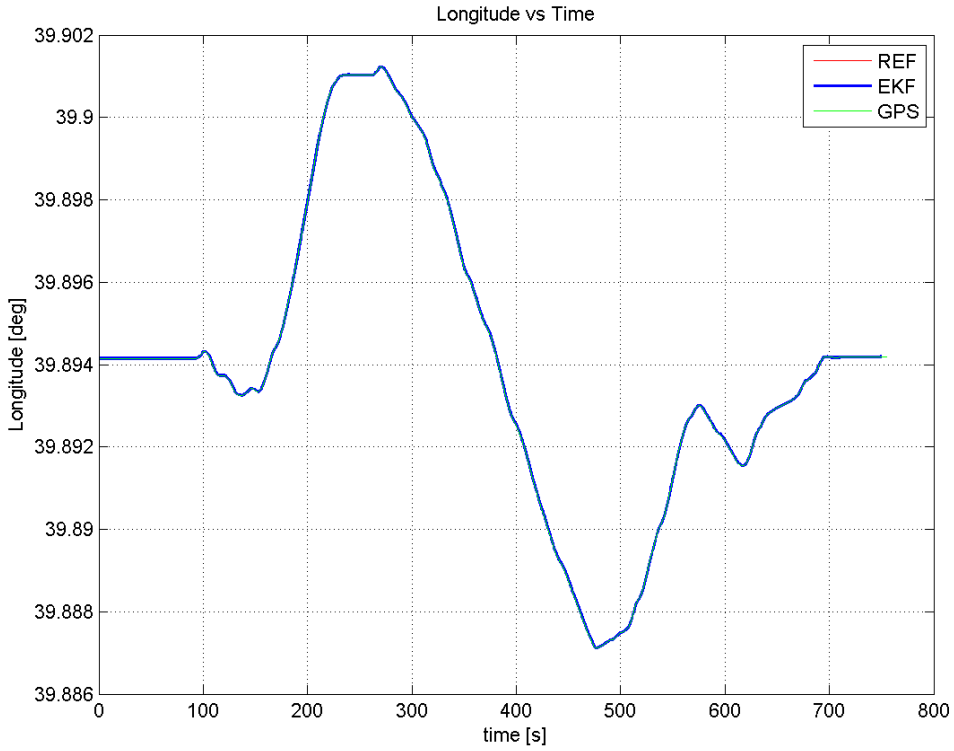



Figure 6.45: Reference, simulink EKF model and GPS latitude data

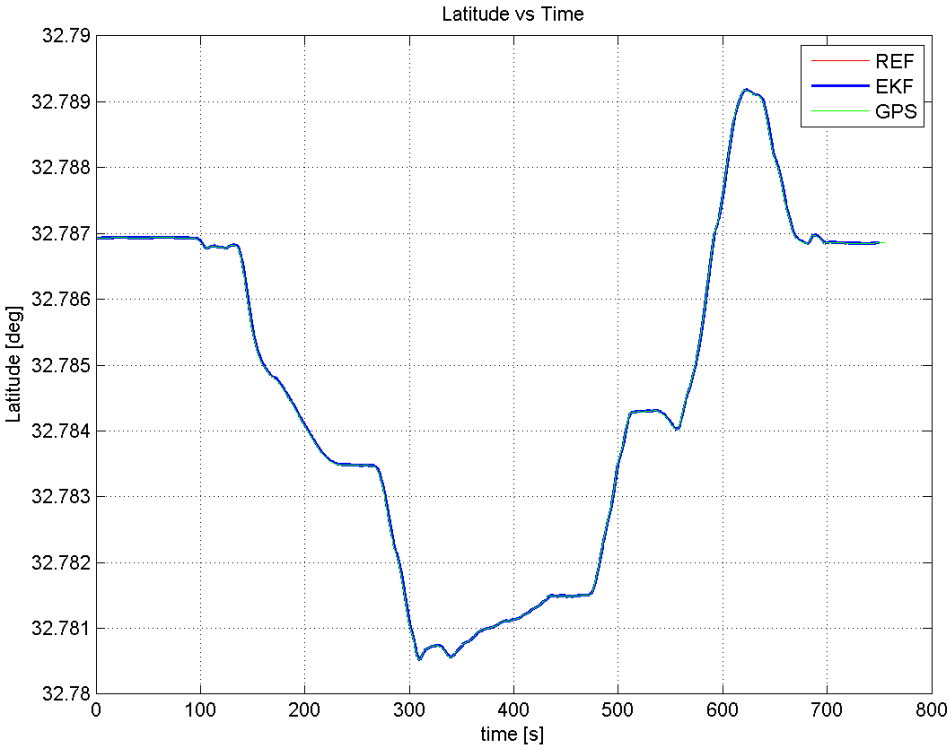



Figure 6.46: Reference, simulink EKF model and GPS longitude data

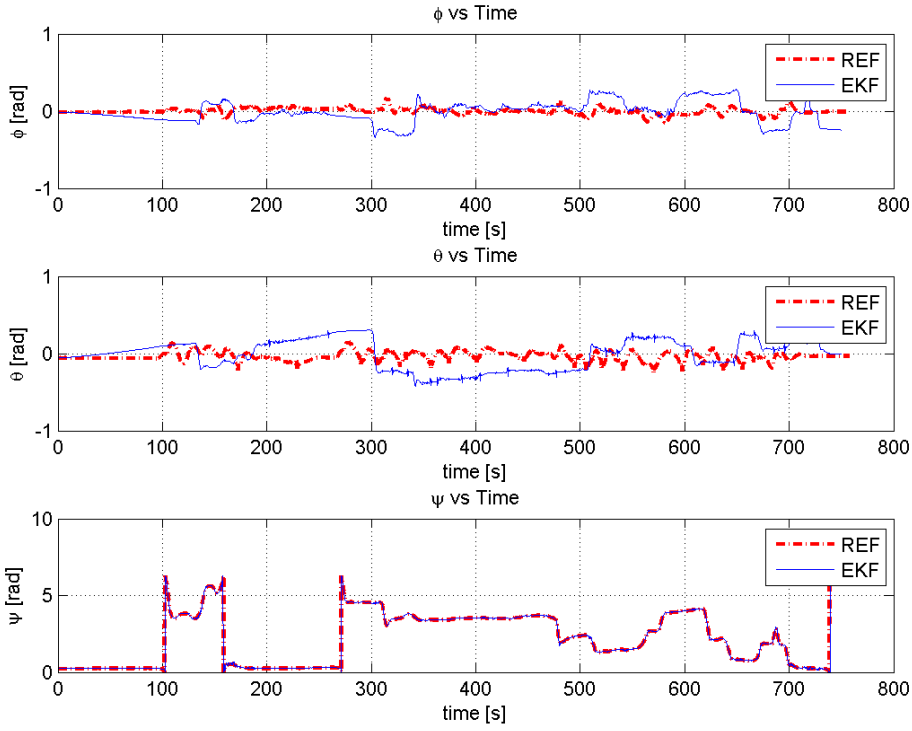



Figure 6.47: EKF attitude results for road test data

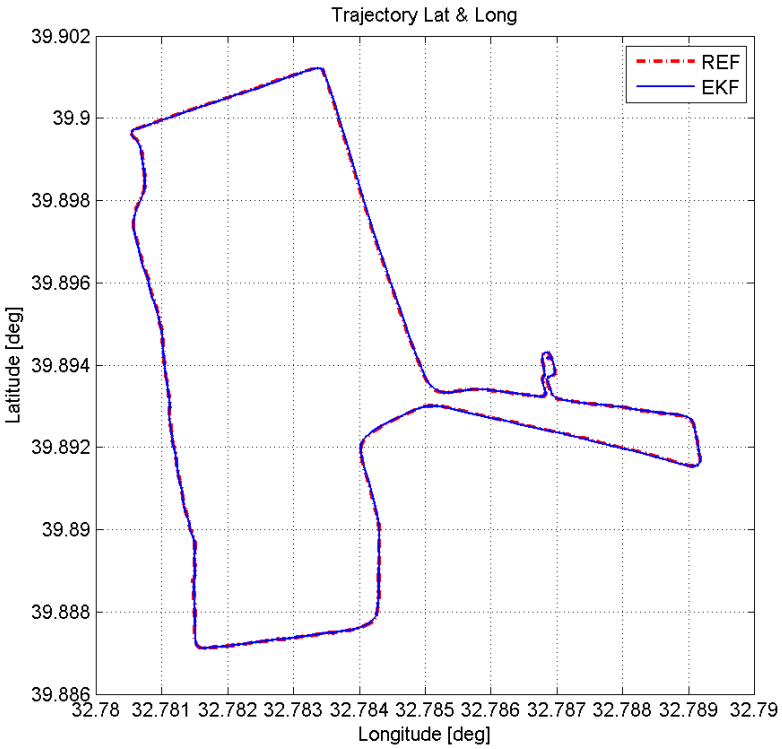



Figure 6.48: EKF latitude and longitude for road test data

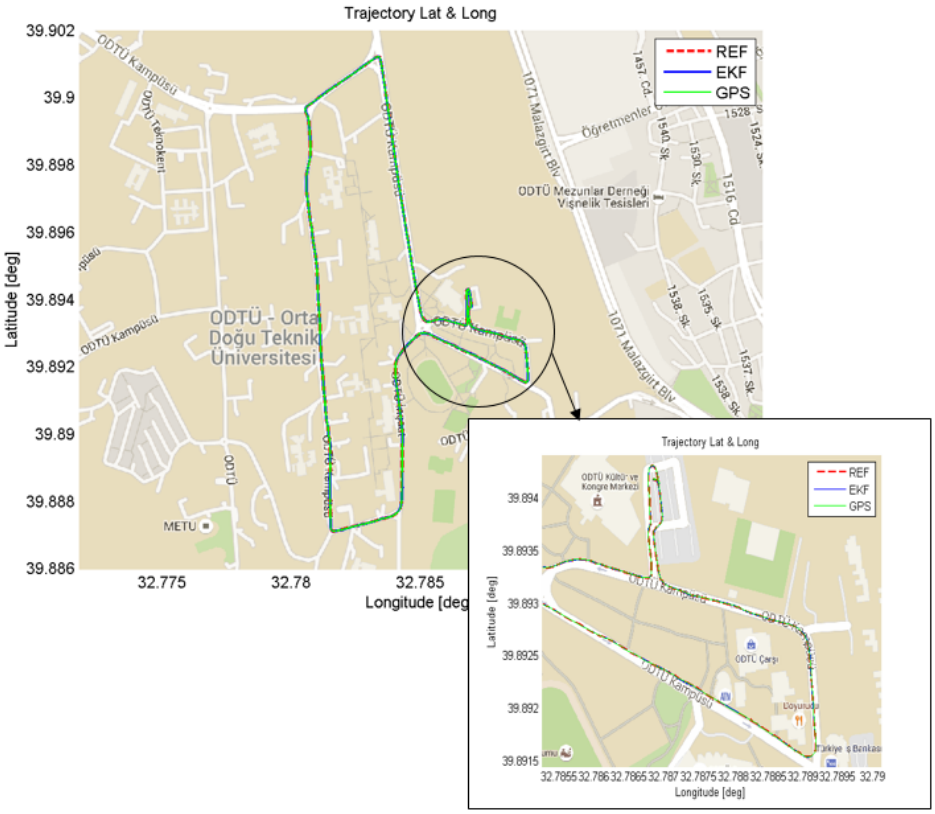



Figure 6.49: EKF latitude and longitude for road test data on google map

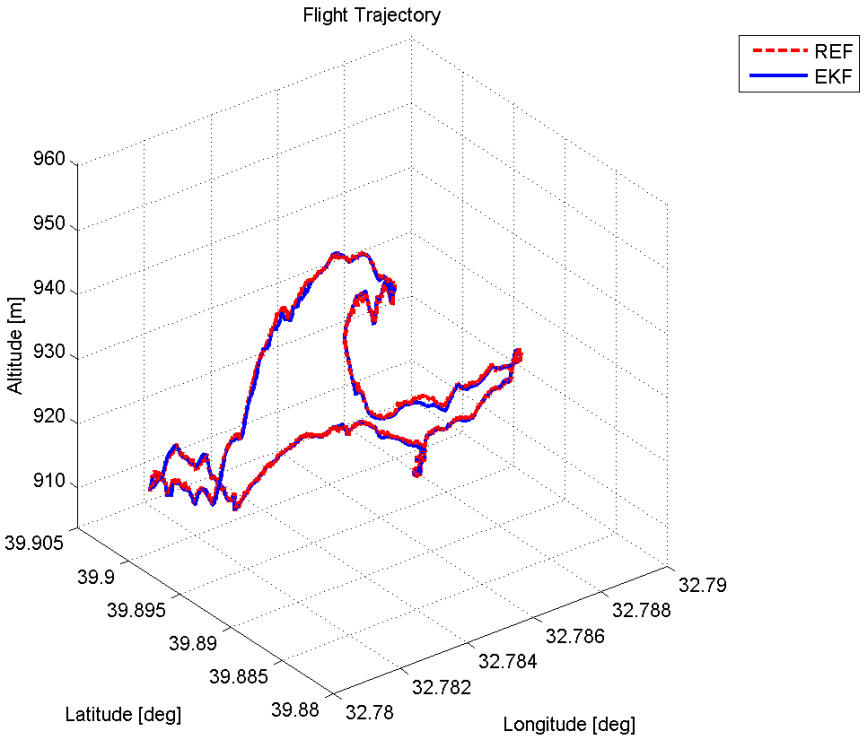



Figure 6.50: EKF 3D flight for road test data

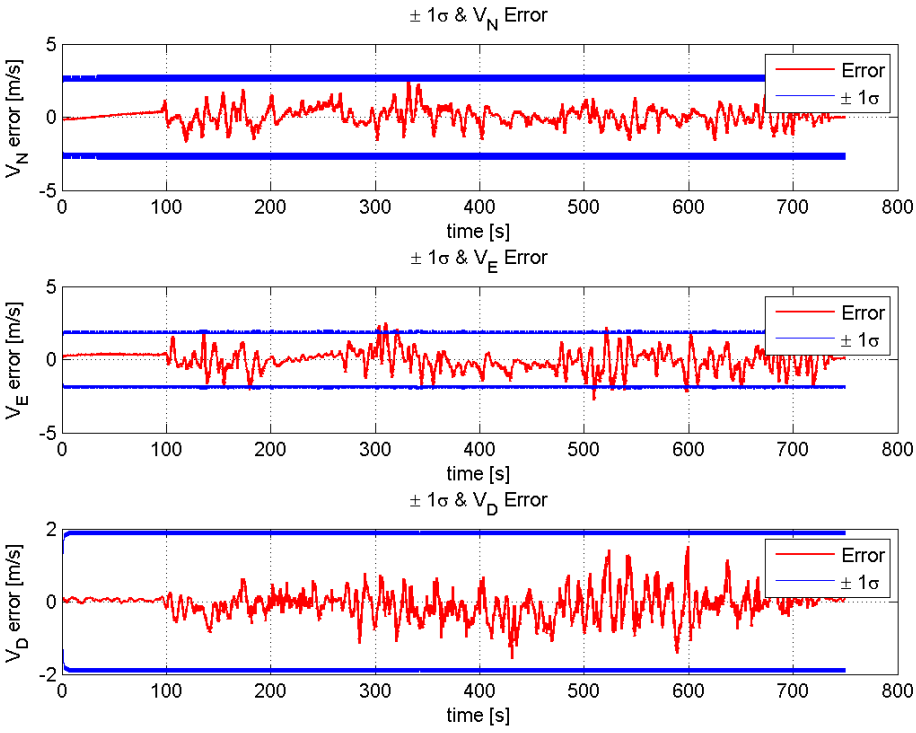



Figure 6.51: EKF velocity estimation error between $\pm 1\sigma$ bound for road test data

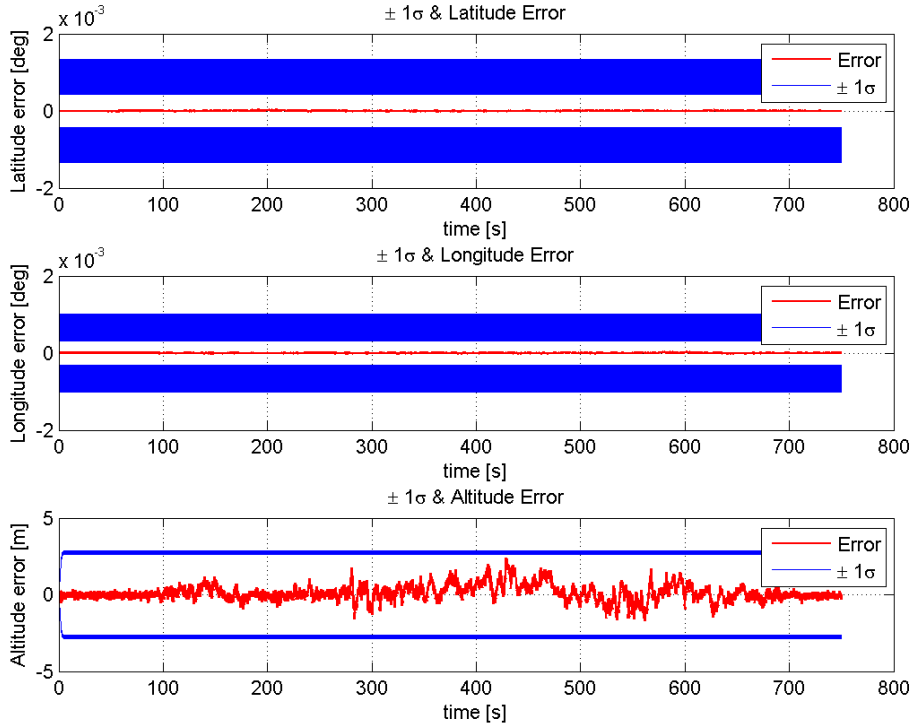



Figure 6.52: EKF position estimation error between $\pm 1\sigma$ bound for road test data

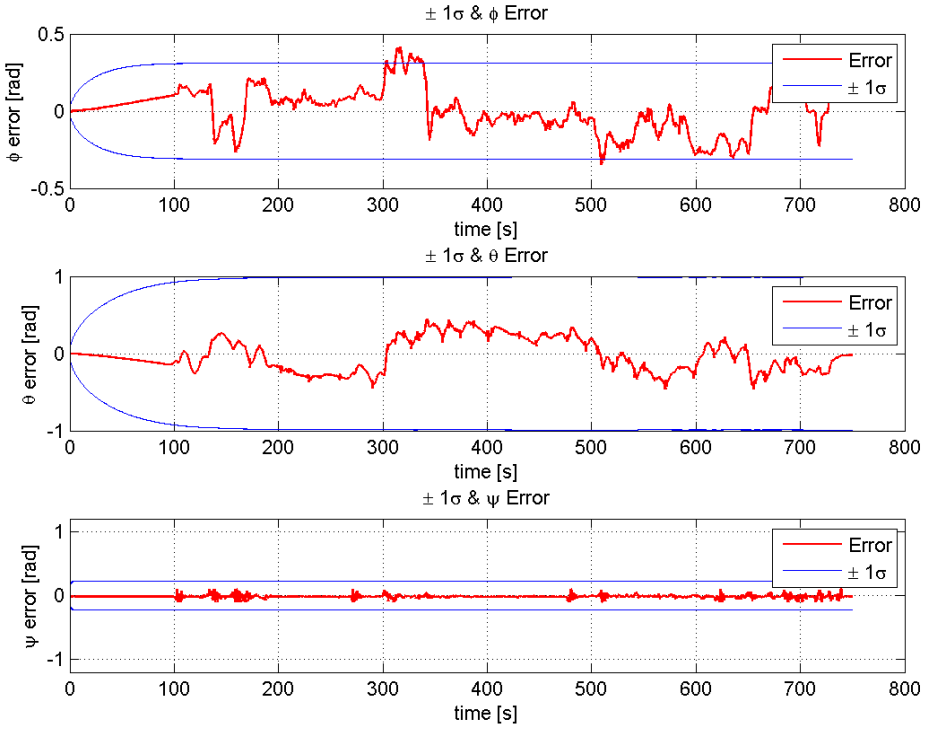



Figure 6.53: EKF attitude estimation error between $\pm 1\sigma$ bound for road test data

Table 6.5: INS/GPS model RMSE results for road test data

| Positions | Latitude [deg] | Longitude [deg] | Altitude [m] |
|---|---|---|---|
| RMSE values | 1.0988e-04 | 1.0227e-04 | 1.1006 |

**Experimental Road Test Results with MLPNN**

Now the MLPNN performance will be shown. For MLPNN test, artificially the GPS signal is blocked between 150-200 seconds of road test. During this time interval EKF performance without MLPNN is compared with EKF/MLPNN model when GPS signal is lost. Moreover, the root mean square errors are compared for latitude, longitude and altitude to show the improvement in case of MLPNN. Figures between 6.54 and 6.56 show the velocity, position and attitude EKF results without MLPNN when GPS signal is lost. Also 2D and 3D trajectory results of EKF without MLPNN are given in figures 6.57 and 6.58. The results with MLPNN are given between figures 6.59 and 6.61. 2D and 3D trajectory results of EKF with MLPNN are given in figures 6.62 and 6.63.

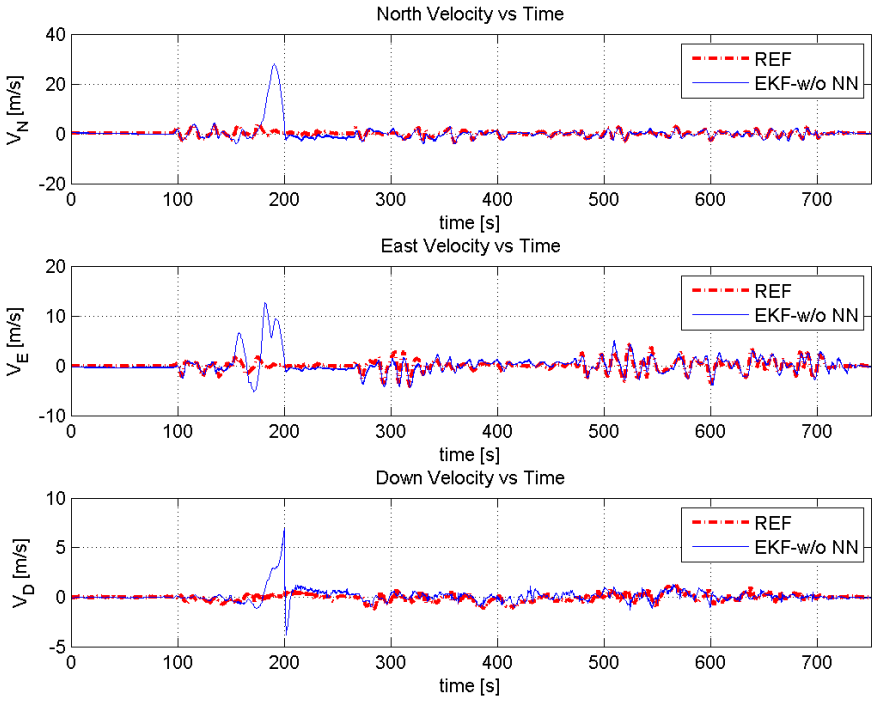



Figure 6.54: EKF velocity results for GPS outage between 150-200

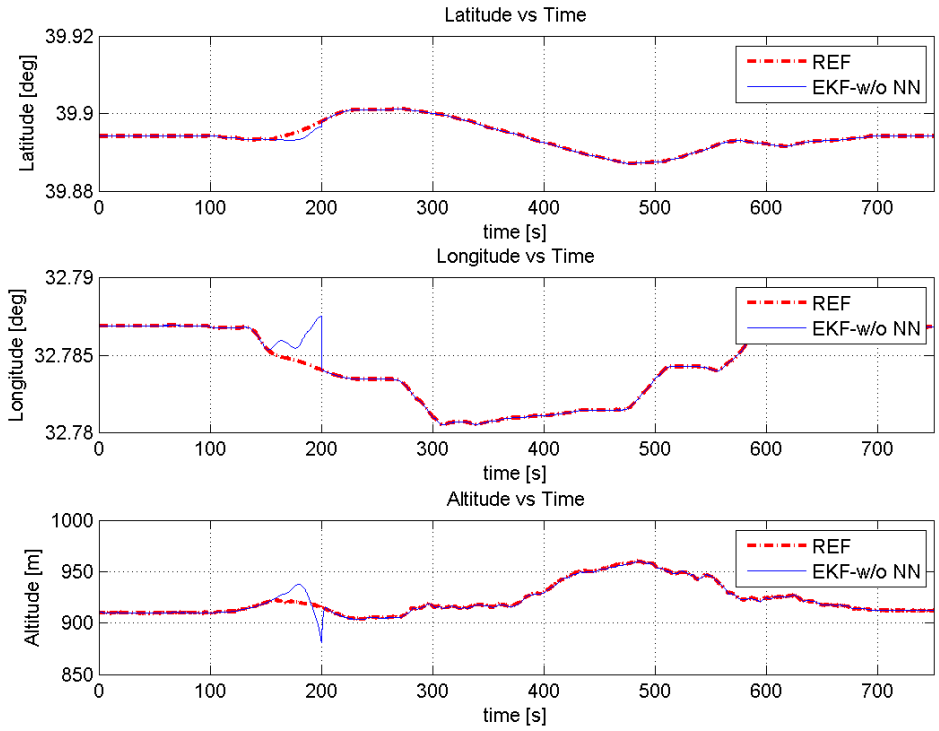



Figure 6.55: EKF position results for GPS outage between 150-200

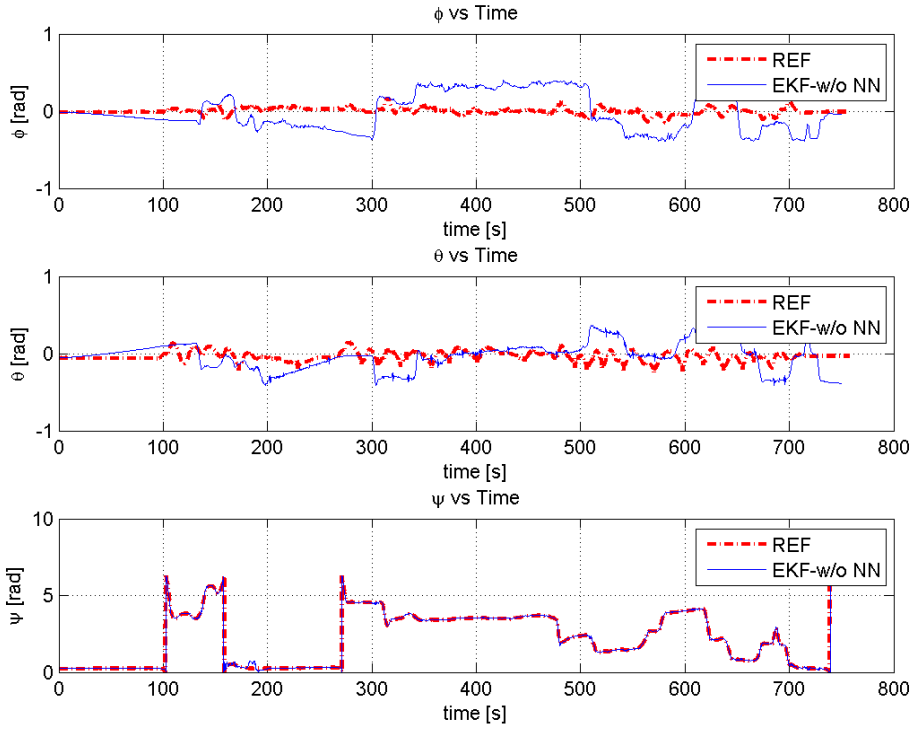



Figure 6.56: EKF attitude results for GPS outage between 150-200

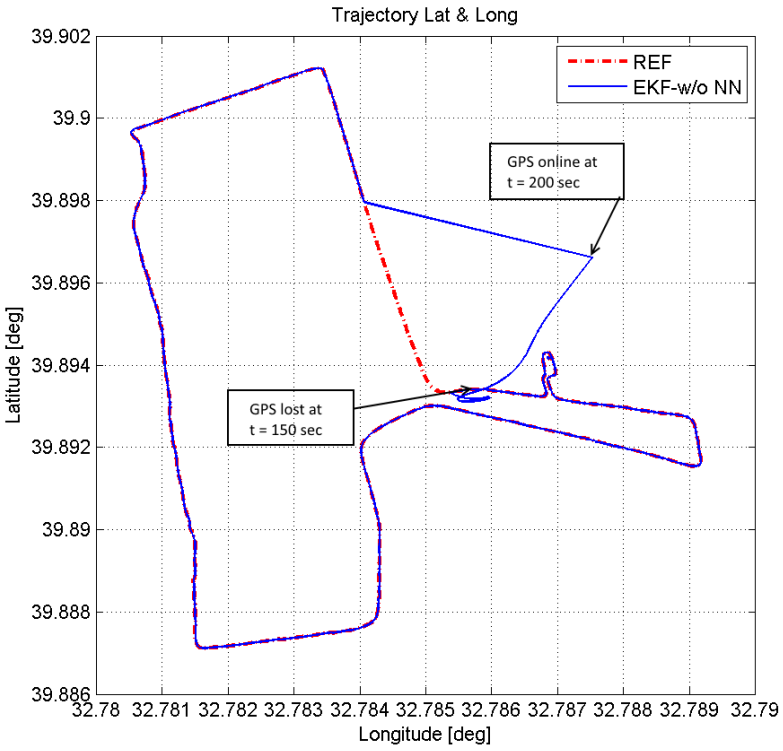



Figure 6.57: EKF latitude and longitude for GPS outage between 150-200

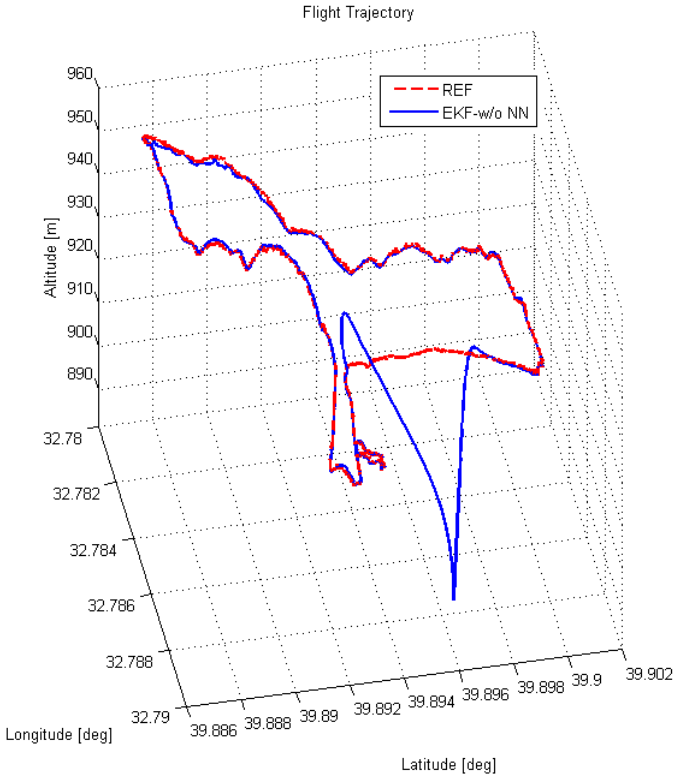



Figure 6.58: EKF 3D flight for GPS outage between 150-200

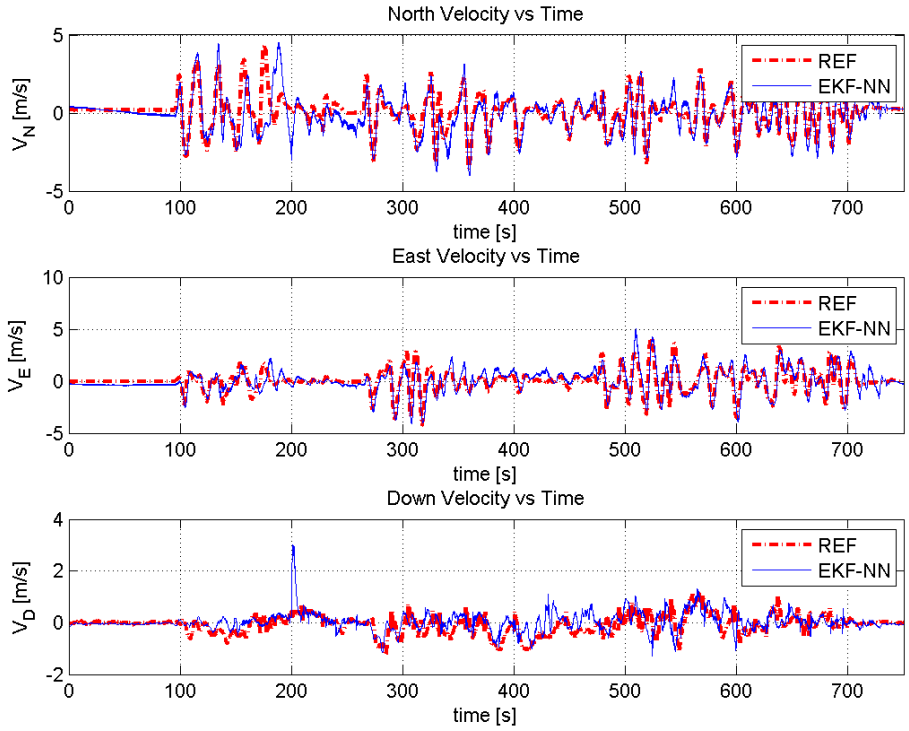



Figure 6.59: EKF velocity results for MLPNN between 150-200

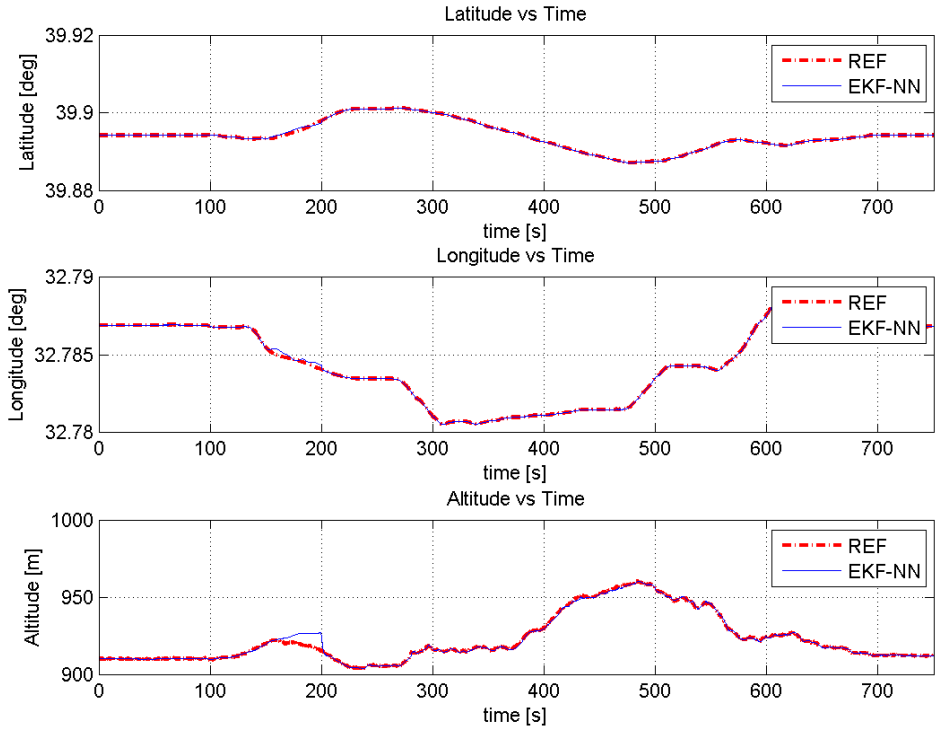



Figure 6.60: EKF position results for MLPNN between 150-200

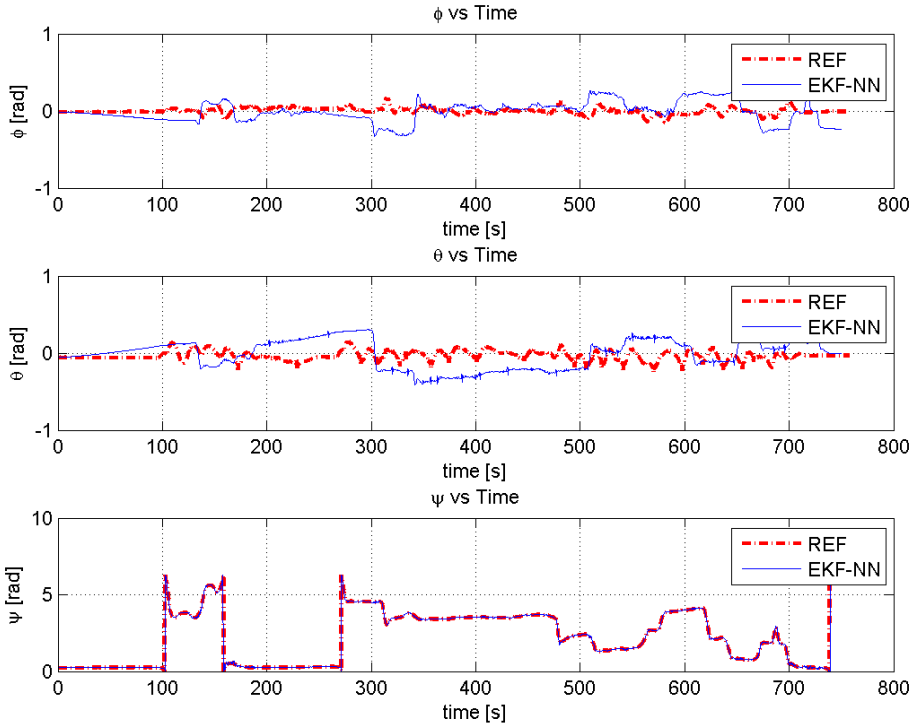



Figure 6.61: EKF attitude results for MLPNN between 150-200

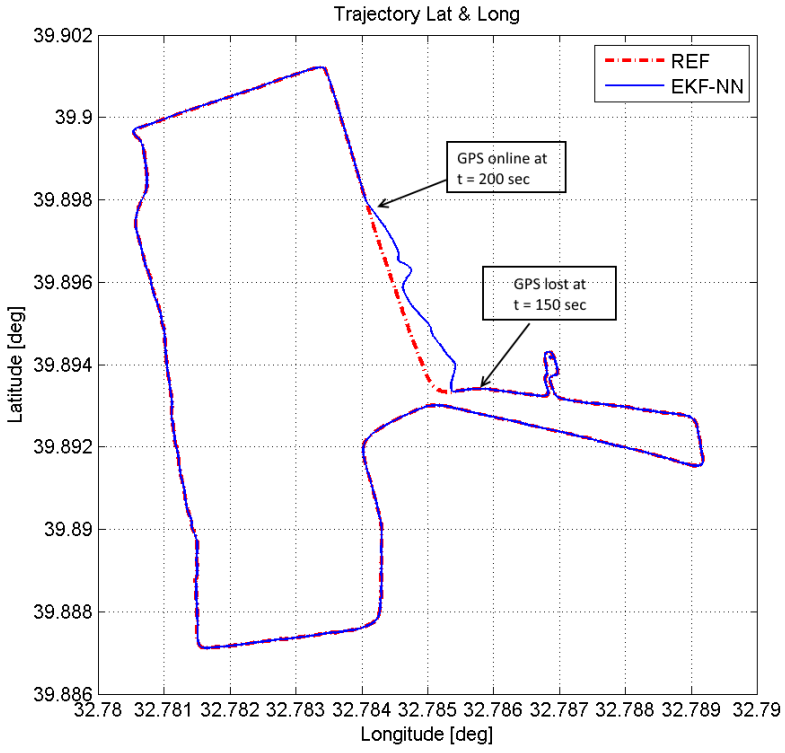



Figure 6.62: EKF latitude and longitude for MLPNN between 150-200

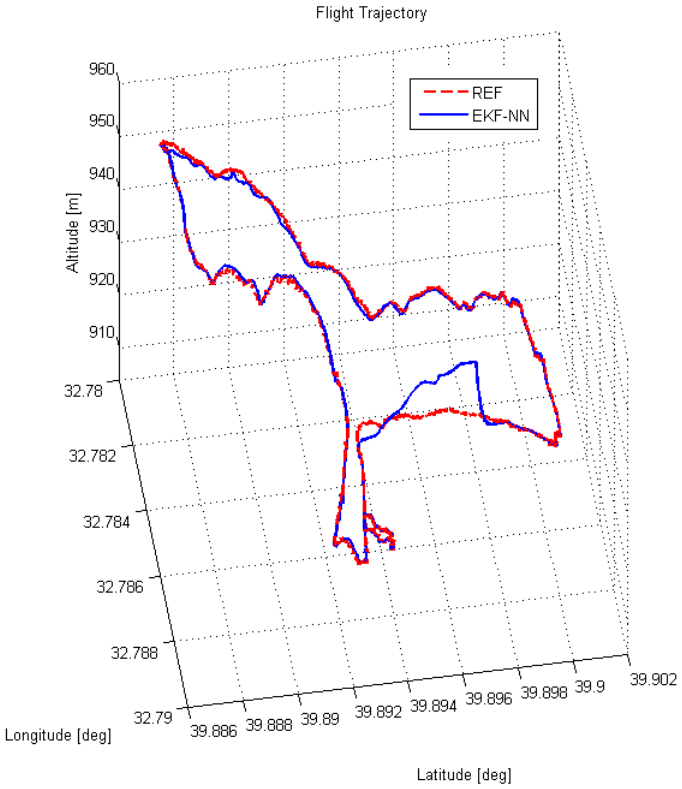



Figure 6.63: EKF 3D flight for MLPNN between 150-200

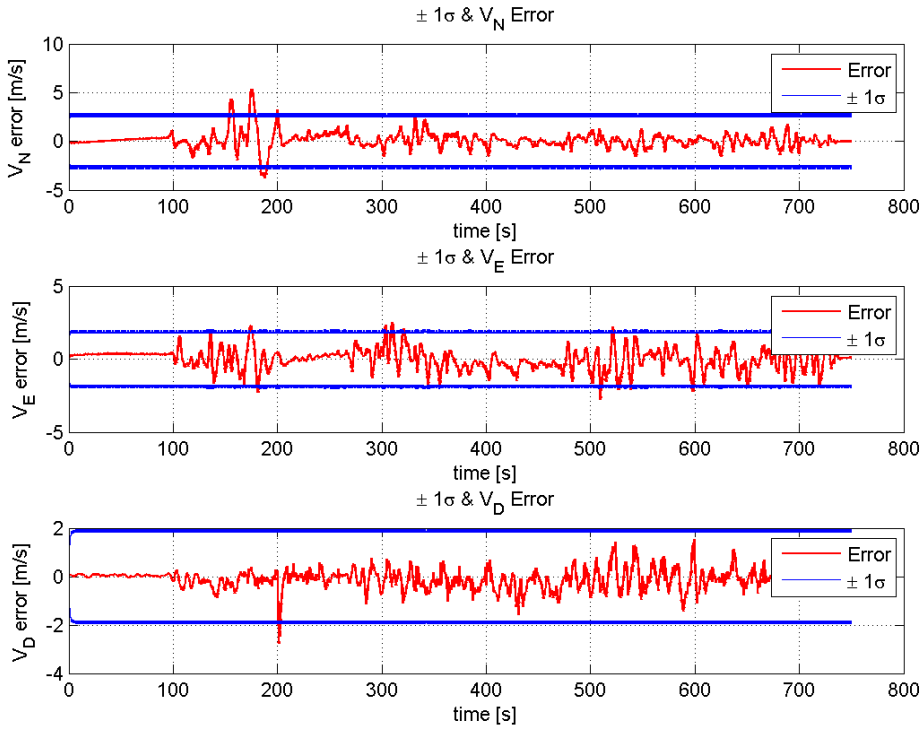



Figure 6.64: EKF velocity estimation error between $\pm 1\sigma$ bound for MLPNN between 150-200

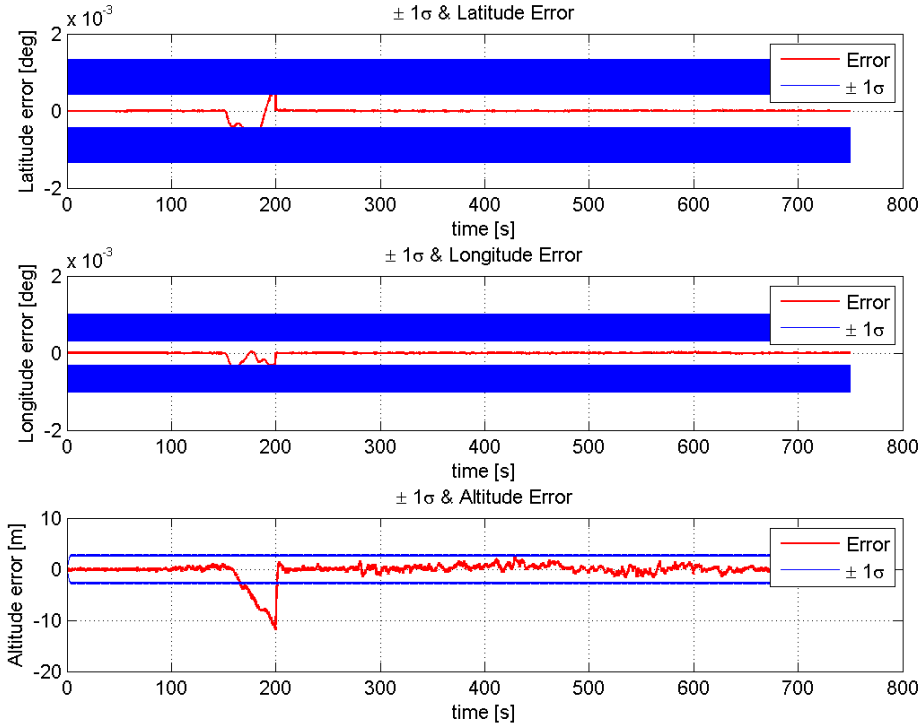



Figure 6.65: EKF position estimation error between $\pm 1\sigma$ bound for MLPNN between 150-200

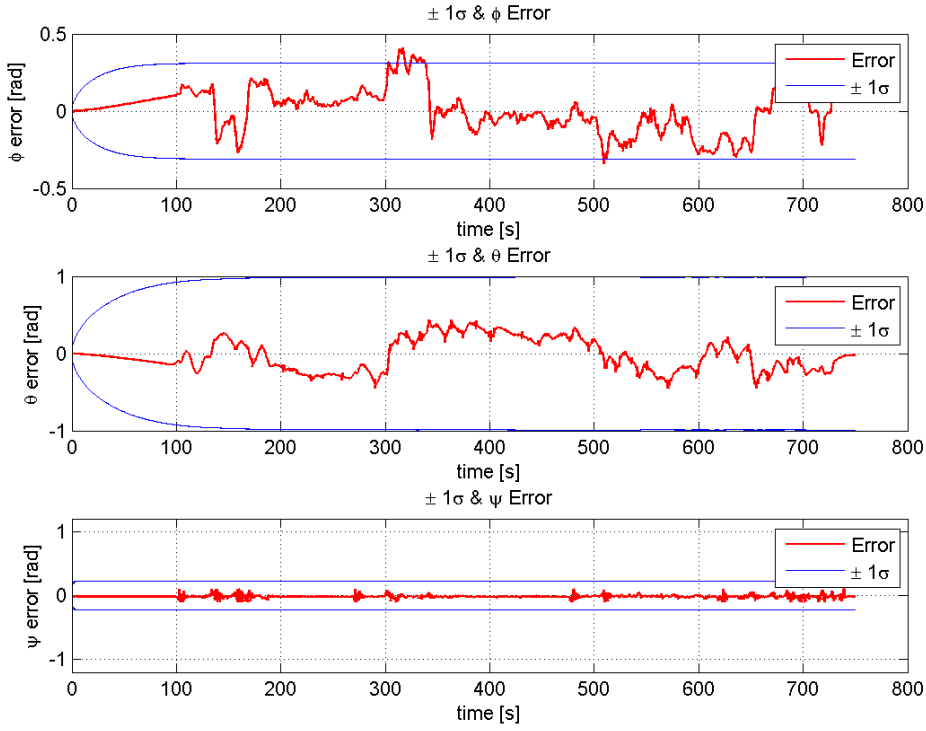



Figure 6.66: EKF attitude estimation error between $\pm 1\sigma$ bound for MLPNN between 150-200

The RMSE results between EKF/MLPNN and EKF without MLPNN are given in Table 6.7.

Table 6.6: Road test RMSE results only for GPS outage between 150-200 s

| Positions | RMSE for NN | RMSE w/o NN | Percentage Improvement |
|---|---|---|---|
| Latitude [deg] | 4.7550e-04 | 0.0014 | 66.8504% |
| Longitude [deg] | 2.4119e-04 | 0.0017 | 85.9383% |
| Altitude [m] | 6.1280 | 11.9866 | 48.8762% |

Table 6.7: Road test RMSE results GPS outage between 150-200 s for all trajectory

| Positions | RMSE for NN | RMSE w/o NN | Percentage Improvement |
|---|---|---|---|
| Latitude [deg] | 1.5466e-04 | 3.9548e-04 | 60.8934% |
| Longitude [deg] | 1.0962e-04 | 4.4857e-04 | 75.5633% |
| Altitude [m] | 1.9517 | 3.3578 | 41.8769% |

As it can be seen from the Table 6.6and Table 6.7,the improvement in latitude is almost 66%, in latitude 85% and in height 48% for GPS outage section and the improvement in latitude is almost 60%, in latitude 75% and in height 40% for all trajectory, respectively. Both GPS outage section RMSE and total trajectory RMSE gives consistent results. Of course these values are depend on the learning capabilities of neural network and the variety of data that is given to MLPNN to learn before GPS outage.

### 6.3.4 Experimental Flight Path Test Results

The experimental test results will be given under two section. In first section comparison between EKF solution and reference experimental data will be shared. In second section, MLPNN performance will be given by comparing MLPNN-EKF model with a second model which does not include MLPNN implementation. The artificial GPS signal outages cases are given in Table 6.8.

Table 6.8: The duration of GPS outages

| Outage Cases | Case 1 | Case 2 | Case 3 |
|---|---|---|---|
| Time Interval (s) | 55-85 | 80-120 | 180-200 |

**Experimental Flight Test Results with EKF**

In following figures the experimental data set results are compared with the simulink navigation mechanization and Kalman filter algorithm. The figures include acceleration, angular rates, velocity, position and attitude results respectively.

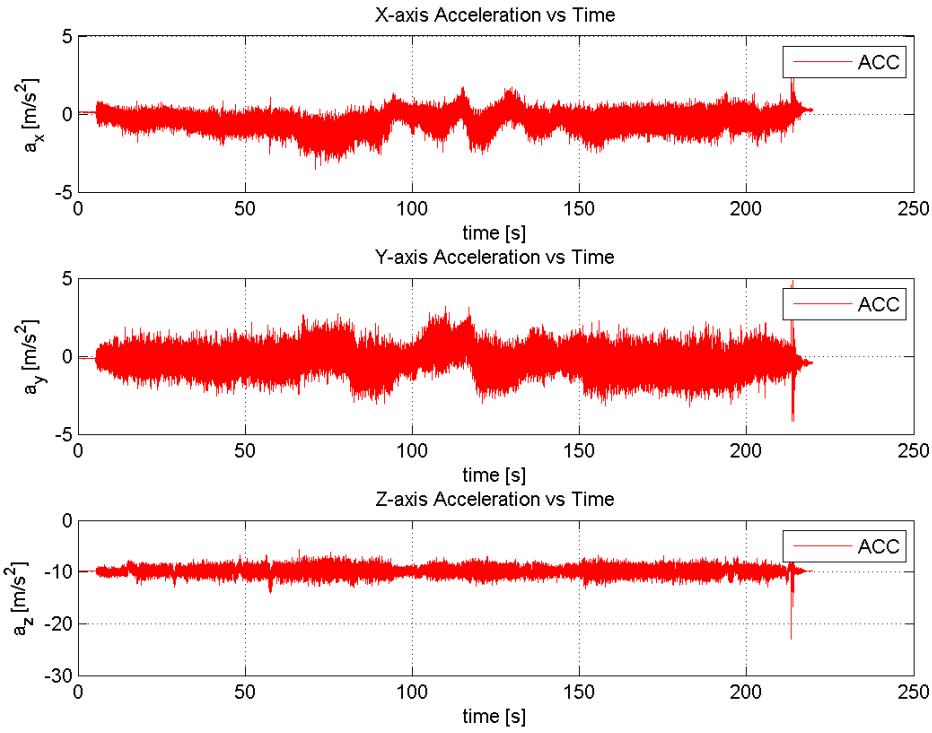



Figure 6.67: Three axis acceleration results for field test data

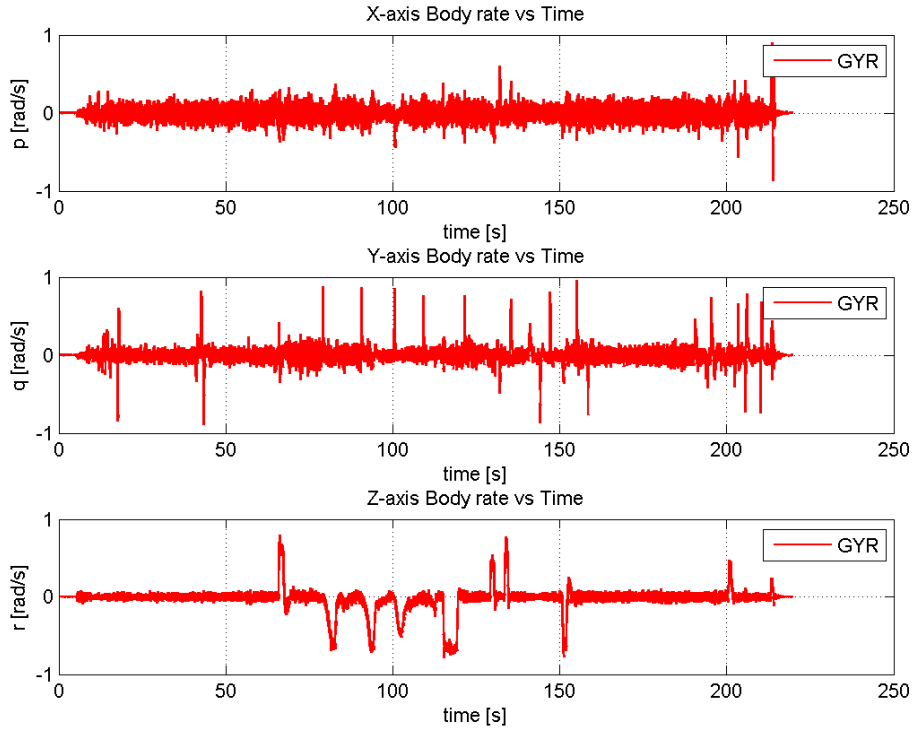



Figure 6.68: Three axis body angular rate results for field test data

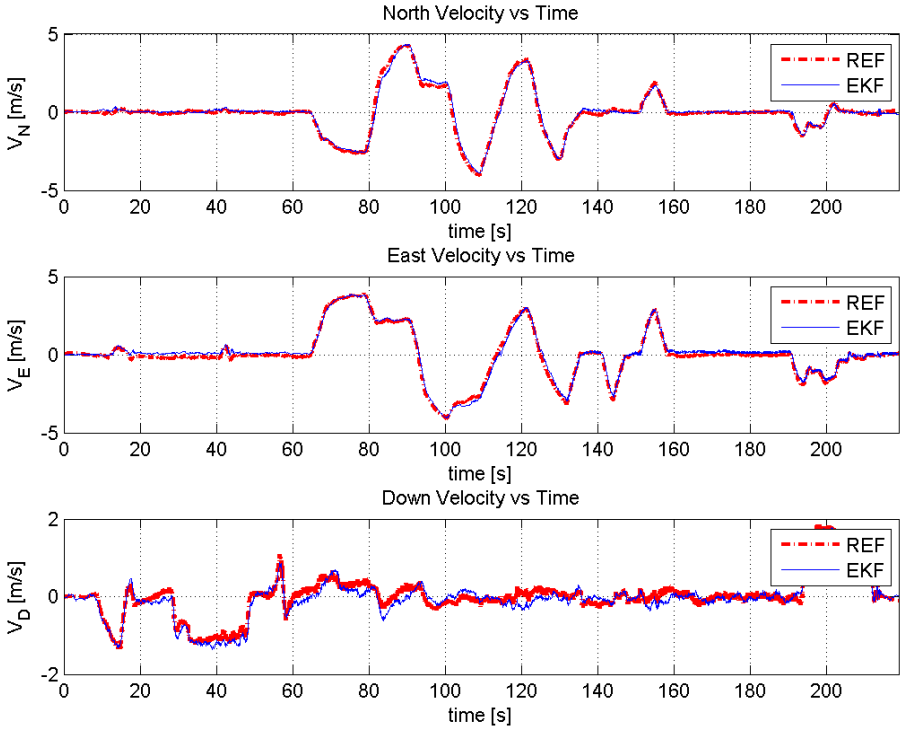



Figure 6.69: EKF velocity results for field test data

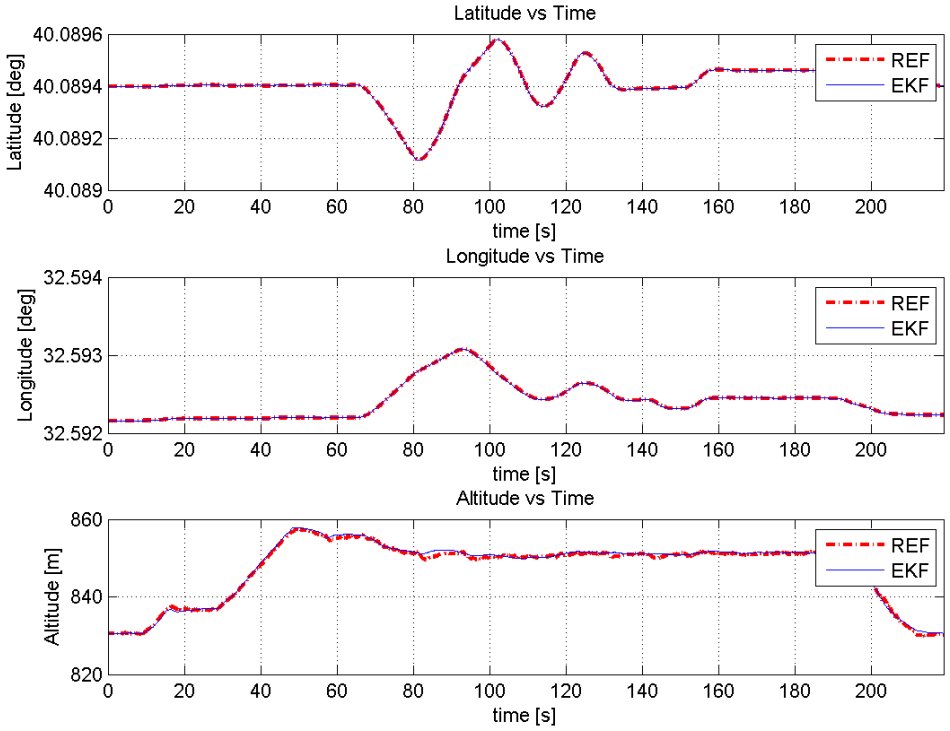



Figure 6.70: EKF position results for field test data

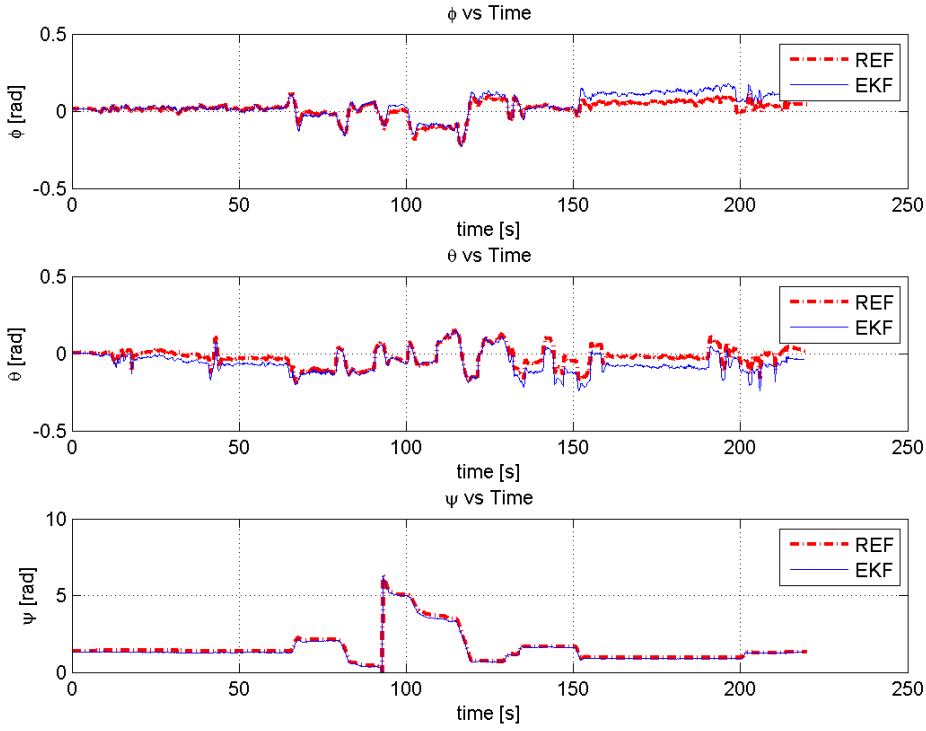



Figure 6.71: EKF attitude results for field test data

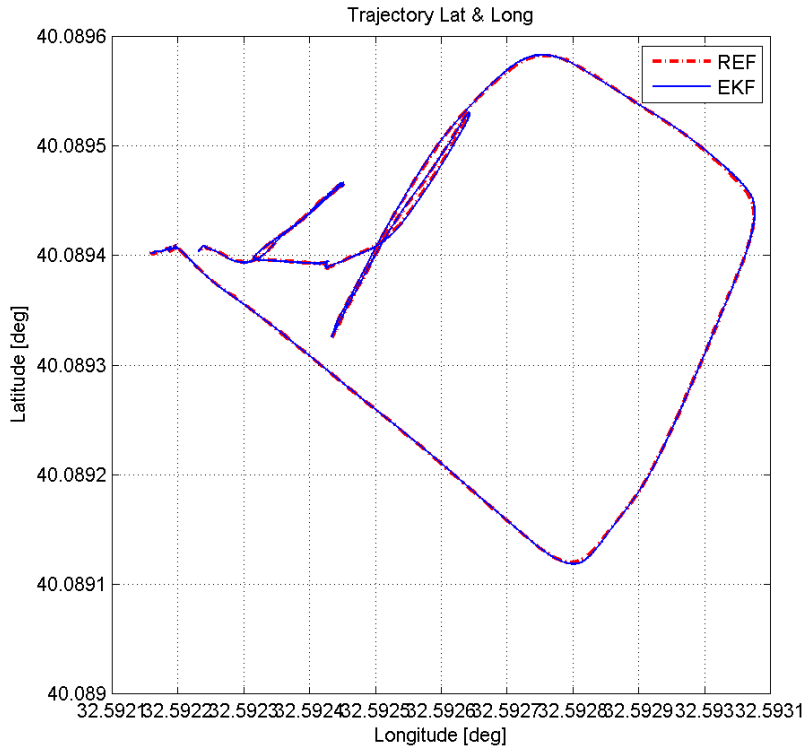



Figure 6.72: EKF latitude and longitude for field test data

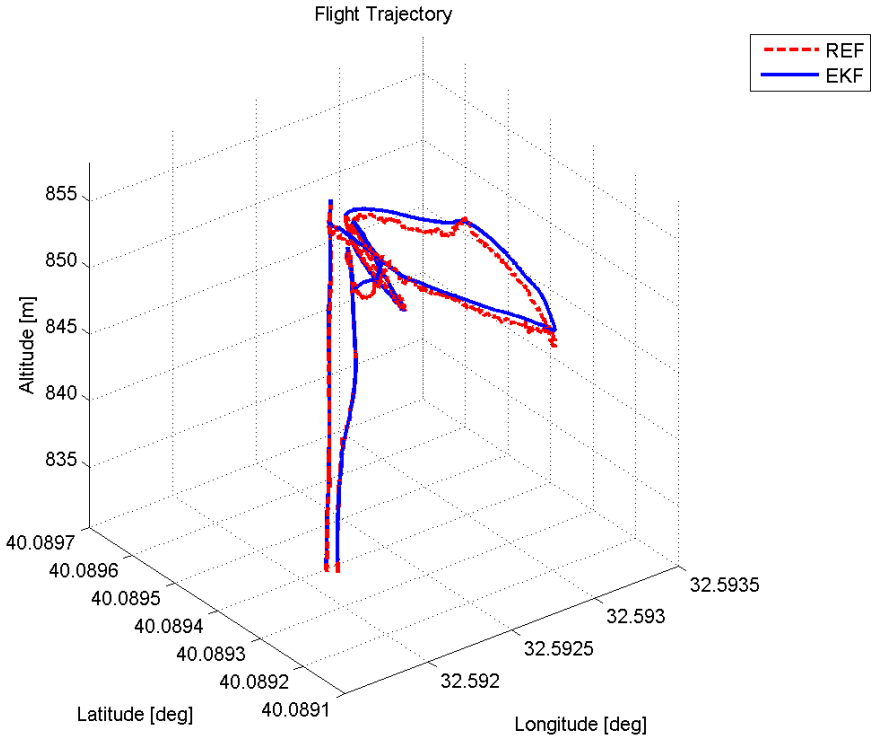



Figure 6.73: EKF 3D flight for field test data

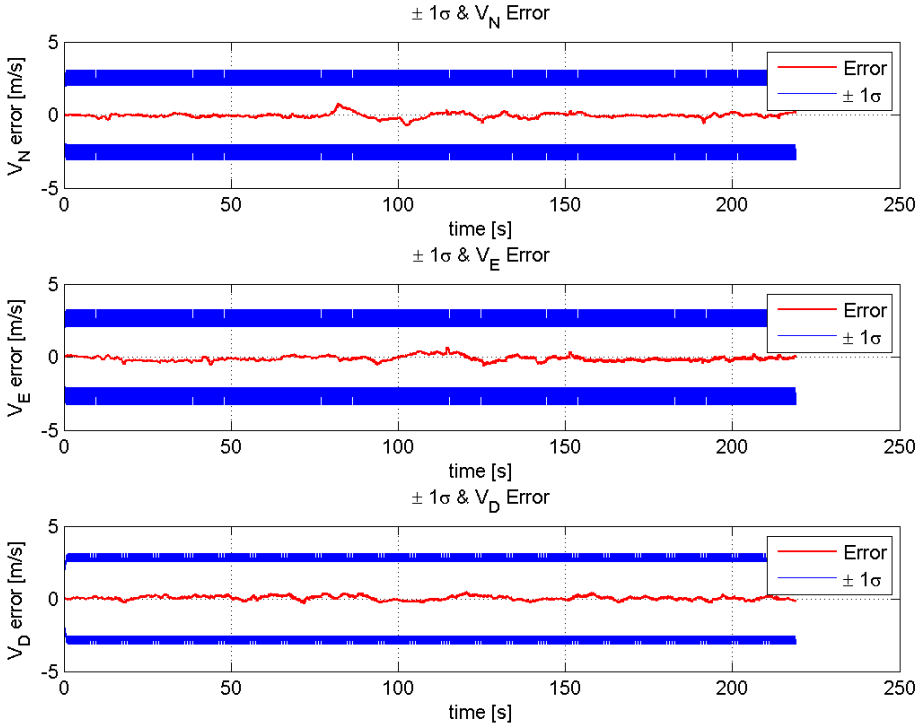



Figure 6.74: EKF velocity estimation error between $\pm 1\sigma$ bound for field test data

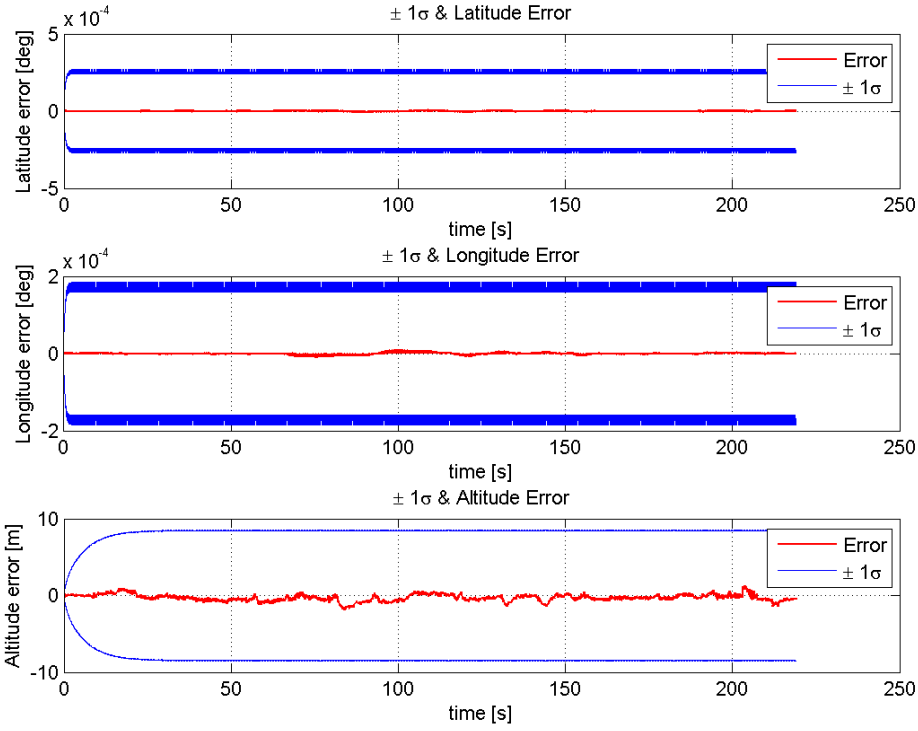



Figure 6.75: EKF position estimation error between $\pm 1\sigma$ bound for field test data

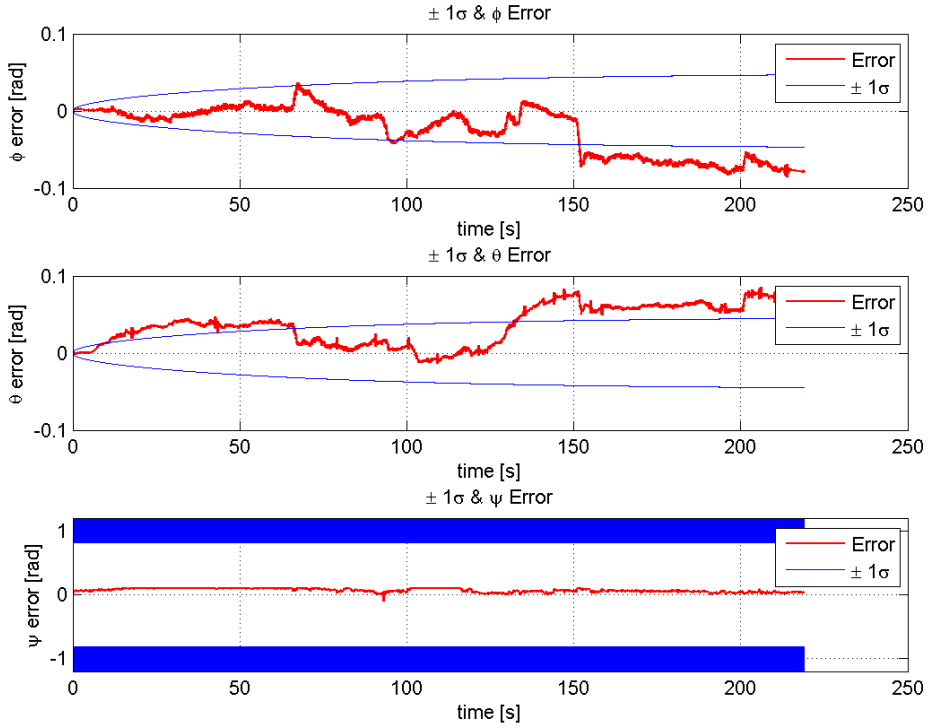



Figure 6.76: EKF attitude estimation error between $\pm 1\sigma$ bound for field test data

When figures 6.72, 6.73 are examined, it can be seen that EKF results are almost similar with reference field data which is obtained from sensor systems on quadrocopter. The error estimation performance of Kalman filter can be seen from figures 6.74, 6.75 and 6.76. The RMSE result of INS/GPS simulink model with respect to reference flight trajectory data is given in Table 6.9.

Table 6.9: INS/GPS model RMSE results for flight test data

| Positions | Latitude [deg] | Longitude [deg] | Altitude [m] |
|---|---|---|---|
| RMSE values | 1.0074e-05 | 1.6494e-05 | 0.9253 |

**Experimental Flight Test with MLPNN**

MLPNN performance results will be given for three gps outage cases which are shown in Table 6.8. The figures are shown in below. Respectively,for case 1 the figures between 6.77 and 6.81 show without MLPNN implementation and the figures between 6.82 and 6.86 show the MLPNN implementation case. Error estimation

results for MLPNN are given in figures 6.87,6.88 and 6.88. For case 2,the figures between 6.90 and 6.94 show without MLPNN implementation and the figures between 6.95 and 6.99 show the MLPNN implementation case. Error estimation results for MLPNN are given in figures 6.100, 6.101 and 6.101. And for the final case, the figures between 6.103 and 6.107 show without MLPNN implementation and the figures between 6.108 and 6.112 show the MLPNN implementation case. Error estimation results for MLPNN are given in figures 6.113, 6.114 and 6.114.

**GPS outages between 55-85**

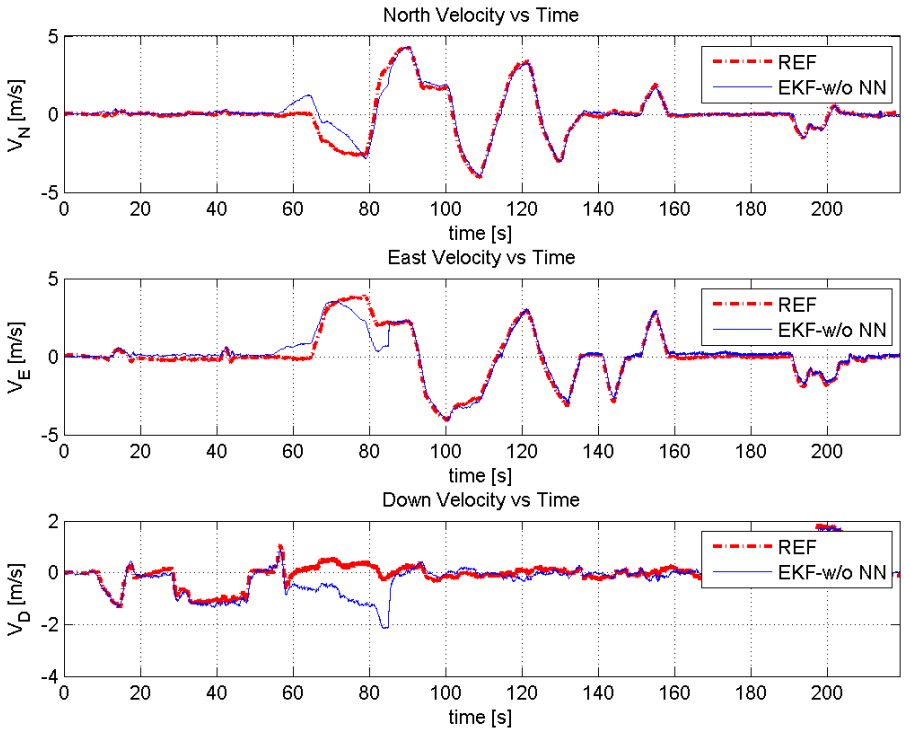



Figure 6.77: EKF velocity results for GPS outage between 55-85

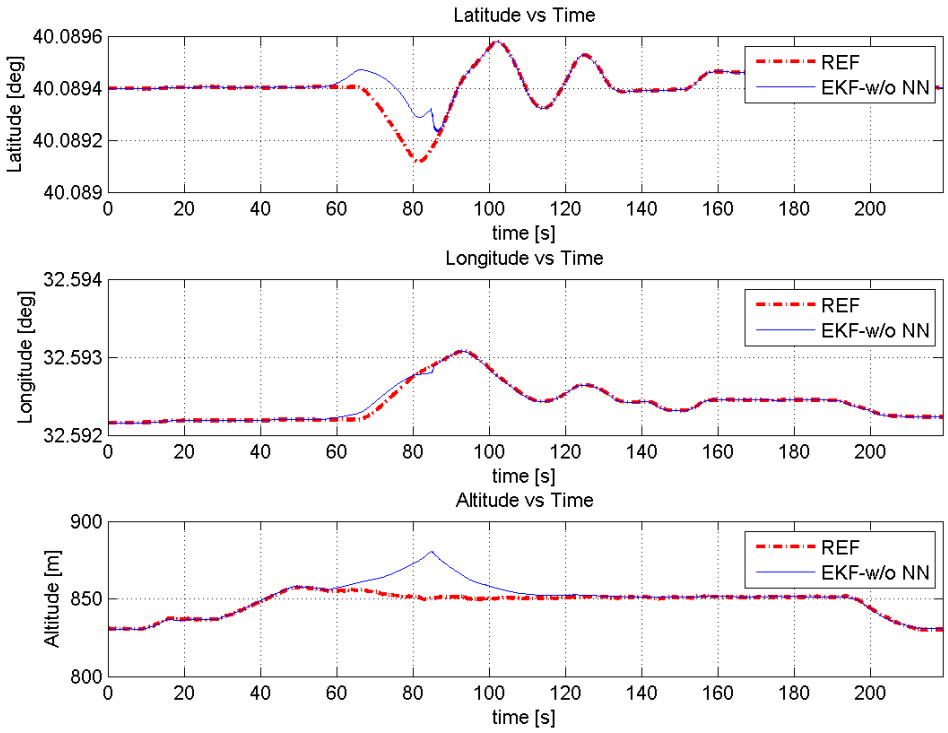



Figure 6.78: EKF position results for GPS outage between 55-85

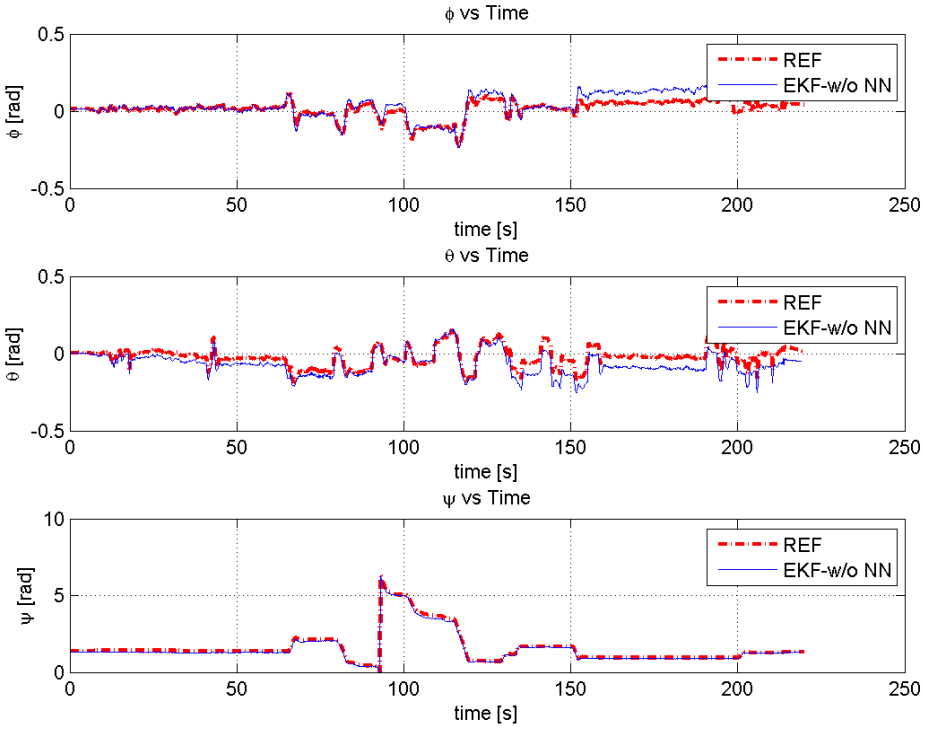



Figure 6.79: EKF attitude results for GPS outage between 55-85

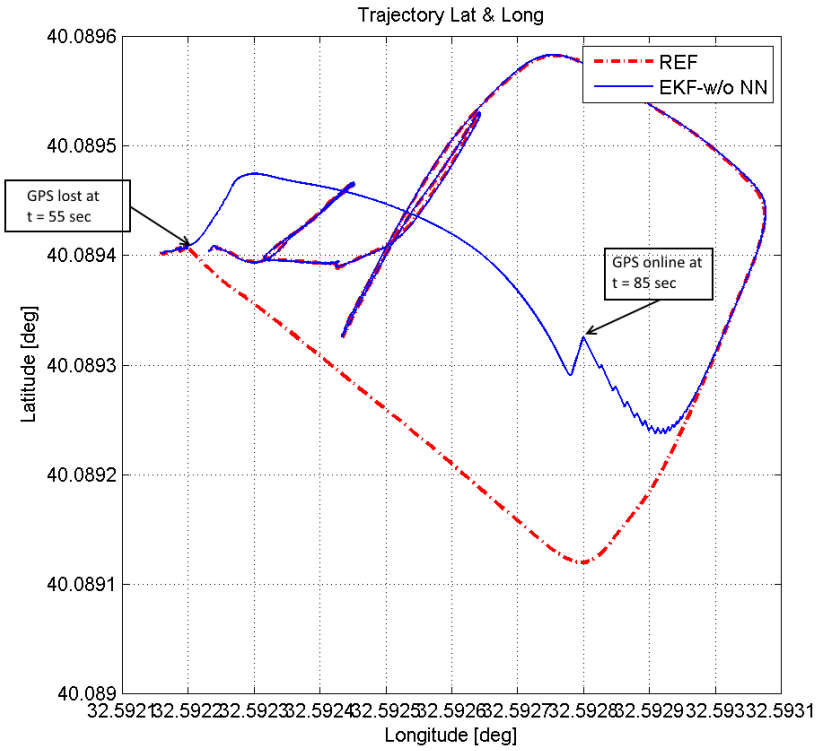



Figure 6.80: EKF latitude and longitude for GPS outage between 55-85

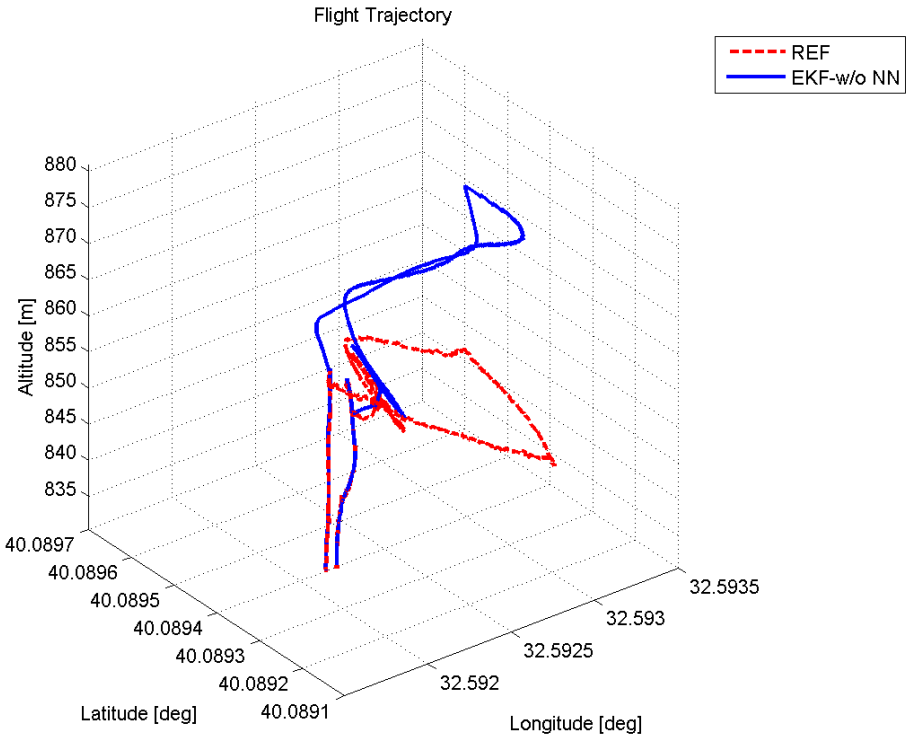



Figure 6.81: EKF 3D flight for GPS outage between 55-85

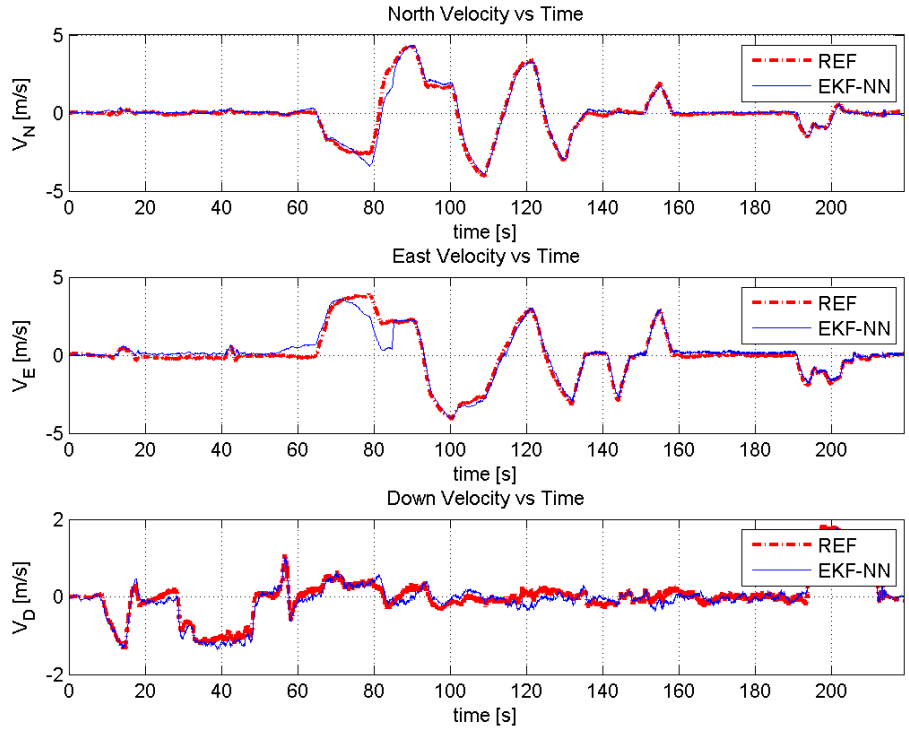



Figure 6.82: EKF velocity results for MLPNN between 55-85

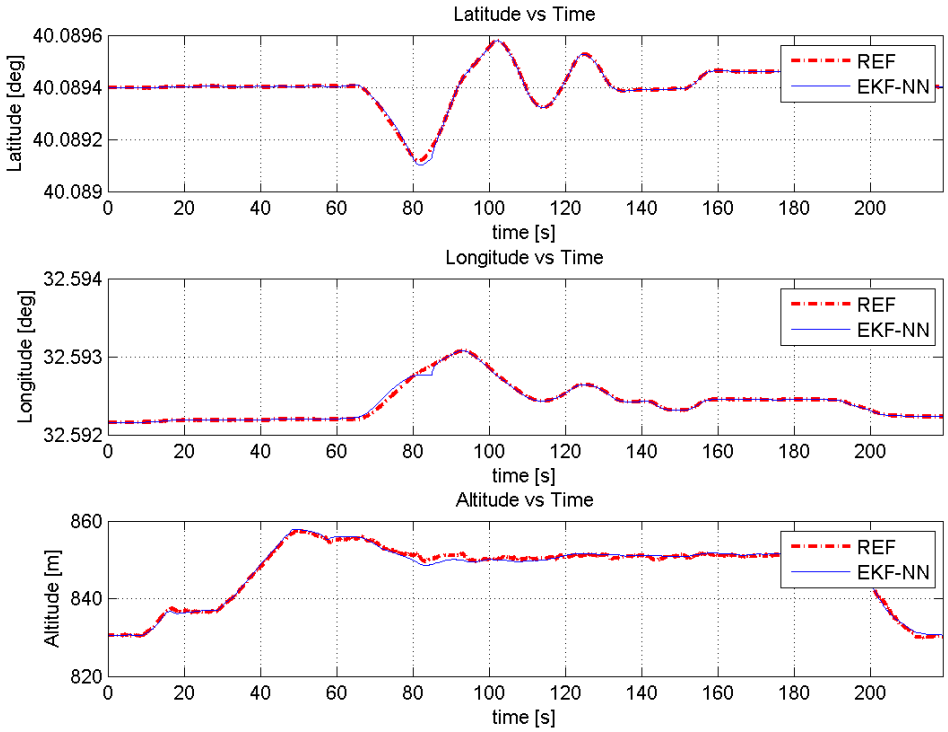



Figure 6.83: EKF position results for MLPNN between 55-85

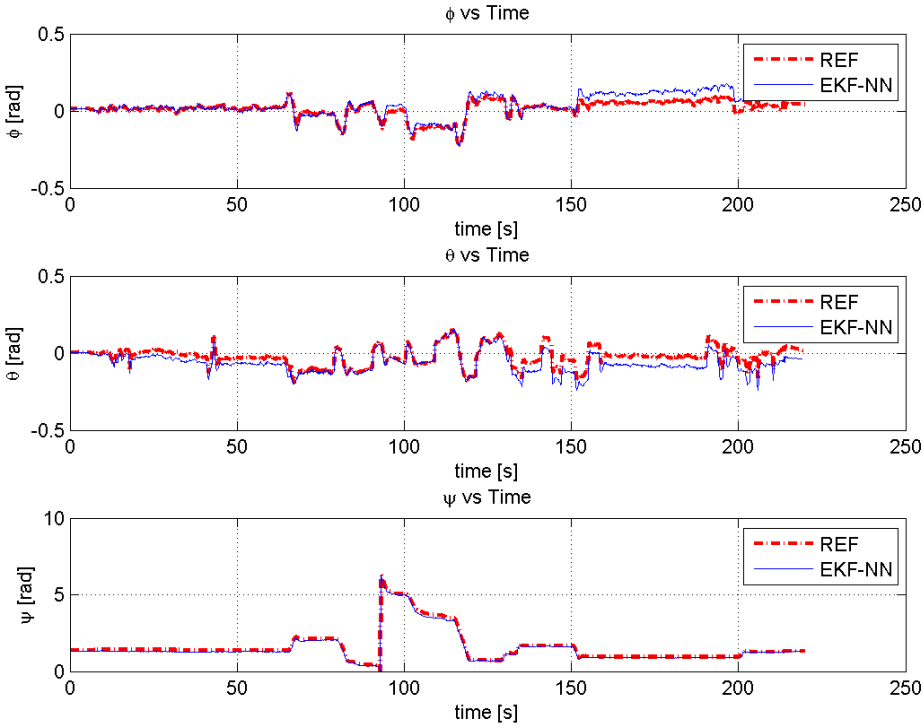



Figure 6.84: EKF attitude results for MLPNN between 55-85

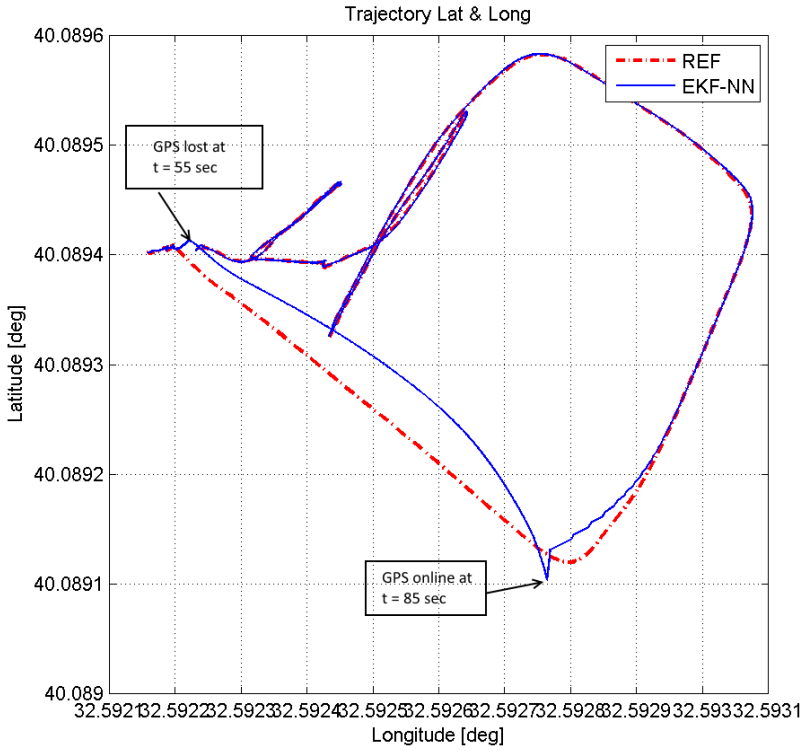



Figure 6.85: EKF latitude and longitude for MLPNN between 55-85

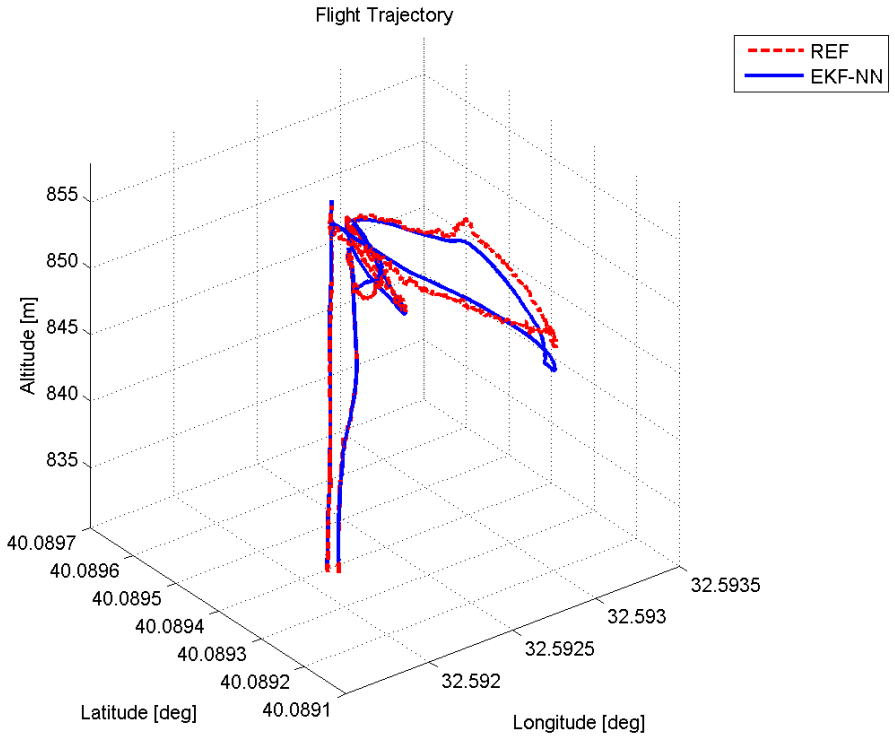



Figure 6.86: EKF 3D flight for MLPNN between 55-85

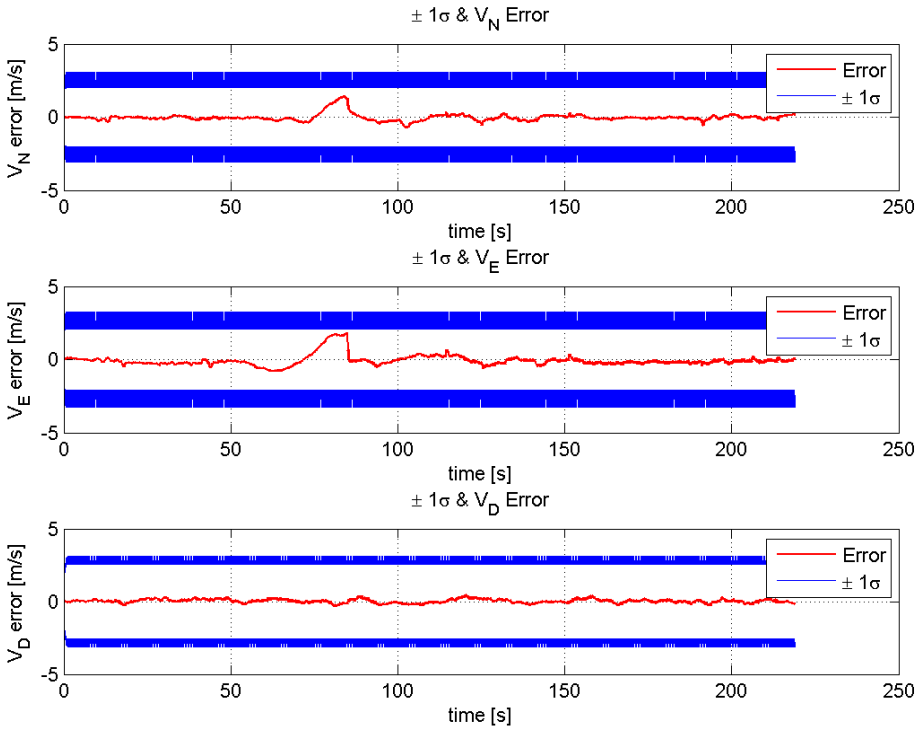



Figure 6.87: EKF velocity estimation error between $\pm 1\sigma$ bound for MLPNN between 55-85

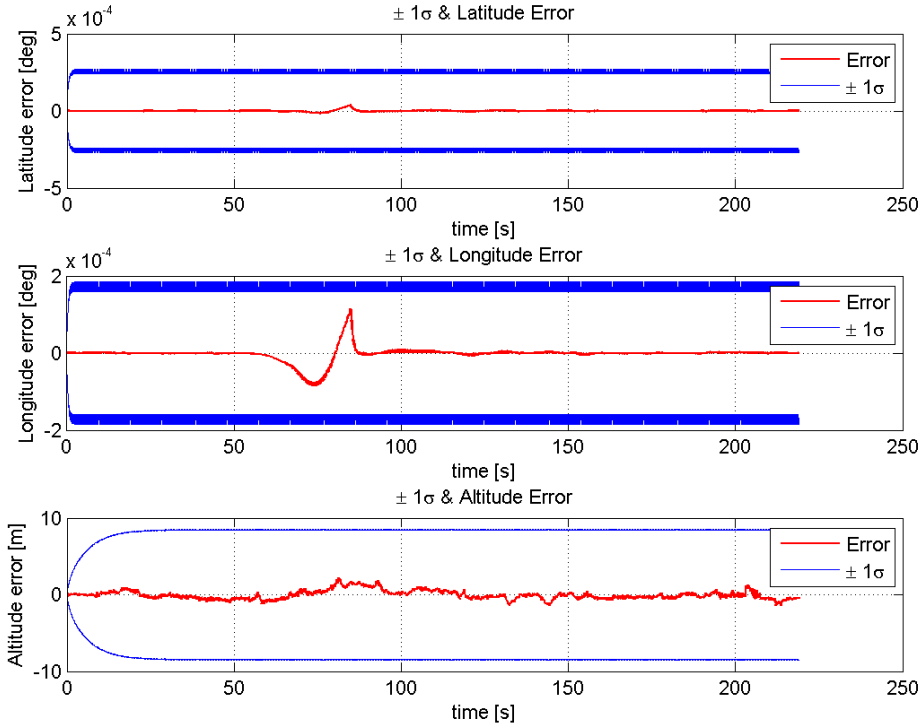



Figure 6.88: EKF position estimation error between $\pm 1\sigma$ bound for MLPNN between 55-85

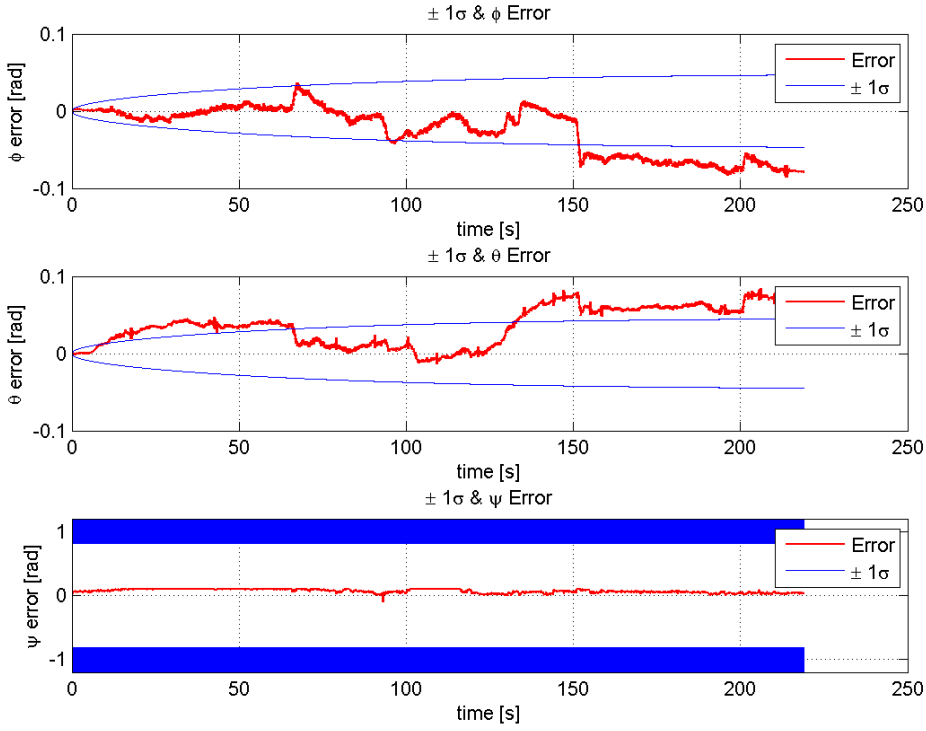



Figure 6.89: EKF attitude estimation error between $\pm 1\sigma$ bound for MLPNN between 55-85

**GPS outages between 80-120**

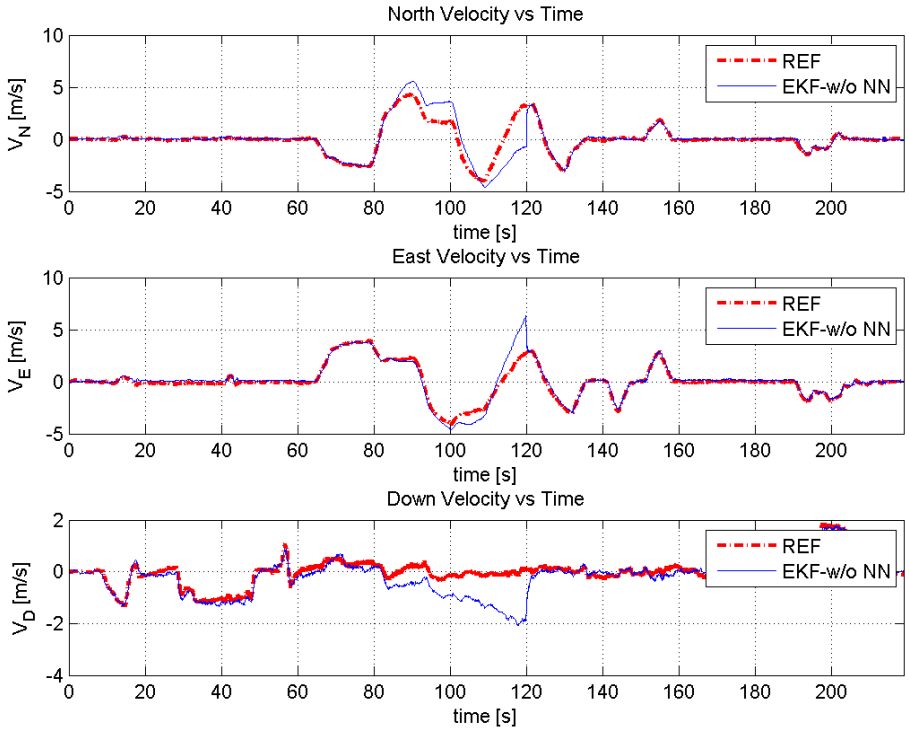



Figure 6.90: EKF velocity results for GPS outage between 80-120

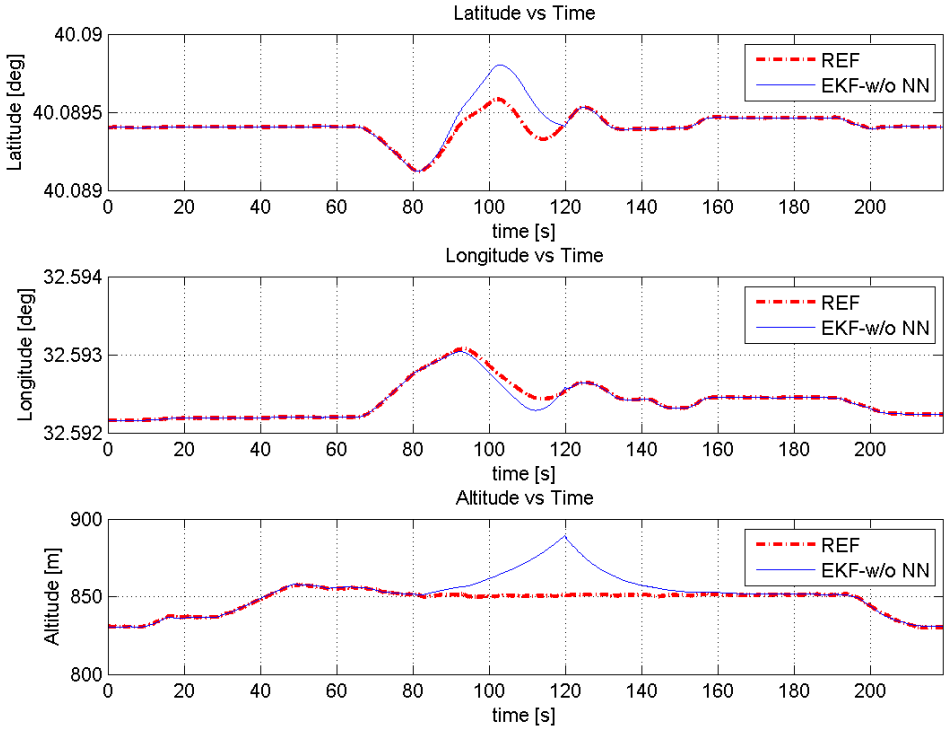



Figure 6.91: EKF position results for GPS outage between 80-120

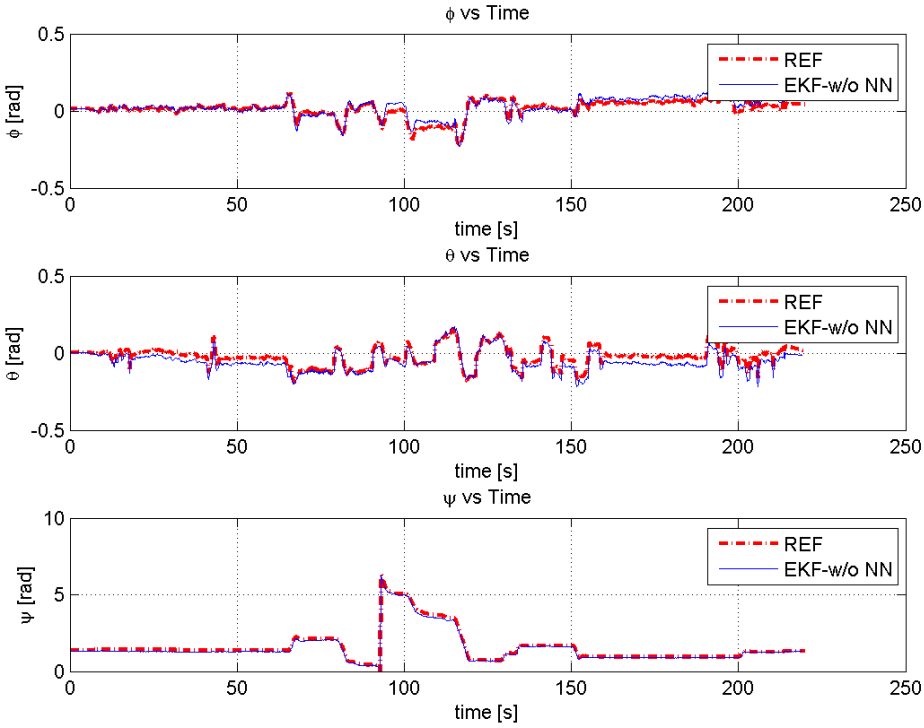



Figure 6.92: EKF attitude results for GPS outage between 80-120

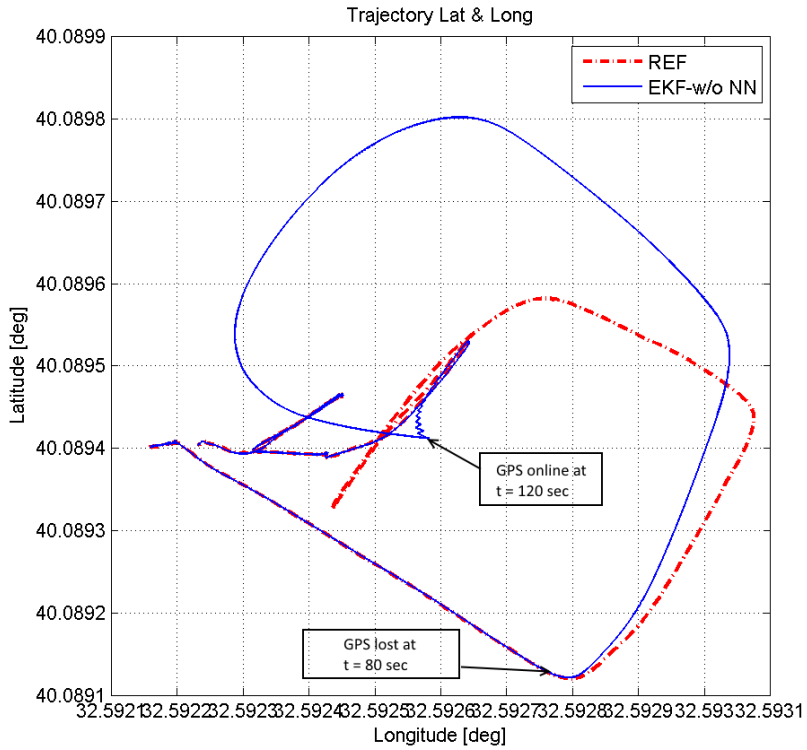



Figure 6.93: EKF latitude and longitude for GPS outage between 80-120

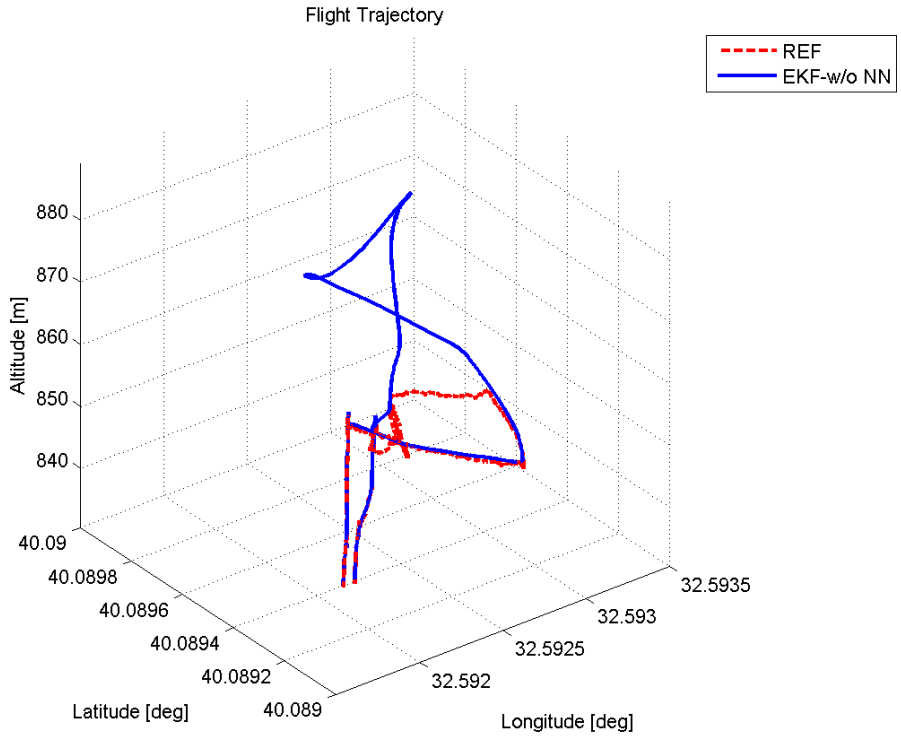



Figure 6.94: EKF 3D flight for GPS outage between 80-120

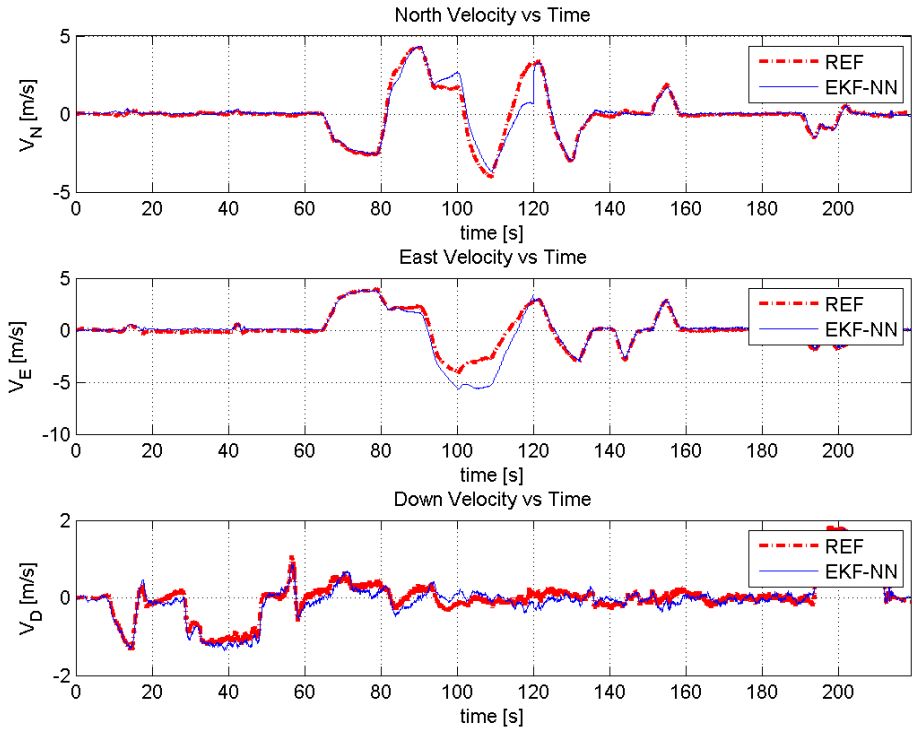



Figure 6.95: EKF velocity results for MLPNN between 80-120

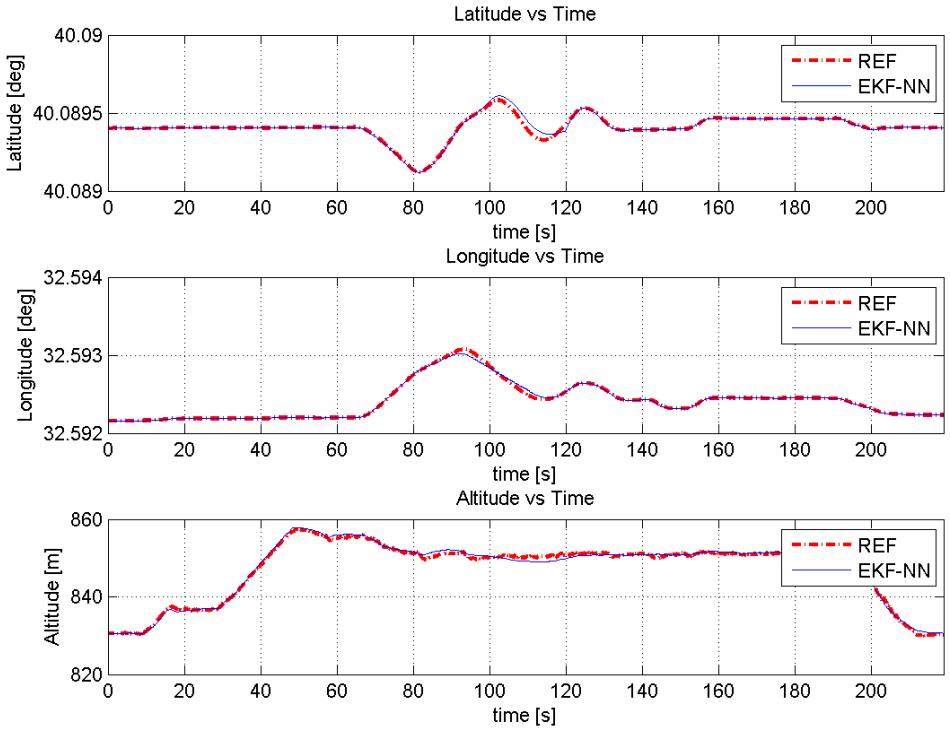



Figure 6.96: EKF position results for MLPNN between 80-120

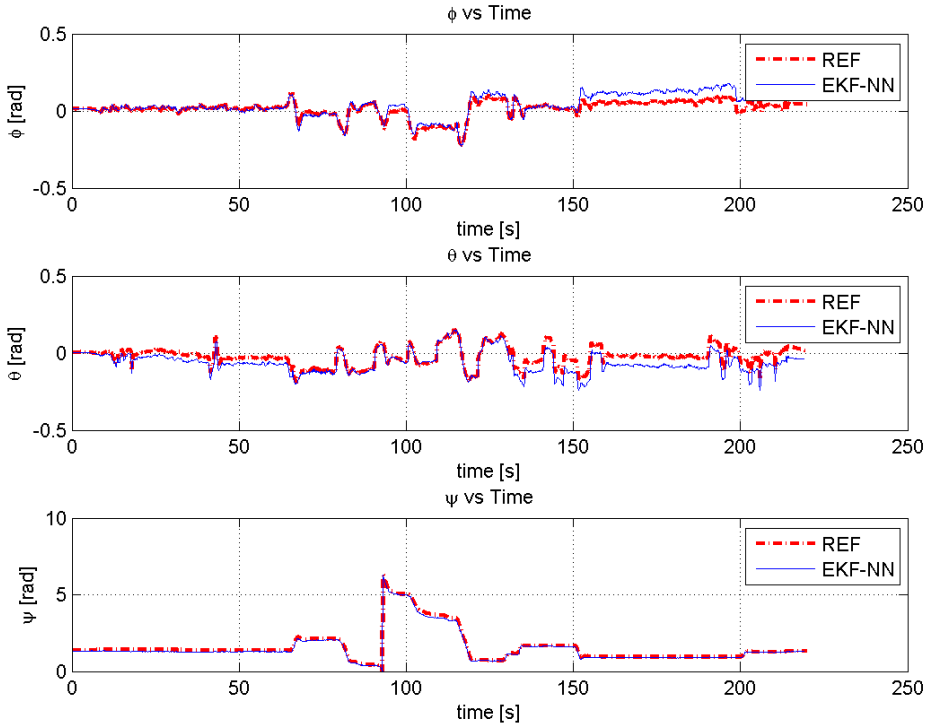



Figure 6.97: EKF attitude results for MLPNN between 80-120

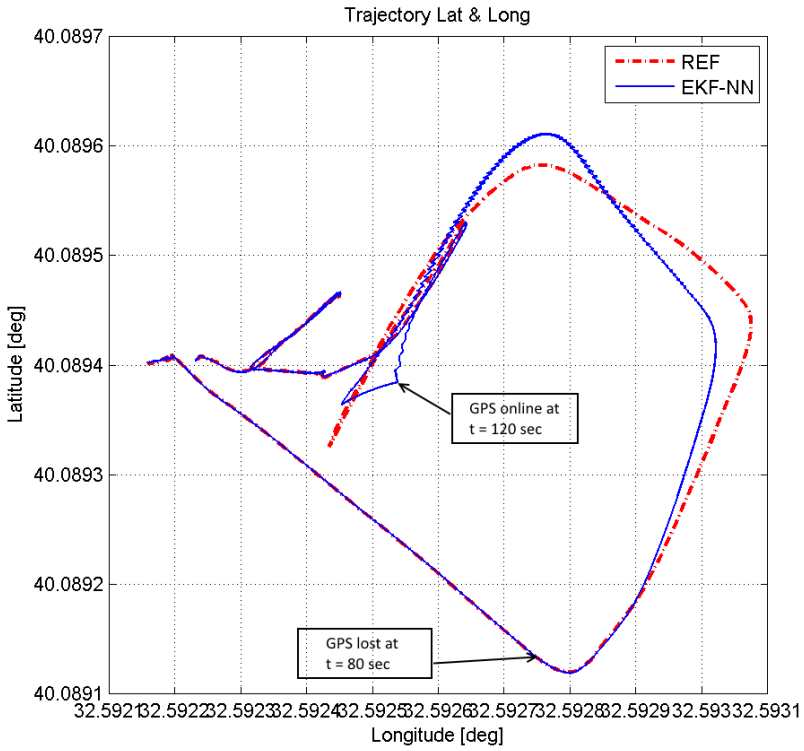



Figure 6.98: EKF latitude and longitude for MLPNN between 80-120

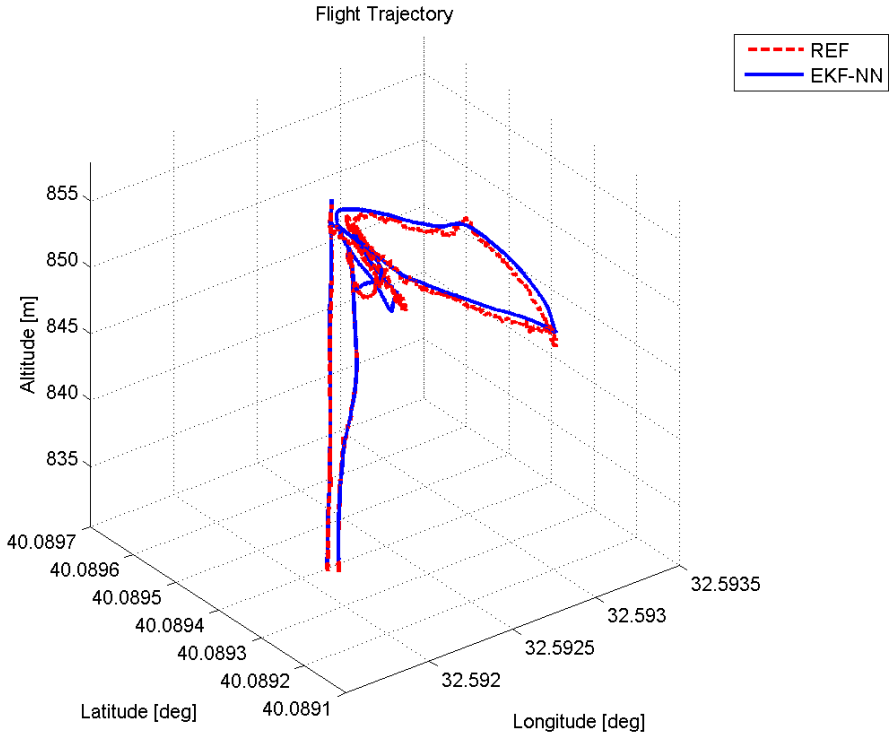



Figure 6.99: EKF 3D flight for MLPNN between 80-120

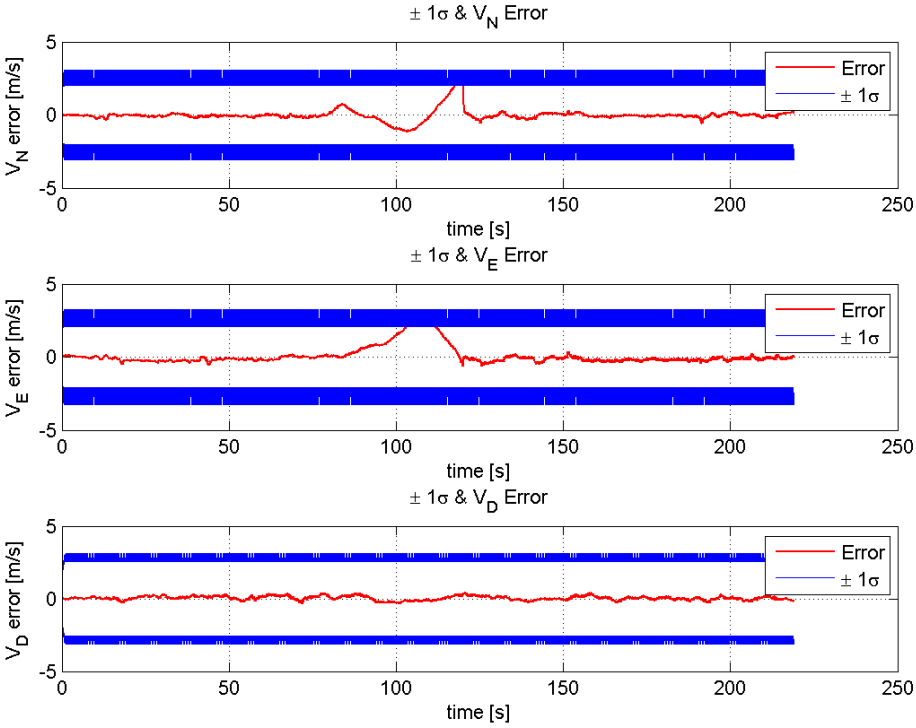



Figure 6.100: EKF velocity estimation error between $\pm 1\sigma$ bound for MLPNN between 80-120

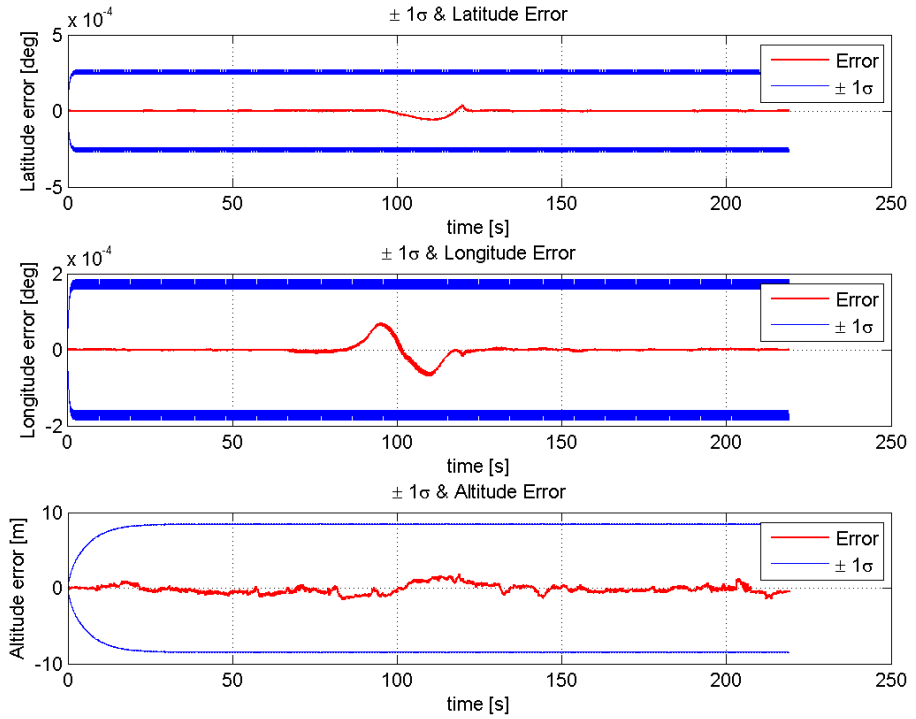



Figure 6.101: EKF position estimation error between $\pm 1\sigma$ bound for MLPNN between 80-120

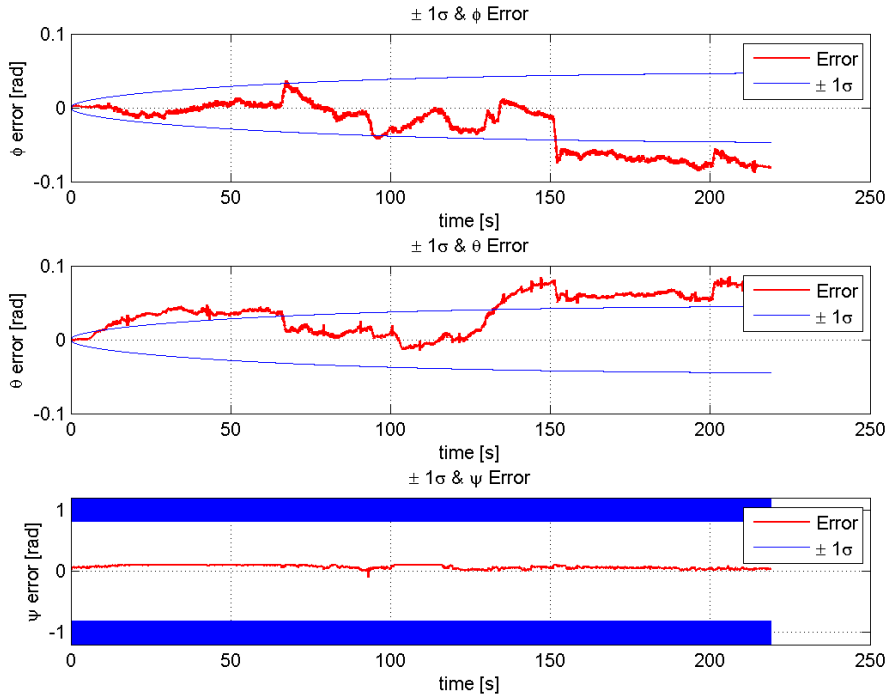



Figure 6.102: EKF attitude estimation error between $\pm 1\sigma$ bound for MLPNN between 80-120

**GPS outages between 180-200**

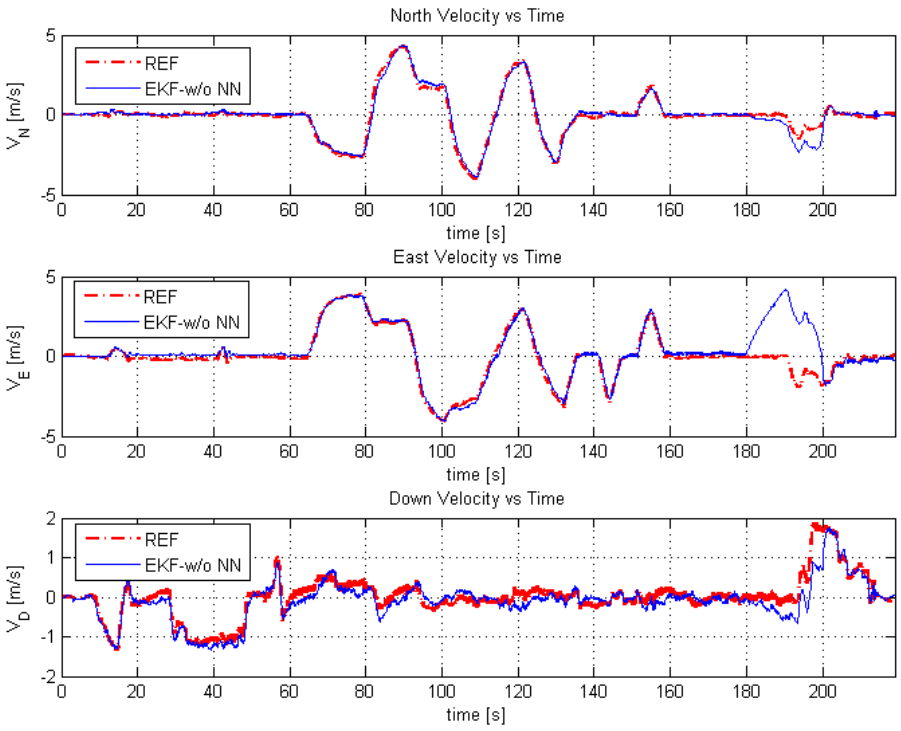



Figure 6.103: EKF velocity results for GPS outage between 180-200

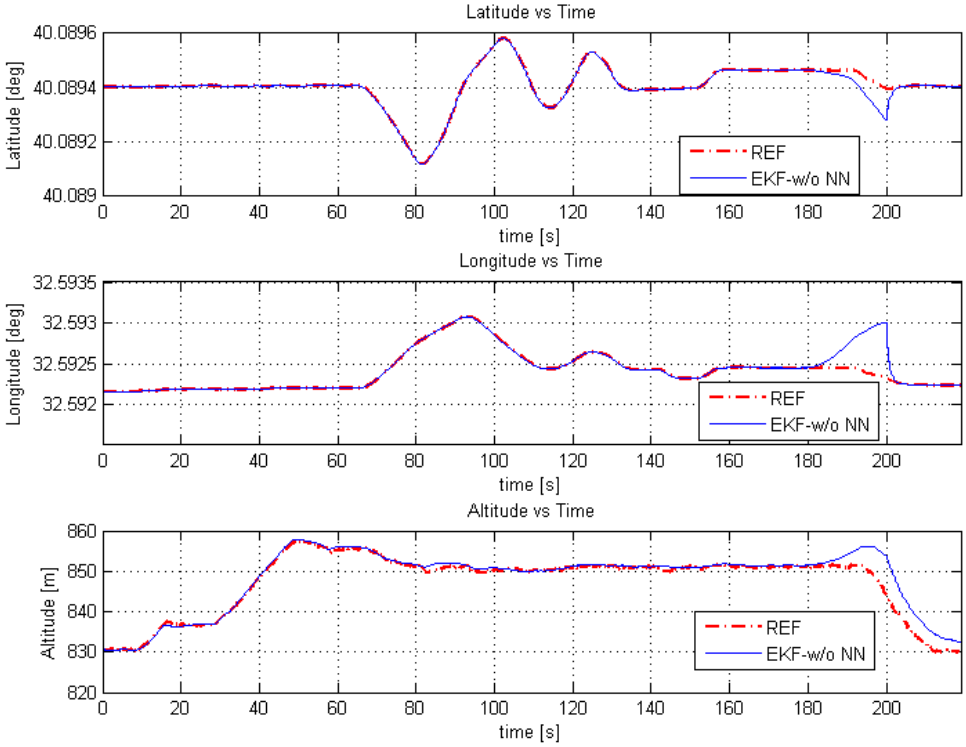



Figure 6.104: EKF position results for GPS outage between 180-200

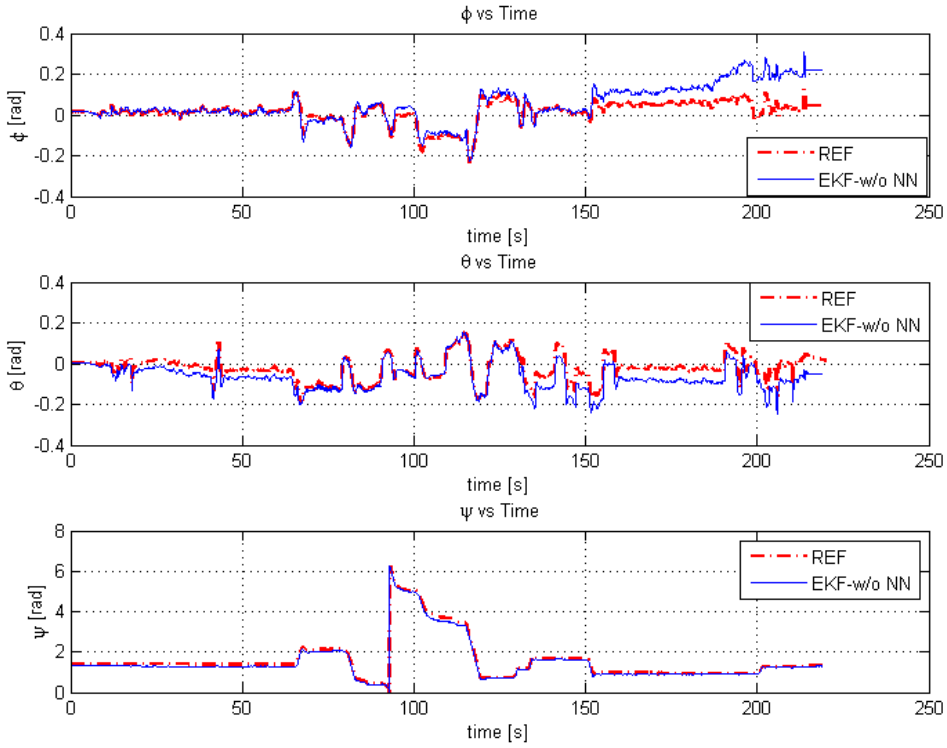



Figure 6.105: EKF attitude results for GPS outage between 180-200

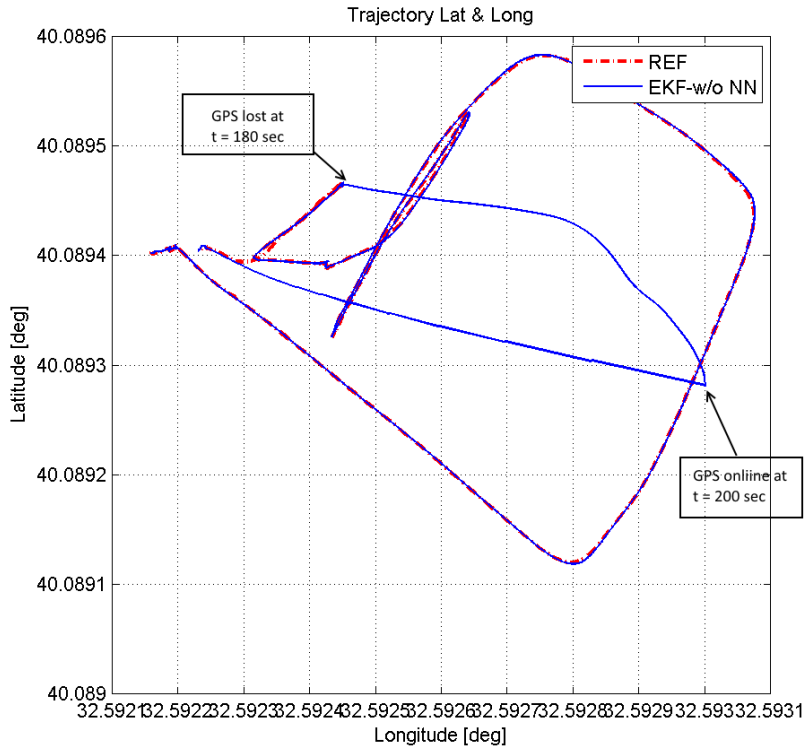



Figure 6.106: EKF latitude and longitude for GPS outage between 180-200

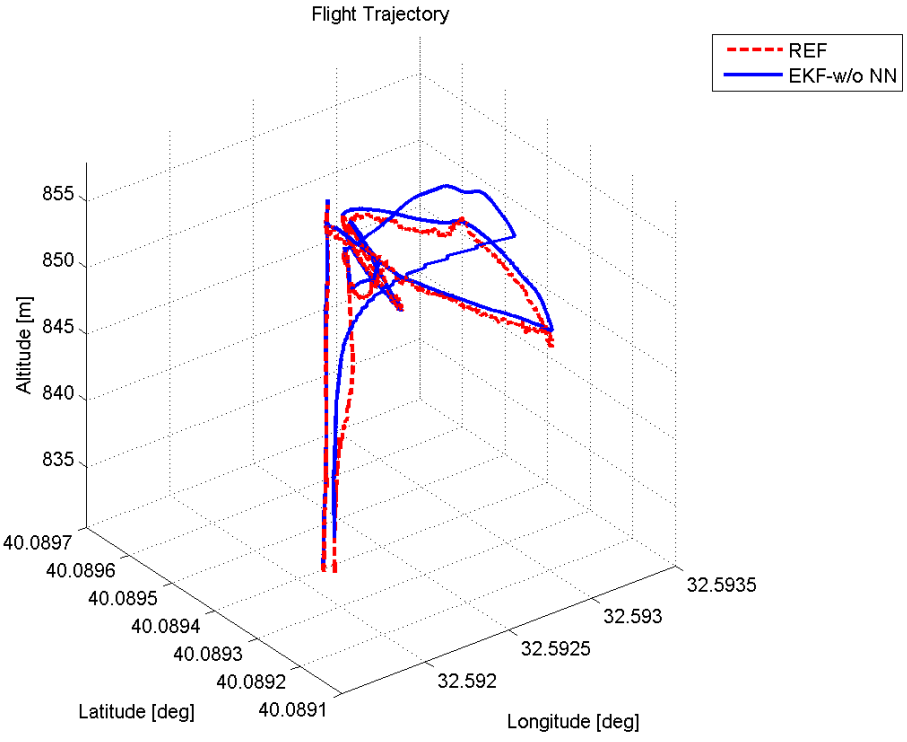



Figure 6.107: EKF 3D flight for GPS outage between 180-200

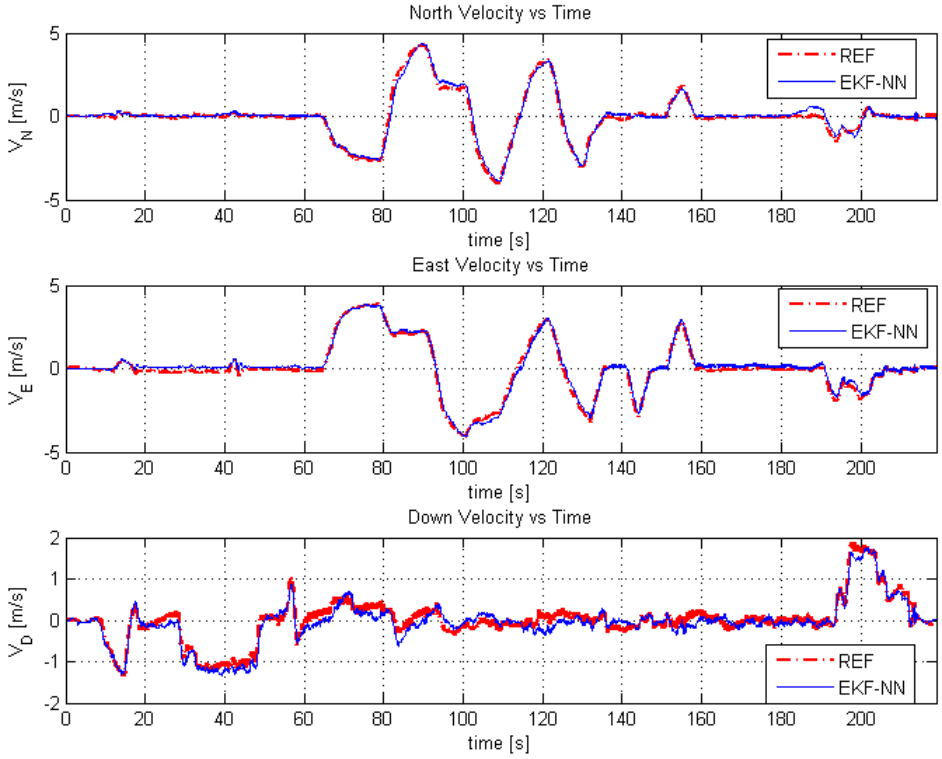



Figure 6.108: EKF velocity results for MLPNN between 180-200

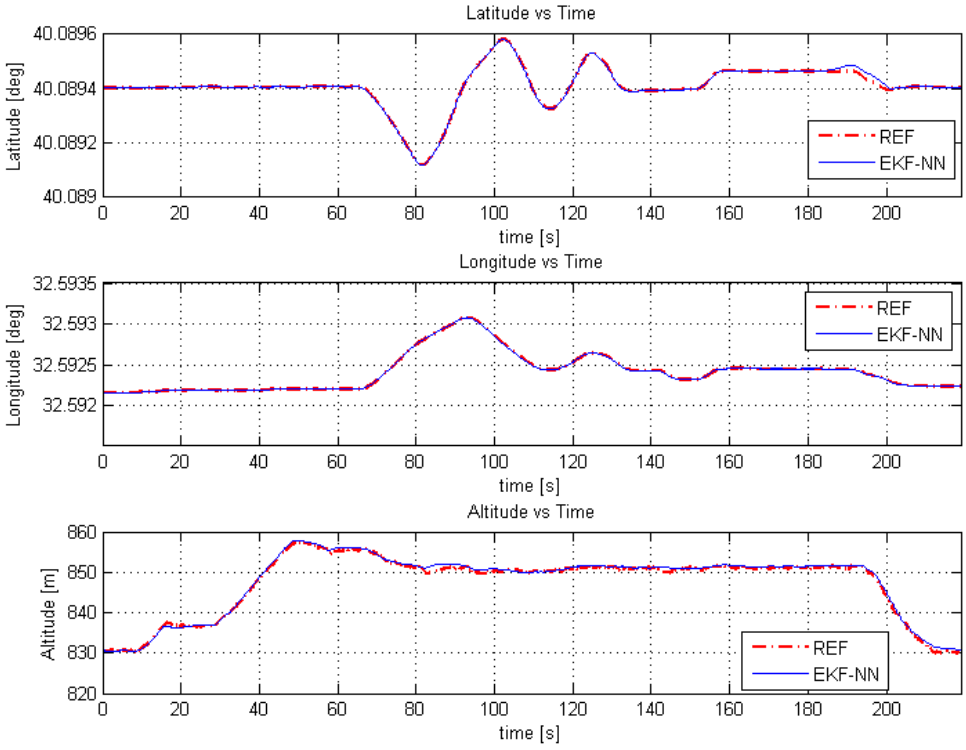



Figure 6.109: EKF position results for MLPNN between 180-200

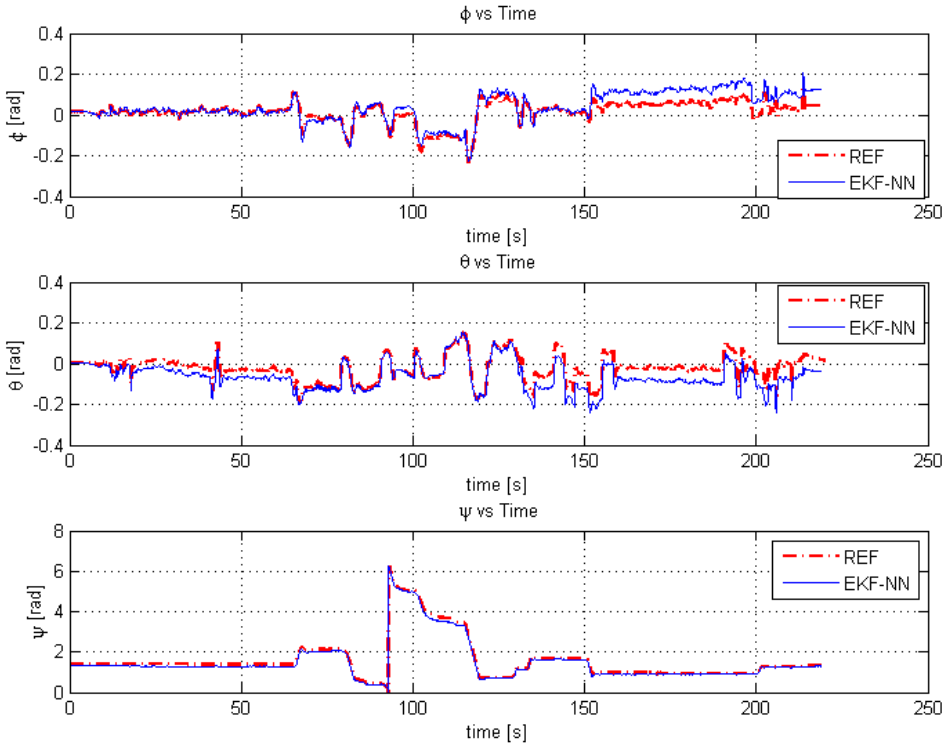



Figure 6.110: EKF attitude results for MLPNN between 180-200

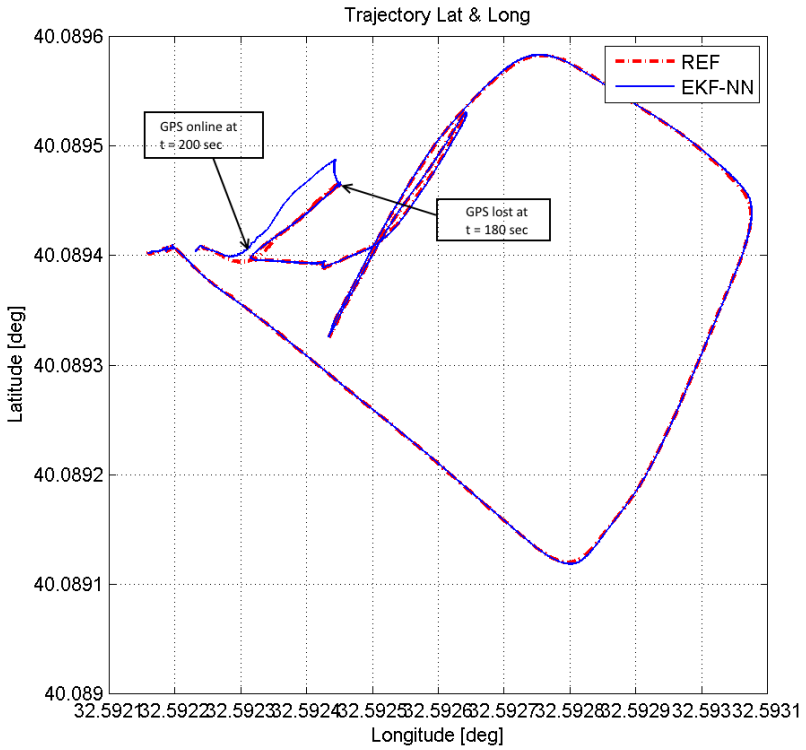



Figure 6.111: EKF latitude and longitude for MLPNN between 180-200

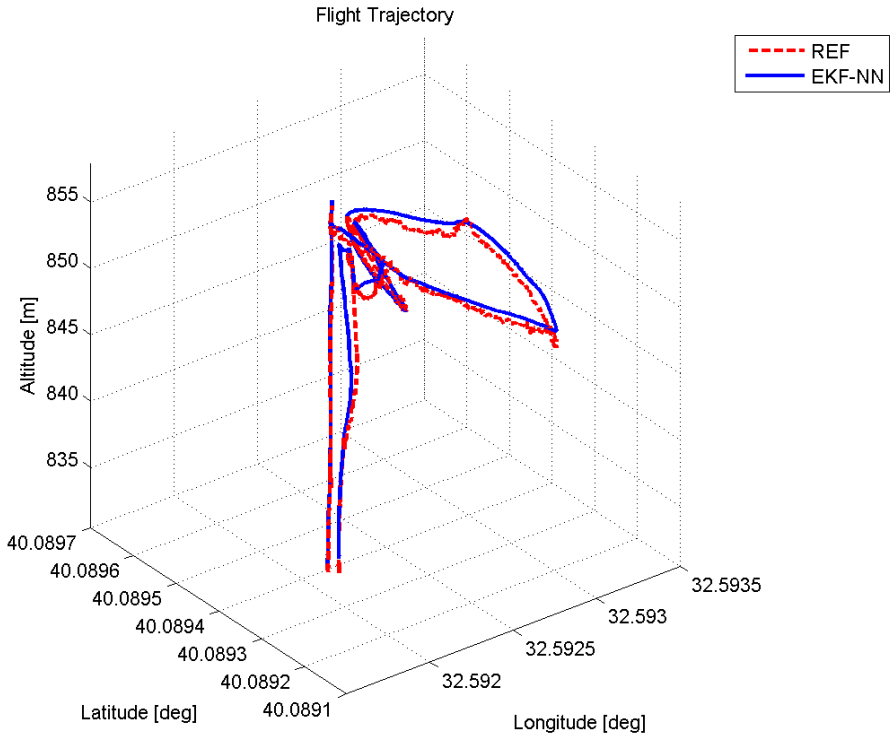



Figure 6.112: EKF 3D flight for MLPNN between 180-200

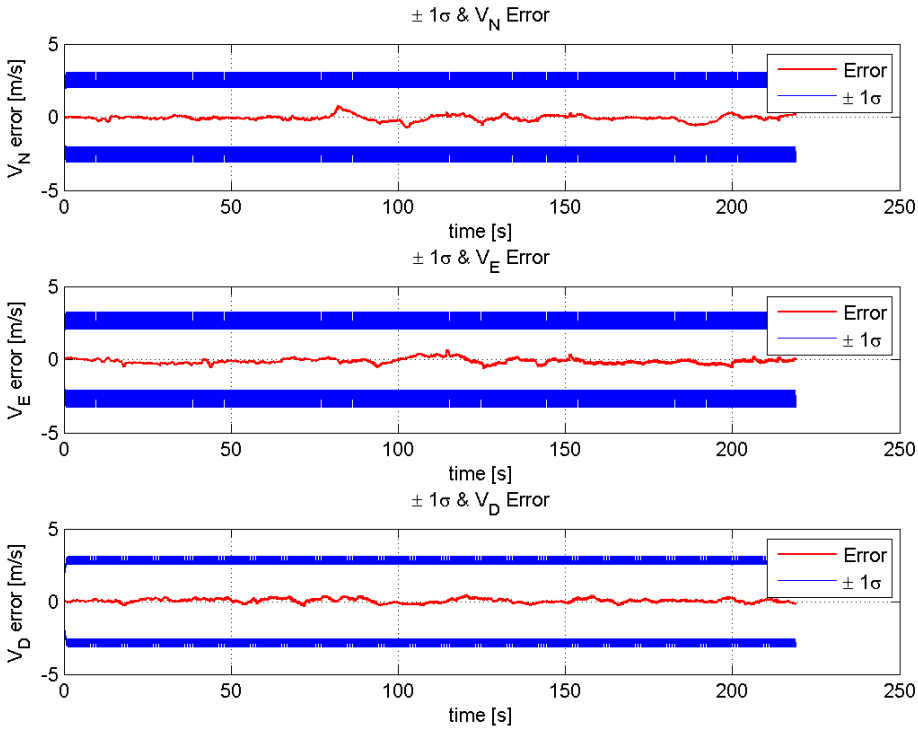



Figure 6.113: EKF velocity estimation error between $\pm 1\sigma$ bound for MLPNN between 180-200

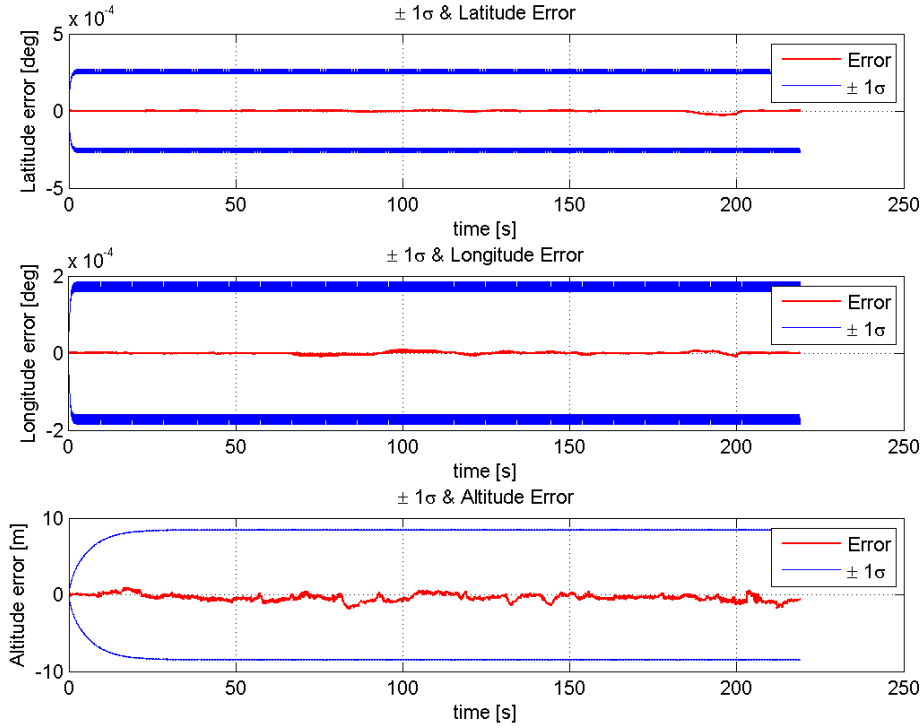



Figure 6.114: EKF position estimation error between $\pm 1\sigma$ bound for MLPNN between 180-200

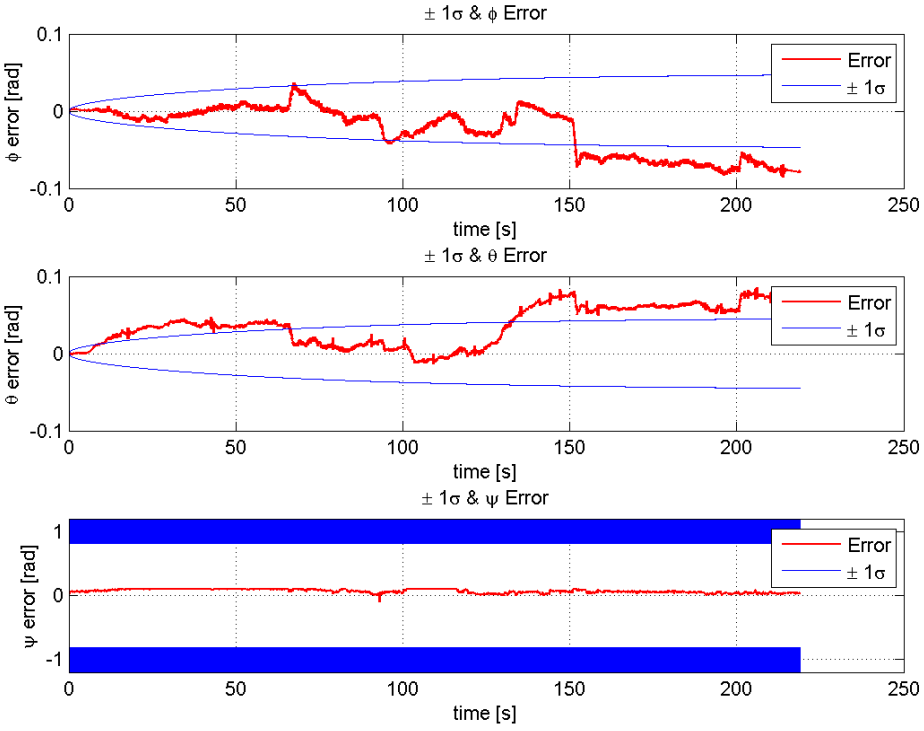



Figure 6.115: EKF attitude estimation error between $\pm 1\sigma$ bound for MLPNN between 180-200

According to NN test results, the root mean square errors (RMSE) for the each GPS outages are given in Tables 6.10, 6.12 and 6.14 for GPS outage section only on trajectory and in Tables 6.11, 6.13 and 6.15 for all trajectory.

Table 6.10: Case 1 RMSE results only for GPS outage between 55-85 s

| Positions | RMSE for NN | RMSE w/o NN | Percentage Improvement |
|---|---|---|---|
| Latitude [deg] | 1.1070e-05 | 1.1897e-04 | 90.6948% |
| Longitude [deg] | 4.9036e-05 | 7.2791e-05 | 32.6347% |
| Altitude [m] | 0.7815 | 14.0023 | 94.4190% |

Table 6.11: Case 1 RMSE results GPS outage between 55-85 s for all trajectory

| Positions | RMSE for NN | RMSE w/o NN | Percentage Improvement |
|---|---|---|---|
| Latitude [deg] | 1.2156e-05 | 4.7134e-05 | 74.2089% |
| Longitude [deg] | 2.3015e-05 | 2.8645e-05 | 19.6531% |
| Altitude [m] | 0.9441 | 7.1628 | 86.8201% |

Table 6.12: Case 2 RMSE results only for GPS outage between 80-120 s

| Positions | RMSE for NN | RMSE w/o NN | Percentage Improvement |
|---|---|---|---|
| Latitude [deg] | 2.9341e-05 | 1.5167e-04 | 80.6552% |
| Longitude [deg] | 3.7891e-05 | 9.0199e-05 | 57.9919% |
| Altitude [m] | 0.9782 | 17.6107 | 94.4454% |

Table 6.13: Case 2 RMSE results GPS outage between 80-120 s for all trajectory

| Positions | RMSE for NN | RMSE w/o NN | Percentage Improvement |
|---|---|---|---|
| Latitude [deg] | 1.9749e-05 | 6.7384e-05 | 70.6919% |
| Longitude [deg] | 2.4430e-05 | 3.6022e-05 | 32.1794% |
| Altitude [m] | 0.9671 | 9.7688 | 90.1001% |

Table 6.14: Case 3 RMSE results only for GPS outage between 180-200 s

| Positions | RMSE for NN | RMSE w/o NN | Percentage Improvement |
|---|---|---|---|
| Latitude [deg] | 1.8193e-05 | 5.1139e-05 | 64.4254% |
| Longitude [deg] | 3.9581e-06 | 3.7987e-04 | 98.9580% |
| Altitude [m] | 0.5269 | 4.4415 | 88.1380% |

Table 6.15: Case 3 RMSE results GPS outage between 180-200 s for all trajectory

| Positions | RMSE for NN | RMSE w/o NN | Percentage Improvement |
|---|---|---|---|
| Latitude [deg] | 1.2839e-05 | 1.6969e-05 | 24.3353% |
| Longitude [deg] | 1.6756e-05 | 1.2599e-04 | 86.7004% |
| Altitude [m] | 1.0925 | 2.8206 | 61.2683% |

According to RMSE results, latitude, longitude and altitude deviation in case of GPS outages are decreased significantly by using MLPNN. As it can be seen from the results, generally altitude improvement is better than latitude and longitude improvement. This result basically depends on random initial weight assignment, learning rate parameter, number of neurons and the characteristic of altitude data to be learned from MLPNN. The differences between percentage achievements are caused by the past experiences and learned data by MLPNN. The prediction performance of MLPNN directly related to past data that is given to MLPNN to determine weight values in back propagation algorithm.

# CHAPTER 7

# CONCLUSIONS AND FUTURE WORKS

In this thesis, INS/GPS and magnetometer integration and estimation technique have been studied. For the estimation procedure Extended Kalman Filter (EKF) has been used. To improve the estimation of EKF during GPS outage case *Artificial Neural Network* has been utilized. By using Multilayer Perceptron Neural Network (MLPNN) model the estimation performance of velocity and position states in EKF have been improved. The test case for the simulink model of EKF and MLPNN are based on two procedure. Firstly designed EKF algorithm is verified and validated by using TAI helicopter simulator. Secondly both EKF and MLPNN performance are proven by using road test and quadcopter field test data.

In first, the inertial navigation algorithm and sensor model are presented. The detailed explanation of navigation mechanization algorithm and its applications to navigation systems are given. Also modelling of sensor systems which include accelerometer, gyroscope, GPS and magnetometer are mentioned. The modelling procedure of these sensors are discussed in detail.

After given sensor models and navigation equations, the estimation techniques and neural network design are discussed. For the estimation technique Kalman filter algorithm is presented. The dynamic error model of navigation equations is obtained and the integration types of Kalman filter are given for INS/GPS and magnetometer sensor fusion system. After that an introduction to neural network modelling is given. In the scope of neural network the learning algorithms, network modelling and optimization with Least Mean Square techniques are examined. Multilayer perceptron concept in neural network modelling is discussed and the application of

MLPNN into INS/GPS navigation system is given in detail.

Lastly, the designed simulink model which includes INS/GPS integration with EKF and MLPNN is tested both in simulator and field test data. For the simulator environment TAI helicopter simulator is used. In field test land vehicle and quadcopter are utilized separately to collect sensor data. Also the estimation performance of EKF and the performance of MLPNN are observed during GPS outages.

At the end of road test and quadcopter test it is concluded that MLPNN is increased the navigation performance over EKF during GPS outage times. When GPS is online the covariance matrix are stable between sigma bound results and EKF gives bounded results in state estimation. However, when GPS aiding is not obtained the state estimation errors between sigma bounds are increasing.

During GPS outage periods, it is observed that the standalone navigation is not sufficient to eliminate propagated errors. Especially during large GPS outage periods, the position and velocity precisions are lost. Because of that the first order linearized system is not sufficient to prevent error accumulation navigation accuracy and performance are reduced.

One of the most important observation is the process, system and measurement noise matrix selection in EKF design procedure. To reach steady state solution quickly the process noise matrix is selected as zero matrix. As for the system and measurement noise matrix selection, auto-covariance least square technique is used. Initial diagonal noise matrix elements are selected by using ALS method. However some of the state estimation performance is not obtained in a desired way. Therefore by tuning the diagonal elements of system and measurement matrix desired performance form EKF estimation is achieved.

In design of artificial neural network, it is experienced that the performance of MLPNN is directly affected by learning rate parameter, initial weight selection and the learning characteristic of desired data to be learned. By selecting different learning rate parameter, the convergence performance of back propagation algorithm is observed. To small and large learning rates are avoided. If small learning rate

selected, learning process will be slow. If large learning rate is selected learning process is failed. Also initial weights are selected in a same way because in calculation of sigmoidal function saturated region and origin should not be selected as starting point on learning surface. Therefore by considering proper learning rate and weight selection, with well-designed MLPNN, the learning performance increases greatly.

Another observation during GPS absence, the velocity and position predictions are achieved between 20-50 seconds GPS outages. For different scenarios and different path sections the performance of MLPNN is observed. At the end the MLPNN shows great improvement over standalone EKF during GPS absence. RMSE results of velocity and position solutions also support this observation.

Finally, in future works adaptive methods can be added to model to estimate the initial weights for MLPNN and Kalman Filter system and measurement covariance matrices. For example, Bayesian, Maximum likelihood, correlation and covariance matching methods or adaptive Kalman filter can be used. Also by adding different neural network structure such as RBFNN, recurrent neural network, deep neural network, etc. into model the performance results can be observed and compared during GPS outage.

# REFERENCES


[1] David H. Titterton and John L. Weston. *Strapdown Inertial Technology-2nd Edition*. The Institution of Electrical Engineers, VA, USA, 2004.

[2] Aboelmagd Noureldin, Tashfeen B. Karamat and Jacques Georgy. *Fundamentals of Inertial Navigation, Satellite-based Positioning and their Integration*. Springer, Calgary, Canada, 2013.

[3] Mehmet Erçin Özgeneci. Mems sensor based underwater AHRS (attitude and heading reference system) aided by compass and pressure sensor. Master's thesis, METU, September,2012.

[4] Paul G. Savage. *Strapdown Analytics*. Artech House, Inc., Maple Plain, Minnesota, 2007.

[5] Johan Bijker. Development of an attitude heading reference system for an airship. Master's thesis, University of Stellenbosch, December,2006.

[6] Walid Abdel-Hamid. *Accuracy Enhancement of Integrated MEMS-IMU/GPS Systems for Land Vehicular Navigation Applications*. PhD thesis, University of Calgary, January,2005.

[7] Paul G. Savage. Strapdown inertial navigation integration algorithm design part 1: Attitude algorithms. *Journal of Guidance,Control and Dynamics*, 21, January-February 1998.

[8] Serdar Kırımoğlu. Multisensor dead reckoning navigation on a tracked vehicle using Kalman filter. Master's thesis, METU, September,2012.

[9] Paul G. Savage. Strapdown inertial navigation integration algorithm design part 2: Velocity and position algorithms. *Journal of Guidance,Control and Dynamics*, 21, March-April 1998.

[10] Robert M. Rogers. *Applied Matematics in Integrated Navigation Systems, Second Edition*. AIAA Education Series, Gainesville, Florida, 2003.

[11] Alper Öztürk. Development, implementation, and testing of a tightly coupled integrated INS/GPS system. Master's thesis, METU, August,2003.

[12] Elliott D. Kaplan, Christopher J. Hegarty. *Understanding GPS Principles and Applications*. Strapdown Associates Inc., Norwood, MA, 2006.

[13] Tamer Akça. An addaptive unscented Kalman filter for tightly-coupled INS/GPS integration. Master's thesis, METU, February,2012.

[14] Umar Iqbal. *Multi-Sensor Data fusion for Vehicular Navigation Applications*. PhD thesis, Queen's University, July,2012.

[15] Global Positioning System Directorate System Engineering and Integration. Navstar GPS space segment/navigation user interfaces. *IS-GPS-200*, September 2013.

[16] Robert G. Brown , Patrick Y.C. Hwang. *Introduction to Random Signals and Applied Kalman Filtering with Matlab Exercises and Solutions, 3rd Edition*. John Wiley and Sons, Inc., Canada, 1997.

[17] Hang Guo et al. Kalman filtering for GPS/magnetometer integrated navigation system. *Advances in Space Research*, 45:1350–1357, 2010.

[18] Valérie Renaudin, Muhammad Haris Afzal, Gérard Lachapelle. New method for magnetometers based orientation estimation. *IEEE/ION Plans*, May 2010.

[19] Brian J. Odelson, Alexander Lutz, James B. Rawlings. The autocovariance least-squares method for estimating covariances: Application to model-based control of chemical reactors. *IEEE Transactions on control system technology*, 14, May 2006.

[20] Ming Ge, Eric C. Kerrigan. Noise covariance estimation for time-varying and nonlinear systems. *The 19th World Congress the International Federation of Automatic Control*, August 2014.

[21] Sameh Nassar. *Improving the Inertial Navigation System (INS) Error Model for INS and INS/DGPS Applications*. PhD thesis, University of Calgary, November,2003.

[22] Simon Haykin. *Neural Networks - A Comprehensive Foundation, 2nd Edition*. Prentice Hall Internationa, Inc., Hamilton,ontario,Canada, 1999.

[23] Tao Zhang, Xiaosu Xu. A new method of seamless land navigation for GPS/INS integrated system. *Journal of the International Measurement Confederation*, 45:691–701, 2012.

[24] Burak H. Kaygısız, Aydan M. Erkmen, İsmet Erkmen. GPS/INS enhancement using neural networks for autonomous ground vehicle applications. volume 55, Las Vegas,Nevada, 2003.

[25] Aboelmagd Noureldin , Ahmed Osman, Naser El-Sheimy. Online INS/GPS integration with a radial basis function neural network. *IEEE Systems Magazine*, March 2005.

[26] Naser El-Sheimy, Kai-Wei Chiang and Aboelmagd Noureldin. The utilization of artificial neural networks for multisensor system integration in navigation and positioning instruments. *IEEE Transactions and Instruentation and Measurement*, 55, 2006.

[27] Lorinda Semeniuk, Aboelmagd Noureldin. Bridging GPS outages using neural network estimates of ins position and velocity errors. *Measurement Science and Technology*, 17:2783–2798, 2006.

[28] KaiWei Chiang, Aboelmagd Noureldin, Naser El-Sheimy. Constructive neural-networks-based MEMS/GPS integration scheme. *IEEE Transactions on Aerospace and Electronic Systems*, 44, April 2008.

[29] Lepinsy Chanthalansy. Neural network augmented tightly coupled Kalman filter for low-cost reduced inertial navigation sensor and GPS integration. Master's thesis, Royal Military College of Canada, May,2010.

[30] A. Noureldin, Ahmed El-Shafie, Mohamed Bayoumi. GPS/INS integration utilizing dynamic neural networks for vehicular navigation. *An International Journal on Multi-Sensor, Multi-Source Information Fusion*, 12:48–57, 2011.

[31] Malleswaran M. et al. Performance comparison of HONNs and FFNNs in GPS and INS integration for vehicular navigation. MIT,Anna University,Chennai, June 2011.

[32] Xiyuan Chen et al. Novel hybrid of strong tracking Kalman filter and wavelet neural network for GPS/INS during GPS outages. *Journal of the International Measurement Confederation*, 46:3847–3854, 2013.

# APPENDIX A

# AUTOCOVARIANCE LEAST-SQUARES METHOD

In chapter 4in equation 4.37 the auto-covariance least square method general formula is given. The detailed explanation of symbols and terms is given below. For more information the references [19] and [20] can be used.

The symbols which are used in ALS method are listed below.

- The Kronecker product is symbolized with $\otimes$.
- The symbol $\oplus$ symbolizes matrix direct sum.

$$\bigoplus_{j=1}^{M} H_j = \begin{bmatrix} H1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & H_M \end{bmatrix} \tag{A.1}$$

- $(.)_s$ symbol stands for vectorization operation. If $a_i$ is the ith column of matrix $A$,

$$vec(A) = A_s = \begin{bmatrix} a_1^T & \cdots & a_n^T \end{bmatrix} \tag{A.2}$$

- $\mathcal{I}_{N,p}$ with dimension $(pN)^2 \times p^2$ denotes the permutation matrix.
- $\mathcal{D}_n$ with dimension $n^2 \times \frac{n(n+1)}{2}$ denotes duplication matrix.
- $\mathcal{M}_l^{m,n}$ is an auxiliary matrix,

$$\mathcal{M}_l^{m,n} = \begin{bmatrix} 0_{m\times(l-1)} & I_m & 0_{m\times(n-m-l+1)} \end{bmatrix} \tag{A.3}$$

The calculation of parameters $\Gamma_i,\ \Omega_i,\ \Phi_i,\ \Psi_i$ in equation 4.37 are given below.

$$\tilde{F}_s = \left( \bigoplus_{k=k_0}^{k_0+M-1} C_k \right) \left( I_{nM} - \begin{bmatrix} 0 & 0 \\ \bigoplus_{k=k_0}^{k_0+M-2} & 0 \end{bmatrix} \right)^{-1} \begin{bmatrix} \bar{A}_{k_0-1} \\ 0 \end{bmatrix} \tag{A.4}$$

$$\tilde{H} = \mathcal{M}_1^{p,pN} \tag{A.5}$$

$$\tilde{B} = \left( \bigoplus_{k=k_0}^{k_0+M-1} C_k \right) \left( I_{nM} - \begin{bmatrix} 0 & 0 \\ \bigoplus_{k=k_0}^{k_0+M-2} & 0 \end{bmatrix} \right)^{-1} \bigoplus_{k=k_0-1}^{k_0+M-2} G_k \tag{A.6}$$

$$\tilde{D} = - \left( \bigoplus_{k=k_0}^{k_0+M-1} C_k \right) \left( I_{nM} - \begin{bmatrix} 0 & 0 \\ \bigoplus_{k=k_0}^{k_0+M-2} & 0 \end{bmatrix} \right)^{-1} \bigoplus_{k=k_0-1}^{k_0+M-2} A_k L_k \tag{A.7}$$

$$\tilde{S}_i = \mathcal{M}_{pi+1}^{pN,pM} \tag{A.8}$$

$$\tilde{J}_i = (\mathcal{M}_1^{r(i+1),rM})^T \tag{A.9}$$

$$\tilde{U}_i = (\mathcal{M}_1^{p(i+1),pM})^T \tag{A.10}$$

$$\tilde{O}_i = (\mathcal{M}_{p+1}^{p,pM})^T \tag{A.11}$$

$$\tilde{P}_i = (\mathcal{M}_{p(i+1)+1}^{p(N-1),pM})^T \tag{A.12}$$

$$\Gamma_i = \tilde{S}_i \tilde{F}_s \quad \bar{\Gamma}_i = \tilde{H} \Gamma_i \quad \Omega_i = \tilde{S}_i \tilde{B} \tilde{J}_i \quad \bar{\Omega}_i = \tilde{H} \Omega_i \quad \Phi_i = \tilde{S}_i \tilde{D} \tilde{U}_i \quad \bar{\Phi}_i = \tilde{H} \Phi_i$$

$$\Psi_i = \begin{bmatrix} I_p \\ \tilde{P}_i \tilde{D} \tilde{O}_i \end{bmatrix} \tag{A.13}$$

# APPENDIX B

# SENSORS DATASHEETS

## U-blox M8N GPS


Figure B.1: U-blox M8N GPS receiver and compass

**Ublox Neo-M8N GPS with Compass**
**Introduction:**
A new generation Ublox GPS NEO-M8N,with low power consumption and high precision, the ultimate accuracy is 0.6 meters, actually almost 0.9 meters, greater than the previous generation NEO-7N 1.4-1.6 meters accuracy,support GPS/QZSS L1 C/A, GLONASS L10F, BeiDou B1 protocol and mode or more.

**GPS Chip parameters:**
Receiver type 72-channel u-blox M8 engine
GPS/QZSS L1 C/A, GLONASS L10F, BeiDou B1
SBAS L1 C/A: WAAS, EGNOS, MSAS
Galileo-ready E1B/C (NEO-M8N)
Nav. update rate1 Single GNSS: up to 18 HZ
Concurrent GNSS: up to 10 Hz
Position accuracy2 2.0 m CEP
Acquisition2 Cold starts: 26 s
Aided starts: 2 s
Reacquisition: 1.5 s
Sensitivity2 Tracking & Nav: −167 dBm
Cold starts: −148 dBm
Hot starts: −156 dBm
Oscillator TCXO (NEO-M8N/Q),
Crystal (NEO-M8M)
RTC crystal Built-In
Anti jamming Active CW detection and removal. Extra
onboard SAW band pass filter (NEO-M8N/Q)
Supported antennas Active and passive
Data-logger For position, velocity, and time (NEO-M8N)

Operating temp. –40° C to 85° C
Storage temp. –40° C to 85° C (NEO-M8N/Q)
Supply voltage 1.65 V to 3.6 V (NEO-M8M)
Power consumption4 23 mA @ 3.0 V (continuous)
(1 Hz, GPS mode only)
Backup Supply 1.4 to 3.6 V
Timepulse Configurable 0.25 Hz to 10 MHz
**Model Features:**
- Quick satellite searching, only need 10s to find 6 satellite in open space
- Built-in compass, refresh rate up to 10GHz
- Support GPS+BD+SBAS, or GPS+GLONASS+SBAS

## Pixhawk Autopilot

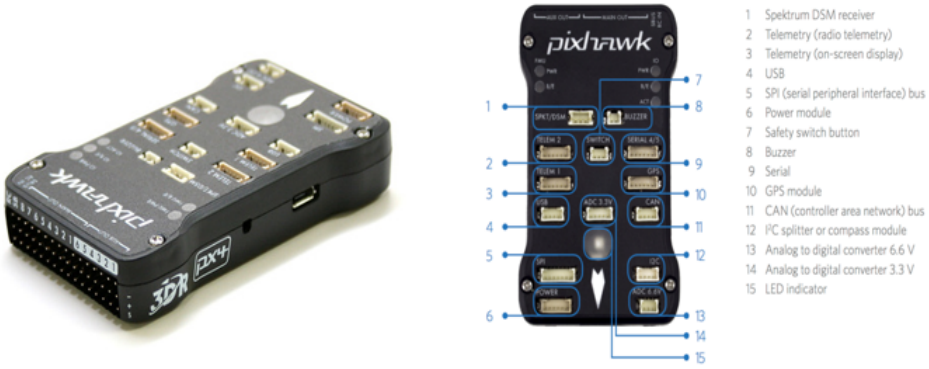



Figure B.2: Pixhawk autopilot and sensor module

**Specifications**
**Processor**

- 32bit STM32F427 Cortex M4 core with FPU
- 168 MHz
- 256 KB RAM
- 2 MB Flash
- 32 bit STM32F103 failsafe co-processor

**Sensors**

- ST Micro L3GD20H 16 bit gyroscope
- ST Micro LSM303D 14 bit accelerometer / magnetometer
- Invensense MPU 6000 3-axis accelerometer/gyroscope
- MEAS MS5611 barometer

**Interfaces**

- 5x UART (serial ports), one high-power capable, 2x with HW flow control
- 2x CAN (one with internal 3.3V transceiver, one on expansion connector)
- Spektrum DSM / DSM2 / DSM-X® Satellite compatible input
- Futaba S.BUS® compatible input and output
- PPM sum signal input
- RSSI (PWM or voltage) input
- I2C
- SPI
- 3.3 and 6.6V ADC inputs
- Internal microUSB port and external microUSB port extension

**Power System and Protection**

- Ideal diode controller with automatic failover
- Servo rail high-power (max. 10V) and high-current (10A+) ready
- All peripheral outputs over-current protected, all inputs ESD protected

# APPENDIX C

## SIMULINK MODELS

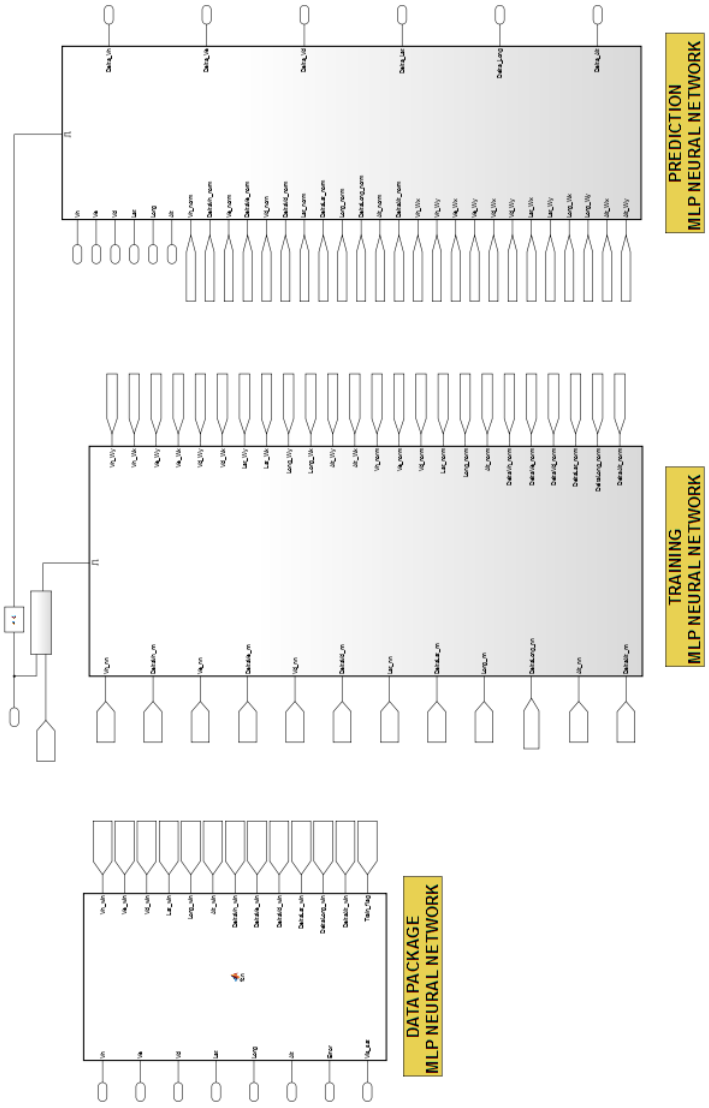



Figure C.1: Artificial neural network (ANN) training and prediction parts

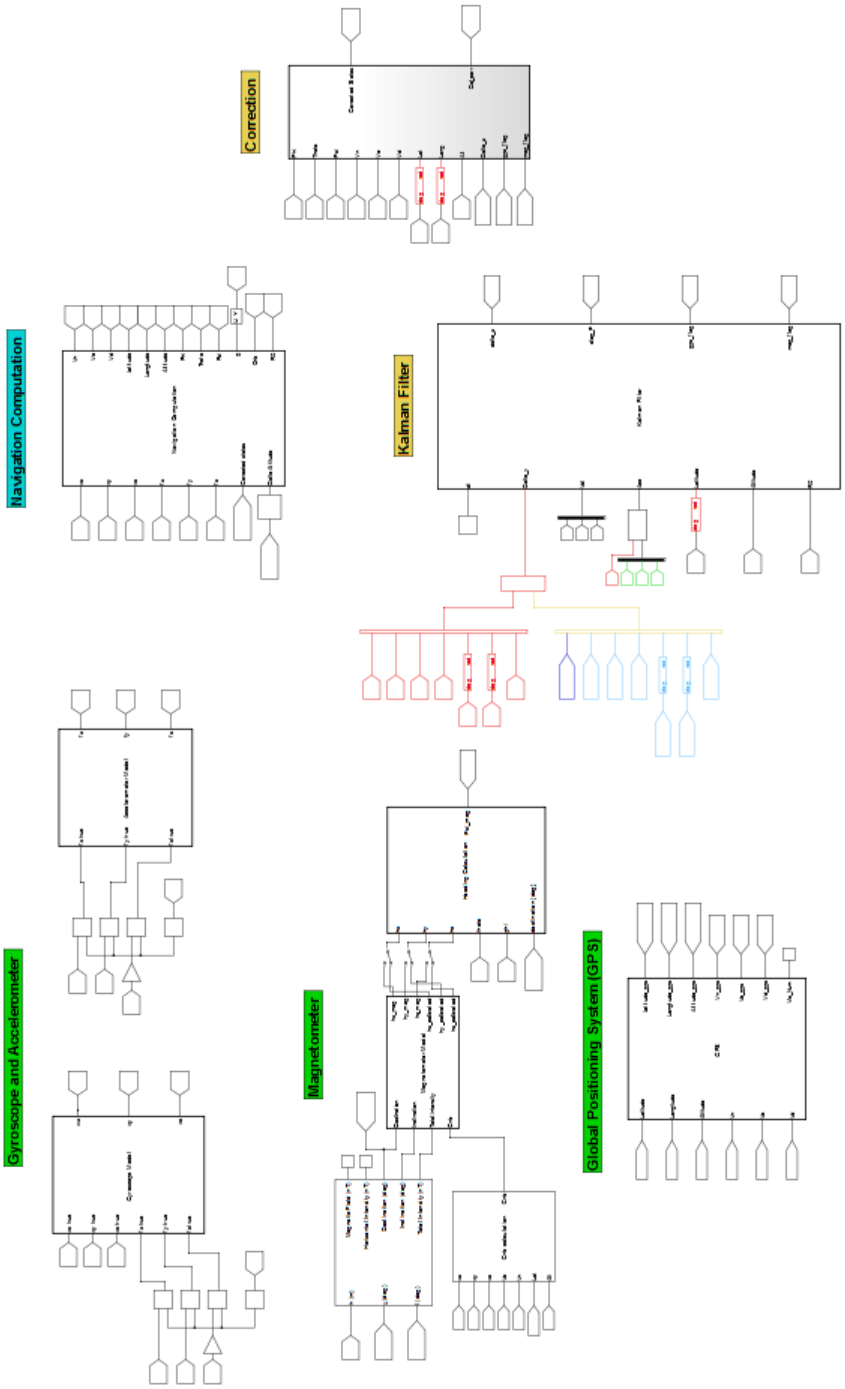



Figure C.2: Simulink model for navigation computation with sensor models and Kalman filter

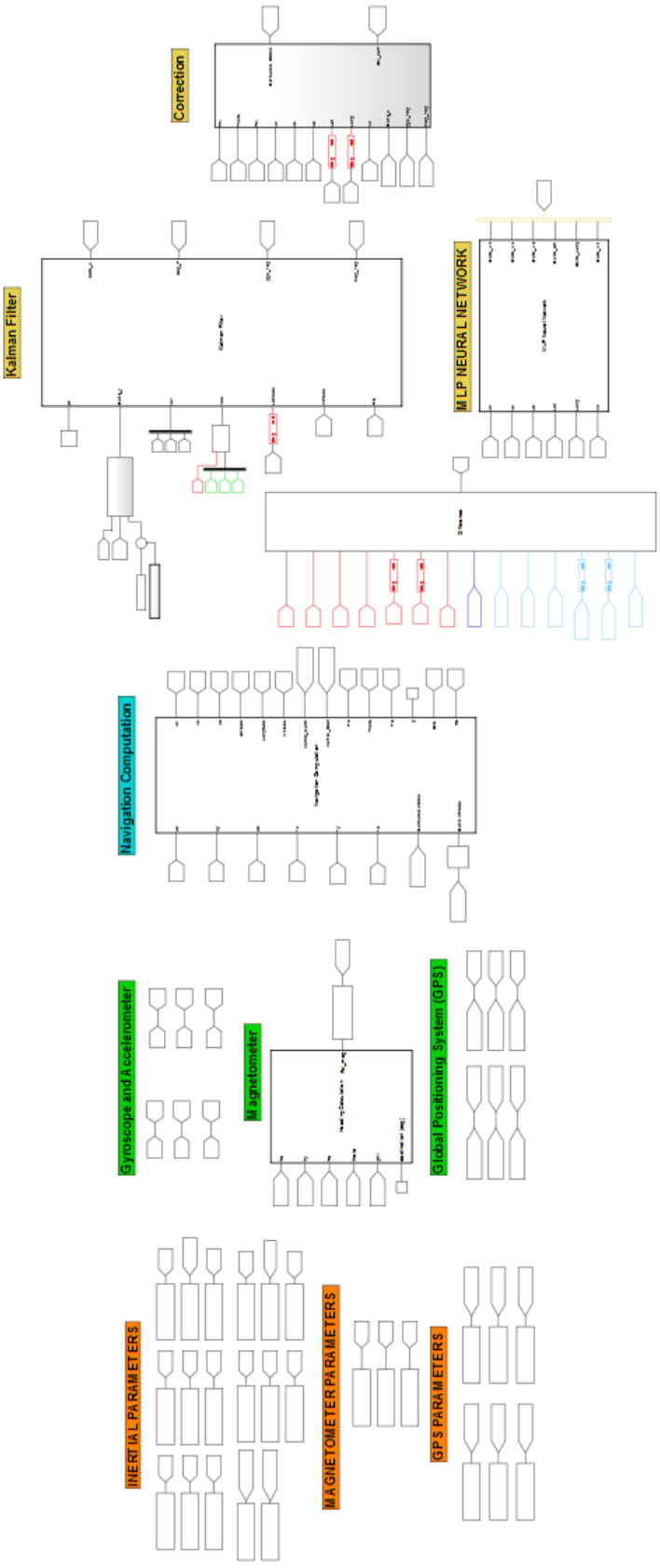



Figure C.3: Simulink navigation model with artificial neural network (ANN) structure