Czech Technical University in Prague

Faculty of Electrical Engineering

Department of control Engineering

Diploma Thesis

Performance comparison of Extended and Unscented Kalman Filter implementation in INS-GPS integration

Erasmus Mundus Programme

SpaceMaster

L

Prague,2009

Author: Joshy Madathiparambil Jose

Declaration

I, Joshy M. Jose, honestly declare that I have worked out my diploma thesis individually and all resources (literature,projects,SW etc) that I used are stated in the attached list.

InPrague, date _____ _____

Signature

Acknowledgements

I would like to thank my supervisors Martin Hromcik at CTU and Martin Orejas, Honeywell Brno for their advice and timely help for completion of this thesis. I would like also to thank my LTU supervisor Andreas Johanson for his helps and comments.

Abstract

The objective of this thesis is to implement an unscented kalman filter for integrating INS with GPS and to analyze and compare the results with the extended kalman filter approach. In a loosely coupled integrated INS/GPS system, inertial measurements from an IMU (angular velocities and accelerations in body frame) are integrated by the INS to obtain a complete navigation solution and the GPS measurements are used to correct for the errors and avoid the inherent drift of the pure INS system. The standard approach is to use an extended kalman filter in complementary form to model the errors of the INS states and use the GPS measurements to estimate corrections for these errors which are then feedback to the INS. Although the unscented kalman filter is more computational intensive, it is supposed to outperform the extended kalman filter and be more robust to initial errors. The main goal of this work is to analyze the difference in performance and robustness between both implementations. As a first step, a simplified Attitude estimation of a stabilized platform is implemented in both, the UKF and the EKF and eventually the UKF will be implemented in a more complex realistic 3D navigation problem and compare against the current model used by Honeywell.

Contents

List of Figures

List of Tables

List of Symbols

φ	Latitude
λ	Longitude
$\mathbf{h}$	Height in navigation coordinates
$\mathbf{x}, \mathbf{X}$	State vector
$\underline{\mathbf{a}}$	Vector notation
$\times$	Cross product
ECEF	Earth Centered Earth Fixed
ECI	Earth Centered Inertial
KF	Kalman Filter
EKF	Extended Kalman Filter
UT	Unscented Transform
UKF	Unscented Kalman Filter
UAV	Unmanned Aerial Vehicle
INS	Inertial Navigation System
GPS	Global Positioning System
NED	North-East-Down
IMU	Inertial Measurement Unit
$\mathbf{f^b}$	Specific force
$\mathbf{\Omega}$	Skew symmetric matrix of corresponding rotation rate
$\mathbf{\Omega^n_{ie}}$	Skew symmetric matrix of earth rotation rate
$\omega^{\mathbf{k}}_{\mathbf{ij}}$	Rotation of j (body, nav-frame etc.) with respect to i-frame expressed in k-frame
$\omega^{\mathbf{b}}_{\mathbf{ib}}$	Rotation of body with respect to inertial frame expressed in body frame
$\mathbf{C}$	Rotation matrix
$\mathbf{C^b_r}$	Rotation matrix from r frame to b frame
$\mathbf{C^n_b}$	Rotation matrix from body frame to nav. frame
$\mathbf{G}$	Acceleration due to gravity
$\mathbf{M}$	Radius of curvature of meridian
$\mathbf{N}$	Radius of curvature of prime vertical
$\mathbf{S_g}$	Scale factor, gyroscope
$\mathbf{S_a}$	Scale factor, accelerometer
$\mathbf{b_g}$	Bias, gyroscope

$\mathbf{b_a}$	Bias, accelerometer
$\mathbf{G}$	Acceleration due to gravity

Chapter 1

Introduction

1.1 Introduction

Navigation was an important and interesting filed of study in all times. Today with the technologies like GPS, everyone is benefited. It has become an inexpensive and common tool in day today life. It's almost everywhere, in mobile phones and navigators in vehicles. There are different tools and techniques for navigation but Inertial Navigation System and GPS are the most important of them. Each of these has some limitations and advantages. Inertial navigation system is fairly accurate for a short period of time but it tend to drift from reality as time elapses. GPS gives good accuracy over long period, but it is also not immune from errors and some times there may be problem with availability of signal. One of the solutions in such situations is to combine the measurements from both of these instruments and integrate it using some sensor fusion algorithms. The most commonly used method in such a situation is the kalman filter. There are mainly two different strategies in this sensor fusion, loosely coupled and tightly coupled integration. In loosely coupled integration, there is no effort to correct the GPS signals from its errors. But in tightly coupled integration, usually there are two kalman filters involved, One to correct the errors of GPS using its error models and the other for INS-GPS integration. For nonlinear systems and especially for INS-GPS integration applications, extended kalman filter has proven and widely used for more that three decades. Common approach is to consider the errors of the process instead of the mechanization equations itself. In 1997, Julier S.J and Uhlmann J.K introduced a new extension, 'UKF' of kalman filter for nonlinear systems. This approach claims to be superior to EKF when the system nonlinearly is higher. Also it is believed that UKF is more immune to the initial errors. Due to its computational complexity, it was

not considered for applications like navigation. This work try to investigate the difference between two approaches in terms of computational complexity and Performance achieved especially with the initial errors. major tasks in this study are

- To investigate the performance of both EKF and UKF filter in state estimation and error rejection.
- To compare the computational complexity of both Algorithms.
- To compare the performance with initial errors of UKF and EKF.

1.2 Thesis Layout

The chapter 2 explains in detail the inertial navigation system, co-ordinate frames, inertial sensors and its error sources. Also INS mechanization is well explained. Chapter 3 briefly explains the Global Positioning System, its working and error sources. Chapter 4 explains different kalman filter algorithms KF, EKF and UKF. Chapter 5 describes the state model and measurement model proposed for UKF implementation of the INS-GPS integration. Chapter 6 throughly explains the implementation in matlab. An attitude estimation problem was finally studied to acquaintance with the filtering techniques and to understand the difference in EKF and UKF. It is also briefly explained. The trajectory generator used to generate the input data is also explained and the different softwares (mainly matlab toolkits) used in simulation is also highlighted. Chapter 7 summarizes the results and chapter 8 concludes the work. Next chapter 9 suggest some future works. Some of the very useful literatures which were very helpful in formulating understanding about the topics were pointed below. Strapdown Inertial Navigation Technology, 2nd Edition by David H. Titterton and John L. Weston[21], fairly covers the Strapdown Technology. Applied Mathematics in Integrated Navigation System by Robert M. Rogers [4] also very interesting literature which clearly explains mechanization equation and derivation of error equations used in ekf implementation. Another literature to mention is Aircraft Control and Simulation by Brian L. Stevens, Frank L. [3] (first chapter available from www.wiley.com) which contain beautiful explanation of geodesy, earth's gravitation, terrestrial navigation and kinematics and dynamics of aircraft motion. PhD thesis by Eun Hwan Shin [23] tried to implement the UKF for INS-GPS integration which would be very useful in the further development of this work.

Chapter 2

Strapdown Inertial Navigation System

An inertial navigation system is based on classical mechanics to provide the direction of acceleration of the body concerned and also rotational motion of the body with respect to the inertial frame. The traditional inertial navigation system consists of accelerometers which give the information of the direction in which body is accelerating and gyroscopes which measures the rotational motion of body with respect to the inertial reference system. They are together know as Inertial Measurement Unit. Inertial navigation system is self contained in the sense that there is no need of a signal from outside the system for navigation. Unlike the stable platform techniques (where the inertial sensors are attached to a stable platform and isolated from the rotational motion of the vehicle) the sensors are attached directly to the body of the vehicle in strapdown technology.

2.1 Coordinate Frames

The purpose of coordinate frames here is to exchange the information between interfacing systems in efficient manner. Below is some of the systems commonly used in navigation implementations.

2.1.1 Earth Centered Inertial Frame

The earth centered inertial (ECI) system is oriented with respect to the sun. Its origin is fixed at the center of the earth. The z axis points northward along the earth's rotation axis. The x axis points outward in the earth's equatorial plane exactly at the sun or to vernal equinox. The y axis completes the right

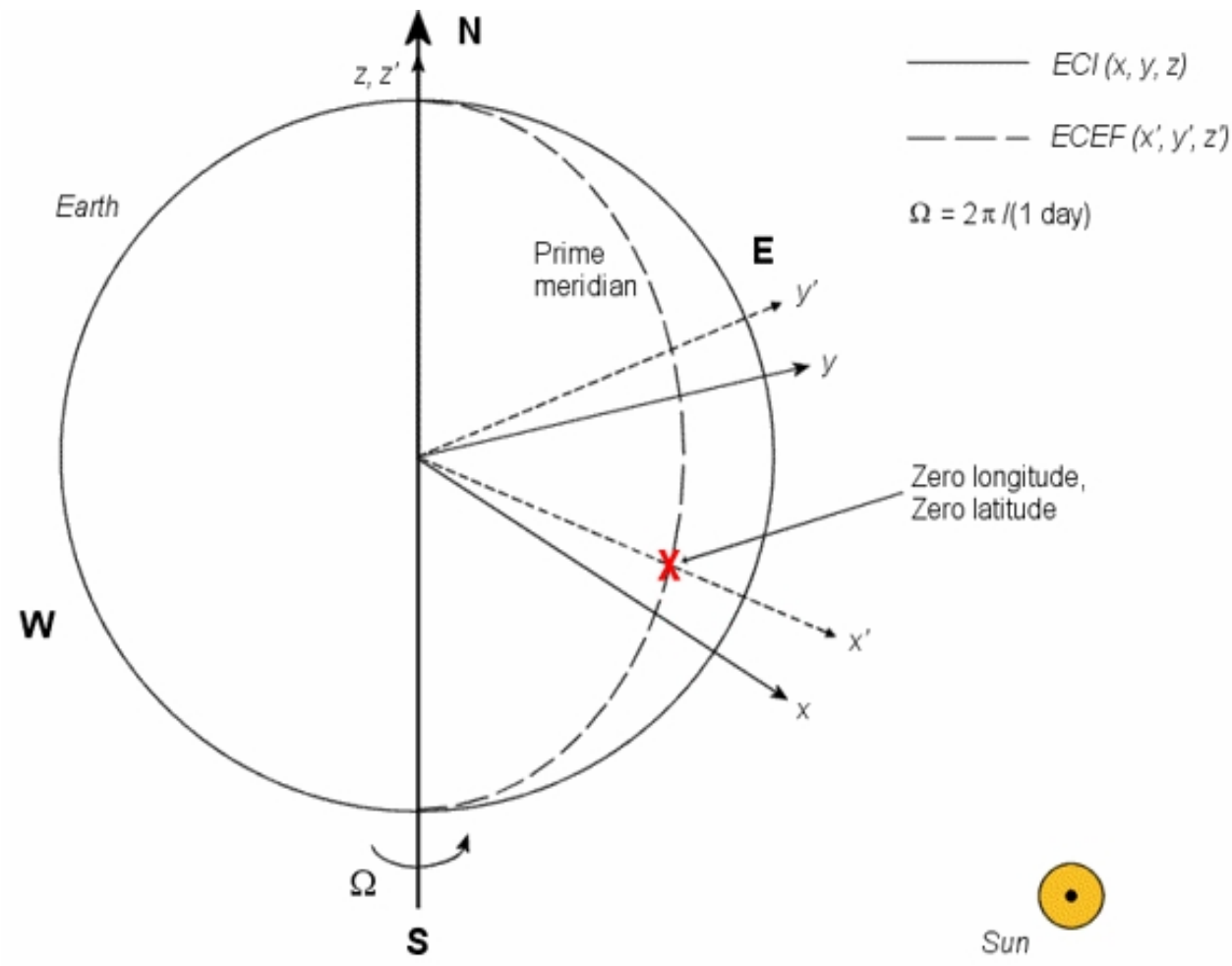

Figure 2.1: ECI frame, image downloaded from mathworks.com[12]

hand system pointing towards eastward direction.

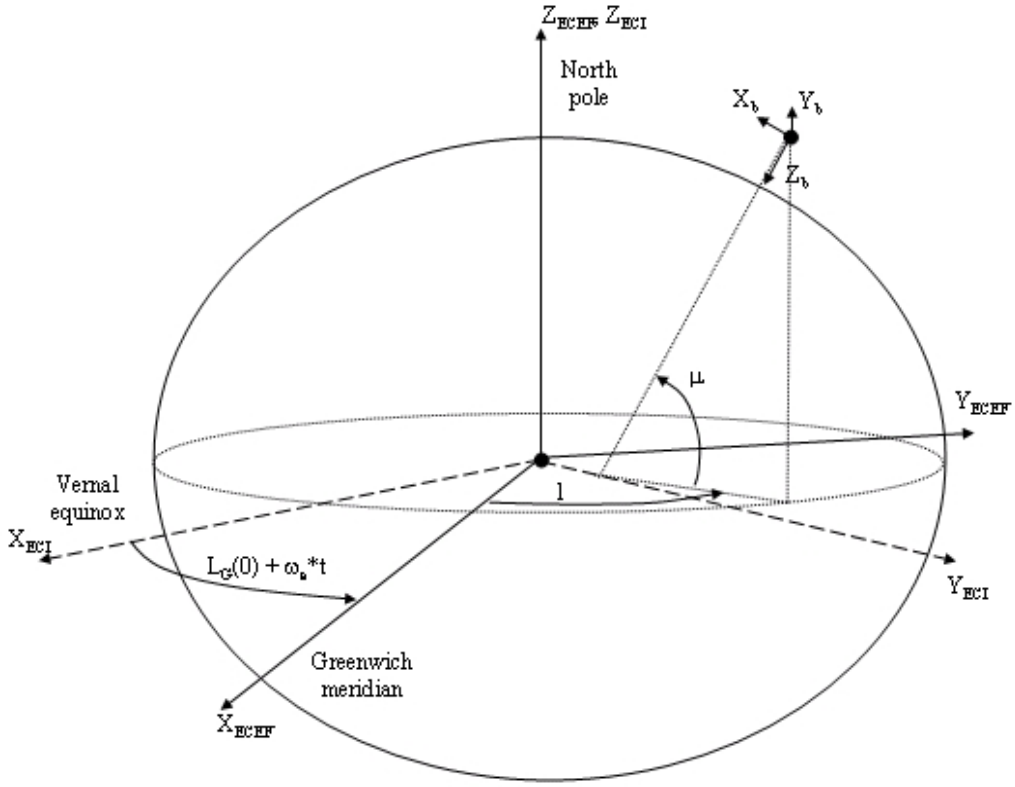

Figure 2.2: ECI and ECEF frame, image downloaded from mathworks.com[12]

2.1.2 Earth Centered Earth Fixed Frame

This coordinate system is fixed within the earth and its rotation. It is centered at the center of earth and the z axis is parallel to and aligned with the direction of earth rotation. x axis is along the Greenwich meridian and y axis complete the right hand system[4].

2.1.3 Local Geodetic Frame (n-frame)

This coordinate system is particularly useful in representing vehicle attitude and velocity for operations on or near the surface of the earth. A commonly used such a coordinate system is the North-East-Down (NED) system.

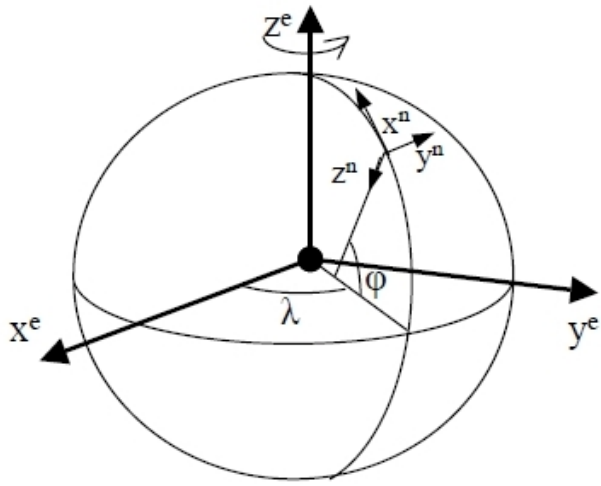

Figure 2.3: ECEF and navigation frame

2.1.4 Body Frame (b-frame)

The body frame is rigidly attached to the body of the vehicle and defined by its geometry. The x axis is along the longitudinal axis and z axis pointing downward and y axis completing the right angle system. In the case of strapdown INS sensors, its triad frame is identical to the body axis of the vehicle.

2.2 Inertial Measurement Unit

The inertial measurement unit consists of two 3-axis sensors, accelerometer and gyroscope. IMU with the navigation computer is known as INS. The sensors widely used these days are MEMS sensors. Advancements in MEMS sensor technology has produced cost effective accelerometers and gyroscopes which have advantage of reduced cost and weight and offer more reliability. But these sensors are more affected by inaccuracies influenced by temperature variation and bias. A detailed description of the technology is not intended here.

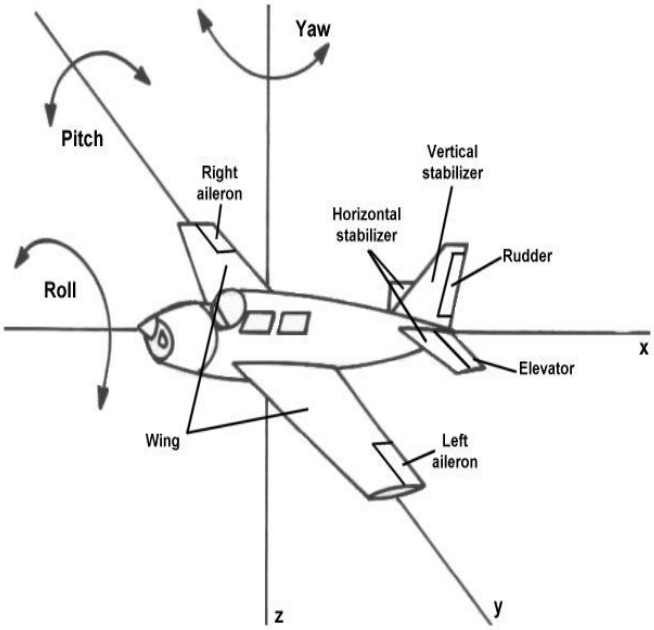

Figure 2.4: Body axis, image downloaded from http://www.aerospaceweb.org [11]

2.2.1 Accelerometer

Accelerometer measures the specific force (f^b) associated with the body. Specific force is the difference between true acceleration in space and acceleration due to gravity. The common MEMS accelerometer contains a small plate attached to torsion levers. This plates move under acceleration which changes the capacitance between them. This change in capacitance is proportional to the linear acceleration.

$$f^b = \ddot{r} - G \tag{2.1}$$

where G is acceleration due to gravity and r is the position vector

2.2.2 Gyroscope

Gyroscope is used to measure the orientation of the body and it gives the angular rate as outputs which corresponds to the rotation of body with respect to inertial frame expressed in body frame (ω_{ib}^b). The core of a MEMS gyro is a vibrating lever. When undergoing an angular rotation, its vibration frequency is changed and detect the rotation.

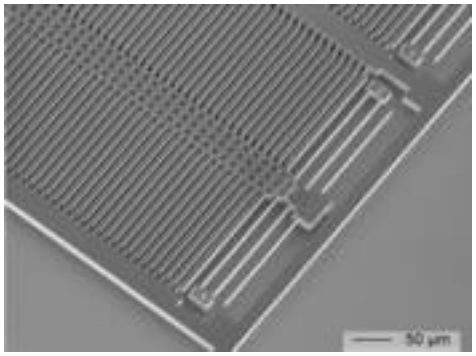

Figure 2.5: MEMS accelerometer, image downloaded from www.hsg-imit.de[10]

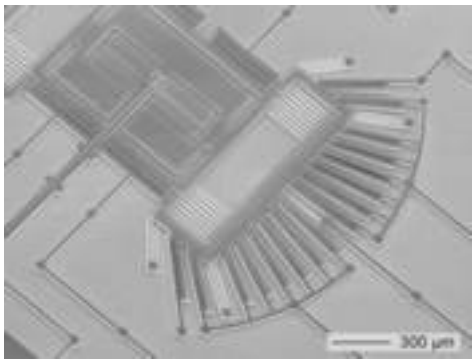

Figure 2.6: MEMS gyroscope, image downloaded from www.hsg-imit.de[10]

2.2.3 IMU Errors

Below is a brief introduction of some major error sources associated with MEMS sensors.

Accelerometer Bias error

There is some constant offset in accelerometer output which changes slightly after each turn on.

Misalignment and Nonorthogonality

The axis of the sensors may be misaligned from the body axis which will produce an error measurement

Accelerometer Scale Factor error

This error is resulted from the scale factor which is used to convert the value to common measurement units. This error is proportional to the sensed acceleration.

Gyroscope Drift

The gyroscope drift or bias error results from bias which is a constant offset from correct output but this varies after each turn on of the sensor.

Gyroscope Scale Factor error

This error is proportional to sensed angular rates resulting from variation in the scale factor of the gyro sensor.

Random Noise

Random noise errors associated with measurements

Nonlinearity due to Temperature variations

The operation of the MEMS technology is affected considerably by temperature variations.

2.3 Coordinate Transformation

2.3.1 Euler Rotations

Euler angles were developed by Leonhard Euler to describe the orientation of a rigid body in 3-D space. Spatial orientation of any frame is represented by three sequence of rotation described by euler angle. In order to represent a vector in different coordinate system, we use coordinate transformation. There are different methods to do it, but the most common method is euler angles roll (θ), pitch (ϕ) and yaw (ψ). It is important to specify the order of rotation and most common one is

- A positive ϕ.
- A positive θ.
- A positive ψ.

In the case of a coordinate transformation from n-frame to body frame for an aircraft, this sequence corresponds first to a right handed rotation around the aircraft's z axis (positive ψ), followed by a right handed rotation around aircraft's y axis (positive θ) and a right handed rotation around aircraft's x axis (positive ϕ). The rotation matrix $C_{b/r}$ for this sequence of rotation can be written as

$$C_r^b = \begin{bmatrix} C\theta\, C\psi & C\theta\, S\psi & -S\theta \\ (-C\phi\, S\psi + S\phi\, S\theta\, C\psi) & (C\phi\, C\psi + S\phi\, S\theta\, C\psi) & S\phi\, C\theta \\ (-S\phi\, S\psi + C\phi\, S\theta\, C\psi) & (-S\phi\, C\psi + C\phi\, S\theta\, C\psi) & C\phi\, C\theta \end{bmatrix} \tag{2.2}$$

Where C and S represent cosine and sine respectively. The euler angle can obtain from above rotation matrix as follows

$$\begin{aligned} \theta &= -sin^{-1}(C_{13}) \\ \phi &= atan2(C_{23}, C_{33}) \\ \psi &= atan2(C_{12}, C_{11}) \end{aligned} \tag{2.3}$$

Where C_{ij} is the $(i,j)^{th}$ element of $C_{b|r}$ and atan2 is four quadrant inverse tangent.

2.4 Equation of Motion

2.4.1 INS kinematic Equations

In navigation applications, navigation information is commonly required in navigation frame (Local Geodetic Frame). Position and velocity expressed

in navigation frame as

$$\underline{r}^n = [\varphi \;\; \lambda \;\; h]^T \tag{2.4}$$

$$\underline{v}^n = [V_N \;\; V_E \;\; V_D]^T \tag{2.5}$$

Where φ is latitude, λ is longitude, h is altitude, V_N is velocity towards north, V_E is velocity towards east and V_D is velocity towards down. Motion of the vehicle is expressed by the INS kinematic equation or navigation equation. Derivation and more explanation about these equations can be found in many literatures, for example in Strapdown Inertial Navigation Technology[4]. This equation has 3 parts position, velocity and attitude equations as shown below.

$$\dot{\underline{r}}^n = \begin{bmatrix} \dot{\varphi} \\ \dot{\lambda} \\ \dot{h} \end{bmatrix} = \begin{bmatrix} \frac{1}{M+h} & 0 & 0 \\ 0 & \frac{1}{(N+h)cos\varphi} & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} V_N \\ V_E \\ V_D \end{bmatrix} \tag{2.6}$$

$$\dot{\underline{V}}_e^n = C_b^n \underline{f}^b - (2\underline{\omega}_{ie}^n + \underline{\omega}_{en}^n) \times \underline{V}^n + \underline{g}^n \tag{2.7}$$

$$\begin{aligned} \dot{C}_b^n &= C_b^n \Omega_{nb}^b \\ &= C_b^n (\Omega_{ib}^b - \Omega_{in}^b) \end{aligned} \tag{2.8}$$

where M and N are radius of curvature of meridian and prime vertical. C_b^n is the rotation matrix from b-frame to n-frame. g is gravity vector. Ω_{nb}^b is skew symmetric matrix of w_{nb}^b and w_{nb}^b is equal to $w_{ib}^b - C_n^b(w_{ie}^n + w_{en}^n)$. f^b is the specific force which is the difference between true acceleration in space and acceleration due to gravity. ω_{ib}^b is the output of the gyroscope and ω_{ie}^n is the earth rotation rate with respect to inertial frame expressed in navigation frame. ω_{en}^n is the rotation rate of the navigation frame with respect to ECEF frame.

2.4.2 INS Mechanization

This section gives a brief explanation about how the INS calculate the navigation frame values from the IMU measurements. The inertial navigation sensors measure f^b the specific force (accelerometer) and the rotation of body (gyroscope) ω_{ib}^b. The process of converting this measurements to navigation information have mainly 4 steps. (For more about this, please check Adriano's thesis report[22])

- Correction of raw measurement data.

- Attitude update (C_b^n)
- Transform specific force into n-frame.
- Velocity and position calculation

Usually the IMU sensors outputs the velocity and angular increments ($\Delta\tilde{\underline{V}}_f^b$ and $\Delta\tilde{\underline{\theta}}_{ib}^b$) in body frame for a sampling time period.

Correction of measurement data

The raw measurement data is corrupted by turn on bias, in run bias, scale factor errors and measurement noise. These errors can be measured on ground or it can be estimated during operation. Such measurements can be corrected according to the following equations.

$$\Delta\underline{\theta}_{ib}^b = \begin{bmatrix} \frac{1}{(1+S_{gx})} & 0 & 0 \\ 0 & \frac{1}{(1+S_{gy})} & 0 \\ 0 & 0 & \frac{1}{(1+S_{gz})} \end{bmatrix} (\Delta\tilde{\underline{\theta}}_{ib}^b - \underline{b}_g\Delta t) \tag{2.9}$$

$$\Delta\underline{V}_f^b = \begin{bmatrix} \frac{1}{(1+S_{ax})} & 0 & 0 \\ 0 & \frac{1}{(1+S_{ay})} & 0 \\ 0 & 0 & \frac{1}{(1+S_{az})} \end{bmatrix} (\Delta\tilde{\underline{V}}_f^b - \underline{b}_a\Delta t) \tag{2.10}$$

where $\underline{b}_a$ and $\underline{b}_g$ are bias of the accelerometer and gyroscope respectively. Similarly $\underline{s}_a$ and $\underline{s}_g$ are the scale factor errors of the accelerometer and gyroscope respectively. Δt is the sampling time.

Attitude update

The Body angular increments with respect to the navigation frame can be represented as

$$\begin{aligned} \Delta\underline{\theta}_{nb}^b &= [\Delta\theta_x \ \Delta\theta_y \ \Delta\theta_z] \\ &= \Delta\underline{\theta}_{ib}^b - C_n^b(\underline{\omega}_{ie}^n + \underline{\omega}_{en}^n)\Delta t \\ C_n^b &= (C_b^n)^T \end{aligned} \tag{2.11}$$

The direction cosine matrix (C_b^n) is calculated from the angular increments using quaternion approach. In quaternion approach, the rotation matrix is

expressed by a single rotation angle about a fixed axis. The angular increment obtained before can be used to update the quaternion vector $\underline{q}$ as,

$$\underline{q}_{k+1} = \underline{q}_k + 0.5 \begin{bmatrix} c & s\,\Delta\theta_z & -s\,\Delta\theta_y & s\,\Delta\theta_x \\ -s\,\Delta\theta_z & c & s\,\Delta\theta_x & s\,\Delta\theta_y \\ s\,\Delta\theta_y & -s\,\Delta\theta_x & c & s\,\Delta\theta_z \\ -s\,\Delta\theta_x & -s\,\Delta\theta_y & -s\,\Delta\theta_z & c \end{bmatrix} \underline{q}_k \tag{2.12}$$

where

$$\begin{aligned} s &= \frac{2}{\Delta\theta} sin \frac{\Delta\theta}{2} \\ c &= 2(cos\frac{\Delta\theta}{2} - 1) \\ \Delta\theta &= \sqrt{\Delta\theta_x^2 + \Delta\theta_y^2 + \Delta\theta_z^2} \end{aligned} \tag{2.13}$$

The direction cosine matrix C_b^n can finally obtained in terms of quaternion as

$$C_n^b = \begin{bmatrix} (q_0^2 + q_1^2 - q_2^2 - q_3^2) & 2\,(q_1\,q_2 + q_0\,q_3) & 2\,(q_1\,q_3 - q_0\,q_2) \\ 2\,(q_1\,q_2 - q_0\,q_3) & (q_0^2 - q_1^2 + q_2^2 - q_3^2) & 2\,(q_2\,q_3 + q_0\,q_1) \\ 2\,(q_1\,q_3 + q_0\,q_2) & 2\,(q_2\,q_3 - q_0\,q_1) & (q_0^2 - q_1^2 - q_2^2 + q_3^2) \end{bmatrix} \tag{2.14}$$

Transformation of specific force into n-frame

For the implementation of the INS equations, specific force needs to convert into the n-frame. This is done as follows

$$\Delta\underline{V}_f^n = C_b^n \begin{bmatrix} 1 & 0.5\,\Delta\theta_z & -0.5\,\Delta\theta_y \\ -0.5\,\Delta\theta_z & 1 & -0.5\,\Delta\theta_x \\ 0.5\,\Delta\theta_y & -0.5\,\Delta\theta_x & 1 \end{bmatrix} \Delta\underline{V}_f^b \tag{2.15}$$

Velocity and position update

The velocity increment in n-frame is obtained by applying the coriolis and gravity correction

$$\Delta\underline{V}^n = \Delta\underline{V}_f^n - (2\underline{\omega}_{ie}^n + \underline{\omega}_{en}^n) \times \underline{V}^n \Delta t + \underline{\gamma}^n \Delta t \tag{2.16}$$

where γ^n is $[0\ \ 0\ \ \gamma]^T$ and γ is the normal gravity at the geodetic latitude φ and height h. Once the velocity increment is obtained the updated velocity is given by

$$\underline{V}_{k+1}^n = \underline{V}_k^n + \Delta\underline{V}_{k+1}^n \tag{2.17}$$

Finally the position in n-frame can be obtained by integrating the velocity

$$\underline{r}_{k+1}^{n} = \underline{r}_{k}^{n} + \frac{1}{2} D^{-1} \left(\underline{V}_{k}^{n} + \underline{V}_{k+1}^{n} \right) \Delta t \tag{2.18}$$

where

$$D = \begin{bmatrix} \frac{1}{M+h} & 0 & 0 \\ 0 & \frac{1}{(N+h)cos\varphi} & 0 \\ 0 & 0 & -1 \end{bmatrix} \tag{2.19}$$

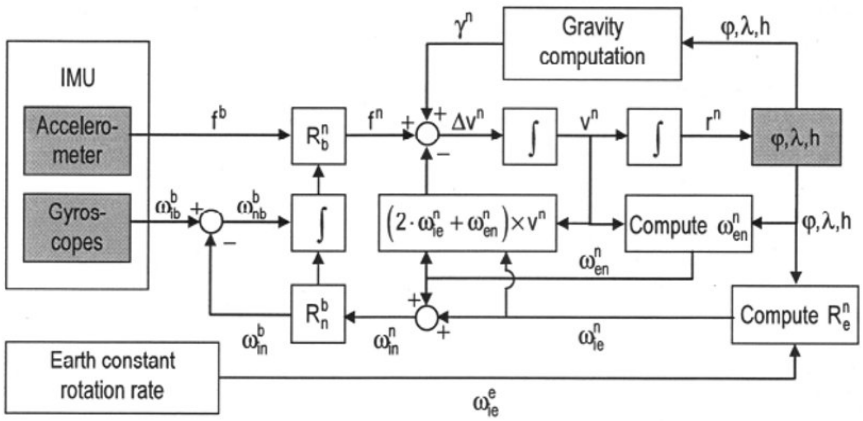

Figure 2.7: INS mechanization, image from MSc thesis of Adriano Solimeno [22]

Chapter 3

Global Positioning System

This chapter briefly cover the GPS navigation system. Since the implementation considered is loosely coupled, there is not much interest in considering the errors or there is no effort to correct for the GPS errors. The information presented here is mainly collected from wikipedia[7] and kowoma website[9].

3.1 GPS

The global positioning system is a satellite based navigation system which was developed for US military purposes during 70's and declared completely operational in 1995. Generally it has two quality of service, one for military and another for civilian purposes. Also they have the option to degrade the quality of the signals whenever and wherever they need, known as "selective availability". This policy was reconsidered in 2000 and ended the selective availability and gives an accuracy of 100m to 20 m for civilian users.

3.1.1 System Information

The whole GPS system has three components satellites or space segment, control centers (control segment) and GPS receiver modules (user segment). The GPS space segment consists of 24 satellites orbiting in six different orbital planes (four each in a plane) and completing one rotation in 11 hrs 58 minutes. Figure 3.1 gives an approximate idea about the satellite position and orbits. These planes have an inclination of 55 and are separated by 60 right ascension of the ascending node (Fig. 3.2). These satellites have an average orbital radius of 20200 km. This arrangement make sure that at least six satellites are visible in any point of earth. GPS satellites continually send navigation messages at 50 bit/s and main information contained are the time

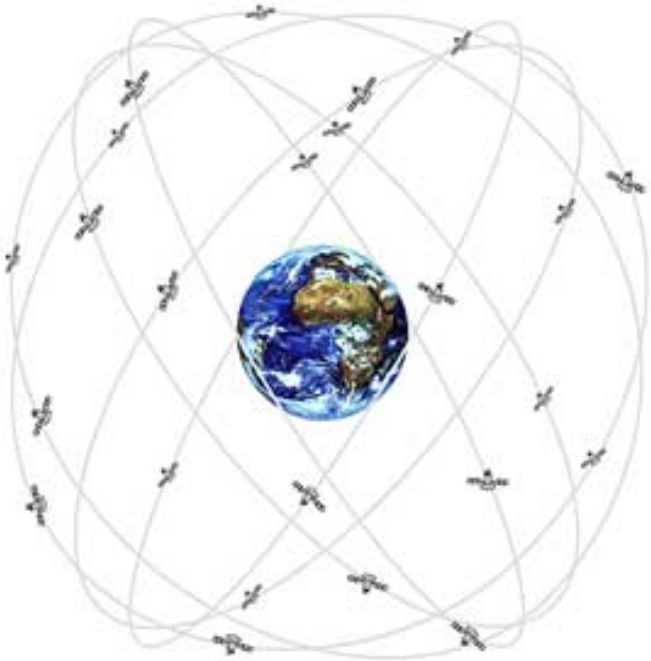

Figure 3.1: Orbits of GPS satellites, image courtesy of kowoma.de

when message was sent, precise orbital information (the ephemeris), and the general system health and rough orbits of all GPS satellites (the almanac). The data format received is usually known as "NMEA" message. The receiver measures the transit time of each message and computes the distance to each satellite[7]. The control center continuously check the health of the satellites and do necessary correction when needed. The main monitoring stations are in Hawaii, Kwajalein, Ascension Island, Diego Garcia, Colorado Springs, Colorado and some other monitor stations operated by the National Geospatial-Intelligence Agency. The tracking information is sent to the master control station. Then mainly two corrections have to perform, the satellite clocks are synchronized to very high precision and also necessary orbital maneuvering have to perform on satellites which are diverted from the correct orbits. During orbital maneuvering, satellite is marked unhealthy and the signal is excluded by the receiver.

The GPS receivers receive the signals from different satellites and calculate the position information. The important components of GPS receivers are antenna, receiver processors and a highly stable clock (crystal oscillator). At least visibility of 4 satellites are necessary for the accurate determination of the position. In some applications a kalman filter (tightly coupled implementation) can be employed to correct for the errors in the GPS signal. In

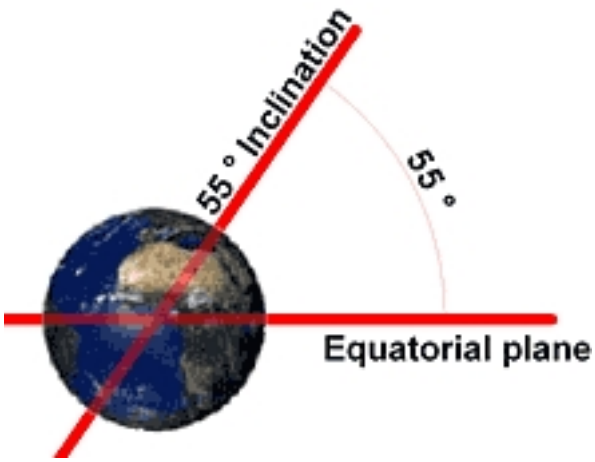

Figure 3.2: Orbital inclination of GPS satellites, image courtesy of kowoma.de

the presence of some auxiliary sensors even three satellites can determine the position well accurately.

3.1.2 Calculation of Position

All satellites broadcast at the same two frequencies, 1.57542 GHz (L1 signal) and 1.2276 GHz (L2 signal)[7]. It employs CDMA technique to transmit these signal. The low rate message signal is encoded with Pseudo Random codes and each satellites has different codes. Two distinct CDMA encodings are used, the coarse/acquisition (C/A) code (Gold code) and the precise (P) code [7]. The L1 signal employs both C/A and P codes and it's for the civilian receivers . L2 employs P codes and it's meant for military purposes.

GPS receivers use the geometric trilateration to combine the information from different satellite to predict the correct location. The GPS message contains the information about the time when message was sent, precise orbital information, health of the system and rough information regarding the orbits of other satellites. The receiver measures the time of transit of each message and compute the distance to the satellite. If we know the distance from one satellite, we can assume that the receiver is on the surface of a sphere centered by the satellite having the radius equal to the distance. Intersection of two spheres (if it intersect at more than one point) will be a circle and when three spheres intersect result will be two points. Intersection of two spheres is shown in figure 3.3 to have an understanding. We can assume that the indicated position of the GPS receiver is at the intersection of four

spheres with the assumption of no errors. But in reality there is a large error contribution by the receiver clock. So the estimated distance from satellite to the receiver (pseudo range) is corrupted by error. The assumption of intersection of three spheres may be valid in such a case but the intersection of four spheres is unlikely. The distance from the valid estimate of GPS receiver position to the surface of the sphere corresponding to the fourth satellite can be used to compute a clock correction [7].

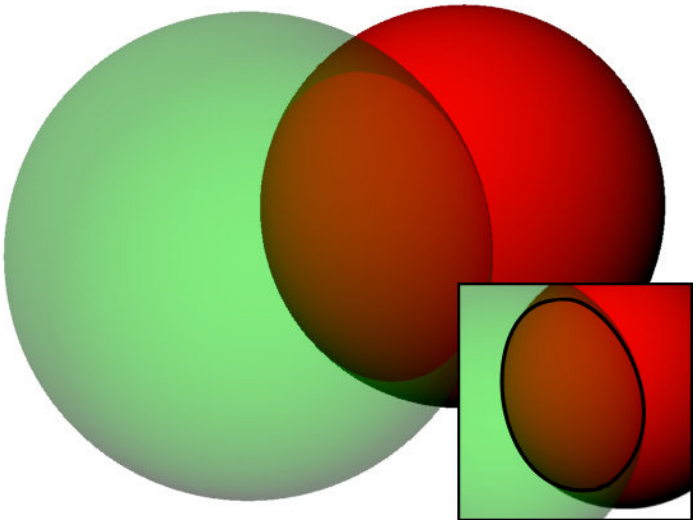

Figure 3.3: Sphere intersection, image courtesy of wikipedia

3.2 Errors in GPS

Though the GPS system is very accurate in theory, its performance is affected by many error sources. Some of them are explained in the next section.

Clock and Calculation Errors

In GPS satellites, highly accurate atomic clock (eg. fig 3.4) is used but it also has some inaccuracy and problem with synchronization of all clocks in satellites. In practice the clocks are allowed to operate in small relative drift but it is estimated in ground station and clock correction data is generated and send in the navigation message (GPS time). This GPS time is used in the calculation of the pseudo range. The clock in the receiver is usually low cost quartz oscillator which have very low accuracy. Due to this, some error in the

navigation parameter is introduced. Also calculation on the microprocessors in the receivers cause the round off errors.

Figure 3.4: Atomic clock used in GPS satellites, image courtesy of kowoma.de

Atmospheric Effects

Atmospheric properties have considerable influence on the accuracy of the GPS signal. The major effect is the delay in ionosphere and troposphere. In ionosphere, the presence of electrons and other particles affect the propagation of the signals and these charge distribution often affected by the solar activities. This effect is less when the satellite is overhead and increases when it moves to horizon. Where as in troposphere the water vapor causes the refraction of the signals and there by causing the phase delay. Please see figure 3.5 for better understanding.

Multipath

Navigation signals reflecting from the geographical objects and building are also received in the GPS receiver. These received signals interfere with the pure signals and add the errors in signals. Multipath is a major issue when navigating in cities. Figure 3.6 will give more understanding about this effect.

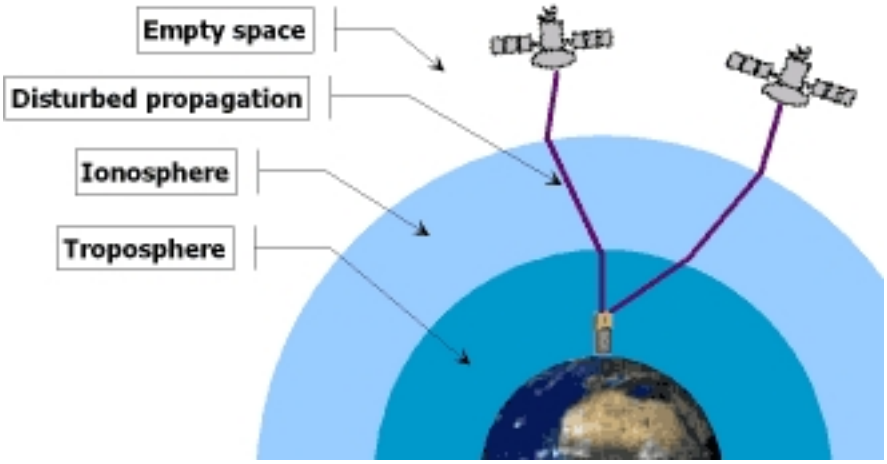

Figure 3.5: Effect of atmosphere on GPS signals, image courtesy of kowoma.de

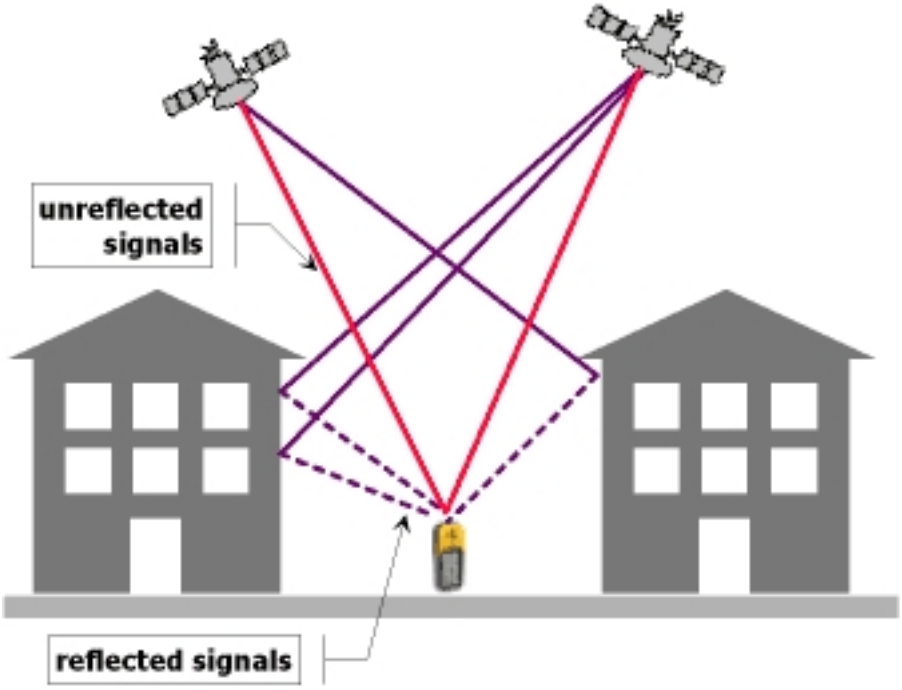

Figure 3.6: Multipath, image courtesy of kowoma.de

Satellite Orbital Errors

Ephemeris data which is transmitted to receiver is calculated from the orbital dynamics of the satellites. These calculation involve the gravity model and it is a curve fit to the measured data. The trajectory of the satellites can be affected by many factors like atmosphere and other perturbations. Satellites also need constant monitoring and maneuvering to keep the correct trajectory.

Relativity

Relativity effects have some contribution to the errors. For example the relativistic time slowing due to the speed of the satellite of about 1 part in 10^{10}, the gravitational time dilation that makes a satellite run about 5 parts in 10^{10} faster than an earth based clock, and the Sagnac effect due to rotation relative to receivers on earth[7].

Measurement Noise

There is substantial contribution to error by the measurement noise. These noise are created in stages of signal propagation and processing. Some well known sources of error are the receiver noise, quantization noise and noise due to electronics.

Summery of Errors

The errors contributed from some of the major sources are listed as a table 3.1. These errors added together can accumulate up to $\pm 15\,\mathrm{m}$ without considering the selective availability. With the addition of satellite based augmentation system (like WAAS and EGNOS) accuracy can be improved up to or less than $\pm 5\,\mathrm{m}$. Such systems will compensate for the ionospheric effects and also improve orbit and clock errors.

Ionospheric effects	$\pm$ 5 m
Shifts in the satellite orbit	$\pm$ 2.5 m
Clock errors	$\pm$ 2 m
Multipath effect	$\pm$ 1 m
Tropospheric effects	$\pm$ 0.5 m
Calculation and rounding errors	$\pm$ 1 m

Table 3.1: Summery of GPS errors obtained from kowoma.de[9]

Chapter 4

Sensor Fusion (Kalman Filtering)

4.1 Introduction to Kalman Filter

The kalman filter is a recursive filter which estimate the states of the dynamics of a system by noisy measurement. It was published in 1960 by Rudolf E. Kalman and it is now used in many fields of engineering, Economics and Science. It is known as linear quadratic estimator and together with linear quadratic regulator it solves the linear quadratic gaussian control problems [6]. The kalman filter has two distinct steps, prediction and update. Predict or time update step utilize the previous state estimate information to predict the current estimate of state variables. In the second step, also known as measurement update, the measurement information at current time step is used to correct the estimate to get more accurate state information

Time Update

$$\textit{Predicted state} \quad \hat{X}_{k|k-1} = F_k \hat{X}_{k-1|k-1} + B_{k-1} u_{k-1} \tag{4.1}$$

$$\textit{Predicted estimate covariance} \quad P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_{k-1} \tag{4.2}$$

Measurement Update

$$\textit{Innovation (residual) covariance} \quad S_k = H_k P_{k|k-1} H_k^T + R_k \tag{4.3}$$

$$\textit{Optimal kalman gain} \quad K_k = P_{k|k-1} H_k^T S_k^{-1} \tag{4.4}$$

$$\textit{Updated state estimate} \quad \hat{X}_{k|k} = \hat{X}_{k|k-1} + K_k (Z_k - H_k \hat{X}_{k|k-1}) \tag{4.5}$$

$$\textit{Updated estimate covariance} \quad P_{k|k} = (I - K_k H_k) P_{k|k-1} \tag{4.6}$$

Where $\hat{X}_{k|k}$ is the estimate of state at time k given observations up to and including time k, K is kalman gain, H is measurement model, Z_k is the measurement at time k, F is state transition matrix, B is input matrix, P is estimate covariance, R is measurement covariance and Q is process covariance.

4.2 Extended Kalman Filter

The simple kalman filter is applicable only to linear systems. But the real world problems are usually nonlinear either in process model or in measurement model or both. Extended kalman filter is applicable to such problems with the condition that the process model and measurement model to be differential functions of state variables. In EKF, nonlinear state equation and measurement equation are represented as

$$\dot{X}(t) = f(X(t), U(t)) + w(t) \tag{4.7}$$

$$Z_m(t) = h(X(t), U(t)) + v(t) \tag{4.8}$$

Where f is the nonlinear state equation, h is the nonlinear measurement equation, U is the input vector. w and v represent process and measurement noise. The above state and measurement nonlinear equations are linearized about the prior estimate of the state at each instant of time by calculating the Jacobian with respect to the state variables as.

$$F_k = \left(\frac{\partial f}{\partial X}\right); \quad H_k = \left(\frac{\partial h}{\partial X}\right); \tag{4.9}$$

And for discrete time implementation of EKF, the linearized system is discretized in time by computing the system state transition matrix from F_k as [17], where $\triangle t$ is time interval.

$$\Phi_{k|k-1} = e^{F_k \triangle t} \cong [I + F_k \triangle t] \tag{4.10}$$

The time update and measurement update are as follows.

Time Update

$$\hat{X}_{k|k-1} = \hat{X}_{k-1|k-1} + \int_{k-1}^{k} f(X(t), U(t))\, \mathrm{d}t \tag{4.11}$$

$$P_{k|k-1} = \Phi_{k|k-1} P_{k-1} \Phi_{k|k-1}^T \tag{4.12}$$

The state integration can either done using simple approximation or using fourth order Runge-Kutta method.

Measurement Update

$$S_k = H_k P_{k|k-1} H_k^T + R_k \tag{4.13}$$
$$K_k = P_{k|k-1} H_k^T S_k^{-1} \tag{4.14}$$
$$\hat{X}_{k|k} = \hat{X}_{k|k-1} + K_k(Z_k - h(\hat{X}_{k|k-1})) \tag{4.15}$$
$$P_{k|k} = (I - K_k H_k) P_{k|k-1} \tag{4.16}$$

4.3 Unscented Kalman Filter

The unscented kalman filter was proposed in 1997 as alternative to extended kalman Filter. When the system become highly nonlinear, ekf is less efficient in estimation. There are two drawbacks in ekf, that it is too difficult to tune and only reliable for systems which are almost linear on the time scale of the update intervals [18]. In this new approach *"Unscented Transform"* [20] is used to parameterize mean and covariance which is founded on the intuition that *it is easier to approximate a Gaussian distribution than it is to approximate an arbitrary nonlinear function or transformation* [18].

4.3.1 Unscented Transform

The unscented transform is a method for calculating the statistics of a random variable which undergo nonlinear transformation [16]. For example consider a random variable x with dimension L through a nonlinear function $y = f(x)$, and x has covariance P_{xx} and mean $\overline{x}$. A set of points "sigma points" are chosen such that their mean and covariance are $\overline{x}$ and P_{xx} respectively. These points are applied in the nonlinear function $y = f(x)$ to get the $\overline{y}$ and P_{yy}[18]. It is important to note that the sigma points are not chosen in random but rather according to some deterministic algorithm. The n-dimensional random variable x is approximated by *2L+1* weighted sigma points as,

$$\begin{aligned} X_0 &= \overline{x} \\ X_i &= \overline{x} + (\sqrt{(L+\lambda)P_{xx}}\,)_i \text{ , i} = 1, ..L \\ X_{i+L} &= \overline{x} - (\sqrt{(L+\lambda)P_{xx}}\,)_{i-L} \text{ , i} = L+1, ..2L \end{aligned} \tag{4.17}$$

$$
\begin{aligned}
W_0^m &= \lambda/(L+\lambda) \\
W_0^c &= \lambda/(L+\lambda) + (1-\alpha^2+\beta) \\
W_i^m &= W_i^c = 1/2(L+\lambda)\ \text{, i} = 1, ..2L
\end{aligned} \tag{4.18}
$$

Where W_i^m and W_i^c are the weights of the mean and covariance calculation associated with i^{th} point. $\lambda = \alpha^2(L+k) - L$ is a scaling parameter and α determine the spread of the sigma points around $\overline{x}$. k is secondary scaling parameter and β corresponds to the prior knowledge of the distribution of X (for Gaussian distributions, β=2). The default values for these parameters are $\alpha = 10^{-4}$, $\beta = 2$ and $k = 3 - L$. The method instantiate each sigma point through the function $y = f(x)$ resulting in a set of transformed sigma points, mean and covariance as

$$
Y_i = f(X_i) \tag{4.19}
$$

The mean is given by the weighted average of the transformed points,

$$
\overline{y} = \sum_{i=0}^{2L} W_i^m Y_i \tag{4.20}
$$

The covariance is the weighted outer product of the transformed points,

$$
P_{yy} = \sum_{i=0}^{2L} W_i^c [Y_i - \overline{y}][Y_i - \overline{y}]^T \tag{4.21}
$$

Figure 4.1 clearly shows the idea of EKF and UKF in a nice manner.

4.3.2 Unscented Kalman Filter

This method claims to have superior performance than ekf and directly comparable to second order Gaussian filter and do not restrict to assume that the distributions of noise sources are Gaussian [18]. There can be two different implementations, non-augmented and augmented ukf. In augmented ukf the state and covariance matrices are augmented with process and measurement noise components. This method results in better state estimation when the noise is non-additive but the estimation process will become more computational intensive due to the increase in number of states. Non-augmented ukf is implemented in this thesis. Considering the nonlinear state and measurement equations as before ((4.7) and (4.7)), the time update and measurement update are as follows.

Time Update

The time update is as explained in unscented transform where a set of *2L+1*

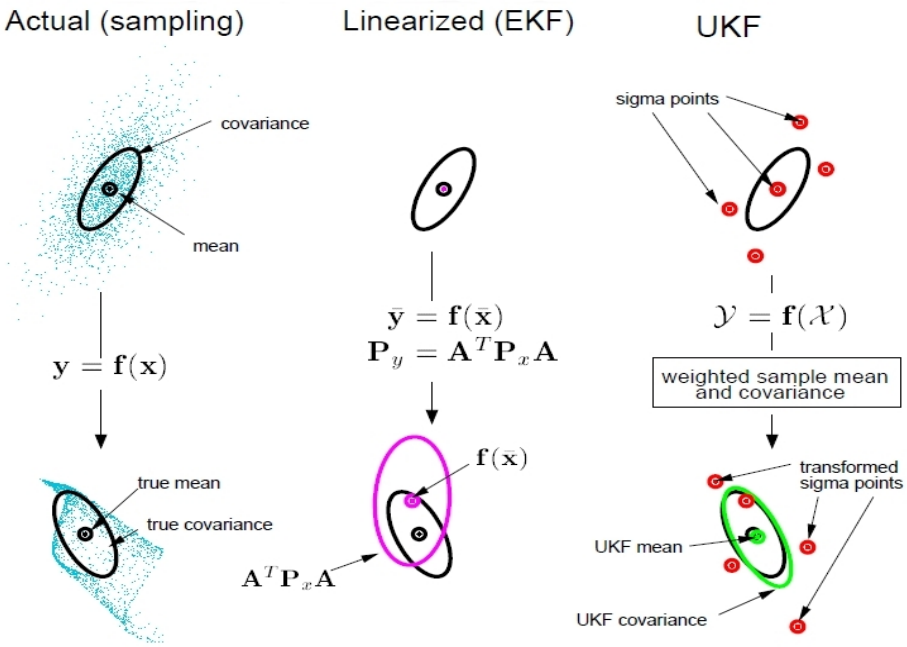

Figure 4.1: Comparison of mean propagation in EKF , UKF and Sampling. image form the reference[19]

sigma points ($X^i_{k-1|k-1},\ i = 0, ..2L$) are calculated from the previous known mean ($x_{k-1|k-1}$) of state vectors according to the equations (4.17), where L is the number of states. Then these sigma points are propagated through the non-linear state equations (4.7) to obtain the updated state vector and covariance matrix ($\hat{X}_{k|k-1}$ and $\hat{P}_{xx,k|k-1}$) according to the equations (4.20) and (4.21) respectively as

$$\hat{X} = \sum_{i=0}^{2L} W_i^m X^i_{k-1|k-1} \tag{4.22}$$

$$P_{k|k-1} = \sum_{i=0}^{2L} W_i^c [X^i_{k|k-1} - \hat{X}_{k|k-1}][X^i_{k|k-1} - \hat{X}_{k|k-1}]^T \tag{4.23}$$

Measurement Update

As in time update, a set of *2L+1* sigma points ($X^i_{k|k-1},\ i = 0, ..2L$) are derived from the updated state and covariance matrices where L is the dimension of the state. The sigma points are propagated through the observation

function h as

$$\gamma_k^i = h\left(X_{k|k-1}^i\right) \ , \ i = 0, ..2L \tag{4.24}$$

The predicted measurement and measurement covariance are calculated with weighted sigma points as follows

$$\hat{Z}_k = \sum_{i=0}^{2L} W_i^m \gamma_k^i \tag{4.25}$$

$$P_{zz,k} = \sum_{i=0}^{2L} W_i^c [\gamma_k^i - \hat{Z}_k][\gamma_k^i - \hat{Z}_k]^T \tag{4.26}$$

The state-measurement cross covariance matrix is obtained as

$$P_{xz,k} = \sum_{i=0}^{2L} W_i^c [X_{k|k-1}^i - \hat{X}_{k|k-1}][\gamma_k^i - \hat{Z}_k]^T \tag{4.27}$$

and ukf kalman gain, updated state and covariance are respectively

$$K_k = P_{xz,k}\, P_{zz,k}^{-1} \tag{4.28}$$

$$\hat{X}_{k|k} = \hat{X}_{k|k-1} + K_k\,(Z_k - \hat{Z}_k) \tag{4.29}$$

$$P_{k|k} = P_{k|k-1} - K_k\, P_{zz,k}\, K_k^T \tag{4.30}$$

Chapter 5

INS-GPS Integration

5.1 System Process Model for Integration

In ekf implementation of ins-gps integration, the errors of the system mechanization is modeled. There are different approaches, proposed by different researchers. In all such implementation, idea is to avoid the need of calculating Jacobian necessary for ekf by simplifying the error equations using different approximations and assumptions. But when the errors are very large, these methods fail to give a reasonable estimation. Also problems with the initial errors and alignment. In ukf implementation the mechanization equation itself is considered for implementation. To utilize the full potential of ukf algorithm, mechanization equations explained in section 2.4.1 ((2.6), (2.7), (2.8)) need to consider. The state of the system is augmented by adding bias and scale factor errors as state variables.

5.2 Sensor Modeling

The INS sensors used in this study are of MEMS technology. The noise sources considered here in modeling are bias errors, scale factor errors and white random noise.

5.2.1 Inertial sensor error models

The measurement equation for accelerometer and gyroscope can be written as follows

$$\underline{\tilde{f}}^b = \underline{f}^b + \underline{b}_a + diag(\underline{f}^b)\, S_a + \underline{w}_a \tag{5.1}$$

$$\underline{\tilde{\omega}}^b_{ib} = \underline{\omega}^b_{ib} + \underline{b}_g + diag(\underline{\omega}^b_{ib})\, S_g + \underline{\omega}_g \tag{5.2}$$

where $\underline{\tilde{f}}^b$ is the measured accelerometer value and $\underline{f}^b$ is the true value. $\underline{\tilde{\omega}}^b_{ib}$ is the measured gyroscope value and $\underline{\omega}^b_{ib}$ is the true value. $\underline{b}_a$ and $\underline{b}_g$ are the biases of accelerometer and gyroscope respectively. S_g and S_a are scale factor errors of gyroscope and accelerometer.

The bias error of low cost MEMS sensors are sum of a constant bias (turn-on bias) plus a non constant variation (in-run bias)

$$\underline{b}(t) = \underline{b}_{tob}(t) + \delta\underline{b}(t) \tag{5.3}$$

where $\underline{b}_{tob}$ is turn on bias and $\delta\underline{b}(t)$ is the in-run bias drift. The random constant although being constant, can vary on each turn on. So the best model is the random constant.

$$\dot{\underline{b}}(t) = 0 \tag{5.4}$$

The in-run bias can be modeled as first order Gauss-Markov process.

$$\delta\dot{\underline{b}}(t) = -\frac{1}{T_b}\delta\underline{b}(t) + \underline{w}_b(t) \tag{5.5}$$

The scale factor error is usually a constant but for MEMS sensors it can vary with time so it is also modeled as first order Gauss-Markov process as

$$\delta\dot{\underline{S}}(t) = -\frac{1}{T_S}\delta\underline{S}(t) + \underline{w}_S(t) \tag{5.6}$$

The random noise is modeled as zero mean white Gaussian noise.

5.3 Measurement Equations

The equations for measurement update can be summarized as follows. Though GPS can be considered as very accurate, it is still affected by some errors. It is modeled as white Gaussian noise. Finally all the measurement equations

are given below.

$$
\begin{aligned}
\underline{\tilde{f}}^b &= \underline{f}^b + \underline{b}_a + diag(\underline{f}^b)\, S_a + \underline{w}_a \\
\underline{\tilde{\omega}}^b_{ib} &= \underline{\omega}^b_{ib} + \underline{b}_g + diag(\underline{\omega}^b_{ib})\, S_g + \underline{\omega}_g \\
\tilde{\varphi}_{gps} &= \varphi + w_\varphi \\
\tilde{\lambda}_{gps} &= \lambda + w_\lambda \\
\tilde{h}_{gps} &= h + w_h \\
\tilde{V}_{N,gps} &= V_N + w_{V_N} \\
\tilde{V}_{E,gps} &= V_E + w_{V_E} \\
\tilde{V}_{D,gps} &= V_D + w_{V_D}
\end{aligned}
\tag{5.7}
$$

The variables $\tilde{\varphi}_{gps}, \tilde{\lambda}_{gps}, \tilde{h}_{gps}, \tilde{V}_{N,gps}, \tilde{V}_{E,gps}$ *and* $\tilde{V}_{D,gps}$ are the corresponding measured values from GPS unit and w is white Gaussian noise.

Chapter 6

Implementation in Matlab

Unscented kalman filter algorithm was implemented and simulated using matlab. AeroSim an aeronautical simulation block set was utilized for generating the trajectory.

6.1 Trajectory Generator

In order to simulate the algorithms, one proper trajectory was necessary. Freely available aeronautical simulation blockset from Unmanned Dynamics (AeroSim-blockset) was employed to generate the trajectory of an aircraft. This toolbox can be downloaded for academic and non-commercial use for free of charge [13]. According to their website, AeroSim blockset is a Matlab/Simulink block library which provides components for rapid development of nonlinear 6-DOF aircraft dynamic models. In addition to aircraft dynamics the block set also includes environment models such as standard atmosphere, background wind, turbulence and earth models (geoid reference, gravity and magnetic field). This block set currently only works under windows operating system. Some of the features they claim are

- Full 6-DOF simulation of nonlinear aircraft dynamics
- Visual output to Microsoft Flight Simulator and FlightGear Flight Simulator
- Complete aircraft models that can be customized via parameter files
- Sample aircraft models including the Aerosonde UAV and the North American Navion
- Ability to automatically generate C code from Simulink aircraft models using Real-Time Workshop.

After installation of the blockset (which is explained in detail in the manual [14]) the the blocks are available in the simulink library browser. It includes different simple blocks to complete aircrafts and some demo blocks. Here the inertial navigation demo block shown in figure 6.2 is modified and extended for the implementation of thesis. A 6-DOF Aerosond UAV model (see fig. 6.1) with equations implemented in geodetic frame is used in it. Details of how this blockset can be initialized and configured is well explained in the user manual of blockset [15].

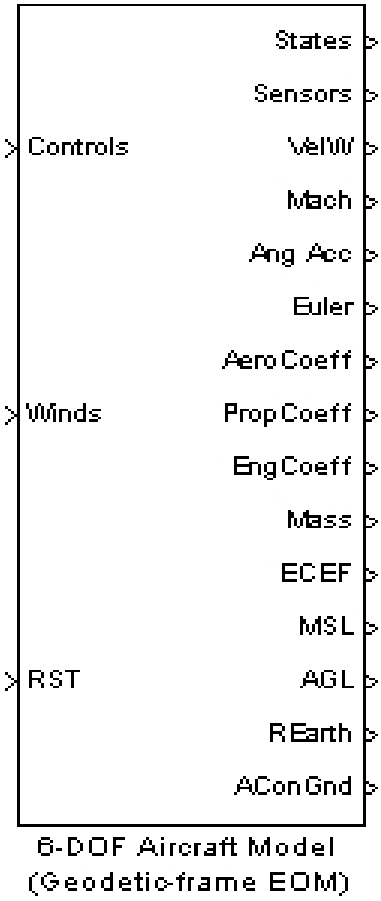

Figure 6.1: 6-DOF aircraft block used in the simulation (Aerosond UAV model), image courtesy of Unmanned Dynamics [13]

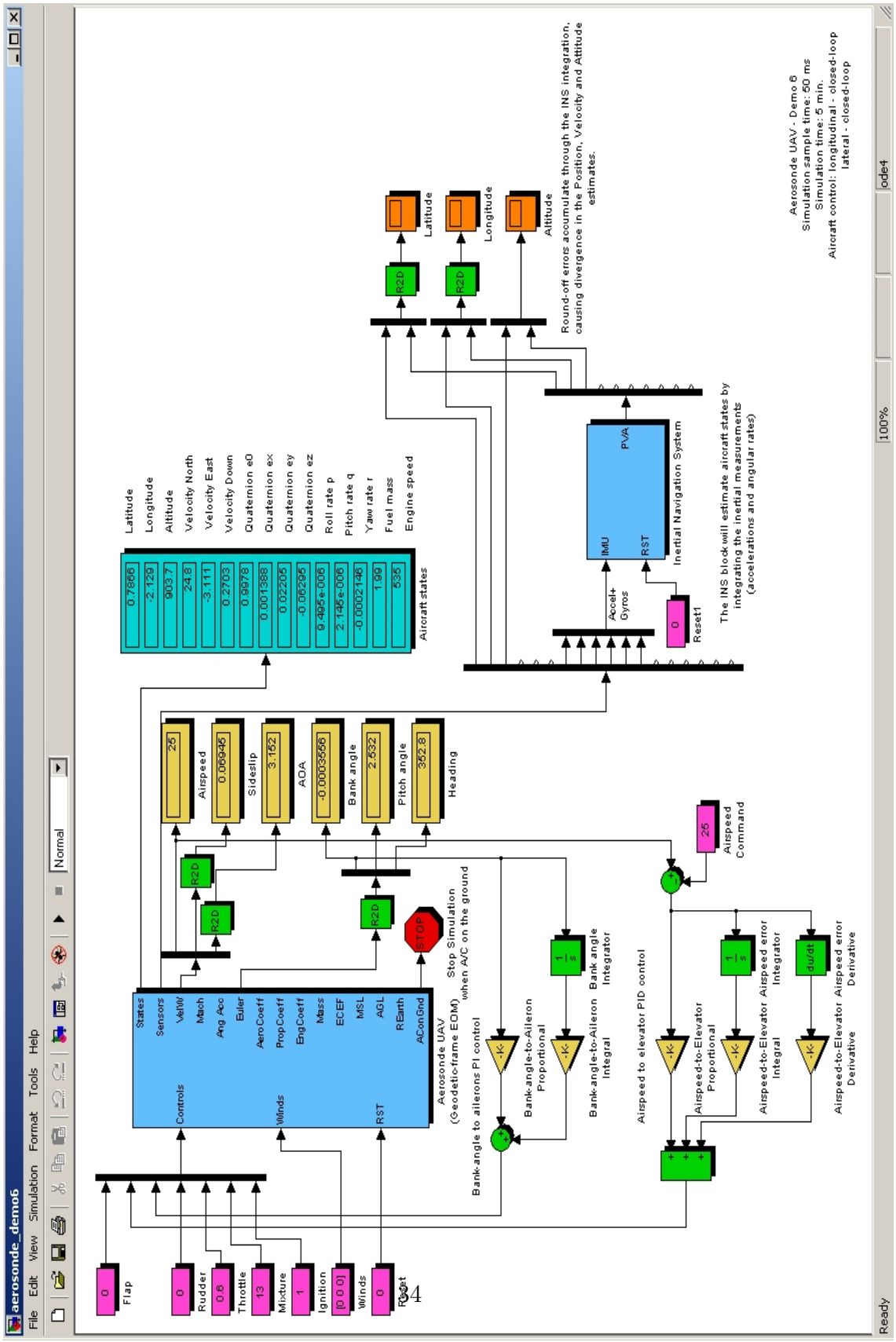

Figure 6.2: Inertial navigation demo block, image courtesy of Unmanned Dynamics [13]

6.2 Software for UKF Implementation

There are some useful packages and blocksets available from different sources for implementation of Kalman filters and smoothers. The ReBEL toolkit is one of such toolkit available from the OGI School of Science and Engineering, Oregon Health and Science University. They claim it is free for academic use. Another useful toolbox, EKF/UKF toolbox for matlab from Department of Biomedical Engineering and Computational Science (BECS), Center of Excellence in Computational Complex Systems Research, Helsinki University of Technology is available from their homepage [25]. In this thesis for implementation of ukf, the functions available from this toolbox is used in greater extend but I have modified some of the matlab files for more convenience in my implementations. Also there are a number of good examples implementations along with this blockset. Interestingly the Reentry Vehicle Tracking problem which was used (Julier and Uhlmann [18]) to demonstrate the performance of UKF is also implemented and included as demonstration.

6.3 INS-GPS Integration Implementation

The state vectors are φ, λ , h, V_N, V_E, V_D , q_0, q_1, q_2, q_3, b_{a_x}, b_{a_y}, b_{a_z}, b_{g_x}, b_{g_y}, b_{g_z}, S_{a_x}, S_{a_y}, S_{a_z}, S_{g_x}, S_{g_y}, S_{g_z} . There are 22 state variables in this proposed implementation. They are composed of position, velocity, quaternion, bias of sensors and scale factor of sensors as noted above. Non-augmented ukf implementation is performed here. Suitable initial conditions are set for the states. Other variables needed for the operation of the blocks are set using initialization file. In order to generate the trajectory and sensor data, simulink Aerosond UAV model is simulated with suitable initial conditions. To choose a particular trajectory the control inputs (rudder, throttle etc.) and environmental constraints are applied to the model. The UAV model outputs the sensor data which is used in later stage for filter implementation and comparison. One set of measurement data is passed to the unaided INS module which will generate the uncompensated output. Other set of data is passed over to the aided INS block and ukf filter implementation for estimation. Figure 6.3 shows the ins-gps integration implementation and figure 6.5 shows in more detail about how estimated values are applied in correction of navigation information. Figure 6.4 shows the INS error modeling in simulink.

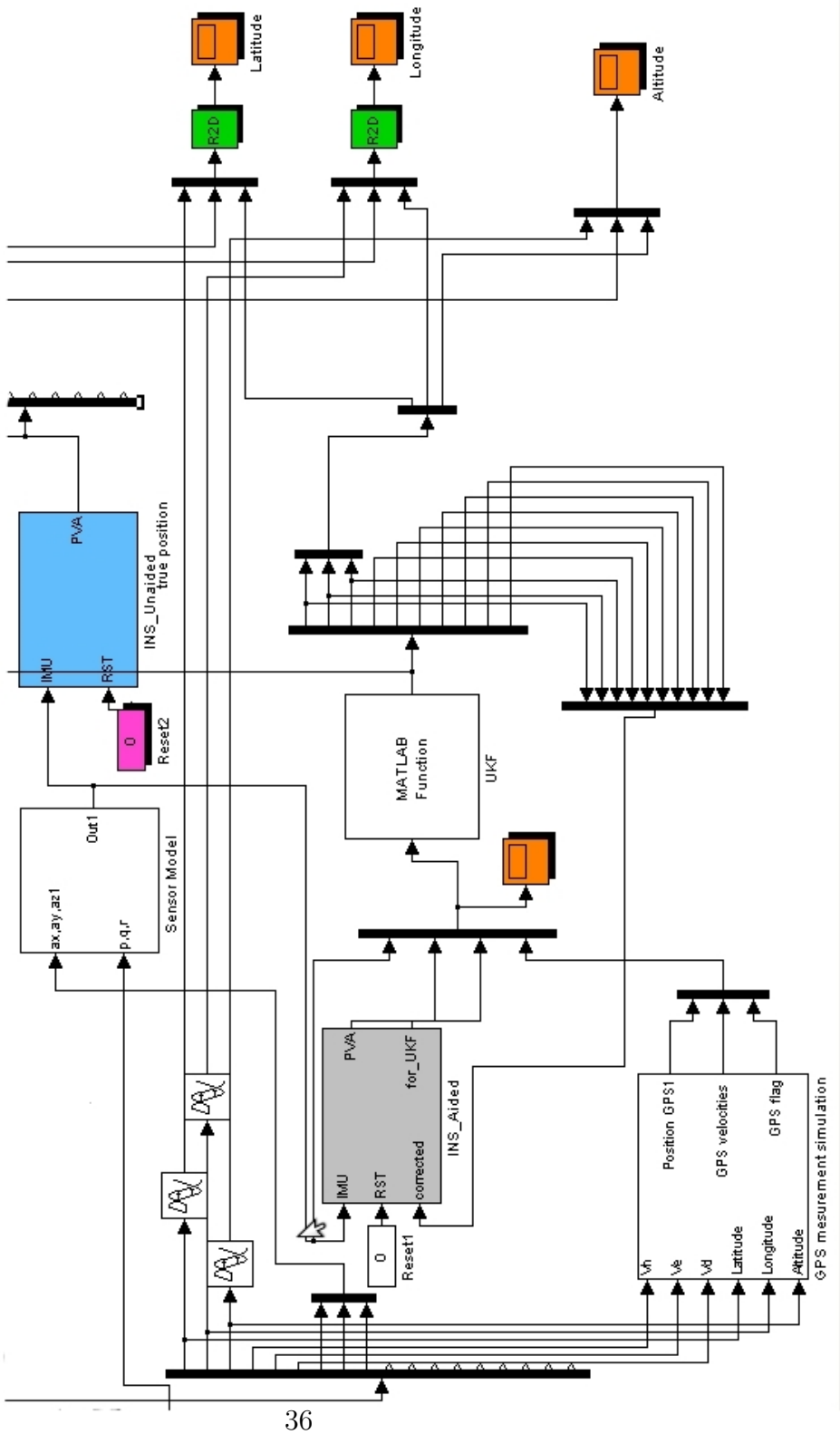

Figure 6.3: INS-GPS integration (only the relevant part is shown)

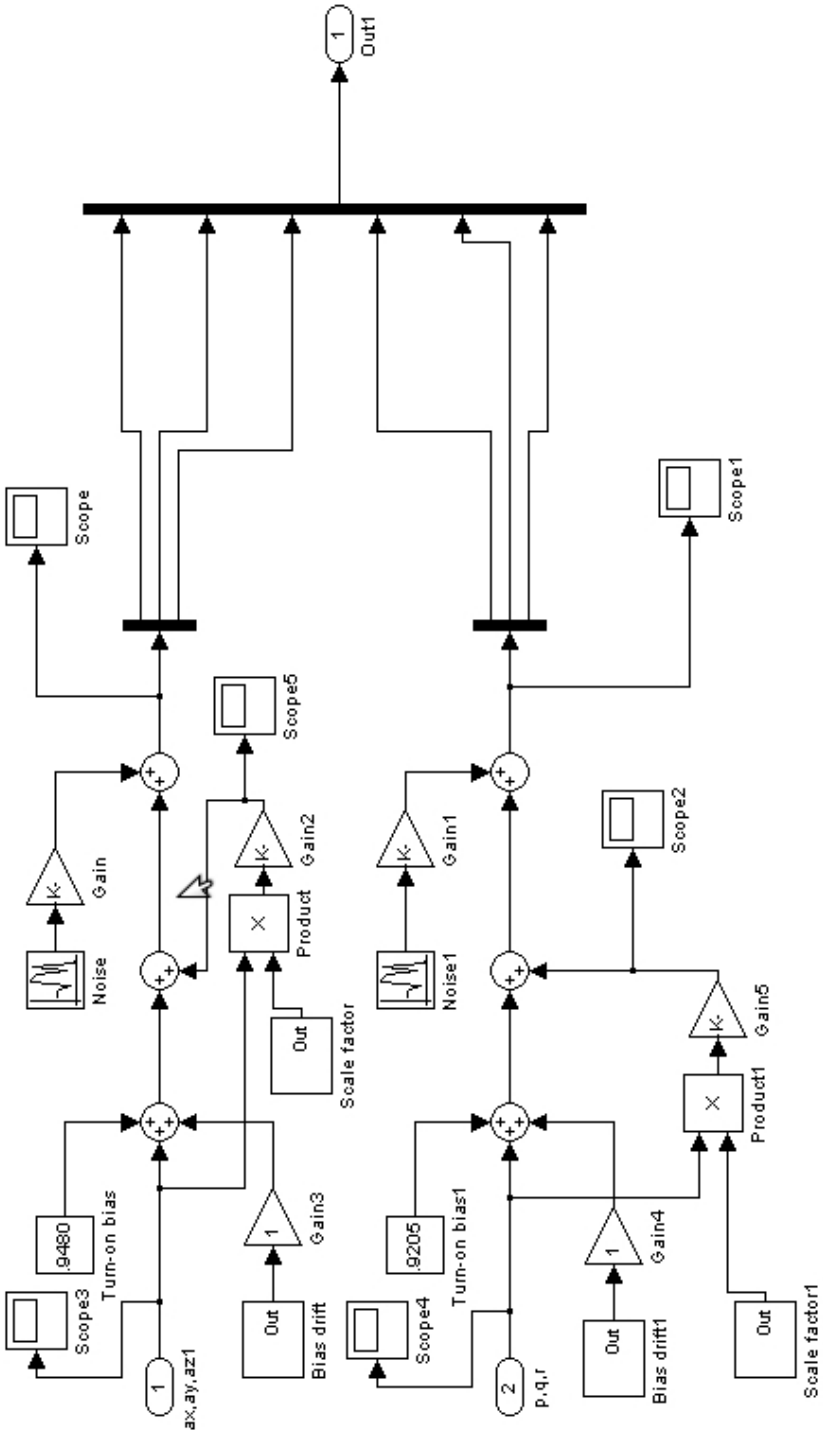

Figure 6.4: INS error model

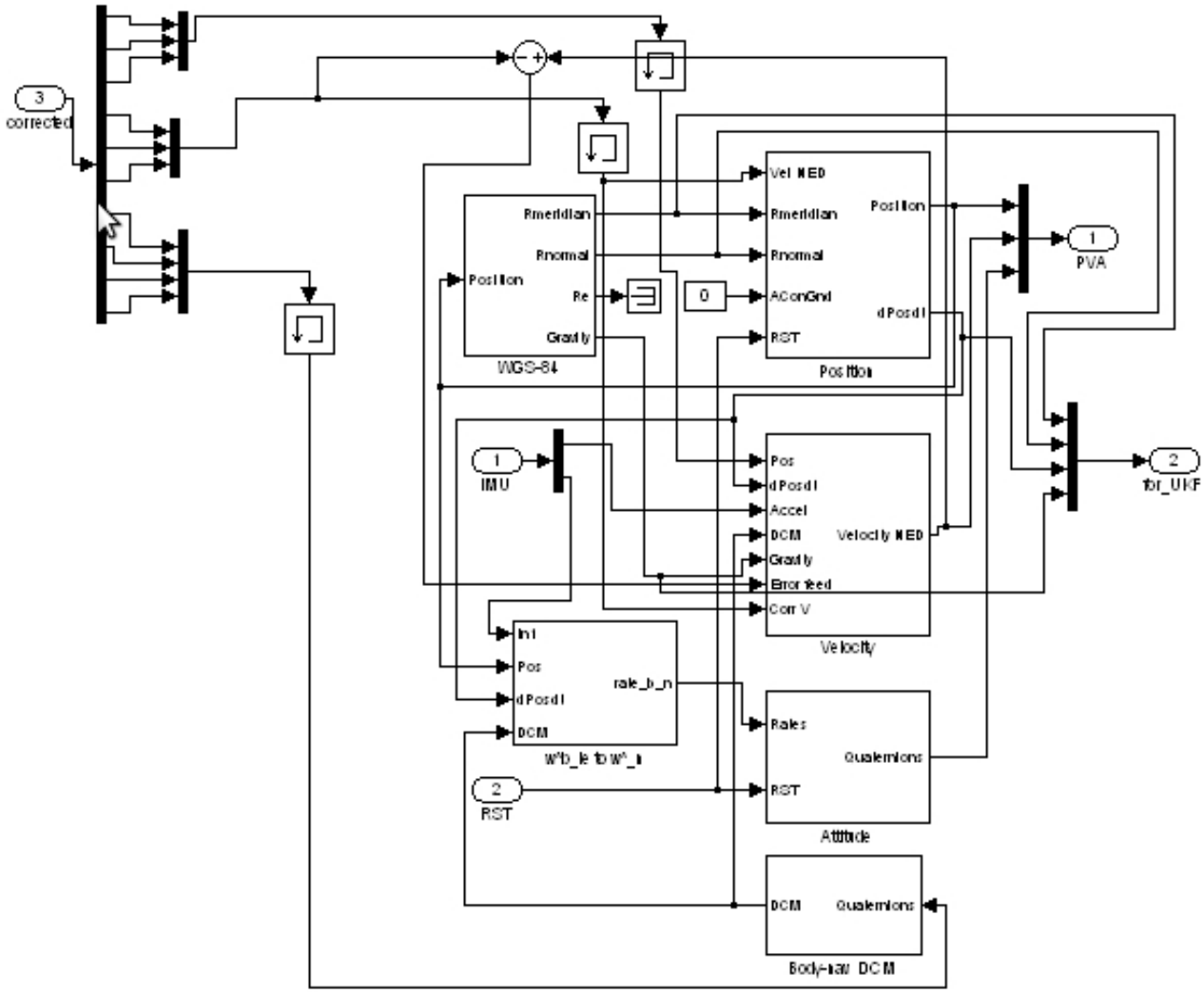

Figure 6.5: INS corrected using estimated information

In the original implementation in Aerosim blockset, the sensor output from gyroscope was the rate of the body with respect to the navigation frame (ω_n^b), but the gyroscope measurement is with respect to inertial frame. INS block was implemented to use this value to generate the navigation information. So model is modified to get the correct gyro output and then INS block is also modified according to that. Figure 6.7 shows the modification and figure 6.6 shows the original INS implementation.

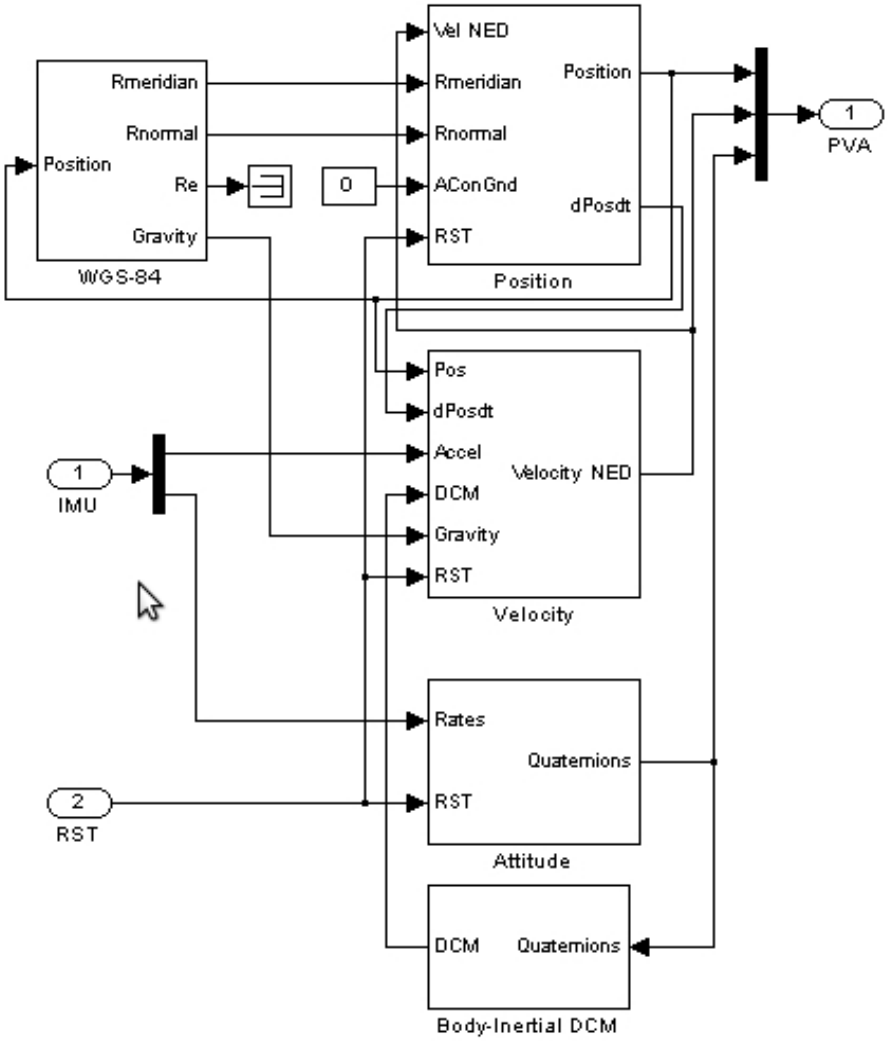

Figure 6.6: INS block, aerosim

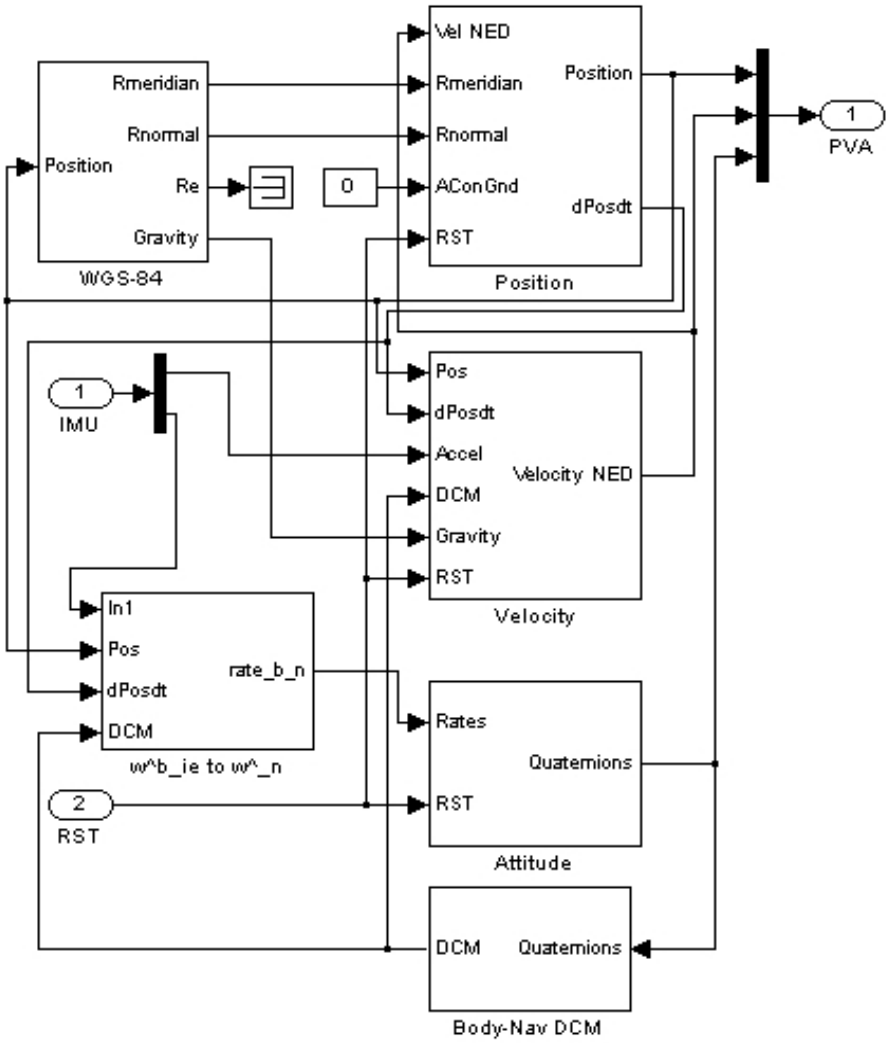

Figure 6.7: INS block, modified

6.4 Attitude Estimation Implementation

An attitude estimation problem explained in [1] (of a stabilized platform) was considered finally due to the difficulties faced with INS-GPS integration and also due to time constraints. The main difficulties were the instability of cholesky factorization implementation "chol" and availability of trajectory generator. Sensors used in this implementation are 3-axis accelerometer, gyroscope and magnetometer. Here is a brief problem description and equations used in the filtering implementation.

6.4.1 Kinematic Equation in Quaternion

The kinematic equation of rigid body motion is derived as follows in the literature [3] and is widely used in digital navigation processing. Analogous to the euler kinematic equation, let F_r be a reference frame and F_b be a rotating frame and orientation at any time 't' can be represented in quaternion as $q_{b|r}(t)$. Let the instantaneous angular velocity of F_b along a unit vector $\hat{k}$ be w, then in δt interval $\delta q_{b|r}$ can be approximated as

$$\delta q_{b|r}(\delta t) \approx \begin{bmatrix} 1 \\ \hat{k}\ w\delta t/2 \end{bmatrix} \tag{6.1}$$

and finally the derivative of quaternion is obtained as

$$\dot{q} = \frac{1}{2} q_{b|r} * w^b_{b|r} \tag{6.2}$$

Where '*' is quaternion multiplication and $w^b_{b|r}$ is the angular velocity of b frame with respect to r frame expressed in b frame. And in matrix form

$$\begin{bmatrix} \dot{q}_0 \\ \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} = 1/2 \begin{bmatrix} 0 & -P & -Q & -R \\ P & 0 & R & -Q \\ Q & -R & 0 & P \\ R & -Q & -P & 0 \end{bmatrix} \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \tag{6.3}$$

Here P, Q and R are angular velocities in body axis and q_0, q_1, q_2, q_3 are components of quaternion. These equations are the basics of attitude estimation implementation in digital navigation aids.

6.4.2 Sensor Modeling

For attitude estimation of stabilized platform, the sensors used are 3-axis accelerometers, gyroscope and magnetometer. The sensor errors considered

are temperature drift, CG offset, bias and random noise. The bias errors (both in accelerometer, gyroscope and magnetometer) are considered as state variables. The accelerometer is modeled as

$$\begin{bmatrix} A_x \\ A_y \\ A_z \end{bmatrix}_{measured} = \begin{bmatrix} A_x \\ A_y \\ A_z \end{bmatrix}_{true} + \begin{bmatrix} -\ 0.0056T\ +\ 0.043 \\ 0.00056\ -\ 0.0023T\ +\ 0.000143 \\ 0.034\ -\ 0.0096T\ +\ 0.0073 \end{bmatrix} + \begin{bmatrix} -(q^2+r^2)X_{A_x} \\ -(p^2+r^2)X_{A_y} \\ -(p^2+q^2)X_{A_z} \end{bmatrix} + \begin{bmatrix} B_{A_x} \\ B_{A_y} \\ B_{A_z} \end{bmatrix} \tag{6.4}$$

And gyroscope is modeled as follows

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix}_{measured} = \begin{bmatrix} p \\ q \\ r \end{bmatrix}_{true} + \begin{bmatrix} -\ 0.0065T^3\ +\ 0.0045T^2\ -\ 0.0026T\ +\ 0.053 \\ 0.00056T^2\ -\ 0.000823T\ +\ 0.000543 \\ 0.000037T^3\ +\ 0.014T^2\ -\ 0.0094T\ +\ 0.000373 \end{bmatrix} + \begin{bmatrix} B_{p_x} \\ B_{q_y} \\ B_{r_z} \end{bmatrix} \tag{6.5}$$

Where

$$\begin{bmatrix} -(q^2+r^2)X_{A_x} \\ -(p^2+r^2)X_{A_y} \\ -(p^2+q^2)X_{A_z} \end{bmatrix} \tag{6.6}$$

accounts for the CG offset error. A_x, A_y and A_z are the accelerometer values. p, q and r are the gyroscope values and they are angular rates in body frame. X_{A_x}, X_{A_y} and X_{A_z} are the CG offset of accelerometer. T is temperature and B is corresponding bias value. The temperature drift is not estimated but just corrected before the estimation and in real situation the temperature sensor is used to get the accurate temperature which will be used for above temperature drift calibration. Constant bias error is considered in all sensors and the random noise is white Gaussian noise.

6.4.3 Matlab Implementation

For simulation, the sensor measurement data was generated using simulink blocks. The simulink blocks are shown in figures 6.8, 6.9 and 6.10. The state model consists of 15 state variables including four quaternion and the bias of each sensors. Quaternion was used in the implementations because of its advantages over euler angles.

The state model is

$$
\begin{aligned}
\dot{q}_0 &= \frac{1}{2}(-q_1\,p - q_2\,q - q_3\,r) \\
\dot{q}_1 &= \frac{1}{2}(-q_0\,p + q_2\,r - q_3\,q) \\
\dot{q}_2 &= \frac{1}{2}(-q_0\,q - q_1\,r + q_3\,p) \\
\dot{q}_3 &= \frac{1}{2}(-q_0\,r + q_1\,q - q_2\,p) \\
\dot{p} &= -\frac{1}{\tau}\,p \\
\dot{q} &= -\frac{1}{\tau}\,q \\
\dot{r} &= -\frac{1}{\tau}\,r \\
\dot{B}_p &= 0 \\
\dot{B}_q &= 0 \\
\dot{B}_r &= 0 \\
\dot{B}_{A_x} &= 0 \\
\dot{B}_{A_y} &= 0 \\
\dot{B}_{A_z} &= 0 \\
\dot{B}_{H_x} &= 0 \\
\dot{B}_{H_y} &= 0
\end{aligned}
\tag{6.7}
$$

where τ is time constant

The measurement model is

$$
\begin{aligned}
1 &= (q_0^2 + q_1^2 + q_2^2 + q_3^2) \\
A_{x,m} &= A_{x_1} + B_{A_x} + w_{A_x} \\
A_{y,m} &= A_{y_1} + B_{A_y} + w_{A_y} \\
A_{z,m} &= A_{z_1} + B_{A_z} + w_{A_z} \\
H_{x,m} &= H_{b_x} + B_{H_x} + w_{H_x} \\
H_{y,m} &= H_{b_y} + B_{H_y} + w_{H_y}
\end{aligned}
\tag{6.8}
$$

where

$$\begin{bmatrix} A_{x_1} \\ A_{y_1} \\ A_{z_1} \end{bmatrix} = \begin{bmatrix} A_{b_x} \\ A_{b_y} \\ A_{b_z} \end{bmatrix} - \begin{bmatrix} (q^2 + r^2)\, X_{A_x} \\ (p^2 + r^2)\, X_{A_y} \\ (p^2 + q^2)\, X_{A_z} \end{bmatrix} - \begin{bmatrix} -\ 0.0056T\ +\ 0.043 \\ 0.00056\ -\ 0.0023T\ +\ 0.000143 \\ 0.034\ -\ 0.0096T\ +\ 0.0073 \end{bmatrix} \tag{6.9}$$

and

$$\begin{bmatrix} A_{b_x} \\ A_{b_y} \\ A_{b_z} \end{bmatrix} = \begin{bmatrix} (q_0^2 + q_1^2 - q_2^2 - q_3^2) & 2(q_1q_2 + q_0q_3) & 2(q_1q_3 - q_0q_2) \\ 2(q_1q_2 - q_0q_3) & (q_0^2 - q_1^2 + q_2^2 - q_3^2) & 2(q_2q_3 + q_0q_1) \\ 2(q_1q_3 + q_0q_2) & 2(q_2q_3 - q_0q_1) & (q_0^2 - q_1^2 - q_2^2 + q_3^2) \end{bmatrix} \begin{bmatrix} A_{g_x} \\ A_{g_y} \\ A_{g_z} \end{bmatrix} \tag{6.10}$$

$$\begin{bmatrix} H_{b_x} \\ H_{b_y} \end{bmatrix} = \begin{bmatrix} (q_0^2 + q_1^2 - q_2^2 - q_3^2) & 2(q_1q_2 + q_0q_3) & 2(q_1q_3 - q_0q_2) \\ 2(q_1q_2 - q_0q_3) & (q_0^2 - q_1^2 + q_2^2 - q_3^2) & 2(q_2q_3 + q_0q_1) \\ 2(q_1q_3 + q_0q_2) & 2(q_2q_3 - q_0q_1) & (q_0^2 - q_1^2 - q_2^2 + q_3^2) \end{bmatrix} \begin{bmatrix} H_{g_x} \\ H_{g_y} \\ H_{g_z} \end{bmatrix} \tag{6.11}$$

where A_{g_x}, A_{g_y} and A_{g_z} are the gravitational acceleration along geodetic frame (eg. [0 0 -9.81] m/s^2) at the location. H_{g_x}, H_{g_y} and H_{g_z} are earth magnetic vector along geodetic frame (eg. [18 0 45] μ tesla) at the location. B is the bias of corresponding sensors.

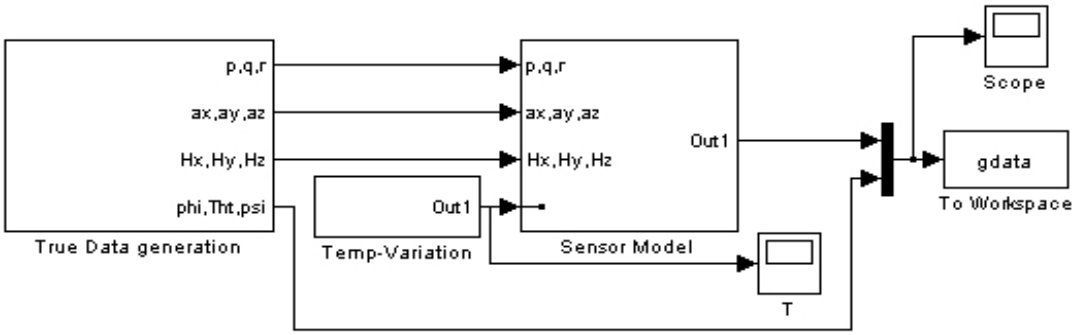

Figure 6.8: Data generation simulink block, attitude estimation

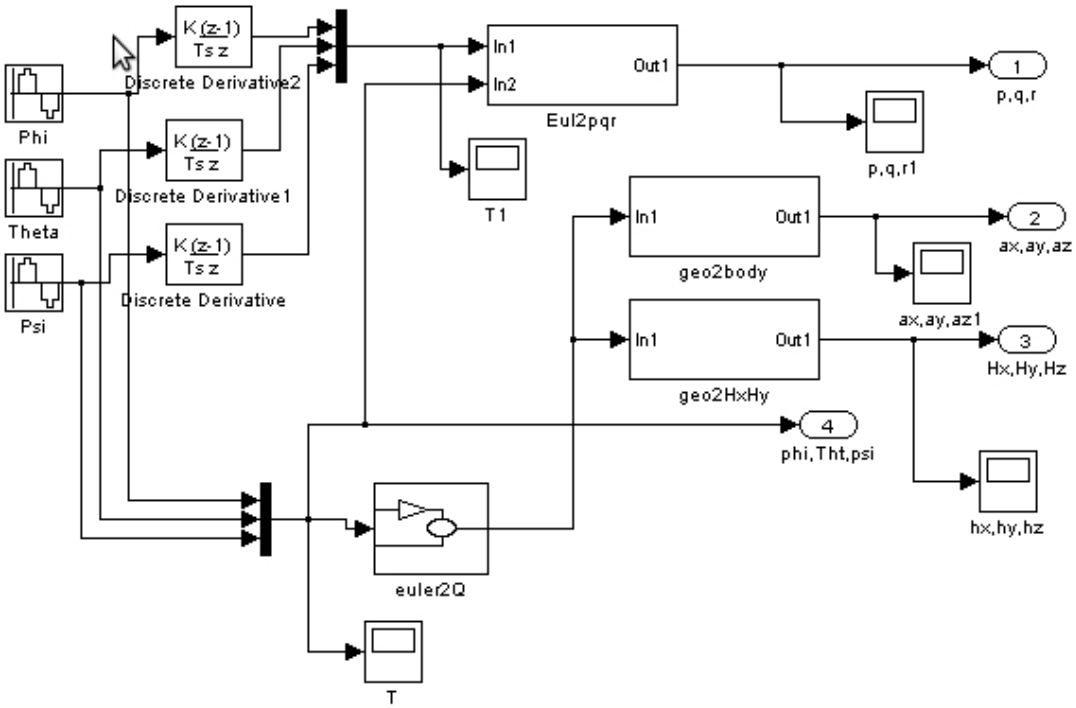

Figure 6.9: Data generation in detail

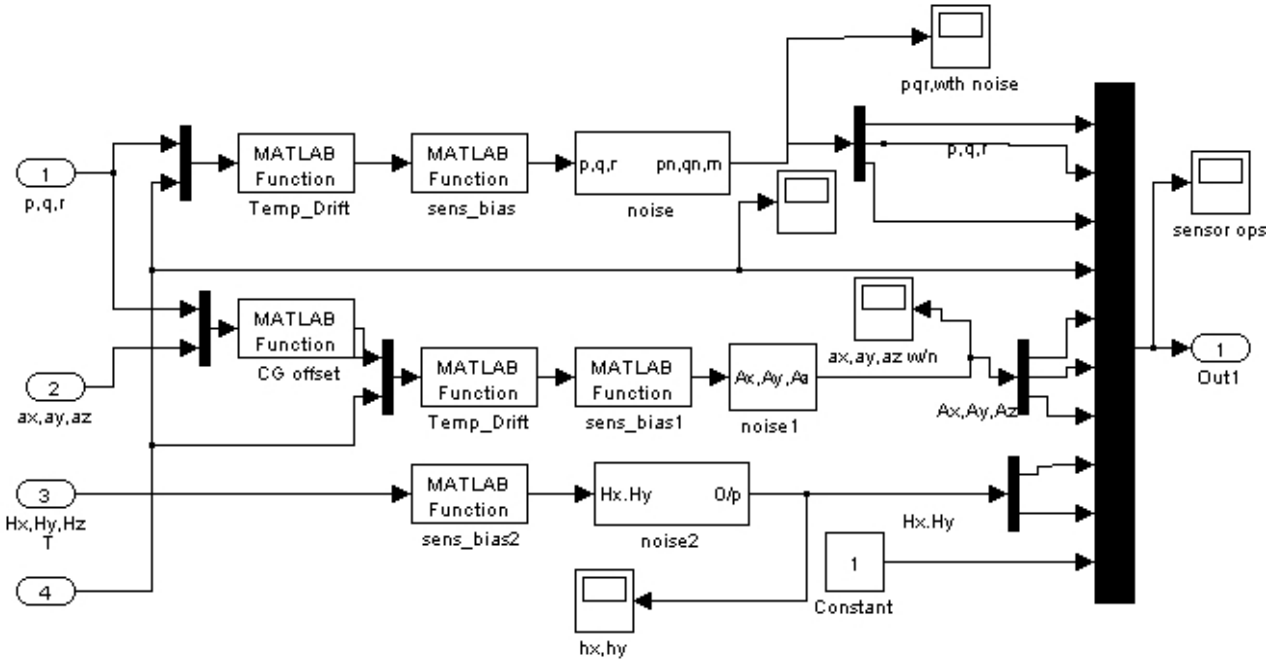

Figure 6.10: Sensor modeling

Chapter 7

Results

The main objectives of the thesis were to compare the computational requirements of both algorithm and then the performance of the filter with different initial errors. Ins-gps integration implementation faced different difficulties and needed more time, so an attitude estimation problem was finally considered. Please note that the objectives of the thesis were achieved with this attitude estimation problem. Both extended and unscented implementation were done using matlabR2007b in a Lenovo 32-bit Dual Core laptop. System configuration is given in table 7.1.

The main difficulty faced in tuning ukf filter was the instability of 'chol' decomposition which was used as an approximation for calculating square root of P matrix. When if the R and Q matrix are not chosen properly, the algorithm will fail in the first step or after a few iterations. This made it difficult to tune the filter. By observation the step where the instability occurs was found to be at (4.23) in time update where process covariance prediction is done. This is the step where the P matrix becomes negative definite for the first time. Below is the detailed discussion of the results obtained from implementations.

System Configuration	
Operating System	Ubuntu Release 9.04, Kernel Linux 2.6.28-11-generic, GNOME 2.26.1
Hardware Configuration	Dual Core Processor, Processor 0: Genuine Intel(R) CPU T2080@1.73GHz, Processor 1: Genuine Intel(R) CPU T2080@1.73GHz
Software Tool used for simulation	Matlab R2007b

Table 7.1: System Configuration for Simulations

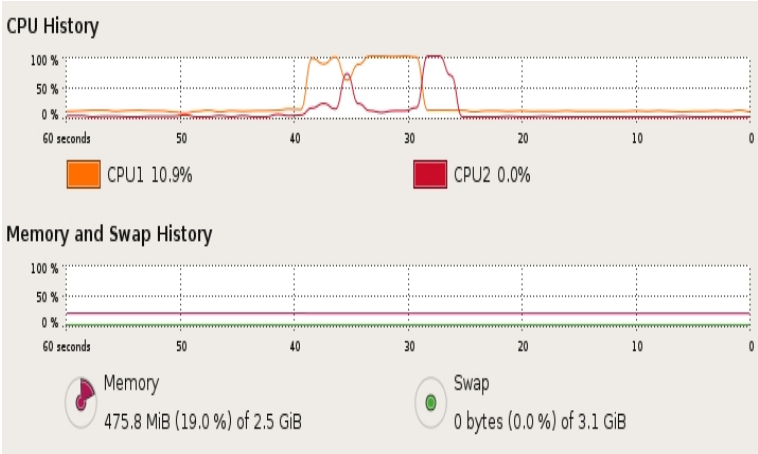

Figure 7.1: Snap of system monitor during EKF algorithm execution

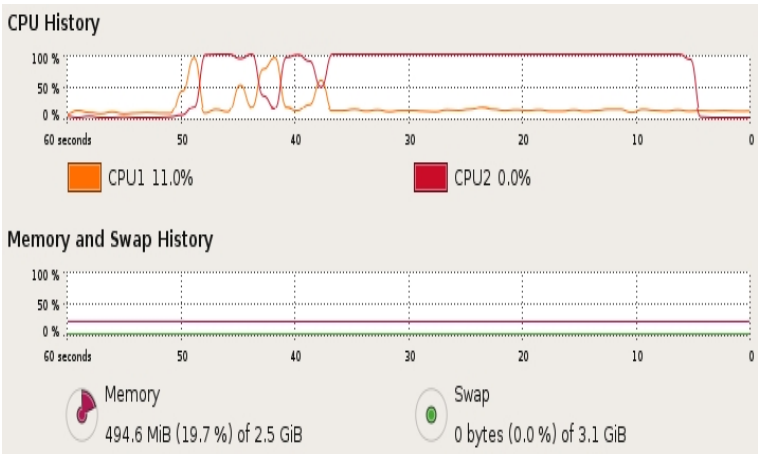

Figure 7.2: Snap of system monitor during UKF algorithm execution

CPU time used		
	EKF	UKF
CPU time used for processing of data (3500 data samples)	.77s	31.64s
CPU time used for single time update step	$2.59 \times 10^{-04} s$	$56 \times 10^{-04} s$
CPU time used for single measurement update step	$4.42 \times 10^{-04} s$	$52 \times 10^{-04} s$

Table 7.2: Comparison of computational resources utilized

7.1 UKF Vs EKF

In order to compare the performance of the ukf and ekf, estimation of attitudes (roll, pitch and yaw) of stabilized platform in both implementations were considered. Different aspects of filters were considered, such as sum of squares of difference of filtered values to the true attitude values, system resources like CPU time used and performance with initial errors. These results are summarized in tables.

Table 7.2 compares the CPU time used by both implementations. Also figures 7.1 and 7.2 are the snaps of the system monitor window during execution of ekf and ukf implementations respectively. The CPU time used by ekf and ukf are found to be .77 seconds and 31.64 seconds respectively. For a single prediction step in ekf and ukf, the average time used was 2.59×10^{-04} and 56×10^{-04} seconds respectively. Where as the measurement update took 4.42×10^{-04} and 52×10^{-04} seconds respectively. Table 7.3 gives the sum of squares comparison of both ukf and ekf implementations. The figures 7.3, 7.4 and 7.5 are from the corresponding ekf implementation results and figures 7.6, 7.7 and 7.8 are from ukf implementation results. Though it is not clear from the graphs, when considering the sum of squares of first 500 data samples, it is clear that ukf has less variation from reality in the beginning. It is also clear from the figures 7.5 and 7.8 that estimation of psi angle is worse in ekf implementation but better in ukf.

Sum of Squares of Difference		
Attitude angle	EKF	UKF
Phi	36.86	.1926
Theta	15.04	.1939
Psi	465.75	.1833
After 500 data samples(filter stabilized)		
Phi	0.0022	.1635
Theta	0.0021	.1739
Psi	63.57	.1568

Table 7.3: Estimation performance comparison

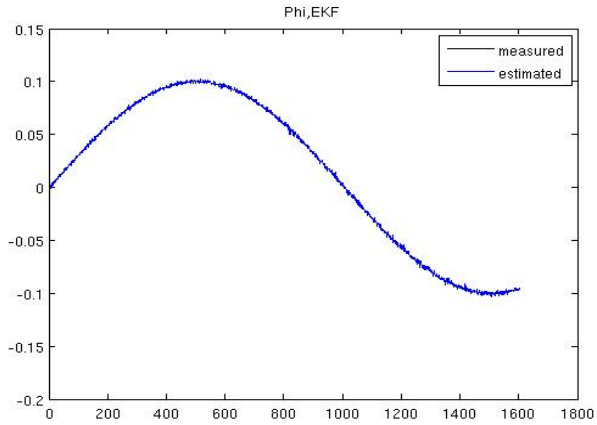

Figure 7.3: phi angle, EKF

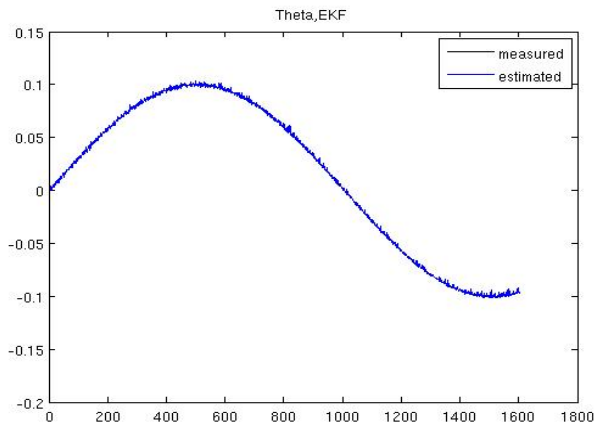

Figure 7.4: theta angle, EKF

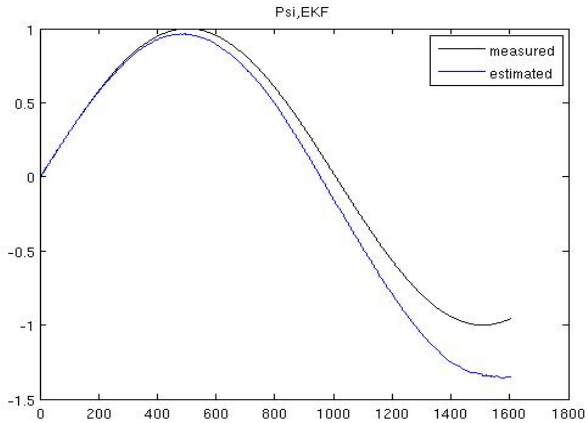

Figure 7.5: psi angle, EKF

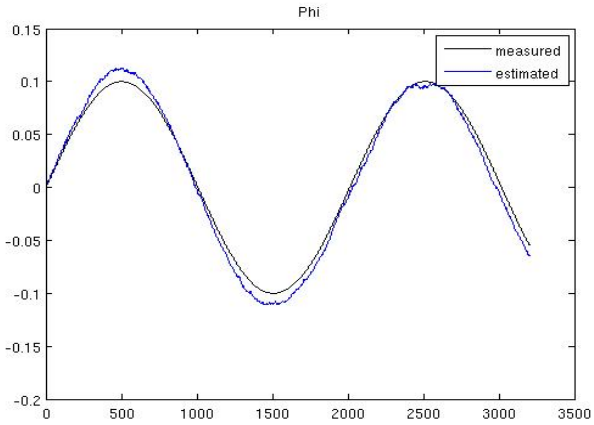

Figure 7.6: phi angle, UKF

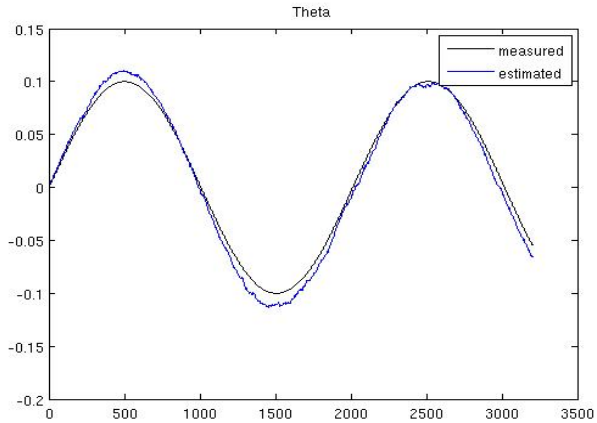

Figure 7.7: theta angle, UKF

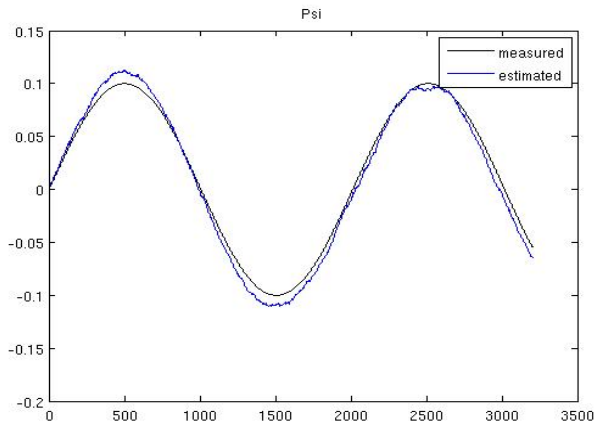

Figure 7.8: psi angle, UKF

In order to test the performance of the filter with different initial errors, three different set of initial conditions are applied with more variation from first set to third set. The results of sum of squares of difference are presented as tables 7.4 and 7.5 for ekf and ukf respectively. The values for first 200 data samples and complete samples are separately shown to understand the performance in the initial stage of estimation. It is seen that the ukf attain stability before ekf when the initial conditions are more away from reality. Also estimation of psi angle (see figure 7.9) in ekf was degraded considerably with increasing initial errors but ukf has a good result (see figure 7.10). The figures 7.11 and 7.12 are results of ekf implementation with second set (medium errors) of initial values. The figures 7.13 and 7.14 are results of ekf implementation with third set (larger errors) of initial values. It is clear form comparing theses two graphs that as initial error increase, ekf estimation takes more time to stabilize. The figure 7.15 is result of ukf implementation with third set (larger errors) of initial values and it is not affected much by the initial errors. In order to enhance the understanding of results, residuals (difference between actual and estimated values) are plotted for simulations with different set of initial errors. Figures 7.16, 7.17 and 7.18 are residual plots with smaller initial errors. Figures 7.19, 7.20 and 7.21 are residual plots with medium initial errors. Figures 7.22, 7.23 and 7.24 are residual plots with larger initial errors. The ekf has more variations in the beginning, but both algorithms have similar performance in a long run.

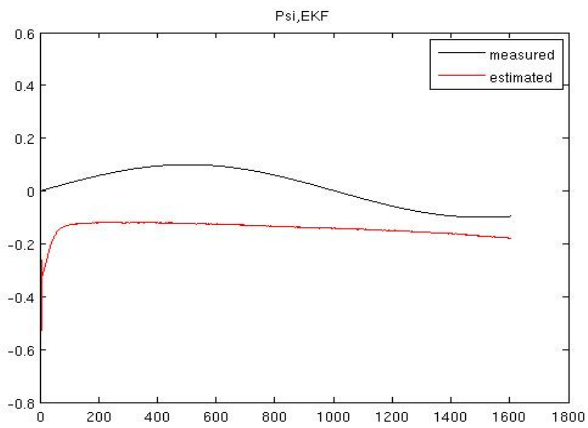

Figure 7.9: psi angle, ekf with third set (larger errors) of initial values

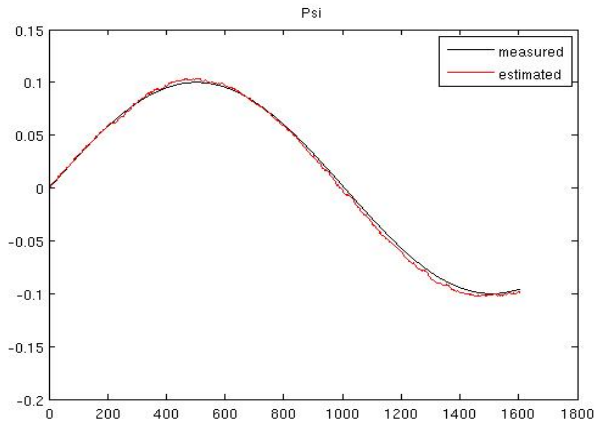

Figure 7.10: psi angle, ukf with third set (larger errors) of initial values

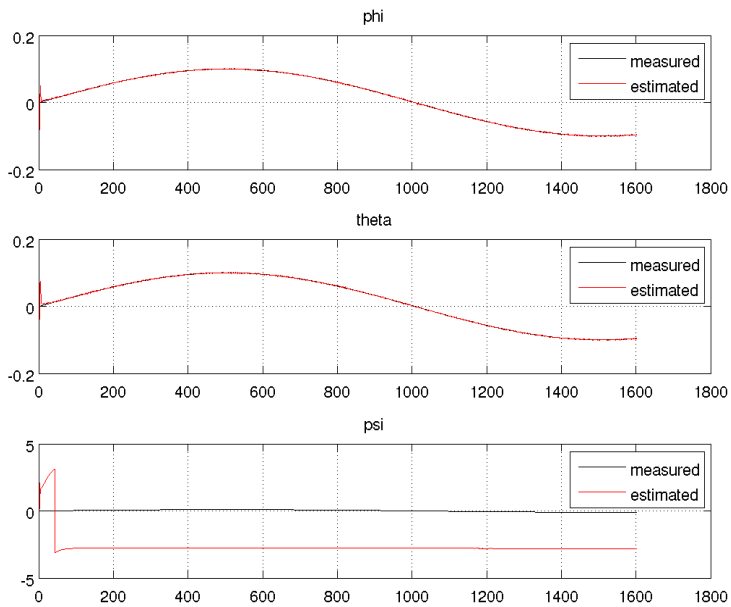

Figure 7.11: ekf results with second set (medium errors) of initial values

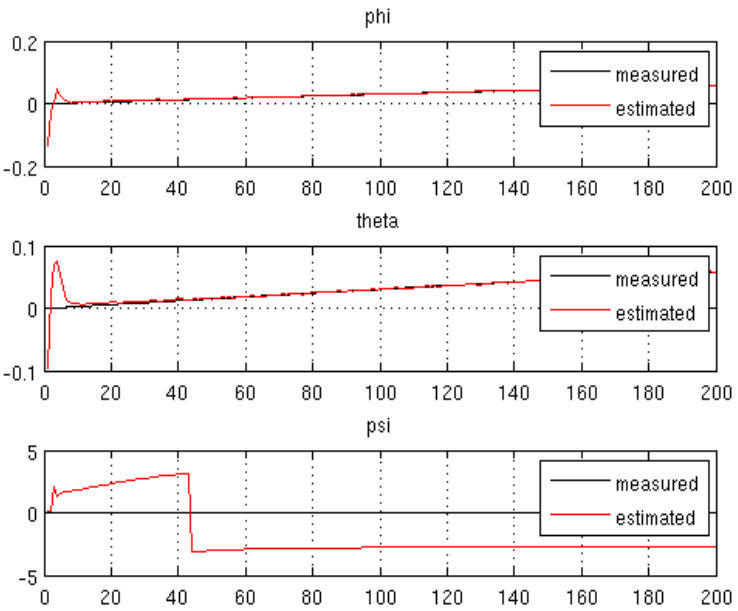

Figure 7.12: First 200 data samples of ekf results with second set (medium errors) of initial values

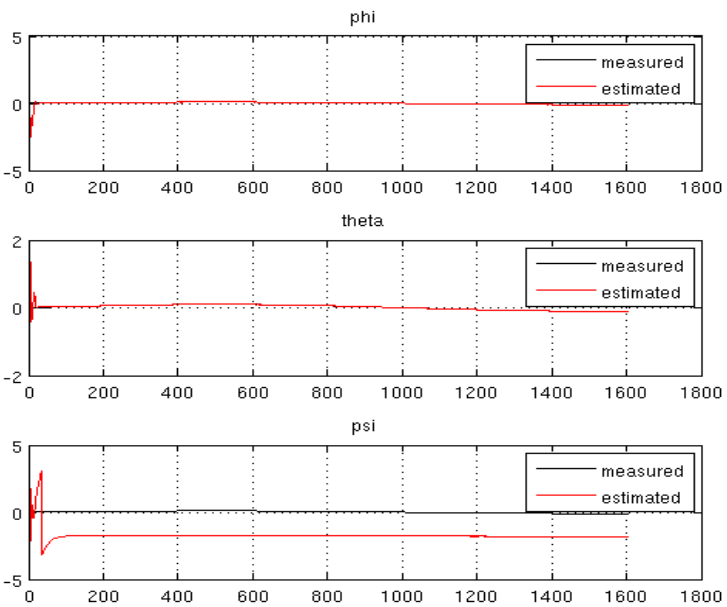

Figure 7.13: ekf results with third set (larger errors) of initial values

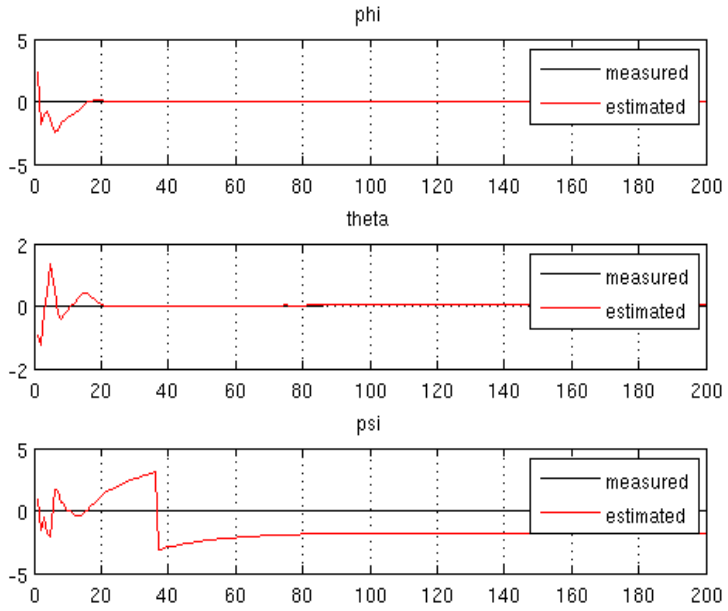

Figure 7.14: First 200 data samples of ekf results with third set (larger errors) of initial values

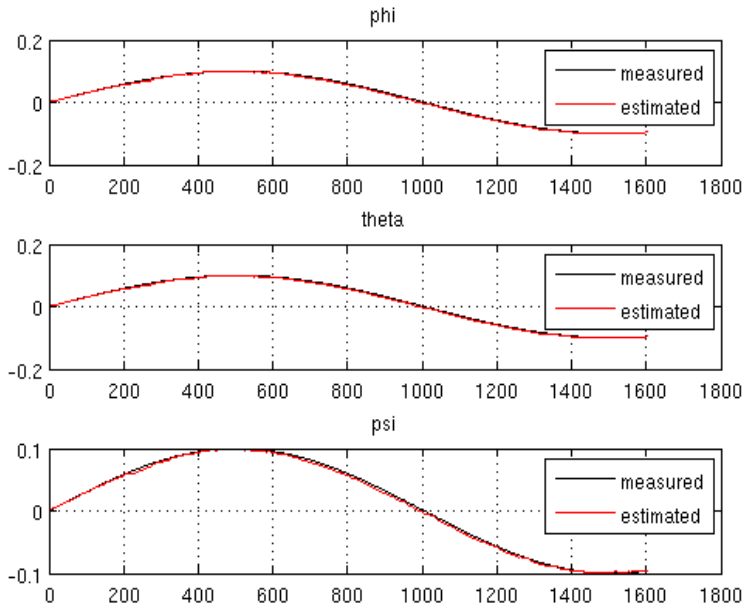

Figure 7.15: ukf results with third set (larger errors) of initial values

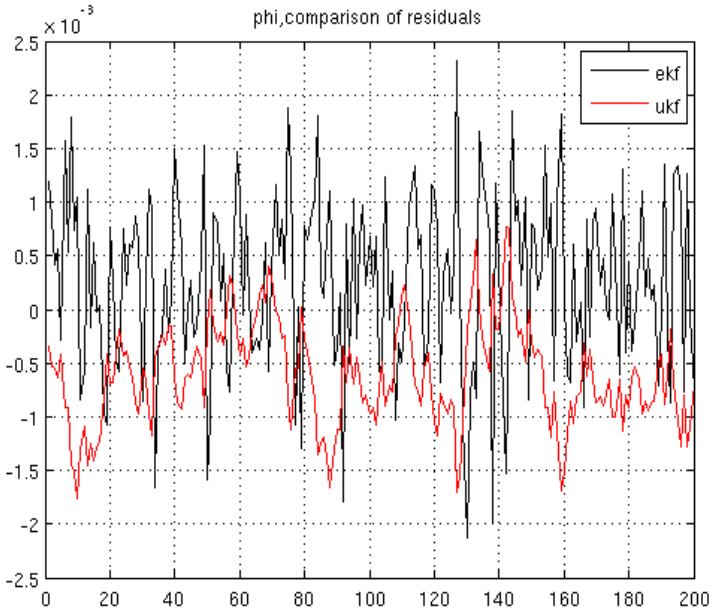

Figure 7.16: Residual of phi angle estimate with small initial errors (first 200 data samples)

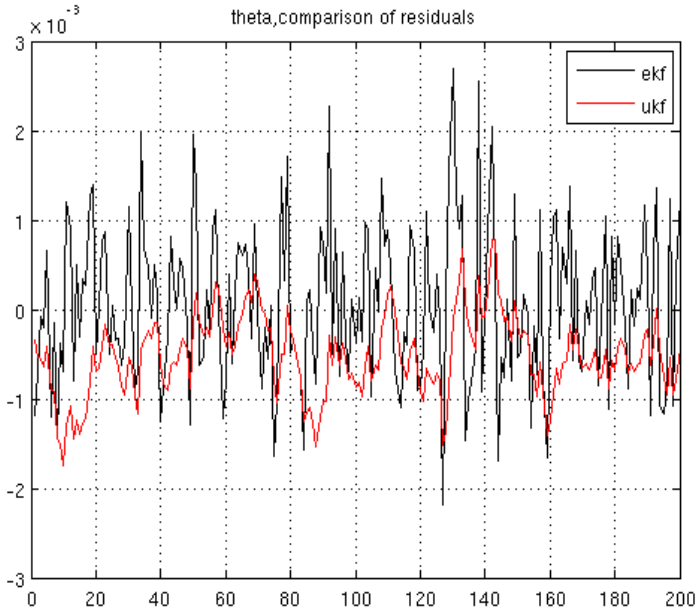

Figure 7.17: Residual of theta angle estimate with small initial errors (first 200 data samples)

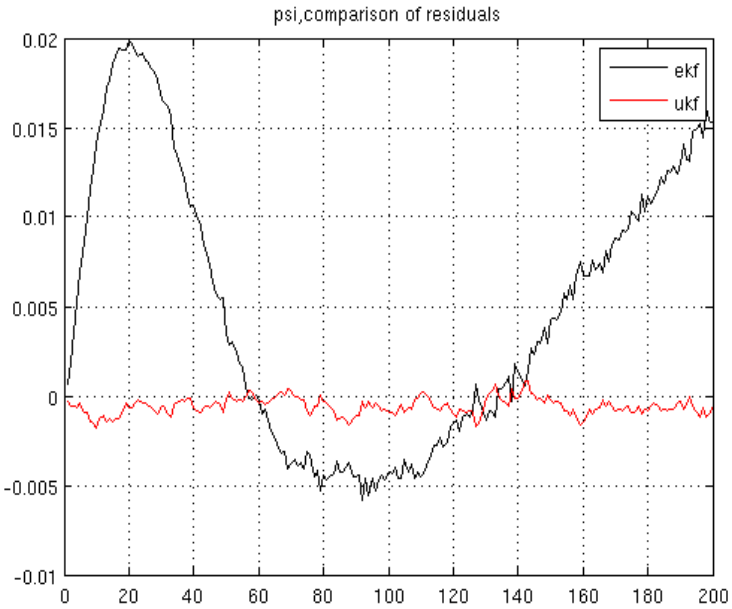

Figure 7.18: Residual of psi angle estimate with small initial errors (first 200 data samples)

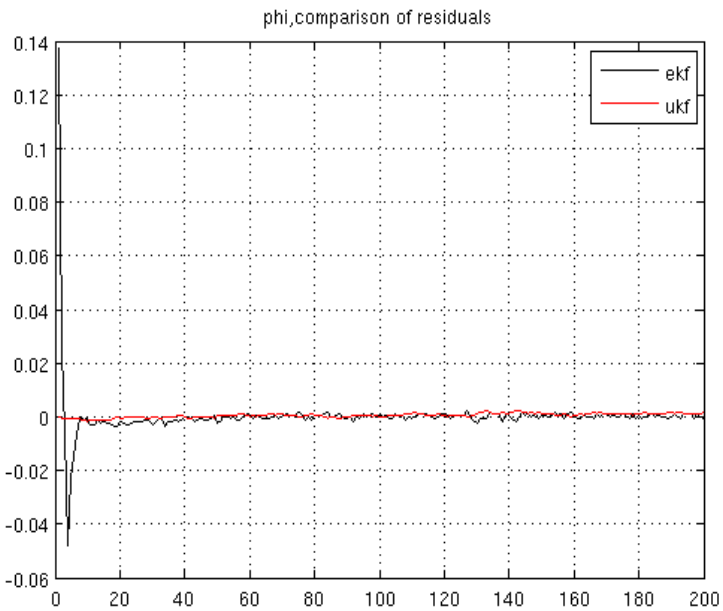

Figure 7.19: Residual of phi angle estimate with medium initial errors (first 200 data samples)

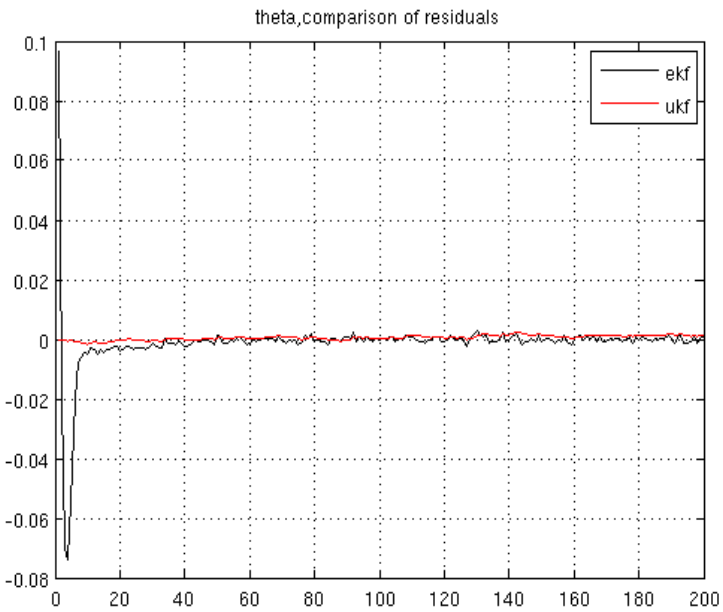

Figure 7.20: Residual of theta angle estimate with medium initial errors (first 200 data samples)

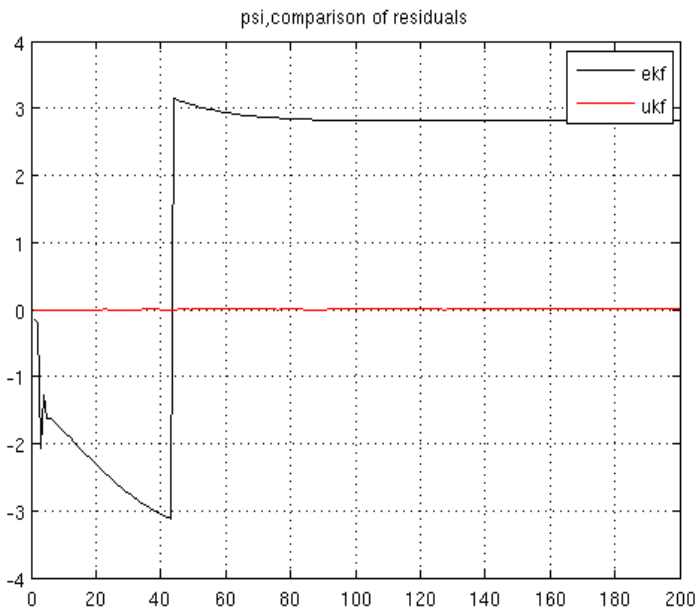

Figure 7.21: Residual of psi angle estimate with medium initial errors (first 200 data samples)

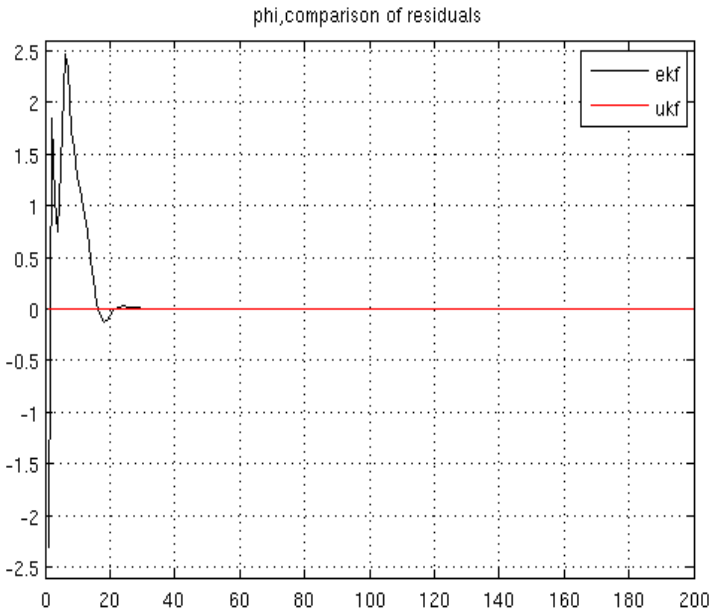

Figure 7.22: Residual of phi angle estimate with larger initial errors (first 200 data samples)

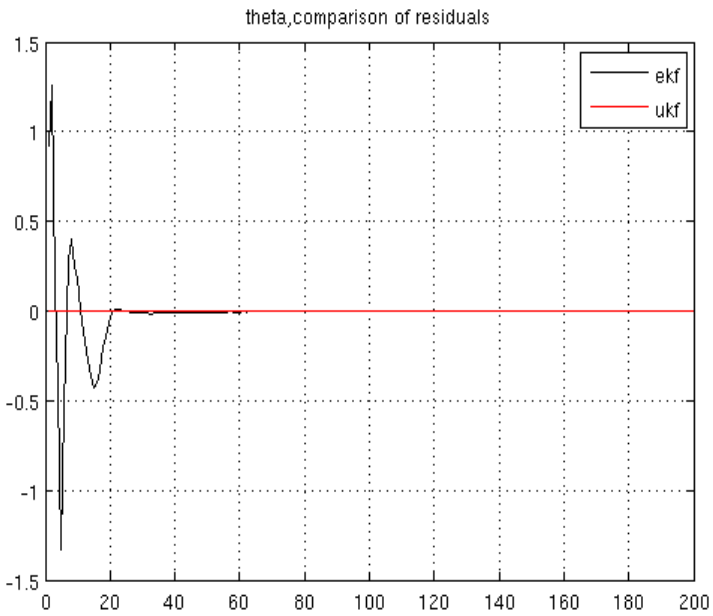

Figure 7.23: Residual of theta angle estimate with larger initial errors (first 200 data samples)

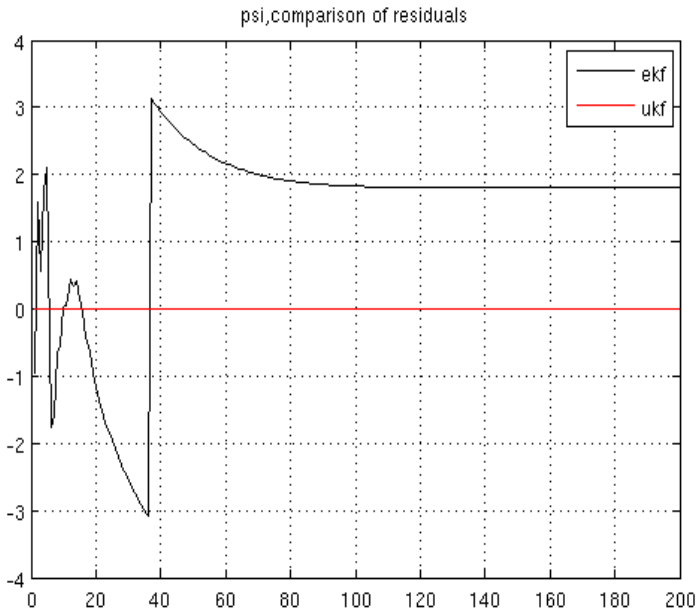

Figure 7.24: Residual of psi angle estimate with larger initial errors (first 200 data samples)

EKF			
first 200 data samples			
	1st set	2nd set	3rd set
phi	$1.4650 * 10^{-4}$	0.0226	33.7858
theta	$1.5413 * 10^{-4}$	0.0239	5.7586
psi	0.0164	$1.552 * 10^{3}$	748.31
complete samples			
	1st set	2nd set	3rd set
phi	0.0013	0.0238	33.7870
theta	0.0012	0.0250	5.7597
psi	4.8335	$1.2517 * 10^{4}$	$5.2688 * 10^{3}$

Table 7.4: Comparison of sum of squares of difference in ekf with different initial conditions

UKF			
first 200 data samples			
	1st set	2nd set	3rd set
phi	$1.6147 * 10^{-4}$	$1.6571 * 10^{-4}$	$1.7014 * 10^{-4}$
theta	$1.5587 * 10^{-4}$	$1.7598 * 10^{-4}$	$1.7601 * 10^{-4}$
psi	$1.4510 * 10^{-4}$	$1.7423 * 10^{-4}$	$1.8607 * 10^{-4}$
complete samples			
	1st set	2nd set	3rd set
phi	0.0102	0.0120	0.0140
theta	0.0116	0.0118	0.0128
psi	0.0109	0.0111	0.0119

Table 7.5: Comparison of sum of squares of difference in ukf with different initial conditions

7.2 INS-GPS Integration Results

Because of the lack of time, the complete tuning of the UKF filter for INS-GPS integration had to stop in the middle. Some available results so far is added below. Figures 7.25, 7.26 and 7.27 are results of estimation of latitude longitude and altitude respectively. Both implementation and tuning of the filter have to be improved from the current state and so far no considerable result is obtained.

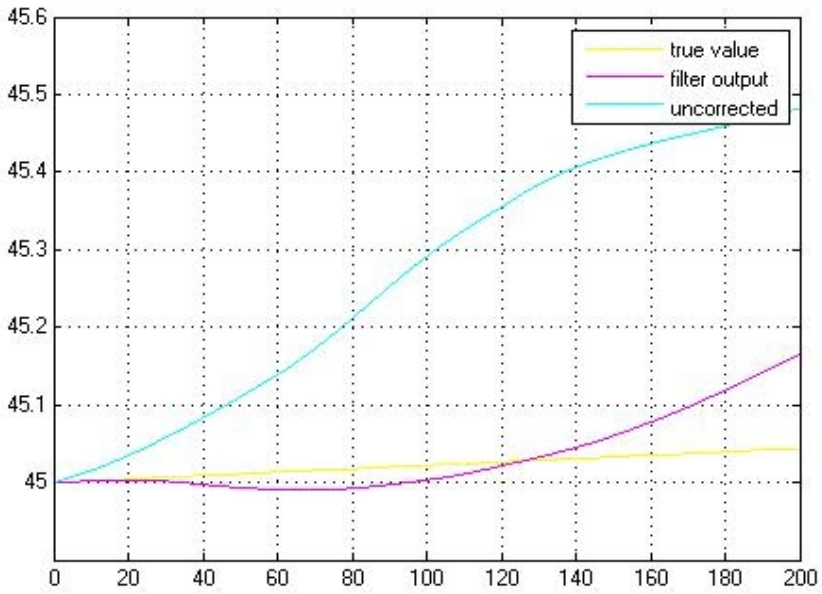

Figure 7.25: latitude

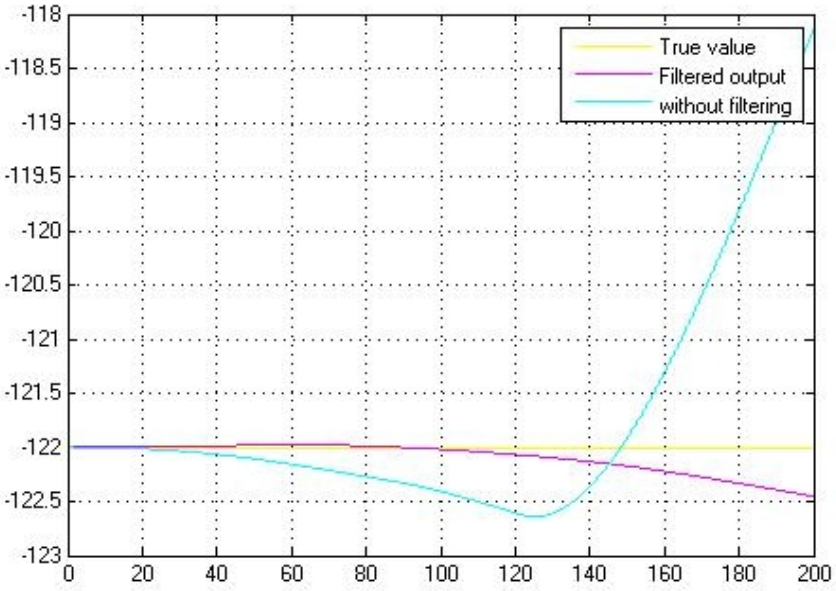

Figure 7.26: longitude

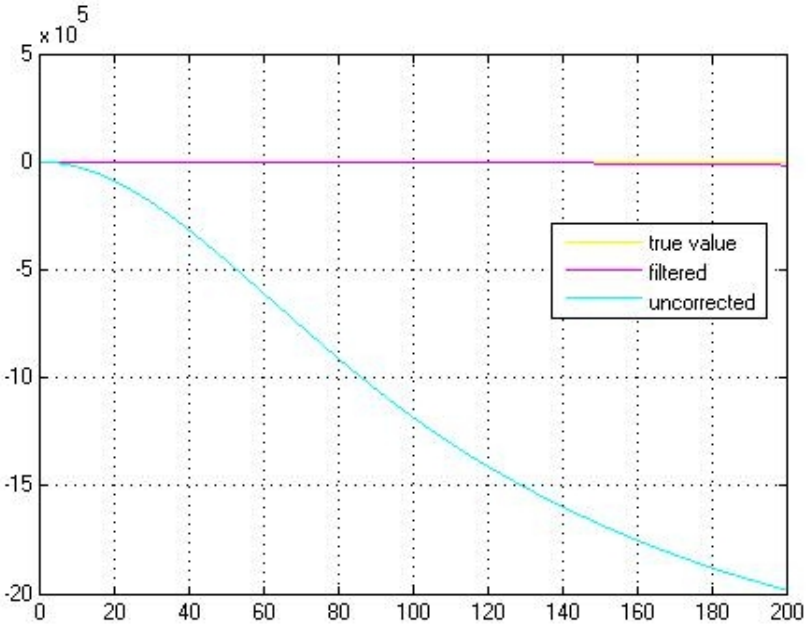

Figure 7.27: altitude

Chapter 8

Conclusion

Attitude estimation of a stabilized platform is studied and implemented to compare ekf and ukf. The original plan of implementing the ins-gps integration and to compare performance was not able to finish due to different difficulties faced and also due to time constraints. The attitude estimation results show that the performance comparison of the ekf and ukf filter is not that important to mention. But ukf performance with large initial error is very good, which makes it very acceptable for applications like navigation, where system is more susceptible to initial errors. But computational complexity of ukf algorithms compared to ekf is very evident with approximately 40 times more time consuming in terms of CPU time usage.

Chapter 9

Future Works

The information gathered and experience gained are useful in completing the implementation of ins-gps integration. Also the Square Root UKF can be tested because it claims to be more stable and less computational intensive. In Shin's PhD thesis [23], he point out the need of special treatment for the position (lat, long, alt) and quaternion states in UKF implementation. The reason being is that they are not belonging to any vector space. According to him, intrinsic characteristics of rotation needs to consider in quaternion estimation and also suggested the intrinsic gradient descent algorithm for that. It would also be interesting to test some auxiliary sensors like odometer and speedometer in navigation implementation in land vehicle applications.

Bibliography

[1] N. Shantha Kumar, T. Jann 2004 *Estimation of attitudes from a low-cost miniaturized inertial platform using Kalman Filter-based sensor fusion algorithm* . In Sadhana Vol. 29,Part 2, Pages 217-235.

[2] Randal W. Beard,Dep. of Electrical and Computer Engineering,Brigham university,Provo,Utah 2007 *State Estimation for micro Air Vehicles, Studies in Computational intelligence.* Springer Berlin/Heidelberg

[3] Brian L. Stevens,Frank L. Lewis 2003 *Aircraft control and Simulation, Edition 2.* Wiley-IEEE,ISBN 0471371459, 9780471371458

[4] Robert M. Rogers *Applied Mathematics in integrated Navigation System, Edition 2.* American Institute of Aeronautics and Astronautics,Inc.

[5] Anderson B D,Moore J B 1979 *Optimal Filtering.* Englewood Cliffs, NJ, Prentice Hall

[6] $http://en.wikipedia.org/wiki/Kalman_filter$ *Kalman Filter* . accessed on 13 june 2009

[7] $http://en.wikipedia.org/wiki/Global_Positioning_System$ *Global Positioning System* . accessed on 3rd august 2009

[8] $http://www.kowoma.de/en/gps/errors.htm$ *Global Positioning System* . accessed on 3rd august 2009

[9] $http://www.kowoma.de/en/gps/index.htm$ *Global Positioning System* . accessed on 3rd august 2009

[10] $http://www.hsg-imit.de/en/$ *Inertial Sensors* . accessed on 13th august 2009

[11] $http://www.aerospaceweb.org/question/performance/q0146.shtml$ *Body Frame* . accessed on 13th august 2009

[12] *http : //www.mathworks.com ECI,ECEF frames* . accessed on 13th august 2009

[13] *http : //www.u−dynamics.com/aerosim/ AeroSim,u-dynamics* . Hood River, OR 97031, USA

[14] Unmanned Dynamics *AEROSIM BLOCKSET VERSION 1.2 - INSTALLATION INSTRUCTIONS*. Hood River, OR 97031, USA

[15] Unmanned Dynamics *AEROSIM BLOCKSET Version 1.2 Users Guide*. Hood River, OR 97031, USA

[16] Eric A. Wan, Rudolph van der Merwe *http : //cslu.cse.ogi.edu/nsel/ukf/test.html The Unscented Kalman Filter for Nonlinear Estimation* . accessed on 14 june 2009

[17] Collinson R P G 1998 *Introduction to Avionics*. London: Chapman and Hall

[18] Julier S.J,Uhlmann J.K 1997 *A New Extension of the Kalman Filter to Nonlinear Systems*. In Proc. of AeroSense: The 11th Int. Symp. on Aerospace/Defence Sensing, Simulation and Controls.

[19] Rudolph van der Merwe , Eric A. Wan 1997 *THE SQUARE-ROOT UNSCENTED KALMAN FILTER FOR STATE AND PARAMETER-ESTIMATION*. Oregon Graduate Institute of Science and Technology Beaverton, Oregon 97006, USA

[20] Julier S.J,IDAK Industries,Jefferson City,MO,USA 2002 *The Scaled Unscented Transformation*. American Control Conference,2002,proceedings of the 2002, Volume:6 On Page(s) 4555-4559

[21] David H. Titterton, John L. Weston *Strapdown Inertial Navigation Technology,2nd Edition*. The Institution of Electrical Engineers.

[22] Adriano Solimeno 2007 *Low-Cost INS/GPS Data Fusion with Extended Kalman Filter for Airborne Applications,MSc Thesis*. Universidade Tecnica de lisboa

[23] Eun Hwan Shin 2005 *Estimation Techniques for Low-Cost Inertial Navigation,Phd Thesis*. Department of Geomatics Engineering,Calgary,Alberta,CA

[24] Jouni Hartikainen,Simo Srkk 2008 *Optimal filtering with Kalman filters and smoothers a Manual for Matlab toolbox EKF/UKF*. Department of Biomedical Engineering and Computational Science, Helsinki University of Technology, Espoo, Finland.

[25] *http://www.lce.hut.fi/research/mm/ekfukf/*. Centre of Excellence in Computational Complex Systems Research, Helsinki University of Technology, Espoo, Finland.