EXTENDED KALMAN FILTERING

APPLIED TO

A FULL ACCELEROMETER STRAPDOWN

INERTIAL MEASUREMENT UNIT

by
Alexandre Alves Santiago
Lieutenant-Commander (Naval Engineer) Brazilian Navy
(1979)

Submitted to the Department of
Aeronautics and Astronautics
in Partial Fulfillment of the Requirements for the
Degree of

MASTER OF SCIENCE
in Aeronautics and Astronautics

at the

Massachusetts Institute of Technology

September 1992

Signature of Author ______________________________
Department of Aeronautics and Astronautics
September 1st, 1992

Certified by ______________________________
Wallace Earl vander Velde, Sc.D.
Professor of Aeronautics and Astronautics, Thesis Supervisor

Accepted by ______________________________
Professor Harold Y. Wachman, Chairman, Departmental Graduate Committee
Department of Aeronautics and Aeronautics

EXTENDED KALMAN FILTERING APPLIED TO A FULL ACCELEROMETER INERTIAL MEASUREMENT UNIT

By

Alexandre Alves Santiago

Submitted to the Department of Aeronautics and Astronautics on September 1st, 1992, in partial fulfillment of the requirements for the degree of Master of Science in Aeronautics and Astronautics.

ABSTRACT

The objective of the present work consists of analyzing the performance of a Strapdown Full Accelerometer Inertial Measurement Unit and barometric altimeter aided by GPS data. The inertial measurement unit utilizes no gyroscopes; instead, both translation and rotation information is derived from a complement of 12 accelerometers located at different points on the vehicle. The extended Kalman filter is used to incorporate the aiding information into the navigation computation. The flight path, guidance law and dynamical behavior will be considered for one particular vehicle so as to establish a comparison between the desired and the actual flight path.

Thesis Supervisor: Dr. W. E. Vander Velde
Title: Professor of Aeronautics and Astronautics

Acknowledgements

First of all, I would like to express my gratitude to Professor Wallace Vander Velde for his support, guidance, inspiration and enthusiasm. Professor Vander Velde was not only my lecturer in the subjects of Statistical Problems in Automatic Control, Multivariable Control Systems and Strapdown Inertial Navigation Systems but also my thesis advisor since my first working day at M.I.T. He also gave me the opportunity of working in the field of Inertial Navigation Systems, which will be of great value for the next steps in my career.

I would like to thank Professor Lena Valavani for giving me a good background in the field of Estimation and Control of Stochastic Processes. The knowledge she gave me was very helpful in my research work.

I am grateful to M.I.T. for providing a great academic environment under which I spent about two years.

This acknowledgement is far from complete without mentioning the great support, attention and encouragement I received from my beloved wife Evelyn. She shared all moments of stress and difficulties I had with love, patience and faith in a better future and she enjoys the happiness I feel of getting along with one of the greatest challenges I ever faced in my life.

This work was sponsored by the Brazilian Navy. Special thanks are due to the Brazilian Naval Commission in Washington, D.C. and to the Instituto de Pesquisas da Marinha in Rio de Janeiro, Brazil.

Table of Contents

Chapter 1

INTRODUCTION

1.1 Background and Motivation

The idea of Inertial Navigation is always associated with the technique of determining the attitude, position and velocity of a vehicle by means of an inertial measurement unit in possible conjunction with any sort of external aiding.

When we think of an Inertial Measurement Unit in classical terms, our first approach consists of integrating a set of three accelerometers and three gyroscopes into a single unit. The accelerometers span the three or two dimensional space in which the navigation is carried out and the gyroscopes are devised to perform the attitude computation.

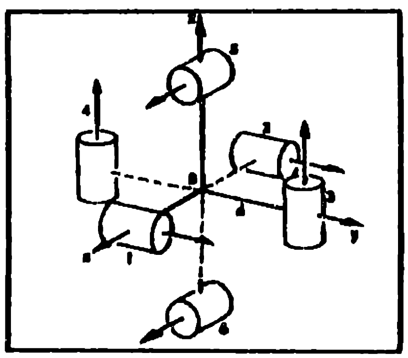

Figure 1: Six Accel.I.M.U

The idea of doing inertial navigation solely by means of accelerometers is not a new one. Prof. Wrigley, Hollister and Denhard, in their book Gyroscope Theory, Design, and Instrumentation [1], mention this possibility. Figure 1 is a reproduction of a figure from that reference. It gives a complement of 6 accelerometers from which one can derive linear and angular accelerations. It is not expected that navigation based on accelerometer information only can be as accurate as that using gyroscopes and accelerometers, but with the high quality of external aiding information now available, such a system may provide accuracy suitable for many purposes. If so, this configuration would have advantages over gyroscope systems in size, weight, power, and most importantly, cost.

The purpose of this work is to evaluate the performance achievable with an aided full accelerometer inertial navigation system. The configuration of Figure 1 is shown to be unstable without external aiding. An augmented version, with 12 accelerometers, is presented, which is at least neutrally stable without aiding.

1.2 Contribution of the Thesis

The aim of this work is to demonstrate the following facts:

a) It is possible to develop a full accelerometer inertial measurement unit in such a way that angular and linear accelerations can be obtained as linear combinations of the accelerometer outputs;

b) This system is shown to be at least neutrally stable and;

c) When receiving high quality external aiding through an Extended Kalman Filter during initialization and during flight it is feasible to use such a system to perform navigation in short range missions.

Chapter 2

SYSTEM MECHANIZATION

2.1 The Hypothetical Vehicle

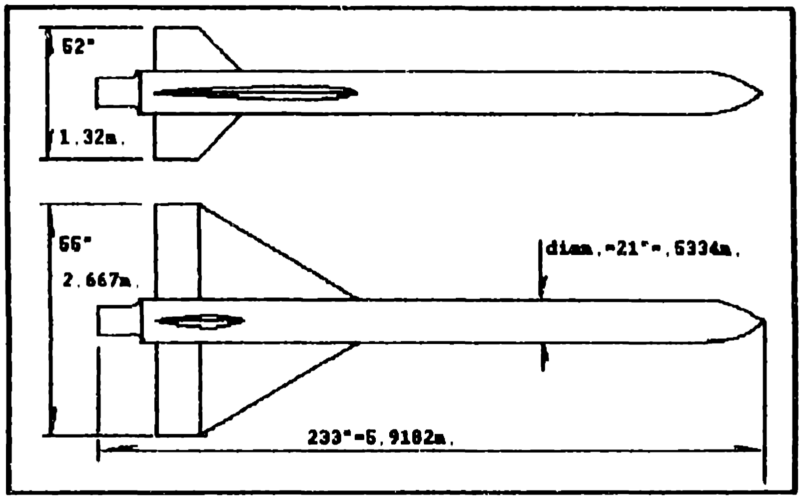

Figure 2 : The Hypothetical Vehicle

For this work I will consider two different arrangements. The first one consists of a vehicle in which a six accelerometer inertial measurement unit is assembled in a structure whose maximum dimension is one meter. In the second lay-out there are twelve accelerometers distributed along the vehicle structure.

In both lay-outs the vehicle is assumed to be a rigid body, in which no flutter or any kind of structural vibration occurs.

2.2 Description of the Sensors

For the purpose of this work it is assumed that the accelerometers are all of the same kind with the following stochastic characteristics:

- Mean Squared Output noise: $(10\ \mu g)^2$
- Bias Standard Deviation: $35 \mu g$

Effects such as scale factors, temperature sensitivity and misalignment during assembly were not taken into account.

In both configurations that will be analyzed, the six or the twelve accelerometer I.M.U., the vertical channel is always stabilized by means of a baro-altimeter. The stochastic model for this instrument is assumed to be a bias plus white noise as follows:

- White Noise Intensity: $2*10^{-11}m^2s$
- Bias Standard Deviation: 0.2 m.

In order to simplify the simulation, the effects of temperature, altitude, speed and weather correlation pattern were not taken into consideration.

2.3 The Full Accelerometer I.M.U. Layout

2.3.1 The Six Accelerometer I.M.U.

The lay-out of the six accelerometer I.M.U. is basically the same as presented in Figure 1. In this case the distance "d" was assumed to be 25 centimeters. I assume that there is no input axis misalignment in this lay-out so as to simplify the problem. The frame which carries the set of six accelerometers is located in the center of gravity of the vehicle described in Figure 2, section 2.1.

2.3.2 The Twelve Accelerometer I.M.U.

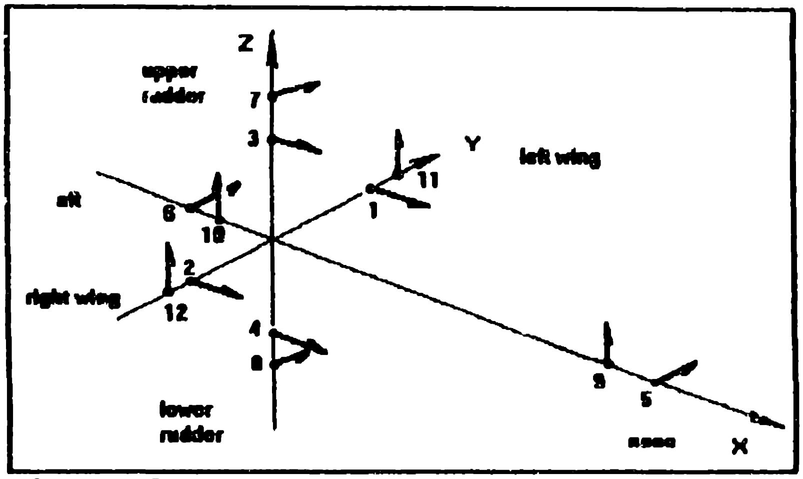

Figure 3: Twelve Accel. I.M.U. Lay-Out

In this case there is no inertial measurement unit in the sense that all instruments are grouped into a cluster. Instead of that the set of twelve accelerometers are distributed through the structure of the vehicle.

This is the reason for assuming that the vehicle is a rigid body, because under real circumstances all structural vibrating modes, distortions due to accelerations and flutter should have been taken into account.

Although this configuration may not seem to be a practical implementation, it is very useful in order to demonstrate that one is allowed to vary the number and location of the accelerometers provided that their input axes are properly arranged in space.

In Figure 3 the accelerometers are numbered from 1 to 12 and one can see the relative position that they occupy in the vehicle structure.The distances (d_i) from each accelerometer to the origin of the vehicle frame is as follows:

$d_1=d_2=1.2$ m.	$d_5=3.6$m.	$d_7=d_8=0.6$m.	$d_{10}=0.43$m.
$d_3=d_4=0.55$m.	$d_6=0.48$m.	$d_9=3.55$m.	$d_{11}=d_{12}=1.25$m.

The nature of the problem is not affected if these distances are changed in order to conform to the vehicle structure. The most important characteristic of this arrangement is the orientation of the accelerometer input axes.

2.4 External Aiding to the System

In terms of external aiding three sources of information are considered. The first one is the baro-altimeter which is incorporated into the system in order to stabilize the vertical channel. The second source of information is assumed to be the **Global Positioning System (GPS)**.

The following reasons led to the choice of the GPS satellite navigation system as an aiding system in order to set up the Extended Kalman Filter:

- available all over the world;
- provides three-dimensional position and velocity information with very well known accuracy and noise characteristics;
- this system is designed to be robust against jamming;
- low cost user equipment available;

According to Reference [2], the GPS measurements were assumed to have the following characteristics:

<u>Accuracy</u>

Position 3D $\leq$ 30m SEP 50

Velocity $\leq$ .1m/sec. SEP 50,

where SEP stands for spherical error probability, in this case 50%.

Based on that all simulations were run using a position standard deviation of 15m. and a velocity standard deviation of 0.1m/sec. because these represent typical stochastic characteristics of a vast number of receivers.

The information from the GPS receiver is presented directly to the Kalman Filter. The GPS receiver itself is not modelled in this work.

The third external aiding source is assumed to be a high quality inertial measurement unit that should be available at or on the launching platform of the vehicle. The launching platform inertial measurement unit should be able to determine not only the position and velocity of the vehicle but also to compute and transfer the vehicle attitude for initialization purposes.

It will also be assumed that the receiver presents loss of phase for accelerations greater than 3.5g and loss of code for accelerations greater than 10g. The maximum speed in which the receiver operates properly is 1000 Km/h.

Chapter 3

SYSTEM OPERATION

3.1 Coordinate Reference Frames

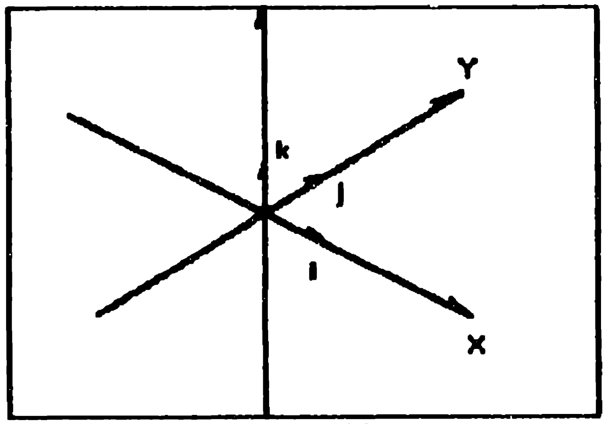

Figure 4: Direct three-orthogonal axes reference Frame

In this work the direct triad (three orthogonal axes) reference frame is used. In this kind of reference frame the vectorial or cross product of the unitary vector along the X axis times the unitary vector along the Y axis is equal to the unitary vector along the Z axis. With the Y axis taken to the left of the X axis, the Z axis points upwards instead of downwards as in the conventional case. This can be seen in Figure 4 above.

3.1.1 Vehicle Reference Frame

This frame is a set of orthogonal axes fixed in the rigid vehicle body as described in section 3.1. The origin of the body axes is in the center of gravity and its directions are the principal axes of inertia as shown in Figure 5. The sign conventions are:

X_b - positive forward

Y_b - positive port (to the left)

Z_b - positive up

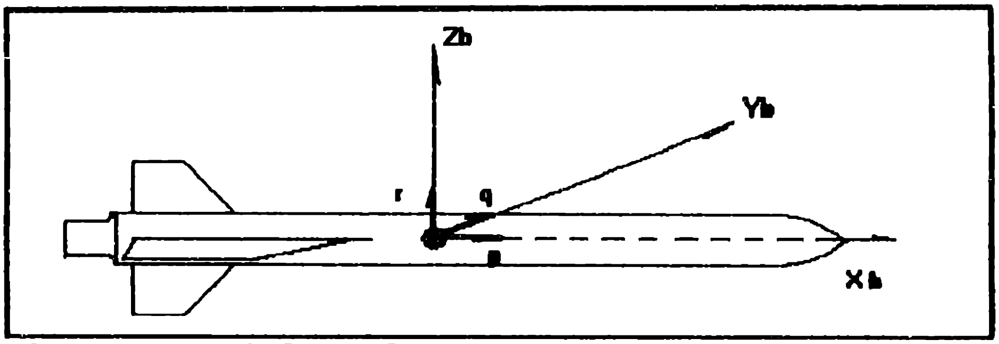

Figure 5:Vehicle Reference Frame

In the Figure 5, p, q, r represent the roll rate, the pitch rate and the yaw rate respectively. Those rates will be referred to as w'_x ,w'_y ,w'_z and the corresponding Euler Angles Φ, Θ and Ψ will be denoted simply by Θ_x, Θ_y and Θ_z.

3.1.2 Inertial Reference Frame

For the purpose of this simulation the Inertial Reference Frame and the Navigation Frame are the same and the Flat Earth Model is assumed. Therefore the position is expressed in terms of X, Y, and Z as being the longitudinal, lateral and vertical motion.

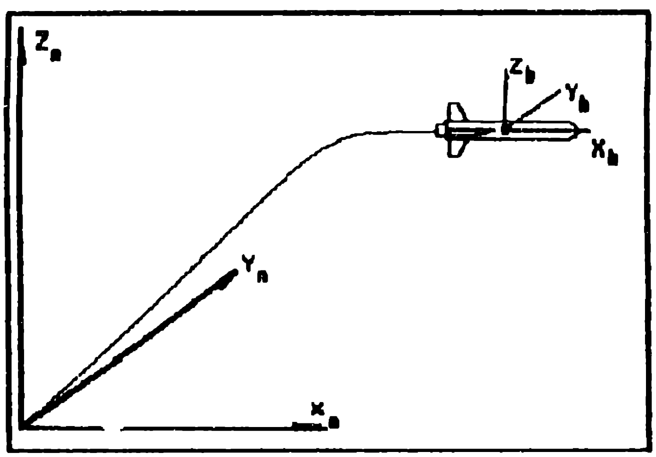

Figure 6:Inertial Reference Frame

3.2 Coordinate Frame Relationship

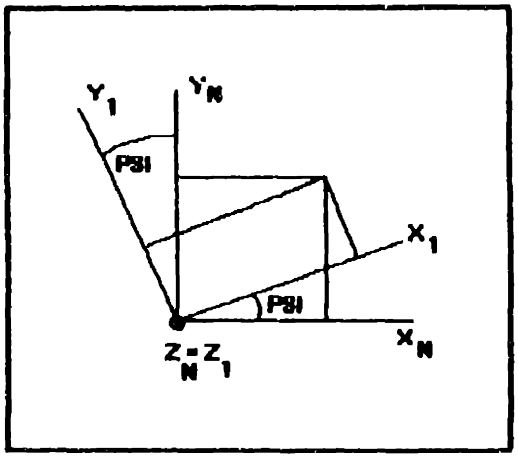

Figure 7:Transformation Matrix Yaw angle Ψ

As the vehicle has six degrees of freedom in space, it is necessary to derive a transformation matrix for the purpose of transforming any vector from the vehicle reference frame into the inertial reference frame which, in this case, is the navigation frame also. This goal can be achieved by using intermediate frames.

At first I consider the yaw angle Ψ and a first intermediate frame (1) and the relationship between frames is:

$$\begin{vmatrix} x_N \\ y_N \\ z_N \end{vmatrix} = \begin{vmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{vmatrix} \times \begin{vmatrix} x_1 \\ y_1 \\ z_1 \end{vmatrix} \rightarrow r_N = C_1^N \times r_1 \qquad (1)$$

Now the pitch angle represents the motion between the first and the second reference frames as shown in Figure 8.

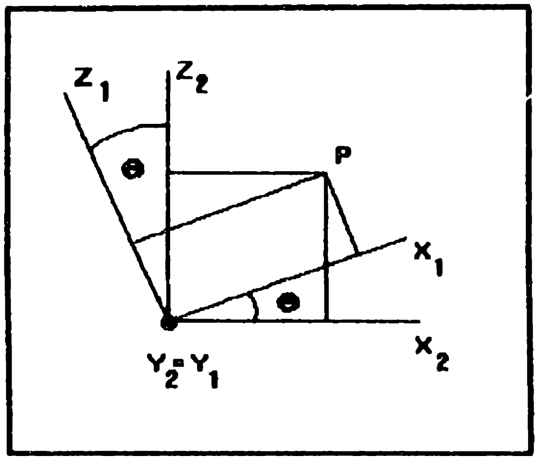

Figure 8:Transformation Matrix - Pitch Angle θ

The transformation from the second to the first reference frame will be (2):

$$\begin{vmatrix} x_1 \\ y_1 \\ z_1 \end{vmatrix} = \begin{vmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{vmatrix} \times \begin{vmatrix} x_2 \\ y_2 \\ z_2 \end{vmatrix} \rightarrow r_1 = C_2^1 \times r_2 \qquad (2)$$

The last transformation is from the body frame into the second reference frame:

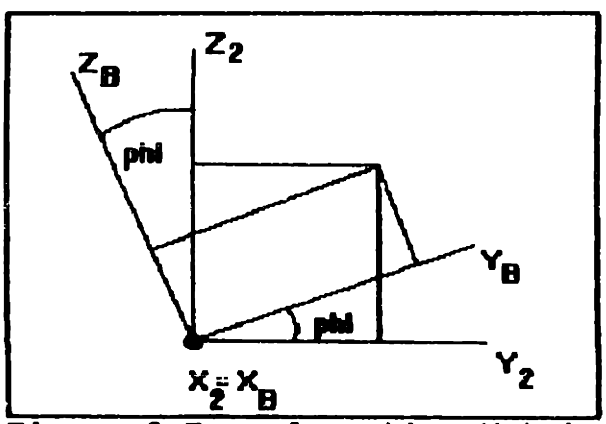

Figure 9:Transformation Matrix Roll angle Φ

This last transformation is given by (3):

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \times \begin{bmatrix} x_B \\ y_B \\ z_B \end{bmatrix} \rightarrow r_2 = C_B^2 \times r_B \tag{3}$$

Now I am able to compute the transformation matrix Cbn from the body (vehicle) frame into the navigation frame by multiplying these three transformation matrices:

$$C_b^n = C_1^N C_2^1 C_B^2 =$$

$$= \begin{bmatrix} \cos\psi\cos\theta & \cos\psi\sin\theta\sin\phi - \sin\psi\cos\phi & \cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi \\ \sin\psi\cos\theta & \sin\psi\sin\theta\sin\phi + \cos\psi\cos\phi & \sin\psi\sin\theta\cos\phi - \cos\psi\sin\theta \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix} \tag{4}$$

3.3 Transformation Matrix Algorithms

According to ref.[3], the function of the attitude reference system is to indicate the relationship between two coordinate systems and to display some parameters which communicate that transformation to an observer, or to transform some vector from one coordinate frame to another.

The matrix C_b^n (Eq. 4) is the first approach to this problem and it is called the Direction Cosine Matrix.

3.3.1 Direction Cosine Matrix

The coordinate transformation matrix is defined to be the matrix which transforms the components of a vector from one coordinate frame to another. In the present case the goal is to transform angular and linear acceleration from the vehicle reference frame into the navigation reference frame. This transformation can be understood as:

$$V^n = C_b^n \times V^b \tag{5}$$

where v is any given vector. v^b is this vector expressed in the body (vehicle) coordinates and v^n is the same vector in the navigation (in this case also inertial) reference frame.

$$C_b^n = \begin{vmatrix} C_{xx} & C_{xy} & C_{xz} \\ C_{yx} & C_{yy} & C_{yz} \\ C_{zx} & C_{zy} & C_{zz} \end{vmatrix} \tag{6}$$

The name direction cosine matrix comes from the fact that: where $C_{ij}=\cos\theta_{ij}$ is the projection that a unit vector along the j^{th} axis of the body frame has on the i^{th} axis of the navigation frame. This can be seen in Figure 10.

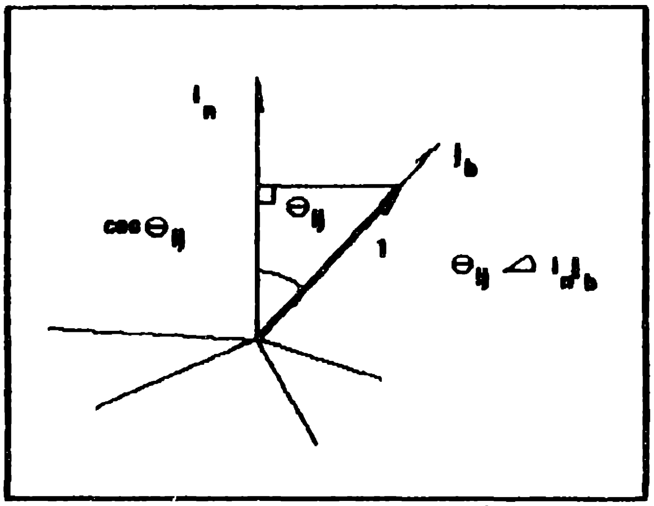

Figure 10: Direction Cosine

Therefore each element of this matrix is the cosine of the angle between two coordinate directions. This matrix is also orthogonal, which means that:

$$C_b^{n^{-1}} = C_b^{n^T} \qquad |C_b^n| = 1 \tag{7}$$

The derivative of equation (5) can be expressed as follows:

$$\dot{v}^n = \dot{C}_b^n \times v^b + C_b^n \times \dot{v}^b \tag{8}$$

The derivatives of the vectors as they appear in equation (8) can be understood as being the rate of change of these vectors observed from the same coordinatization frame:

$$\dot{v}^{n}=\left(\frac{\partial v}{\partial t}\right)_{n}^{n}$$
$$\dot{v}^{b}=\left(\frac{\partial v}{\partial t}\right)_{b}^{b} \tag{9}$$

If the navigation frame were not inertial, the angular rates could be referred to a third (and inertial) reference frame by using the Law of Coriolis:

$$\left(\frac{\partial v}{\partial t}\right)_{I}^{N}=\left(\frac{\partial v}{\partial t}\right)_{N}^{N}+\omega_{IN}{}^{N}\times v^{N}$$
$$=C_{B}^{N}\left[\left(\frac{\partial v}{\partial t}\right)_{B}^{B}+\omega_{IB}^{B}\times v^{B}\right] \tag{10}$$

Simplifying the notation,

$$\dot{v}^{N}+\omega_{I_N}{}^{N}\times v^{N}=C_{B}^{N}\left(\dot{v}^{B}+\omega_{IB}^{B}\times v^{B}\right) \tag{11}$$

and substituting $\dot{v}^{n}$ in (11) implies that,

$$\dot{C}_{B}^{N}v^{B}+C_{B}^{N}\dot{v}^{B}+\omega_{I_N}{}^{N}\times v^{N}=C_{B}^{N}\dot{v}^{B}+C_{B}^{N}\omega_{IB}^{B}\times v^{B} \tag{12}$$

and cancelling out equal terms at the left and right sides of (12) leads to,

$$\dot{C}_{B}^{N}=C_{B}^{N}\left[\omega_{IB}{}^{B}\times\right]-\left[\omega_{IN}{}^{N}\times\right]C_{B}^{N} \tag{13}$$

where the bracketed terms are an operator that can be defined as

$$[\omega\times]=\begin{vmatrix} 0 & -\omega_{z} & \omega_{y} \\ \omega_{z} & 0 & -\omega_{x} \\ -\omega_{y} & \omega_{x} & 0 \end{vmatrix} \tag{14}$$

This is a differential equation which can be integrated given the input angular velocities $W_{IB}{}^{B}$ and $W_{IN}{}^{N}$. In this case the navigation frame is also the inertial frame, so that I → N ⇒ W_{IN} →

$W_{NN}=0$ and the differential equation (12) turns out to be,

$$\dot{C}_B^N = C_B^N\left[\omega_{NB}^B \times\right] \tag{15}$$

3.3.2 Euler Angles

In order to compute the attitude from the Euler's Angles it is necessary to calculate the angular velocity of the vehicle with respect to the navigation frame. From section 3.2 the angular velocity $W_{NB}^{(B)}$ can be expressed as follows:

$$\omega_{NB}^B = C_1^B\begin{vmatrix}0\\0\\\dot{\psi}\end{vmatrix} + C_2^B\begin{vmatrix}0\\\dot{\theta}\\0\end{vmatrix} + \begin{vmatrix}\dot{\phi}\\0\\0\end{vmatrix} \tag{16}$$

This can be written as,

$$\omega_{NB}^B = \begin{vmatrix}1 & 0 & -\sin\theta\\0 & \cos\phi & \cos\theta\sin\phi\\0 & -\sin\phi & \cos\theta\cos\phi\end{vmatrix} \times \begin{vmatrix}\dot{\phi}\\\dot{\theta}\\\dot{\psi}\end{vmatrix} \tag{17}$$

The differential equation comes from equation (17) by taking the inverse of the matrix as can be seen below,

$$\begin{vmatrix}\dot{\phi}\\\dot{\theta}\\\dot{\psi}\end{vmatrix} = \begin{vmatrix}1 & \sin\phi\tan\theta & \cos\phi\tan\theta\\0 & \cos\phi & -\sin\phi\\0 & \dfrac{\sin\phi}{\cos\theta} & \dfrac{\cos\phi}{\cos\theta}\end{vmatrix} \times \omega_{NB}^B \tag{18}$$

Note that there is a singularity at $\theta=\mp 90$ deg.

W_{NB}^B can be computed as follows,

$$\omega_{NB}^B = \omega_{IB}^B - C_B^{N^T}\omega_{I_N}{}^N \tag{19}$$

but in this case the navigation frame is also the inertial frame so that $W_{NB}^B=W_{IB}^B$ and $W_{IN}^N=0$.

3.3.3 Rotation Vector

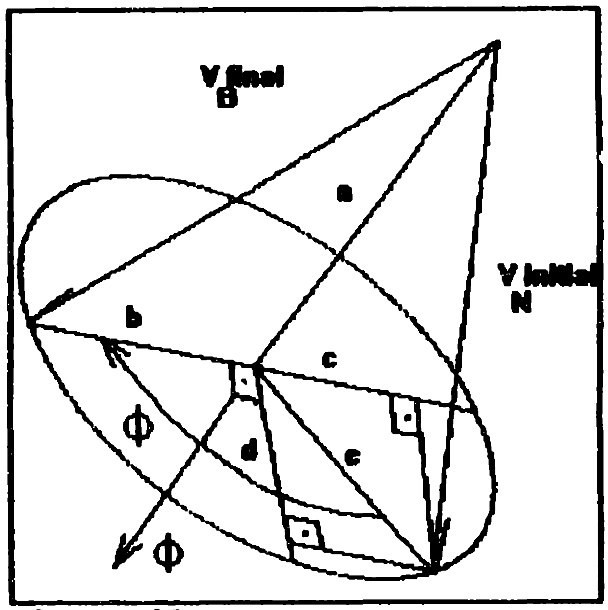

Figure 11:
Rotation Vector

Another way of computing attitude consists of determining the rotation vector. There is a single axis rotation which carries the navigation frame (N) into coincidence with the vehicle frame (B).

One interesting property is that the rotation vector coordinates are the same in both initial and final reference frames. The magnitude of the rotation vector is equal to the rotation expressed in radians and its direction is given by the right hand rule.

In the Figure 11 it is possible to see that the final vector in the B (vehicle) frame can be decomposed into one part along the rotation vector and another perpendicular to it. The part along the rotation vector is denoted by $\underline{a}$ and is given by,

$$\vec{a}=\frac{1}{\phi^2}(\vec{\phi}^T\vec{v})\vec{\phi}=\vec{\phi}\frac{1}{\phi^2}\vec{\phi}^T\vec{v}=\frac{1}{\phi^2}\vec{\phi}\vec{\phi}^T\vec{v} \qquad (20)$$

The component perpendicular to the rotation vector is denoted by $\underline{b}$ and is found to be,

$$\vec{b}=\vec{v}-\frac{1}{\Phi^2}(\vec{\Phi}^T\vec{v})\vec{\Phi}=\left[I-\frac{1}{\Phi^2}\vec{\Phi}\vec{\Phi}^T\right]\vec{v} \qquad (21)$$

In order to determine the initial vector V_N it is necessary to find out the components c and d in Figure 11. The component d is perpendicular to a and b and its magnitude is $|e|\sin\Phi$ The component c is collinear to b and its magnitude is $|e|\cos\Phi$ The magnitude of e is the same as that of b and the following relationships hold:

$$\left|\frac{1}{\Phi}\vec{\Phi}\times\vec{v}\right|=|v|\sin\theta$$
$$\left|\left[I-\frac{1}{\Phi^2}\vec{\Phi}\vec{\Phi}^T\right]\vec{v}\right|=|v|\sin\theta \qquad (22)$$
$$\Phi=\sqrt{\vec{\Phi}^T\vec{\Phi}}$$

The relationship between v^N and v^B is then expressed as follows,

$$v^N=\frac{1}{\Phi^2}\vec{\Phi}\vec{\Phi}^T\vec{v}^B+\sin\Phi\frac{1}{\Phi}[\vec{\Phi}\times]\vec{v}^B+\cos\Phi\left[I-\frac{1}{\Phi^2}\vec{\Phi}\vec{\Phi}^T\right]\vec{v}^B \qquad (23)$$

and knowing that,

$$[\vec{\Phi}\times]^2=\vec{\Phi}\vec{\Phi}^T-\Phi^2 I \qquad (24)$$

the transformation matrix can be expressed as,

$$C_B^N=I+\frac{\sin\Phi}{\Phi}[\vec{\Phi}\times]+\frac{1-\cos\Phi}{\Phi^2}[\vec{\Phi}\times]^2 \qquad (25)$$

Once it is known that the rotation vector is invariant under coordinate transformation between B and N, the following equality can be differentiated,

$$C_B^N\vec{\Phi}=\vec{\Phi} \qquad (26)$$

After some algebraic work this will lead to the following differential equation,

$$\vec{\phi}' = \vec{\omega} + \frac{1}{2}\vec{\phi}\times\vec{\omega} + A(\phi)[\vec{\phi}\times(\vec{\phi}\times\vec{\omega})] \quad (27)$$

where,

$$\begin{aligned} \phi &= \sqrt{\vec{\phi}^T\vec{\phi}} \\ A(\phi) &= \frac{1}{\phi^2}\left[1 - \frac{\phi\sin\phi}{2(1-\cos\phi)}\right] \\ \vec{\omega} &= \vec{\omega}^B_{NB} \end{aligned} \quad (28)$$

In this case $\underline{W}_{NB}{}^{B}$ is equal to $\underline{W}_{IB}{}^{B}$ and $A(\Phi)$ is well defined as the angle $\Phi \to 0$ as follows,

$$Lim_{\phi\to 0}A(\phi) = \frac{1}{12}\left(1 + \frac{\phi^2}{60}\right) \quad (29)$$

3.3.4 Quaternion or Cayley-Klein Method

All the previously discussed methods present singularities at $\Phi = \mp n2\pi$ for the rotation vector or $\Theta = \mp n\pi/2$ for the Euler angles. One way to avoid these singularities without integrating the full direction cosine matrix is to utilize the Quaternion Method.

The quaternion method is a four parameter representation of the relationship between coordinate frames and can be represented as follows,

$$\vec{Q}^T = [a, b, c, d] \quad (30)$$

$$\vec{Q}^T = [a, \vec{\rho}^T] \quad (31)$$

where the four parameters a, b, c, d are real numbers. The last three numbers can be collected in a three dimensional vector where a and $\boldsymbol{\rho}$ are defined below as,

$$
\begin{aligned}
a &= \cos\left(\frac{\Phi}{2}\right) \\
\vec{\beta} &= \left(\frac{\sin\frac{\Phi}{2}}{\Phi}\right)\vec{\Phi} = \sin\left(\frac{\Phi}{2}\right)\vec{1}_{\Phi}
\end{aligned} \tag{32}
$$

where Φ is the rotation vector.

As this is a four parameter representation for a three degree of freedom rotation, there must be a constraint among these parameters which is given by,

$$
\begin{aligned}
a^2+\vec{\beta}^T\vec{\beta} &= \cos^2\frac{\Phi}{2}+\sin^2\frac{\Phi}{2}=1 \\
&\rightarrow a^2+b^2+c^2+d^2=1
\end{aligned} \tag{33}
$$

From the rotation vector theory it is possible to derive the following differential equation for the quaternion,

$$
\begin{aligned}
\dot{a} &= -\frac{1}{2}\vec{\beta}^T\vec{\omega} \\
\vec{\beta}' &= \frac{1}{2}(a\vec{\omega}-\vec{\omega}\times\vec{\beta}) = \\
&= \frac{1}{2}(a\vec{\omega}+\vec{\beta}\times\vec{\omega})
\end{aligned} \tag{34}
$$

The transformation matrix can be formed as,

$$
\begin{aligned}
C_B^N &= I+2a[\vec{\beta}\times]+2[\vec{\beta}\times]^2 \\
&= I+2a[\vec{\beta}\times]+2[\vec{\beta}\vec{\beta}^T-\rho^2 I] \\
&= (1-2\rho^2)I+2a[\vec{\beta}\times]+2\vec{\beta}\vec{\beta}^T
\end{aligned} \tag{35}
$$

or it can be expressed in the vectorial form,

$$
\vec{v}^N=\vec{v}^B+2a\vec{\beta}\times\vec{v}^B+2\vec{\beta}\times(\vec{\beta}\times\vec{v}^B) \tag{36}
$$

The differential equation (34) can be also written in terms of the quaternion components as follows,

$$\dot{a}=-\frac{1}{2}(b\omega_x+c\omega_y+d\omega_z) \quad \dot{c}=\frac{1}{2}(a\omega_y-b\omega_z+d\omega_x)$$
$$\dot{b}=\frac{1}{2}(a\omega_x+c\omega_z-d\omega_y) \quad \dot{d}=\frac{1}{2}(a\omega_z+b\omega_y-c\omega_x) \tag{37}$$

If the transformation matrix is expressed in terms of the quaternion elements the equality below holds,

$$C_B^N=\begin{bmatrix} a^2+b^2-c^2-d^2 & 2(bc-ad) & 2(ac+bd) \\ 2(ad+bc) & a^2-b^2+c^2-d^2 & 2(cd-ab) \\ 2(db-ac) & 2(ab+cd) & a^2-b^2-c^2+d^2 \end{bmatrix} \tag{38}$$

In order to determine the attitude of the vehicle according to the usual conventions, its is necessary to define the Elevation (E), the Bank angle (B) and the Heading (H) as being the negative of the Euler angles θ, ϕ and ψ due to the fact that the axis z was defined upwards in both body and reference frames. The Euler's angles can be recovered according to,

$$\begin{aligned} \theta&=\sin^{-1}[2(ac-bd)] \\ \phi&=\sin^{-1}\left[\frac{2(ab+cd)}{\cos\theta}\right] \\ \psi&=\sin^{-1}\left[\frac{2(ad+bc)}{\cos\theta}\right] \end{aligned} \tag{39}$$

Note that Φ changes quadrant when $a^2-b^2-c^2+d^2 < 0$ and ψ does the same when $a^2+b^2-c^2-d^2 < 0$. To calculate $\cos(\theta)$, the following relation holds,

$$\cos\theta=\sqrt{1-4(ac-bd)^2} \tag{40}$$

3.4 Accelerometer Signal Processing

The navigation algorithm is designed to perform the navigation based on the outputs of the accelerometers and that of the baro-altimeter used in order to stabilize the vertical channel.

The first step to achieve this goal consists of a proper arrangement of the accelerometers in space and the correct processing of their information.

3.4.1 General Dynamics

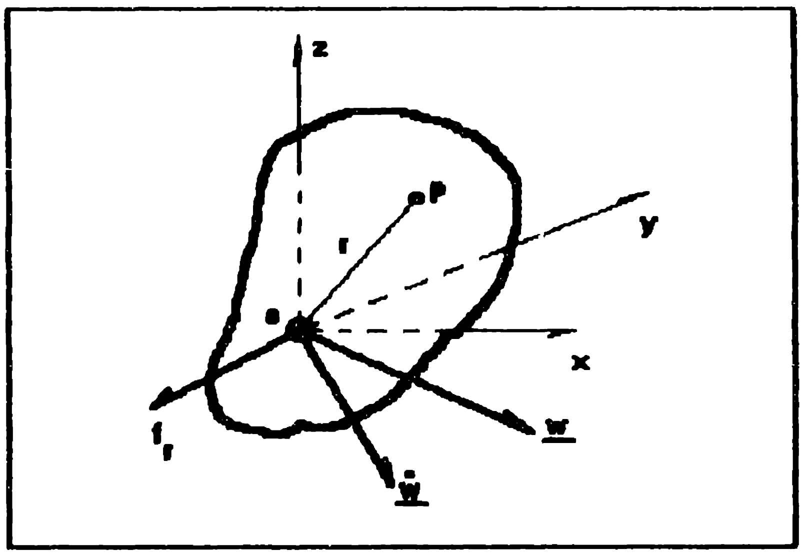

Figure 12: Rigid body

In the Figure 12 is shown a rigid body subjected to a linear acceleration f_x, an angular velocity w and an angular acceleration w'. The objective here is to determine the acceleration at point P.

$$\vec{a}_P=\vec{f}_o+\vec{\omega}'\times\vec{r}+\vec{\omega}\times(\vec{\omega}\times\vec{r}) \tag{41}$$

The equation above can be expanded as follows,

$$\begin{vmatrix}a_x\\a_y\\a_z\end{vmatrix}=\begin{vmatrix}f_x\\f_y\\f_z\end{vmatrix}+\begin{vmatrix}0 & -\omega_z' & \omega_y'\\ \omega_z' & 0 & -\omega_x'\\ -\omega_y' & \omega_x' & 0\end{vmatrix}\begin{vmatrix}r_x\\r_y\\r_z\end{vmatrix}+\begin{vmatrix}0 & -\omega_z & \omega_y\\ \omega_z & 0 & -\omega_x\\ -\omega_y & \omega_x & 0\end{vmatrix}\begin{vmatrix}0 & -\omega_z & \omega_y\\ \omega_z & 0 & -\omega_x\\ -\omega_y & \omega_x & 0\end{vmatrix}\begin{vmatrix}r_x\\r_y\\r_z\end{vmatrix} \tag{42}$$

and after multiplying the matrices this expression is reduced to,

$$\begin{vmatrix} a_x \\ a_y \\ a_z \end{vmatrix} = \begin{vmatrix} f_x + r_z\omega'_y - r_y\omega'_z + r_y\omega_x\omega_y + r_z\omega_x\omega_z - r_x\left(\omega_y^2 + \omega_z^2\right) \\ f_y + r_x\omega'_z - r_z\omega'_x + r_z\omega_y\omega_z + r_x\omega_x\omega_y - r_y\left(\omega_x^2 + \omega_z^2\right) \\ f_z + r_y\omega'_x - r_x\omega'_y + r_x\omega_x\omega_z + r_y\omega_y\omega_z - r_z\left(\omega_x^2 + \omega_y^2\right) \end{vmatrix} \tag{43}$$

Now it is possible to apply the equation above to determine the acceleration as measured by each accelerometer in two different lay-outs. In the next sections it will be considered the six and the twelve accelerometer lay-outs.

3.4.2 The Six Accelerometer I.M.U.

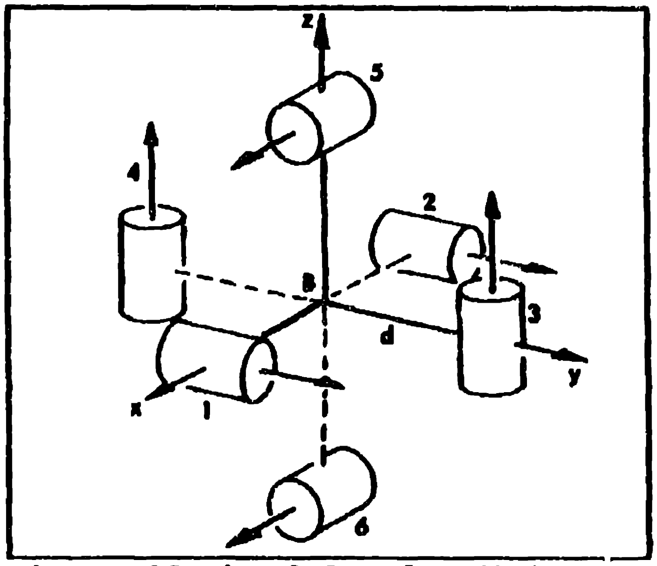

Figure 13:The 6 Accel.I.M.U

The Figure above represents the six accelerometer lay-out.

Consider the acceleration as measured from the accelerometer number 1. This accelerometer has its input axis pointing along the positive y axis and its position corresponds to $r=r_x=d$. Consequently only the component a_y from equation (43) is measured.

If this reasoning is applied to all six accelerometers, then it is possible to derive the following set of equations representing the output of each one of the six accelerometers,

$$\begin{aligned}
a_1&=f_y+d\omega_z'+d\omega_x\omega_y\\
a_2&=f_y-d\omega_z'-d\omega_x\omega_y\\
a_3&=f_z+d\omega_x'+d\omega_y\omega_z\\
a_4&=f_z-d\omega_x'-d\omega_y\omega_z\\
a_5&=f_x+d\omega_y'+d\omega_x\omega_z\\
a_6&=f_x-d\omega_y'-d\omega_x\omega_z
\end{aligned} \qquad (44)$$

From the above set of equations (44) one is able to obtain the expressions for the angular accelerations W_x', W_y' and W_z' ,

$$\omega_x' = \frac{(a_3 - a_4)}{2d} - \omega_y \omega_z = \alpha_x^b$$
$$\omega_y' = \frac{(a_5 - a_6)}{2d} - \omega_z \omega_x = \alpha_y^b \qquad (45)$$
$$\omega_z' = \frac{(a_1 - a_2)}{2d} - \omega_x \omega_y = \alpha_z^b$$

and for the linear accelerations as follows,

$$f_x = \frac{(a_5 + a_6)}{2}$$
$$f_y = \frac{(a_1 + a_2)}{2} \qquad (46)$$
$$f_z = \frac{(a_3 + a_4)}{2}$$

The equation (45) exhibits an unstable behavior. For example, if W_z assumes a constant negative value, then the first two equations for W_x' and W_y' become unstable. The consequence of this fact is that the set of nonlinear differential equations (45), which is responsible for the attitude determination, is highly unstable in open loop conditions. For small values of W and using external aiding, this system can be stabilized by an extended Kalman Filter as will be seen later in this work.

3.4.3 The Twelve Accelerometer I.M.U.

The way to avoid the quadratic terms in the right hand side of the differential equation set (45) was presented by Prof. Wallace Earl Vander Velde and consists of adding six more accelerometers to the inertial measurement unit. This makes the realization of a marginally stable system possible as will be seen below.

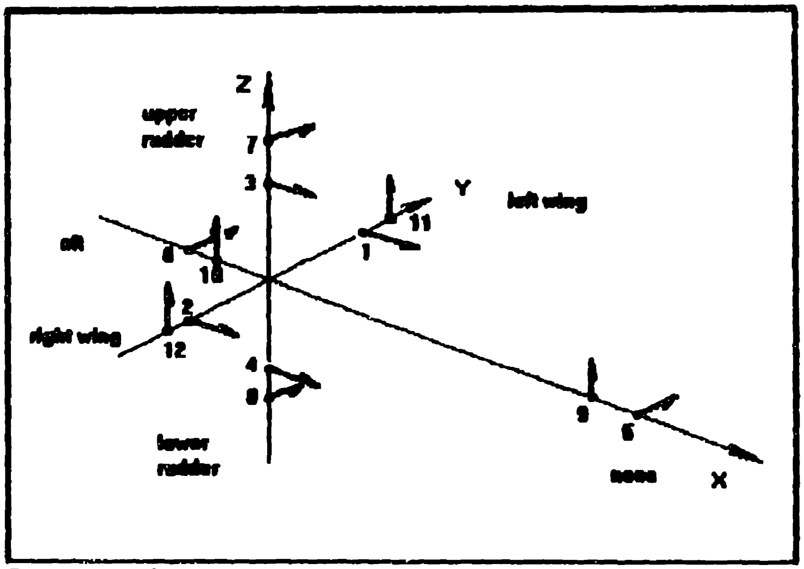

Figure 14:Twelve Accel.I.M.U.

The figure number 3 is repeated above for convenience.The position of each accelerometer is assumed to be:

d1=d2=1.2 m.   d7=d8=.6 m.

d3=d4=.55 m   d9=3.55 m.

d5=3.6 m.   d10=.43m.

d6=.48m   d11=d12=1.25m.

Using equation (43) it is possible to obtain from the above lay-out the expression for each of the accelerometer measurements as shown below,

$$
\begin{array}{ll}
a_1=f_x-d_1\omega'_z+d_1\omega_x\omega_y & a_7=f_y-d_7\omega'_x+d_7\omega_y\omega_z \\
a_2=f_x+d_2\omega'_z-d_2\omega_x\omega_y & a_8=f_y+d_8\omega'_x-d_8\omega_y\omega_z \\
a_3=f_x+d_3\omega'_y+d_3\omega_x\omega_z & a_9=f_z-d_9\omega'_y+d_9\omega_x\omega_z \\
a_4=f_x-d_4\omega'_y-d_4\omega_x\omega_z & a_{10}=f_z+d_{10}\omega'_y-d_{10}\omega_x\omega_z \\
a_5=f_y+d_5\omega'_z+d_5\omega_x\omega_y & a_{11}=f_z+d_{11}\omega'_x+d_{11}\omega_y\omega_z \\
a_6=f_y-d_6\omega'_z-d_6\omega_x\omega_y & a_{12}=f_z-d_{12}\omega'_x-d_{12}\omega_y\omega_z
\end{array}
\qquad (47)
$$

Combining the equations for a_1,a_2,a_3 and a_4 in (47) leads to,

$$f_x=\frac{(a_1+a_2+a_3+a_4)}{4} \tag{48}$$

The second set of four equations in (47) gives,

$$f_y=\frac{d_6}{2(d_5+d_6)}a_5+\frac{d_5}{2(d_5+d_6)}a_6+\frac{a_7}{4}+\frac{a_8}{4} \tag{49}$$

and from the third set of four equations in (47) comes,

$$f_z=\frac{d_{10}}{2(d_9+d_{10})}a_9+\frac{d_9}{2(d_9+d_{10})}a_{10}+\frac{a_{11}}{4}+\frac{a_{12}}{4} \tag{50}$$

Combining the equations for a_7, a_8, a_{11} and a_{12} in (47) an expression for W_x' can be obtained,

$$\alpha_x^b=\omega_x'=-\frac{a_7}{4d_7}+\frac{a_8}{4d_7}+\frac{a_{11}}{4d_{11}}-\frac{a_{12}}{4d_{11}} \tag{51}$$

From the equations for a_3, a_4, a_9 and a_{10} we obtain two expressions for W_y',

$$\begin{aligned} \alpha_y^b=\omega_y'&=\frac{a_3}{2d_3}+\frac{a_{10}}{2d_{10}}-\frac{f_x}{2d_3}-\frac{f_z}{2d_{10}} \quad (\times 2d_3d_{10}) \\ \alpha_y^b=\omega_y'&=-\frac{a_4}{2d_4}-\frac{a_9}{2d_9}+\frac{f_x}{2d_4}+\frac{f_z}{2d_9} \quad (\times 2d_4d_9) \end{aligned} \tag{52}$$

The equations will be weighted by the product of the accelerometer positions in order to emphasize the pair which has the larger arm, which contributes to a better angular acceleration resolution when the system is not being externally aided. In this case, the pair of accelerometers a_4 and a_9 has the larger arm as can be seen from the lay-out in Figure (12).

The same procedure is applied to the equations for a_1, a_2, a_5 and a_6 resulting in,

$$\alpha_z^b=\omega_z'=-\frac{a_1}{2d_1}-\frac{a_6}{2d_6}+\frac{f_x}{2d_1}+\frac{f_y}{2d_6} \quad (\times 2d_1d_6)$$
$$\alpha_z^b=\omega_z'=\frac{a_2}{2d_2}+\frac{a_5}{2d_5}-\frac{f_x}{2d_2}-\frac{f_y}{2d_5} \quad (\times 2d_2d_5) \tag{53}$$

and the criterion for weighting was the same of that for equation (52).

The next step consists of eliminating f_x and f_z from equation (52) and f_x and f_y from equation (53). At that very point the weighting criterion will prove to be useful. In fact there are two expressions for W_y' and W_z' and the weighting is used to reflect the fact that the accelerometer pair with larger arm will be the most reliable to determine the angular acceleration. This reasoning leads to the following expression for W_y'

$$\begin{aligned}\alpha_y^b=\omega_y'=&\frac{(d_9-d_{10})}{8(d_3d_{10}+d_4d_9)}a_1+\frac{(d_9-d_{10})}{8(d_3d_{10}+d_4d_9)}a_2+\frac{(d_9+3d_{10})}{8(d_3d_{10}+d_4d_9)}a_3+\\&-\frac{(3d_9+d_{10})}{8(d_3d_{10}+d_4d_9)}a_4-\frac{d_4}{2(d_3d_{10}+d_4d_9)}a_9+\frac{d_3}{2(d_3d_{10}+d_4d_9)}a_{10}\end{aligned} \tag{54}$$

and the following one for W_z'.

$$\begin{aligned}\alpha_z^b=\omega_z'=&-\frac{(d_5+3d_6)}{8(d_1d_6+d_2d_5)}a_1+\frac{(d_6+3d_5)}{8(d_1d_6+d_2d_5)}a_2+\frac{(d_6-d_5)}{8(d_1d_6+d_2d_5)}a_3+\\&+\frac{(d_6-d_5)}{8(d_1d_6+d_2d_5)}a_4+\frac{d_2}{2(d_1d_6+d_2d_5)}a_5-\frac{d_1}{2(d_1d_6+d_2d_5)}a_6\end{aligned} \tag{55}$$

The most important conclusion is that the set of equations (48), (49), (50), (51), (54) and (55) give the linear and angular accelerations of the origin of the vehicle reference frame as linear combinations of the accelerometer outputs. There are no ω_x, ω_y or ω_z terms, as are necessary with the six accelerometer configurations, which can represent unstable feedback effects.

The linear characteristic of this system makes it suitable to receive external aiding through an Extended Kalman Filter with good performance in terms of estimation. This technique also leads to a marginally stable system which can be used for short times in the open-loop condition, i.e., in the event of external aiding failure or jamming.

The above mentioned ideas will be discussed in detail in the next sections.

3.5 The Navigation Algorithm

Here it will be illustrated how a navigation algorithm can be applied to a twelve accelerometer inertial measurement unit as described in the previous section (3.4.3).

The basic idea of a navigation algorithm consists of two coupled problems. The first one is the updating of the attitude by integrating the angular acceleration, the quaternion dynamics and constructing the transformation matrix C_b^n. The C_b^n matrix in its turn is used to transform the linear acceleration from the body (vehicle) reference frame into the navigation reference frame.

From the previous section it is known that the angular and the linear accelerations turn out to be linear combinations of the accelerometer outputs. In this way it is possible to construct from the coefficients of equations (48), (49), (50), (51), (54) and (55) one matrix which will be called the "C" matrix so that,

$$\begin{bmatrix} f_X^B \\ f_Y^B \\ f_Z^B \\ \alpha_X^B \\ \alpha_Y^B \\ \alpha_Z^B \end{bmatrix} = \begin{bmatrix} C_{1,1} & C_{1,2} & \cdots & C_{1,12} \\ \cdot & & & \cdot \\ \cdot & & & \cdot \\ \cdot & & & \cdot \\ \cdot & & & \cdot \\ C_{6,1} & \cdot & \cdots & C_{6,12} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \cdot \\ \cdot \\ a_{11} \\ a_{12} \end{bmatrix} \qquad (56)$$

This matrix C basically is the one which processes the accelerometer outputs into the information that will be used by the navigation software. As can be seen from equation (56), the accelerometer outputs are converted into a vector in which the first three elements are the specific force components and the last three elements are the angular acceleration components (α_i) coordinatized in the vehicle reference frame.

The Figure below illustrates the basic structure of the navigation algorithm.

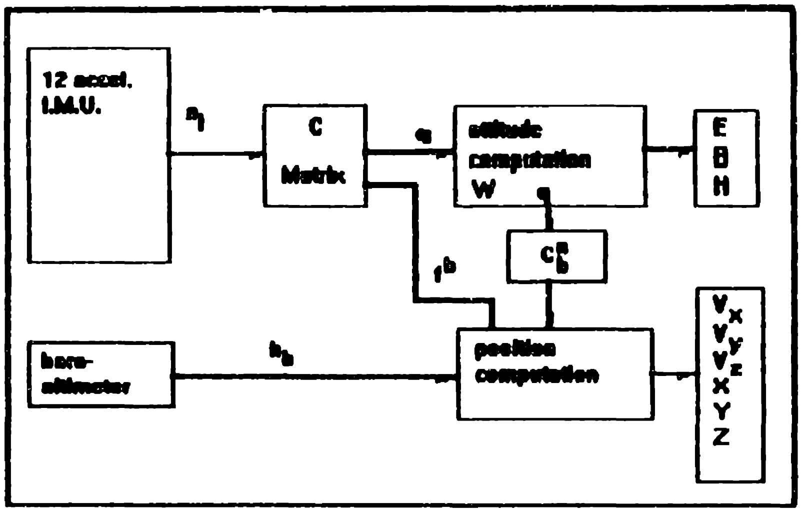

Figure 15:Navigation Algorithm

3.5.1 Attitude Computation

The attitude computation consists of integrating the angular acceleration and the quaternion dynamics so that it is possible to obtain the vehicle attitude by calculation of the Euler angles and to compute the transformation matrix C_b^n. Once the C_b^n matrix is calculated, the linear acceleration vector can be expressed in the navigation reference frame. To deal with this part the following differential equations are integrated,

$$\begin{aligned}\dot{a}&=-\frac{1}{2}(b\omega_x+c\omega_y+d\omega_z)\\ \dot{b}&=\frac{1}{2}(a\omega_x+c\omega_z-d\omega_y)\\ \dot{c}&=\frac{1}{2}(a\omega_y-b\omega_z+d\omega_x)\\ \dot{d}&=\frac{1}{2}(a\omega_z+b\omega_y-c\omega_x)\end{aligned} \quad \wedge \quad \begin{aligned}\omega_x'&=\alpha_x^b\\ \omega_y'&=\alpha_y^b\\ \omega_z'&=\alpha_z^b\end{aligned} \qquad (57)$$

The right hand sides of the three last equations (α_x^b, α_y^b, and α_z^b) correspond to the last elements of the processed accelerometer output vector from equation (56).

After one step of integration the transformation matrix C_b^n can be updated from the quaternion as already shown,

$$C_B^N = \begin{bmatrix} a^2+b^2-c^2-d^2 & 2(bc-ad) & 2(ac+bd) \\ 2(ad+bc) & a^2-b^2+c^2-d^2 & 2(cd-ab) \\ 2(db-ac) & 2(ab+cd) & a^2-b^2-c^2+d^2 \end{bmatrix} \tag{58}$$

and the Euler angles can be also immediately obtained as,

$$\theta=\sin^{-1}[2(ac-bd)] \quad \phi=\sin^{-1}\left[\frac{2(ab+cd)}{\cos\theta}\right] \quad \psi=\sin^{-1}\left[\frac{2(ad+bc)}{\cos\theta}\right]$$
$$[E \ B \ H]^T = -\frac{180}{\Pi}[\theta \ \phi \ \psi]^T \tag{59}$$

where E, B, H correspond to Elevation, Bank angle and Heading in Degrees.

3.5.2 Position Computation

The position computation in a strapdown inertial navigation system requires the stabilization of the vertical channel. This stabilization is necessary because unaided inertial navigation is unstable in the vertical direction. The instability arises from the effect of position errors on the calculated value of gravity. The resulting error in gravity causes an acceleration error in the same direction as the position error.

To stabilize the vertical channel using measurements from a barometric altimeter, the following equation is integrated,

$$\begin{bmatrix} \dot{v}_x^n \\ \dot{v}_y^n \\ \dot{v}_z^n \end{bmatrix} = \begin{bmatrix} C_{xx} & C_{xy} & C_{xz} \\ C_{yx} & C_{yy} & C_{yz} \\ C_{zx} & C_{zy} & C_{zz} \end{bmatrix} \begin{bmatrix} f_x^b \\ f_y^b \\ f_z^b \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ g_z \end{bmatrix} - k_2 \begin{bmatrix} 0 \\ 0 \\ (Z^n-h_b) \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ h_{aux} \end{bmatrix} \tag{60}$$

The above equation (60) accomplish three steps. At the right hand side of the equation, the first term transforms the measured specific force vector $\underline{f}^b$ from vehicle coordinates into navigation coordinates by multiplying it by the transformation matrix C_b^n. The second term at the right hand side subtracts out the estimated gravity acceleration. There are numerous models (stochastic and deterministic) to estimate g_z, but in this work for simplicity it will be a constant value equal to 9.80665 m/s^2. The third term at the right hand side introduces a feedback compensation where the error signal is the difference between the calculated altitude Z^n and the measured barometric altitude. The fourth term is the auxiliary variable h_{aux} , which has the following dynamics,

$$\dot{h}_{aux}=k_3(Z_n-h_b) \tag{61}$$

and its objective is to eliminate the steady state error which is the difference between the indicated altitude Z_n and the barometric altimeter measurement h_b.

From this point on the last step is just the integration of the velocities to obtain the position of the vehicle, which is carried out by the following equation,

$$\begin{aligned}\dot{X}^n&=v_x^n\\ \dot{Y}^n&=v_y^n\\ \dot{Z}^n&=v_z^n-k_1(Z_n-h_b)\end{aligned} \tag{62}$$

According to reference [4], the problem now consists of adjusting the gains k_1, k_2, and k_3 in order to make the system track the baro-altimeter according to a given specification. In this work it is desired that all modes decay with a time constant of 50 seconds, so that the vertical channel must be expressed in terms of its own differential equation as follows,

$$\begin{vmatrix}\dot{v}_z^n\\ \dot{h}_{aux}\\ \dot{Z}^n\end{vmatrix}=\begin{vmatrix}0 & -1 & -k_2\\ 0 & 0 & k_3\\ 1 & 0 & -k_1\end{vmatrix}\begin{vmatrix}v_z^n\\ h_{aux}\\ Z_n\end{vmatrix}+\begin{vmatrix}1 & -1 & k_2\\ 0 & 0 & -k_3\\ 0 & 0 & k_1\end{vmatrix}\begin{vmatrix}f_z^n\\ g_z\\ h_b\end{vmatrix} \tag{63}$$

In this subsystem the state variables are v_z^n, h_{aux} and Z^n and the inputs are f_z^n, g_z and h_b where f_z^n is the specific vertical force measurement in the navigation frame, g_z is the estimated gravity and h_b is the measured barometric-altimeter output.

The characteristic equation of this system is,

$$\lambda^3+\lambda^2 k_1+\lambda k_2+k_3=0 \tag{64}$$

The inertial navigation system takes care of the high frequency information in terms of altitude while the baro-altimeter has a slower response. For this reason all modes will be set to have a time constant of 50 seconds, which was considered to be satisfactory for this application. In this way all eigenvalues will be placed at -0.02 and the characteristic equation should be as follows,

$$(\lambda+0.02)^3=(\lambda^3+60\times10^{-3}\lambda^2+1.2\times10^{-3}\lambda+8\times10^{-6})=0 \tag{65}$$

If the coefficients are equated in equations (64) and (65), the values of the gains are found to be: $k_1=6\times10^{-2}$, $k_2=1.2\times10^{-3}$ and $k_3=8\times10^{-6}$.

CHAPTER 4

THE EXTENDED KALMAN FILTER

4.1 Introduction

Every inertial navigation unit has noisy sensors, in the present case, noisy accelerometers.

The baro-altimeter is another source of error and it is directly coupled to the inertial system to provide vertical channel stabilization as was shown in the previous chapter. The noise corrupts the measurements which is not desirable for navigation purposes.

This full accelerometer inertial navigation system is nonlinear because the quaternion dynamics have quadratic terms of the type (q_i x W_j) as can be seen on the right hand side of equation (57). There is another nonlinear operation when the transformation matrix C_b^n multiplies the specific force measurement. The six accelerometer inertial measurement unit has non-linearities in the angular acceleration measurement as can be seen in equation (45), but here the work will be specialized to the twelve accelerometer case.

In order to implement a practical system using this type of inertial measurements, it is necessary to introduce external aiding information and estimate all the states in such a way that the error covariance matrix is minimized. An extended Kalman filter is used for this purpose. According to Peter S. Maybeck [5], "the basic idea of the extended Kalman filter is to relinearize the system about each estimate $\hat{x}(t_i^+)$ once it has been computed. As soon as a new state estimate is made, a new and better reference state trajectory is incorporated into the estimation process. In this manner, one enhances the validity of the assumption that deviations from the reference (nominal) trajectory are small enough to allow linear perturbation techniques to be employed with adequate results".

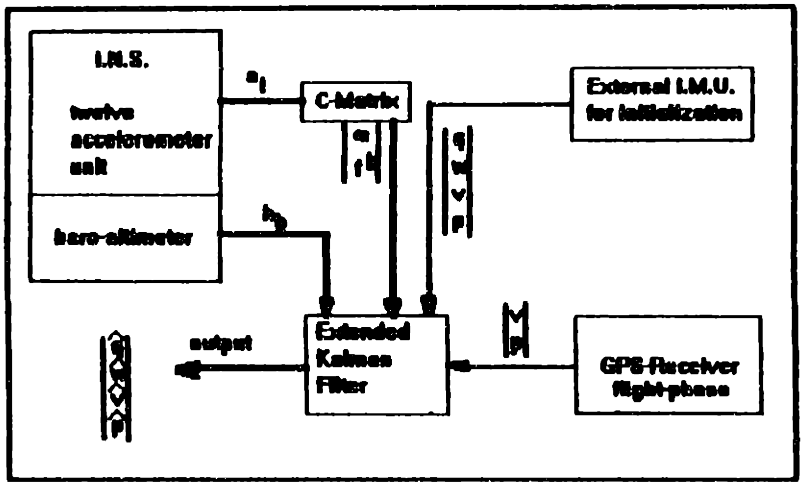

Figure 16:E.K.F. Navigator

Figure 16 shows the interaction among the many parts of the system. The Extended Kalman Filter is the central part. The inertial measurement unit sends the outputs of the accelerometers to a processing block called "C-Matrix" which produces a vector of angular and linear acceleration expressed in body coordinates. This input vector is sent to the Kalman filter. The baro-altimeter also sends its information to the Kalman Filter.

In this design the navigation is done by the Kalman filter. In practical implementations, this kind of design leads to an extremely low bandwidth because each computation cycle would take at least one second to be done. Consequently this bandwidth would not allow this information to be used in the guidance and attitude control of the vehicle, due to the high frequency modes of the vehicle dynamics. The second constraint would be the reliability of the design. If there is any failure in the Kalman filter, the navigation information would be completely lost. In a practical implementation, the Kalman filter would be used to estimate the error in the inertial navigation information rather than the full navigation state.

Although this is not the best choice in terms of reliability in a practical design, it makes complete simulation a feasible task on a personal computer. The Kalman filter does the navigation because it generates the full state estimate, instead of the error estimate.

The external measurement is given by an external high accuracy inertial navigation system during the initialization phase. It is assumed that this external I.M.U. is able to give the initial attitude, angular velocity, linear velocity and position of the vehicle while in the launching platform within a great precision.

After the boost phase, when the vehicle experiences a great acceleration, the external measurement of velocity and position is given by a GPS receiver.

4.2 General Extended Kalman Filter Equation

Our deterministic navigation system is described by the differential equations (57), (60), (61) and (62) but the stochastic model which incorporates the presence of noise can be represented by the following differential equation,

$$\vec{x}'(t)=\vec{f}[\vec{x}(t),\vec{u}(t),t]+G(t)\,\vec{w}(t) \tag{66}$$

where the vectors have the following meaning:

$$\begin{aligned}\vec{x}&=[a\ b\ c\ d\ \omega_x\ \omega_y\ \omega_z\ v_x\ v_y\ v_z\ h_{aux}\ X\ Y\ Z\,]^T\\ \vec{u}&=[\alpha_x^b\ \alpha_y^b\ \alpha_z^b\ f_x^b\ f_y^b\ f_z^b\ h_b]^T\\ \vec{w}&=[W_1\ W_2\ \cdots\ W_{11}\ W_{12}\ W_{13}]^T\end{aligned} \tag{67}$$

Here the state vector $\underline{x}$ represents the quaternion, angular velocity, linear velocity, auxiliary variable and position. The vector $\underline{u}$ represents the pre-processed accelerometer input and the baro-altimeter measurement. Finally, the vector $\underline{W}$ corresponds to the 12 accelerometer ($W_{1\to12}$) and altimeter (W_{13}) white noises and G(t) is a matrix which incorporates the noise into the process. The zero-mean white Gaussian noise vector has a covariance kernel defined as,

$$E\{\vec{w}(t)\vec{w}^T(t+\tau)\}=Q(t)\delta(\tau) \tag{68}$$

The external measurement, which is also noisy, is modelled as,

$$\vec{z}(t_i)=\vec{h}\,[\vec{x}(t_i),\, t_i]+\vec{v}(t_i) \tag{69}$$

where $\underline{v}$ is a white Gaussian noise sequence of zero mean and covariance kernel

$$E\{\vec{v}(t_i)\vec{v}^T(t_j)\}=\begin{cases} R(t_i) & t_i=t_j \\ 0 & t_i\neq t_j \end{cases} \tag{70}$$

To linearize the process it is necessary to describe the perturbation by a first order approximation as follows,

$$\delta\vec{x}'(t)=F\,[t;x_n(t)]\;\delta\vec{x}(t)+G(t)\vec{w}(t) \tag{71}$$

where F is the n-by-n matrix of partial derivatives of f with respect to the state vector evaluated along the nominal trajectory, which in this case will be the estimated one The matrix F is defined as,

$$F\,[t;\vec{x}_n(t)]=\left.\frac{\partial\vec{f}\,[\vec{x},\vec{u}(t),\,t]}{\partial\vec{x}}\right|_{x=x_n(t)} \tag{72}$$

Now the perturbed measurement can be modelled as,

$$\delta\vec{z}(t_i)=H\,[t_i;\vec{x}_n(i)]\;\delta\vec{x}(t_i)+\vec{v}(t_i) \tag{73}$$

where H is defined as,

$$H\,[t_i;\vec{x}_n(t_i)]=\left.\frac{\partial h\,[\vec{x},\,t_i]}{\partial x}\right|_{x=x_n(t_i)} \tag{74}$$

The above technique is known as a linearized or perturbation Kalman filter.

In this design is employed full-state estimation and an extended Kalman filter. It means that instead of working with errors, the full-state will be considered and instead of linearizing about a nominal trajectory, the system will be linearized about each estimate $\hat{x}(t_i^+)$ once it has been computed. This affects the F and H matrices in that they will be computed as,

$$F\left[t;\hat{X}(t/t_i)\right]=\frac{\partial \vec{f}\left[\vec{X},\vec{u}(t),t\right]}{\partial \vec{X}}\bigg|_{x=\hat{X}(t/t_i)} \tag{75}$$

and,

$$H\left[t_i;\hat{X}(t_i^-)\right]=\frac{\partial h\left[\vec{x},t_i\right]}{\partial x}\bigg|_{x=\hat{x}(t_i^-)} \tag{76}$$

The extended Kalman filter measurement update incorporates the measurement $z(t_i,w_j)$ by means of the optimal gain calculation,

$$K(t_i)=P\left(t_i^-\right)H^T\left[t_i;\hat{X}\left(t_i^-\right)\right]\left\{H\left[t_i;\hat{X}(t_i^-)\right]P(t_i^-)H^T\left[t_i;\hat{X}(t_i^-)\right]+R(t_i)\right\}^{-1} \tag{77}$$

the state estimate update,

$$\hat{X}(t_i^+)=\hat{X}(t_i^-)+K(t_i)\left\{\vec{z}_i-\vec{h}\left[\hat{X}(t_i^-),t_i\right]\right\} \tag{78}$$

and the covariance matrix update,

$$\begin{aligned} P(t_i^+) &= P(t_i^-)-K(t_i)\ H\left[t_i;\hat{X}(t_i^-)\right]\ P(t_i^-)\\ &= \left\{I-K(t_i)H\left[t_i;\hat{X}(t_i^-)\right]\right\}\ P(t_i^-)\ \left\{I-K(t_i)H\left[t_i;\hat{X}(t_i^-)\right]\right\}^T+\\ &\quad +K(t_i)R(t_i)K^T(t_i) \end{aligned} \tag{79}$$

The covariance matrix reflects the uncertainty in the knowledge of each state variable.

Between Measurements, the state estimates are propagated by integrating the nonlinear navigation equations, which are represented by the equation (66) without the noise term. This leads to the following equation,

$$\hat{X}(t_{i+1}^-)=\hat{X}(t_i^+)+\int_{t_i}^{t_{i+1}}\vec{f}\left[\hat{X}(t/t_i),\vec{u}(t),t\right]dt \tag{80}$$

and this integration is carried out by a 5^{th} order Runge-Kutta-

Fehlberg routine.

The covariance matrix is propagated by the usual linearized relation as follows,

$$P(t_{i+1}^-)=\Phi[t_{i+1},t_i;\hat{x}(\tau/t_i)]P(t_i^+)\Phi^T[t_{i+1},t;\hat{x}(\tau/t_i)]^\dagger + \int_{t_i}^{t_i+1}\Phi[t_{i+1},\tau;\hat{x}(\tau/t_i)]G(\tau)Q(\tau)G^T(\tau)\Phi^T[t_{i+1},\tau;\hat{x}(\tau/t_i)]d\tau \tag{81}$$

In the above expression, Φ denotes the transition matrix associated with $F[\tau;\ \hat{x}\ (\tau/t_i\ )]$ for all $\tau\in[t_i\ ,t_{i+1}\ )$. The propagation of the covariance matrix will be discussed in detail in the section 4.6.

The initial conditions for the propagation are provided by equations (77) and (78);

$$\begin{aligned}\hat{x}(t_i/t_i)&=\hat{x}(t_i^+)\\ P(t_i/t_i)&=P(t_i^+)\end{aligned} \tag{82}$$

and upon integrating (80) and (81) to the next sample time, $\hat{x}(t_{i+1}^-\ )$ and $P(t_{i+1}^-\ )$ are defined as

$$\begin{aligned}\hat{x}(t_{i+1}^-)&=\hat{x}(t_{i+1}/t_i)\\ P(t_{i+1}^-)&=P(t_{i+1}/t_i)\end{aligned} \tag{83}$$

4.3 Sensor Stochastic Modelling

Usually the order of the system may be augmented in order to incorporate the stochastic modelling of the sensors or any dynamics in the measurements. In this section the modelling of the sensors will be discussed.

4.3.1 Accelerometer

It is already known from the previous section that we have assumed an accelerometer with the following stochastic characteristics:

- Mean Squared Output Noise: $(10\mu g)^2$
- Bias Standard Deviation: $(35\mu g)$

From the mean squared output noise it is possible to derive the corresponding white noise intensity that drives the accelerometer. The mean squared output is to be understood as the area under the power spectral density curve as shown in the figure below,

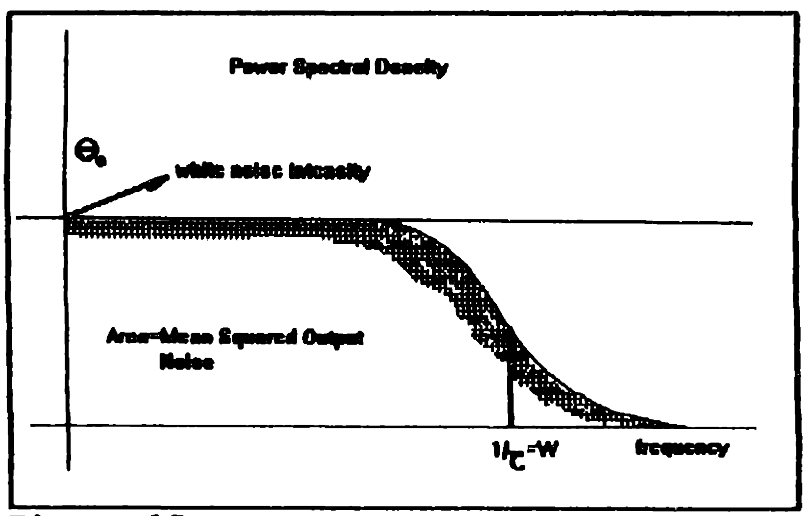

Figure 17: Accel.Power Spectral Density

The accelerometer noise is now modelled as a first order Markov process with a very short correlation time in order to validate the white noise assumption. The power spectral density function is expressed as follows,

$$\Theta(\omega)=\frac{\Theta_0}{1+\tau^2\omega^2}\rightarrow$$
$$\rightarrow\Theta(s)=\frac{\frac{1}{\tau^2}\Theta_0}{\frac{1}{\tau^2}-s^2}=\frac{\frac{1}{\tau^2}\Theta_0}{\left(\frac{1}{\tau}+s\right)\left(\frac{1}{\tau}-s\right)} \tag{84}$$

where the third term corresponds to the spectral factorization.

The mean squared value, as can be seen from the last Figure, is the integral of the area below the power spectral density function, and, in this case, this area is represented by the following integral,

$$\text{Mean Squared Output}=\frac{1}{2\pi j}\int_{-j\infty}^{j\infty}\frac{\frac{1}{\tau^2}\Theta_0}{\left(\frac{1}{\tau}+s\right)\left(\frac{1}{\tau}-s\right)}ds$$
$$=\frac{\frac{1}{\tau^2}\Theta_0}{\frac{2}{\tau}} \tag{85}$$
$$=\frac{1}{2\tau}\Theta_0$$

This leads to the expression below,

$$\Theta_{0\,accel.}=2\tau\times M.S.O \tag{86}$$

Now it is assumed that the accelerometer bandwidth is equal to 1000 rad/sec, which implies that $\tau=1\mathrm{x}10^{-3}$ sec. In this way the accelerometer white noise intensity will be $\Theta_0=1.9234\mathrm{x}10^{-11}\mathrm{m}^2/\mathrm{s}^3$.

Once the white noise intensity is determined, it is possible to write the **Q matrix** in the extended Kalman filter formulation, which refers to the process noise.

The error dynamics of one accelerometer can be the represented as a bias with standard deviation of 35µg plus a zero mean white

Gaussian noise with intensity [illegible] $=1.9234\times10^{-11} m^2/s^3$ as in the equation below,

$$\begin{gathered}\delta a_i = a_{b_i} + w_{a_i} \\ \dot{a}_{b_i} = 0\end{gathered} \tag{87}$$

4.3.2 Baro-Altimeter

The baro-altimeter is a noiseless instrument in practical terms, but according to Reference [4], it is subjected to the error sources as described below,

$$\delta h_b = e_{po} + e_{hsf} h + C_{sp} v^2 + e_{inst} \tag{88}$$

where:

- δh_b= baro-altimeter error;
- e_{po}= error due to the variation in altitude of a constant pressure surface;
- e_{hsf}= scale factor error due to non-standard temperature;
- C_{sp}= coefficient of static pressure measurement error;
- e_{inst}= instrument errors.

The error e_{po} has the following dynamics,

$$\begin{gathered}\dot{e}_{p_o} = -\omega_{alt} e_{p_o} + \omega_{p_o} \\ \omega_{alt} = \frac{v}{d_{alt}} \\ N_{alt} = \frac{2}{\omega_{alt}} \sigma^2_{alt}\end{gathered} \tag{89}$$

where:

- d_{alt}= correlation distance of weather patterns;
- σ_{alt}= standard deviation of the variation in altitude of a constant pressure surface;
- N_{alt}= power spectral density of the white noise w_{po};
- v= vehicle speed.

Typical values for the model parameters are σ_{alt}=500 ft. and d_{alt} $=1.6\times10^{6}$ft.

The error due to the temperature variation is directly related to the altitude and is described as,

$$e_{temp}=e_{hsf}h \qquad (90)$$
$$\dot{e}_{hsf}=0$$

having a bias-like behavior. It is a good assumption to make $\sigma_{hsf}=.03$

The static pressure measurement error $e_{sp}=c_{sp}v^2$ has its coefficient defined as

$$\dot{c}_{sp}=0 \qquad (91)$$

and the standard deviation of this coefficient can be assumed to be $\sigma_{sp}=1.5 \times 10^{-4}$ ft/(ft/sec)^2.

These parameters may vary from region to region and reflect average conditions at the East Coast of the United States. Considering that the hypothetical vehicle is intended for short range missions, the baro-altimeter stochastic dynamics can be well represented by a bias plus white noise as follows,

$$\delta h=h_{bias}+w_b \qquad (92)$$
$$\dot{h}_{bias}=0$$

and w_b is a fictitious white noise whose intensity is made of the same order as that of the accelerometer, so that $Q_{bar}=2 \times 10^{-11}$ meters^2 x sec. Considering that the altitude is a well known parameter at the launching platform, for simulation purposes the initial standard deviation of the bias was taken to be 0.2 meters.

4.4 State Estimate Propagation for the Six Accelerometer I.M.U.

In order to address performance issues it is necessary to design an extended Kalman filter for the six accelerometer system so that a comparison with the 12 accelerometer design is possible.

As shown previously, the stochastic system can be modelled as,

$$\vec{x}'(t)=\vec{f}\,[\vec{x}(t),\vec{u}(t),t]+G(t)\vec{w}(t) \tag{93}$$

and now this system has to be augmented in order to include the stochastic dynamics of the accelerometers and of the baro-altimeter. To this purpose, the x vector is augmented as follows,

$$\vec{x}=[a\ b\ c\ d\ \omega_x\ \omega_y\ \omega_z\ v_x\ v_y\ v_z\ h_{aux}\ X\ Y\ Z\ a_{b1}\ a_{b2}\ a_{b3}\ a_{b4}\ a_{b5}\ a_{b6}\ h_{bias}] \tag{94}$$

where a, b, c, d is the quaternion state; ω_i is the angular velocity; h_{aux} is the auxiliary variable for the vertical channel stabilization; X, Y, Z is the position; a_{ib} are the accelerometer biases; and h_{bias} is the baro-altimeter bias. The white noise vector enters the system via the matrix G, which reflects how the white noise enters through the accelerometers and baro-altimeter input.

The way of obtaining the state estimate dynamics consists basically of eliminating the bias from the inputs and this will lead to the following dynamics. For the quaternion estimate it is,

$$\begin{aligned}
\hat{a}' &= -\frac{1}{2}(\hat{b}\hat{\omega}_x+\hat{c}\hat{\omega}_y+\hat{d}\hat{\omega}_z)\\
\hat{b}' &= \frac{1}{2}(\hat{a}\hat{\omega}_x+\hat{c}\hat{\omega}_z-\hat{d}\hat{\omega}_y)\\
\hat{c}' &= \frac{1}{2}(\hat{a}\hat{\omega}_y-\hat{b}\hat{\omega}_z+\hat{d}\hat{\omega}_x)\\
\hat{d}' &= \frac{1}{2}(\hat{a}\hat{\omega}_z+\hat{b}\hat{\omega}_y-\hat{c}\hat{\omega}_x)
\end{aligned} \tag{95}$$

For the angular acceleration estimates the following holds,

$$
\begin{aligned}
\hat{\omega}_x' &= \frac{(a_3-a_4)}{2d}-\hat{\omega}_y\hat{\omega}_z-\frac{(\hat{a}_{b3}-\hat{a}_{b4})}{2d}\\
\hat{\omega}_y' &= \frac{(a_5-a_6)}{2d}-\hat{\omega}_z\hat{\omega}_x-\frac{(\hat{a}_{b5}-\hat{a}_{b6})}{2d}\\
\hat{\omega}_z' &= \frac{(a_1-a_2)}{2d}-\hat{\omega}_x\hat{\omega}_y-\frac{(\hat{a}_{b1}-\hat{a}_{b2})}{2d}
\end{aligned}
\tag{96}
$$

For the velocity estimate with respect to the navigation frame it is

$$
\begin{vmatrix}\hat{v}_x^{n'}\\ \hat{v}_y^{n'}\\ \hat{v}_z^{n'}\end{vmatrix}=\begin{vmatrix}\hat{C}_{xx} & \hat{C}_{xy} & \hat{C}_{xz}\\ \hat{C}_{yx} & \hat{C}_{yy} & \hat{C}_{yz}\\ \hat{C}_{zx} & \hat{C}_{zy} & \hat{C}_{zz}\end{vmatrix}\begin{vmatrix}\hat{f}_x^b\\ \hat{f}_y^b\\ \hat{f}_x^b\end{vmatrix}-\begin{vmatrix}0\\0\\g_z\end{vmatrix}-k_2\begin{vmatrix}0\\0\\(\hat{z}^n-h_b+\hat{h}_{bias})\end{vmatrix}-\begin{vmatrix}0\\0\\\hat{h}_{aux}\end{vmatrix}
\tag{97}
$$

where the estimated specific force in body coordinates is

$$
\begin{aligned}
\hat{f}_x^b &= \frac{(a_5+a_6)}{2}-\frac{(\hat{a}_{b5}+\hat{a}_{b6})}{2}\\
\hat{f}_y^b &= \frac{(a_1+a_2)}{2}-\frac{(\hat{a}_{b1}+\hat{a}_{b2})}{2}\\
\hat{f}_z^b &= \frac{(a_3+a_4)}{2}-\frac{(\hat{a}_{b3}+\hat{a}_{b4})}{2}
\end{aligned}
\tag{98}
$$

and C_b^n is,

$$
\hat{C}_B^N=\begin{bmatrix}\hat{a}^2+\hat{b}^2-\hat{c}^2-\hat{d}^2 & 2(\hat{b}\hat{c}-\hat{a}\hat{d}) & 2(\hat{a}\hat{c}+\hat{b}\hat{d})\\ 2(\hat{a}\hat{d}+\hat{b}\hat{c}) & \hat{a}^2-\hat{b}^2+\hat{c}^2-\hat{d}^2 & 2(\hat{c}\hat{d}-\hat{a}\hat{b})\\ 2(\hat{d}\hat{b}-\hat{a}\hat{c}) & 2(\hat{a}\hat{b}+\hat{c}\hat{d}) & \hat{a}^2-\hat{b}^2-\hat{c}^2+\hat{d}^2\end{bmatrix}
\tag{99}
$$

The auxiliary variable estimated will be,

$$
\hat{h}_{aux}'=k_3(\hat{z}-h_b+\hat{h}_{bias})
\tag{100}
$$

The position estimate differential equation turns out to be,

$$
\begin{aligned}
\hat{X}' &= \hat{v}_x^n \\
\hat{Y}' &= \hat{v}_y^n \\
\hat{Z}' &= \hat{v}_z^n - k_1(\hat{Z} - h_b + \hat{h}_{bias})
\end{aligned}
\tag{101}
$$

Now the dynamics is augmented so as to include the biases,

$$
\left[\hat{a}'_{b1} \;\; \hat{a}'_{b2} \;\; \hat{a}'_{b3} \;\; \hat{a}'_{b4} \;\; \hat{a}'_{b5} \;\; \hat{a}'_{b6} \;\; \hat{h}'_{bias}\right]^T = \vec{0}
\tag{102}
$$

All the above state estimate differential equations represent 21 states in the augmented system for six accelerometers. They will be integrated during the propagation phase of the extended Kalman filter by a Runge-Kutta-Fehlberg 5^{th} order algorithm.

4.5 State Estimate Propagation for the Twelve Accelerometer I.M.U.

Although now the system has 12 accelerometers, it is not necessary to estimate the 12 accelerometer biases plus one baro-altimeter bias. Even if there were a number of accelerometers greater than 12 (e.g. 48 accelerometers) the number of states in the extended Kalman filter can be kept to a minimum of 21 states. The reason for that is the "C" matrix, which represents the linear combinations of accelerometer outputs required to obtain angular and linear accelerations. The number of rows in the "C" matrix is consequently always 6 corresponding to the three components of angular acceleration and the other three relative to the specific force measurements. The number of columns will be the same as the number of accelerometers.

The state estimate equations will be basically the same as in the previous section except for the fact that the specific forces and angular accelerations will be considered to be inputs to the system instead of the accelerometers signals, which are pre-processed.

The quaternion state estimate dynamics is

$$\begin{aligned}
\hat{a}' &= -\frac{1}{2}\left(\hat{b}\hat{\omega}_x+\hat{c}\hat{\omega}_y+\hat{d}\hat{\omega}_z\right)\\
\hat{b}' &= \frac{1}{2}\left(\hat{a}\hat{\omega}_x+\hat{c}\hat{\omega}_z-\hat{d}\hat{\omega}_y\right)\\
\hat{c}' &= \frac{1}{2}\left(\hat{a}\hat{\omega}_y-\hat{b}\hat{\omega}_z+\hat{d}\hat{\omega}_x\right)\\
\hat{d}' &= \frac{1}{2}\left(\hat{a}\hat{\omega}_z+\hat{b}\hat{\omega}_y-\hat{c}\hat{\omega}_x\right)
\end{aligned} \tag{103}$$

and the angular velocity dynamics now are

$$\begin{aligned}
\hat{\omega}_x' &= \alpha_x^b-\hat{\alpha}_{x_{bias}}\\
\hat{\omega}_y' &= \alpha_y^b-\hat{\alpha}_{y_{bias}}\\
\hat{\omega}_z' &= \alpha_z^b-\hat{\alpha}_{z_{bias}}
\end{aligned} \tag{104}$$

Note that now a generalized angular acceleration bias is being estimated instead of the individual accelerometer biases. They are the second terms on the right hand side of the above equations.

The same procedure will be followed in the velocity and position dynamics leading to,

$$\begin{bmatrix}\hat{v}_x^{n\prime}\\ \hat{v}_y^{n\prime}\\ \hat{v}_z^{n\prime}\end{bmatrix}=\begin{bmatrix}\hat{C}_{xx} & \hat{C}_{xy} & \hat{C}_{xz}\\ \hat{C}_{yx} & \hat{C}_{yy} & \hat{C}_{yz}\\ \hat{C}_{zx} & \hat{C}_{zy} & \hat{C}_{zz}\end{bmatrix}\begin{bmatrix}f_x^b-\hat{f}^b_{x_{bias}}\\ f_y^b-\hat{f}^b_{y_{bias}}\\ f_z^b-\hat{f}^b_{z_{bias}}\end{bmatrix}-\begin{bmatrix}0\\0\\g_z\end{bmatrix}-k_2\begin{bmatrix}0\\0\\(\hat{z}^n-h_b+\hat{h}_{bias})\end{bmatrix}-\begin{bmatrix}0\\0\\\hat{h}_{aux}\end{bmatrix} \tag{105}$$

Once again the individual accelerometer biases will not be estimated but rather their net result in terms of specific forces measurements. This can be seen in the vector of differences that is being multiplied by the estimated transformation matrix. The estimated transformation matrix is to be determined in the same way as in the previous section,

$$\hat{C}_B^N=\begin{bmatrix}\hat{a}^2+\hat{b}^2-\hat{c}^2-\hat{d}^2 & 2(\hat{b}\hat{c}-\hat{a}\hat{d}) & 2(\hat{a}\hat{c}+\hat{b}\hat{d})\\ 2(\hat{a}\hat{d}+\hat{b}\hat{c}) & \hat{a}^2-\hat{b}^2+\hat{c}^2-\hat{d}^2 & 2(\hat{c}\hat{d}-\hat{a}\hat{b})\\ 2(\hat{d}\hat{b}-\hat{a}\hat{c}) & 2(\hat{a}\hat{b}+\hat{c}\hat{d}) & \hat{a}^2-\hat{b}^2-\hat{c}^2+\hat{d}^2\end{bmatrix} \tag{106}$$

and the auxiliary variable is also the same,

$$\hat{h}_{aux}^{\prime}=k_3(\hat{z}-h_b+\hat{h}_{bias}) \tag{107}$$

as well as the position,

$$\begin{aligned}\hat{x}^{\prime}&=\hat{v}_x^n\\ \hat{y}^{\prime}&=\hat{v}_y^n\\ \hat{z}^{\prime}&=\hat{v}_z^n-k_1(\hat{z}-h_b+\hat{h}_{bias})\end{aligned} \tag{108}$$

Now the bias terms will augment the system in the following way,

$$\frac{d}{dt}\left[\hat{f}^b_{x_{bias}}\ \hat{f}^b_{y_{bias}}\ \hat{f}^b_{z_{bias}}\ \hat{a}^b_{x_{bias}}\ \hat{a}^b_{y_{bias}}\ \hat{a}^b_{z_{bias}}\ \hat{h}_{bias}\right]^T=\vec{0} \tag{109}$$

Note that now the state estimate vector differs from the previous case in the sense that it does not reflect the physical hardware, in this case, the number of accelerometers,

$$\hat{x}=\left[\hat{a}\ \hat{b}\ \hat{c}\ \hat{d}\ \hat{\omega}_x\ \hat{\omega}_y\ \hat{\omega}_z\ \hat{v}_x\ \hat{v}_y\ \hat{v}_z\ \hat{h}_{aux}\ \hat{X}\ \hat{Y}\ \hat{Z}\ \hat{f}^b_{x_{bias}}\ \hat{f}^b_{y_{bias}}\ \hat{f}^b_{z_{bias}}\ \hat{\alpha}^b_{x_{bias}}\ \hat{\alpha}^b_{y_{bias}}\ \hat{\alpha}^b_{z_{bias}}\ \hat{h}_{bias}\right]^T \tag{110}$$

In order to implement this technique it is necessary to change the Q matrix, which is the process noise intensity matrix. Instead of a 13-by-13 matrix so as to capture the 12 accelerometers plus a baro-altimeter, it will turn out to be a 7-by-7 matrix in order to represent the intensities of 3 angular and 3 linear acceleration biases plus the baro-altimeter bias. The way to do that is simply to consider the Q matrix to be,

$$Q=\begin{bmatrix} \left(C\,\Theta_{0\ accel}\,C^T\right) & 0 \\ 0 & \left(\Theta_{0\ alt}\right) \end{bmatrix} \tag{111}$$

It is possible to recognize that the Q_{11} partition corresponds to accelerometer white noise intensity reflected by the "C" matrix at the angular and linear acceleration measurements.

4.6 Covariance Matrix Propagation

The covariance matrix reflects the uncertainty of each state estimate variable by means of its covariance at the diagonal terms. The initial covariance matrix, at time t_o, should reflect the uncertainty in the knowledge of the states with physical meaning, the covariance of the biases, and arbitrary uncertainties assigned to auxiliary variables or others without physical meaning. It is assumed also that the initial covariance matrix contains only diagonal terms. The off-diagonal or cross-correlation terms will show up in a natural way once the covariance matrix reaches steady-state.

According to [5], the propagation in continuous time of the covariance matrix is characterized by the following differential equation,

$$\dot{P}(t/t_1)=F\left[t;\hat{x}(t/t_1)\right]\times P(t/t_1)+P(t/t_1)\times F^{T}\left[t;\hat{x}(t/t_1)\right]+G(t)\times Q(t)\times G^{T}(t) \tag{112}$$

The F matrix is a 21-by-21 sparse matrix with some terms that are constant and others that are functions of the states. The same applies to the G matrix, which is a 21-by-7 matrix. The G matrix reflects how the modified noise sources α_x, α_y, α_z, f_x^b, f_y^b, and f_z^b affect the system. Here we will present only the 12 accelerometer design case. The derivatives for the six accelerometer I.M.U. are shown in section 5.2.
The variable elements are,

$$F(1\to4,1\to7)=\frac{\partial\left[\hat{a}'\ \hat{b}'\ \hat{c}'\ \hat{d}'\right]^{T}}{\partial\left[\hat{a}\ \hat{b}\ \hat{c}\ \hat{d}\ \hat{\omega}_x\ \hat{\omega}_y\ \hat{\omega}_z\right]^{T}}=\begin{vmatrix} 0 & -.5\hat{\omega}_x & -.5\hat{\omega}_y & -.5\hat{\omega}_z & -.5\hat{b} & -.5\hat{c} & -.5\hat{d} \\ .5\hat{\omega}_x & 0 & .5\hat{\omega}_z & -.5\hat{\omega}_y & .5\hat{a} & -.5\hat{d} & .5\hat{c} \\ .5\hat{\omega}_y & -.5\hat{\omega}_z & 0 & .5\hat{\omega}_x & .5\hat{d} & .5\hat{a} & -.5\hat{b} \\ .5\hat{\omega}_z & .5\hat{\omega}_y & -.5\hat{\omega}_x & 0 & -.5\hat{c} & .5\hat{b} & .5\hat{a} \end{vmatrix} \tag{113}$$

and,

$$F(8\rightarrow10,1\rightarrow4)=\frac{\frac{\partial}{\partial t}\left[\hat{v}_x^n\ \hat{v}_y^n\ \hat{v}_z^n\right]^T}{\partial\left[\hat{a}\ \hat{b}\ \hat{c}\ \hat{d}\right]^T}=2\times\begin{vmatrix}f_x^b\\f_y^b\\f_z^b\end{vmatrix}^T\times\left|\begin{vmatrix}\hat{a}&\hat{b}&-\hat{c}&-\hat{d}\\-\hat{d}&\hat{c}&\hat{b}&-\hat{a}\\\hat{c}&\hat{d}&\hat{a}&\hat{b}\end{vmatrix}\begin{vmatrix}\hat{d}&\hat{c}&\hat{b}&\hat{a}\\\hat{a}&-\hat{b}&\hat{c}&-\hat{d}\\-\hat{b}&-\hat{a}&\hat{d}&\hat{c}\end{vmatrix}\begin{vmatrix}-\hat{c}&\hat{d}&-\hat{a}&\hat{b}\\\hat{b}&\hat{a}&\hat{d}&\hat{c}\\\hat{a}&-\hat{b}&-\hat{c}&\hat{d}\end{vmatrix}\right|^T \quad (114)$$

Now another group of derivatives which belongs to the F matrix is,

$$F(8\rightarrow10,15\rightarrow17)=\frac{\frac{\partial}{\partial t}\left[\hat{v}_x^n\ \hat{v}_y^n\ \hat{v}_z^n\right]^T}{\partial\left[\hat{f}_{x_{bias}}\ \hat{f}_{y_{bias}}\ \hat{f}_{z_{bias}}\right]^T}=-\hat{C}_b^n \qquad (115)$$

The G matrix will have its only variable part equal to the negative of the above equality as follows,

$$G(8\rightarrow10,1\rightarrow3)=\frac{\frac{\partial}{\partial t}\left[\hat{v}_x^n\ \hat{v}_y^n\ \hat{v}_z^n\right]^T}{\partial\left[f_x^b\ f_y^b\ f_z^b\right]^T}=\hat{C}_b^n \qquad (116)$$

In terms of constant blocks of the F matrix ,the following terms are defined;

$$F(5\rightarrow7,18\rightarrow20)=\frac{\frac{\partial}{\partial t}\left[\hat{\omega}_x\ \hat{\omega}_y\ \hat{\omega}_z\right]}{\partial\left[\hat{\omega}_{x_{bias}}^b\ \hat{\omega}_{y_{bias}}^b\ \hat{\omega}_{z_{bias}}^b\right]}=-I^{(3\times3)} \qquad (117)$$

$$F(12,6)=\frac{\partial\hat{X}'}{\partial\hat{v}_x}=F(13,9)=\frac{\partial\hat{Y}'}{\partial\hat{v}_y}==F(14,10)=\frac{\partial\hat{Z}'}{\partial\hat{v}_z}=1 \qquad (118)$$

$$F(10,11)=\frac{\partial\hat{v}_z'}{\partial\hat{h}_{aux}}=k2 \qquad (119)$$

$$F(14,14)=\frac{\partial\hat{z}'}{\partial\hat{z}}=F(14,21)=\frac{\partial\hat{z}'}{\partial\hat{h}_{bias}}=-k_1 \tag{120}$$

$$F(10,14)=\frac{\partial\hat{v}_z'}{\partial\hat{z}}=F(10,21)=\frac{\partial\hat{v}_z'}{\partial\hat{h}_{bias}})=-k_2 \tag{121}$$

$$F(11,14)=\frac{\partial\hat{h}_{aux}'}{\partial\hat{z}}=F(11,21)=\frac{\partial\hat{h}_{aux}}{\partial\hat{h}_{bias}}=k_3 \tag{122}$$

The G matrix will have the following constant terms,

$$G(5\to7,4\to6)=\frac{\partial\left[\hat{\omega}_x'\ \hat{\omega}_y'\ \omega_z'\right]^T}{\partial\left[\alpha_x^b\ \alpha_y^b\ \alpha_z^b\right]^T}=I^{(3\times3)} \tag{123}$$

$$\begin{aligned}G(10,7)&=\frac{\partial\hat{v}_z'}{\partial h_b}=k_2\\ G(11,7)&=\frac{\partial\hat{h}_{aux}'}{\partial h_b}=-k_3\\ G(14,7)&=\frac{\partial\hat{z}'}{\partial h_b}=k_1\end{aligned} \tag{124}$$

All that has been done until now refers to the continuous propagation of the covariance matrix. In the real application there are no continuous measurements though. In this way it is possible to discretize the system. An advantage of this procedure is to reduce the computational burden. If the propagation of the covariance matrix were continuous, it would be necessary to integrate all its elements using a Runge-Kutta routine, which means the integration of 21^2 (=441) states.

According to [6], the discretized covariance matrix will propagate as follows,

$$P(k+1)=\Phi P(k)\Phi^T+\frac{\Gamma R_{wni}\Gamma^T}{\Delta T} \tag{125}$$

In the above equation, Φ and Γ correspond to F and G in the continuous case and ΔT is the sampling time. R_{wni} is the white noise intensity matrix from the continuous case. In order to calculate Φ and Γ, it is necessary to compute Ψ as follows,

$$\Psi=I+\frac{F\Delta T}{2}\left(I+\frac{F\Delta T}{3}\left(\cdots\frac{F\Delta T}{N-1}\left(I+\frac{F\Delta T}{N}\right)\right)\right) \tag{126}$$

In the present case the required numerical precision is reached for N=5. Now Φ and Γ can be computed as,

$$\begin{aligned}\Phi&=I+F\,\Delta T\,\Psi\\ \Gamma&=\Psi\,\Delta T\,G\end{aligned} \tag{127}$$

The covariance matrix must be always symmetric, but the numerical integration can affect this characteristic. In order to enforce this property after each propagation it is necessary to proceed as follows,

$$P_{sym}=\frac{(P_{old}+P_{old}^T)}{2} \tag{128}$$

4.7 Pseudo-Noise Addition and Quaternion Normalization

Pseudo-noise addition is used in this work to improve the Kalman gain in the attitude determination. Many simulations have shown that the attitude determination was the most sensitive part of the system. When pseudo-noise is added to the quaternion dynamics, the extended Kalman algorithm will give more weight to the external aiding in order to obtain a better attitude estimate. It is equivalent to describing the quaternion dynamics as,

$$
\begin{aligned}
a' &= -\frac{1}{2}(b\omega_x + c\omega_y + d\omega_z) + w_a \\
b' &= \frac{1}{2}(a\omega_x + c\omega_z - d\omega_y) + w_b \\
c' &= \frac{1}{2}(a\omega_y - b\omega_z + d\omega_x) + w_c \\
d' &= \frac{1}{2}(a\omega_z + b\omega_y - c\omega_x) + w_d
\end{aligned}
\tag{129}
$$

The intensity of the white noise w_a, w_b, w_c and w_d were made equal to 1×10^{-9} for simulation purposes during the flight phase of the hypothetical vehicle.

After each update cycle and after some integration steps it is necessary to normalize the quaternion. When pseudo-noise is added to the quaternions, this requirement turns out to be mandatory, because the magnitude of the quaternion, which should be always 1, becomes a stochastic variable as well. To do that the following procedure holds,

$$
q_{norm} = [a \; b \; c \; d]^T = \frac{q_{old}}{\sqrt{q_{old}^T \times q_{old}}}
\tag{130}
$$

The normalization procedure also makes the transformation matrix C_b^n orthonormal, otherwise the Euler angles could be even complex numbers, which does not correspond to reality.

The addition of pseudo-noise also has implications in the Q matrix, which reflects the generalized white noise intensities. Now the Q matrix has to be augmented in order to include four pseudo-noises. Instead of a 7-by-7 matrix, the Q matrix now will be 11-by-

11 as follows,

$$Q_{aug} = \begin{vmatrix} \begin{vmatrix} (C\,\Theta_{0\,accel}\,C^T) & \underline{0} \\ \underline{0} & (\Theta_{0\,alt}) \end{vmatrix} & \underline{0} \\ \underline{0} & \Theta_{psnoise} \times I^{(4\times4)} \end{vmatrix} \qquad (131)$$

The G matrix is also changed so as to reflect the four pseudo noise terms added to the quaternion dynamics. Instead of being a 21-by-7 matrix, during the flight phase it will be a 21-by-11 matrix with the following non-zero additional terms,

$$\begin{aligned} G(1,8) &= \frac{\partial\hat{a}'}{w_a} = 1 \\ G(2,9) &= \frac{\partial\hat{b}'}{w_b} = 1 \\ G(3,10) &= \frac{\partial\hat{c}'}{w_c} = 1 \\ G(4,11) &= \frac{\partial\hat{d}'}{w_d} = 1 \end{aligned} \qquad (132)$$

4.8 External Aiding during Initialization

The main objective of the initialization phase in any inertial navigation system is to provide the bias compensation. In this particular system it is impossible in practical terms to determine the bias estimates during flight because of their heavy effect on the attitude computation .

It is assumed that there exists a high quality external inertial measurement unit which is able to provide the vehicle's system its attitude, angular and linear velocity and position. The H matrix characterizes which states are measured from the external source of information. In this case H is a 13-by-21 matrix because there are 13 external measurements (the z vector) and 21 states. The H matrix is as follows,

$$H_{inic} = \left| \begin{vmatrix} I^{(4x4)} \\ 0^{(9x4)} \end{vmatrix} \begin{vmatrix} 0^{(4x3)} \\ I^{(3x3)} \\ 0^{(6x3)} \end{vmatrix} \begin{vmatrix} 0^{(7x3)} \\ I^{(3x3)} \\ 0^{(3x3)} \end{vmatrix} \begin{vmatrix} 0^{(13x1)} \end{vmatrix} \begin{vmatrix} 0^{(10x3)} \\ I^{(3x3)} \end{vmatrix} \begin{vmatrix} 0^{(13x7)} \end{vmatrix} \right| \tag{133}$$

The R matrix represents the white measurement noise. In this case it is assumed that all measurement noises have the same intensity. For simulation purposes the R matrix is made equal to $1x10^{-24}$ x $I^{(13,13)}$.

4.9 External Aiding during Take-off and Flight

After all biases have been compensated during the initialization phase, the vehicle takes off and later assumes a straight and level flight path. During the boost phase the vehicle is assumed to reach an acceleration close to 10g.

At this point the second phase begins and now the system will receive external aiding from the GPS satellite system. The Kalman filter will begin to operate after the boost phase because the high acceleration would cause either loss of phase or loss of code in the GPS receiver. In this way in the beginning of the trajectory the system would navigate without external aiding.

Now in the flight phase only six states will be measured from the external source of information, the velocity and position of the vehicle. It is assumed the velocity and position measurement vector correspond to the same reference point with respect to which the inertial measurements are being made.

Under these circumstances the H matrix can be expressed as follows,

$$H_{flight} = \left| \, |0^{(6x7)}| \begin{vmatrix} I^{(3x3)} \\ 0^{(3x3)} \end{vmatrix} |0^{(6x1)}| \begin{vmatrix} 0^{(3x3)} \\ I^{(3x3)} \end{vmatrix} |0^{(6x7)}| \, \right| \tag{134}$$

4.10 Open-Loop Condition

In order to evaluate the performance of the 12 accelerometer inertial measurement unit with respect to the 6 accelerometer I.M.U., the GPS information loss will be simulated after both extended Kalman filters reach steady state. The performance criterion is the capacity of keeping the attitude information within a error of ∓2 degrees in terms of Bank angle, Heading and Pitch angle.

After the extended Kalman filter navigator reaches steady state, both configurations, with 6 or 12 accelerometers, will be submitted to a 360 degrees turn without banking and without external aiding as will be discussed in the next chapter.

Chapter 5

THE SIMULATION OF THE SYSTEM

5.1 The Trajectory of the Vehicle

In order to evaluate the performance of both the navigation system and the extended Kalman filter it is necessary to do a stochastic simulation of the complete system. The nominal trajectory of the vehicle was designed to represent as close as possible the initialization, launching phase and maneuver of the hypothetical vehicle until it reaches a straight and level flight under cruise conditions. After the extended Kalman filter reaches steady-state, the GPS loss is simulated and the vehicle is subjected to a complete turn of 360 degrees in order to visualize the open loop stability characteristics of both the six and the twelve accelerometer systems. The Figure below illustrates the different phases of the simulated flight.

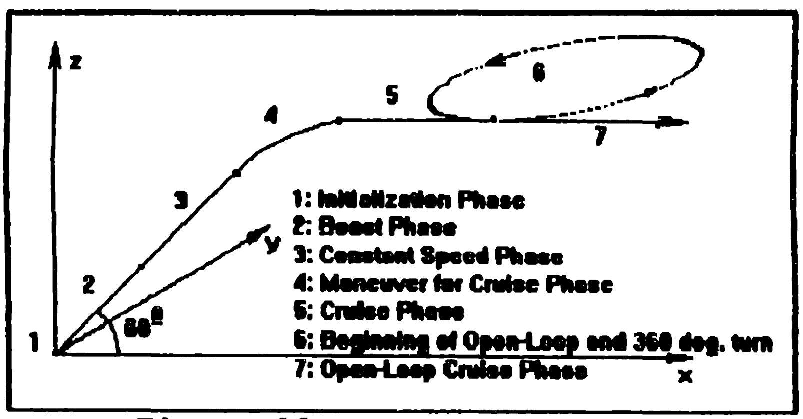

Figure 18:Simulation Phases

5.2 The Specific Force Simulation

The next step consists of simulating what each accelerometer measures as a function of time. To simulate each accelerometer output it is necessary to refer to the following equations,

$$
\begin{array}{lll}
a_1=f_x-d_1\omega_z'+d_1\omega_x\omega_y & a_5=f_y+d_5\omega_z'+d_5\omega_x\omega_y & a_9=f_z-d_9\omega_y'+d_9\omega_x\omega_z \\
a_2=f_x+d_2\omega_z'-d_2\omega_x\omega_y & a_6=f_y-d_6\omega_z'-d_6\omega_x\omega_y & a_{10}=f_z+d_{10}\omega_y'+d_{10}\omega_x\omega_z \\
a_3=f_x+d_3\omega_y'+d_3\omega_x\omega_z & a_7=f_y-d_7\omega_x'+d_7\omega_y\omega_z & a_{11}=f_z+d_{11}\omega_x'+d_{11}\omega_y\omega_z \\
a_4=f_x-d_4\omega_y'-d_4\omega_x\omega_z & a_8=f_y+d_8\omega_x'-d_8\omega_y\omega_z & a_{12}=f_z-d_{12}\omega_x'-d_{12}\omega_y\omega_z
\end{array}
\tag{135}
$$

During the initialization the vehicle is at the launching pad with a pitch angle of 60 degrees for nearly 60 to 120 seconds. The only specific forces that are being measured correspond to the gravity acceleration as follows,

$$
\begin{array}{l}
f_x^b=g\sin\theta \\
f_z^b=g\cos\theta
\end{array}
\tag{136}
$$

so that $a_1=a_2=a_3=a_4=$ g sin θ and $a_9=a_{10}=a_{11}=a_{12}=$ g cos θ.

During the boost phase the vehicle will be submitted to an acceleration of 50 m/s^2 during 5 seconds so as to drive it to a velocity of 250 m/sec. Now the specific forces are,

$$
\begin{array}{l}
f_x^b=g\sin\theta+50m/s^2 \\
f_z^b=g\cos\theta
\end{array}
\tag{137}
$$

The accelerometer outputs will be ,

$$
\begin{array}{l}
a_1=a_2=a_3=a_4=g\ \sin\theta+50m/s^2 \\
a_9=a_{10}=a_{11}=a_{12}=g\ \cos\theta
\end{array}
\tag{138}
$$

The next phase corresponds to a steady speed flight in which the vehicle continues to gain altitude and keep its pitch angle of 60 degrees. Corresponding to the assumption of constant velocity, the specific force is taken to be the negative of the gravitational acceleration as it is in the initialization phase.

During the maneuver phase it is necessary to simulate three dynamic effects. The first parameter to be fixed is the time of maneuver which in this case is set to be 8 seconds. Knowing the time of maneuver it is possible to calculate the angular velocity which will be kept constant until the end of the maneuver. In the beginning of the maneuver the vehicle will be subjected to an acceleration pulse which will confer the calculated angular velocity to it. The duration of the angular acceleration pulse is equal to the sampling interval. In this very moment the vehicle is also subjected to half of the centrifugal acceleration plus the gravity effect. The beginning of the maneuver can be summarized as follows. The angular velocity and acceleration will be,

$$\omega_y^b = \frac{\theta_0}{T_{maneuver}} \qquad (139)$$
$$\alpha_y = \omega_y' = \frac{\omega_y}{\Delta T}$$

and the specific forces can be described as,

$$\vec{f}^{\,b} = -\vec{g} + \vec{f}_{\alpha_y} + \vec{f}_{\frac{\omega_y x v_z^b}{2}} \qquad (140)$$

where -g is the negative of the gravity acceleration, $f_{\alpha y}$ is the specific force due to the angular acceleration pulse and the last term $f_{(\omega y.v)/2}$ is one half of the centrifugal acceleration during this sampling interval. The accelerometer outputs will be as follows,

$$a_1=a_2=g\sin\theta \quad a_3=g\sin\theta+d_3\alpha_y \quad a_4=g\sin\theta-d_4\alpha_y$$
$$a_9=-\frac{\omega_y v_x^b}{2}+g\cos\theta-d_9\alpha_y \quad a_{10}=-\frac{\omega_y v_x^b}{2}+g\cos\theta+d_{10}\alpha_y \tag{141}$$
$$a_{11}=a_{12}=-\frac{\omega_y v_x^b}{2}+g\cos\theta;$$

After this moment, until the end of the maneuver, there will be no angular acceleration meaning that the vehicle will have constant pitch rate until it levels at $\theta=0$. The only specific forces to be simulated will be the centrifugal acceleration and the gravity as below,

$$\vec{f}^b(t)=f_{\omega_y\times v_x^b}(t)-\vec{g}(t) \tag{142}$$

and the accelerometer outputs will be,

$$\begin{aligned} a_1=a_2=a_3=a_4&=g\sin(\theta_0-\omega_y(t-t_{man})) \\ a_9=a_{10}=a_{11}=a_{12}&=-\omega_y\, v_x^b+g\cos(\theta_0-\omega_y(t-t_{man})) \end{aligned} \tag{143}$$

where $t-t_{man}$ is the difference between the current time and the time of the beginning of the maneuver.

At the end of the maneuver another angular acceleration pulse is generated in order to bring the angular velocity down to zero. This angular acceleration pulse is the negative of the first one. Besides this pulse, half of the centrifugal acceleration and the gravity acceleration are again applied to the system so that the specific force will be,

$$\vec{f}^b=-\vec{g}-\vec{f}_{\alpha_y}+\vec{f}_{\frac{\omega_y\times v_x^b}{2}} \tag{144}$$

and during the sampling interval that corresponds to the end of the maneuver the accelerometer outputs will be held as,

$$a_3=-d_3\alpha_y \quad a_4=d_4\alpha_y$$
$$a_9=-\frac{\omega_y v_x^b}{2}+g+d_9\alpha_y \quad a_{10}=-\frac{\omega_y v_x^b}{2}+g-d_{10}\alpha_y \quad a_{11}=a_{12}=-\frac{\omega_y v_x^b}{2}+g \tag{145}$$

After the end of the maneuver the vehicle will execute a straight and level flight until the extended Kalman filter reaches steady-state. During this time the nominal accelerometer readings will be,

$$a_9=a_{10}=a_{11}=a_{12}=g \tag{146}$$

Once the extended Kalman filter reaches steady-state, meaning that the covariance matrix assumes a constant H_2-norm, the GPS data loss will be simulated. This is what is called the open-loop condition or navigating without external aiding. Under these circumstances a complete turn (360 degrees) of the vehicle will be simulated. The purpose of this last maneuver is to compare the stability behavior of the 6 with respect to 12 accelerometer system. The GPS loss is simulated simply by making the update time very large and plotting the propagation phase.

The duration of the turn must be fixed as a parameter. In the simulation this turn will be completed in 60 seconds. Knowing that, it is possible to compute the angular velocity and the angular acceleration pulse as being,

$$\begin{gathered}\omega_z^b=\frac{2\pi}{T_{turn}}\\ \alpha_z^b=\frac{\omega_z}{\Delta T}\end{gathered} \tag{147}$$

and the accelerometer outputs in the beginning of the turn will be,

$$\begin{gathered}a_1=-d_1\alpha_z \quad a_2=d_2\alpha_z\\ a_5=d_5\alpha_z+\frac{\omega_z\times V_x^b}{2} \quad a_6=-d_6\alpha_z+\frac{\omega_z\times V_x^b}{2}\\ a_7=a_8=\frac{\omega_z\times V_x^b}{2}\\ a_9=a_{10}=a_{11}=a_{12}=g\end{gathered} \tag{148}$$

At the next sampling time there will be no angular acceleration, consequently the angular velocity will be kept constant. The centrifugal acceleration will have its entire magnitude of ($\omega_z x v_x^b$) as can be seen below,

$$\begin{gathered} a_5=\omega_z \times v_x^b \quad a_6=\omega_z \times v_x^b \\ a_7=a_8=\omega_z \times v_x^b \\ a_9=a_{10}=a_{11}=a_{12}=g \end{gathered} \tag{149}$$

At the end of the turn there will be another angular acceleration pulse so as to stop the turn and resume the straight flight and the centrifugal acceleration will be reduced to half of its value. This leads to the following accelerometer outputs.

$$\begin{gathered} a_1=d_1\alpha_z \quad a_2=-d_2\alpha_z \\ a_5=-d_5\alpha_z+\frac{\omega_z \times v_x^b}{2} \quad a_6=d_6\alpha_z+\frac{\omega_z \times v_x^b}{2} \\ a_7=a_8=\frac{\omega_z \times v_x^b}{2} \\ a_9=a_{10}=a_{11}=a_{12}=g \end{gathered} \tag{150}$$

After the turn is over, the specific forces will be simply represented by the action of gravity,

$$a_9=a_{10}=a_{11}=a_{12}=g \tag{151}$$

All that has been done so far represents the nominal accelerometer outputs. In fact each of these outputs will have a random bias with normal distribution, zero-mean and standard deviation of 35μg and a discrete white noise process represented as random numbers with normal distribution, zero mean and variance equal to the white noise intensity divided by the sampling time ΔT. The extended Kalman filter will obtain the best estimate of the nominal outputs until the system reaches steady-state and the GPS loss occurs, but the so far estimated bias will always be subtracted out of the accelerometer outputs.

5.3 External I.M.U., GPS and Baro-Altimeter Simulation

The external I.M.U. is what will make the initialization possible. In this work it will be simply an external source of the quaternion, angular and linear velocities and position. All measurements will be contaminated with a discrete white noise sequence whose standard deviation magnitude is 1×10^{-12}.

Once the specific forces are determined as functions of time for each of the twelve accelerometers, each phase can be integrated to obtain the nominal velocities and position as functions of time also. These calculations will not be presented in this section because they are implemented in the code as will be shown in the next section. With the nominal trajectory and velocity it is possible to simulate the GPS and baro-altimeter signals for every sampling time if necessary.

The external I.M.U. and the GPS information are simulated at each update cycle of the extended Kalman filter and the baro-altimeter is simulated at each sampling time because it is integrated with the twelve accelerometers for the vertical channel stabilization.

5.4 Software Implementation

The simulation was implemented using MATLAB™ version 3.5k on a Personal Computer with a 386/387 Intel™ CPU. Two codes were designed: The first one simulates the 12 accelerometer inertial measurement unit; and the second one corresponds to the six accelerometer I.M.U.. The GPS loss is simulated by making the update interval of the Kalman filter very large while storing data from the Runge-Kutta-Fehlberg subroutine, which plays the role of the navigator.

The routine FAIMU5a.M is the code for the 12 accelerometer I.M.U as follows:

```
% 16.ThG Thesis
% Routine FAIMU5.m
% Simulation of a twelve accelerometer Inertial Measurement Unit
% This routine simulates a twelve accelerometer Inertial
% Measurement
% Unit driven by white noise with bias, stabilized by a % %   %
% barometric-
% altimeter and an Extended Kalman Filter using GPS measurements.
% It estimates the navigation state instead of the error state.
% The input for each accelerometer will be a normally distributed
% white noise of (10 micro-g)² mean squared output error, plus a
% bias of 35 micro-g standard deviation.
format short e

% Global variables
global  Cbn FF PP QQ GG hb
global RR C F Fh dW deltaT


% Lay-out of accelerometers
d1=1.2;d2=d1;
d3=.55;d4=d3;
d5=3.6;
d6=.48;
d7=.6;d8=d7;
d9=3.55;
d10=.43;
d11=1.25;d12=d11;

% Gravity
gz=9.80665;

% Vertical Channel Gains
k1=6e-2;k2=1.2e-3;k3=8e-6;

% Accelerometer output coefficients matrix
C=zeros(6,12);
C(1,1:4)=1/4*ones(1,4);
C(2,5)=d6/(2*(d5+d6));C(2,6)=d5/(2*(d5+d6));C(2,7:8)=1/4*[1 1];
C(3,9)=d10/(2*(d9+d10));C(3,10)=d9/(2*(d9+d10));C(3,11:12)=1/4*[1
1];
C(4,7:8)=1/(4*d7)*[-1 1];C(4,11:12)=1/(4*d11)*[1 -1];
C(5,1:2)=(d9-d10)/(8*(d3*d10+d4*d9))*[1 1];
C(5,3)=(d9+3*d10)/(8*(d3*d10+d4*d9));
C(5,4)=-(3*d9+d10)/(8*(d3*d10+d4*d9));
C(5,9:10)=1/(2*(d3*d10+d4*d9))*[-d4 d3];
C(6,1)=-(d5+3*d6)/(8*(d1*d6+d2*d5));
C(6,2)=(d6+3*d5)/(8*(d1*d6+d2*d5));
C(6,3:4)=(d6-d5)/(8*(d1*d6+d2*d5))*[1 1];
```

```
C(6,5:6)=1/(2*(d1*d6+d2*d5))*[d2 -d1];


% Description of State Variables:

% State Estimate Dynamics:
% ah(1) bh(2) ch(3) dh(4)      (Quaternion estimates)
% Wxh(5) Wyh(6) Wzh(7)         (Angular Velocities)
% Vxh(8) Vyh(9) Vzh(10)         (Velocities in the Navigation
Frame)
% hah(11) xh(12) yh(13) zh(14)(Auxiliary Variable and Position
Estimates
%                               in the Navigation Frame)
% (dUib)i=1:12(15:20)          (Linear and angular accel. biases)
% hbias(27)                    (Baro-altimeter error modelling)


% The interval of integration is one second and  the  5th order
%Runge- Kutta-Fehlberg is used:

% Input data
tsim=0;

% Initial elevation at launching platform
elev=input('Launching elevation  [60 degrees]')
if isempty(elev),
elev=60
end
theta=elev*pi/180;

% Boost acceleration
ba=input('Boost acceleration [50 m/s2]')
if isempty(ba),
ba=50
end

% Sampling Time
deltaT=input('Sampling time [0.025 sec.]')
if isempty(deltaT),
deltaT=.025
end

% Update interval UT for initialization phase
UTI=input('Update interval for initialization [UTI=deltaT]')
if isempty(UTI),
UTI=deltaT
end

% Update interval for the cruise phase
UTC=input('Update interval for cruise [5 sec.]')
if isempty(UTC),
UTC=5
```

```
end

% Initialization time
tba=input('Initialization Time [120 sec.]')
if isempty(tba),
tba=120
end

% End of acceleration
da=input('Duration of Boost Phase [5 sec.]')
if isempty(da),
da=5
end
tbc=tba+da;
% Begin of pitch maneuver
tt1=tbc+16;
tm=input(['Begin of pitch maneuver [',num2str(tt1),' sec.]'])
if isempty(tm),
tm=tt1
end
% End of Pitch Maneuver
tt2=tm+8;
tc=input(['End of pitch Maneuver [',num2str(tt2),' sec.]'])
if isempty(tc),
tc=tt2
end

% Angular speed during maneuver (wy)
wy=theta/(tc-tm);
% Angular accelerations at "tm" and "tc"
dwy=wy/deltaT;

% Moment of GPS loss
tloss=input(['Moment of GPS loss [',num2str(tc+50*UTC),'sec.]'])
if isempty(tloss)
tloss=tc+50*UTC
end

% Beginning of the turn
TOLoop=input(['Beginning of the turn
[',num2str(tloss+10),'sec]'])
if isempty(TOLoop)
TOLoop=tloss+10
end

% Duration of the turn
tturn=input('Duration of the turn [60 sec.]: ')
if isempty(tturn),
tturn=60
end
```

This is the calculation of the angular velocity and angular

acceleration during the 360 deg. turn

```
wz=2*pi/tturn;
dwz=wz/deltaT;

% Time of simulation in Seconds TSIM
tt3=TOLoop+tturn;
TSIM =input(['Time of simulation [',num2str(tt3),' sec.]'])
if isempty(TSIM),
TSIM=tt3
end

% History
XT=zeros(7,(TSIM/(4*deltaT)));
xt=zeros(3,(TSIM/(4*deltaT)));
jjj=1;
jj=1;


% Initial Conditions:
xi=zeros(21,1);
% Quaternions:
```

Note that the angle theta should have a negative sign but it is kept positive so as to agree with the GPS, the baro-altimeter and the specific force formulation.

```
xi(1)=cos(theta/2);
xi(3)=-sin(theta/2);

% Initial Direction Cosine Matrix
Cbn(1,1)= xi(1)^2 + xi(2)^2 - xi(3)^2 - xi(4)^2;
Cbn(1,2)= 2*(xi(2)*xi(3) - xi(1)*xi(4));
Cbn(1,3)= 2*(xi(1)*xi(3) + xi(2)*xi(4));
Cbn(2,1)= 2*(xi(1)*xi(4) + xi(2)*xi(3));
Cbn(2,2)= xi(1)^2 - xi(2)^2 +xi(3)^2 -xi(4)^2;
Cbn(2,3)= 2*(xi(3)*xi(4) - xi(1)*xi(2));
Cbn(3,1)= 2*(xi(4)*xi(2) - xi(1)*xi(3));
Cbn(3,2)= 2*(xi(1)*xi(2) + xi(3)*xi(4));
Cbn(3,3)= xi(1)^2 - xi(2)^2 - xi(3)^2 + xi(4)^2;

% Initial Covariance Matrix
```

This part of the code will define the initial uncertainty in the knowledge of the states when the vehicle is at the launching pad.

```
PP=zeros(21);
% Quaternions;
qu=input('Quaternions uncertainty [1e-3]')
if isempty(qu),
qu=1e-3
end
```

```
PP(1:4,1:4)=qu^2*eye(4);
% Angular Velocity rad/sec;
wu=input('Ang. Vel. uncertainty [1e-3]')
if isempty(wu),
wu=1e-3
end
PP(5:7,5:7)= wu^2*eye(3);
% Velocities in the Navigation Frame;
vu=input('Velocity uncertainty [1e-6]')
if isempty(vu),
vu=1e-6
end
PP(8:10,8:10)=vu^2*eye(3);
% Auxiliary Variable dha;
PP(11,11)= (1e-6)^2;
% Position
pu=input('Position uncertainty [1e-3]')
if isempty(pu),
pu=1e-3
end
PP(12:14,12:14)=pu^2*eye(3);
% Accelerometer Biases ;
abu=input('Accel.bias uncertainty [35e-6*gz]')
if isempty(abu),
abu=35e-6*gz
end
PP(15:20,15:20)=abu^2*C*C';
% Baro-Altimeter error bias;
% hbias;
hbu=input('Baro-altimeter bias uncertainty [.02]')
if isempty(hbu),
hbu=.02
end
PP(21,21)= hbu^2;



% Accelerometer White Noise Intensity
```

At this point the accelerometer white noise intensity is calculated from its mean squared output error.

```
msa=input('Mean Squared Accelerometer Output [(10e-6*gz)^2] ')
if isempty(msa),
msa=(10e-6*gz)^2
end
tau=1e-3;
awni=2*tau*msa

% Barometer White Noise Intensity
```

Here the baro-altimeter is given a pseudo noise which is assumed

to have the same magnitude of that of the accelerometer.

```
bwni=input('Barometer White Noise Intensity [2e-11] ')
if isempty(bwni),
bwni=2e-11
end
```

Pseudo-noise is also added to the quaternion dynamics in order to improve the Kalman gain for the first four state variables.

```
psnoise=input('Pseudo-Noise to Attitude Dynamics [1e-9]')
if isempty(psnoise),
psnoise=1e-9
end

% All other states are set to zero.

% Constant Extended Kalman Filter Matrices
```

At that point the the process noise intensity matrix is constructed so that the first block "QQI1" corresponds to the noise effect in terms of angular and linear accelerations.

```
% Process noise intensity Matrix QQ (Initialization)
QQI1=awni*C*C';
QQI2=zeros(6,1);
QQI3=zeros(1,6);
QQI4=bwni;
QQI=[QQI1 QQI2;QQI3 QQI4];

% Addition of pseudo-noise to quaternions dynamics and to angular
% velocity dynamics (Flight)
```

For the flight phase the addition of pseudo-noise to the quaternion dynamics is taken in to account.

```
QQF1=QQI;
QQF2=zeros(7,4);
QQF3=zeros(4,7);
QQF4=psnoise*eye(4);
QQF=[QQF1 QQF2;QQF3 QQF4];

% Error Dynamic Output (Measurement) Matrix HH1(for
initialization phase)
HH1=zeros(13,21);
HH1(:,1:4)=[eye(4);zeros(9,4)];
HH1(:,5:7)=[zeros(4,3);eye(3);zeros(6,3)];
HH1(:,8:10)=[zeros(7,3);eye(3);zeros(3)];
HH1(:,12:14)=[zeros(10,3);eye(3)];

% Error Dynamic Output (Measurement) Matrix HH2 (for flight
phase)
HH2=zeros(6,21);
HH2(:,8:10)=[eye(3);zeros(3)];
```

```
HH2(:,12:14)=[zeros(3);eye(3)];


% Measurement Covariance Matrix RR1
```

It is assumed that all elements of the measurement vector during initialization have a small amount of noise.

```
uri=input('Initialization Measurement Uncertainty [1e-12]')
if isempty(uri),
uri=1e-12
end
RR1=uri^2*eye(13);


% Measurement Covariance Matrix RR2
urv=input('GPS Vel.Meas. Uncertainty [.1 m/s]')
if isempty(urv),
urv=.1
end
urp=input('GPS Pos.Meas. Uncertainty [15 m]')
if isempty(urp),
urp=15
end
RR2=[urv^2*eye(3) zeros(3);zeros(3) urp^2*eye(3)];

prt=input('Enter 1 to print plots using GPP.M ')

% Dimensioning of the FF Matrix
```

Here are the constant terms of the "F" matrix

```
FF=zeros(21);
FF(5:7,18:20)=-eye(3);
FF(10,11)=-1;FF(10,14)=-k2;FF(10,21)=-k2;
FF(11,14)=k3;FF(11,21)=k3;
FF(12,8)=1;
FF(13,9)=1;
FF(14,10)=1;FF(14,14)=-k1;FF(14,21)=-k1;

% Error Dynamic Input Matrix GG   (Constant Coefficients only)
% Initialization phase
```

This is the "G" matrix during the initialization phase.

```
GGI=zeros(21,7);
GGI(5:7,4:6)=eye(3);
GGI(10,7)=k2;
GGI(11,7)=-k3;
GGI(14,7)=k1;

% Flight phase
```

During the flight phase it is necessary to take into account the addition of pseudo-noise to the quaternion dynamics.

```
GGF=zeros(21,11);
GGF(5:7,4:6)=eye(3);
GGF(10,7)=k2;
GGF(11,7)=-k3;
GGF(14,7)=k1;
% Addition of pseudo-noise to quartenion
GGF(1,8)=1;GGF(2,9)=1;GGF(3,10)=1;GGF(4,11)=1;


% Accelerometer bias generation
```

This is the generation of the accelerometer random biases.

```
rand('normal')
bias=abu*rand(12,1);

% Altimeter bias generation
hbias=hbu*rand(1);

% Simulation
% Initialization phase
UT=UTI;
HH=HH1;
RR=RR1;
QQ=QQI;
GG=GGI;

% First bias generated specific force estimate
Fh=zeros(3,1);

% Auxiliary Variables
```

This corresponds to the beginning of the simulation.

```
sim=1;

while sim == 1

% Update Phase
% Kalman Gain Calculation
```

This forces the covariance matrix to be allways symmetric.

```
PP=(PP+PP')/2;

if tsim < tba,
% Initialization phase

KK=PP*HH'*inv(HH*PP*HH'+RR);
```

```
% Alignment Measurement Generation
xalig=uri*rand(13,1);
xalig(1)=cos(theta/2)+uri*rand(1);
xalig(3)=-sin(theta/2)+uri*rand(1);

% Measurement
% For full state estimation
z=xalig;

% For nominal trajectory simulation
xn=0;
hn=0;


elseif(tsim >= tba),
% Flight phase

UT=tbc-tba;
if tsim>tbc,
UT=UTC;
end

% To run Open Loop after Steady State
```

If the condition below is satisfied, then the update time will be made equal to the complete time of simulation, meaning the system will not receive external aiding. In this way the loss of the GPS information will be simulated and the extended Kalman filter will only execute the propagation phase until the program reaches its end.

```
if(tsim>tloss)
N1='GPS Loss';
disp(N1)
% pause
UT=TSIM;
end

QQ=QQF;
GG=GGF;
HH=HH2;
RR=RR2;

% Kalman Gain Calculation
KK=PP*HH'*inv(HH*PP*HH'+RR);

% GPS Measurement simulation

if(tsim<tba)
 vxn=0;
 vzn=0;
 hn=0;
```

```
 xn=0;

elseif (tsim>=tba & (tbc>=tsim)),
% phase1
 vxn=ba*(tsim-tba)*cos(theta);
 vzn=ba*(tsim-tba)*sin(theta);
 hn=.5*ba*(tsim-tba)^2*sin(theta);
 xn=.5*ba*(tsim-tba)^2*cos(theta);

elseif ((tsim>tbc) & (tm>=tsim)),
% phase2
 vxn=ba*(tbc-tba)*cos(theta);
 vzn=ba*(tbc-tba)*sin(theta);
 hn=(.5*ba*(tbc-tba)^2+ba*(tbc-tba)*(tsim-tbc))*sin(theta);
 xn=(.5*ba*(tbc-tba)^2+ba*(tbc-tba)*(tsim-tbc))*cos(theta);
end

if ((tsim>tm)  & (tc>=tsim)),
% phase 3
 vxn=ba*(tbc-tba)*cos(theta-wy*(tsim-tm));
 vzn=ba*(tbc-tba)*sin(theta-wy*(tsim-tm));
 hn1=(.5*ba*(tbc-tba)^2+ba*(tbc-tba)*(tm-tbc))*sin(theta);
 hn=hn1+ba*(tbc-tba)/wy*(cos(theta-wy*(tsim-tm))-cos(theta));
 xn1=(.5*ba*(tbc-tba)^2+ba*(tbc-tba)*(tm-tbc))*cos(theta);
 xn=xn1+ba*(tbc-tba)/wy*(-sin(theta-wy*(tsim-tm))+sin(theta));


elseif (tsim>tc),
% phase4
 vxn=ba*(tbc-tba);
 vzn=0;
 hn1=(.5*ba*(tbc-tba)^2+ba*(tbc-tba)*(tm-tbc))*sin(theta);
 hn=hn1+ba*(tbc-tba)/wy*(1-cos(theta));
 xn1=(.5*ba*(tbc-tba)^2+ba*(tbc-tba)*(tm-tbc))*cos(theta);
 xn2=xn1+ba*(tbc-tba)/wy*sin(theta);
 xn=xn2+ba*(tbc-tba)*(tsim-tc);
end


% GPS Measurement Generation
xgps=zeros(6,1);
xgps(1)=vxn+urv*rand(1);
xgps(2)=urv*rand(1);
xgps(3)=vzn+urv*rand(1);
xgps(4)=xn+urp*rand(1);
xgps(5)=urp*rand(1);
xgps(6)=hn+urp*rand(1);

% Measurement
% For full state estimation
z=xgps;
```

```
end

% State estimate update
 xi(1:21)=xi(1:21) + KK*(z- HH*xi(1:21));

% Covariance Matrix Update
PP = (eye(PP)-KK*HH)*PP*(eye(PP)-KK*HH)'+KK*RR*KK';

% Quaternion Normalization
xi(1:4)=xi(1:4)/sqrt(xi(1:4)'*xi(1:4));

% Propagation Phase

for k=1:(UT/deltaT)
if tsim>TSIM,
break
end


% Position and covariance matrix at  GPS loss
if(abs(tloss-tsim)<1e-4)
XOL=xi(12);
YOL=xi(13);
ZOL=xi(14);
PPss=PP;
end


% Accelerometer Input Generation

% Specific forces simulation
```

Here the specific forces and consequently, the accelerometer readings, will be generated as a function of time.

```
if((tsim>=tba) & (tsim<tbc)),
 thr=[ba*ones(4,1);zeros(8,1)];
 N1='Boost Phase';

elseif((tsim>=tbc) & (tsim<tm)),
 thr=zeros(12,1);
 N1='Constant Speed Take-off';
end

if(abs(tsim-tm)<1e-4),
 wydot=[0 0 d3*dwy -d4*dwy 0 0 0 0 -d9*dwy d10*dwy 0 0]';
 g1=gz*sin(theta)*ones(4,1);
 g3=(-wy*ba*(tbc-tba)/2+gz*cos(theta))*ones(4,1);
 grav=[g1;zeros(4,1);g3];
 N1='Beginning of the pitch maneuver';
 disp(N1)
% pause
```

```
end

if(((tsim-tm)>1e-4)&((tc-tsim)>1e-4)),
 wydot=zeros(12,1);
 g1=gz*sin(theta-wy*(tsim-tm));
 g3=-wy*ba*(tbc-tba)+gz*cos(theta-wy*(tsim-tm));
 grav=[g1*ones(4,1);zeros(4,1);g3*ones(4,1)];
 N1='Pitch Maneuver';

elseif(abs(tsim-tc)<1e-4),
 wydot=-[0 0 d3*dwy -d4*dwy 0 0 0 0 -d9*dwy d10*dwy 0 0]';
 grav=[zeros(8,1);(-wy*ba*(tbc-tba)/2+gz)*ones(4,1)];
 N1='End of Pitch Maneuver';
 disp(N1)
% pause

elseif(((tsim>tc)&((TOLoop-tsim)>1e-4))|((tsim-(TOLoop+tturn))>1e
-4))
 wydot=zeros(12,1);
 grav=[zeros(8,1);gz*ones(4,1)];
 wzdot=zeros(12,1);
 N1='Aided straight and level flight';

elseif(abs(tsim-TOLoop)<1e-4),
 N1='Beginning of the turn';
 disp(N1)
% pause

 wzv=wz*ba*(tbc-tba);
 wzdot1=[(-d1*dwz),d2*dwz,0,0];
 wzdot2=d5*dwz+wzv/2;
 wzdot3=-d6*dwz+wzv/2;
 wzdot4=[wzv/2*[1 1],zeros(1,4)];
 wzdot =[wzdot1,wzdot2,wzdot3,wzdot4]';

elseif(((tsim-TOLoop)>1e-4)&(((TOLoop+tturn)-tsim)>1e-4))
 wzdot=[zeros(1,4),wzv*ones(1,4),zeros(1,4)]';
 N1='Turn';

elseif(abs(tsim-(TOLoop+tturn))<1e-4),
 N1='End of the turn';
 disp(N1)
% pause

 wzdot1=[(d1*dwz),(-d2*dwz),0,0];
 wzdot2=-d5*dwz+wzv/2;
 wzdot3= d6*dwz+wzv/2;
 wzdot4=[wzv/2*[1 1],zeros(1,4)];
 wzdot =[wzdot1,wzdot2,wzdot3,wzdot4]';


elseif(tsim<tba),
```

```
 thr=zeros(12,1);

grav=[gz*sin(theta)*ones(4,1);zeros(4,1);gz*cos(theta)*ones(4,1)]
;
 wydot=zeros(12,1);
 wzdot=zeros(12,1);
 N1='Initialization';
end

% Accelerometer output generation
```

Here the bias and the discret white noise sequence is added to the nominal accelerometer outputs.

```
aa=sqrt(awni/deltaT)*rand(12,1) +bias +thr +grav +wydot+wzdot;

% Output from the accelerometers
```

Here the accelerometer output vector is processed by the the "C" matrix, which represents the linear combination that produces angular and linear acceleration information. "F" stands for specific forces f_x ,f_y , and f_z and "dW" to the angular acceleration vector, both coordinatized in the body frame.

```
output=C*aa;

F=output(1:3);
dW=output(4:6);

% Barometer Input Generation
% Altimeter Measurement and nominal trajectory simulation

if(tsim<tba)
 hn=0;
 xn=0;

elseif( (tsim>=tba) & (tbc>=tsim)),
% phase1
 hn=.5*ba*(tsim-tba)^2*sin(theta);
 xn=.5*ba*(tsim-tba)^2*cos(theta);

elseif ((tsim>tbc) & (tm>=tsim)),
% phase2
 hn=(.5*ba*(tbc-tba)^2+ba*(tbc-tba)*(tsim-tbc))*sin(theta);
 xn=(.5*ba*(tbc-tba)^2+ba*(tbc-tba)*(tsim-tbc))*cos(theta);

elseif ((tsim>tm)  & (tc>=tsim)),
% phase 3
 hn1=(.5*ba*(tbc-tba)^2+ba*(tbc-tba)*(tm-tbc))*sin(theta);
 hn=hn1+ba*(tbc-tba)/wy*(cos(theta-wy*(tsim-tm))-cos(theta));
 hn2=hn;
```

```
 xn1=(.5*ba*(tbc-tba)^2+ba*(tbc-tba)*(tm-tbc))*cos(theta);
 xn=xn1+ba*(tbc-tba)/wy*(-sin(theta-wy*(tsim-tm))+sin(theta));
 xn2=xn;

elseif((tsim>tc)&((TOLoop-tsim)>1e-4))
% phase4

 hn1=(.5*ba*(tbc-tba)^2+ba*(tbc-tba)*(tm-tbc))*sin(theta);
 hn=hn1+ba*(tbc-tba)/wy*(1-cos(theta));
 xn1=(.5*ba*(tbc-tba)^2+ba*(tbc-tba)*(tm-tbc))*cos(theta);
 xn2=xn1+ba*(tbc-tba)/wy*sin(theta);
 xn=xn2+ba*(tbc-tba)*(tsim-tc);
 xnt=xn;
elseif(((tsim-TOLoop)>1e-4)&(tsim<(TOLoop+tturn))),
 xn=xnt;

elseif(tsim-(TOLoop+tturn)>1e-4),
 xn=xnt+ba*(tbc-tba)*(tsim-tturn-tc);
end

hb=hn+sqrt(bwni/deltaT)*rand(1)+hbias;

% Simulation Phase

% Covariance Matrix Propagation
```

FF(i,j) and GG(i,j) now correspond to the time varying elements of the "F" and "G" matrices.

```
% The "FF" Matrix

FF(1,2)=-.5*xi(5);FF(1,3)=-.5*xi(6);FF(1,4)=-.5*xi(7);
FF(1,5)=-.5*xi(2);FF(1,6)=-.5*xi(3);FF(1,7)=-.5*xi(4);

FF(2,1)=-FF(1,2);FF(2,3)=-FF(1,4);FF(2,4)=FF(1,3);
FF(2,5)=.5*xi(1);FF(2,6)=FF(1,7);FF(2,7)=-FF(1,6);

FF(3,1)=-FF(1,3);FF(3,2)=FF(1,4);FF(3,4)=-FF(1,2);
FF(3,5)=-FF(1,7);FF(3,6)=FF(2,5);FF(3,7)=FF(1,5);

FF(4,1)=-FF(1,4);FF(4,2)=-FF(1,3);FF(4,3)=FF(1,2);
FF(4,5)=FF(1,6);FF(4,6)=-FF(1,5);FF(4,7)=FF(2,5);

a=xi(1);b=xi(2);c=xi(3);d=xi(4);

FF(8,1:4) =2*Fh'*[ a b -c -d;-d  c b -a; c  d  a b];
FF(9,1:4) =2*Fh'*[ d c  b  a; a -b c -d;-b -a  d c];
FF(10,1:4)=2*Fh'*[-c d -a  b; b  a d  c; a -b -c d];

FF(8:10,15:17)=-Cbn;
```

```
% Error Dynamic Input Matrix (Variable Coefficients)

GG(8:10,1:3)=Cbn;


 % Discrete Covariance Matrix Propagation
```

In order to speed up the computation, the covariance matrix is propagated linearly between sampling times by the discrete algorithm presented below. The propagation of the covariance matrix is done until the GPS loss condition is simulated, which will hapen after the covariance matrix reaches steady-state.

```
if((tloss-tsim)>1e-4),
ksi=eye(FF);
nn=8;
for kk=1:(nn-1)
 ksi=eye(FF)+FF*ksi*deltaT/(nn-kk+1);
end
gamma=deltaT*ksi*GG;
phi=eye(FF)+FF*deltaT*ksi;
PP=phi*PP*phi'+gamma*QQ*gamma'/deltaT;
end

% Simulation Sub-Routine
```

The Runge-Kutta-Fehlberg is used to propagated the state estimate dynamics.

```
t0=tsim;
tf=tsim+deltaT;
[t,xs]=ode23('ekfdyn11',t0,tf,xi);
st=max(size(t));
% Quaternion Normalization
xi=(xs(st,:))';xi(1:4)=xi(1:4)/sqrt(xi(1:4)'*xi(1:4));

% XT is the estimated position and attitude history
```

The attitude, position and bias residual information is displayed continuously during the simulation every four sampling times.

```
if(rem(jjj,4)<1e-6),
El = -asin(2*(xi(1)*xi(3) - xi(2)*xi(4)));
Elev=El*180/pi;
Bank =- asin(2*(xi(1)*xi(2) + xi(3)*xi(4))/cos(El))*180/pi;
Heading = -asin(2*(xi(1)*xi(4) + xi(2)*xi(3))/cos(El))*180/pi;
clc
EBHxyzt=[Elev Bank Heading xi(12) xi(13) xi(14) tsim]
N2=':      Accel., ang.accel. and alt. residual Biases';
disp([N1 N2])
res=[(C(1:3,:)*bias-xi(15:17))';(C(4:6,:)*bias-xi(18:20))';0 0
(hbias-xi(21))]
```

```
xn_z=[xn 0 hn z(4) z(5) z(6)]
```

The position, attitude and H-2 covariance matrix norm history is stored at the XT matrix every four sampling times. The matrix xt stores the nominal trajectory except during and after the turn maneuver.

```
XT(:,jj)=[xi(12:14);Elev;Bank;Heading;norm(PP)];
xt(:,jj)=[xn;0;hn];
jj=jj+1;
end

jjj=jjj+1;
tsim=tsim+deltaT;


% End of propagation
end

% End of simulation
if(tsim>TSIM),
sim=0
end


% End of Update
end

% Presentation
clc
N1='Time of Flight in Seconds';
disp(N1)
Time=tsim
N1='Final Elevation in Degrees';
disp(N1)
El = -asin(2*(xi(1)*xi(3) - xi(2)*xi(4)));
Elev=El*180/pi
pause
clc
N1='Final Bank angle in Degrees';
disp(N1)
Bank = asin(2*(xi(1)*xi(2) + xi(3)*xi(4))/cos(El))*180/pi
N1='Final Heading in Degrees';
disp(N1)
Heading = -asin(2*(xi(1)*xi(4) + xi(2)*xi(3))/cos(El))*180/pi
pause
clc
N1='Transformation Matrix C from Body to Navigation Frame';
disp(N1)
Cbn
pause
clc
```

```
N1='Accelerometer Generated Bias';
disp(N1)
bias
pause
clc

N1='Barometer bias and its compensation';
disp(N1)
hbias
hcomp=xi(21)
pause
clc
N1='Time of Initialization';
disp(N1)
tba
N1='Update Time during Initialization';
disp(N1)
UTI
N1='Sampling Time';
disp(N1)
deltaT
pause
clc
N1='Take-off Phase';
disp(N1)
N1='Acceleration Time';
disp(N1)
tbc-tba
pause
clc
N1='Begin of maneuver';
disp(N1)
tm
N1='End of maneuver';
disp(N1)
tc
pause
clc
N1='Moment of GPS Loss';
disp(N1)
TOLoop
pause
clc
N1='Position of GPS Loss (Turn begins)';
disp(N1)
XOL
YOL
ZOL
pause
clc
N1='Update Time during Flight';
disp(N1)
```

```
UTC
pause
clc
N1='Steady State Covariance Matrix ';
disp(N1)
PPss
pause
clc
N1='Steady State Standard Deviations';
disp(N1)
sdev=sqrt(diag(PPss)');
pause
clc
n1='Quaternion S.Dev';
disp(n1)
sdev(1:4)
n1='Angular Velocity S.Dev';
disp(n1)
sdev(5:7)
pause
clc
n1='Vel.S.Dev Nav. Frame';
disp(n1)
sdev(8:10)
n1='Vert.Chan.Aux.Var.S.Dev';
disp(n1)
sdev(11)
n1='Position S.Dev';
disp(n1)
sdev(12:14)
pause
clc
n1='Accel.Bias.S.Dev';
disp(n1)
sdev(15:17)
n1='Angular Accel. Bias S.Dev.';
disp(n1)
sdev(18:20)
n1='Baro-Alt.Bias S.Dev';
disp(n1)
sdev(21)
pause
clc

N1='Residual accelerometer bias effect';
disp(N1)
N1='Acceleration Residual Bias';
disp(N1)
C(1:3,:)*bias-xi(15:17)
N1='Ang.Accel.Residual Bias';
disp(N1)
C(4:6,:)*bias-xi(18:20)
```

```
N1='Residual altimeter bias';
disp(N1)
hbias-xi(21)
pause
clc
N1='Estimated angular velocities (Steady-State)';
disp(N1)
xi(5:7)'
pause
clc
N1='Final estimated altitude';
disp(N1)
xi(14)
N1='Final nominal altitude';
disp(N1)
hn
N1='Final Estimated Velocity';
disp(N1)
xi(8)
N1='Accelerometer White Noise Intensity [m2/s3]';
disp(N1)
awni
N1='Barometer White Noise Intensity [m2s]';
disp(N1)
bwni
N1='Quaternion Dynamics Pseudonoise Intensity';
disp(N1)
psnoise
clg
subplot(211)
plot(XT(1,:),XT(3,:),xt(1,:),xt(3,:),'-.',XOL,ZOL,'*')
% plot(XT(1,:),XT(3,:),xt(1,:),xt(3,:),'-.')
ylabel('Altitude Change (Meters)')
xlabel(' __simulated -.-. nominal * Open Loop')
% xlabel(' __simulated -.-. nominal ')
title([' Flight Path  Time of Simulation
=',num2str(tsim),'Sec.'])
grid
subplot(212)
plot(XT(1,:),XT(2,:),xt(1,:),xt(2,:),'-.',XOL,YOL,'*')
% plot(XT(1,:),XT(2,:),xt(1,:),xt(2,:),'-.')
ylabel('Lateral Motion Y (Meters)')
xlabel('Distance X N-Frame Axis  ')
title(['Trajectory Time of Simulation =',num2str(tsim),'Sec.'])
grid
pause

if(prt==1),
print
end

clg
```

```
tt=linspace(0,tsim,max(size(XT)));
plot(tt,XT(4,:),'-.',tt,XT(5,:),'--',tt,XT(6,:))
ylabel('Angle in Degrees')
title('Attitude Estimate')
xlabel(['-.Elev,--Bank,__Head, simul. of',num2str(tsim),'Sec.'])
grid
pause

if(prt==1)
print
end

plot(tt,XT(7,:))
title('Norm of the Covariance Matrix')
xlabel('Time (sec.)')
grid
if(prt==1),
print
end
pause

iniplot=1
while iniplot==1
N231='Covariance  Matrix during Initialization';
disp(N231)
tplot=input('Time in Seconds for Plot = ')
plot(tt(1:(tplot/deltaT)),XT(7,1:(tplot/deltaT)))
title('Norm of Covariance Matrix during Initialization')
xlabel('Time (sec.)')
grid
if(prt==1),
print
end
pause
iniplot=input('Enter 1 for new Init.Cov.Matrix plot ')
end
```

The propagation of the state estimates corresponds to the integration of the set of differential equations as described by the subroutine below:

```
% Routine EKFDYN11.M  I.M.U. Dynamics  and E.K.F Propagation
% This routine simulates the dynamics of an inertial measurement
% unit composed of twelve accelerometers
% as the propagation equations of the E.K.F. of the routine
% FAIMU5a.M.
% This function will generate the derivatives to be returned to
% a Runge-Kutta-Fehlberg Algorithm (ODE45.M)
```

```
% State Estimate  Propagation
% The state variable numbering will be as follows:
% ah(1) bh(2) ch(3) dh(4) wxh(5) wyh(6) wzh(7)
% vxh(8) vyh(9) vzh(10) hah(11) xh(12) yh(13) zh(14)
% Acceleration biases ab(15:17)
% Angular Acceleration Biases wb(18:20)
% Altimeter Bias hb(21)

% This is function ekfdyn10.m
function xd=ekfdyn11(t,x);

% Constants
% gz=gravity[m/s2] ; k1,k2 and k3 are PID gains for the
barometric
% aided vertical channel;
gz=-9.80665;
k1=6e-2;k2=1.2e-3;k3=8e-6; % Time constant=~50 sec.


% Normalization of the quaternion
q1=x(1:4);q1=q1./sqrt(q1'*q1);x(1:4)=q1;

% Derivatives of Quaternions
xd(1)= -.5*(x(2)*x(5) + x(3)*x(6) + x(4)*x(7));
xd(2)=  .5*(x(1)*x(5) + x(3)*x(7) - x(4)*x(6));
xd(3)=  .5*(x(1)*x(6) + x(2)*x(7) + x(4)*x(5));
xd(4)=  .5*(x(1)*x(7) + x(2)*x(6) - x(3)*x(5));


% Angular Velocities  .ate Estimate  Dynamics
xd(5:7)=dW-x(18:20);

% Calculation of Cbn Body to Navigation Frame Transformation
Matrix
Cbn(1,1)= x(1)^2 + x(2)^2 - x(3)^2 - x(4)^2;
Cbn(1,2)= 2*(x(2)*x(3) - x(1)*x(4));
Cbn(1,3)= 2*(x(1)*x(3) + x(2)*x(4));
Cbn(2,1)= 2*(x(1)*x(4) + x(2)*x(3));
Cbn(2,2)= x(1)^2 - x(2)^2 +x(3)^2 -x(4)^2;
Cbn(2,3)= 2*(x(3)*x(4) - x(1)*x(2));
Cbn(3,1)= 2*(x(4)*x(2) - x(1)*x(3));
Cbn(3,2)= 2*(x(1)*x(2) + x(3)*x(4));
Cbn(3,3)= x(1)^2 - x(2)^2 - x(3)^2 + x(4)^2;

% Specific forces estimate
Fh=F-x(15:17);

% Derivatives of Velocities Vx, Vy and Vz in Navigation
Coordinates
Vprime = Cbn*Fh;
xd(8)= Vprime(1);
xd(9)= Vprime (2);
```

```
xd(10)= Vprime(3)+ gz - k2*(x(14) - hb+x(21)) - x(11);

% Auxiliary variable for the vertical channel
xd(11)= k3*(x(14) - hb + x(21));

% Derivatives of Coordinates X, Y and Z in Navigation Frame
xd(12)=x(8);
xd(13)=x(9);
xd(14)=x(10) - k1*(x(14) - hb+x(21));

% Generalized Bias Modelling
xd(15:21)=zeros(1,7);
```

The six accelerometer inertial measurement unit was also simulated in order to show the unstability of this system when not receiving external aiding through an extended Kalman filter. The code is the routine FAIMU0.m as below:

Only the parts of the code which are specialized in the six accelerometer I.M.U. will be presented. The parts of the code which correspond exactly to the 12 accelerometer code will be omitted.

```
% 16.ThG Thesis
% Routine FAIMU0.m
% Simulation of a six accelerometer Inertial Measurement Unit
% This routine simulates a six accelerometer Inertial Measurement

% Unit driven by white noise with bias, stabilized by a
barometric-
% altimeter and an Extended Kalman Filter using GPS measurements.
% It uses full state measurement
% The input for each accelerometer will be a normally distributed
% white noise of (10 micro-g)² mean squared output error, plus a
% bias of 35 micro-g standard deviation.
format short e

% Global variables
global  Cbn  hb aa dd F dW C  Fh

% Lay-out of accelerometers
dd=.5;

% Accelerometer Output Processing Matrix
C=zeros(6);
C(1,5:6)=.5*[1 1];C(2,1:2)=.5*[1 1];C(3,3:4)=.5*[1 1];
```

```
C(4,3:4)=.5/dd*[1 -1];C(5,5:6)=.5/dd*[1 -1];C(6,1:2)=.5/dd*[1
-1];

% Gravity
gz=9.80665;

% Vertical Channel Gains
k1=6e-2;k2=1.2e-3;k3=8e-6;


% Description of State Variables:

% State Estimate Dynamics:
% ah(1) bh(2) ch(3) dh(4)      (Quaternion estimates)
% Wxh(5) Wyh(6) Wzh(7)         (Angular Velocities)
% Vxh(8) Vyh(9) Vzh(10)         (Velocities in the Navigation
Frame)
% hah(11) xh(12) yh(13) zh(14)(Auxiliar Variable and Position
Estimates
%                               in the Navigation Frame)
% (dUib)i=1:12(15:20)          ( accel. biases)
% hbias(27)                    (Baro-altimeter bias)
```

****
The part omitted is the same as in the twelve accelerometer case.

****

```
% Accelerometer Biases ;
abu=input('Accel.bias uncertainty [35e-6*gz]')
if isempty(abu)
abu=35e-6*gz
end
PP(15:20,15:20)=abu^2*eye(6);
% Baro-Altimeter error bias;
```

****
The same as in the 12 accel. case.
****

```
% Constant Extended Kalman Filter Matrices

% Error Dynamic Input Covariance Matrix QQ (Initialization)
QQI1=awni*eye(6);
QQI2=zeros(6,1);
QQI3=zeros(1,6);
QQI4=bwni;
QQI=[QQI1 QQI2;QQI3 QQI4];

% Addition of pseudo-noise to quaternions dynamics and to angular
% velocity dynamics (Flight)
QQF1=QQI;
```

```
QQF2=zeros(7,4);
QQF3=zeros(4,7);
QQF4=psnoise*eye(4);
QQF=[QQF1 QQF2;QQF3 QQF4];

******************************************************************
     The same as in the 12 accel. case.
******************************************************************

% Dimensioning of the FF Matrix
FF=zeros(21);
FF(5:7,15:20)=-C(4:6,:);
FF(10,11)=-1;FF(10,14)=-k2;FF(10,21)=-k2;
FF(11,14)=k3;FF(11,21)=k3;
FF(12,8)=1;
FF(13,9)=1;
FF(14,10)=1;FF(14,14)=-k1;;FF(14,21)=-k1;

% Error Dynamic Input Matrix GG   (Constant Coefficients only)
% Initialization phase
GGI=zeros(21,7);
GGI(5:7,1:6)=C(4:6,:);
GGI(10,7)=k2;
GGI(11,7)=-k3;
GGI(14,7)=k1;



% Flight phase
GGF=zeros(21,11);
GGF(5:7,1:6)=C(4:6,:);
GGF(10,7)=k2;
GGF(11,7)=-k3;
GGF(14,7)=k1;
% Addition of pseudo-noise to quaternion and angular
% velocity dynamics
GGF(1,8)=1;GGF(2,9)=1;GGF(3,10)=1;GGF(4,11)=1;
******************************************************************
     The same as in the 12 accel. case.
******************************************************************
% Accelerometer Input Generation

% Specific forces simulation

if ((tsim>=tba) & (tsim<tbc))
 thr=[zeros(4,1);ba*ones(2,1)];
 N1='Boost Phase';

elseif ((tsim>=tbc) & (tsim<tm))
 thr=zeros(6,1);
 N1='Constant Speed Take-off';
end
```

```
if abs(tsim-tm)<1e-4
 wydot=dd*[zeros(4,1);dwy*[1;-1]];

grav=[zeros(2,1);(-wy*ba*(tbc-tba)/2+gz*cos(theta))*ones(2,1);gz*
sin(theta)*ones(2,1)];
 N1='Beginning of the pitch maneuver';
 disp(N1)
 pause
end


if(((tsim-tm)>1e-4)&((tc-tsim)>1e-4)),
 wydot=zeros(6,1);
 g1=gz*sin(theta-wy*(tsim-tm));
 g3=-wy*ba*(tbc-tba)+gz*cos(theta-wy*(tsim-tm));
 grav=[zeros(2,1);g3*ones(2,1);g1*ones(2,1)];
 N1='Pitch Maneuver';

elseif(abs(tsim-tc)<1e-4)
 wydot=-dd*[zeros(4,1);dwy*[1;-1]];
 grav=[zeros(2,1);(-wy*ba*(tbc-tba)/2+gz)*ones(2,1);zeros(2,1)];
 N1='End of Pitch Maneuver';
 disp(N1)
 pause


elseif(((tsim>tc)&((TOLoop-tsim)>1e-4))|((tsim-(TOLoop+tturn))>1e
-4))
 wydot=zeros(6,1);
 grav=[zeros(2,1);gz*ones(2,1);zeros(2,1)];
N1='Aided straight and level flight';

elseif(abs(tsim-TOLoop)<1e-4),
 N1='Beginning of the turn';
 disp(N1)
 pause
wzv=wz*ba*(tbc-tba);
wzdot=[dd*dwz*[1 -1] wzv/2*[1 1] 0 0]';

elseif(((tsim-TOLoop)>1e-4)&(((TOLoop+tturn)-tsim)>1e-4))
N1='Turn';
wzdot=[0 0 wzv*[1 1] 0 0]';

elseif(abs(tsim-(TOLoop+tturn))<1e-4),
 N1='End of the turn';
 disp(N1)
 pause
wzdot=[dd*dwz*[-1 1] wzv/2*[1 1] 0 0]';

elseif(tsim<tba),
N1='Initialization';
 thr=zeros(6,1);
```

```
grav=[zeros(2,1);gz*cos(theta)*ones(2,1);gz*sin(theta)*ones(2,1)]
;
 wydot=zeros(6,1);
 wzdot=zeros(6,1);
end


% White noise generation plus bias and gravity
aa=sqrt(awni/deltaT)*rand(6,1) +bias +thr +grav +wydot+ wzdot;

% Output from the accelerometers
output=C*aa;

F=output(1:3);
dW=output(4:6);

% Barometer Input Generation
*****************************************************************
     The same as in the 12 accel. case.
*****************************************************************
% Covariance Matrix Propagation

% The "FF" Matrix

FF(1,2)=-.5*xi(5);FF(1,3)=-.5*xi(6);FF(1,4)=-.5*xi(7);
FF(1,5)=-.5*xi(2);FF(1,6)=-.5*xi(3);FF(1,7)=-.5*xi(4);

FF(2,1)=-FF(1,2);FF(2,3)=-FF(1,4);FF(2,4)=FF(1,3);
FF(2,5)=.5*xi(1);FF(2,6)=FF(1,7);FF(2,7)=-FF(1,6);

FF(3,1)=-FF(1,3);FF(3,2)=FF(2,3);FF(3,4)=-FF(1,2);
FF(3,5)=-FF(1,7);FF(3,6)=FF(2,5);FF(3,7)=-FF(1,5);
FF(4,1)=FF(1,4);FF(4,2)=-FF(1,3);

FF(4,1)=-FF(1,4);FF(4,2)=-FF(2,4);FF(4,3)=FF(1,2);
FF(4,5)=FF(1,6);FF(4,6)=-FF(1,5);FF(4,7)=FF(2,5);

FF(5,6)=-xi(7);FF(5,7)=-xi(6);
FF(6,5)=-xi(7);FF(6,7)=-xi(5);
FF(7,5)=-xi(6);FF(7,6)=-xi(5);
a=xi(1);b=xi(2);c=xi(3);d=xi(4);

FF(8,1:4)=2*Fh'*[a b -c -d; -d c b -a; c d a b];
FF(9,1:4)=2*Fh'*[d c b a;a -b c -d;-b -a d c];
FF(10,1:4)=2*Fh'*[-c d -a b;b a d c;a -b -c d];


FF(8:10,15:20)=-Cbn*C(1:3,:);

% Error Dynamic Input Matrix (Variable Coefficients)
```

```
GG(8:10,1:6)=-FF(8:10,15:20);

% Discrete Covariance Matrix Propagation
******************************************************************
     The same as in the 12 accel. case.
******************************************************************
clc
EBHxyzt=[Elev Bank Heading xi(12) xi(13) xi(14) tsim]
N2=':      Accel., ang.accel. and alt. residual Biases';
disp([N1 N2])
res=[(C(1:3,:)*(bias-xi(15:20)))';(C(4:6,:)*(bias-xi(15:20)))';0
0 (hbias-xi(21))]
xn_z=[xn 0 hn z(4) z(5) z(6)]
XT(:,jj)=[xi(12:14);Elev;Bank;Heading;norm(PP)];
xt(:,jj)=[xn;0;hn];
jj=jj+1;
end

jjj=jjj+1;
tsim=tsim+deltaT;

% End of propagation
end
******************************************************************
     The same as in the 12 accel. case.
******************************************************************
end
```

The routine which gives the set of differential equations to the integration sub-routine represents the propagation equations for six accelerometer inertial measurement unit as follows:

```
% Routine EKFDYN.M  I.M.U. Dynamics  and E.K.F Propagation
% This routine simulates the dynamics of an inertial measurement
% unit composed of six accelerometers driven by white noise
% and the propagation equations of the corresponding E.K.F.
% coded in the routine FAIMU0.M
% This function will generate the derivatives to be returned to
% a Runge-Kutta-Fehlberg Algorithm (ODE45.M)

% State Estimate  Propagation
% The state variable numbering will be as follows:
% ah(1) bh(2) ch(3) dh(4) wxh(5) wyh(6) wzh(7)
% vxh(8) vyh(9) vzh(10) hah(11) xh(12) yh(13) zh(14)
% Accelerometer biases ab(15:20)
% Altimeter Bias hb(21)

% This is function ekfdyn.m
function xd=ekfdyn(t,x);
```

```
% Constants
% gz=gravity[m/s2] ; k1,k2 and k3 are PID gains for the
barometric
% aided vertical channel;
gz=-9.80665;
k1=6e-2;k2=1.2e-3;k3=8e-6; % Time constant=~50 sec.


% Normalization of the quaternion
q1=x(1:4);q1=q1./sqrt(q1'*q1);x(1:4)=q1;

% Derivatives of Quaternions
xd(1)= -.5*(x(2)*x(5) + x(3)*x(6) + x(4)*x(7));
xd(2)=  .5*(x(1)*x(5) + x(3)*x(7) - x(4)*x(6));
xd(3)=  .5*(x(1)*x(6) + x(2)*x(7) + x(4)*x(5));
xd(4)=  .5*(x(1)*x(7) + x(2)*x(6) - x(3)*x(5));



% Angular Velocities State Estimate  Dynamics
xd(5:7)=dW-C(4:6,:)*x(15:20)-[x(6)*x(7);x(5)*x(7);x(5)*x(6)];

% Calculation of Cbn Body to Navigation Frame Transformation
Matrix
Cbn(1,1)= x(1)^2 + x(2)^2 - x(3)^2 - x(4)^2;
Cbn(1,2)= 2*(x(2)*x(3) - x(1)*x(4));
Cbn(1,3)= 2*(x(1)*x(3) + x(2)*x(4));
Cbn(2,1)= 2*(x(1)*x(4) + x(2)*x(3));
Cbn(2,2)= x(1)^2 - x(2)^2 +x(3)^2 -x(4)^2;
Cbn(2,3)= 2*(x(3)*x(4) - x(1)*x(2));
Cbn(3,1)= 2*(x(4)*x(2) - x(1)*x(3));
Cbn(3,2)= 2*(x(1)*x(2) + x(3)*x(4));
Cbn(3,3)= x(1)^2 - x(2)^2 - x(3)^2 + x(4)^2;

% Specific forces estimate (in Body Coordinates)
Fh=F-C(1:3,:)*x(15:20);
% Derivatives of Velocities Vx, Vy and Vz in Navigation
Coordinates
Vprime = Cbn*Fh;
xd(8)= Vprime(1);
xd(9)= Vprime (2);
xd(10)= Vprime(3)+ gz - k2*(x(14) - hb+x(21)) - x(11);

% Auxiliary variable for the vertical channel
xd(11)= k3*(x(14) - hb + x(21));

% Derivatives of Coordinates X, Y and Z in Navigation Frame
xd(12)=x(8);
xd(13)=x(9);
xd(14)=x(10) - k1*(x(14) - hb+x(21));
```

```
% Generalized Bias Modelling
xd(15:21)=zeros(1,7);
```

Chapter 6

SIMULATION RESULTS

6.1 Simulation of the Twelve Accelerometer I.M.U.

The codes presented in section 5.5 display the trajectory of the vehicle as calculated by the I.M.U. for 600 seconds of simulation. A Monte-Carlo simulation would be very useful, but the computational resources available for this work were not adequate for this job because MATLAB™ is an interpreted language not fast enough for programming loops, and the PC 386 machine is also not fast enough for this application.

The above mentioned facts lead to the presentation of one typical simulation as follows.

```
Time of Simulation in Seconds
Time =  600.0250

Final Elevation in Degrees
Elev =    0.4586

Final Bank angle in Degrees
Bank =   -0.0125

Final Heading in Degrees
Heading =    0.3647

Final Transformation Matrix C from Body to Navigation Frame
Cbn =
    0.9999    0.0064   -0.0080
   -0.0064    1.0000    0.0003
    0.0080   -0.0002    1.0000

Accelerometer Generated Bias [m/s^2 ]
bias =   1.0e-03 *[0.3999 0.2152 0.0258 0.1207 -0.2391  0.5822
                   0.0203 0.6168 0.0906 0.2992 -0.4964 -0.2407]

Barometer bias and its compensation (meters)
hbias =    0.0249
hcomp =    0.0249

Time of Initialization (sec.)
```

```
tba =   120

Update Time during Initialization (sec.)
UTI =    0.1000

Update Time during Flight (sec.)
UTC =     5

Sampling Time (sec.)
deltaT =    0.0250

Take-off Phase
Acceleration Time (Boost phase)
ans =     5 sec.

Begin of pitch maneuver
tm =   141 (sec.)

End of pitch maneuver
tc =   149 (sec.)

Moment of GPS Loss
tloss =   399 (sec.)

Beginning of the 360 deg. turn
TOLoop =  409 (sec.)

Duration of the turn
tturn =   60 (sec.)

Position at which the turn maneuver begins (meters)
XOL =   6.6466e+04
YOL =   -0.0312
ZOL =   4.9600e+03

Steady State Covariance Matrix ( before GPS loss)

Steady State Standard Deviations
Quaternion S.Dev
ans =    0.0002    0.0003    0.0003    0.0036

Angular Velocity S.Dev (rad/sec)
ans =   1.0e-04 *[0.2137    0.2375    0.4921]

Vel.S.Dev in the Nav. Frame (m/s)
ans =    0.0834    0.0817    0.0592

Vert.Chan.Aux.Var.S.Dev (m/s²)
ans =   4.4666e-04

Position S.Dev (meters)
ans =    2.6798    2.6819    0.6115
```

```
Accel.Bias.S.Dev (m/s²)
ans =    1.0e-06 *[0.8889    0.2276    0.5495]

Angular Accel. Bias S.Dev.(rad/sec²)
ans =    1.0e-06 *[0.1532    0.1764    0.1485]

Baro-Alt.Bias S.Dev (meters)
ans =    4.1956e-07

Residual accelerometer bias effect
Acceleration Residual Bias (m/s² )
ans =    1.0e-06 *[-0.0059   -0.2338    0.2479]

Ang.Accel.Residual Bias (rad/sec²)
ans =    1.0e-07 *[0.7659   -0.1277    0.2774]

Residual altimeter bias (meters )
ans =    3.9537e-07

Estimated angular velocities (Steady-State) (rad/sec)
ans =    1.0e-04 *[0.1512    0.0262   -0.1487]

Final estimated altitude
ans =    4.9593e+03  (meters)

Final nominal altitude
hn =    4.9603e+03   (meters)

Accelerometer White Noise Intensity [m2/s3]
awni =    1.9234e-11

Barometer White Noise Intensity [m².sec]
bwni =    2.0000e-11

Quaternion Dynamics Pseudonoise Intensity
psnoise =    1.0000e-09
```

The code is able to plot the estimated and the nominal trajectories as can be seen in Figure 19 . The circular path due to the 360 deg. turn appears to be an ellipse due to the different scales used for the X and Y axes. The asterisk indicates the point where the GPS loss occurs meaning that from that point on the system receives no external aiding.

The attitude history can be seen in Figure 20 . The pitch maneuver can be seen at the left of the plot, where the pitch angle goes linearly from 60 degrees to near zero degrees. The turn maneuver can be seen at the right part of the plot. The yaw angle goes linearly from zero to minus 90 deg., returning to plus 90 deg. and then to zero deg., indicating therefore a complete turn of 360 deg. This is due to the fact that the yaw angle Ψ comes from a sine function as below,

$$\psi=\sin^{-1}\left[\frac{2(ad+bc)}{\cos\theta}\right] \tag{152}$$

The roll angle remains close to zero during all the simulation time.

The covariance matrix norm plot is shown in Figure 21. This plot shows how the covariance matrix reaches steady-state during the flight phase and the straight line at the right side of the plot shows the norm magnitude corresponding to the last update before the GPS loss.

The covariance norm during initialization is shown in Figure 22. The high quality external I.M.U. information reduces the uncertainty considerably after the first update. These parameter values were chosen in order to reduce the simulation time necessary to obtain a good bias estimate.

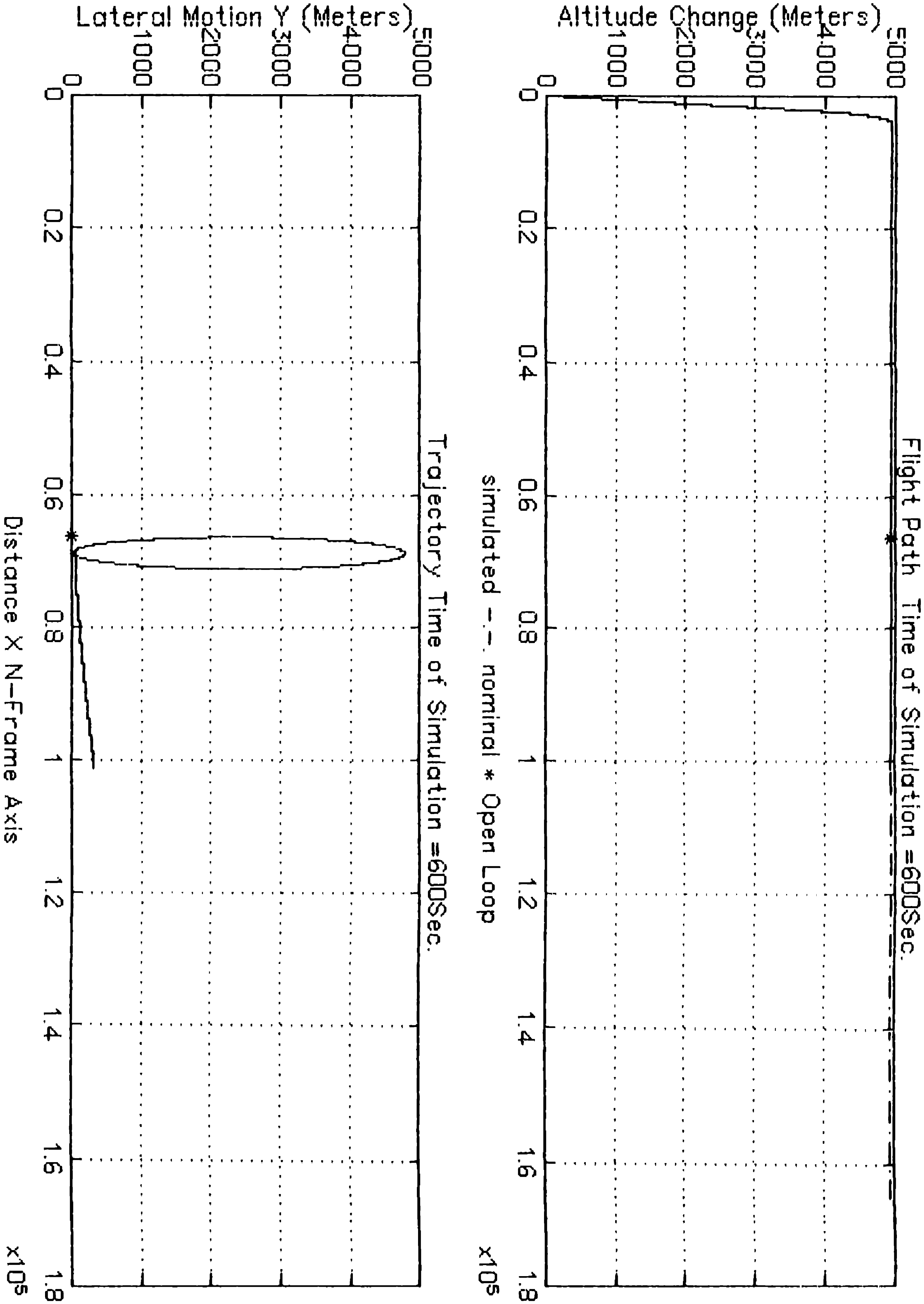

Figure 19: Estimated and Nominal Trajectories (12 Accelerometer I.M.U.)

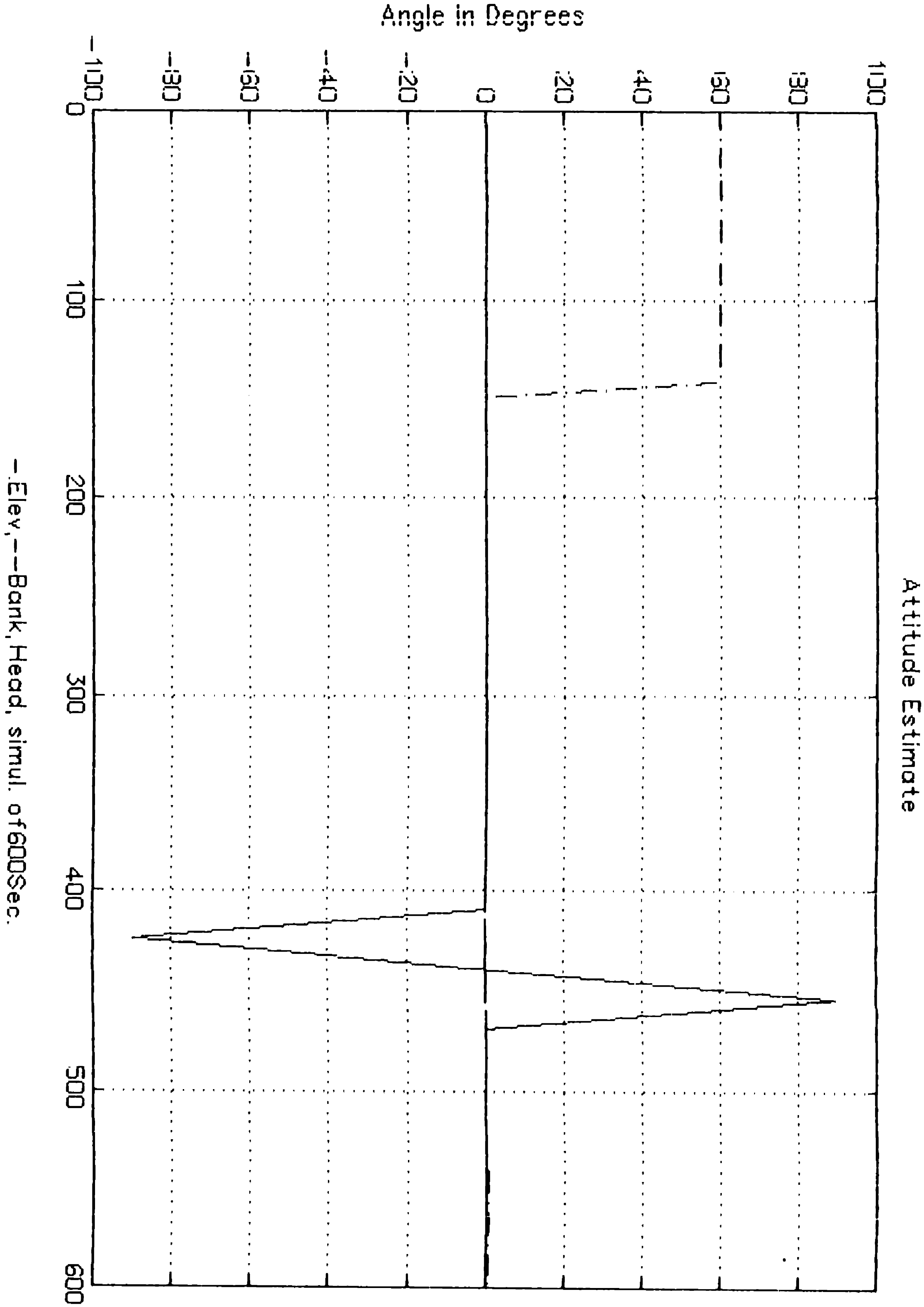

Figure 20: Estimated Attitude History
(12 Accelerometer I.M.U.)

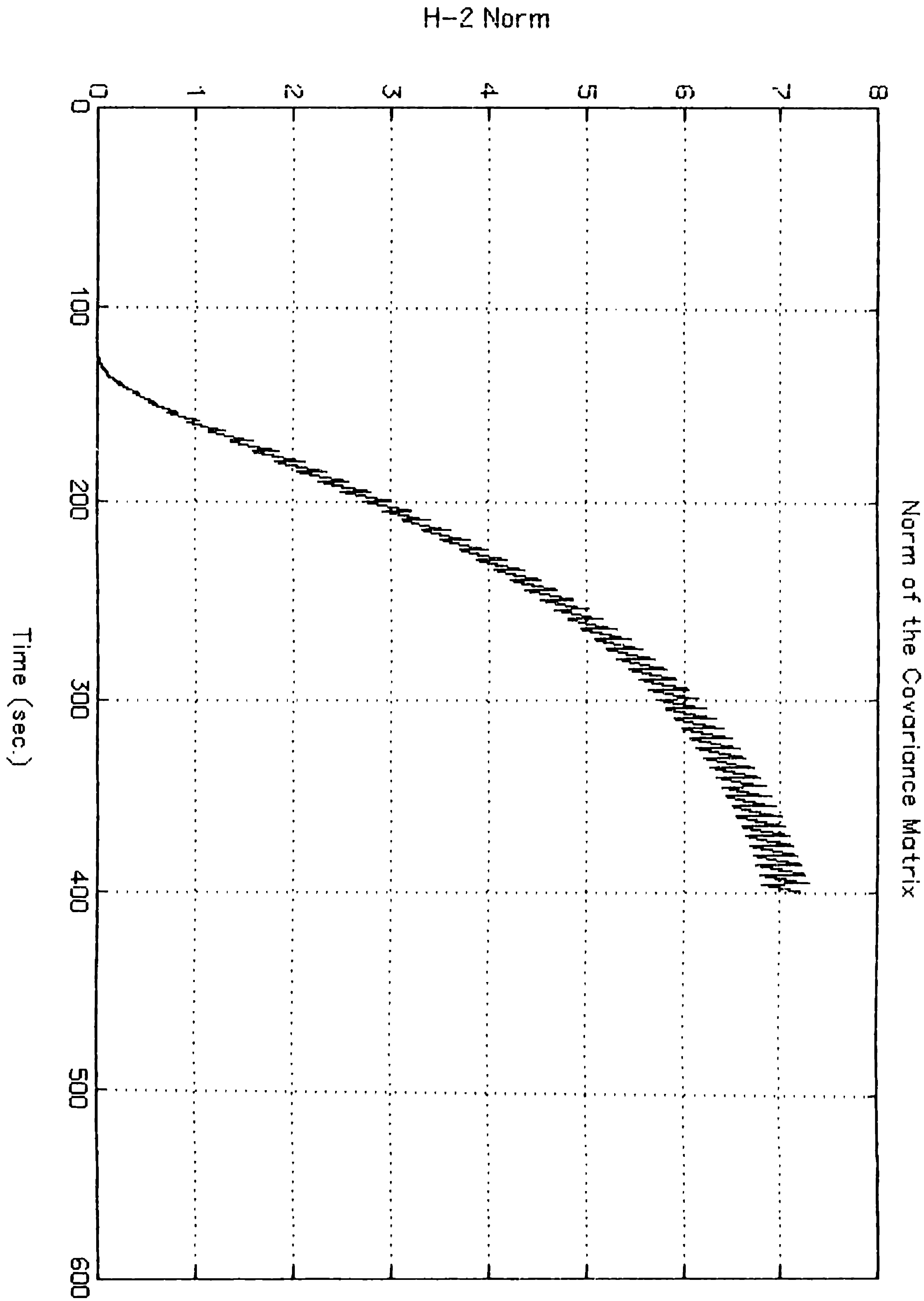

Figure 21: Covariance Matrix Norm (Flight Phase)
(12 Accelerometer I.M.U.)

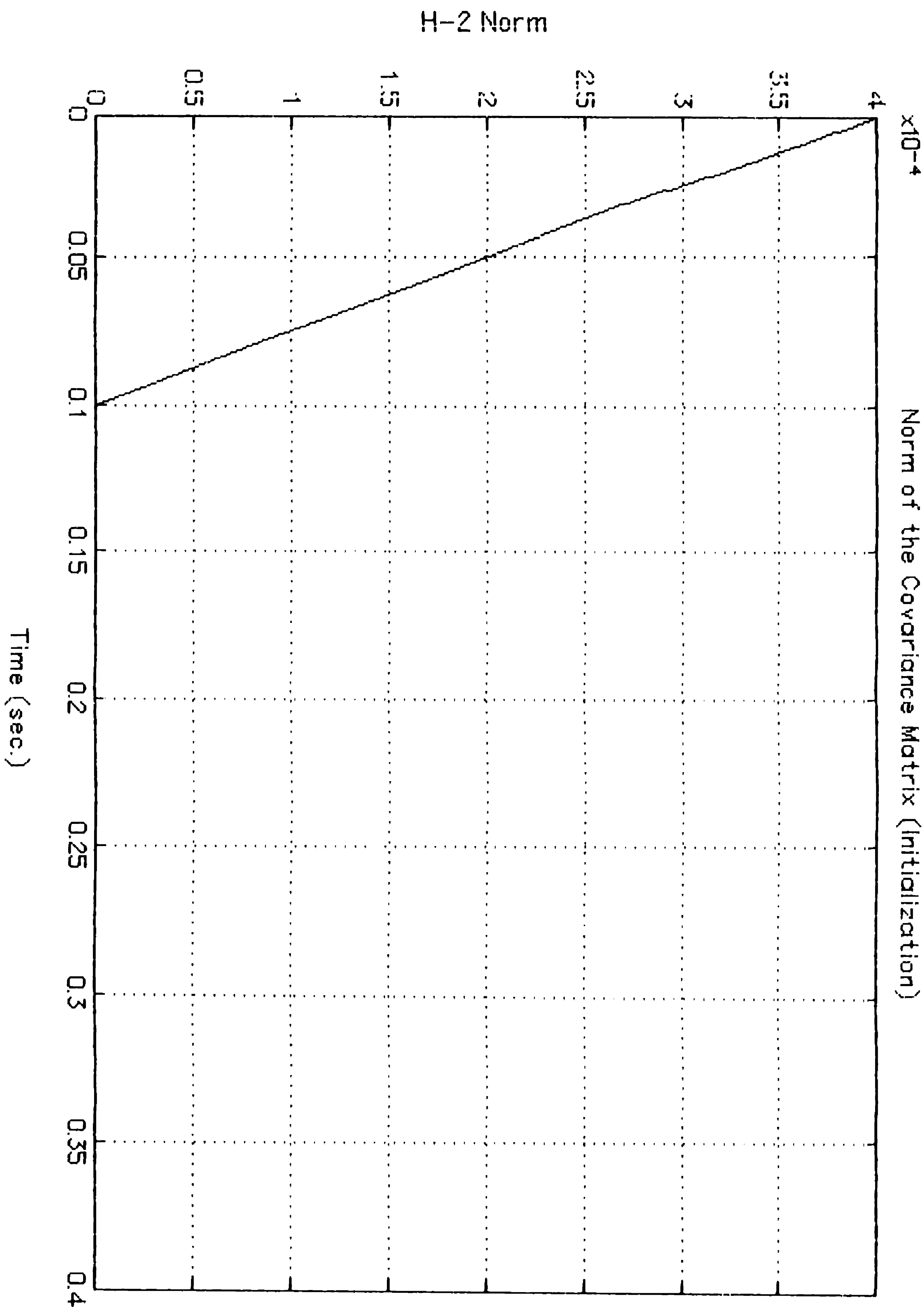

**Figure 22: Covariance Matrix Norm (Initialization Phase)
(12 Accelerometer I.M.U.)**

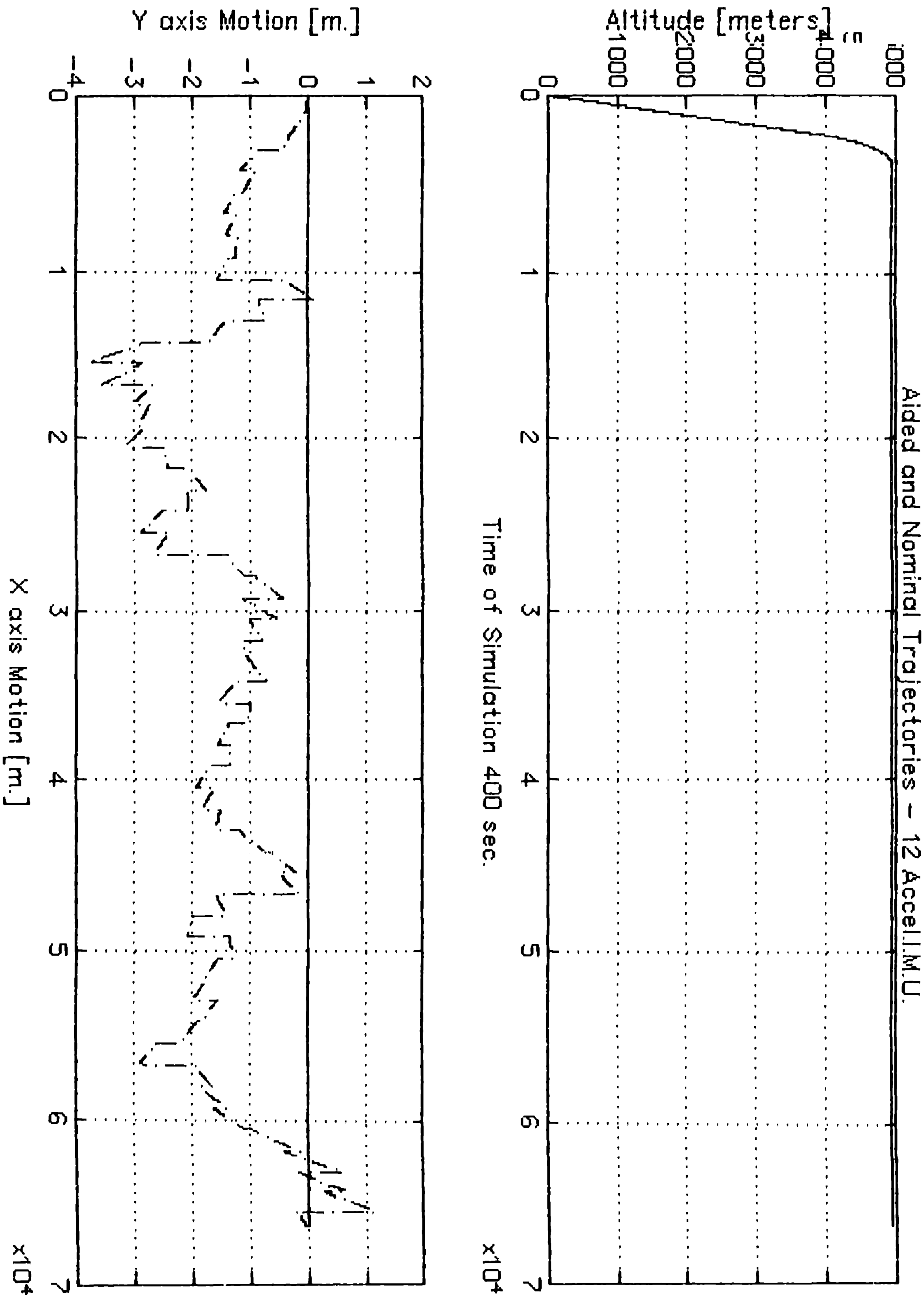

Figure 23: Aided Trajectory (12 Accelerometer I.M.U.)

The asterisk in the trajectory of Figure 19 marks the point where the system stops receiving external aiding from the GPS satellite system. The navigation system was then able to track a 360 degree turn at a speed of 250 meters per second, corresponding to a circle with a diameter of 4774.65 meters executed in 60 seconds. After the turn the computed attitude was:

Final Elevation in Degrees = 0.4586
Final Bank angle in Degrees = -0.0125
Final Heading in Degrees = 0.3647 .

These deviations in attitude were also responsible for the trajectory deviation after the turn. In general the bias residuals and the low sampling rate are the most important contributing factors to these deviations.

The performance during the aided phase can be better seen in figure 23. In this Figure the estimated trajectory until the GPS loss is presented, so that it is possible to observe the effect of the updates correcting the position.

6.2 Simulation of the Six Accelerometer I.M.U.

The unstable system with six accelerometers was also simulated in order to show how it behaves when the GPS loss is simulated. The maneuver introduced a positive angular acceleration and consequently a positive angular velocity along the body frame Z axis in the beginning of the turn. If the angular velocity components along the X and Y axes were kept zero there would be no stability problem, but a small residue along those two axes may cause instability as can be seen in the Figure 24. The effect of attitude instability is shown in the trajectory computation, which diverges from the circular path as can be seen in Figure 25.

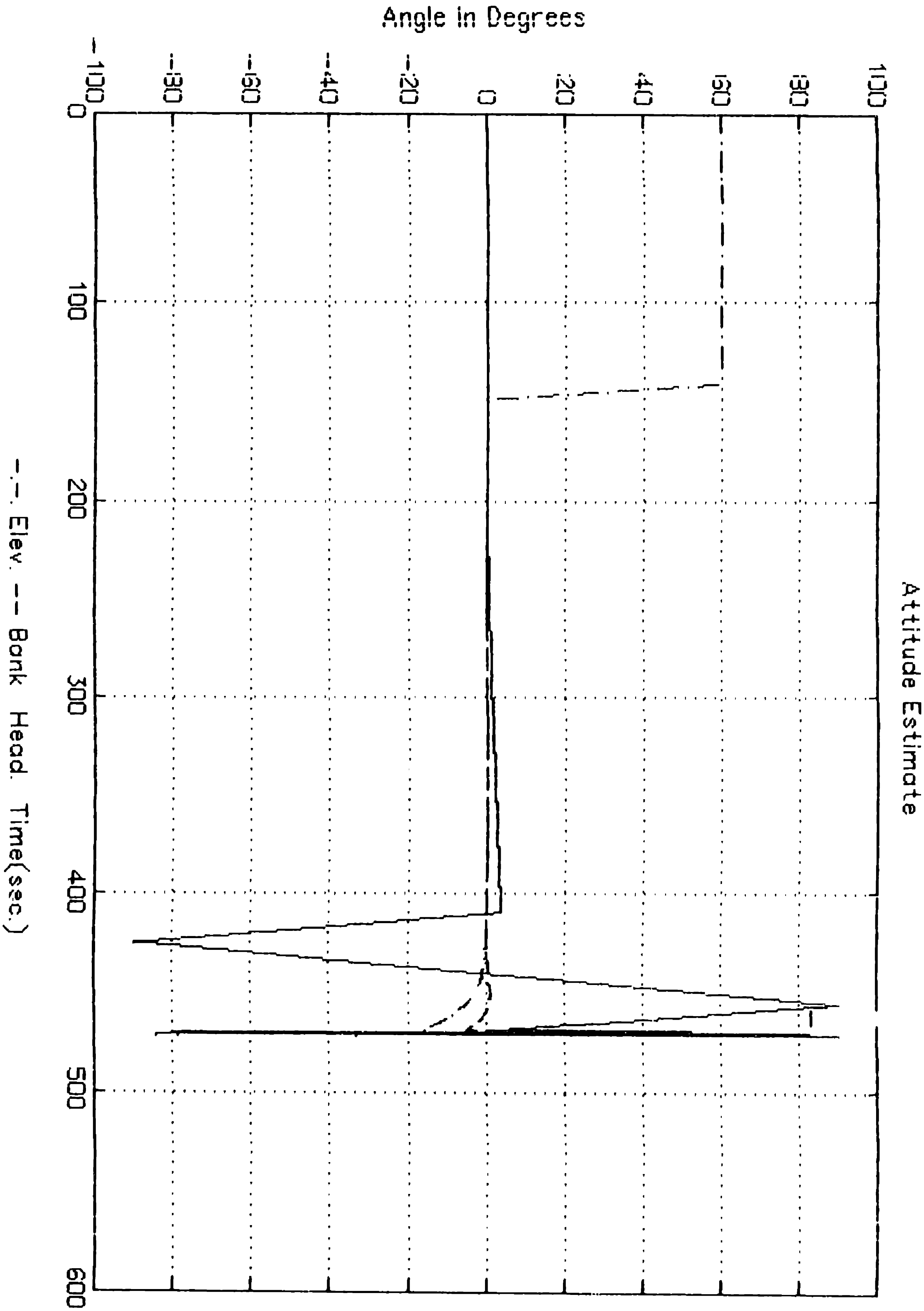

Figure 24: Unstable Attitude Dynamics (6 Accelerometer I.M.U.)

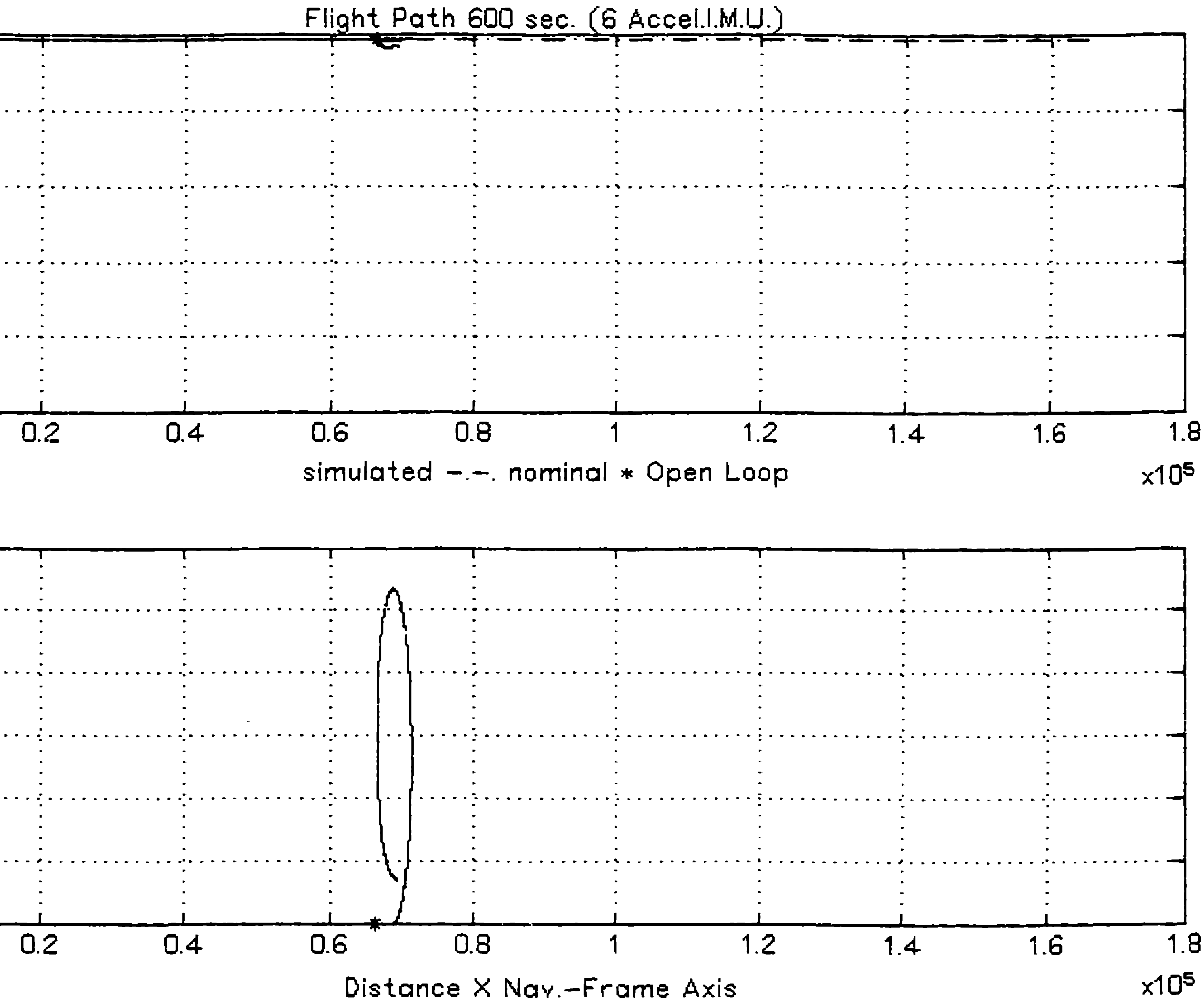

Figure 25: Unstable Trajectory (6 Accelerometer I.M.U.)

Chapter 7

CONCLUSIONS AND FUTURE RESEARCH

7.1 Conclusions

It was possible to demonstrate that:

a) With twelve accelerometers or more, provided that they are properly positioned in space, it is possible to obtain linear and angular acceleration as a linear combination of the accelerometer outputs.

b) By using a high quality external inertial measurement unit during the initialization it is possible to obtain very good estimates of the accelerometer biases. Other sources of accelerometer and barometric-altimeter errors may also be included in the model, thereby augmenting even more the number of states. The initialization can be performed in a time frame of less than five minutes depending on the nature of the vehicle mission. The initialization procedure allows the vehicle to navigate successfully during the boost phase, in which it receives no other external aid but the barometric-altimeter.

c) During the flight phase, the external aiding provided by the GPS navigation system through an extended Kalman filter improves considerably the performance of the system with updates every 5 seconds. The twelve accelerometer I.M.U. keeps track of the high frequency dynamics of the vehicle, while the baro-altimeter and the GPS system track the low frequency dynamics in terms of speed and position.

d) After the extended Kalman filter reaches steady-state, meaning that the covariance matrix norm will not vary within one percent from the steady-state solution, a GPS information loss was simulated and the vehicle was subjected to a 360 degrees turn in 60 seconds at a tangential speed of 250 m/sec. In the 12 accelerometer I.M.U. the system was able to track this maneuver in a stable way, while this tracking is impossible to be performed by a 6 accelerometer I.M.U. due to its inherent unstable behavior.

e) This inertial navigation system is a low cost system which may be employed in ground vehicles and short range missiles.

7.2 Future Research

Further work should be accomplished in developing a better integration algorithm and incorporating the navigation in geographic coordinates. A better integration algorithm should be specialized to the standard form of accelerometer outputs which are increments of integrated specific forces, ΔV, and thus be able to perform the integration in less time within the required precision. The navigation in geographic coordinates would allow a simulation closer to the real application.

In order to perform Monte-Carlo simulations, the code should be written in "C", Fortran, or other compiled language.

According to Professor W.E. Vander Velde, who first had the insight of this system, a stable full accelerometer inertial measurement unit can also be implemented with nine accelerometers , and this work shows that this number can be even greater than twelve. Considering that nowadays there are many low cost accelerometer options, different lay-outs, ranging from 9 to, say, 48 accelerometers could also be investigated in terms of estimation, and fault-tolerance characteristics.

The implementation of an extended Kalman filter in terms of error estimates rather than with full state estimation could also be investigated. This would allow the system to incorporate a guidance law at a higher sampling rate.

The laboratory setup, in the event of a prototype construction, should have a dual-axis test table in order to simulate the system in the one-g field, and the accelerometer output should be transmitted directly to an on-line computing workstation in order to allow a better software and hardware testing.

Bibliography

[1] Walter Wrigley, Walter M. Hollister, and William G. Denhard, Gyroscope Theory, Design, and Instrumentation, The M.I.T. Press. October, 1968.

[2] O.Glaser and F.Krumm, A Kalman Filter Concept for an Augmented Low-Cost Inertial Navigation System based on Fiber Optic Gyros, Symposium Gyro Technology 1989, Stuttgart, Germany.

[3] Wallace Earl Vander Velde, Strapdown Inertial Navigation Systems, Lecture Notes - Subject 16.43, MIT, Department of Aeronautics and Astronautics, 1992.

[4] William S. Widnall and Peter A. Grundy, Inertial Navigation System Error Models, TR-03-73. Intermetrics Incorporated, Cambridge, Massachusetts. May, 1973.

[5] Peter S. Maybeck, Stochastic Model, Estimation, and Control, Volumes 1 and 2, Academic Press, Inc., New York, 1979.

[6] Gene F. Franklin, J.David Powell and Michael L.Workman, Digital Control of Dynamic Systems, 2^{nd} edition, Adisson-Wesley Publishing Company, U.S.A, June 1990.