Development of a GPS-Aided Navigation System without INS for the USAF A-10 Aircraft

by

Captain Jeffrey W. Eggers, USAF

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

February 1995

Signature of Author _____
Department of Aeronautics and Astronautics
December 12, 1994

Certified By _____
Professor John J. Deyst, Jr.
Thesis Supervisor, Massachusetts Institute of Technology

Approved By _____
Edward V. Bergmann and James I. Donna
Thesis Supervisors, Charles Stark Draper Laboratory

Accepted By _____
Professor Harold Y. Wachman
Chairman, Department Graduate Committee

I hereby assign my copyright of this thesis to the Charles Stark Draper Laboratory, Inc., Cambridge, Massachusetts.

Jeffrey W. Eggers

The Charles Stark Draper Laboratory hereby grants to MIT permission to reproduce and to distribute publicly paper and electronic copies of this thesis document in whole or in part.

Development of a GPS-Aided Navigation System without INS for the USAF A-10 Aircraft

by

Captain Jeffrey W. Eggers, USAF

February, 1990

Abstract

Global Positioning System (GPS) receivers are normally integrated into fighter aircraft as a complement to the already existing Inertial Navigation System (INS). The INS provides wide-bandwidth, high-rate position and velocity estimates which are prone to drift rate errors, while the GPS provides low-rate, but very accurate and drift-free estimates. The GPS data are used to estimate errors in the INS data and generate corrections. But, it is the high-rate INS data that is used by the aircraft for flight control and weapons delivery calculation. If the INS fails, the GPS data is generally of too low a rate to be used for these purposes.

Fighter aircraft usually have another source of high-rate, wide-bandwidth data from attitude gyroscopes, heading sensors, angle of attack vanes, and the pitot static system. These systems contain several sources of error and fail to account for winds. The purpose of this thesis is to develop a method for using these secondary data sources, in combination with the GPS, to synthesize a high-rate, wide-bandwidth, accurate navigation solution. This thesis develops a two-stage navigation filter which uses GPS to estimate errors in the secondary data sources and then transforms the data into smooth, Earth-surface frame position and velocity values.

Thesis Supervisor: John J. Deyst, Jr.
Title: Jerome C. Hunsaker Professor of Aeronautics and Astronautics

Acknowledgments

My thanks to Ed Bergmann for supervising my thesis and for telling me that measurements are usually held in a column vector. Thanks to Jim Donna for all the Physics lessons and reminding me what a cross product is. Thanks to Captain Tim Oram for supplying recorded wind data. Thanks to all the pilots at the 57 Test Group for supplying some aircraft performance data. Thanks to Professor Deyst for rushing through the final review so I could move back to Nevada. Finally, a special thanks to my wife Diane for editing my grammar and keeping me nearly sane through the whole process.

Biography

The Author graduated from the United States Air Force Academy in 1984 with majors in Electrical Engineering and Mathematics. He then attended pilot training and spent seven years flying the A-10 aircraft in the U.S. Air Force. The last four years were spent at the 57th Test Group performing operational test and evaluations. The author completed requirements for a Master of Science in Aeronautics and Astronautics in December 1994 concentrating in Guidance, Navigation and Control.

Table of Contents

1. Introduction

1.1 Overview of the Problem

Engineers at the C. S. Draper Laboratory are integrating Global Positioning System (GPS) navigation equipment into the current A-10 aircraft navigation system. The integration involves estimating the aircraft navigation state from data supplied by the GPS, Inertial Navigation Set (INS), and several other aircraft sensors. One of the requirements of the design is to provide backup navigation modes for use when any one or more sensors fail. If the GPS fails, the aircraft still has a high-rate fairly accurate navigation sensor in the INS. However, if the INS fails, the GPS signal alone is unable to provide the high-rate velocity data required by the other aircraft systems such as autopilot and weapons delivery computer.

Several approaches have been considered for dealing with the problem of INS failure. One approach would change the input filter of the autopilot and weapons delivery computer to smooth the low-rate navigation signal from the GPS. Another approach would essentially do the same thing in the navigation computer before the signal is sent to the other computer. Both of these suffer from the fact that the high frequency information found in the INS signal is absent no matter how smooth the output. The third and most comprehensive approach is to combine information from the other aircraft sensors, such as heading, pitch, and airspeed, with the GPS information to produce a high-rate navigation solution. Unfortunately, signals other than those from the INS and GPS have poorly defined error characteristics and come in a variety of data rates. A navigation system based on these signals would have to account for these errors either through filtering of the outputs or estimating and correcting the sources of the errors. Even with this approach, the error characteristics of the output navigation signal will differ from those of a signal based on INS information. Therefore, the input filter of the autopilot and weapons delivery computer will have to be adjusted in any case.

1.2 Objective

The objective of this thesis is to develop and evaluate a mathematical model of a navigation filter described above in the third approach. The overall system will be developed in block diagram form and then the mathematical models of each of the system blocks will be derived. Testing of the models is performed by simulation in Matlab. If testing validates the mathematical models, then engineers at the C. S. Draper Laboratory can implement the concepts into their existing navigation software as a backup mode of operation.

1.3 Outline of the Thesis

The presentation shall first cover the background of the project to include the current system description and a formal problem statement. The top-level design section will show a block diagram of the proposed system and describe the functions of each block. The following chapters will describe the detailed design of each of the top-level blocks and summaries of their testing. The conclusion will summarize the results of the project and suggest future avenues of development. Appendix A contains the Matlab functions developed for the project. Appendix B contains the Matlab scripts used in testing and plots of the results.

2. Background

2.1 A-10 Aircraft Description

The A-10 is a single-seat ground-attack aircraft designed to deliver free-fall and forward firing weapons on point targets while performing aggressive maneuvers. Airspeeds can range from 120 to 450 nautical miles per hour. The aircraft is capable of performing 7.3G acceleration turns resulting in turn rates of up to 21 degrees per second. The on-board Gatling gun is capable of hitting targets at ranges up to 2.5 nautical miles, and free-fall weapons can be released from altitudes above 10,000 feet. The very high maneuverability of the aircraft and long range weapons capability require the navigation system to provide high-rate, accurate, and wide-bandwidth state estimates.

2.2 Current Navigation System Description

The current A-10 navigation system consists of an Inertial Navigation Set (INS), Heading and Attitude Reference System (HARS), Central Air Data Computer (CADC), Low Altitude Safety and Targeting Enhancement (LASTE) computer, Angle-Of-Attack (AOA) vane, and a Stability Augmentation System (SAS). These systems are minimally integrated in the aircraft. The INS acts as the primary navigation data source (except for altitude which is taken from the CADC). The HARS measures aircraft attitude and heading using a gyroscopically stabilized platform slaved to the local vertical by an erection loop and synchronized with a magnetic compass sensor to correct for heading drift errors. HARS attitudes and headings are ignored by the navigation system when the INS is operative. The CADC measures static and dynamic air pressures and outside air temperature from which it calculates altitude and airspeed. The purpose of the SAS system is to dampen yaw oscillations and as a by-product it produces an estimate of the aircraft sideslip. The sideslip and AOA data are provided to the LASTE computer for autopilot and weapons delivery use, but are not currently integrated into the navigation solution.

2.3 New Navigation System Architecture

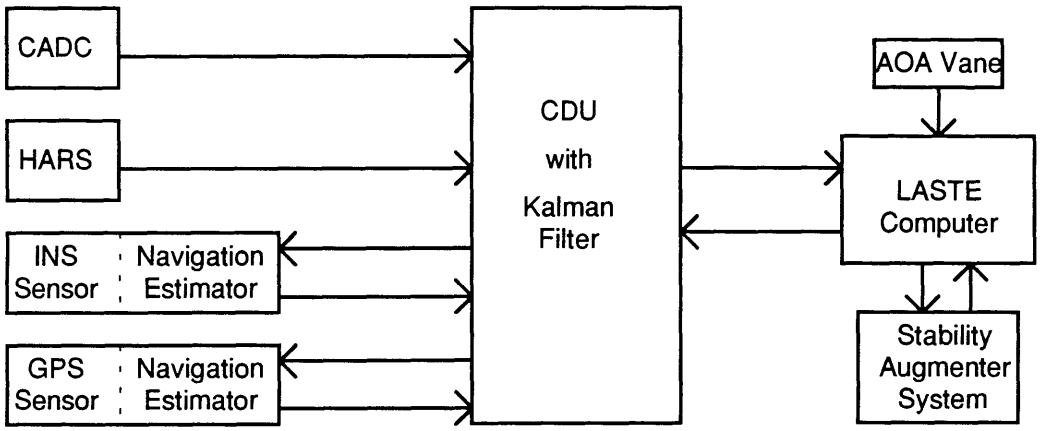

Figure 1 New Navigation system architecture

The GPS integration program will significantly modify the navigation system by adding a Control Display Unit (CDU) which will integrate the navigation information from all of the available sensors using a Kalman filter. The INS directly measures aircraft attitude and specific force. These measurements are combined and integrated over time to yield velocity and position and are provided to the CDU at 50Hz. The GPS receiver directly measures position and velocity from satellite information once a second. The navigation estimator uses "aiding" information from the INS to extrapolate a position and velocity solution which it sends back to the CDU at 12.5Hz. (Note the GPS is an existing system designated by the government and cannot be modified. The system was designed to provide a stand-alone 12.5Hz navigation state solution.) In the normal navigation mode, the CDU corrects the INS output using information from the GPS and the CADC to provide a 50Hz navigation solution to the LASTE computer. The LASTE computer combines the navigation solution with AOA and sideslip to provide autopilot and weapons aiming functions. The AOA and sideslip measurements are not directly available to the CDU, but could be passed through the LASTE computer with a software update.

2.4 Problem Statement

In current design, if the INS fails, the CDU uses the 12.5Hz GPS output as its navigation signal. It passes a 50 Hz signal to the LASTE computer by replicating each state reported by the GPS three times. An additional problem is that the GPS navigation estimator is now drawing on HARS and CADC data instead of INS data to propagate the states between satellite updates. The HARS and CADC data are not corrected for AOA and sideslip values, nor are they as accurate as the INS. Therefore, the GPS velocity estimations diverge from the actual state until the next satellite measurement is incorporated and the estimates are instantly corrected. This causes a ramp type error, followed by a discontinuity, repeated once a second, as shown in Figure 2. As currently configured, the autopilot and weapons functions of LASTE cannot function adequately with the data errors and discontinuities. The objective of this thesis is to redesign the GPS/HARS mode to provide a 50Hz navigation signal to the LASTE computer, with the best accuracy feasible.

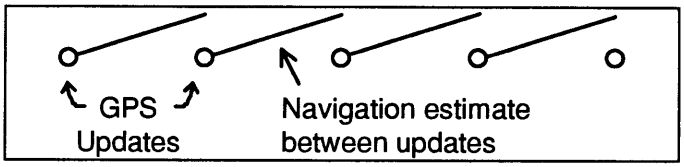

Figure 2 Example of positional navigation error.

3. Top Level Design

3.1 Block Diagram

The proposed approach is to design a cascade filter with a high-rate navigation estimator running at 50Hz and a low-rate error estimator running once each second. The high-rate section will convert heading, attitude, AOA, sideslip, and airspeed information into Earth-surface frame velocity measurements. It will then combine these measurements with the 12.5Hz position and velocity measurements from the GPS to provide position and velocity estimates at 50Hz. Concurrently, the low-rate portion will estimate applicable bias, scale factor, and gyroscope tilt errors in the sensor signals and adjust the high-rate estimator parameters and output to compensate. The result will be fed into the LASTE computer and back to the GPS for velocity aiding.

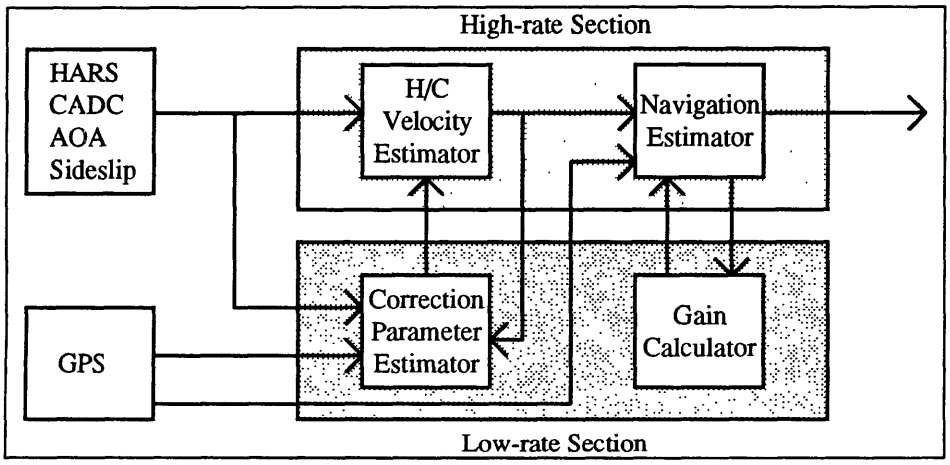

Figure 3 Top Level Design

3.2 Reference Frames and Variable Definitions

Several reference frames [1] will be used in the ensuing discussion. The subscript L will denote the Earth-surface frame. This reference frame uses an origin on the Earth's surface, x_L points true North, y_L points East, and z_L points downward. When describing velocities however, the subscripts n, e, d will denote the North, East, and down velocities in the Earth-surface frame. The subscript A will denote the atmosphere fixed reference frame. The origin is a point in the air near the aircraft which follows the average motion

of the air mass. The body fixed reference frame will use the subscript B. The origin is the center of mass of the aircraft; x_B is the roll axis; y_B points out the right wing; and z_B points out the bottom. The HARS reference frame (subscript H) is the same as the Earth reference frame except that the axes are oriented to magnetic North and differ by the local magnetic variation. When discussing body frame velocities, the subscripts x, y, and z will be used without the B. Therefore, V_x, V_y, V_z represent the velocity of the aircraft with respect to the Earth surface resolved into the body frame. V_n, V_e, and V_d represent that same velocity resolved into the Earth-surface frame.

3.3 HARS/CADC Velocity Estimator

The HARS/CADC velocity estimator takes true airspeed and body angles and derives velocities in the Earth-surface frame. The HARS provides the aircraft body angles at 50Hz and the CADC provides airspeeds at 25Hz. The slower data rate of the CADC will not be a significant source of error since the aircraft's airspeed changes at a much slower rate than the body angles. First, the CADC reported airspeed is broken up into body frame coordinates using the AOA and sideslip angles. Next, the body frame velocities are rotated into the atmosphere frame using the heading pitch and bank values. Finally, winds are added to shift the atmosphere frame velocities to Earth-surface frame. The corrections calculated in the error estimator (described below) will primarily be applied through the HARS/CADC velocity estimator. The complete design will be discussed in the next chapter.

3.4 Navigation Estimator

The navigation estimator takes the output of the HARS/CADC velocity estimator and combines them with the position and velocity estimates from the GPS. The design will be discussed in detail in later chapters; but some points bear discussing here. First, the navigation estimate must be completed 50 times each second. Also, the processor must perform other tasks such as running the error estimator, input/output functions, etc.

Therefore, a primary goal of the design is to minimize computational burden. This affects the modeling of noise characteristics and the form of estimator used (steady state versus time varying). Later chapters will show the navigation estimator to be a modified form of a steady state estimator.

3.5 Error Estimator

The error estimator rate will depend on the computation time available, but is intended to run on the order of once per second. It will use a Kalman filter to estimate the errors in the signals provided to the navigation estimator and generate corrections. The error estimator states will be the correction parameters for these signals. The correction parameters include corrections for gyroscope misalignment, Euler angle measurement biases, airspeed scale factor, airspeed bias, and Earth frame wind velocities.

3.6 Gain Calculator

The gain calculator will account for the low-rate changes in the optimal fixed g n values for the navigation estimator. The choice of a fixed gain design will be discussed in detail in the navigation estimator chapter. In the current design, the optimal fixed gain varies only with the aircraft latitude. These calculations were simple enough to include in the high-rate section, so the gain calculator was absorbed into the navigation estimator. However, the gain calculator may be used in follow-on developments to rotate the noise characteristics as the aircraft turns to allow a noise model tailored to the body frame rather than the Earth-surface frame. In this case, the gain calculator will have to be moved back into the low-rate section.

4. HARS/CADC Velocity Estimator

4.1 Body Frame Velocities

The HARS/CADC velocity estimator uses airspeed and body angles to derive velocities in the atmosphere frame. First, velocities are developed in body coordinates and then rotated into the atmosphere frame. Since the pitot tube lies along the x-axis of the aircraft, it will directly sense V_x rather than the total velocity **V**.

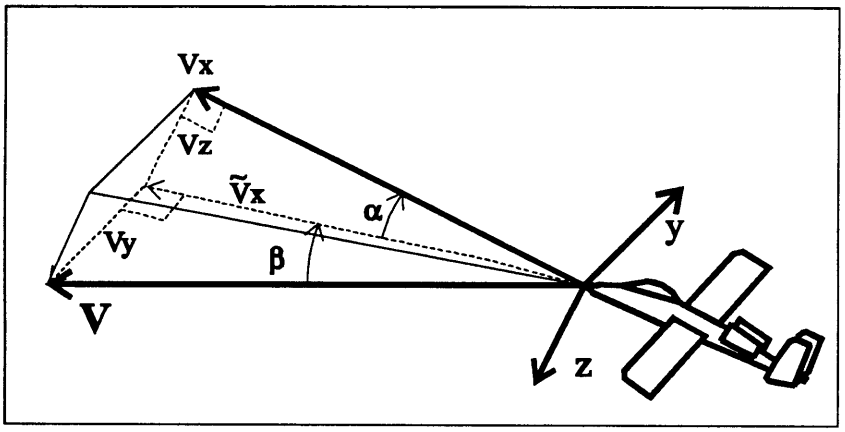

Figure 4 AOA and Sideslip

AOA and sideslip are represented by the variables α and βrespectively and are defined as [1]

$$\alpha = \tan^{-1}\frac{V_z}{V_x} \qquad \beta = \mathrm{SIN}^{-1}\frac{V_y}{|V|}$$

where |V| is the magnitude of the total velocity. This means that AOA is measured in the plane formed by V_x and V_z about the y axis. Sideslip is measured in the plane formed by V_y and **V** about the **V**×y axis. (Since these values are not currently provided by LASTE, the specifications must be written to ensure they are so defined.)

The following formulas can be derived from figure 4.

$$\tilde{V}_x = |V|\cos(\beta)$$

$$V_x = \tilde{V}_x\cos(\alpha) = |V|\cos(\beta)\cos(\alpha)$$

$$V_y = -|V|\sin(\beta)$$

$$V_z = \tilde{V}_x\sin(\alpha) = |V|\cos(\beta)\sin(\alpha)$$

Since the CADC measures V_x the above equations can be rewritten as

V_x = CADC Airspeed

$V_y = -V_x \tan(\beta)/\cos(\alpha)$

$V_z = V_x \tan(\alpha)$

The equations can now be rewritten in vector notation to give the body frame velocities $\underline{V}_B$ as a function of the CADC airspeed V_C.

$$\underline{V}_B = \begin{bmatrix} 1 \\ -\tan(\beta)/\cos(\alpha) \\ \tan(\alpha) \end{bmatrix} V_C = \underline{V}_C^B V_C$$

4.2 Rotating Body Frame to Atmosphere Frame

The velocities are then rotated into atmosphere frame using direction cosine matrices.

$$\begin{bmatrix} v_n \\ v_e \\ v_d \end{bmatrix}_A = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}_B$$

ψ - azimuth angle measured about the down axis, positive clockwise

θ - pitch angle measured about the down×x axis, positive clockwise

ϕ - bank angle measured about the x axis, positive clockwise

Note that the heading angle ψ must first be corrected for the magnetic variation since HARS uses a magnetic heading sensor to stabilize the directional gyro. Multiplying the three rotation matrices together gives the equation

$$\begin{bmatrix} v_n \\ v_e \\ v_d \end{bmatrix}_A = \begin{bmatrix} \cos\psi\cos\theta & \cos\psi\sin\theta\sin\phi - \sin\psi\cos\phi & \cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi \\ \sin\psi\cos\theta & \sin\psi\sin\theta\sin\phi + \cos\psi\cos\phi & \sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}_B$$

The eventual goal is to achieve Earth-surface frame velocities. Since the only difference between atmosphere frame velocities and those of Earth-surface frame is the addition of a wind vector, the direction cosine matrix will be designated as the body-to-Earth-surface-frame rotation matrix C_B^E. Combining the rotation matrix and the body-frame velocity equation gives the (almost) final HARS/CADC velocity equation.

$$\underline{V}_H = C_B^E \, \underline{V}_B, \qquad \underline{V}_B = \underline{V}_C^B \, V_C$$

$$\underline{V}_H = C_B^E \, \underline{V}_C^B \, V_C$$

where $\underline{V}_H$ is the Earth-surface frame velocities derived from the HARS/CADC data. This is the basic HARS/CADC velocity equation; but it will be modified in Chapter 7 to allow the error estimator to apply corrections to the HARS and CADC data and to account for wind.

5. Navigation Estimator Design

5.1 Description of Model

The navigation system model is derived in the standard discrete-time state-space format:

$$\underline{x}(k+1) = \mathbf{A}\underline{x}(k) + \mathbf{B}\underline{u}(k) + \mathbf{G}\underline{w}(k)$$
$$\underline{y}(k) = \mathbf{C}\underline{x}(k) + \mathbf{D}\underline{u}(k) + \underline{v}(k)$$

where $\underline{x}$ represents the states and $\underline{y}$ represents the measurements. The sensor signals are treated as the measurements and there are no inputs ($\underline{u}$), so **B** and **D** are both zero. The states and measurements are given by:

$$\text{States: } [p_n \quad p_e \quad p_d \quad v_n \quad v_e \quad v_d]^T$$
$$\text{Measurements: } [p_{Gn} \quad p_{Ge} \quad p_{Gd} \quad v_{Gn} \quad v_{Ge} \quad v_{Gd} \quad v_{Hn} \quad v_{He} \quad v_{Hd}]^T$$

where p_n: Latitude in degrees

p_e: Longitude in degrees

p_d: Altitude in feet

$v_{n,e,d}$: North, East, and Down velocities with respect to the Earth surface.

G = GPS, H = HARS/CADC.

Note that the filter was initially designed and tested with units of feet in all the position states and feet per second in the velocity states. The model was later converted to the current units, which is how the model is derived below. However, many of the basic filter properties were investigated with the first set of units. Because of this, plots of the results may be in either set of units. The change of units did not significantly affect any of the filter characteristics.

The filter operates at 50Hz which gives a 0.02 second cycle time. Considering the velocities to be constant over this short time period leads to the propagation equations:

$$p_n(k+1) = p_n(k) + v_n(k) \text{ Nm/Hr} * 1\text{Hr}/3600\text{Sec} * 1\text{Degree}/60\text{Nm} * 0.02 \text{ Sec/Cycle}$$
$$= p_n(k) + v_n(k)/1.08e7$$

The number of nautical miles per degree of longitude changes depending on the current

latitude so the East position equation must compensate:

$$p_e(k+1) = p_e(k) + v_n(k)/(\cos(p_n(k)) * 1.08e7)$$

The down position is in feet whereas the down velocity is in knots:

$$p_d(k+1) = p_d(k) + v_d(k)\ \text{Nm/Hr} * 0.02\ \text{Sec/Cycle} * 1\ \text{Hr/3600Sec} * 6080\text{Feet/1Nm}$$

The **A** matrix is:

$$\begin{bmatrix} 1 & 0 & 0 & 9.259e-8 & 0 & 0 \\ 0 & 1 & 0 & 0 & \frac{9.259e-8}{\cos(p_n(k))} & 0 \\ 0 & 0 & 1 & 0 & 0 & \frac{38}{1125} \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

This maintains the velocity constant while propagating the position. Obviously, the model is incorrect in that it does not account for acceleration. But, no inputs or direct measures of acceleration are available, so acceleration is modeled as a zero-mean noise and will be discussed further in the next section. The **C** matrix maps the six states back to the nine measures.

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

C Matrix

The estimator uses the following discrete-time Kalman filter equations [2]:

$$\mathbf{K}(k) = \mathbf{P}_p(k)\mathbf{C}^T\left[\mathbf{C}\mathbf{P}_p(k)\mathbf{C}^T + \mathbf{R}(k)\right]^{-1}$$

$$\underline{x}_u(k) = \underline{x}_p(k) + \mathbf{K}(k)\left(\underline{y}(k) - \mathbf{C}\underline{x}_p(k)\right)$$

$$\mathbf{P}_u(k) = [\mathbf{I} - \mathbf{K}(k)\mathbf{C}]\mathbf{P}_p(k)[\mathbf{I} - \mathbf{K}(k)\mathbf{C}]^T + \mathbf{K}(\mathrm{k})\mathbf{R}(k)\mathbf{K}^T(k)$$

$$\underline{x}_p(k+1) = \mathbf{A}\underline{x}_u(k)$$

$$\mathbf{P}_p(k+1) = \mathbf{A}\mathbf{P}_u(k)\mathbf{A}^T + \mathbf{G}\mathbf{Q}(k)\mathbf{G}^T$$

where **Q** contains the process noise variances and **R** contains the measurement noises (described below).

5.2 Noise Characteristics

The noise characteristics for the high-rate filter are based on the premise that the low-frequency sensor errors are uncorrellated with the high-frequency position and velocity noise. And that the error estimator will remove the low-frequency noise leaving a comparatively small, zero-mean, white noise error with respect to the navigation state. The measurement error variances are based on probable error values for the GPS and HARS sensors. These values were supplied by Mr. Jim Donna, a project engineer at Draper Laboratory. The probable error values were divided by 0.674 to give the standard deviation of the errors assuming a normal distribution. This gave a GPS position error of 7.4 feet standard deviation and 0.49 f/s in velocity. It is important to realize that these are the high-frequency components of the GPS error. They describe the volatility of the GPS errors over short duration (on the order of seconds) and assume no satellite changes affect the GPS position solution during the period of estimation. The current design of the navigation system already has a technique for managing satellite changes (Q bumping) so it will not be considered further here. The GPS and HARS noise values were then converted into the appropriate units and squared to get the variance

$\mathbf{R}$ = diag[4.14e-10, 4.14e-10/cos^2(p_n), 55.03, 0.19, 0.19, 0.19, 3.09e-2, 3.09e-2, 3.09e-2]

where the first three values correspond to GPS position, the next three to GPS velocity, and the last three to HARS velocity.

Derivation of the process noise characteristics was based on the unmodeled dynamics. Acceleration is based on the model:

$$\dot{\underline{v}} = 0 + \text{noise}$$

The A-10 can turn at a maximum of 7G's of acceleration [3] which is equivalent to 224f/s^2. Applying the acceleration over the 0.02 second filter propagation time gives a

maximum velocity change of 4.48f/s between updates. Since this is a maximum, one-half was used as an estimate of the acceleration standard deviation. This value is somewhat arbitrary. Choosing a larger value will make the filter more responsive to aircraft maneuvering, but less damped during unaccelerated flight. Choosing a smaller number will do the reverse. Propagating the position over 0.02 seconds with the standard deviation of the velocity error gave the standard deviation of the position error as 0.0224 feet. Replicating these for each direction, converting them to their appropriate units and squaring each term gives:

$\mathbf{Q}$ = diag [3.7704e-15, 3.7704e-15/$\cos^2(p_n)$, 5.0176e-4, 1.7591, 1.7591, 1.7591]

where the first three values correspond to the position estimation and the last three to the velocity estimate.

Note that the acceleration capability of the aircraft is not the same in each axis, nor are the probable errors of the HARS/CADC measurements. In both of these cases, the difference is apparent in the body coordinate frame and must be rotated into the Earth-surface frame based on the aircraft heading, pitch and bank. Using a time varying noise matrix requires the propagation of the error covariances in the filter equations. Since the update rate of the navigation filter is very fast compared to the turn rate of the aircraft, it was decided not to use the orientation dependent noise characteristics in the high-rate filter. The low-rate filter can be made to account for heading changes and calculate new optimal fixed gains for the high-rate filter if this turns out to be necessary.

5.3 Multirate Signal Problem

5.3.1 Multirate Measurements

There are four HARS velocity measurements for each GPS position and velocity measurement. To accommodate the different rates, the measurement vector changes from nine states when GPS is available to three states when only HARS is available. Correspondingly, the **C** and **R** matrices are reduced in order. The new **C** matrix (called

C1 in the Matlab scripts) is that part of the full matrix that maps HARS velocities to the velocity states. **R1** is a submatrix of **R** which contains those covariances exclusively derived from HARS noises. The gain matrix **K** reduces from 6x9 to 6x3, but is calculated with the same Riccati equation.

$$\mathbf{C1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{R1} = \begin{bmatrix} 309 & 0 & 0 \\ 0 & 309 & 0 \\ 0 & 0 & 309 \end{bmatrix}$$

5.3.2 Repeated Fixed Gains

The existence of multiple data rates dictates that the optimal filter cannot be fixed gain. This is because the error covariances necessarily change each time the components of the measurement change. Each time the GPS measurements are included, the error covariances drop; each time the GPS measurements are excluded the covariances rise. However, because of the repetitive structure of the measurements, it was possible for the gains to converge to a series of four repeated values. To test this hypothesis, a 1200 point simulation was run using *navest*, a Matlab function developed by the author for this project. The *navest* function, included in Appendix A, is a discrete-time, mulitrate, variable-gain filter which allows the user to designate the number of high-rate samples per low-rate sample, and the subset of measurements to use for the high-rate sample. The script file *Kalm9* simulated the measurements and called *navest*. The results showed that the error covariances converged to a set of four repeating values. The velocity covariances converged very quickly. The position values rapidly converged to within a few percent of their final values, but then continued to converge

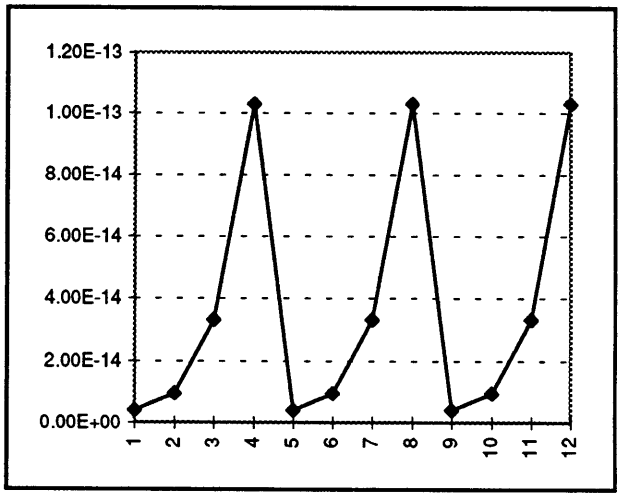

Figure 5 Steady State Position Error Variances After Update

asymptotically slowly. The fact that they did actually converge was confirmed by initiating the filter with values above and below the final values and noting that the covariances monotonically converged to some center value. By alternately running above and below passes, the values were determined to six significant digits. The Matlab script and plots of the gain values are shown in Appendix B. Figure 5 shows a representation of just the steady-state position-error variances after the update step. As a result, the optimal Kalman gains also showed a steady state repetition of four values.

5.3.3 Latitude Compensation

The steady state values varied according to the aircraft latitude. Table 1 shows the

Table 1 Filter Gain Values

				K1				
.82511	0	0	3.80E-11	0	0	2.37E-10	0	0
0	.82511	0	0	3.80E-11	0	0	2.37E-10	0
0	0	6.22E -03	0	0	7.87E-05	0	0	4.92E-04
1771.5	0	0	.13591	0	0	.84944	0	0
0	1771.5	0	0	.13591	0	0	.84944	0
0	0	2.76E-07	0	0	.13591	0	0	.84944
				K2				
0	0	0	0	0	0	1.34E-09	0	0
0	0	0	0	0	0	0	1.34E-09	0
0	0	0	0	0	0	0	0	4.96E-4
0	0	0	0	0	0	.98300	0	0
0	0	0	0	0	0	0	.98300	0
0	0	0	0	0	0	0	0	.98300
				K3				
0	0	0	0	0	0	1.57E-09	0	0
0	0	0	0	0	0	0	1.57E-09	0
0	0	0	0	0	0	0	0	5.72E-04
0	0	0	0	0	0	.98304	0	0
0	0	0	0	0	0	0	.98304	0
0	0	0	0	0	0	0	0	.98304
				K4				
0	0	0	0	0	0	1.57E-09	0	0
0	0	0	0	0	0	0	1.57E-09	0
0	0	0	0	0	0	0	0	5.73E-04
0	0	0	0	0	0	.98304	0	0
0	0	0	0	0	0	0	.98304	0
0	0	0	0	0	0	0	0	.98304

four Kalman gains for zero degrees of latitude. The highlighted entries vary with latitude while the rest remain fixed.

Plotting the highlighted steady state gains as a function of latitude revealed a simple relationship between the gains at the equator and at other latitudes (Figures 6 and 7). In each case but one, the gain at a particular latitude equaled the gain at the equator divided by the cosine of the latitude. In the case of the gain which relates the East GPS velocity measurement to the East position state, the gain at latitude equaled the gain at the equator times the cosine of the latitude.

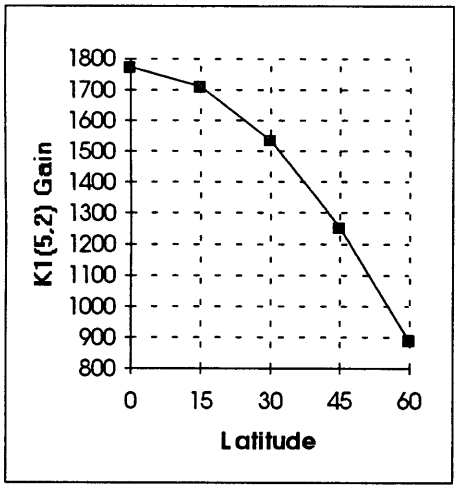

Figure 6 Gain Vs Latitude

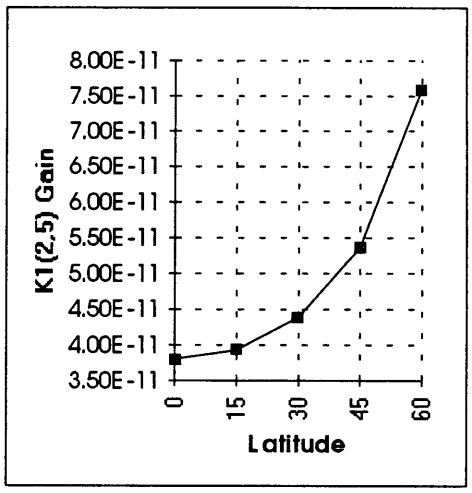

Figure 7 Gain Vs Latitude

These values were used to design a multiple-fixed-gains filter where the Riccati equation is removed and the four fixed gains are rotated into the measurement update equation. This is the filter represented in the Matlab function *ssnavest* included in Appendix A.

6. Navigation Estimator Testing

6.1 General Description

Three tests were run on the navigation filter. The first was a 24 second constant velocity test to evaluate how well the filter tracked a non maneuvering trajectory. The second was a four second simulation including a single velocity change over a two second time period. The third simulated a 90 degree hard turn from North to East over four seconds. In each case, a normally-distributed, zero-mean, random noise was added to the measurements. The GPS position noise had an RMS value of 7.4 ft, the GPS velocity had 0.74 f/s, and the HARS velocity had 0.3 f/s. The GPS measurements were then run through a zero order hold function to convert them to a 12.5Hz signal. This was done with the *gpssig* function in Appendix A.

6.2 Constant Velocity Test

The constant velocity test was initiated with a 400f/s North velocity, 200f/s East velocity, and zero down velocity. The Matlab script *Kalm6*, which ran the test, is included in Appendix B along with the plots of the test results. The errors in the velocity estimate resembled zero-mean white noise. Statistics gathered on the errors in the velocity estimate reveal that the mean errors were 0.0121, -0.0105, and -0.0038 f/s across the three directions and the RMS errors were 0.2914, 0.2868, and 0.2908 f/s. These RMS values represent a significant reduction compared to the 0.74 f/s RMS error in the GPS measurements, but only a very slight change to the 0.3 f/s RMS error of the HARS velocities. This is not unexpected because the gains which map the HARS velocities to the state velocities were on the order of 0.9 while those that map GPS velocity to the state velocity were on the order of 0.1 every fourth step and zero on the other three. So it would be expected that the HARS velocity noises translate to the state velocities with very little attenuation.

The plots of the errors in the position state resemble a zero-mean random walk process with bounded standard deviation. The position error means were -0.0479, 0.2327, and -0.3819ft. The RMS errors were 0.5662, 0.5922, and 0.6448 ft. The RMS errors represent significant reductions compared to the GPS measurement RMS error of 7.4 feet. This small of an error is more due to the small velocity error covariances propagating across a short time interval, and the very small plant position noise covariances.

6.3 Single Velocity Change Test

The single velocity change test was initiated with a 400f/s North velocity, 200f/s East velocity, and zero f/s down velocity. At one second into the run, a 150 f/s^2 acceleration was added to the down velocity measurements. This translates to 3 f/s velocity change per high-rate update cycle which is slightly above the assumed standard deviation of the plant noise of 2.24 f/s per cycle. The acceleration was maintained for two seconds and then returned to zero for the final second of the test. The Matlab script, *Kalm6,* which ran the test is included in Appendix B along with the plots of the test results.

Due to the extreme reliance on HARS velocity data, the filter showed no apparent lag in response to the acceleration input. The RMS errors in velocity showed only a 0.6% increase over the constant velocity case while the RMS errors in position showed a 0.8% increase.

6.4 Hard Turn Test

The hard turn test was initiated with a 500f/s North velocity and zero East and down velocity. An immediate turn to the East is simulated by providing four stages of acceleration starting with 50 f/s^2 South and 200 f/s^2 East acceleration and ending with 200f/s^2 South and 50 f/s^2 East acceleration. This had the effect of turning the velocity vector from due North to due East in four seconds. The maximum accelerations observed

were 4f/s per cycle as compared to the 2.24f/s per cycle assumed standard deviation of the plant noise. The Matlab script *Kalm6,* which ran the test, is included in Appendix B along with the plots of the test results.

Once again, the filter proved to be very responsive to the acceleration inputs. The velocity RMS errors showed little reduction from the HARS velocity RMS errors. Both the velocity and position RMS errors showed a 0.8% increase from the constant velocity case.

7. Error Estimator Design

7.1 Introduction of Correction Parameters

7.1.1 HARS/CADC Velocity Equation

The first stage of the high-rate section converts HARS/CADC measurements into Earth-surface frame velocities. Unfortunately, the measurements are not perfect and corrections need to be applied. Recall the equation for HARS Earth frame velocities: $\underline{V}_H = C_B^E \; \underline{V}_C^B \; V_C$. V_C is the CADC airspeed; $\underline{V}_C^B$ translates CADC airspeeds into body frame velocities; and C_B^E rotates body coordinates to Earth-surface frame. Including the parameters in the equation gives:

$$\underline{V}_H = C_B^E(\psi,\theta,\phi)\,\underline{V}_C^B(\alpha,\beta)\; V_C$$

where ψ,θ,ϕ = heading, pitch, yaw, and α,β = AOA, sideslip. Adding correction parameters for gyroscope misalignment, Euler angle biases, airspeed scale factor, and airspeed bias gives:

$$\underline{V}_H = (I + Z)\, C_B^E(\psi+\Delta\psi,\, \theta+\Delta\theta,\, \phi+\Delta\phi)\, \underline{V}_C^B[V_S(V_C - 300) + V_B + 300] + \underline{W}$$

where Z is a rotation matrix that corrects for gyroscope misalignment; the Δvalues correct the Euler angles; and $\underline{W}$ is the wind vector. $[V_S(V_C - 300) + V_B + 300]$ is the adjustment of the CADC supplied velocity V_C using linear corrections about 300 knots. VS is the scale factor; V_B is the airspeed bias

Z is derived from a series of rotation matrices about the n, e, d directions. C_B^E is the product of three rotation matrices affecting heading, pitch, and bank. The heading matrix always rotates about the vertical axis, and exactly matches the Z matrix about the down direction. There is no value in tracking separate correction parameters for vertical misalignment and heading bias, because they are calculated in the same manner and have the same effect. Therefore, $\Delta\psi$ will be dropped in favor of Z.

7.1.2 Derivation of Misalignment Corrections

This derivation is based on linearizing the rotation equation discarding terms of order greater than one. Let $\underline{\zeta} = [\ \zeta_N\ \zeta_E\ \zeta_D\ ]^T$ represent a small rotation about the direction of $\underline{\zeta}$ where each of the $\zeta_{N,E,D}$ represents the misalignment of the respective HARS axis in radians. The total misalignment angle would be $\sqrt{\zeta_N^2+\zeta_E^2+\zeta_D^2}$. The cross product $\underline{\zeta}\times\underline{\tilde{V}}$ provides the error vector that corrects $\underline{\tilde{V}}$ to $\underline{V}$, where $\underline{\tilde{V}}$ is the Earth-surface frame velocity not yet corrected for misalignment (Figure 8).

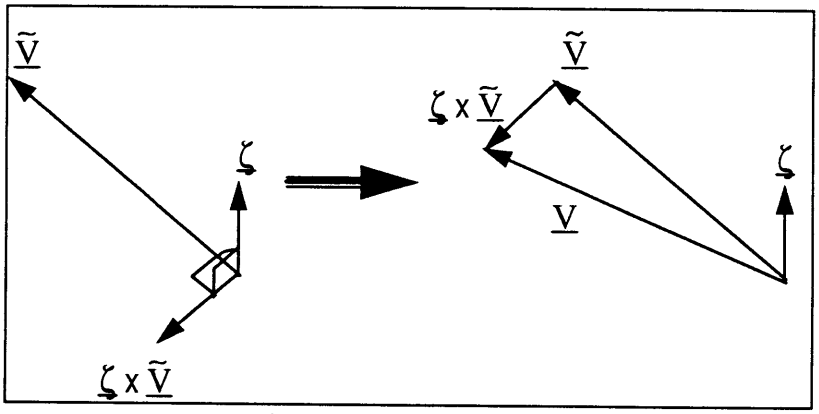

Figure 8 Effect of (I+Z)

$$\underline{\zeta}\times\underline{\tilde{V}} = \begin{vmatrix} N & E & D \\ \zeta_N & \zeta_E & \zeta_D \\ \tilde{V}_N & \tilde{V}_E & \tilde{V}_D \end{vmatrix} = (\zeta_E\tilde{V}_D - \zeta_D\tilde{V}_E)N + (\zeta_D\tilde{V}_N - \zeta_N\tilde{V}_D)E + (\zeta_N\tilde{V}_E - \zeta_E\tilde{V}_N)D$$

$$= \begin{bmatrix} \zeta_E\tilde{V}_D - \zeta_D\tilde{V}_E \\ \zeta_D\tilde{V}_N - \zeta_N\tilde{V}_D \\ \zeta_N\tilde{V}_E - \zeta_E\tilde{V}_N \end{bmatrix} = \begin{bmatrix} 0 & -\zeta_D & \zeta_E \\ \zeta_D & 0 & -\zeta_N \\ -\zeta_E & \zeta_N & 0 \end{bmatrix} \begin{bmatrix} \tilde{V}_N \\ \tilde{V}_E \\ \tilde{V}_D \end{bmatrix} = Z \begin{bmatrix} \tilde{V}_N \\ \tilde{V}_E \\ \tilde{V}_D \end{bmatrix}$$

But $\underline{V} = \underline{\tilde{V}} + \underline{\zeta}\times\underline{\tilde{V}} = I\underline{\tilde{V}} + Z\underline{\tilde{V}} = (I+Z)\underline{\tilde{V}}$.

$$\text{Then } (I+Z) = \begin{bmatrix} 1 & -\zeta_D & \zeta_E \\ \zeta_D & 1 & -\zeta_N \\ -\zeta_E & \zeta_N & 1 \end{bmatrix}$$

7.2 Error Estimator Models

The low-rate filter will estimate the correction parameters ζ_N, ζ_E, ζ_D, $\Delta\theta$, $\Delta\phi$, V_S, V_B, W_N, W_E, W_D (Table 2). The State vector and measurement vector will be:

$$\underline{x} = [\zeta_N, \zeta_E, \zeta_D, \Delta\theta, \Delta\phi, V_S, V_B, W_N, W_E, W_D]$$

$$\underline{y} = [V_{GN}, V_{GE}, V_{GD}]$$

Table 2 Error Estimator States

$\zeta_{N,E,D}$	Represents the angle in radians that the HARS Gyroscope axes are misaligned.
$\Delta\theta$	Represents the pitch bias angle in radians.
$\Delta\phi$	Represents the bank bias angle in radians.
V_S	Represents the CADC airspeed scale-factor error.
V_B	Represents the CADC airspeed bias error.
$W_{N,E,D}$	Represents the wind velocities in knots.
$V_{GN,E,D}$	Represents the GPS velocity measurements in knots.

Figure 9 shows a block diagram of how the error estimator interacts with the rest of the system.

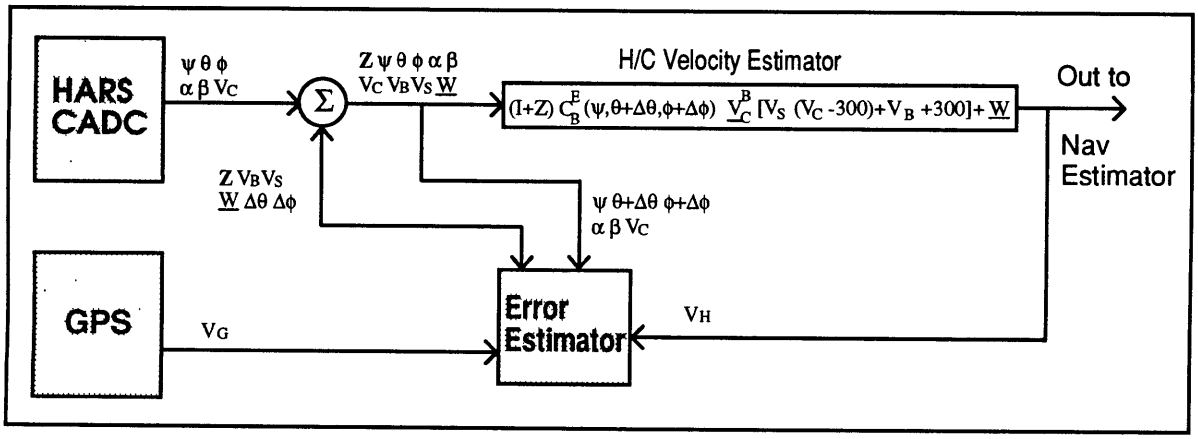

Figure 9 Error Estimator Block Diagram

7.2.1 HARS Misalignment Model

Several models were considered and tested for the HARS misalignment. These included modeling the misalignments as constants and as Markov processes with various noise and damping characteristics. The final decision was to use a first order Markov process based on expected steady state misalignment values and estimated erection loop control gains. The HARS gyroscopes have a constant component to their drift rate[4]. This rate is random from flight to flight with a standard deviation of about 3 degrees per

hour, but remains constant once the gyroscopes are aligned. On top of this drift are smaller random components on the order of 0.05 degrees per hour. The erection control loops provide torque inputs to correct the gyroscope drift during periods of relatively unaccelerated flight and have a time constant of about 150 seconds. The balance between the drift rates and the erection loops gives a steady state misalignment variance on the order of 0.1 degrees^2.

The first order Markov process model is given by

$$\dot{x} = -\frac{1}{\tau}x + \text{noise}$$

where $\tau = 150$. The noise is found by dividing the expected variance by the time constant and converting to radians per second.

$$\frac{(.1\ \text{deg})^2}{150\ \text{sec}} * \left(\frac{\pi\ rad}{180\ \text{deg}}\right)^2 = 2.031x10^{-8}\ \frac{rad^2}{\text{sec}}$$

Integrating the Markov process gives the discrete time propagation equation.

$$x(k+1) = e^{-\frac{\Delta t}{\tau}} x(k) + w(k)$$

where $e^{-\frac{\Delta t}{\tau}} = 0.9934$ and w(k) is a Gaussian random variable with a variance of $2.031x10^{-8}$. This formula is used for the first three components of the function f in the state update equation discussed later.

7.2.2 Wind Model

The wind estimates need to be very responsive. Wind gusts or rapid wind changes due to climbs or dives affect the pitot static measurement of airspeed much faster than they affect the actual aircraft motion. Therefore, a major part of the velocity difference between CADC and GPS will be the wind speed. Since the CADC is affected by the current wind speed, long duration wind speed averages commonly used with inertial systems will not suffice as corrections.

Wind above the ground boundary layer (on the order of 1000 feet) is usually modeled as homogeneous, isotropic, Gaussian, and frozen turbulence[1]. It is isotropic because above the boundary layer, the wind speed variances are equal in all directions.

Within the boundary layer the vertical variance decreases relative to the horizontals. The wind model is frozen in space because the speed of the aircraft is high relative to the wind speed changes; therefore the wind velocities can be modeled as changing only because the aircraft is flying into new areas. Unfortunately, wind speeds are chaotic and do not lend themselves well to encompassing variance numbers. The most common approach is to develop a Fourier model of observed wind speed data and then select dominant wave numbers in the scale of interest. The choice of scale is rather arbitrary, but for this project it is appropriate to ignore "buffeting" but track "short" duration wind changes. Etkin[1] provides an example of a dominant wave length of about 3.9Nm. The maximum velocity change from the mean would occur in one quarter wavelength or .97Nm. At a nominal speed of 300Kts this distance would be traveled in 11.6 seconds. Wind speed variations on the order of ten seconds are in the desired range for this project. Etkin[1] states that wind speeds can vary from very little to as much as 16fps in this scale. Taking the worst case, this would give a wind speed variance of 256 $(fps)^2$. Assuming the variance grows linearly with time, this gives a 22.1 $(fps)^2$ variance in one second (the proposed update time of the filter) or 7.7 Kts^2.

Another source of wind data was provided by Captain Tim Oram of the Nellis AFB weather office. Captain Oram provided wind data recorded from weather balloons launched on the Nellis range complex. The balloons recorded the horizontal winds at approximately 100 foot intervals from the surface to 6000 feet above ground. Based on the

Table 3 Wind Speed Data

Sample	Speed	Direction	North Spd	East Spd	ΔN Spd	ΔE Spd
1	22.4	135	-15.84	15.84		
2	22.4	137	-16.38	15.28	-0.54	-0.56
3	21.6	136	-15.54	15.00	0.84	-0.27
4	19.4	145	-15.89	11.13	-0.35	-3.88
5	17.7	153	-15.77	8.04	0.12	-3.09
6	15	166	-14.55	3.63	1.22	-4.41
7	11.3	183	-11.28	-0.59	3.27	-4.22
8	8.9	201	-8.31	-3.19	2.98	-2.60
9	7.6	218	-5.99	-4.68	2.32	-1.49
10	6.2	236	-3.47	-5.14	2.52	-0.46
11	7	245	-2.96	-6.34	0.51	-1.20
12	8	240	-4.00	-6.93	-1.04	-0.58
		Mean	-10.83	3.50	1.08	-2.07
		Std Dev	5.52	9.10	1.50	1.61
		Var	30.44	82.80	2.25	2.61
				MSV	3.41	6.89
				RMS	1.85	2.62

isotropic assumptions and wave number analysis discussed above, wind speeds at 500 foot intervals were selected to represent the amount of change the aircraft might see in one second. Table 3 shows these values and their resulting statistics.

Considering the theoretical properties and the recorded observations, the changes in wind velocity are modeled as a zero-mean noise of equal variance in all directions. The noise value chosen was four knots^2 per second. In the simulation model, however, several different wind models were tested. In one of the models, the wind speeds are noise modulated sinusoids to emulate the wave nature of wind. Using a wave model of wind in the error estimator would have required at least six more states to identify the dominate wave lengths and amplitudes. Given the chaotic nature of wind, the dominant wave number would likely change much faster than the filter would be able to converge, making the addition of the states not worth the extra computational cost.

7.2.3 Euler Angle and Velocity Models

The pitch, bank, and velocity biases and the velocity scale factor are all modeled as constants. Therefore the change in the estimate will be driven only by the noise term.

$$\dot{x} = 0 + \text{noise}$$
$$x(k+1) = x(k) + w$$

The choices of the noise values are rather arbitrary since there is no actual process associated with them. The choices will affect the rate at which the estimate converges to the constant and the volatility of the estimate about the constant. One factor in choosing is that the noise values of these states should be small relative to those of the misalignment and wind states, so that changes in wind and misalignment values are not unduly apportioned into the bias and scale factor states. After a little trial and error, $1x10^{-14}$ was chosen for the pitch and bank biases, $1x10^{-7}$ for the velocity scale factor, and 0.01 for the velocity bias.

7.3 Filter Equations

7.3.1 Discrete Time Kalman Filter

The discrete-time, non-linear filter equations are listed below. The covariance and gain equations result from linearizing the propagation and measurement functions using the first order term of the Taylor series expansion of the functions. This results in the use of the Jacobian of the propagation and measurement functions[2].

State Update	$\underline{x}(k\|k-1) = \underline{f}(\underline{x}(k-1\|k-1),k) + \underline{u}(\underline{x}(k-1\|k-1),k)$
Measurement	$\underline{x}(k\|k) = \underline{x}(k\|k-1) + K(k)[\underline{y}(k) - \underline{h}(k\|k-1)]$
Covariance Update	$P(k\|k\text{-}1) = A(k)\ P(k\text{-}1\|k\text{-}1)\ A^T(k) + G(k)\ Q(k)G^T(k)$
Cov. Measurement	$P(k\|k) = P(k\|k\text{-}1) - K(k)\ H(k)\ P(k\|k\text{-}1)$
Gain Formula	$K(k) = P(k\|k\text{-}1)\ H^T(k)\ [\ H(k)\ P(k\|k\text{-}1)\ H^T(k) + R(k)]^{-1}$

where Q(k) is the covariance matrix of the process noise $\underline{w}$; R(k) is the covariance matrix of the measurement noise $\underline{v}$; A(k) is the Jacobian of $\underline{f}$; and H(k) is the Jacobian of $\underline{h}$.

$$A(k) = \frac{\partial \underline{f}}{\partial \underline{x}} \qquad H(k) = \frac{\partial \underline{h}}{\partial \underline{x}}$$

7.3.2 Propagation Equations

The standard nonlinear discrete time propagation equation is:

$$\underline{x}(k+1) = \underline{f}(\underline{x}(k)) + G(\underline{x}(k))\underline{w}(k) + \underline{u}(\underline{x}(k))$$

There will be no inputs nor transformations of the noise values, so the $\underline{u}$ term and G factor disappear. The state-propagation function $\underline{f}$ is linear in this case, so the propagation equation simplifies to

$$\underline{x}(k+1)=A\underline{x}(k) + \underline{w}(k)$$

where A is populated by the values derived in the Error Estimator Models section.

$$A = \text{diag}(0.99336\ 0.99336\ 0.99336\ 1\ 1\ 1\ 1\ 1\ 1\ 1)$$

The 0.99336 values are for the misalignment states and come from the 150 second time constant. All other states were modeled as changing only by noise so their propagation values are all one.

7.3.3 Measurement Equations

7.3.3.1 Derivation of $\underline{h}$

While the state propagation functions became linear, the functions mapping the states back to the measurements remain highly nonlinear.

$$\underline{y}(k)=\underline{h}(\underline{x}(k)) + \underline{v}(k)$$

where the measurements $\underline{y}$ are the GPS velocities: V_{GN}, V_{GE}, V_{GD}. In this case however, $\underline{h}$ is not a function of just the error states $\underline{x}$, but of the aircraft states also. The function $\underline{h}$ maps the information from the HARS/CADC system into Earth frame velocities, so it is similar to $\underline{V}_H$. In use, the output of the HARS/ CADC velocity estimator will be piped directly into the measurement equations since the correction parameters will be updated each cycle of the low-rate estimator. However, for the purposes of developing the Kalman Filter equations $\underline{h}$ must be written explicitly in terms of the error states.

$$\underline{h} = (I + Z)C_B^E(\psi, \theta + \Delta\theta, \phi + \Delta\phi)\underline{V}_C^B[V_S(V_C - 300) + V_B + 300] + \underline{W}$$

Let $\tilde{C}_B^E$ denote the rotation matrix with $\Delta\theta$ and $\Delta\phi$ as parameters. C_B^E will represent the same matrix without those error corrections. $\tilde{C}_B^E = C_h(\psi)\, C_p(\theta+\Delta\theta)\, C_b(\phi+\Delta\phi)$ where h, p, and b represent heading, pitch, and bank.

$$C_h(\psi) = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$C_p(\theta + \Delta\theta) = \begin{bmatrix} \cos(\theta + \Delta\theta) & 0 & \sin(\theta + \Delta\theta) \\ 0 & 1 & 0 \\ -\sin(\theta + \Delta\theta) & 0 & \cos(\theta + \Delta\theta) \end{bmatrix}$$

$$C_b(\phi + \Delta\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi + \Delta\phi) & -\sin(\phi + \Delta\phi) \\ 0 & \sin(\phi + \Delta\phi) & \cos(\phi + \Delta\phi) \end{bmatrix}$$

$\underline{V}_C^B$ is a vector that transforms the scalar CADC airspeed to body frame coordinates using angle of attack (α) and sideslip (β):

$$\underline{V}_C^B = \begin{bmatrix} 1 \\ -\tan(\beta)/\cos(\alpha) \\ \tan(\alpha) \end{bmatrix}$$

And $[V_S(V_C - 300) + V_B + 300]$ is the adjustment to the CADC supplied velocity V_C using linear corrections about 300 knots.

7.3.3.2 Derivation of $\tilde{C}_B^E$

Recall that $\tilde{C}_B^E = C_h(\psi)\, C_p(\theta+\Delta\theta)\, C_b(\phi+\Delta\phi)$. The objective now will be to simplify the last two factors using small angle approximations and linearize the result by dropping second order terms. Note that c and s will replace cos and sin in the equations.

$$C_p(\theta+\Delta\theta) = \begin{bmatrix} \cos(\theta+\Delta\theta) & 0 & \sin(\theta+\Delta\theta) \\ 0 & 1 & 0 \\ -\sin(\theta+\Delta\theta) & 0 & \cos(\theta+\Delta\theta) \end{bmatrix} \cong \begin{bmatrix} c\theta - \Delta\theta s\theta & 0 & s\theta + \Delta\theta c\theta \\ 0 & 1 & 0 \\ -s\theta - \Delta\theta c\theta & 0 & c\theta - \Delta\theta s\theta \end{bmatrix}$$

$$= \begin{bmatrix} c\theta & 0 & s\theta \\ 0 & 1 & 0 \\ -s\theta & 0 & c\theta \end{bmatrix} + \Delta\theta \begin{bmatrix} -s\theta & 0 & c\theta \\ 0 & 1 & 0 \\ -c\theta & 0 & -s\theta \end{bmatrix} = C_p(\theta) + \Delta\theta \tilde{C}_p$$

$$C_b(\phi+\Delta\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi+\Delta\phi) & -\sin(\phi+\Delta\phi) \\ 0 & \sin(\phi+\Delta\phi) & \cos(\phi+\Delta\phi) \end{bmatrix} \cong \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\phi - \Delta\phi s\phi & -s\phi - \Delta\phi c\phi \\ 0 & s\phi + \Delta\phi c\phi & c\phi - \Delta\phi s\phi \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\phi & -s\phi \\ 0 & s\phi & c\phi \end{bmatrix} + \Delta\phi \begin{bmatrix} 0 & 0 & 0 \\ 0 & -s\phi & -c\phi \\ 0 & c\phi & -s\phi \end{bmatrix} = C_b(\phi) + \Delta\phi \tilde{C}_b$$

$$\tilde{C}_B^E = C_h(\psi)[C_p(\theta) + \Delta\theta\tilde{C}_p][C_b(\phi) + \Delta\phi\tilde{C}_b]$$

$$= C_h(\psi)C_p(\theta)C_b(\phi) + \Delta\theta C_h(\psi)\tilde{C}_p C_b(\phi) + \Delta\phi C_h(\psi)C_p(\theta)\tilde{C}_b + \Delta\theta\Delta\phi C_h(\psi)\tilde{C}_p\tilde{C}_b$$

Dropping the second order term at the end and leaving out the parameter names:

$$\tilde{C}_B^E = C_B^E + \Delta\theta C_h \tilde{C}_p C_b + \Delta\phi C_h C_p \tilde{C}_b$$

$$C_h\tilde{C}_pC_b = \begin{bmatrix} c\psi & -s\psi & 0 \\ s\psi & c\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} -s\theta & 0 & c\theta \\ 0 & 0 & 0 \\ -c\theta & 0 & -s\theta \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 \\ 0 & c\phi & -s\phi \\ 0 & s\phi & c\phi \end{bmatrix} = \begin{bmatrix} -c\psi s\theta & c\psi c\theta s\phi & c\psi c\theta c\phi \\ -s\psi s\theta & s\psi c\theta s\phi & s\psi c\theta c\phi \\ -c\theta & -s\theta s\phi & -s\theta c\phi \end{bmatrix}$$

$$C_hC_p\tilde{C}_b = \begin{bmatrix} c\psi & -s\psi & 0 \\ s\psi & c\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} c\theta & 0 & s\theta \\ 0 & 1 & 0 \\ -s\theta & 0 & c\theta \end{bmatrix}\begin{bmatrix} 0 & 0 & 0 \\ 0 & -s\phi & -c\phi \\ 0 & c\phi & -s\phi \end{bmatrix} = \begin{bmatrix} 0 & c\psi s\theta c\phi + s\psi s\phi & -c\psi s\theta s\phi + s\psi c\phi \\ 0 & s\psi s\theta c\phi - c\psi s\phi & -s\psi s\theta s\phi - c\psi c\phi \\ 0 & c\theta c\phi & -c\theta s\phi \end{bmatrix}$$

7.3.3.3 Combining Z with $\tilde{C}_B^E$

Replacing $\tilde{C}_B^E$ with the results from above gives a new form of $\underline{h}$. Multiplying through by (I + Z) and discarding second order terms (like $Z\Delta\theta$) will give the final form of $\underline{h}$.

$$\begin{aligned}\underline{h} &= (I+Z)[C_B^E + \Delta\theta C_h\tilde{C}_pC_b + \Delta\phi C_hC_p\tilde{C}_b]\underline{V}_C^B[V_S(V_C-300)+V_B+300]+\underline{W} \\ &= [C_B^E + ZC_B^E + \Delta\theta C_h\tilde{C}_pC_b + \Delta\phi C_hC_p\tilde{C}_b]\underline{V}_C^B[V_S(V_C-300)+V_B+300]+\underline{W}\end{aligned}$$

7.3.4 Partial Derivatives of h(x(t),t)

Given $\underline{h}$ as written above the next task is to derive the Jacobian of the measurement equation which is used in the covariance update equations. To simplify notation, let $V^* = V_S(V_C - 300) + V_B + 300$ and let c, s, and t replace cos, sin, and tan in the derivations.

7.3.4.1 Sensitivity to North misalignments

$$\frac{\partial\underline{h}}{\partial\zeta_N} = \frac{\partial Z}{\partial\zeta_N}C_B^E\underline{V}_C^BV^*$$

$$= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} c\psi c\theta & c\psi s\theta s\phi - s\psi c\phi & c\psi s\theta c\phi + s\psi s\phi \\ s\psi c\theta & s\psi s\theta s\phi + c\psi c\phi & s\psi s\theta c\phi - c\psi s\phi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix}\begin{bmatrix} 1 \\ -t\beta / c\alpha \\ t\alpha \end{bmatrix}V^*$$

$$\frac{\partial h_1}{\partial \zeta_N} = 0$$

$$\frac{\partial h_2}{\partial \zeta_N} = [s\theta + c\theta s\phi\ t\beta / c\alpha - c\theta c\phi t\alpha]V^*$$

$$\frac{\partial h_3}{\partial \zeta_N} = [s\psi c\theta - (s\psi s\theta s\phi + c\psi c\phi)t\beta / c\alpha + (s\psi s\theta c\phi - c\psi s\phi)t\alpha]V^*$$

7.3.4.2 Sensitivity to East misalignments

$$\frac{\partial \underline{h}}{\partial \zeta_E} = \frac{\partial Z}{\partial \zeta_E} C_B^E \underline{V}_C^B V^*$$

$$= \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix} \begin{bmatrix} c\psi c\theta & c\psi s\theta s\phi - s\psi c\phi & c\psi s\theta c\phi + s\psi s\phi \\ s\psi c\theta & s\psi s\theta s\phi + c\psi c\phi & s\psi s\theta c\phi - c\psi s\phi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix} \begin{bmatrix} 1 \\ -t\beta / c\alpha \\ t\alpha \end{bmatrix} V^*$$

$$\frac{\partial h_1}{\partial \zeta_E} = [-s\theta - c\theta s\phi\ t\beta / c\alpha + c\theta c\phi t\alpha]V^*$$

$$\frac{\partial h_2}{\partial \zeta_E} = 0$$

$$\frac{\partial h_3}{\partial \zeta_E} = [-c\psi c\theta + (c\psi s\theta s\phi - s\psi c\phi)t\beta / c\alpha - (c\psi s\theta c\phi + s\psi s\phi)t\alpha]V^*$$

7.3.4.3 Sensitivity to down misalignments

$$\frac{\partial \underline{h}}{\partial \zeta_D} = \frac{\partial Z}{\partial \zeta_D} C_B^E \underline{V}_C^B V^*$$

$$= \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} c\psi c\theta & c\psi s\theta s\phi - s\psi c\phi & c\psi s\theta c\phi + s\psi s\phi \\ s\psi c\theta & s\psi s\theta s\phi + c\psi c\phi & s\psi s\theta c\phi - c\psi s\phi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix} \begin{bmatrix} 1 \\ -t\beta / c\alpha \\ t\alpha \end{bmatrix} V^*$$

$$\frac{\partial h_1}{\partial \zeta_D} = [-s\psi c\theta + (s\psi s\theta s\phi + c\psi c\phi)t\beta / c\alpha + (-s\psi s\theta c\phi + c\psi s\phi)t\alpha]V^*$$

$$\frac{\partial h_2}{\partial \zeta_D} = [c\psi c\theta - (c\psi s\theta s\phi - s\psi c\phi)t\beta / c\alpha + (c\psi s\theta c\phi + s\psi s\phi)t\alpha]V^*$$

$$\frac{\partial h_3}{\partial \zeta_D} = 0$$

7.3.4.4 Sensitivity to pitch bias

$$\frac{\partial \underline{h}}{\partial \Delta\theta} = C_h \tilde{C}_p C_b \underline{V}_C^B V^*$$

$$\frac{\partial h_1}{\partial \Delta\theta} = [-c\psi s\theta - c\psi c\theta s\phi\ t\beta / c\alpha + c\psi c\theta c\phi\ t\alpha]V^*$$
$$\frac{\partial h_2}{\partial \Delta\theta} = [-s\psi s\theta - s\psi c\theta s\phi\ t\beta / c\alpha + s\psi c\theta c\phi\ t\alpha]V^*$$
$$\frac{\partial h_3}{\partial \Delta\theta} = [-c\theta + s\theta s\phi\ t\beta / c\alpha - s\theta c\phi\ t\alpha]V^*$$

7.3.4.5 Sensitivity to bank bias

$$\frac{\partial \underline{h}}{\partial \Delta\phi} = C_h C_p \tilde{C}_b \underline{V}_C^B [V_S(V_C - 300) + V_B]$$

$$\frac{\partial h_1}{\partial \Delta\phi} = [-(c\psi s\theta c\phi + s\psi s\phi)t\beta / c\alpha + (-c\psi s\theta s\phi + s\psi c\phi)\ t\alpha]V^*$$
$$\frac{\partial h_2}{\partial \Delta\phi} = [(-s\psi s\theta c\phi + c\psi s\phi)t\beta / c\alpha + (-s\psi s\theta s\phi - c\psi c\phi)\ t\alpha]V^*$$
$$\frac{\partial h_3}{\partial \Delta\phi} = [-c\theta c\phi\ t\beta / c\alpha - c\theta s\phi\ t\alpha]V^*$$

7.3.4.6 Sensitivity to velocity scale factor

$$\frac{\partial \underline{h}}{\partial V_S} = [C_B^E + ZC_B^E + \Delta\theta C_h \tilde{C}_p C_b + \Delta\phi C_h C_p \tilde{C}_b]\underline{V}_C^B(V_C - 300)$$

Note that in the construction of the filter θ and ϕ are updated each cycle by $\Delta\theta$ and $\Delta\phi$. Therefore, for the purposes of deriving the velocity sensitivity, we can absorb the $\Delta\theta$ and $\Delta\phi$ terms into θ and ϕ inside C_B^E and simplify the equation.

$$\frac{\partial \underline{h}}{\partial V_S} = [C_B^E + ZC_B^E]\underline{V}_C^B(V_C - 300)$$

$$\frac{\partial \underline{h}}{\partial V_S} = \begin{bmatrix} 1 & -\zeta_D & \zeta_E \\ \zeta_D & 1 & -\zeta_N \\ -\zeta_E & \zeta_N & 1 \end{bmatrix} \begin{bmatrix} c\psi c\theta & c\psi s\theta s\phi - s\psi c\phi & c\psi s\theta c\phi + s\psi s\phi \\ s\psi c\theta & s\psi s\theta s\phi + c\psi c\phi & s\psi s\theta c\phi - c\psi s\phi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix} \begin{bmatrix} 1 \\ -t\beta / c\alpha \\ t\alpha \end{bmatrix} V^*$$

$$\frac{\partial h_1}{\partial V_S} = \begin{bmatrix} (c\psi c\theta - \zeta_D s\psi c\theta - \zeta_E s\theta) - (c\psi s\theta s\phi - s\psi c\phi - \zeta_D (s\psi s\theta s\phi + c\psi c\phi) + \zeta_E c\theta s\phi) t\beta / c\alpha \\ +(c\psi s\theta c\phi + s\psi s\phi - \zeta_D (s\psi s\theta c\phi - c\psi s\phi) + \zeta_E c\theta c\phi) t\alpha \end{bmatrix}$$
$$*[V_C - 300]$$

$$\frac{\partial h_2}{\partial V_S} = \begin{bmatrix} (s\psi c\theta + \zeta_D c\psi c\theta + \zeta_N s\theta) - (s\psi s\theta s\phi + c\psi c\phi + \zeta_D (c\psi s\theta s\phi - s\psi c\phi) - \zeta_N c\theta s\phi) t\beta / c\alpha \\ +(s\psi s\theta c\phi - c\psi s\phi + \zeta_D (c\psi s\theta c\phi + s\psi s\phi) - \zeta_N c\theta c\phi) t\alpha \end{bmatrix}$$
$$*[V_C - 300]$$

$$\frac{\partial h_3}{\partial V_S} = \begin{bmatrix} (-s\theta - \zeta_E c\psi c\theta + \zeta_N s\psi c\theta) - (c\theta s\phi - \zeta_E (c\psi s\theta s\phi - s\psi c\phi) + \zeta_N (s\psi s\theta s\phi + c\psi c\phi)) t\beta / c\alpha \\ +(c\theta c\phi - \zeta_E (c\psi s\theta c\phi + s\psi s\phi) + \zeta_N (s\psi s\theta c\phi - c\psi s\phi)) t\alpha \end{bmatrix}$$
$$*[V_C - 300]$$

7.3.4.7 Sensitivity to velocity bias

$\frac{\partial \underline{h}}{\partial V_B}$ is the same as $\frac{\partial \underline{h}}{\partial V_S}$ without $[V_C\text{-}300]$

7.3.4.8 Sensitivity to wind

$$\frac{\partial \underline{h}}{\partial W} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

H(k) is then the matrix containing the values derived above.

$$H(k) = \begin{bmatrix} \frac{\partial h_1}{\partial \zeta_N} & \frac{\partial h_1}{\partial \zeta_E} & \frac{\partial h_1}{\partial \zeta_D} & \frac{\partial h_1}{\partial \Delta\theta} & \frac{\partial h_1}{\partial \Delta\phi} & \frac{\partial h_1}{\partial V_S} & \frac{\partial h_1}{\partial V_B} & \frac{\partial h_1}{\partial W_N} & \frac{\partial h_1}{\partial W_E} & \frac{\partial h_1}{\partial W_D} \\ \frac{\partial h_2}{\partial \zeta_N} & \frac{\partial h_2}{\partial \zeta_E} & \frac{\partial h_2}{\partial \zeta_D} & \frac{\partial h_2}{\partial \Delta\theta} & \frac{\partial h_2}{\partial \Delta\phi} & \frac{\partial h_2}{\partial V_S} & \frac{\partial h_2}{\partial V_B} & \frac{\partial h_2}{\partial W_N} & \frac{\partial h_2}{\partial W_E} & \frac{\partial h_2}{\partial W_D} \\ \frac{\partial h_3}{\partial \zeta_N} & \frac{\partial h_3}{\partial \zeta_E} & \frac{\partial h_3}{\partial \zeta_D} & \frac{\partial h_3}{\partial \Delta\theta} & \frac{\partial h_3}{\partial \Delta\phi} & \frac{\partial h_3}{\partial V_S} & \frac{\partial h_3}{\partial V_B} & \frac{\partial h_3}{\partial W_N} & \frac{\partial h_3}{\partial W_E} & \frac{\partial h_3}{\partial W_D} \end{bmatrix}$$

8. Error Estimator Testing

8.1 Testing Resources and Environment

8.1.1 Simulation

Eighteen primary, custom, Matlab routines were written and used (Table 4), along with several small secondary routines, in the simulation, estimation, and analysis portions of this test. The main block diagram is shown in figure 10.

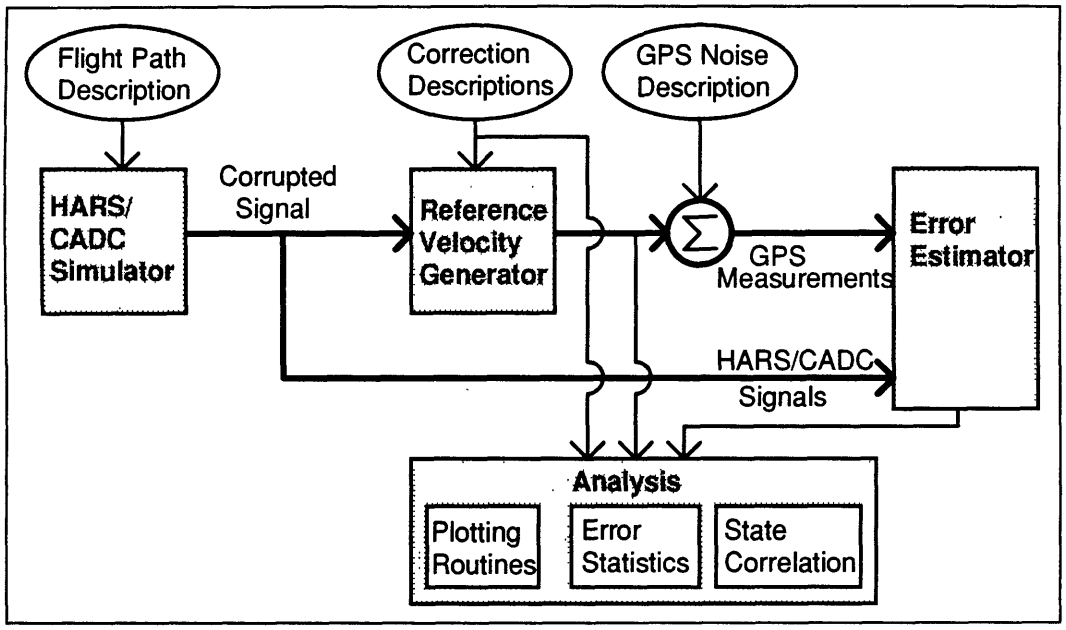

Figure 10 Testing Environment

Table 4 Description of Matlab Routines

Routine	Purpose
hcref	Creates a trajectory based on a general description input and calculates heading, pitch, bank, angle of attack and airspeed.
cross3	Performs cross product of 3x1 vectors. Used in hcref.
mag3	Finds magnitude of 3x1 vectors. Used in hcref.
gpsvel	Creates reference velocity in Earth-surface frame.
evcorr	Corrects Euler angles and airspeeds. Used in gpsvel and errest.
velhars	Calculates body frame velocities. Used in gpsvel and errest.
zwcorr	Performs misalignment corrections and adds wind. Used in gpsvel and errest.
errest	Calculates estimates of correction parameters.
calch	Calculates Jacobian of measurement function. Used in errest.
varplot	Extracts the diagonal elements of the covariance matrix from the Kalman filter and stores in a time based vector for later analysis.
plotcorr	Draws plots of the state estimates versus the true values.
ploterr	Actually three separate routines that draw plots of the error in state estimates versus the square root of the filter covariances.
plotres	Draws plots of the measurement residuals.

statcorr	Calculates and plots correlations between states.
showrms	Calculates the root-mean-square error in the state estimates, measurement residual and reference velocity residual.
phed	Draws plot of flight path versus time.

The simulation begins with a flight path description which consists of time based vectors of G-load, roll rate, sideslip, and thrust-generated acceleration. The HARS/CADC simulator, hcref, propagates initial velocities into a trajectory by applying parameters from the flight path description once each second and storing the results as heading, pitch, bank, airspeed, and AOA. The simulator considers the effects of magnetic variation, AOA and sideslip on heading, G-load and bank on pitch rate and heading rate, pitch angle on bank, G-load on AOA, AOA and sideslip on airspeed measurement, and gravity on velocity in climbs and dives. For the purposes of this simulation, the output of the HARS/CADC simulator is considered to be the error corrupted values just as those from the actual systems would be.

The reference, or true velocities, are created by applying a correction description to the HARS/CADC signals. The objective of the error estimator is to match the values set in the correction description. The reference velocity generator (Figure 11) first adds pitch, bank and magnetic variation corrections to the HARS Euler angles, then corrects CADC airspeed for scale factor and bias. Next, it converts airspeed into body frame velocities accounting for AOA and sideslip. Finally, it rotates the velocities to correct for misalignments and adds winds.

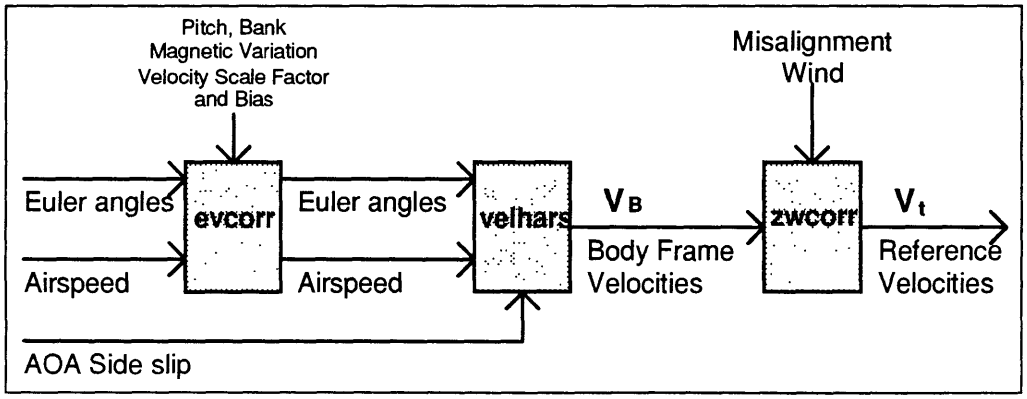

Figure 11 Reference Velocity Generator.

The next step is to generate the GPS velocity signals that are used as the measurements by the error estimator. This is done by simply adding to the reference velocity a zero-mean white noise of variance equal to that of the actual GPS signal.

The error estimator reuses the same error correction routines as the reference velocity generator, but substitutes its estimates of the correction parameters (Figure 12). The output of this process is now called the HARS velocities V_h. These velocities are piped into *calch* which calculates the Jacobian of the measurement function as derived in the last chapter, and into the Kalman filter itself. The state estimates, covariance values, gains, measurement Jacobians, and output velocities are stored for analysis.

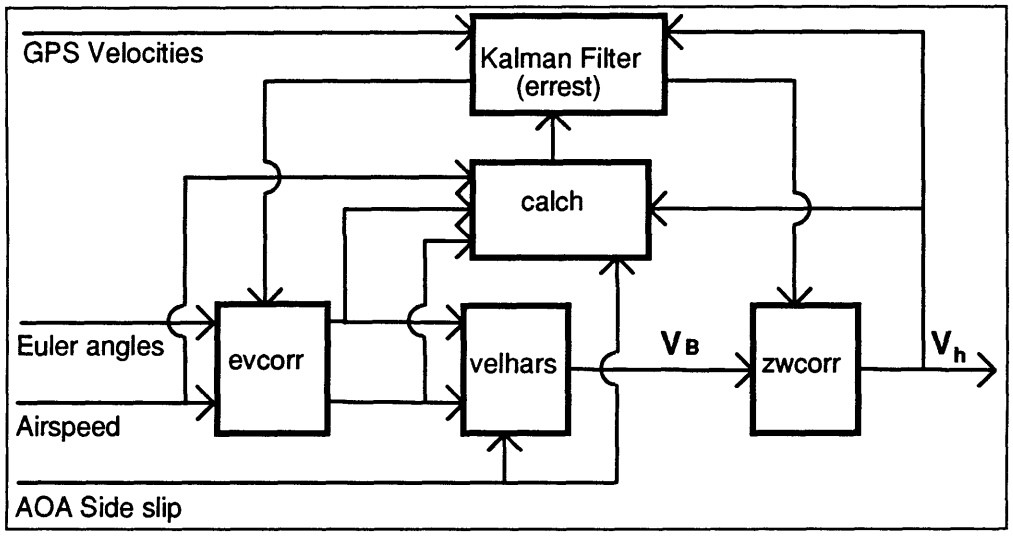

Figure 12 Error Estimator

8.1.2 Analysis

8.1.2.1 State Estimate Plots

State Estimate plots show the progress of the estimate as a function of time (Figure 13). They also overlay the reference values to which the estimates should converge. The x-axis shows the time index in seconds. The y-axis

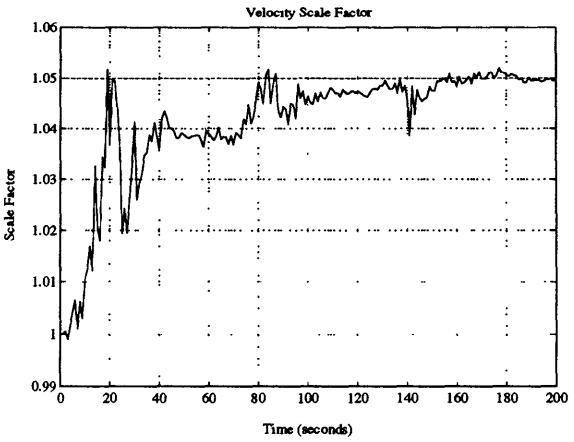

Figure 13 Example of State Estimate Plot

shows the state value. Euler angles and misalignment angles are shown in radians; velocity bias and winds are in knots; and velocity scale factor is unitless.

8.1.2.2 Estimation Error Plots

These plots (Figure 14) show the difference between the state estimate and reference values (solid lines). Overlaid are the plus and minus standard deviations (dashed lines) extracted by taking the square root of the diagonal entries of the error covariance matrix. Ideally, the state estimates should fill the space between the standard deviation lines with only a small amount of spill over. The axes are defined in the same way as the state estimation plots.

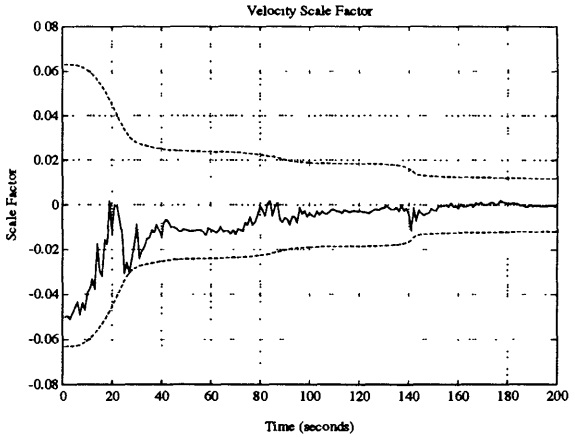

Figure 14 Example of Estimation Error Plot

8.1.2.3 Measurement Residual Plots

These plots (Figure 15) show the difference between the output velocities at the end of each time step and the measurement (GPS) velocities at the new time step. Overlaid are the lines of the standard deviation of the measurement residual:

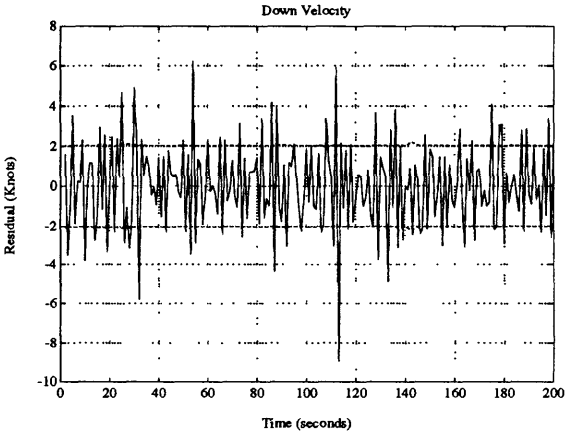

Figure 15 Example of Measurement Residual Plot

$$\sqrt{HPH^T + R}$$

H = Measurement Jacobian

P = State covariance matrix

R = Process noise matrix

8.1.2.4 State Correlation Plots

These plots (Figure 16) depict the individual correlations of all of the state estimates to one chosen state. Values close to positive or negative one show a high degree of correlation. Values near zero show little correlation. The correlations are calculated from entries of the state covariance matrix P.

$$C_{ij} = \frac{P_{ij}}{\sqrt{P_{ii} P_{jj}}}$$

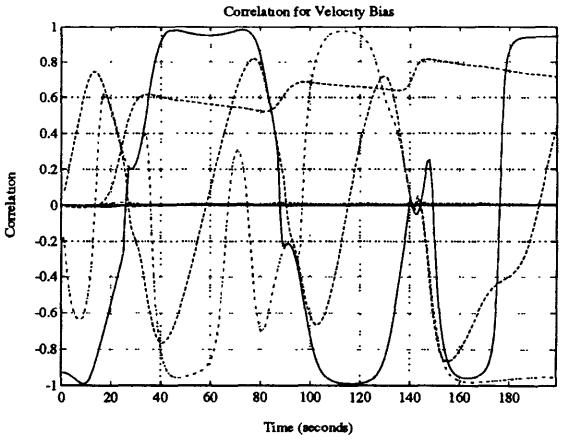

Figure 16 Example of State Correlation Plot

8.1.2.5 Root Mean Square Error

The final tool in analyzing the results is the RMS error. Comparative quality is often difficult to discern from the plots described above. The RMS calculation routine, *showrms*, boils performance down to single numbers for each of the states and the measurement residual. Since the outputs of the low-rate filter section are the corrected HARS/CADC velocities, the goal is to minimize the measurement residual.

8.2 Functional Testing

Initial testing was designed to validate the derivations and software implementation. States in the filter were isolated by zeroing out blocks of the measurement Jacobian, H, and limiting the simulated errors to the states of interest. Through this technique the complex interactions between the states were suppressed. Testing the states two or three at a time allowed them to converge rapidly and precisely to the correct solution. The derivations were shown to be correct as stated, and after a little debugging the software was functionally validated also.

8.3 Robustness and Sensitivity testing

8.3.1 Overview

Early testing revealed a great interdependence among the states in the filter. Changing noise or model parameters of one state in either the filter or the simulation affected estimates of all of the states. Also, the effect on the state was rarely that which was intended. Changes of wind parameters apparently had the greatest effect--followed, not too closely, by changes in misalignment noise values. The purpose of this part of the testing was to evaluate the filter robustness to changes in the simulation's wind model and to investigate the estimates' sensitivities to filter noise values.

8.3.2 Wind Models

The filter was evaluated against four wind models used in the simulation routines: Markov 1, Markov 2, Sinusoidal, and Ramp.

8.3.2.1 Markov Model 1 and 2

The three wind directions were modeled as constants to which a Markov process was added with the intent to achieve a five knot standard deviation. The North, East, and down winds were centered on 10 ,-10 and 0 respectively. The noise used in the filter had a standard deviation of 2 knots, so this was also used in the first Markov model. The appropriate gain was calculated using the steady-state discrete-time Riccati equation.

$$P = APA^T+Q$$

Substituting 5^2 for P and 2^2 for Q gives the gain of 0.9165. The propagation equation for the winds in the simulator became:

$$W(k+1) = .9165(W(k) - \text{initial value}) + \text{initial value} + \text{rand\#} * 2$$

where the rand# gives a normally distributed number with zero mean and unit variance.

The volatility of the wind made it difficult for the filter to track changes in the misalignment states. To see if the misalignment estimates could track the true values better under more benign but still realistic winds, Markov model 1 was modified for lower volatility. The Markov process was still set up to achieve the same steady state variance,

but to use a noise value of 1 instead of 2 knots. Recalculating the gain gave a value of 0.9789.

8.3.2.2 Sinusoidal Wind Model

This model was designed to emulate the wave nature of wind such as may occur in mountainous terrain, near weather fronts, and over unevenly heated terrain. The three components of wind were modeled as separate sinusoids with different periods averaging 240 seconds with an amplitude ±20 knots. The North wind had the fastest period, 160 seconds; East wind had 240; and down wind had 320. A zero-mean normally distributed noise of 1.5 knot standard deviation was added to each wind sample.

8.3.2.3 Ramp Wind Model

This model assumes the aircraft is passing through a steadily changing wind shear from 30 knots down to zero. In order to challenge the filter, the slope of the wind velocity changes was set to 3, 2, and 1 knot per second respectively. Since this would create at most a thirty point simulation, the concept was extended into a triangle wave oscillating between ±30 knots extending to the full 2000 point simulations used with all the other models.

8.3.3 Results of Wind Model Tests

A "set" of runs for a wind model consisted of 12 separate 2000 point simulations each using a different value for the wind noise variance ranging from 100 knots^2 down to 0.1 knots^2. Multiple runs using the same set of initial conditions gave different outputs because of the random number generators used to add noise. To remove random variation between the runs within a set, a single set of trajectory values and input signals was created. This same input data set was used to run the filter at each of the different values of Q. To reduce dependencies on the data set, the whole process was repeated three times. The trajectory description remained the same; but the random number generators used throughout the simulation produced new sets of noises. Misalignment estimation

error, wind estimation error, and measurement residuals were calculated for each run using RMS averaging. Figure 17 shows the results of three sets of runs using the Markov wind model 1. Note that the value of Q that minimizes the wind estimation error is different for each of the three runs. The Q that minimizes the measurement residual and misalignment errors varies also; but they are more difficult to see on the graph because the changes are occurring several decimal places deep.

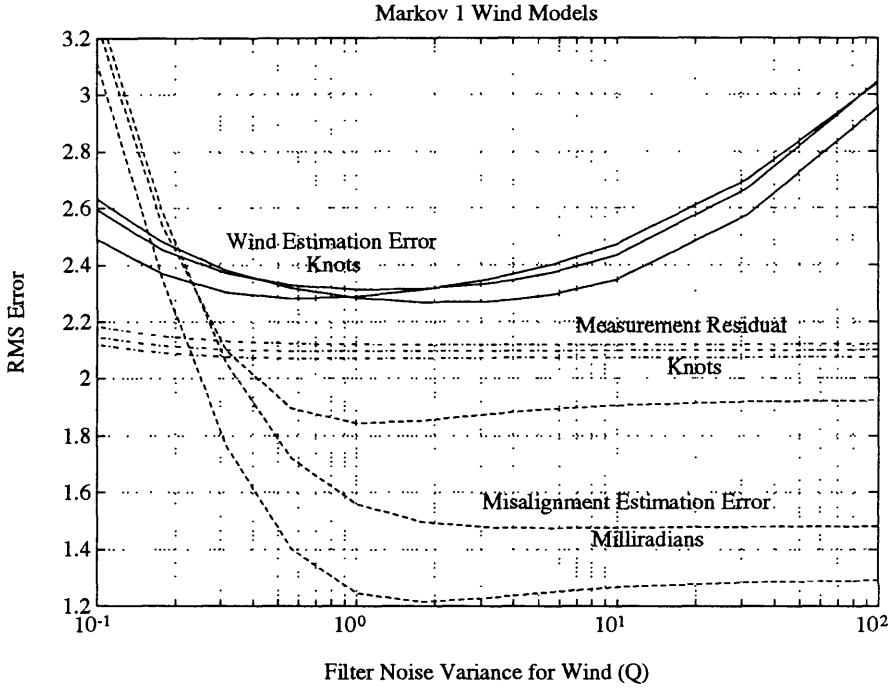

Figure 17 Multiple sets of robustness data for Markov 1

Finally, the RMS errors obtained from each set of runs were averaged using RMS to give a single value for each type of error at each value of Q for each wind model. A subset of the data is shown in table 5 to clarify. The misalignment errors are shown in milliradians, the wind and residual values in knots.

Table 5 Subset of Robustness Testing Data

	Markov Model 2			Sinusoidal Model		
Q	Misalign	Wind	Residual	Misalign	Wind	Residual
100	1.7023	2.3827	1.0904	1.8310	2.8862	2.2401
31.62	1.7007	1.8924	1.0871	1.8282	2.5647	2.2379
10	1.6959	1.5567	1.0855	1.8204	2.4484	2.2341
5.62	1.6911	1.4540	1.0851	1.8137	2.4618	2.2299

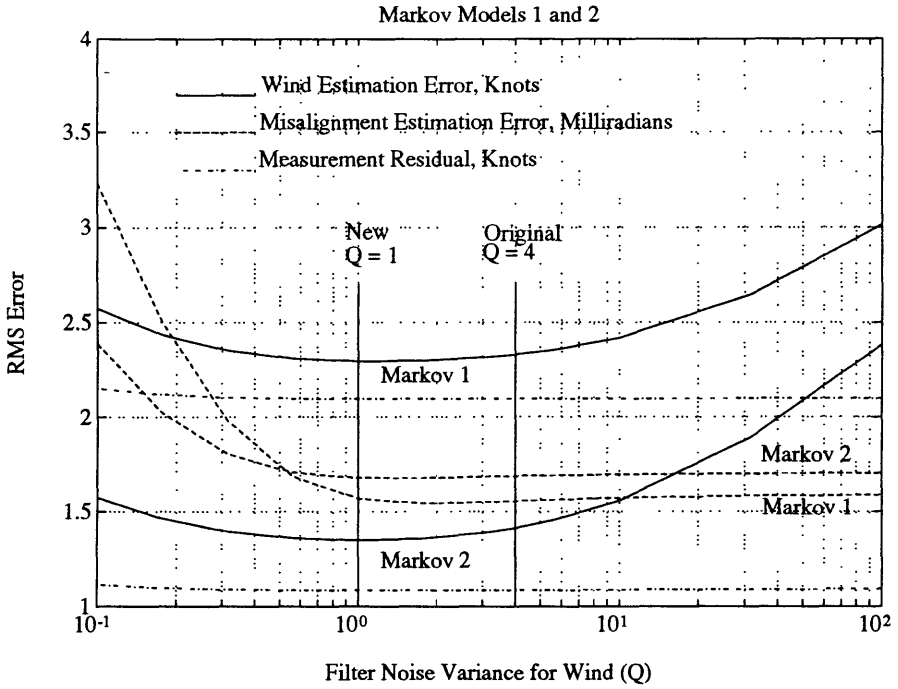

Figure 18 Filter sensitivity for Markov 1 and 2 wind models

Figure 18 shows a plot of the errors versus the wind noise variance, Q, of the filter for the Markov wind models 1 and 2. The two vertical lines merely mark the nominal value of Q=4 set in the design, and the new value chosen for the filter based on this and subsequent tests. From a macroscopic view, the winds simulated by Markov 1 and 2 had similar volatility; the wind speeds were bounded in the same ranges by the Markov models. But the lower internal noise value used in Markov 2 gave the filter a much greater ability to estimate the winds and lower the measurement residual. Curiously, the misalignment estimation errors were very slightly worse for the Markov 2 model over most of the range of Q. This might be due to the fact that the higher wind volatility in the Markov 1 model gave the aircraft more accelleration inputs and hence better observability of the misalignment. For both models, the filter shows little sensitivity to changes in Q; however, the changes that do occur favor a lower noise value.

Figure 19 shows a comparison between the sinusoidal and ramp wind models. The scales of the axes are the same as the Markov-models plot, so direct comparisons can be made to those models also. The standard deviation of the wind velocity changes at each

time step was very close to that of the Markov 1 model for both the sinusoidal and ramp models. As a result the measurement residuals were slightly higher, but very comparable

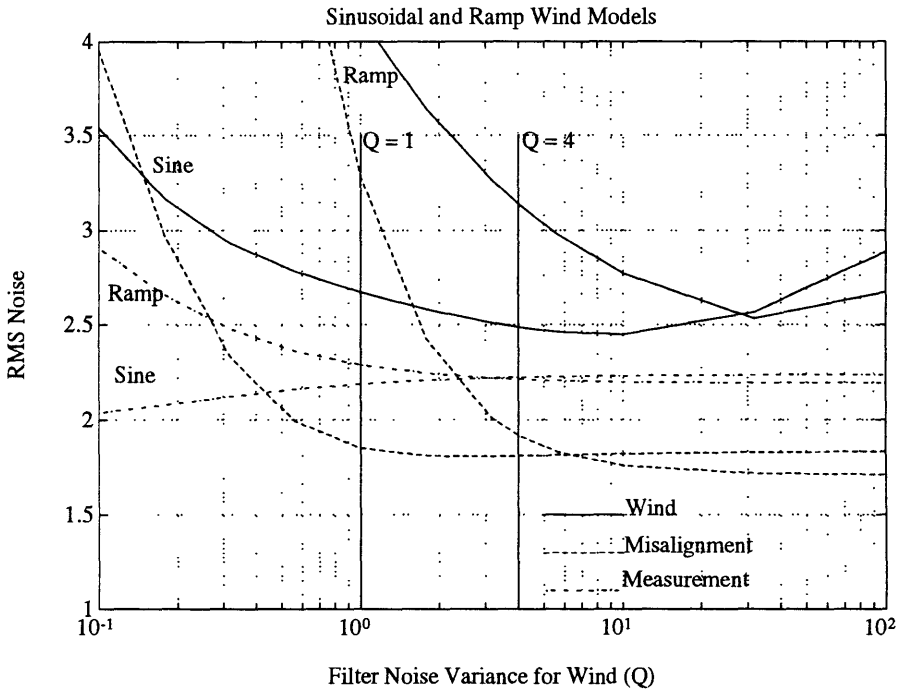

Figure 19 Filter sensitivity to sinusoidal and ramp models

to those of the Markov 1 model until Q dropped below 1, then the residuals diverged. The ramp model contained the worst case where one of the components had a continuous three-knot velocity change each second. Other models assume a low frequency component of wind that changes slowly relative to the noise value and a high frequency component that brings up the total volatility to around the nominal noise value. The high-rate of change of velocity occurring completely in the low frequency component (there was no high frequency noise added) gave the ramp model the worst response curves for wind and misalignment estimation errors. At the nominal value of Q, the wind estimation error for the ramp model was 35% worse than for Markov 1, that of the sinusoidal model was 6.7% worse.

8.3.4 Sensitivity to Misalignment Noise

This test was performed in a manner very similar to that of the wind model testing described above. Three sets of twelve simulation runs were made while varying the filter's

misalignment noise value from 10^{-6} to 10^{-9}. The Markov 2 model was used for the wind simulation. The results of the three sets were averaged together using RMS and are plotted in figure 20. The measurement residual is flat across the whole graph, while the misalignment error begins to climb rapidly above Q=10^{-7}. The nominal value Q=4.03e-8 is very close to the value which minimizes the misalignment estimation error and will be adequate for the rest of the testing.

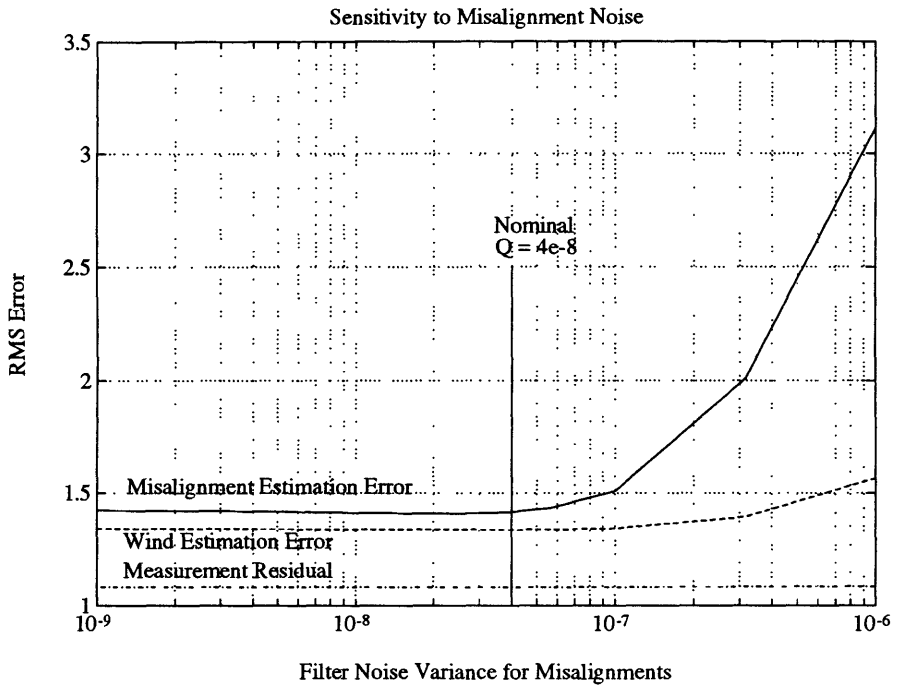

Figure 20 Filter sensitivity to misalignment noise.

8.4 Performance Assessment

8.4.1 Overview

The overall performance assessment was done by running a 2000 second simulation using the filter noise values determined during the sensitivity testing. The wind noise variance was set at one knot^2 and the misalignment noise variance was 4.03e-8 radians^2. The wind model was the Markov 2 version. The flight path was designed to give high observabilitiy to all the states by varying heading, pitch, bank, and airspeed as

much as possible. Selected graphs will be used to illustrate the results. Complete graphs of the performance run can be found in Appendix B.

8.4.2 Misalignment Estimation

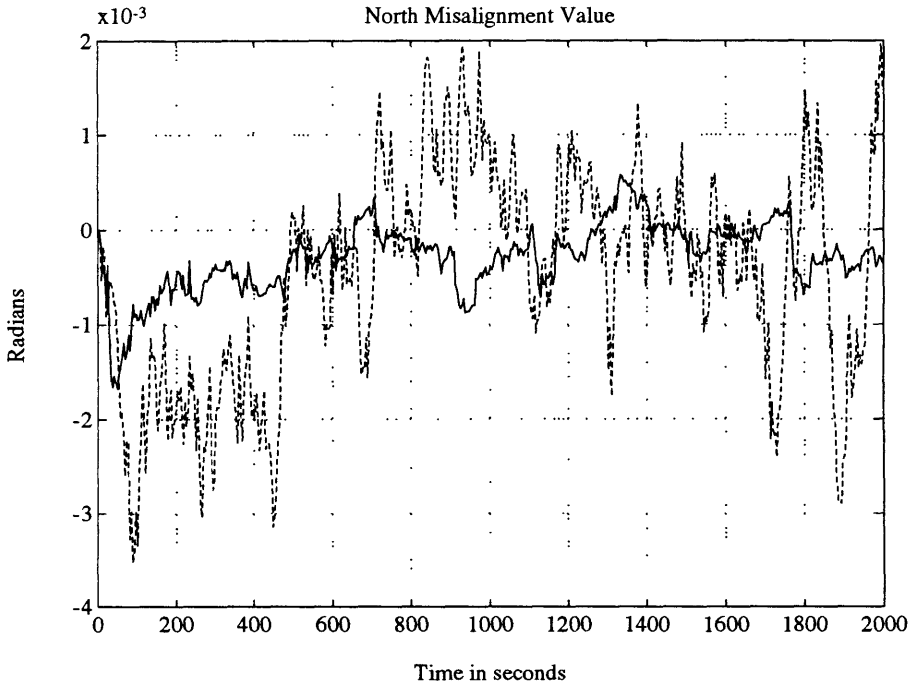

Figure 21 Estimate of North Misalignment

Figure 21 shows the estimate of the North misalignment overlaid on the simulated value. The solid line is the estimate. The graph shows that the filter cannot track the sharp swings in the misalignment values, but can estimate general trends. Increasing the variance of noise values in the filter or lengthening the filter time constant sometimes made the graphs look better to the naked eye, but invariably the RMS errors increased. Figure 22 shows the error in the North misalignment estimate overlaid on the square root of the estimated variance for that state. The graph shows the estimation errors to be consistent with the calculated standard deviations. The estimation errors fill the space between the standard deviation lines only occasionally and slightly exceeding the space. The North misalignment estimates had an RMS error of 1.0931 milliradians; East had 1.2758; and down had 1.2360. The total RMS error was 1.2042 milliradians. Based on observations

of many simulations, the differences seen in the errors among the three directions appear to be due more to random variations than to any systemic cause.

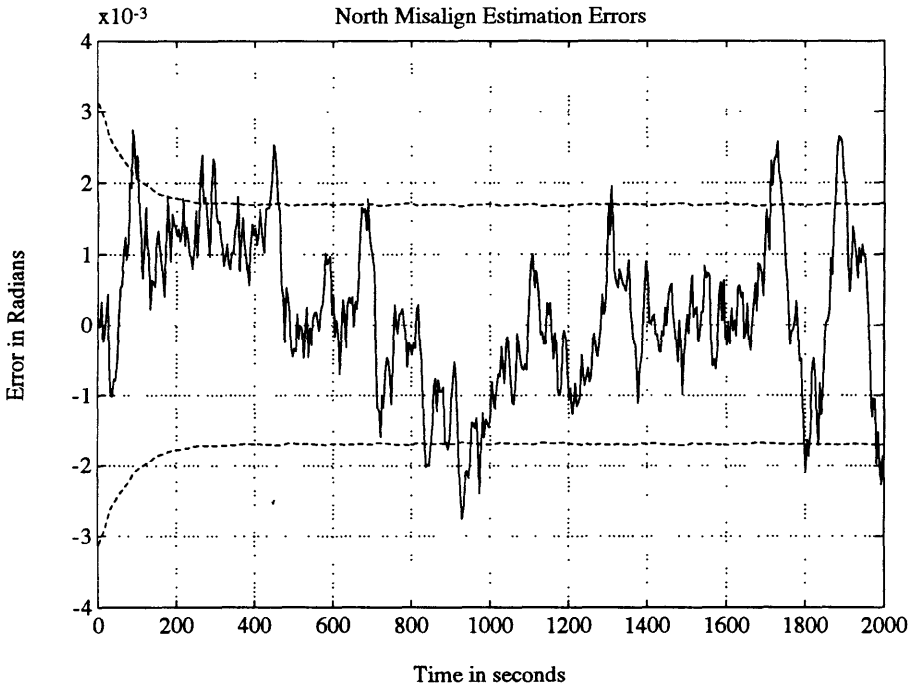

Figure 22 Error in North Misalignment Estimate

8.4.3 Pitch and Bank Corrections

The pitch correction estimates had an RMS error of 0.80935 milliradians. Figure 23 shows a plot of the error in the pitch correction estimate overlaid on the estimated standard deviation. The filter used a process noise value of 10^{-14}. The plot would seem to indicate that the filter could use a smaller noise value to help close down the standard deviation lines. The noise value

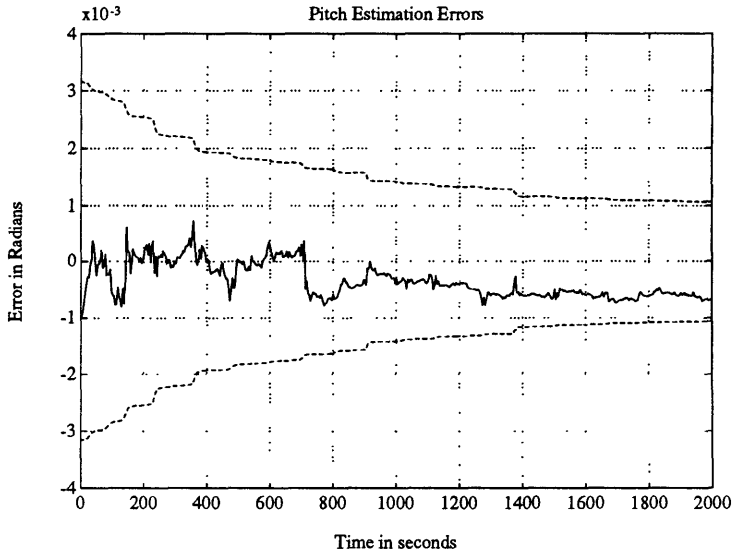

Figure 23 Errors in Estimate of Pitch Correction

was decreased as far as 10^{-40} in this attempt, but had no discernible effect. The only thing that allowed the standard deviations to close down faster was starting with a lower initial variance. The changes in the noise value likely do have an effect in the steady state; but the filter converges so slowly that it cannot be detected in a 2000 point simulation.

Another issue with the pitch correction estimate is that it is moderately correlated with the estimate of the down wind velocity. The filter therefore has difficulty distinguishing between the effects of pitch error and vertical wind velocity. This likely contributes to the observed high frequency volatility and the slow drift away from the correct value observed after the 700 second point. When test runs are made with lower wind noise (both in simulation and in the filter), the pitch correction errors become much closer to zero mean.

Figure 24 shows the estimation errors in the bank correction value. This state suffers greatly from observability problems. When wind and misalignment noise values are significantly decreased, the filter variances begin dropping for this state, and the estimate begins converging to the correct value. But even then, the convergence is very slow. This problem is likely due to low observability of the effects of bank error. Bank is used to translate body frame velocities v_y and v_z into Earth-surface frame velocities. However, v_y and v_z are generated by AOA and sideslip and are typically very small relative to v_x. The AOA rarely surpasses 10 degrees and usually hovers around 2 degrees. Sideslip rarely strays from zero. Therefore bank angle measurement errors on the order of milliradians

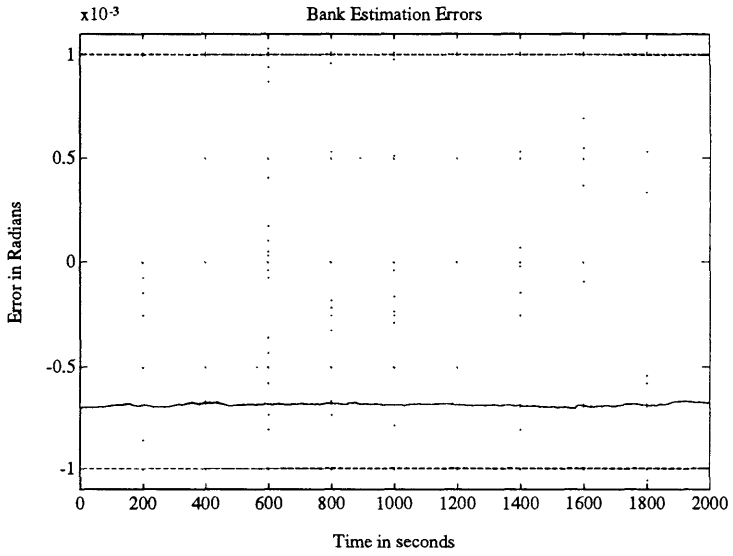

Figure 24 Errors in Estimation of Bank Corrections

would have very small effects on the Earth-surface frame velocities. It may be worthwhile to drop bank corrections as a state and simplify the measurement functions.

8.4.4 Velocity Corrections

Figure 25 shows the error in the estimate of the velocity scale-factor correction. The filter converges quickly and seems to have well-balanced noise values. Figure 26 shows the estimation errors for the velocity bias. The bias term converges a little less quickly than the scale factor, but still does a good job of identifying the error. This difference in behavior is probably due to the differing correlation to wind estimates. The scale factor does not show consistent correlation to any of the other states. The correlation of velocity bias to the various wind directions shifted rapidly between highly positive and highly negative. The overall correlation with any one wind direction varies as the aircraft turns; but there is almost always one or more wind directions that are highly correlated with the velocity bias. The high correlation makes it difficult for the filter to distribute measurement residuals among

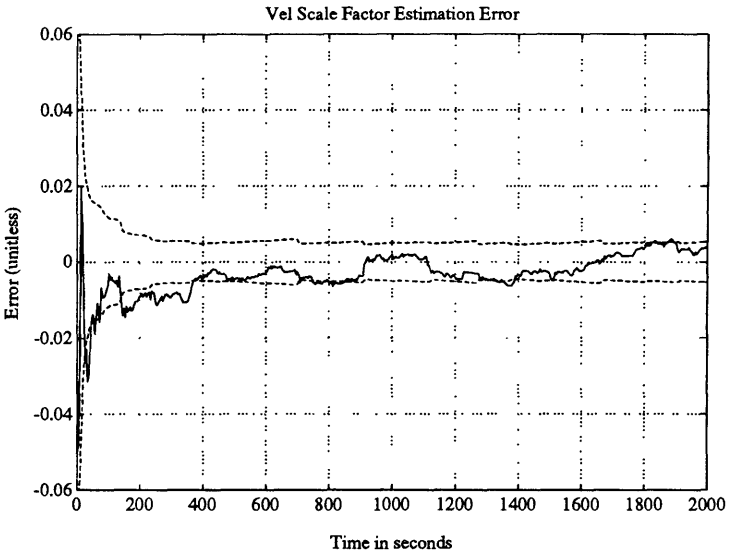

Figure 25 Estimation Error for Velocity Scale Factor

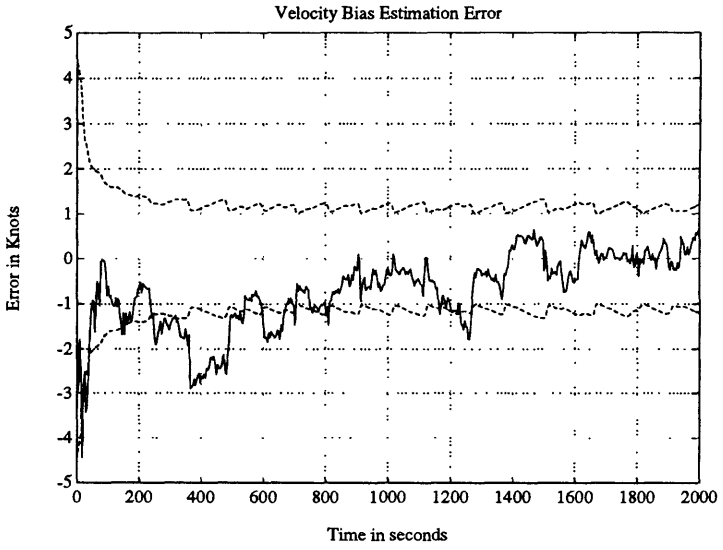

Figure 26 Estimation Error for Velocity Bias

the bias term and the winds; but since the correlation is not consistent, the filter is eventually able to provide a good estimate. The RMS errors for the velocity scale factor and bias were 6.9546e-3 and 1.1404 respectively.

8.4.5 Wind Estimates

Figure 27 shows a plot of the wind estimates and the simulated values. The top line shows the North velocity, the middle line shows the down velocity, and the bottom line shows the East velocity. Figure 28 shows the error in the North wind estimate. The standard deviation lines vary continuously with aircraft maneuvering, but generally contain the errors in the wind estimate. The RMS errors in the estimates were 1.3813, 1.4857 and 1.2246 knots in the North, East, and down directions respectively. The total RMS error was 1.3862 knots. These values were very dependent on the noise value used in the simulation. Typically they were a fraction of a knot higher than the noise value used in the wind simulation.

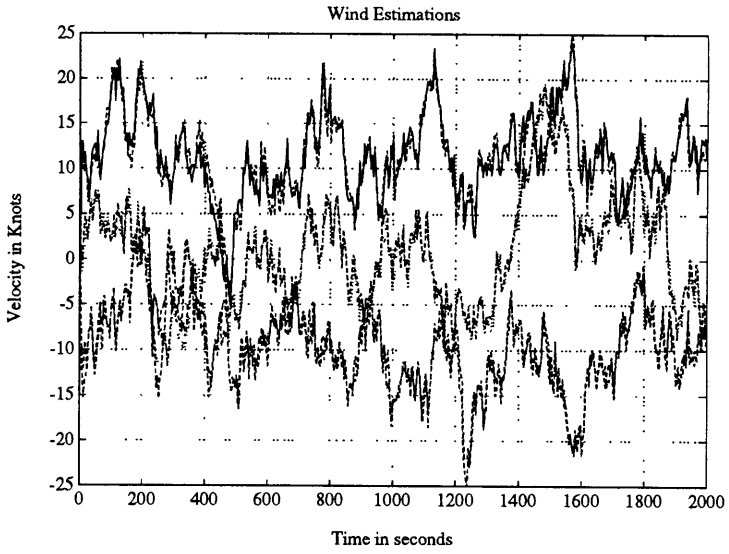

Figure 27 Wind Velocity Estimates

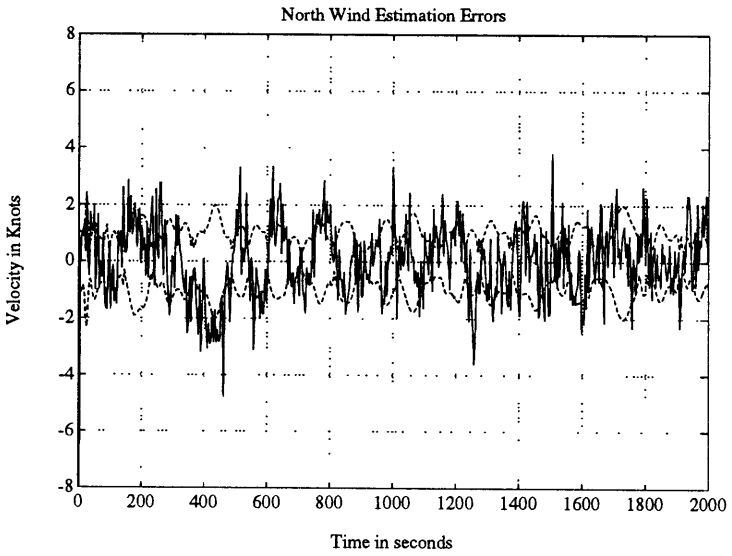

Figure 28 North Wind Estimation Error

8.4.6 Measurement and Truth Residuals

Figure 29 shows the North velocity measurement residual. The RMS residuals were 1.0944, 1.0949, and 1.0547 knots. The total RMS residual was 1.0815 knots. The residuals between the true and estimated velocities were slightly better: 1.0652, 1.0684, and 1.0250 knots. The total RMS residual was 1.0531 knots. As with the wind estimates, all of these values followed slightly above the wind noise value used in the simulation. Note that the velocity residuals show the errors at the end of each one second period. Since the errors are growing during this period, these values show the worst case error. The HARS signals put out 50 estimates during this one second period, so the average error should be much less than shown.

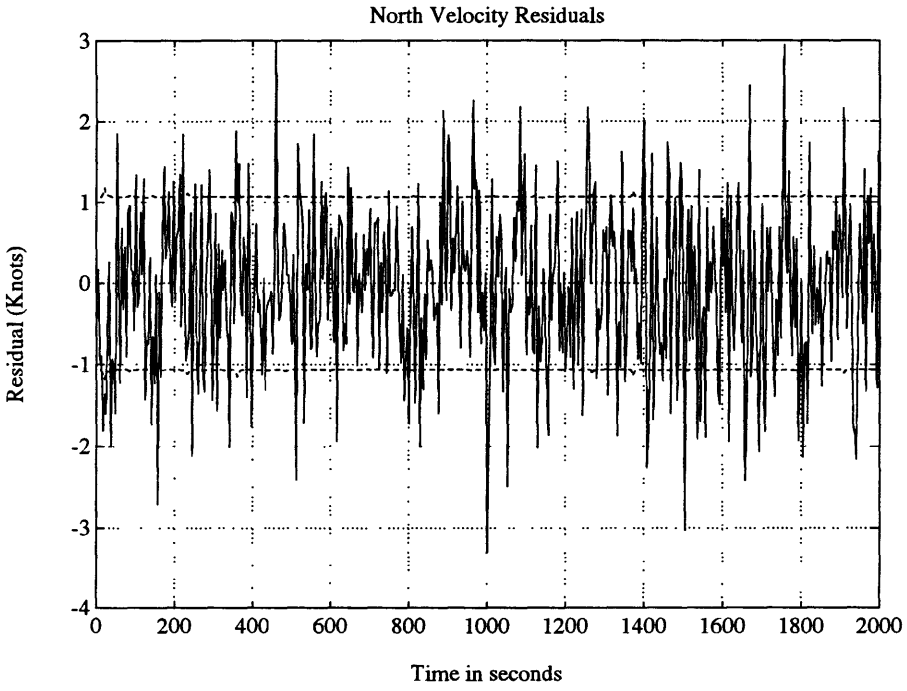

Figure 29 North Velocity Measurement Residual

9. Conclusions

9.1 Recommendations for Future Work

9.1.1 HARS Misalignment Model

The current HARS model is based on the assumption that the erection loop is always active. In reality, the erection loops open when the aircraft experiences a certain level of acceleration. That level is currently unknown. Future studies can evaluate recorded aircraft data to develop a model of the erection loop behavior and then implement the improvement in the filter.

9.1.2 Observability Effects of Flight Path

Early in the filter evaluation, it was observed that the rate of convergence and the accuracy of the state estimates were highly dependent on the flight path. A wings-level, straight-line flight path produced widely diverging state estimates; but the velocity residuals were still very good. To achieve good state estimates all parameters of the flight path were varied continuously. This may be realistic during flight in the target area, but is very unrealistic for flight en route to the target area. Future work should investigate the effects of low dynamic periods of flight.

9.1.3 Coasting Without Measurements

Weapons delivery passes typically involve an inverted roll-in. During the inverted period, the GPS antenna is blocked by the airframe for three to five seconds. Pilots require peak accuracy for weapons delivery three to five seconds after roll-out. Future work should investigate the effects of the blanking period and the speed of recovery.

9.1.4 Combine Navigation and Error Estimators.

This project developed and tested the navigation estimator and the error estimator separately. The navigation estimator was developed first due to its relative simplicity and the error estimator developed later. In future work, the navigation estimator should be retuned based on the results of the error estimator testing. Measurement noise values

need to match the observed output of the error estimator. Process noise values can be reduced to smooth the more volatile estimate expected with the new measurement values. These values should be based off recorded aircraft data rather than assumed models.

9.1.5 Elimination of Bank Angle Corrections

The error estimator demonstrated little ability to estimate errors in the bank angle. This is likely do to the small effect bank angle errors have on the total velocity calculations and the low observability of that effect. Future work may consider dropping this state. Removing the state should not improve the measurement residuals, but it may reduce the computational burden somewhat.

9.2 Summary of Results

The navigation filter showed RMS velocity errors just slightly below the standard deviation of the HARS velocity inputs. In the test runs the HARS velocities had a 0.3 f/s standard deviation noise added. The RMS error in the velocity estimates was 0.2897 f/s. The RMS error in the position estimate was 0.6020 feet. Performing highly dynamic maneuvers increased the RMS errors by only 0.8%. If it was possible to have completely windless environment, these numbers may actually be representative of the filter's performance. However, as was seen in the error estimator testing, the effect of winds will raise the HARS velocity noise an order of magnitude. Even considering this, the testing shows that the filter is effective. It was very responsive in following acceleration inputs and adequately propagated the velocity and position data at the higher data rate of 50Hz.

The error estimator proved fairly robust to changes in the wind models, and insensitive to the process noise values. Estimation accuracy was very dependent on the wind volatility. The most critical issue is the true velocity residual. In all cases, this residual was only very slightly above the wind noise level. With an assumed wind volatility of four knots^2/second the true velocity residual showed an accuracy of about 2.1

knots standard deviation. Again, these are the worst case errors that build up over the one second cycle. The average residual for the 50Hz data signal should be much less.

10. Appendix A :Matlab Routines

10.1 Navigation Estimator Routines

10.1.1 Navest

```
function [K,Xp,Pp,Xm,Pm]=navest(A,G,C,Q,R,x0,P0,Y,n,ys,yf);
%[K,Xp,Pp,Xm,Pm]=navest(A,G,C,Q,R,x0,P0,Y,n,ys,yf);
%     A-10 high-rate navigation estimator. Discrete time multirate
%     Kalman filter algorithm. x0 is row vector of initial states.
%     Y has measurements across columns and time down rows. A is
%     number of high-rate samples per low-rate sample. ys and yf
%     are the start and stop column numbers of the high-rate measures
%     in the Y matrix. p is value before propagating. m is value
%     before incorporating measure.
[dimt dimy]=size(Y);
dimx=length(x0);
offx=dimx-1;

K=zeros(dimt*dimx,dimy);
Xp=zeros(dimt,dimx);
Xm=zeros(dimt+1,dimx);
Pp=zeros(dimt*dimx,dimx);
Pm=zeros(dimt*dimx+dimx,dimx);
ID=eye(P0);
Xm(1,:)=x0;
Pm(1:dimx,:)=P0;
Pt=P0;

tx=1;
for t=1:dimt
 if rem(t-1,n)==0,
  INO=Y(t,:)-Xm(t,:)*C';
  Kt=Pt*C'*inv(C*Pt*C'+R);
  K(tx:tx+offx,:)=Kt;
  Ppt=[ID-Kt*C]*Pt*[ID-Kt*C]'+Kt*R*Kt';
  t
 else
  C1=C(ys:yf,:);
  INO=Y(t,ys:yf)-Xm(t,:)*C1';
  Kt=Pt*C1'*inv(C1*Pt*C1'+R(ys:yf,ys:yf));
  K(tx:tx+offx,ys:yf)=Kt;
  Ppt=[ID-Kt*C1]*Pt*[ID-Kt*C1]'+Kt*R(ys:yf,ys:yf)*Kt';
 end
 Xp(t,:)=Xm(t,:)+INO*Kt';
 Pp(tx:tx+offx,:)=Ppt;
```

```
  Xm(t+1,:)=Xp(t,:)*A';
  Pt=A*Ppt*A'+G*Q*G';
  tx=tx+dimx;
  Pm(tx:tx+offx,:)=Pt;
end
Xm=Xm(1:dimt,:);
Pm=Pm(1:dimt*dimx,:);
```

10.1.2 SSNAVEST

```
function [Xp,Xm]=ssnavest(A,C,x0,K1,K2,K3,K4,Y,n,ys,yf);
%[Xp,Xm]=ssnavest(A,C,x0,K1,K2,K3,K4,Y,n,ys,yf);
%       A-10 high-rate navigation estimator. Discrete time multirate
%       Kalman filter algorithm. Uses multiple fixed gains K1 through
%       K4.  x0 is row vector of initial states.
%       Y has measurements across columns and time down rows.  A is
%       number of high-rate samples per low-rate sample.  ys and yf
%       are the start and stop column numbers of the high-rate measures
%       in the Y matrix.  p is value before propagating. m is value
%       before incorporating measure.
[dimt dimy]=size(Y);
dimx=length(x0);

Xp=zeros(dimt,dimx);
Xm=zeros(dimt+1,dimx);
Xm(1,:)=x0;

for t=1:dimt
  ndx=rem(t-1,n);
  if ndx==0,
    INO=Y(t,:)-Xm(t,:)*C';
    Xp(t,:)=Xm(t,:)+INO*K1';
    t
  else
    C1=C(ys:yf,:);
    INO=Y(t,ys:yf)-Xm(t,:)*C1';
    if ndx==1,
      Xp(t,:)=Xm(t,:)+INO*K2(:,ys:yf)';
    elseif ndx==2,
      Xp(t,:)=Xm(t,:)+INO*K3(:,ys:yf)';
    else
      Xp(t,:)=Xm(t,:)+INO*K4(:,ys:yf)';
    end
  end
  Xm(t+1,:)=Xp(t,:)*A';
end
Xm=Xm(1:dimt,:);
```

10.1.3 Velbod

```
function [vb]=velbod(vel,aoa,ss);
%[vb] = velbod(airspeed, aoa, sideslip)
%       Returns a 3 x N matrix of velocities in body frame coordinates
%       given inputs of 1 x N vectors of airspeed, aoa, and sideslip.
%       Airspeed may be in any units and the resultant velocities will
%       be in the same.  AOA and sideslip must be in radians.
%       The row vectors of VB will be the x, y, z, velocities respectively.

vy=vel.*tan(ss)./cos(aoa)*(-1);
vz=vel.*tan(aoa);

vb=[vel; vy; vz];
```

10.1.4 Velair

```
function [va]=velair(vb,azi,pit,ban);
%[va] = velair(vb, azimuth, pitch, bank)
%       Converts body frame velocities to airmass frame velocities.
%       va and vb are 3 x N matricies.  The row vectors in vb are
%       the x, y, z components of body frame.  The rows of va are the
%       n, e, d components of atmosphere fixed frame.  Azimuth, pitch
%       and bank are in radians (azimuth from true north).

sa=sin(azi);
ca=cos(azi);
sp=sin(pit);
cp=cos(pit);
sb=sin(ban);
cb=cos(ban);

vx=vb(1,:);
vy=vb(2,:);
vz=vb(3,:);

vn=ca.*cp.*vx+(ca.*sp.*sb-sa.*cb).*vy+(ca.*sp.*cb+sa.*sb).*vz;
ve=sa.*cp.*vx+(sa.*sp.*sb+ca.*cb).*vy+(sa.*sp.*cb-ca.*sb).*vz;
vd=sp.*vx*(-1)+cp.*sb.*vy+cp.*cb.*vz;

va=[vn; ve; vd];
```

10.1.5 GPSSIG

```
function [yo]=gpssig(yi);
%Y=gpssig(Y)
%       Returns matrix Y with first 6 columns having every fourth
%       entry replicated over the next three.  Simulates the measurement
```

```
%       matrix of GPS at 50Hz.
yo=yi;
[dimt dimy]=size(yi);
for t=1:dimt
 n=fix((t-1)/4)*4+1;
 yo(t,1:6)=yi(n,1:6);
end
```

10.1.6 KPLOT

```
function [KP]=kplot(K);
%[KP]=kplot(K)
%       Breaks out main elements of gain array for A-1O high-rate est.
%       1) GPS Pos -> Pos  2) GPS Vel -> Pos  3) HARS Vel -> Pos
%       4) GPS Pos -> Vel  5) GPS Vel -> Vel  6) HARS Vel -> Vel
lk=length(K)/6;
KP=zeros(lk,6);
for t=1:lk
 n=(t-1)*6+1;
 m=n+3;
 KP(t,:)=[K(n,1) K(n,4) K(n,7) K(m,1) K(m,4) K(m,7)];
end
```

10.1.7 VARPLOT

```
function [y]=varplot(m);
%[y]=varplot(m); Builds matrix y whos rows are diagonals of submatrices
%       of m.  m must be a matrix whose number of rows is an even multiple
%       of its number of columns.
[dimt dimy]=size(m);
y=zeros(dimt/dimy,dimy);
for t=0:dimt/dimy-1
 r=t*dimy+1;
 y(t+1,:)=diag(m(r:r+dimy-1,:))';
end
```

10.1.8 STAIR2

```
function [xx,yy]=stair2(t,y);
%[xx,yy]=stari2(t,y); plots square line graph for discrete time plots
%    like the normal stairs function, but does not force first and
%    last values to zero.  Use plot(xx,yy) to see graph.
[xx,yy]=stairs(t,y);
lx=length(xx);
xx=xx(2:lx-1);
yy=yy(2:lx-1);
```

10.2 Error Estimator Routines

10.2.1 HCREF

```
function [hed,pit,ban,vel,aoa]=hcref(g,rol,slip,acc,vi,bani,magv);
%[hed,pit,ban,vel,aoa]=hcref(g,rol,slip,acc,vi,bani,magv);
%     hcref provides a reference trajectory in HARS/CADC variables
%     Inputs:
%     g Time based vector of z-axis accelleration, measured in G's
%     rol Time based vector of roll angle CHANGES in radians.
%     slip Time based vector of side-slip angle in radians.
%     vi Column vector of initial velocity in N,E,D coordinates NM/H
%     bani Scalar value of initial bank in radians
%     acc Time based vector of multiplicative thrust change factor.
%     magv is the local magnetic variation in radians
%
%     Outputs:  All are time based column vetors.  Sideslip output is not
%     provided since it is the same as the input.

% Initialize variables
lent=length(g);
v=zeros(lent,1);
aoa=zeros(lent,1);
hed=zeros(lent,1);
pit=zeros(lent,1);
ban=zeros(lent,1);
vel=zeros(lent,1);
nu=[1;0;0];
zu=[0;0;1];
T=1;
ban0=bani;
v0=vi;

% Prime the processing loop
vm=mag3(v0);
vu=v0/vm;  %vu: unit vector along v0
zp=zu-vu'*zu*vu; %projection of zu on plane normal to vu
zp=zp/mag3(zp);

% Main processing loop
for k=1:lent
 vxz=cross3(vu,zp); % unit vector normal to plane formed by v0,zu
 ban0=ban0+rol(k);
 gl=-cos(ban0)*g(k)*zp-sin(ban0)*g(k)*vxz; %vector form of g due to lift
 gt=gl+zu; % total g force including gravity
 v0=v0+360/19*T*gt; % rotate velocity due to g
```

```
 v0=v0*vm/mag3(v0)*acc(k); % correct magnitude error and apply thrust
 v0=v0+360/19*T*vu'*zu*vu; % apply acceleration due to climb or dive
 vm=mag3(v0);
 vu=v0/vm;
% Calulate new bank angle
 gp=gl-vu'*gl*vu; % projection of gl onto plane normal to v0
 gp=gp/mag3(gp);
 zp=zu-vu'*zu*vu; % projection of zu onto plane normal to v0
 zp=zp/mag3(zp);
 ban0=acos(-zp'*gp); % calculate bank from dot product of zp, gp
 quadtest=mag3(cross3(gp,zp)-vu); % correct sign for quadrant
 if quadtest>1,
  ban0=-ban0;
 end
% Calculate AOA
 aoa(k)=.035*mag3(gl);
% Calculate Zero Reference Line
 vxg=cross3(vu,gl); % unit vector normal to vu, gl
 vxg=vxg/mag3(vxg);
 zrl=v0+vm*tan(aoa(k))*gl/mag3(gl)+vm*tan(slip(k))*vxg;
 zrl=zrl/mag3(zrl); % zero reference line
% Calculate Heading
 zrlp=zrl-zu'*zrl*zu; % projection fo zrl onto Earth level plane
 zrlp=zrlp/mag3(zrlp); % unit vector
 hed(k)=acos(zrlp(1)); % arc cosine of north component of zrlp
 if zrl(2)<0,        % correct for quadrant
  hed(k)=2*pi-hed(k);
 end
 hed(k)=hed(k)-magv;   % induce magnetic variation error
% Calculate Pitch
 pit(k)=-acos(zrl'*zrlp)*sign(zrl(3));
% Calculate Bank referenced to Zero Sight Line
 gpz=gl-zrl'*gl*zrl;
 gpz=gpz/mag3(gpz);
 zpz=zu-zrl'*zu*zrl;
 zpz=zpz/mag3(zpz);
 ban(k)=acos(-zpz'*gpz);
 quadtest=mag3(cross3(gpz,zpz)-zrl);
 if quadtest>1,
  ban(k)=-ban(k);
 end
% Calculate Airspeed
 vel(k)=mag3(zrl'*v0*zrl);
end
```

10.2.2 GPSVEL

```
function [Vg]=gpsvel(vel,hed,pit,ban,aoa,ss,ycr,ycw,magv);
%[Vg]=gpsvel(vel,hed,pit,ban,aoa,ss,ycr,ycw,magv);
%       Creates GPS velocities from HARS data and corruption values.
%       ycr=[zn,ze,zd,Dpit,Dban,vs,vb];
%       ycw=n,e,d wind in columns time in rows.

% Initialize variables
 lent=length(vel);
 Vg=zeros(lent,3);

% Main loop
 for k=1:lent;
  % Pack HARS array
  yh(1)=vel(k);
  yh(2)=hed(k);
  yh(3)=pit(k);
  yh(4)=ban(k);
  yh(5)=aoa(k);
  yh(6)=ss(k);
  % Pack correction array
  yc=[ycr,ycw(k,:)]';
  % Correct airspeed and euler angles
  yh=evcorr(yh,yc,magv);
  % Convert airspeed into airmass frame velocities
  vg=velhars(yh);
  % Correct airmass velocities for misalignment and wind
  vg=zwcorr(yc,vg);
  % Pack velocities into matrix
  Vg(k,:)=vg';
 end
```

10.2.3 ERREST

```
function [XC,PS,RTS,YC,VH]=errest(vel,hed,pit,ban,aoa,ss,Vg,magv,A,QC,RC,P,yc);
%[XC,PS,RTS,YC,VH]=errest(vel,hed,pit,ban,aoa,ss,Vg,magv,A,QC,RC,P,yc);
%       Simulates A10 error estimator and velocity estimator
%       Inputs: First 6 are time based column vectors,
%               Vg is 3x3 matrix of GPS velocities
%               QC, RC are the noise matricies for the error estimator
%               P is the initial covariance matrix
%               yc contains initial values for the corrections

% Initialize variables
 lent=length(vel);
 AI=A-eye(A);
```

```
% Dimension Arrays
 XC=zeros(lent,10);
 YC=XC;
 VH=zeros(lent,3);
 PS=zeros(lent*10,10);
 RTS=zeros(lent,3);

% Main Loop
 for k=1:lent;
  % Pack HARS array
   yh(1)=vel(k);
   yh(2)=hed(k);
   yh(3)=pit(k);
   yh(4)=ban(k);
   yh(5)=aoa(k);
   yh(6)=ss(k);
  % Correct airspeed and euler angles
   yh=evcorr(yh,yc,magv);
  % Convert airspeed into airmass frame velocities
   vh=velhars(yh);
  % Correct airmass velocities for misalignment and wind
   vh=zwcorr(yc,vh);
  % Run error estimator
   % Calculate measurement jacobian
    H=calch(yh,yc,vel(k));
%%% Modify H for debug purposes
   % Removes pitch,bank error
   % H(:,4:5)=zeros(3,2);
   % Remove wind error
   % H(:,8:10)=zeros(3,3);
%%% End debug section
   % Get measurments, calculate gain and states
    vg=Vg(k,:)';
    P=A*P*A'+QC;
    Pt=H*P;       % Intermediate variable to save processing
    RT=Pt*H'+RC;  % Residual track
    K=Pt'*inv(RT);
    xc=AI*yc+K*(vg-vh);
    P=P-K*H*P;
    P=(P+P')/2;   % Keep P symmetric
  % Store important variable for analysis
    XC(k,:)=xc';
    YC(k,:)=yc';
    VH(k,:)=vh';
```

```
   stop=k*10;
   PS(stop-9:stop,:)=P;
   RTS(k,:)=[RT(1,1),RT(2,2),RT(3,3)];
  % Add errors to correction terms
  yc=yc+xc;
  % Loop
 end
```

10.2.4 EVCORR

```
function [yh]=evcorr(yh,yc,magv)
%[yh]=evcorr(yh,yc)  Adds corrections to HARS values

yh(1)=yc(6)*(yh(1)-300)+yc(7)+300;
yh(2)=yh(2)+magv;
yh(3)=yh(3)+yc(4);
yh(4)=yh(4)+yc(5);
```

10.2.5 VELHARS

```
function [vh]=velhars(yh);
%[vh] = velhars(yh)
%       Converts HARS data to airmass frame velocities.
%       yh=[vel, azimuth, pitch, bank, AOA, sideslip]

% Initialize Euler angle variables
sa=sin(yh(2));
ca=cos(yh(2));
sp=sin(yh(3));
cp=cos(yh(3));
sb=sin(yh(4));
cb=cos(yh(4));

% Convert Airspeed into body frame velocities
vx=yh(1);
vy=yh(1)*tan(yh(6))/cos(yh(5))*(-1);
vz=yh(1)*tan(yh(5));

% Rotate body frame velocities to airmass frame
vn=ca.*cp.*vx+(ca.*sp.*sb-sa.*cb).*vy+(ca.*sp.*cb+sa.*sb).*vz;
ve=sa.*cp.*vx+(sa.*sp.*sb+ca.*cb).*vy+(sa.*sp.*cb-ca.*sb).*vz;
vd=sp.*vx*(-1)+cp.*sb.*vy+cp.*cb.*vz;

% Pack velocities into vector
vh=[vn; ve; vd];
```

10.2.6 ZWCORR

```
function [vh]=zwcorr(yc,vh);
```

```
%[vh]=zwcorr(yc,vh);
%       Corrects HARS airmass frame velocities with axis misalignment
%       corrections and wind.  Outputs Earth-surface frame velocities.

vn=vh(1)-yc(3)*vh(2)+yc(2)*vh(3)+yc(8);
ve=yc(3)*vh(1)+vh(2)-yc(1)*vh(3)+yc(9);
vd=-yc(2)*vh(1)+yc(1)*vh(2)+vh(3)+yc(10);

vh=[vn;ve;vd];
```

10.2.7 CALCH

```
function [h]=calch(yh,yc,vel)
%[h] = clach(yh,yc,vel) Calculates the jacobian of the error estimator measurement
function
%       Inputs
%         yh: HARS measurements [airspeed, azi, pit, ban, aoa, sideslip]'
%         yc: Measurement correction values [zn ze zd Dpit Dban vs vb wn we wd]'
%       Outputs
%         h: Jacobian matrix of measurement fucntion.  Used in gain equation.

% Dimension h
h=zeros(3,10);

% Initialize Euler Angle Variables
sa=sin(yh(2));
ca=cos(yh(2));
sp=sin(yh(3));
cp=cos(yh(3));
sb=sin(yh(4));
cb=cos(yh(4));
sasp=sa*sp;
sacp=sa*cp;
sasb=sa*sb;
sacb=sa*cb;
casp=ca*sp;
cacp=ca*cp;
casb=ca*sb;
cacb=ca*cb;
spsb=sp*sb;
spcb=sp*cb;
cpsb=cp*sb;
cpcb=cp*cb;
saspsb=sasp*sb;
saspcb=sasp*cb;
sacpsb=sacp*sb;
sacpcb=sacp*cb;
```

```
caspsb=casp*sb;
caspcb=casp*cb;
cacpsb=cacp*sb;
cacpcb=cacp*cb;
ta=tan(yh(5));
tbca=tan(yh(6))/cos(yh(5));

% Fill in h matrix
% Sensitivity to north misalignment error
%h(1,1)=0; (already equals zero)
 h(2,1)=yh(1)*(sp+cpsb*tbca-cpcb*ta);
 h(3,1)=yh(1)*(sacp-(saspsb+cacb)*tbca+(saspcb-casb)*ta);

% Sensitivity to east misalignment error
 h(1,2)=yh(1)*(-sp-cpsb*tbca+cpcb*ta);
%h(2,2)=0;
 h(3,2)=yh(1)*(-cacp+(caspsb-sacb)*tbca-(caspcb+sasb)*ta);

% Sensitivity to down misalignment error
 h(1,3)=yh(1)*(-sacp+(saspsb+cacb)*tbca+(casb-saspcb)*ta);
 h(2,3)=yh(1)*(cacp-(caspsb-sacb)*tbca+(caspcb+sasb)*ta);
%h(3,3)=0;

% Sensitivity to pitch bias
 h(1,4)=yh(1)*(-casp-cacpsb*tbca+cacpcb*ta);
 h(2,4)=yh(1)*(-sasp-sacpsb*tbca+sacpcb*ta);
 h(3,4)=yh(1)*(-cp+spsb*tbca-spcb*ta);

% Sensitivity to bank bias
 h(1,5)=yh(1)*(-(caspcb+sasb)*tbca-(caspsb-sacb)*ta);
 h(2,5)=yh(1)*((-saspcb+casb)*tbca-(saspsb+cacb)*ta);
 h(3,5)=yh(1)*(-cpcb*tbca-cpsb*ta);

% Sensitivity to velocity scale factor and bias
 temp=cacp-yc(3)*sacp-yc(2)*sp;
 temp=temp-(caspsb-sacb-yc(3)*(saspsb+cacb)+yc(2)*cpsb)*tbca;
 temp=temp+(caspcb+sasb-yc(3)*(saspcb-casb)+yc(2)*cpcb)*ta;
 h(1,6)=(vel-300)*temp;
 h(1,7)=temp;
 temp=sacp+yc(3)*cacp+yc(1)*sp;
 temp=temp-(saspsb+cacb+yc(3)*(caspsb-sacb)-yc(1)*cpsb)*tbca;
 temp=temp+(saspcb-casb+yc(3)*(caspcb+sasb)-yc(1)*cpcb)*ta;
 h(2,6)=(vel-300)*temp;
 h(2,7)=temp;
 temp=-sp-yc(2)*cacp+yc(1)*sacp;
```

```
temp=temp-(cpsb-yc(2)*(caspsb-sacb)+yc(1)*(saspsb+cacb))*tbca;
temp=temp+(cpcb-yc(2)*(caspcb+sasb)+yc(1)*(saspcb-casb))*ta;
h(3,6)=(vel-300)*temp;
h(3,7)=temp;

% Sensitivity to wind
h(1:3,8:10)=eye(3);
```

10.2.8 CROSS3

```
function xc=cross3(y,z);
%xc=cross3(y,z);  takes the cross product of 3x1 vectors y, z
xc=[y(2)*z(3)-y(3)*z(2); y(3)*z(1)-y(1)*z(3); y(1)*z(2)-y(2)*z(1)];
```

10.2.9 MAG3

```
function xm=mag3(x);
%xm = mag3(x); takes the magnitude of a 3 element vector x.
xm=sqrt(x(1)^2+x(2)^2+x(3)^2);
```

10.2.10 PLOTCORR

```
function plotcorr(yc,ycr,ycw)
% plotcorr(yc,ycr,ycw);

[ln,wd]=size(yc);
t=1:ln;

clg
subplot(221)
p1=ycw(:,4);
p2=ycw(:,5);
p3=ycw(:,6);
plot(t,yc(:,1),t,yc(:,2),t,yc(:,3),t,p1,'c8.',t,p2,'c11.',t,p3,'c10.');
grid
title('   Misalignments')

subplot(222)
p1=ycr(4)*ones(ln,1);
p2=ycr(5)*ones(ln,1);
plot(t,yc(:,4),t,yc(:,5),t,p1,'r--',t,p2,'g--');grid
title('   Euler Corrections')

subplot(223)
p1=ycr(6)*ones(ln,1);
p2=ycr(7)*ones(ln,1);
p3=(yc(:,6)-ones(ln,1))*10+1;
p4=(p1-ones(ln,1))*10+1;
plot(t,p3,t,yc(:,7),t,p4,'r--',t,p2,'g--');grid
```

```
title('   Velocity Corr')

subplot(224)
p1=ycw(:,1);
p2=ycw(:,2);
p3=ycw(:,3);
plot(t,yc(:,8),t,yc(:,9),t,yc(:,10),t,p1,'r--',t,p2,'g--',t,p3,'b--');
grid
title('Winds')

subplot(111)
```

10.2.11 PLOTMISS

```
function plotmiss(yc,ycw)
% plotmiss(yc,ycw);

[ln,wd]=size(yc);
t=1:ln;
clg
p1=ycw(:,4);
p2=ycw(:,5);
p3=ycw(:,6);
subplot(221)
plot(t,yc(:,1),t,p1,'c8.');grid
title('   North Misalign')
subplot(222)
plot(t,yc(:,2),t,p2,'c11.');grid
title('   East Misalign')
subplot(223)
plot(t,yc(:,3),t,p3,'c10.');grid
title('   Down Misalign')
subplot(111)
```

10.2.12 PLOTERRM

```
% ploterrm
clg
t1=1:ln;
subplot(221)
p1=sqrt(PD(1:ln,1));
p2=-p1;
plot(t1,YC(1:ln,1)-ycw(1:ln,4),t1,p1,'g--',t1,p2,'g--');grid
title('  N Mislgn Corrctns')

subplot(222)
p1=sqrt(PD(1:ln,2));
p2=-p1;
```

```
plot(t1,YC(1:ln,2)-ycw(1:ln,5),t1,p1,'g--',t1,p2,'g--');grid
title(' E Mislgn Corrctns')

subplot(223)
p1=sqrt(PD(1:ln,3));
p2=-p1;
plot(t1,YC(1:ln,3)-ycw(1:ln,6),t1,p1,'g--',t1,p2,'g--');grid
title(' D Mislgn Corrctns')

clear p1
clear p2
subplot(111)
```

10.2.13 PLOTERRE

```
% ploterre
clg
t1=1:ln;
subplot(221)
p1=sqrt(PD(1:ln,4));
p2=-p1;
plot(t1,YC(1:ln,4)-ycr(4)*ones(ln,1),t1,p1,'g--',t1,p2,'g--');grid
title(' Pitch Corrections')

subplot(222)
p1=sqrt(PD(1:ln,5));
p2=-p1;
plot(t1,YC(1:ln,5)-ycr(5)*ones(ln,1),t1,p1,'g--',t1,p2,'g--');grid
title(' Bank Corrections')

subplot(223)
p1=sqrt(PD(1:ln,6));
p2=-p1;
plot(t1,YC(1:ln,6)-ycr(6)*ones(ln,1),t1,p1,'g--',t1,p2,'g--');grid
title(' Vel Scale Factor')

subplot(224)
p1=sqrt(PD(1:ln,7));
p2=-p1;
plot(t1,YC(1:ln,7)-ycr(7)*ones(ln,1),t1,p1,'g--',t1,p2,'g--');grid
title(' Velocity Bias')

clear p1
clear p2
subplot(111)
```

10.2.14 PLOTERRW

```
% ploterrw
clg
t1=1:ln;
subplot(221)
p1=sqrt(PD(1:ln,8));
p2=-p1;
plot(t1,YC(1:ln,8)-ycw(1:ln,1),t1,p1,'g--',t1,p2,'g--');grid
title(' North Wind Corrections')

subplot(222)
p1=sqrt(PD(1:ln,9));
p2=-p1;
plot(t1,YC(1:ln,9)-ycw(1:ln,2),t1,p1,'g--',t1,p2,'g--');grid
title(' East Wind Corrections')

subplot(223)
p1=sqrt(PD(1:ln,10));
p2=-p1;
plot(t1,YC(1:ln,10)-ycw(1:ln,3),t1,p1,'g--',t1,p2,'g--');grid
title(' Down Wind Corrections')

clear p1
clear p2
subplot(111)
```

10.2.15 PLOTRES

```
% plotres
clg
t1=2:ln;

subplot(221)
p1=sqrt(RTS(2:ln,1));
p2=-p1;
plot(t1,Vg(2:ln,1)-VH(2:ln,1),t1,p1,'g--',t1,p2,'g--');grid
title(' North Velocity Residuals')

subplot(222)
p1=sqrt(RTS(2:ln,2));
p2=-p1;
plot(t1,Vg(2:ln,2)-VH(2:ln,2),t1,p1,'g--',t1,p2,'g--');grid
title(' East Velocity Residuals')

subplot(223)
p1=sqrt(RTS(2:ln,3));
```

```
p2=-p1;
plot(t1,Vg(2:ln,3)-VH(2:ln,3),t1,p1,'g--',t1,p2,'g--');grid
title(' Down Velocity Residuals')

clear p1
clear p2
subplot(111)
```

10.2.16 WEED

```
function [outvec]=weed(invect,skip)
%outvec=weed(invect,skip)  outputs subset of input vector
[ln,wd]=size(invect);
outvec=zeros(ln/skip,wd);
t1=1;
for t=1:ln/skip
 outvec(t,:)=invect(t1,:);
 t1=t1+skip;
end
```

10.2.17 PHED

```
% phed
plot(t,hed*180/pi,t,pit*180/pi,t,ban*180/pi);grid
```

10.2.18 SHOWRMS

```
rmsmis=sqrt(mean((YC(:,1:3)-ycw(:,4:6)).^2));
ons=ones(ln,1);
difeul=YC(:,4:7)-[ons*ycr(4) ons*ycr(5) ons*ycr(6) ons*ycr(7)];
rmseul=sqrt(mean(difeul.^2));
clear ons
clear difeul
rmswnd=sqrt(mean((YC(:,8:10)-ycw(:,1:3)).^2));
rmsres=sqrt(mean((Vg-VH).^2));
rmserr=sqrt(mean((Vt-VH).^2));

mrmsmis=sqrt(mean(rmsmis.^2));
mrmswnd=sqrt(mean(rmswnd.^2));
mrmsres=sqrt(mean(rmsres.^2));
mrmserr=sqrt(mean(rmserr.^2));
echo on
% Misalign    Euler    Velocities    Wind    Residuals    Errors
echo off
[rmsmis' [rmseul(1:2)';0] [rmseul(3:4)';0] rmswnd' rmsres' rmserr']
[mrmsmis 0 0 mrmswnd mrmsres mrmserr]
```

10.2.19 STATCORR

```
%function statcorr(PS,state);
% statcorr(PS,state)
```

```
%       Calculates and plots time history of correlation coefficients
[lnt,wdt]=size(PS);
C=zeros(lnt/wdt,wdt);
count=0;
for i=state:wdt:lnt-wdt-1+state
 count=count+1;
 for j=1:10
  C(count,j)=PS(i,j)/sqrt(PS(i,state)*PS(i-state+j,j));
 end
end
axs=[0,count,-1,1];
axis(axs);
plot(C);grid
axis;
```

11. Appendix B: Testing Scripts and Results

11.1 Navigation Estimator Tests

11.1.1 Multiple Steady State Gains

11.1.1.1 Time Varying Filter Script

```
%Kalm9 Runs time varying fliter with degrees and Nm/Hr as units
%Provides multirate gain solution for 0 deg latitude
ln=28;
pnorth=0;
cospn=cos(pnorth*pi/180);
A=eye(6)+diag([1/1.08e7 1/(cospn*1.08e7) 38/1125],3);
B=[zeros(3);eye(3)];
C=[eye(6);zeros(3) eye(3)];
D=zeros(9,3);
U=zeros(ln,3);
x0=[0 0 0 200 100 0];
[Y,x]=dlsim(A,B,C,D,U,x0);
rand('normal')
Y(:,1)=Y(:,1)+rand(ln,1)*2.0336e-5;
Y(:,2)=Y(:,2)+rand(ln,1)*2.0336e-5/cospn;
Y(:,3)=Y(:,3)+rand(ln,1)*7.4184;
Y(:,4:6)=Y(:,4:6)+rand(ln,3)*.43925;
Y(:,7:9)=Y(:,7:9)+rand(ln,3)*.1757;
G=eye(6);
Q=diag([3.7704e-15 3.7704e-15/cospn^2 5.0176e-4 1.7591 1.7591 1.7591]);
R=diag([4.1353e-15 4.1353e-15/cospn^2 55.0326 .19294 .19294 .19294 3.087e-2
3.087e-2 3.087e-2]);
P0=diag([1.5474e-14 1.5474e-14 3.4450e-1 3.0346e-2 3.0346e-2  3.0346e-2]);
Y=gpssig(Y);
n=4;
ys=7;
yf=9;
[K,xp,pp,xm,pm]=navest(A,G,C,Q,R,x0,P0,Y,n,ys,yf);
vp=varplot(pp);
kp=kplot(K);
t=1:ln;
```

11.1.1.2 Gain Results for Zero Degrees Latitude

K1

0.82511	0	0	3.7968E-11	0	0	2.373E-10	0	0
0	0.82511	0	0	3.7968E-11	0	0	2.373E-10	0
0	0	0.0062207	0	0	7.8702E-05	0	0	4.9189E-04
1771.5	0	0	0.13591	0	0	0.84944	0	0
0	1771.5	0	0	0.13591	0	0	0.84944	0
0	0	2.7592E-07	0	0	0.13591	0	0	0.84944

K2

0	0	0	0	0	0	1.3409E-09	0	0
0	0	0	0	0	0	0	1.3409E-09	0
0	0	0	0	0	0	0	0	4.9604E-04
0	0	0	0	0	0	0.983	0	0
0	0	0	0	0	0	0	0.983	0
0	0	0	0	0	0	0	0	0.983

K3

0	0	0	0	0	0	1.5663E-09	0	0
0	0	0	0	0	0	0	1.5663E-09	0
0	0	0	0	0	0	0	0	5.715E-04
0	0	0	0	0	0	0.98304	0	0
0	0	0	0	0	0	0	0.98304	0
0	0	0	0	0	0	0	0	0.98304

K4

0	0	0	0	0	0	1.5702E-09	0	0
0	0	0	0	0	0	0	1.5702E-09	0
0	0	0	0	0	0	0	0	5.728E-04
0	0	0	0	0	0	0.98304	0	0
0	0	0	0	0	0	0	0.98304	0
0	0	0	0	0	0	0	0	0.98304

11.1.1.3 Gain Results for 30 Degree Latitude

K1

0.82511	0	0	3.7968E-11	0	0	2.373E-10	0	0
0	0.82511	0	0	4.3842E-11	0	0	2.7401E-10	0
0	0	0.0062207	0	0	7.8702E-05	0	0	4.9189E-04
1771.5	0	0	0.13591	0	0	0.84944	0	0
0	1534.1	0	0	0.13591	0	0	0.84944	0
0	0	2.7592E-07	0	0	0.13591	0	0	0.84944

K2

0	0	0	0	0	0	1.3409E-09	0	0
0	0	0	0	0	0	0	1.5483E-09	0
0	0	0	0	0	0	0	0	4.9604E-4
0	0	0	0	0	0	0.983	0	0
0	0	0	0	0	0	0	0.983	0
0	0	0	0	0	0	0	0	0.983

K3

0	0	0	0	0	0	1.5663E-09	0	0
0	0	0	0	0	0	0	1.8086E-09	0
0	0	0	0	0	0	0	0	5.715E-4
0	0	0	0	0	0	0.98304	0	0
0	0	0	0	0	0	0	0.98304	0
0	0	0	0	0	0	0	0	0.98304

K4

0	0	0	0	0	0	1.5702E-09	0	0
0	0	0	0	0	0	0	1.8131E-09	0
0	0	0	0	0	0	0	0	5.728E-4
0	0	0	0	0	0	0.98304	0	0
0	0	0	0	0	0	0	0.98304	0
0	0	0	0	0	0	0	0	0.98304

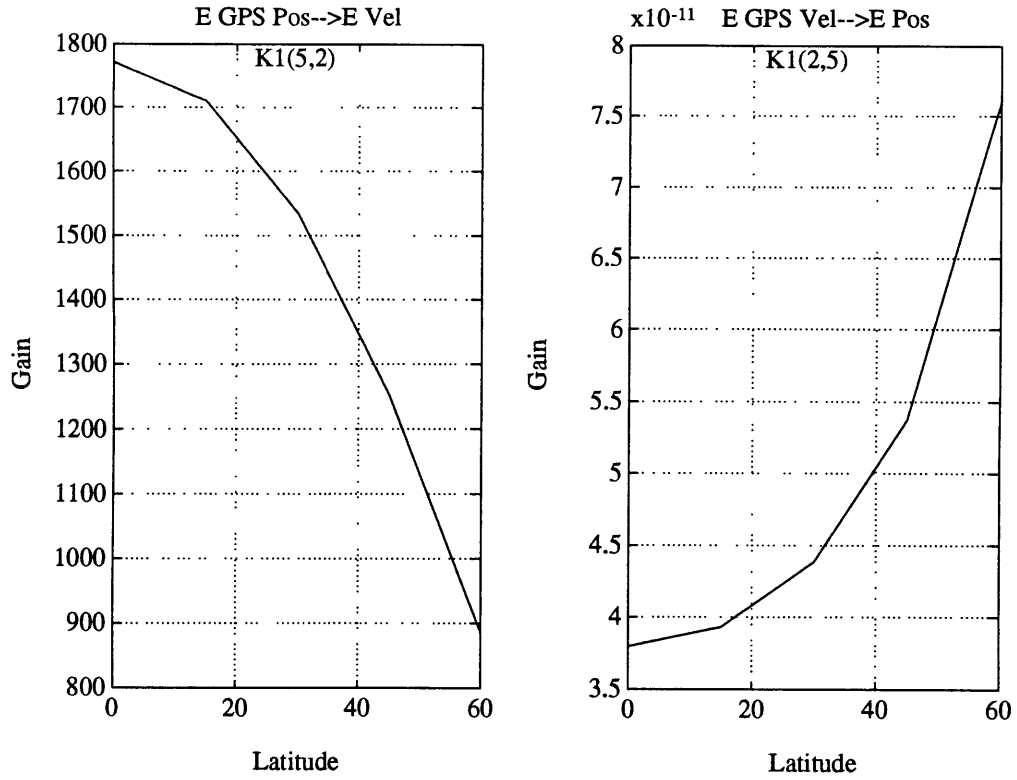

Figure 30 Plot of How Gains Changes with Latitude

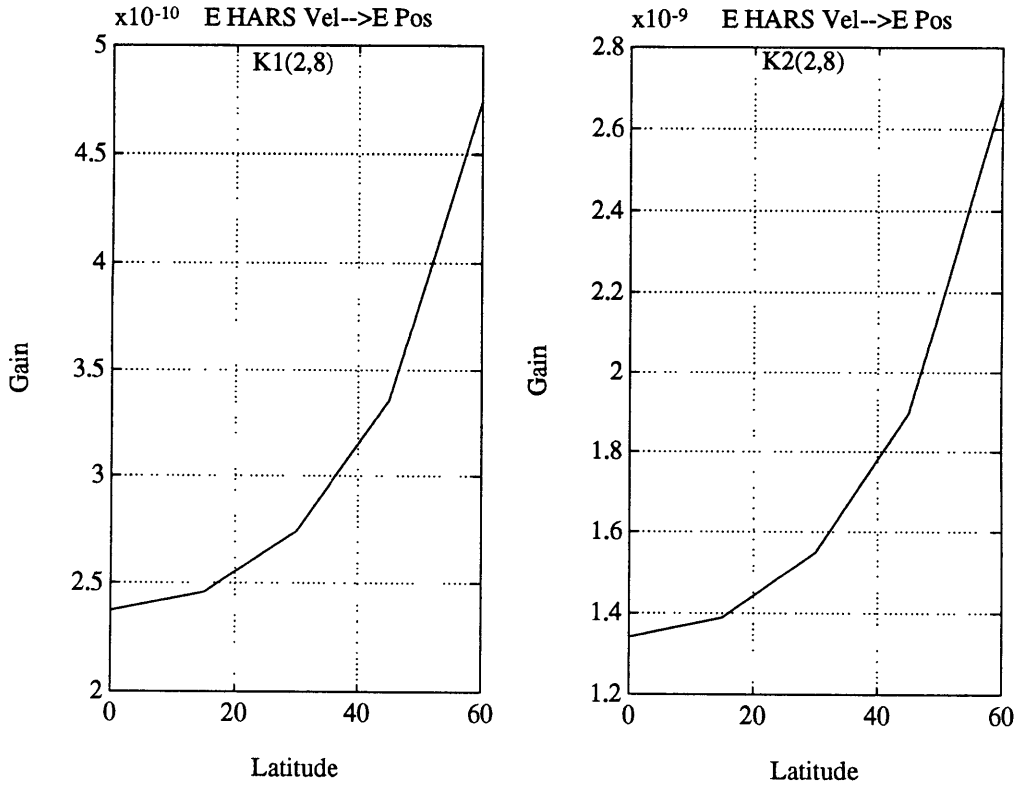

Figure 31 Plot of How Gains Change with Latitude

11.1.2 Constant Velocity Test

```
%Kalm6
%ln=200;
A=eye(6)+diag([.02 .02 .02],3);
B=[zeros(3);eye(3)];
C=[eye(6);zeros(3) eye(3)];
D=zeros(9,3);
U=zeros(ln,3);
x0=[0 0 0 400 200 0];
[Y,x]=dlsim(A,B,C,D,U,x0);
Yt=Y;
rand('normal')
Y(:,1:3)=Y(:,1:3)+rand(ln,3)*7.4;
Y(:,4:6)=Y(:,4:6)+rand(ln,3)*.74;
Y(:,7:9)=Y(:,7:9)+rand(ln,3)*.3;
G=eye(6);
Q=diag([.005 .005 .005 5 5 5]);
R=diag([54.8 54.8 54.8 .548 .548 .548 .0881 .0881 .0881]);

K1=[1.89861e-2 0 0 4.63737e-5 0 0 2.88454e-4 0 0;
   0 1.89861e-2 0 0 4.63737e-5 0 0 2.88454e-4 0;
   0 0 1.89861e-2 0 0 4.63737e-5 0 0 2.88454e-4;
   4.63737e-7 0 0 .136464 0 0 .848834 0 0;
   0 4.63737e-7 0 0 .136464 0 0 .848834 0;
   0 0 4.63737e-7 0 0 .136464 0 0 .848834];
K2=[2.94614e-4 0 0; 0 2.94614e-4 0; 0 0 2.94614e-4;
   .982936 0 0; 0 .982936 0; 0 0 .982936];
K2=[zeros(6) K2];
K3=[3.39709e-4 0 0; 0 3.39709e-4 0; 0 0 3.39709e-4;
   .982975 0 0; 0 .982975 0; 0 0 .982975];
K3=[zeros(6) K3];
K4=[3.40489e-4 0 0; 0 3.40489e-4 0; 0 0 3.40489e-4;
   .982975 0 0; 0 .982975 0; 0 0 .982975];
K4=[zeros(6) K4];

Y=gpssig(Y);
n=4;
ys=7;
yf=9;
[xp,xm]=ssnavest(A,C,x0,K1,K2,K3,K4,Y,n,ys,yf);
t=1:ln;
rms=sqrt(sum((Yt(:,1:6)-xp).^2)/ln)
```

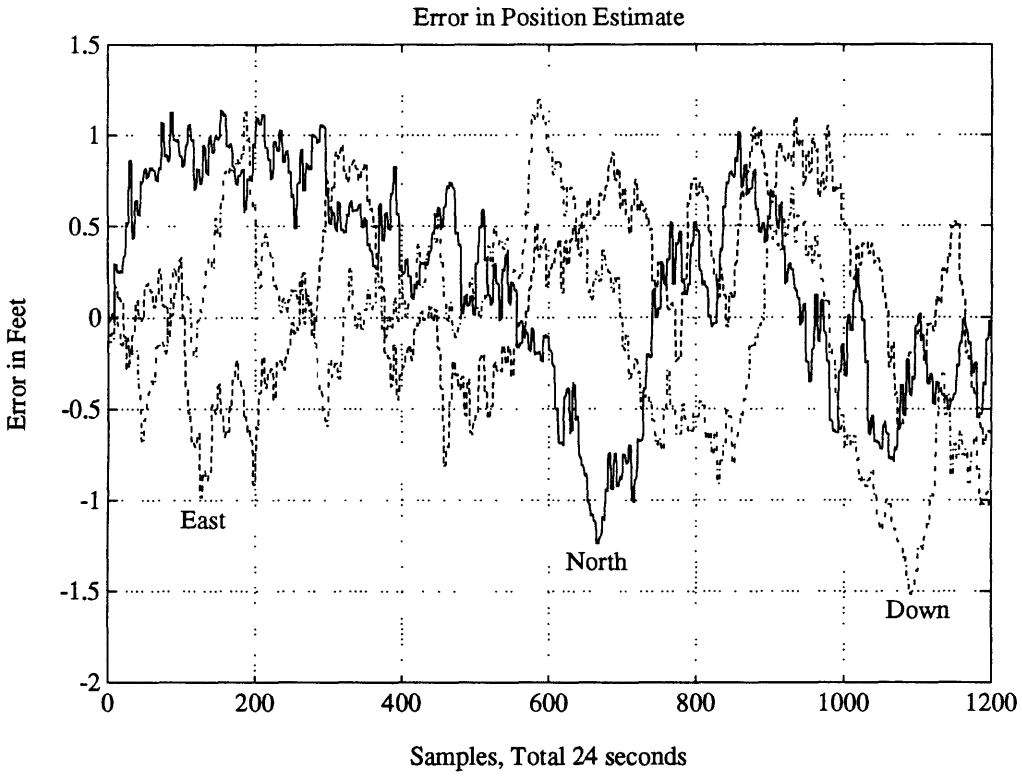

Figure 32 Position Errors for Constant Velocity Test

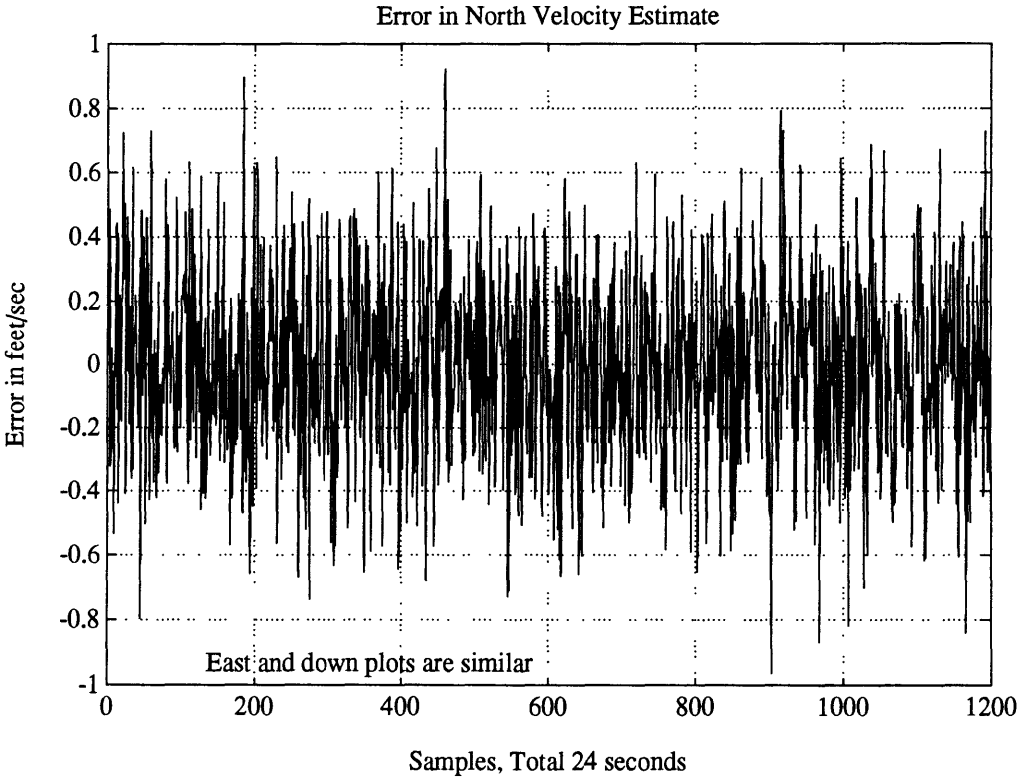

Figure 33 Velocity Error for Constant Velocity Test

11.1.3 Single Velocity Change Test

```
%Kalm7
%Matlab script to simulate A-10 filter using 200 time steps
% with a down velocity change
ln=200;
A=eye(6)+diag([.02 .02 .02],3);
B=[zeros(3);eye(3)];
C=[eye(6);zeros(3) eye(3)];
D=zeros(9,3);
U=zeros(ln,3);
U(51:150,3)=ones(100,1)*3;
x0=[0 0 0 400 200 0];
[Y,x]=dlsim(A,B,C,D,U,x0);
Yt=Y;
rand('normal')
Y(:,1:3)=Y(:,1:3)+rand(ln,3)*7.4;
Y(:,4:6)=Y(:,4:6)+rand(ln,3)*.74;
Y(:,7:9)=Y(:,7:9)+rand(ln,3)*.3;
G=eye(6);
Q=diag([.005 .005 .005 5 5 5]);
R=diag([54.8 54.8 54.8 .548 .548 .548 .0881 .0881 .0881]);

K1=[1.89861e-2 0 0 4.63737e-5 0 0 2.88454e-4 0 0;
   0 1.89861e-2 0 0 4.63737e-5 0 0 2.88454e-4 0;
   0 0 1.89861e-2 0 0 4.63737e-5 0 0 2.88454e-4;
   4.63737e-7 0 0 .136464 0 0 .848834 0 0;
   0 4.63737e-7 0 0 .136464 0 0 .848834 0;
   0 0 4.63737e-7 0 0 .136464 0 0 .848834];
K2=[2.94614e-4 0 0; 0 2.94614e-4 0; 0 0 2.94614e-4;
   .982936 0 0; 0 .982936 0; 0 0 .982936];
K2=[zeros(6) K2];
K3=[3.39709e-4 0 0; 0 3.39709e-4 0; 0 0 3.39709e-4;
   .982975 0 0; 0 .982975 0; 0 0 .982975];
K3=[zeros(6) K3];
K4=[3.40489e-4 0 0; 0 3.40489e-4 0; 0 0 3.40489e-4;
   .982975 0 0; 0 .982975 0; 0 0 .982975];
K4=[zeros(6) K4];

Y=gpssig(Y);
n=4;
ys=7;
yf=9;
[xp,xm]=ssnavest(A,C,x0,K1,K2,K3,K4,Y,n,ys,yf);
t=1:ln;
rms=sqrt(sum((Yt(:,1:6)-xp).^2)/ln)
```

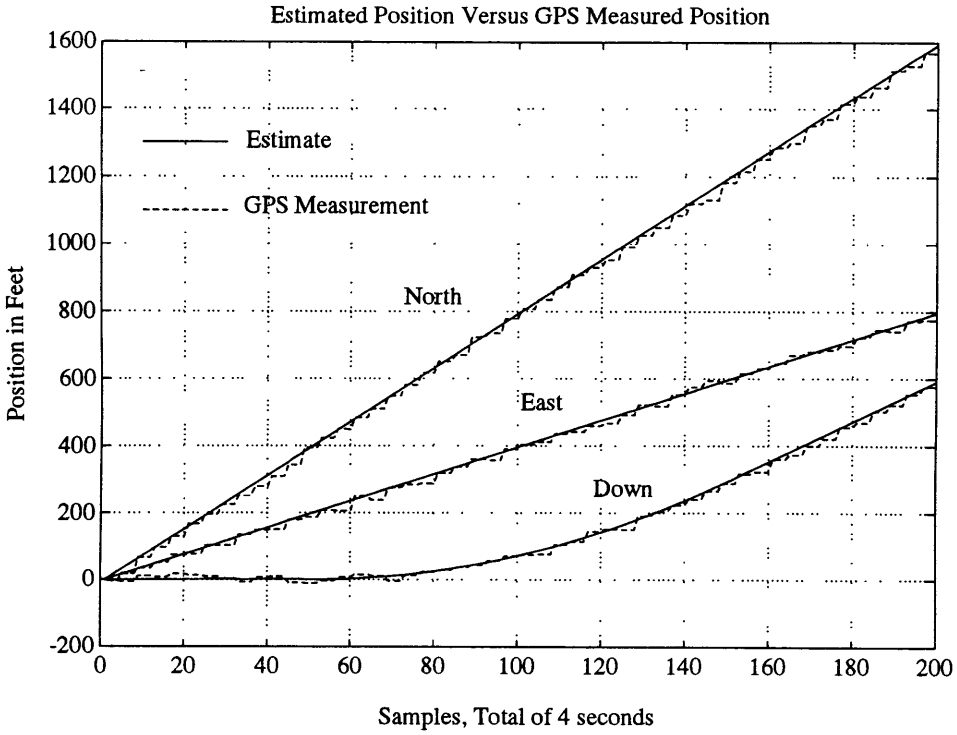

Figure 34 Position Estimates Vs GPS Postion Measurement

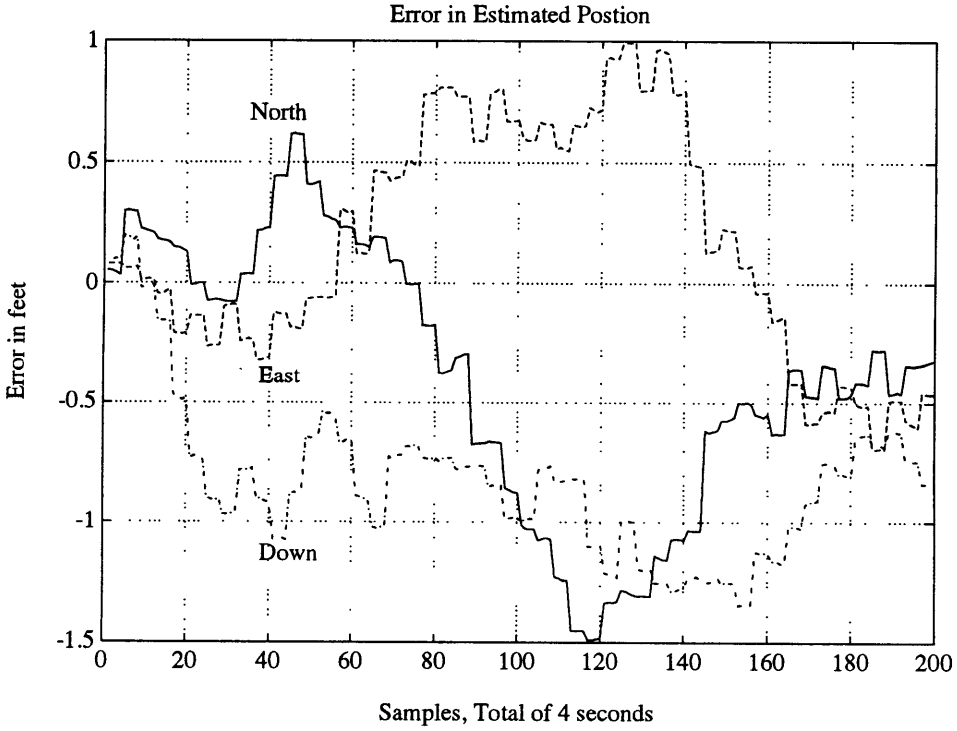

Figure 35 Error in Position Estimates

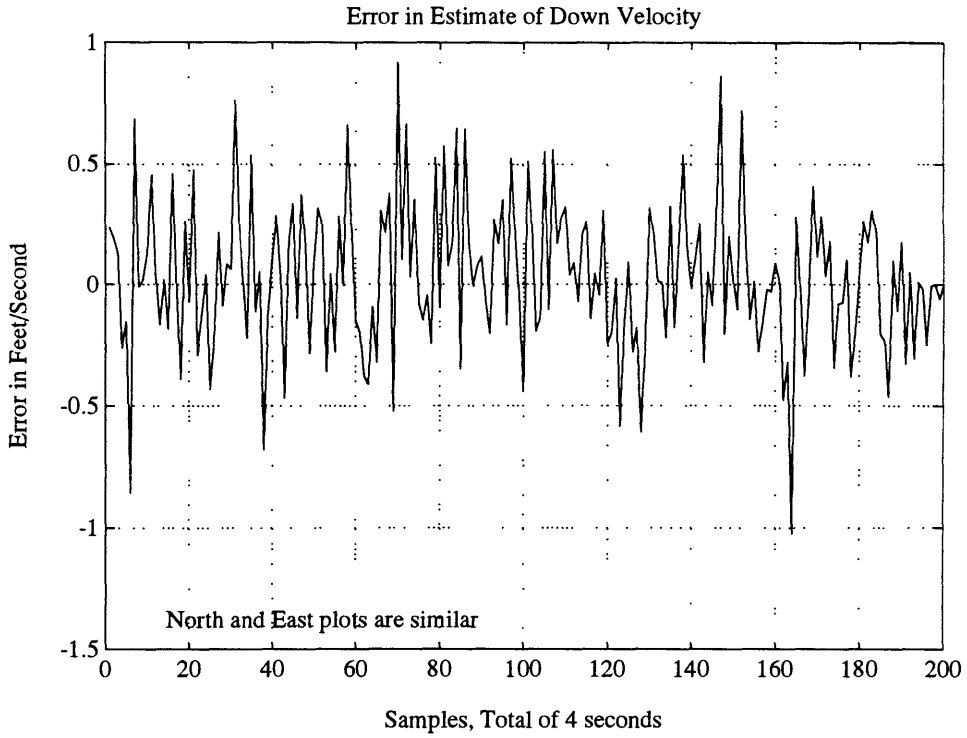

Figure 36 Error in Velocity Estimates

11.1.4 Hard Turn Test

```
%Kalm8
%Matlab script to simulate A-10 filter using 200 time steps
% simulating a 90 degree high-rate turn
ln=200;
A=eye(6)+diag([.02 .02 .02],3);
B=[zeros(3);eye(3)];
C=[eye(6);zeros(3) eye(3)];
D=zeros(9,3);
U=zeros(ln,3);
U(1:50,1:2)=ones(50,1)*[-1,4];
U(51:100,1:2)=ones(50,1)*[-2,3];
U(101:150,1:2)=ones(50,1)*[-3,2];
U(151:200,1:2)=ones(50,1)*[-4,1];
x0=[0 0 0 500 00 0];
[Y,x]=dlsim(A,B,C,D,U,x0);
Yt=Y;
rand('normal')
Y(:,1:3)=Y(:,1:3)+rand(ln,3)*7.4;
Y(:,4:6)=Y(:,4:6)+rand(ln,3)*.74;
Y(:,7:9)=Y(:,7:9)+rand(ln,3)*.3;
G=eye(6);
Q=diag([.005 .005 .005 5 5 5]);
R=diag([54.8 54.8 54.8 .548 .548 .548 .0881 .0881 .0881]);
```

```
K1=[1.89861e-2 0 0 4.63737e-5 0 0 2.88454e-4 0 0;
   0 1.89861e-2 0 0 4.63737e-5 0 0 2.88454e-4 0;
   0 0 1.89861e-2 0 0 4.63737e-5 0 0 2.88454e-4;
   4.63737e-7 0 0 .136464 0 0 .848834 0 0;
   0 4.63737e-7 0 0 .136464 0 0 .848834 0;
   0 0 4.63737e-7 0 0 .136464 0 0 .848834];
K2=[2.94614e-4 0 0; 0 2.94614e-4 0; 0 0 2.94614e-4;
   .982936 0 0; 0 .982936 0; 0 0 .982936];
K2=[zeros(6) K2];
K3=[3.39709e-4 0 0; 0 3.39709e-4 0; 0 0 3.39709e-4;
   .982975 0 0; 0 .982975 0; 0 0 .982975];
K3=[zeros(6) K3];
K4=[3.40489e-4 0 0; 0 3.40489e-4 0; 0 0 3.40489e-4;
   .982975 0 0; 0 .982975 0; 0 0 .982975];
K4=[zeros(6) K4];

Y=gpssig(Y);
n=4;
ys=7;
yf=9;
[xp,xm]=ssnavest(A,C,x0,K1,K2,K3,K4,Y,n,ys,yf);
t=1:ln;
rms=sqrt(sum((Yt(:,1:6)-xp).^2)/ln)
```

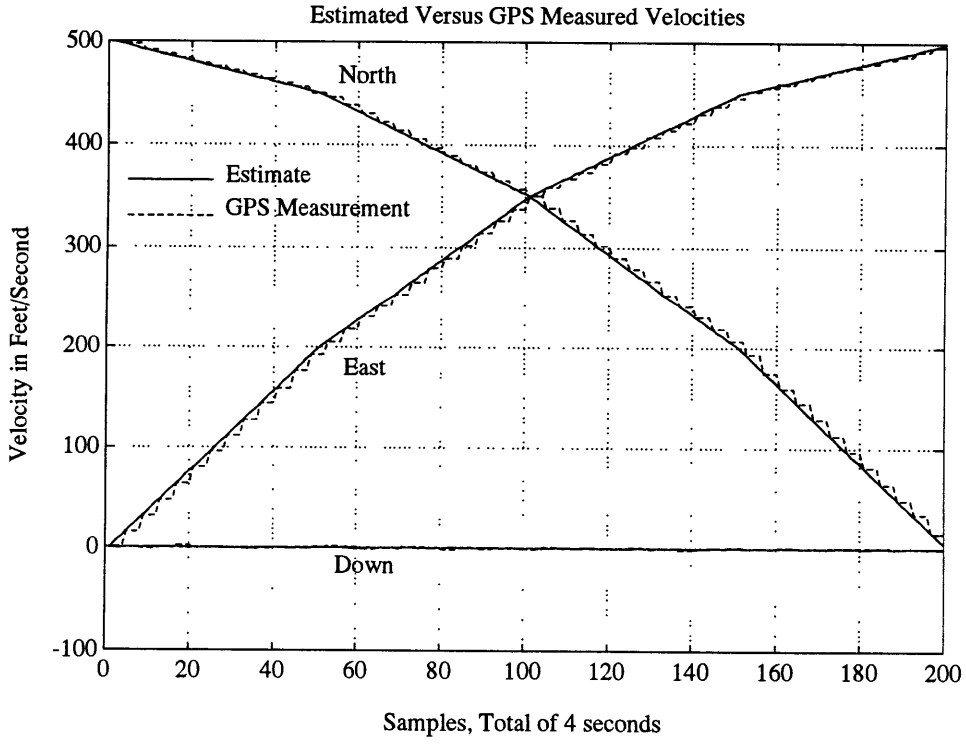

Figure 37 Estimated Versus Measured Velocities

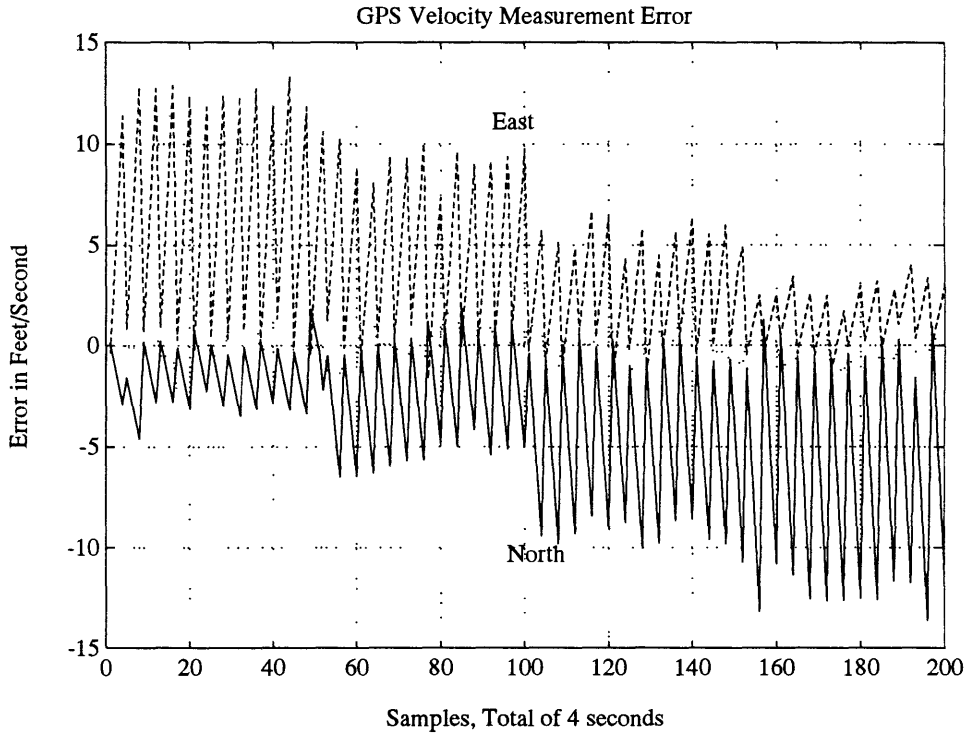

Figure 38 GPS Velocity Measurement Errors

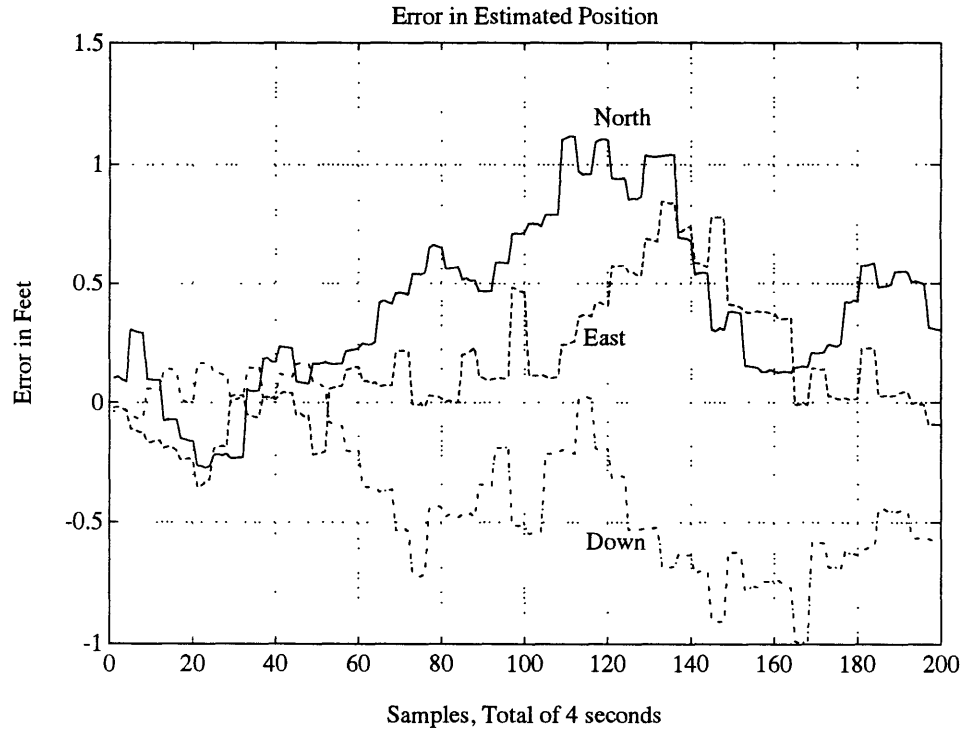

Figure 39 Error in Position Estimate

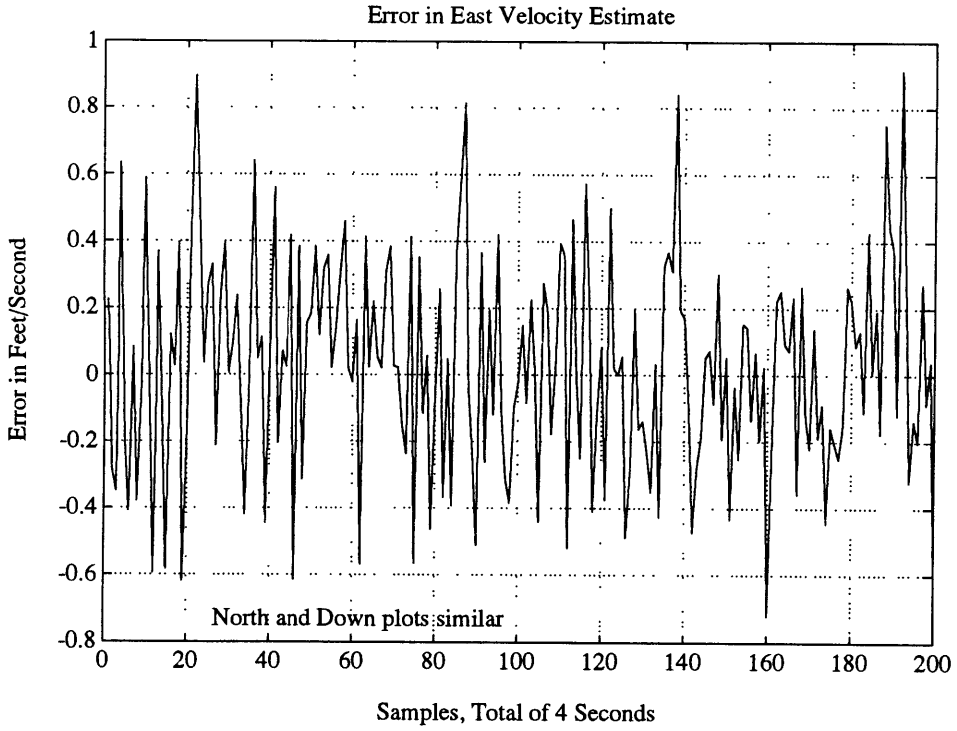

Figure 40 Error in Velocity Estimate

11.2 Error Estimator Performance Test

```
% Error3
% produces variable bank trajectory and simulates error estimator

% Initialize variables
% ln=180;
 t=1:ln;
 QC=diag([2.4e-8 2.4e-8 2.4e-8 1e-12 1e-12 3.2e-4 .5 3.7 3.7 3.7]);
 RC=diag([2 2 2]);
 P=diag([4e-8,4e-8,4e-8,4e-8,4e-8,1e-4,10,100,100,100]);

% Generate trajectory values
 g=ones(ln,1)*1.5;
 rol=ones(ln,1).*sin(t'*2*pi/60)/15;
 ss=zeros(ln,1);
 acc=ones(ln,1);
 vi=[0;320;0];
 bani=-0.25;
 magv=.1;

% Call hcref to get HARS/CADC reference values
 [hed,pit,ban,vel,aoa]=hcref(g,rol,ss,acc,vi,bani,magv);
 echo on
```

```
% HARS/CADC reference trajectory complete
 echo off

% Generate initial values for corrections
 yc=[.00,.00,.00,.0006,.0005,1.05,4.7,0,0,0]';

% Generate error values
 rand('normal')
 ycr=[.009,.005,-.0049,.0006,.0005,1.05,4.7];
 ycw=ones(ln,6);
 % Generate Winds
 ycw=ones(ln,3);
 ycw(:,1)=sin(t'*2*pi/ln*2)*5;
 ycw(:,2)=sin(t'*2*pi/ln*1.5)*7;
 ycw(:,3)=sin(t'*2*pi/ln)*(-3);
 % Generate misalignment random walk
 ycw(1,4:6)=[0 0 0];
 for x=2:ln
   ycw(x,4:6)=ycw(x-1,4:6)+rand(1,3)*5e-6;
 end

% Call gpsvel to generate GPS velocity reference
 [Vt]=gpsvel(vel,hed,pit,ban,aoa,ss,ycr,ycw,magv);
 echo on
 % GPS reference velocity complete
 echo off

% Add noise to HARS reference signals
  vel=vel+rand(ln,1)*1;
  hed=hed+rand(ln,1)*.0005;
  pit=pit+rand(ln,1)*.0005;
  ban=ban+rand(ln,1)*.0005;
  aoa=aoa+rand(ln,1)*.0005;
  ss=ss+rand(ln,1)*.0005;

% Add noise to GPS velocities
  Vg=Vt;
  Vg=Vt+rand(ln,3);

% Call error estimator
 [XC,PS,RTS,YC,VH]=errest(vel,hed,pit,ban,aoa,ss,Vg,magv,QC,RC,P,yc);
 PD=varplot(PS);
% Plot Results
 plotcorr(YC,ycr,ycw)
```

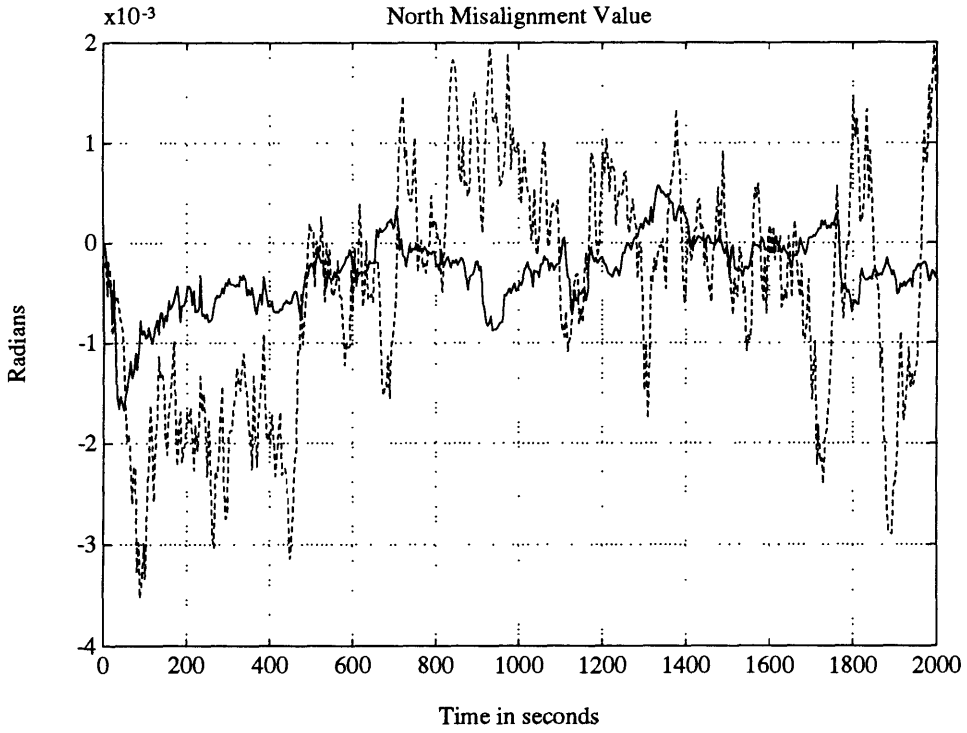

Figure 41 North HARS Misalignment Estimate

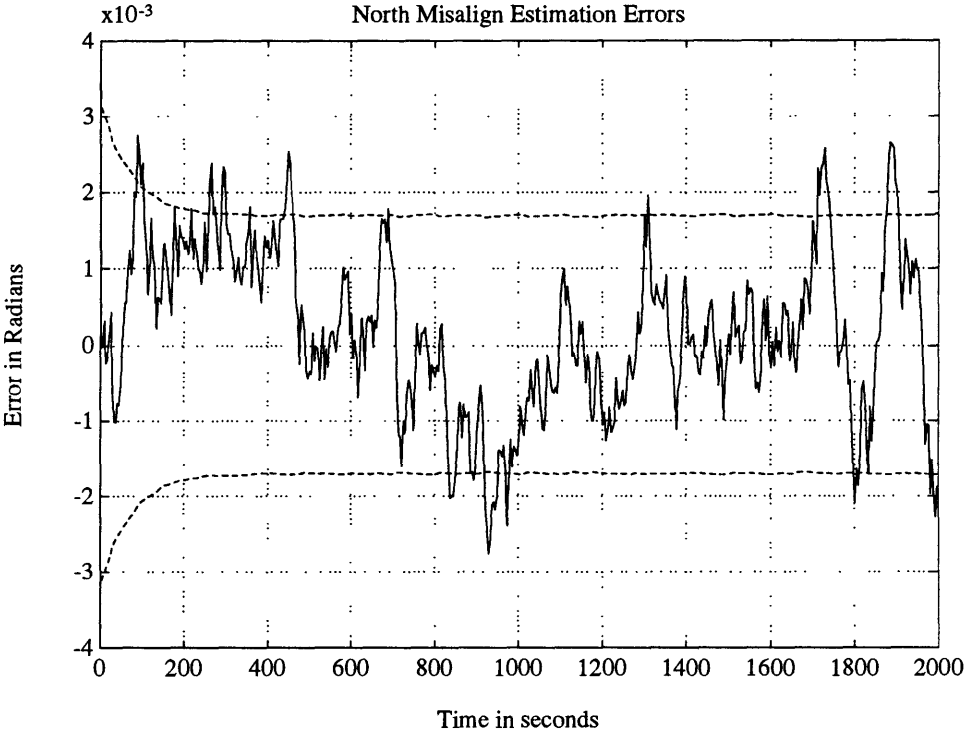

Figure 42 Error in HARS North Misalignment Estimate

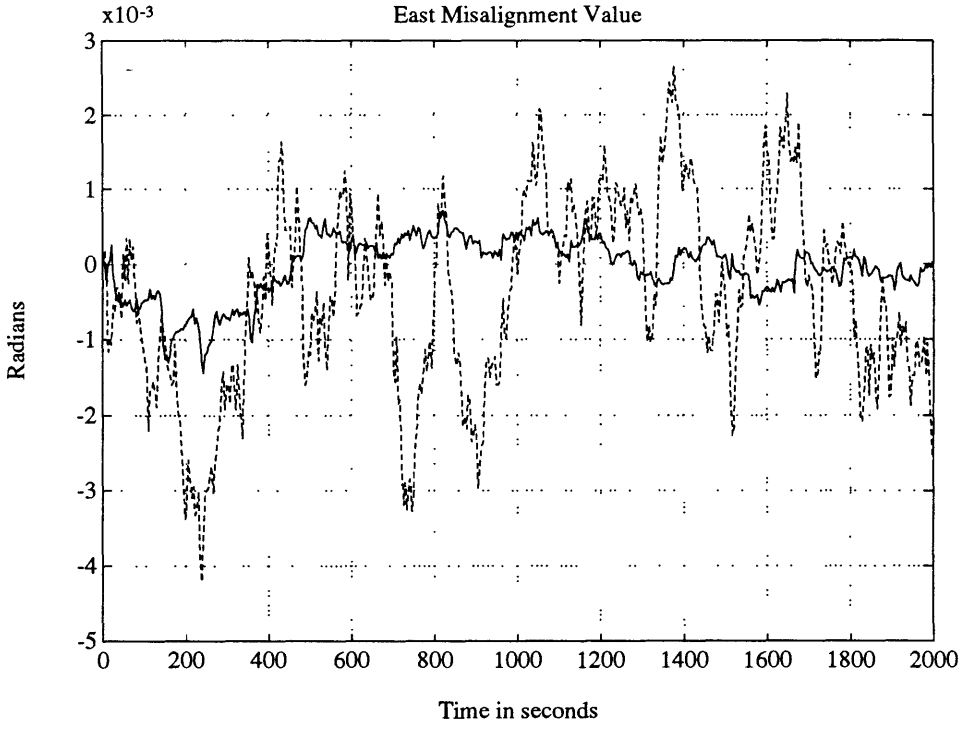

Figure 43 East HARS Misalignment Estimate

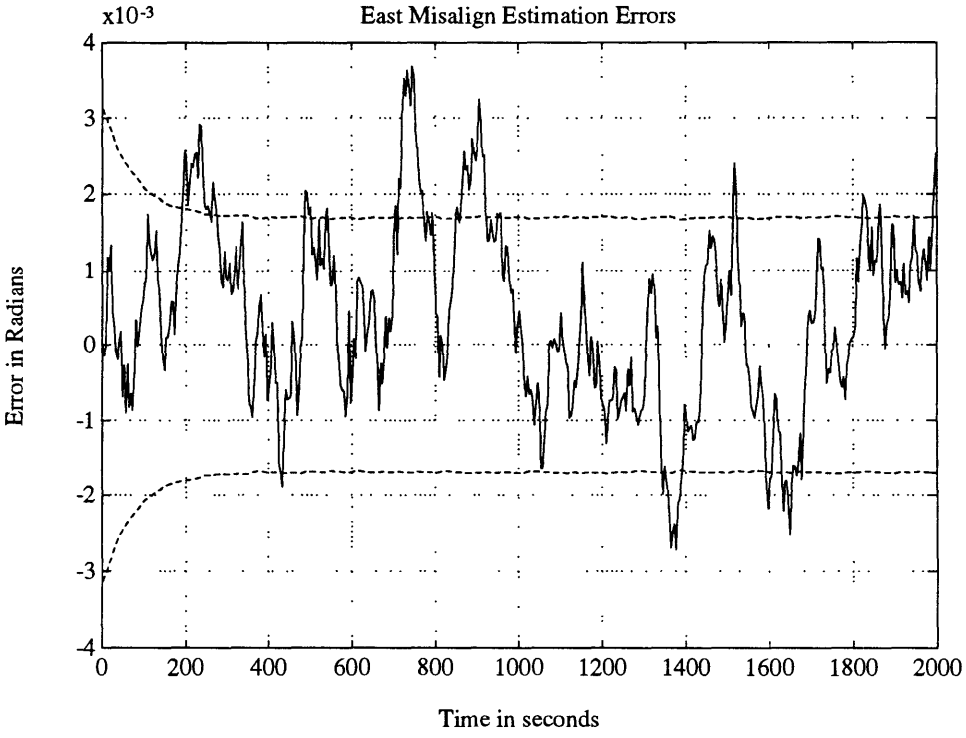

Figure 44 Error in HARS East Misalignment Estimate

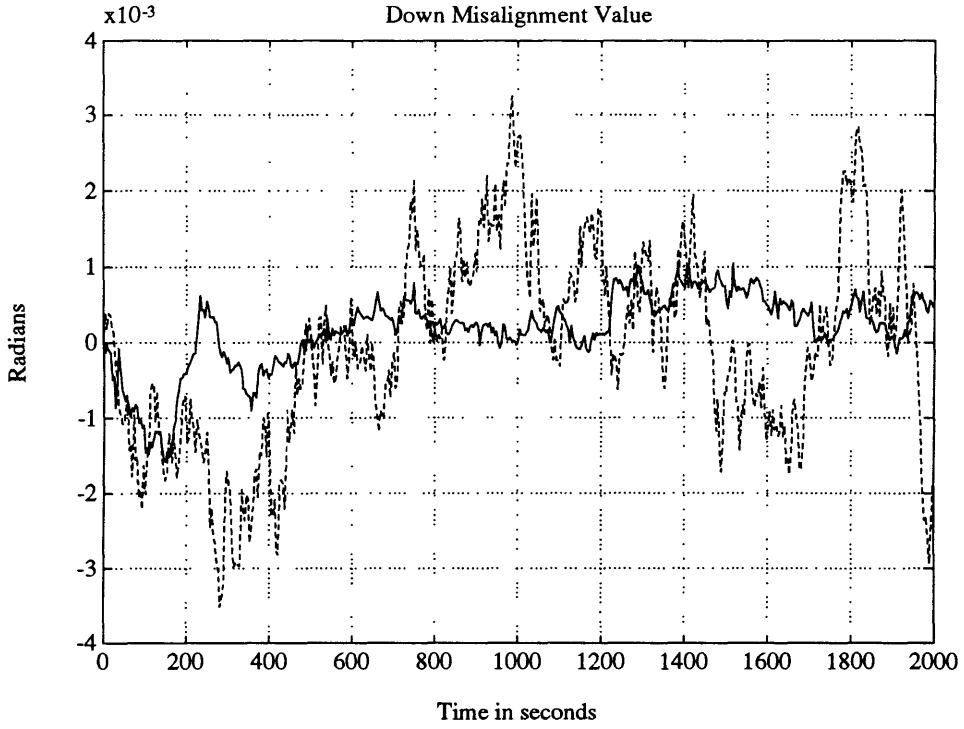

Figure 45 Down HARS Misalignment Estimate

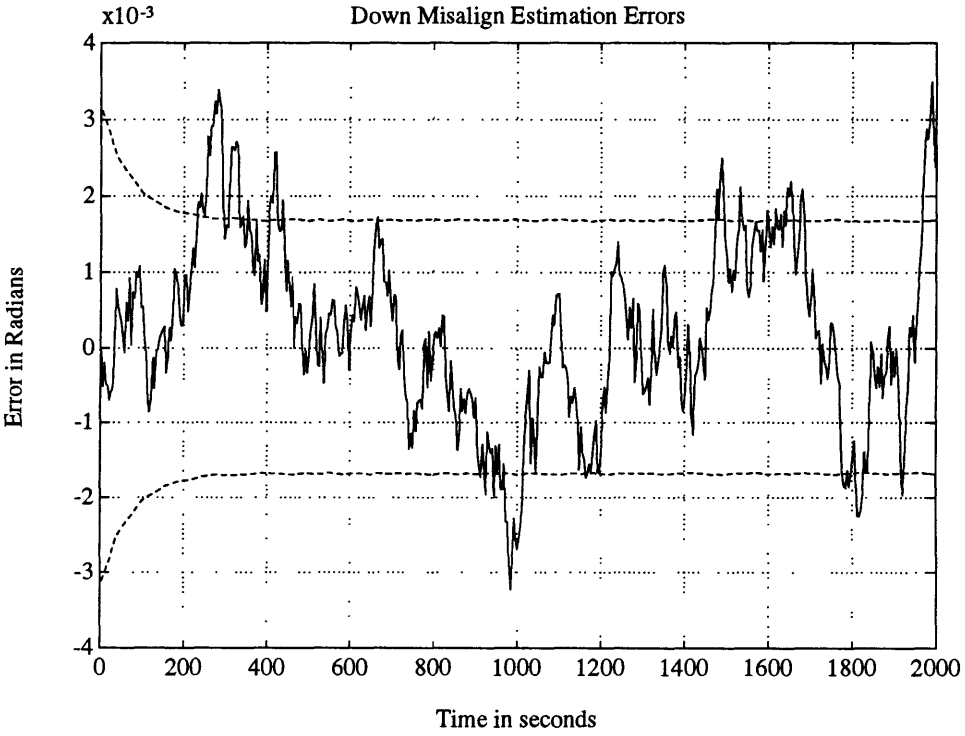

Figure 46 Error in HARS Down Misalignment Estimate

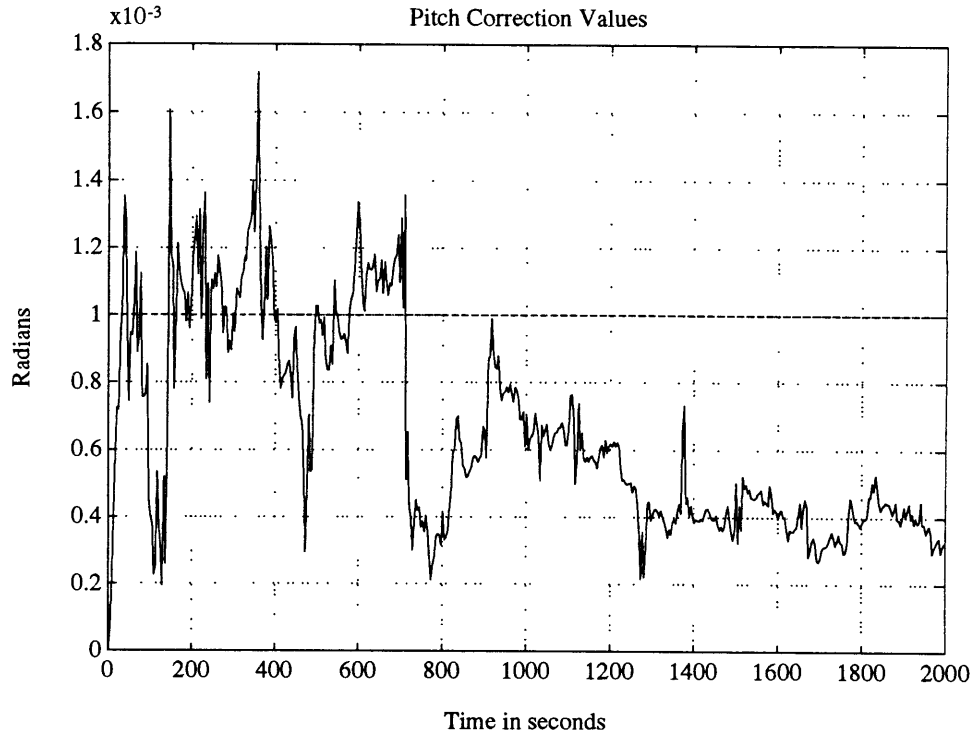

Figure 47 Pitch Correction Estimate

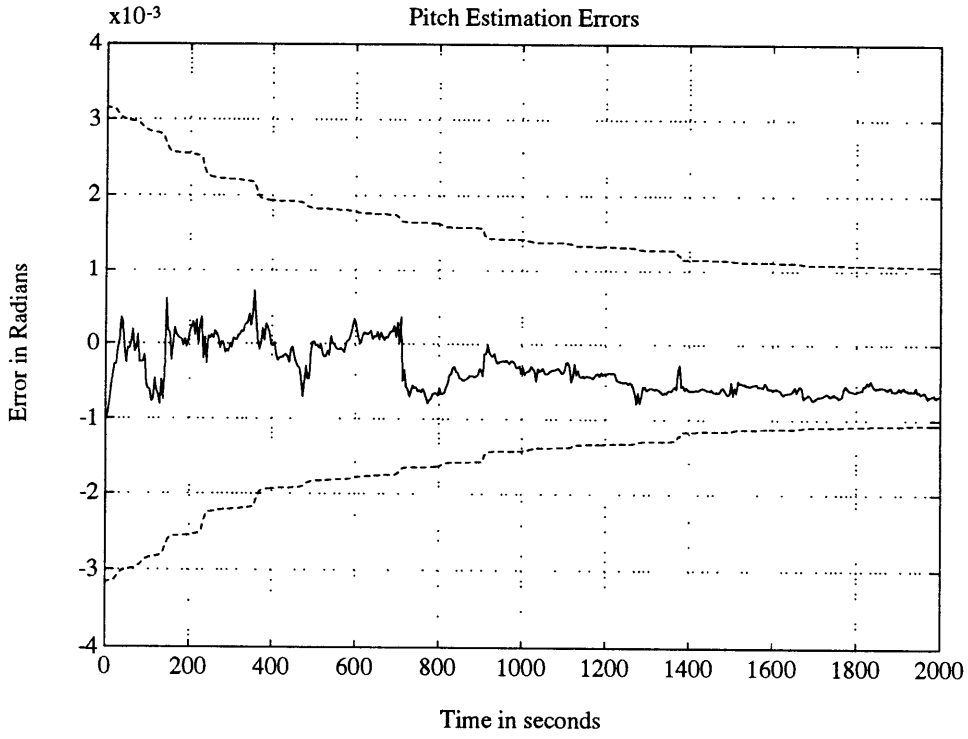

Figure 48 Error in Pitch Correction Estimate

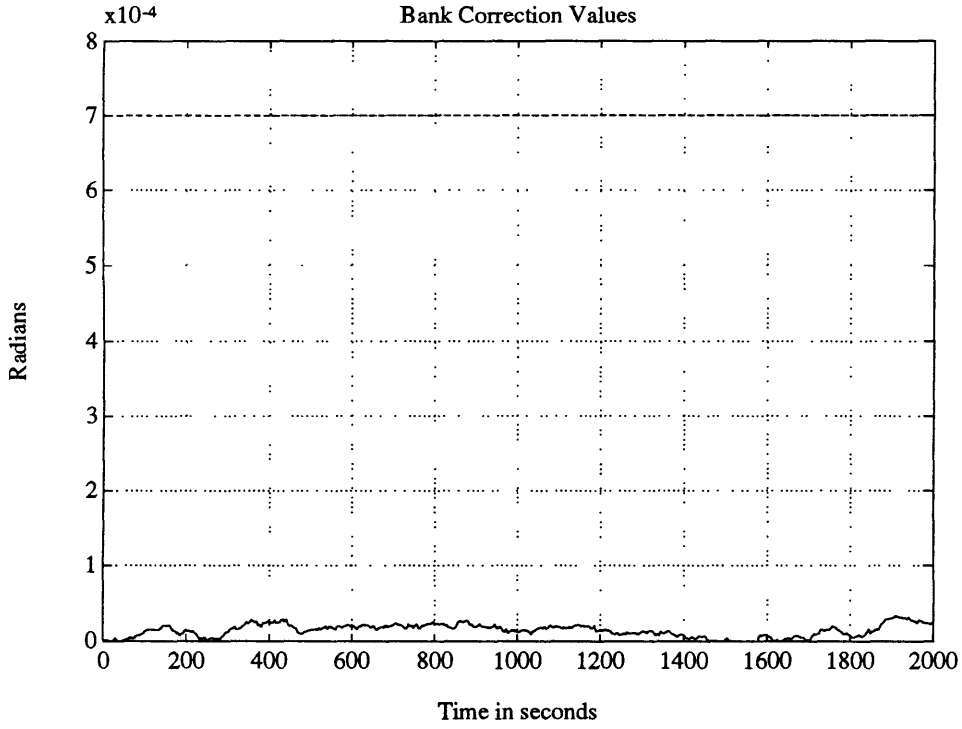

Figure 49 Bank Correction Estimate

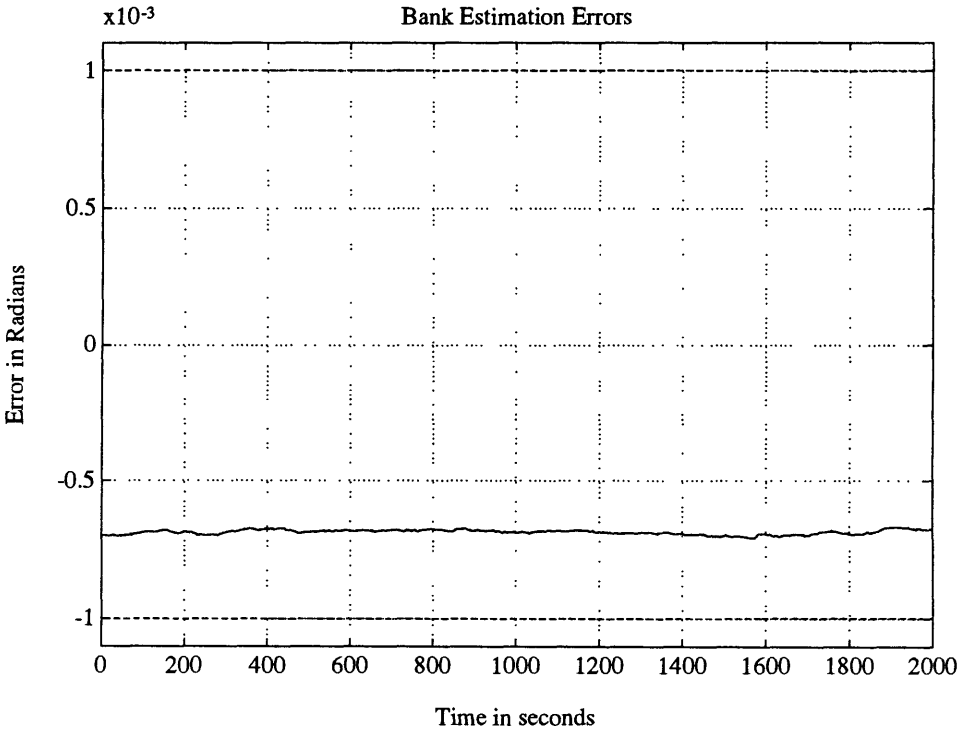

Figure 50 Error in Bank Correction Estimate

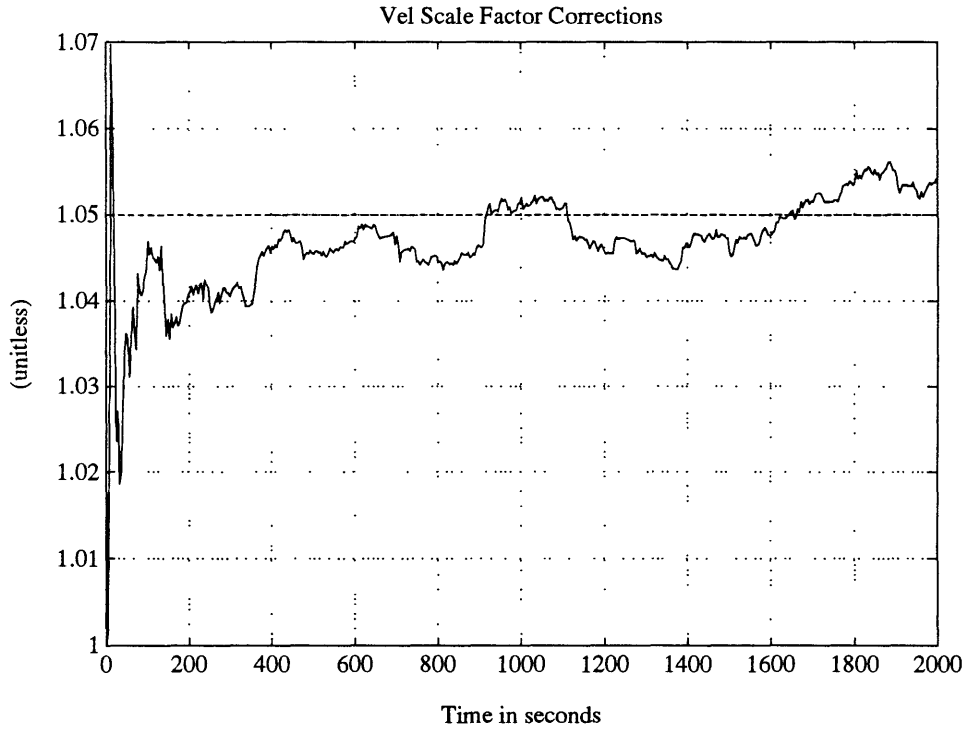

Figure 51 Velocity Scale Factor Estimate

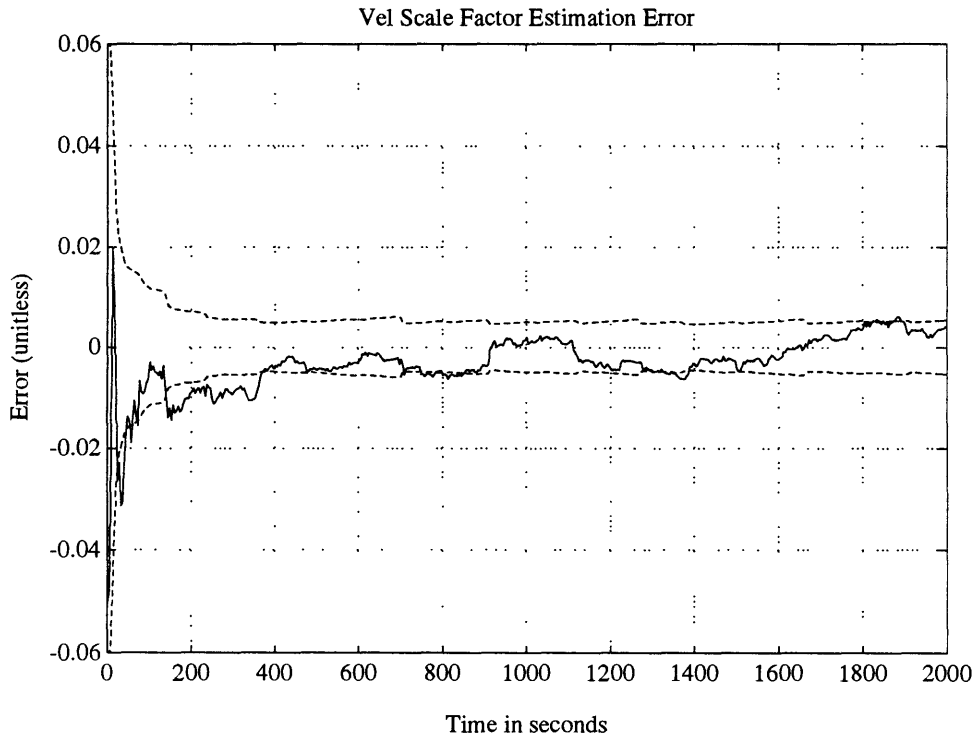

Figure 52 Error in Velocity Scale Factor Estimate

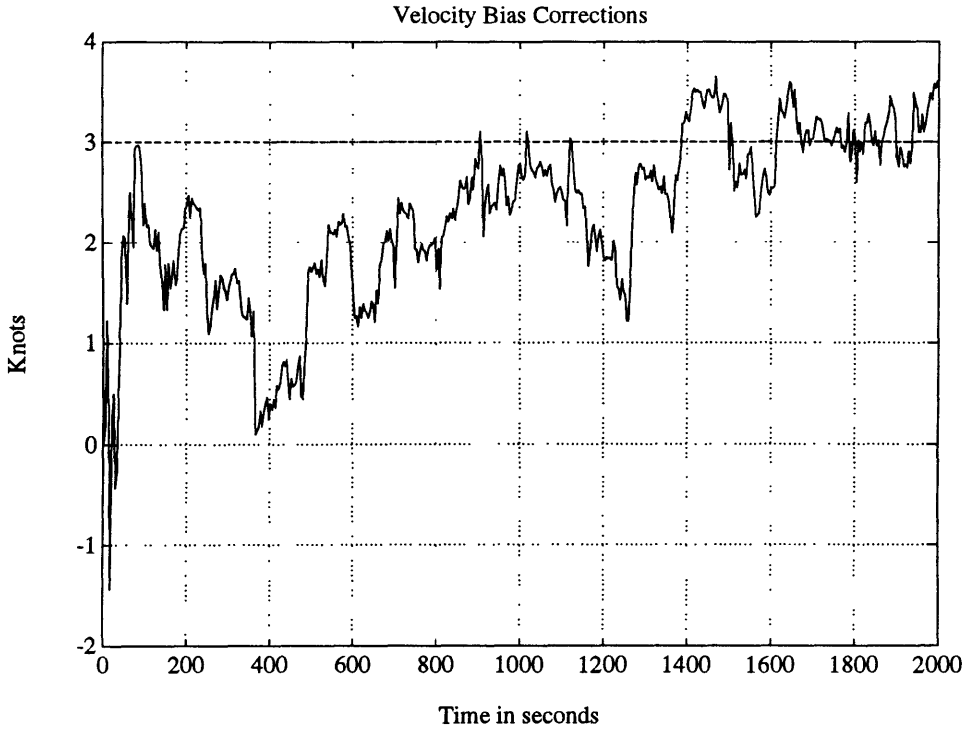

Figure 53 Velocity Bias Estimate

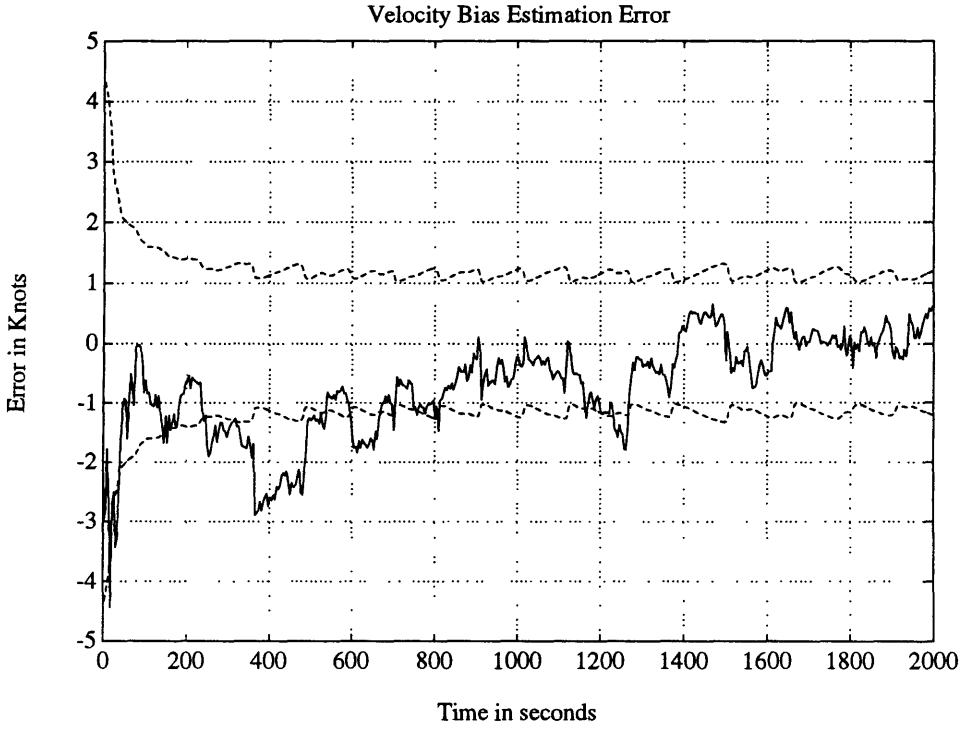

Figure 54 Error in Velocity Bias Estimate

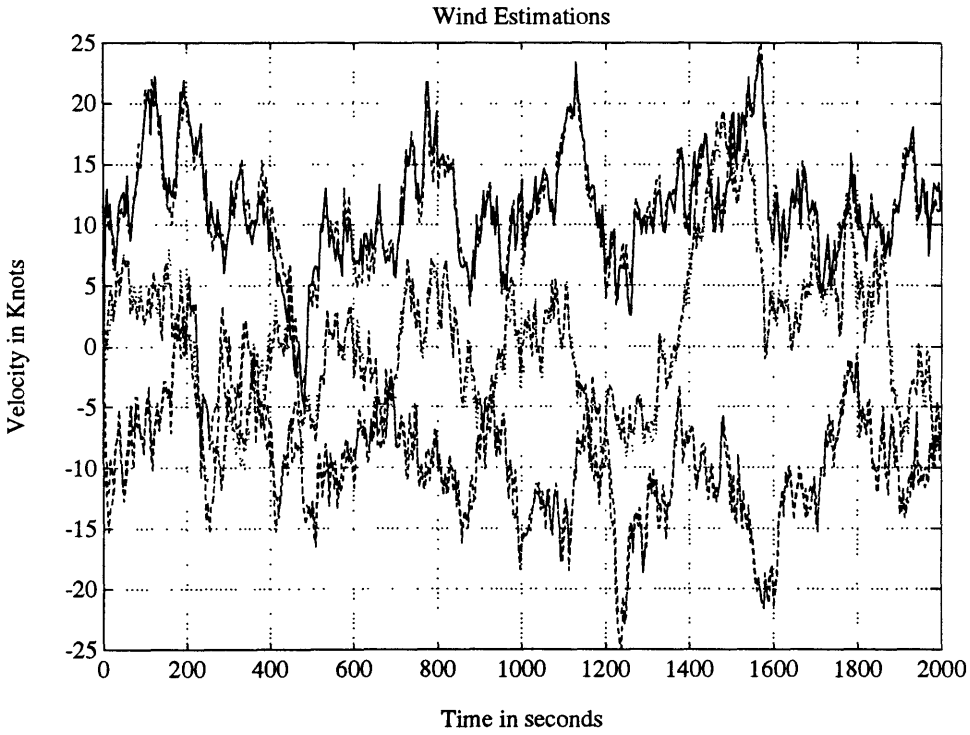

Figure 55 Wind Estimates

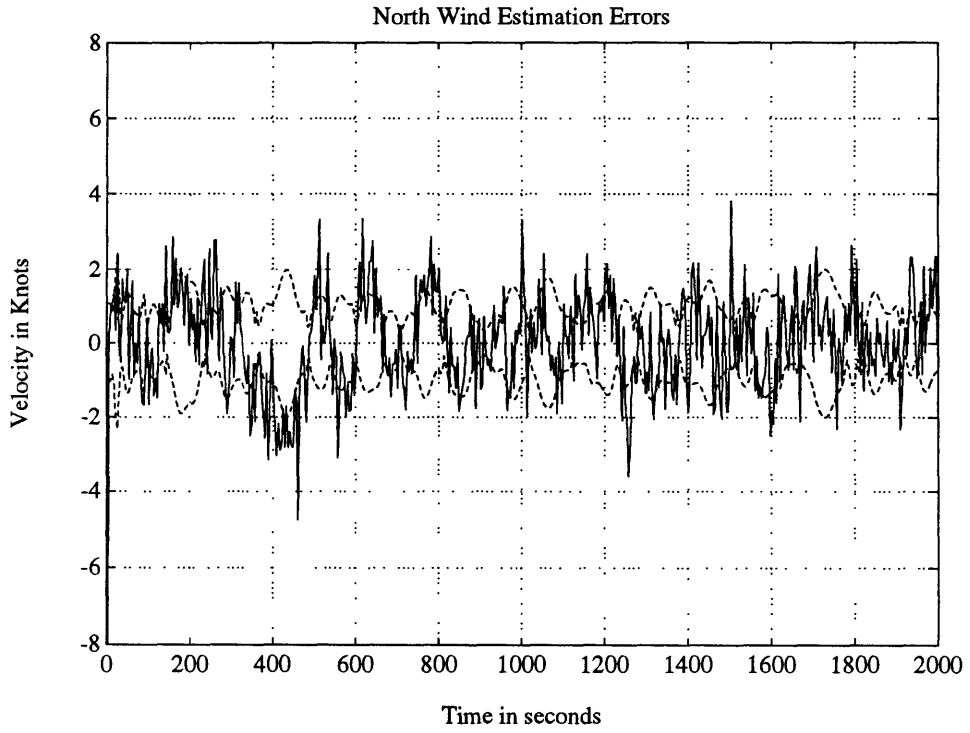

Figure 56 North Wind Estimation Error

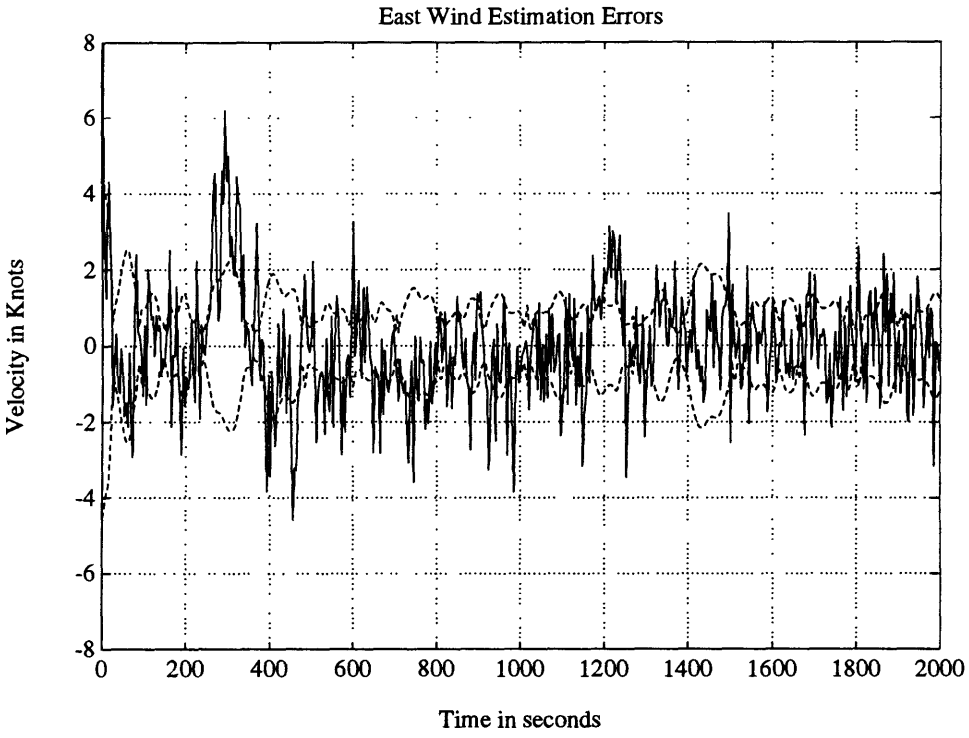

Figure 57 East Wind Estimation Error

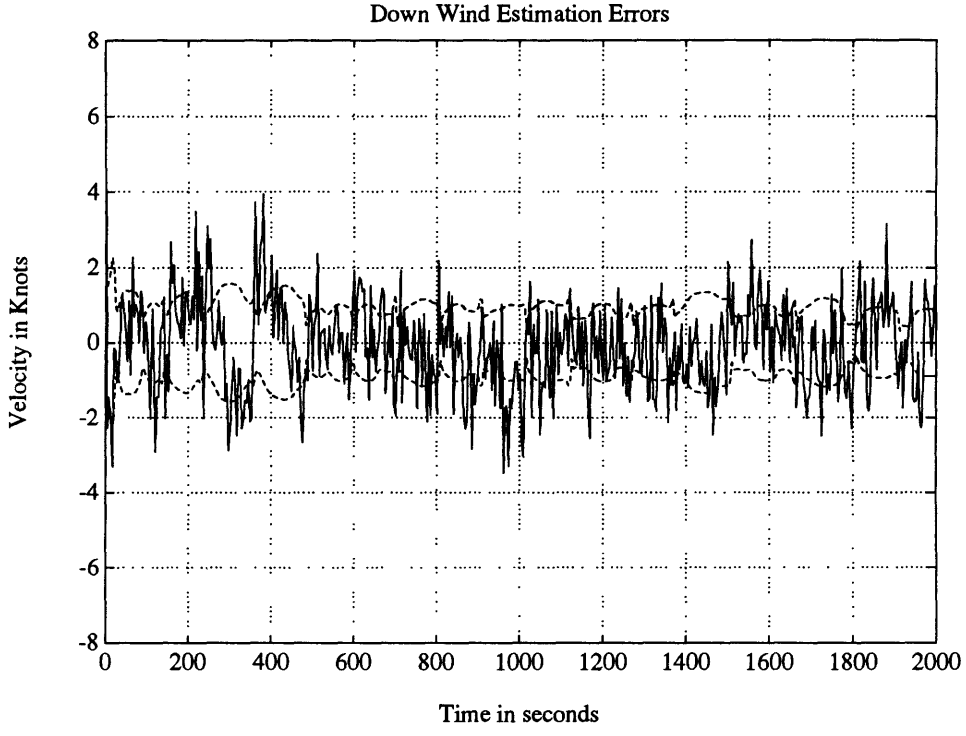

Figure 58 Down Wind Estimation Error

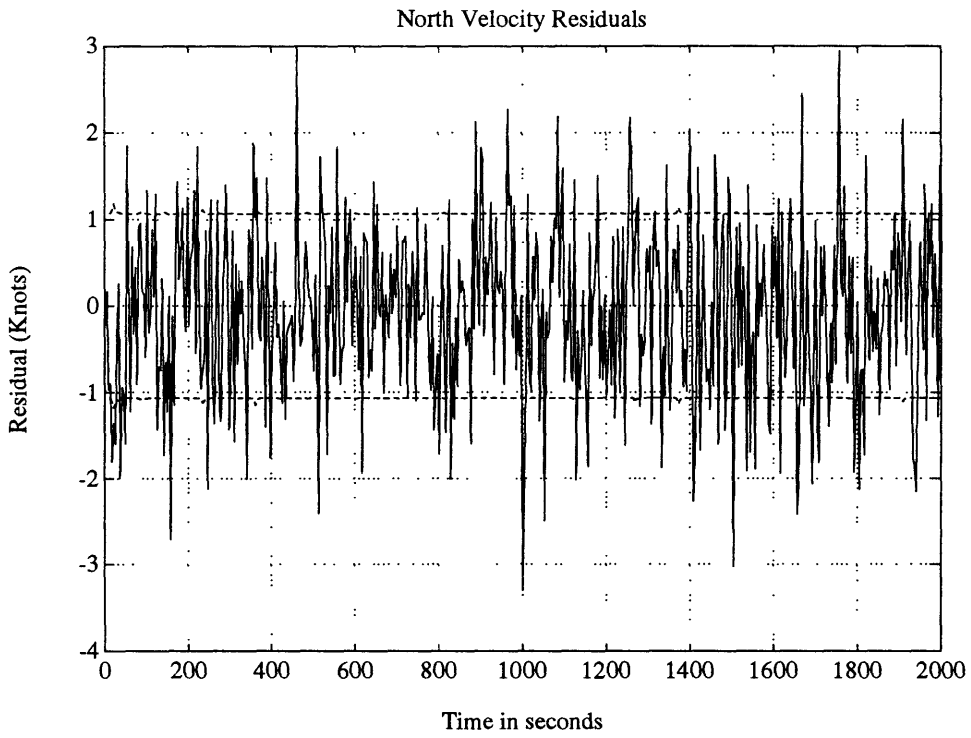

Figure 59 North Velocity Measurement Residual

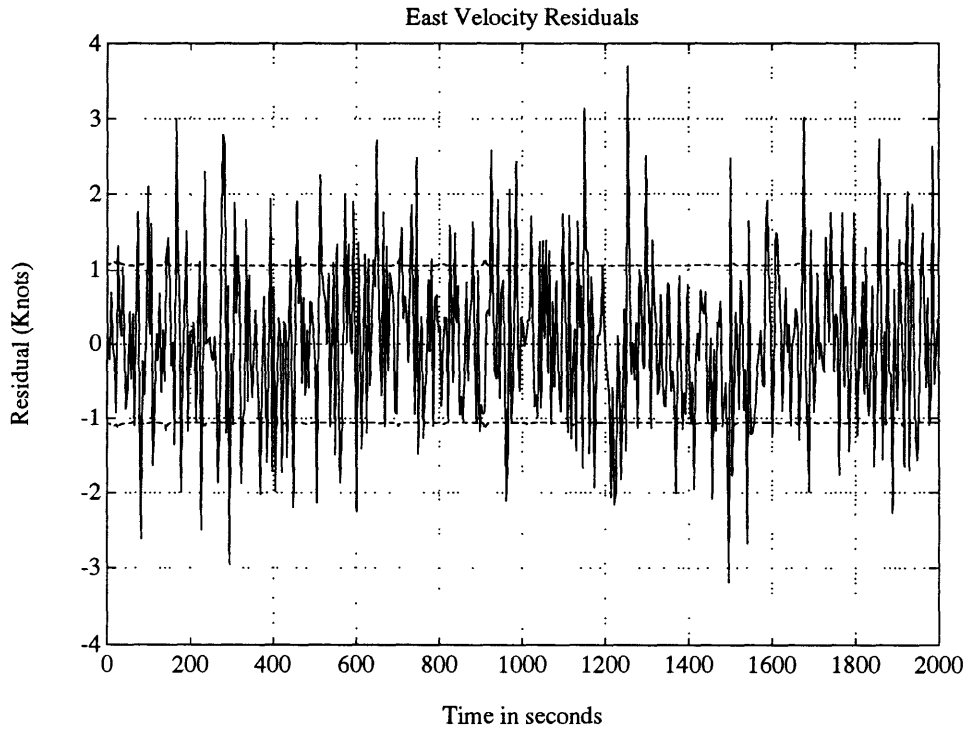

Figure 60 East Velocity Measurement Residual

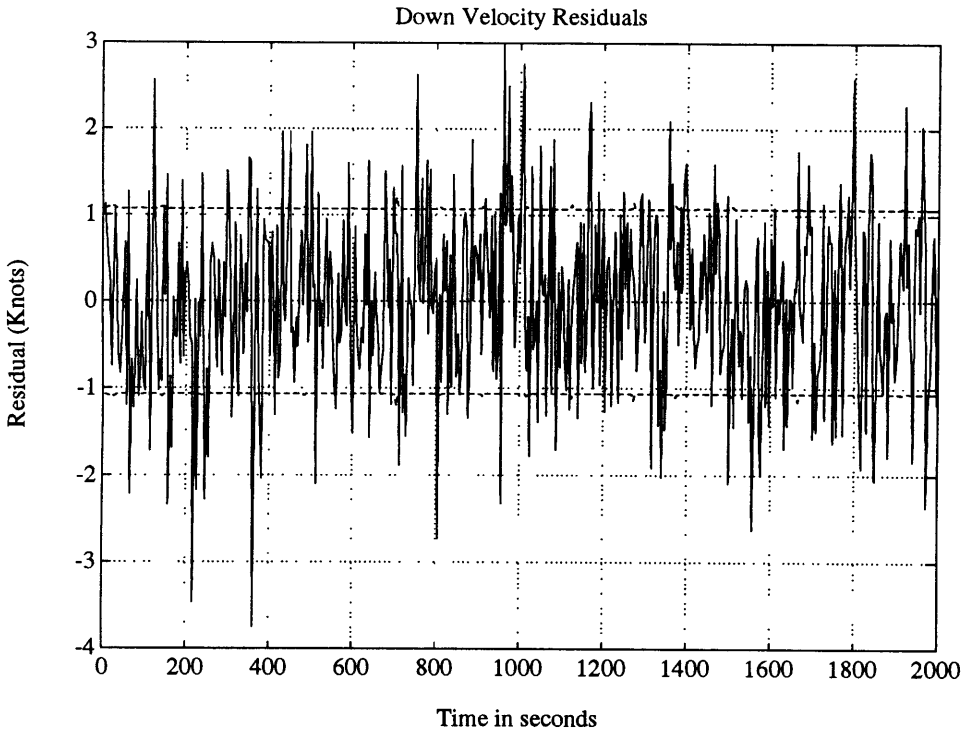

Figure 61 Down Velocity Measurement Residual

12. References

[1] B. Etkin, Dynamics of Atmospheric Flight, John Wiley & Sons, Inc., New York, 1972.

[2] A. S. Willsky, Recursive Estimation Supplementary Notes, Massachusetts Institute of Technology, Cambridge, MA, 1994.

[3] Flight Manual, USAF Series A-10A/OA-10A Aircraft, T.O. 1A-10-1-1, Change 8, Sacremento ALC/TILBE, McClellan AFB, CA, 10 May 1993.

[4] J. Donna, Model of the HARS, Intralab Memorandum, Charles Stark Draper Laboratory, Cambridge, MA, May 24, 1990.