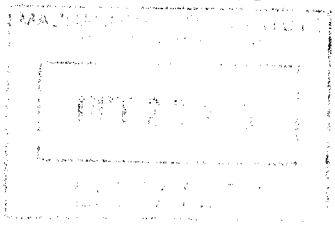
Embedded Avionics with Kalman State Estimation for a Novel Micro-Scale Unmanned Aerial Vehicle



by

Theodore Tzanetos

B.S. Electrical Engineering and Computer Science
Massachusetts Institute of Technology, 2012

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF ENGINEERING IN ELECTRICAL ENGINEERING AND COMPUTER SCIENCE
AT THE
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

JUNE 2013

This work is sponsored by the Department of the Air Force under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions and recommendations are those of the author and are not necessarily endorsed by the United States Government.

Signature of Author:_____
Department of Electrical Engineering and Computer Science
May 24, 2013

Certified by:_____
James K. Roberge
Professor of Electrical Engineering
Thesis Supervisor

Certified by:_____
Ryan D. Eubank
MIT Lincoln Laboratory Technical Staff
Thesis Co-Supervisor

Accepted by:_____
Dennis M. Freeman
Professor of Electrical Engineering
Chairman, Masters of Engineering Thesis Committee

Embedded Avionics with Kalman State Estimation for a Novel Micro-Scale Unmanned Aerial Vehicle

by

Theodore Tzanetos

Submitted to the Department of Electrical Engineering and Computer Science

on May 24, 2013 in Partial Fulfillment of the Requirements for the Degree of Master of Engineering in Electrical Engineering and Computer Science

ABSTRACT

An inertial navigation system leveraging Kalman estimation techniques and quaternion dynamics is developed for deployment to a micro-scale unmanned aerial vehicle (UAV). The capabilities, limitations, and requirements of existing navigation solutions motivate the need for an integrated solution that can be readily applied to small embedded systems and still provide reasonably accurate results. Methods to calibrate and compensate systemic inaccuracies in microelectromechanical systems (MEMS) sensors, commonly used in micro-scale UAV applications, are also developed.

The problems associated with attitude determination and system localization are analyzed in isolation with incremental simulation and field testing. Performance is evaluated against commercially available inertial navigation system solutions. The result is a capable navigation system that, by its structure, trades a small measure of accuracy in order to be easily adapted to the embedded computing constraints of unmanned vehicles in the micro-scale.

Thesis Supervisor: James K. Roberge
Title: Professor of Electrical Engineering

Acknowledgments

First, I would like to thank all my friends and family for their support during my research. This includes my academic adviser Professor Christopher J. Terman who has been an invaluable guide during my time at MIT.

I would like to thank my thesis supervisors Prof. James Roberge and Dr. Ryan Eubank for their time and guidance throughout my research. I am deeply indebted to the MIT Lincoln Laboratory staff who helped me achieve my success, specifically Josh Manore, Sam Stambler, Kenneth Chadwick, Andrew Kopeikin, Matthew Lowe, and Ronald Hoffeld. Special thanks is given to Mark Cutler of the MIT Aerospace Controls Lab for his help. Finally, I would also like to thank Tony Tao and Professor Hansman for their excellent engineering which laid the foundation for this project.

Contents

List of Figures

List of Tables

This page is intentionally left blank.

This page is intentionally left blank.

Chapter 1

Introduction

1.1 Motivation

In the past two decades, unmanned aerial vehicles, more generally known as Unmanned Aircraft Systems (UAS), have experienced nearly exponential growth as they continue to become a more important tool for private, commercial, and military users. According to a 2012 forecast by defense market analysis firm Teal Group Co., the global market for UAS research and applications will nearly double from current procurement expenditures of about $6.6 billion annually to $11.4 billion annually, making UAS an $89 billion market over the next ten years [13]. The Unites States alone has invested over $11.8 billion over the life of one of its most successful UAS platforms, the General Atomics MQ-9 Reaper [8]. It is commonly understood that UAS are ideal for missions that are too dull, dirty, or dangerous for a human pilot. Unsurprisingly, the most common application of modern UAS is in military surveillance and reconnaissance where they provide an unparalleled ability to ingress into a hostile environment without risk to a human operator. Common non-military applications include border patrol, aerial photography, geographic survey, search and rescue, industrial inspection, and agriculture.

The U.S. Air Force, Army, and Navy have fleets of UAS ranging in scale from the commercial-airliner sized Northrop Grumman RQ-4B Global Hawk down to the backpack sized Aerovironment Wasp. The Lockheed Martin Desert Hawk, Aerovironment RQ-11 Raven, and the Naval Research Laboratories RQ-14 Dragon Eye are examples of a few small scale UAS platforms in production today which have successfully

packed all the necessary hardware of a UAS into a small platform. Miniaturizing flight system components presents a particularly steep set of challenges and, perhaps as a direct result, the system and application spaces around that scale remain considerably unexplored. The smaller micro-scale class of UAS, designated MUAS or μUAS, has garnered significant attention within the aerospace community due to the potential for novel application and the interesting challenges encountered in designing μUAS. Notably, there are a number of inherent challenges to miniaturizing such a complicated electro-mechanical system while maintaining the crucial elements that make larger UAS valuable. For example, all UAS require some form of effective navigation system, to enable the aircraft to determine its precise location and orientation with respect to the Earth. In large-scale vehicles these systems are reasonably well understood, if generally expensive, voluminous, and massive. As the vehicle scale decreases the challenges and costs of implementing an effective navigation system increase quickly and large scale solutions become wholly inappropriate. Compound that challenge with the necessity to perform additional in-flight tasks like communications, stabilization and control, and mission planning, and the need for efficient algorithms which optimize processor utilization becomes apparent. Fortunately, trends in embedded sensing and computing, driven in no small part by the consumer electronics industry continuing the Moore's-Law-like progress seen in computing, have brought compelling embedded processing and sensing capabilities into reach for much smaller platforms. Nonetheless, the realization of useful systems on that micro-scale requires a particularly effective application of avionics design and navigation system development. Specifically, trying to solve and invert the large systems of equations necessary to compute a general navigation solution on an embedded platform with limited memory, slow single-threaded execution(as is common in most mainstream micro-controllers), and limited access to support libraries is challenging. The goal of this work was to construct a highly configurable navigation system meant to be leveraged by a size, weight, and power constrained μUAS, but which could also be extended to larger more capable platforms.

1.2 Novel Micro-UAS Platform

In late 2010 engineering students at MIT were tasked, as part of a senior capstone design course, to develop a small foldable μUAS platform for distributed sensing. The challenging design goals called for a vehicle that could be deployed from an airborne platform via a common flare dispenser round (48mm x 62mm x 180mm) to achieve persistent high-altitude (30,000ft) atmospheric sampling . The capstone course yielded

a proof-of-concept design that was subsequently refined by an MIT masters student who guided a small student team to build a light-weight flying prototype in early 2012. The resulting design, called the Locust, was a small tandem-wing aircraft with a pusher-style propeller (Fig. 1.1). [35]

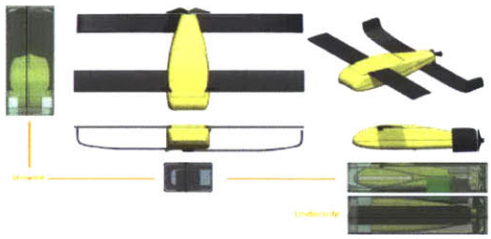


Figure 1.1: Locust μUAS and flare canister housing

The vehicle has only two control surfaces, one to either side of centerline, on the trailing edge of the rear wing in an elevon control scheme. Elevon control describes a mechanism of blending conventional aircraft wing and tail control surface effects into a smaller number of control surfaces, typically near the aft of an aircraft. Despite a reduction in the degrees of control freedom, the blended control surfaces can typically yield sufficient control of both aileron/roll control (traditionally implemented on the main lifting surface) and elevator/pitch control (traditionally implemented on the rear lifting surface) through a combination of symmetric and differential articulation. This blending offers reduced mechanical complexity but typically comes at the cost of decreased control authority. The elevons are controlled by a combination spring/servo mechanism that uses a spring loaded hinge to automatically deflect the surfaces upwards. Miniature servos actuate the elevons through rotating pusher-arms that press down on the control surfaces. A brushless motor provides thrust out of the rear of the μUAS from a pusher-style propeller. The aircraft was designed to collapse to the form-factor of an air-launch flare canister with wings that rotate laterally on vertical hinges to align with the fuselage. Figure 1.1 depicts the airframe in both its flight and "stowed" configurations to accommodate the confines of the flare canister. The original design of the μUAS in its flight configuration is shown in Figure 1.2, whose constructed specifications are detailed in Table 1.1.

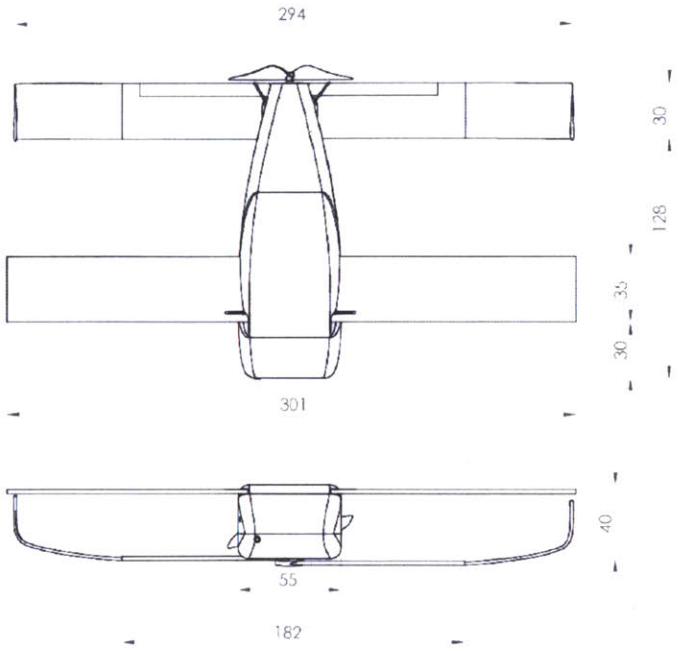


Figure 1.2: Locust μUAS Dimensions

Table 1.1: Locust μUAS Specifications

Wingspan (mm)	301
Length (mm)	158
Width (mm)	55
Height (mm)	40
Weight (g)	250

The project culminated in a favorable evaluation of basic flightworthiness through piloted testing but indicated a clear requirement for on-board stabilization and guidance.

1.3 Problem Statement and Research Objectives

To accomplish missions, most UAS require some degree of autonomous flight control. This is done either in the most basic form, as stabilization aiding for a remote-pilot, or in the more complex case, completely pilotless autonomous operation. The challenges of autonomous flight can be thought of as being comprised of three major components: Guidance, Navigation, and Control (GNC) (Fig. 1.3). Guidance and control systems are

responsible for a wide range of activities including planning a UAS flight path in space and determining the appropriate control signals to drive the UAS towards the intended path. The guidance system determines a desired vehicle state, indicated by signal r in Figure 1.3. The navigation block, as previously described, is concerned with determining the state of the vehicle (position, velocity, and attitude). The navigation system produces an estimate, $\hat{x}$, of the current vehicle state. The difference between the guidance system's desired state, and the navigation system's reported state is called the state error signal, e. The control system determines how to drive the error signal to zero, effectively forcing the µUAS toward the desired state r. The µUAS dynamics determine how the controller's output, u, affects the system output state, z. The output z is measured through sensors which feed in the new state to the navigation system, which closes the GNC loop. Note that noise in the system, indicated by the signals w, ν, and v, can enter in at several locations, making the job of closing the GNC loop harder. The construction of a noise-rejecting and self-correcting navigation system, the effective foundation of autonomous GNC loop, is the goal of this research.

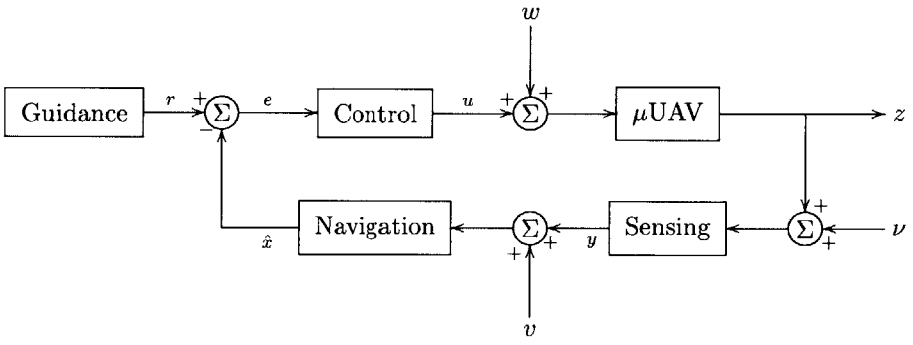


Figure 1.3: Closed-Loop μUAS GNC System

An Inertial Navigation System (INS) uses measured inertial phenomena, like accelerations and rotation rates, along with external navigation measurements, like the Global Positioning System (GPS), in order to keep track of an aircraft's motion and attitude. Ultimately, the INS will be constructed as part of a larger software system commonly know as a Flight Management System (FMS). The FMS encapsulates all of the major interface, monitoring, and management functionality required of an autonomous UAS including, but not limited to, communication management, high-level mission guidance, low-level trajectory guidance, vehicle navigation, flight control and stabilization, and sub-system failure monitoring and recovery. The INS is critical for the vast majority of these tasks, as the FMS requires a timely and accurate estimate of the vehicle's dynamic state to enact guidance and control tasks.

At a minimum the INS must fuse data streams from a wide range of sensor systems to develop an accurate estimate of the vehicle state. Position and orientation are commonly expressed in latitude, longitude, and altitude and vehicle roll, pitch, and yaw, respectively. Small commercial INS solutions which provide these data products are offered from vendors like SBG-Systems and VectorNav. Both companies offer solutions, shown in Figure 1.4, which can provide state information, with varying levels of accuracy, in a footprint approaching the size of a U.S. postage stamp. However, these devices offer little in terms of access or customization of the the internal algorithms which produce the needed data products. Furthermore, although these commercial INS may be small compared to most avionics components in full size UAS, neither are small enough to fit inside the confines of the Locust μUAS without displacing the motor, batteries, servos, or any potential payloads.

Figure 1.4: SBG-500N (L) and VectorNav VN-200 OEM(R)

The lack of a sufficiently flexible/extensible, and form-factor compliant, commercial option leads to the conclusion that a custom INS was needed in order to offer both the required level of customization and system miniaturization necessary to incorporate the INS components into the UAS avionics hardware. Typically an INS uses a sensor package called an Inertial Measurement Unit (IMU), comprised of accelerometers and gyroscopes, in order to directly sense dynamic changes in the an aircraft's state. Specifically, the INS integrates the rotations and acceleration reported by the IMU to track changes in position, velocity, and orientation. The problem with using an IMU in such a direct manner is that the accuracy of the INS result depends heavily on the quality of the sensors in the IMU. Large UAS platforms can employ tactical grade senors which produce accurate measurements that degrade slowly over time. Most μUAS platforms, however, make use of less stable MEMS sensors. Problems like signal drift, noise, and nonlinear responses make the challenge of producing an accurate state prediction through direct integration of a MEMS based IMU extremely difficult, if not impossible over long time scales without some form of correction or aiding. Consequently in order to produce stable and accurate results, most INS, especially those with a MEMS IMU,

utilize some form of state estimation algorithm to incorporate aiding measurements and to mitigate the non-ideal effects of sensors. This thesis explores building and testing a MEMS-based INS solution designed for an embedded environment.

1.4 Outline

Section 2 is a description of the avionics hardware used in the Locust μUAS, along with the MEMS sensors which comprise the IMU. An analysis of typical MEMS error dynamics is given as well as a method to compensate for them in the INS in Section 2.3. A theoretical overview of some of the key topics in inertial navigation is given along with a derivation of Kalman filter state estimator in Section 3. The building blocks of an INS are discussed and incrementally developed in Section 4, culminating in the final test and analysis of the resulting INS produced for the Locust μUAS in Section 5.

Chapter 2

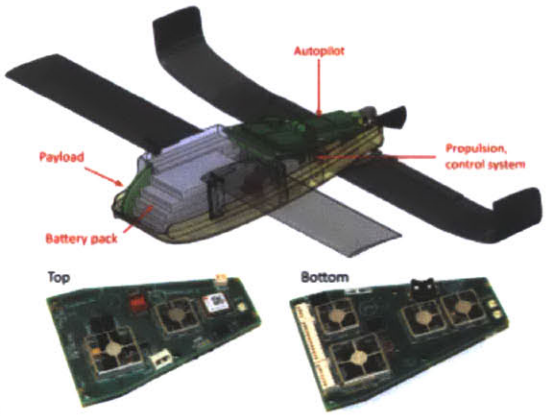
Avionics Systems

The term avionics is generally used to describe flight-related electronics and software systems inside of an aircraft. An avionics suite typically includes the following key elements: central processing system, peripheral processing systems, sensor systems, power systems, and communications systems. A unified avionics board was developed for the Locust μUAS at the outset of the navigation system development effort. A significant effort followed to develop the software interfaces necessary to realize functionality with the avionics hardware. Developing a complete custom avionics package requires the programming of device-level drivers and the development of interface and device management software for all of the avionics board subsystems. This involves communicating with, configuring, and managing data from a variety of disparate electrical subsystems via a range of communication and signaling standards including UART, I2C, SPI, PWM, PPM, as well as general-purpose digital and analog signals. This effort required almost half of development and debugging time for the project since stable, reliable, and efficient low-level functionality is critical to the operation of higher-level algorithms. What follows is a description of the important avionics hardware systems and required software development.

2.1 Primary Avionics

A detailed view of the internals of the flight vehicle and the custom avionics board are shown in Figure 2.1. The avionics board hosts an NXP LPC2148/ARM7TDMI-S based 32-bit RISC processor. The ARM7 chip contains 32kB of RAM, 512kB on-chip Flash ROM, a vectored interrupt controller, 2x10-bit ADCs, 2xUARTs,

2xI^{2}C buses, 2xSPI buses, 2x32-bit timers, PLL, RTC, and more than 45 GPIO pins. The core is programmed using a GNU C Compiler (GCC) based Eclipse Integrated Development Environment (IDE). The avionics board also houses a GPS chipset, a 900MHz wireless transceiver, a Radio-Frequency Identification Device (RFID) programming interface with data storage, power regulation, a 3-axis magnetometer, and an IMU which is discussed in greater detail in Section 2.2 below. Non-volatile storage was added as part of this development effort, through the addition of a custom external Micro-SD card interface. The vehicle is capable of carrying an auxiliary sensor board for estimation error reduction using a comparable secondary IMU and pressure-based airspeed and altimetry.



flight

Figure 2.1: Locust avionics board and internals[35]

The 900MHz link is used for IMU data and telemetry reporting. The datalink chipset, an Atmel AT86RF212, handles most of the necessary wireless communications tasks of packetization, frame filtering, modulation, and encryption. A Commercial-Off-The-Shelf (COTS) Remote Control (RC) receiver is also carried on-board the aircraft and interfaces with the primary avionics, the micro-scale servos, and the Electronic Speed Control (ESC) which drives the 4-pole brushless electric motor. This configuration allows for RC piloted flight during testing and development stages. Switching DC-DC/buck converters regulate the board's voltage to 5V and 3.3V from the µUAS battery pack voltage. The battery is a 11.1V Lithium-Polymer (Li-Po) pack in a 3S1P, or a single three-cell series configuration. Status LEDs are used to report system state and health and a set of micro DIP switches, the red square in Figure 2.1, is used for system configuration after the board has been programmed.

Overall the final avionics software needs to poll approximately 40 measurements, manage multiple com-

munications systems including one wireless datalink, interface with non-volatile storage media, and monitor and issue vehicle control signals dozens of times every second to achieve basic system-wide functionality. The software must also execute all of the previously discussed vehicle management processes including vehicle state estimation and navigation, guidance and trajectory determination, flight control/stabilization, and power and subsystem configuration management. To accomplish this, the software must be efficient and employ careful design at every step in order to be able to conduct all of the required functions at a frequency high enough to capture and manage the rapidly changing dynamics of a small UAS. Furthermore, efficiency and extensibility are required for long-term development in a UAS program as the ability to quickly add and/or modify high-level functionality is critical to respond to complicated and continuously evolving missions.

2.2 Sensor Systems

The primary avionics sensor package includes a u-blox MAX-6Q GPS chipset capable of 5Hz satellite navigation updates and a strapdown nine-degree-of-freedom inertial measurement unit comprised of three 3-axis MEMS sensor packages of accelerometers, magnetometers, and gyroscopes. Magnetometers measure the direction and strength of their local magnetic field, and the accelerometers and gyroscopes measure the accelerations and rotational rates to which the sensor is subject, respectively. For reference, Titterton et al. describes the merits of different IMU systems and configurations including gimballed systems, fiber-optic laser gyroscopes, and MEMS systems, which are typically used for μUAS applications to meet Size, Weight, and Power (SWaP) constraints [39].

At the most basic level and ignoring noise and bias errors, accelerometer signals can be integrated over time to determine vehicle velocity and position from initial values. Gyroscopes can similarly be integrated to determine changes in the μUAS orientation. Magnetometers are used to constrain and measure orientation angles and are most effective for the determination of vehicle heading, as will be discussed later. Tables 2.1-2.3 contain IMU sensor specifications for the primary avionics board.

Table 2.1: Analog Devices ADXL345 Accelerometer Specifications [7]

	Min	Typical	Max
Measurement Range (g)	± 2		± 16
Resolution (bits)	10		13
Sensitivity (LSB/g)	230	256	282
Bias Offset(x,y/z) (milli – g)	$-150/-250$		150/250
Noise RMS(x,y/z) (milli – g)		2.92/4.29	
Data Rate (Hz)	0.1		3200

The accelerometer is capable of limiting its output range, given its full resolution of 13-bits, in order to alter the sensitivity. The sensor was used in full resolution mode/13-bit mode at ± 16g, where 12 bits were used for encoding magnitude and the Most Significant bit (MSb) was used to encode the sign of the signal. The resulting sensitivity is equal to $^{16g}/_{2^{12}} = 3.90625 \cdot 10^{-3} \frac{g}{LSb}$ or equivalently $256 \frac{LSb's}{g}$ which means that the smallest amount of change in acceleration that the sensor can discern is $3.90625 \cdot 10^{-3}$g. The bias offset characteristic for the x-axis and y-axis are identical while the z-axis is prone to a larger bias. Similarly the z-axis is prone to higher levels of noise, as shown in the table.

Table 2.2: ST L3G4200D Gyroscope Specifications [33]

	Min	Typical	Max
Measurement Range ($\frac{degs}{s}$)	± 250		± 2000
Resolution (bits)		16	
Sensitivity @ $2000 \frac{degs}{s}$ ($\frac{degs}{LSb}$)		$70 \cdot 10^{-3}$	
Bias Offset		-	
Noise Rate Density ($\frac{degs}{s}/\sqrt{Hz}$)		0.03	
Data Rate (Hz)	100		800

The gyroscope is configured to operate at the $2000 \frac{degs}{s}$ range at a full resolution of 16-bits. The resulting sensitivity is then $^{2000}/_{2^{15}} = 0.0610 \frac{degs}{LSb}$ which is less than the specified sensitivity. Therefore the entire 15-bit range is not addressable for the specified sensitivity of $0.070 \frac{degs}{s}$. The bias offset, otherwise called the digital zero-rate level, was quoted in the specifications as having a value of $\pm 75 \frac{degs}{s}$ which does not match experimental laboratory testing results of a zero-rate level of $\pm 1 \frac{degs}{s}$; this specification was therefore omitted here as an assumed error in the documentation.

Table 2.3: Honeywell HMC5883L Magnetometer Specifications [17]

	Min	Typical	Max
Measurement Range (gauss)	±1		±8
Digital Resolution @12-bit ADC (milli-gauss)	0.73		4.35
Sensitivity ($\frac{\text{LSb}}{\text{gauss}}$)	230		1370
Bias Offset			
Noise Floor (milli-gauss)		2	
Data Rate (Hz)	0.75		160

The magnetometer measures the field strength up to ±8 gauss in any axis. The sensor can resolve changes of magnetic field strength as small as $7.3 \cdot 10^{-4}$ gauss and report single measurements at a rate of 160Hz.

Table 2.4: u-blox MAX-6Q Specifications [37]

	Min	Typical	Max
Position Accuracy (CEP m)		2.5	
Velocity Accuracy ($\frac{\text{m}}{\text{s}}$)		0.1	
Heading Accuracy (degs)		0.5	
Start Time (s)	3 (aided start)		27 (cold start)
Tracking Sensitivity (dBm)		−161	
Noise(worst case) (LSB rms)		1.1	
Data Rate (Hz)			5

The u-blox chipset provides a complete GPS solution for the Locust μUAS. It is capable of producing many different data products as well as individual satellite and ephemeris data. A summary of its specifications is listed above in Table 2.4. The position accuracy is reported in a Circular Error Probable (CEP), which is a circle large enough whose probability of encircling the GPS antenna's true position is 50%.

2.3 Calibration

In order to use most sensors effectively, they usually must be calibrated and compensated for, which is particularly true for MEMS sensors. Most MEMS sensors suffer from at least basic biasing errors and additive noise, as well as potential nonlinearities in their output response, all of which can cause divergence in the INS solution. Below are some methods which can be used in order to mitigate these sources of error.

2.3.1 Six-Point Accelerometer Calibration

MEMS accelerometers are prone to common performance errors that include slowly drifting biases and response nonlinearities over their full output range. The bias signals, which are more thoroughly modeled in Section 3.4 , are can be thought of as products the of a random walk process. The nonlinear effects of MEMS accelerometers can be described by models discussed by Farrell [12, p410]. Figure 2.2 shows some of the possible response curves that can be expected from a MEMS accelerometer.

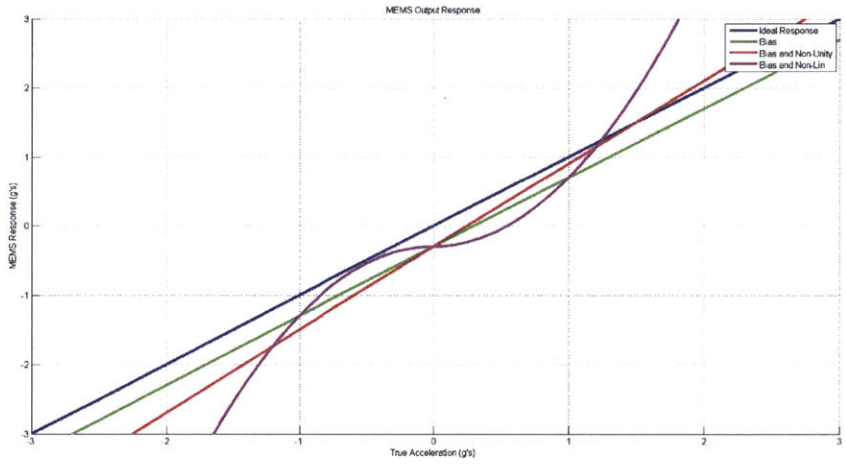


Figure 2.2: MEMS Output Responses

The values shown in the figure are in units of g's. The blue curve depicts the ideal response with a one-to-one input output relationship. The green curve is representative of a bias offset, in this case a bias of $-0.3g$. The red curve represents what can happen if the sensor exhibits some scale factor error as well as a bias. Finally, the purple curve shows an overall nonlinear output response. In this research, a linear scale factor model is used. The output is modeled as follows:

$$a_{out} = ma_{in} - b \tag{2.1}$$

The response is modeled as a linear equation with some non-unity slope m, and some non-zero bias b. Utilizing a piece-wise model, which breaks up the input-output space into several discrete steps, can be used to produce a more accurate estimate of the response of the sensor. If an estimate for the accelerometer bias and input-output slope is available, the MEMS data can be corrected using Equation 2.2.

$$a_{est} = \frac{a_{out} + b_{est}}{m_{est}} = \frac{ma_{in} - b + b_{est}}{m_{est}} \approx a_{in} \quad if\ b_{est} \approx b\ and\ m_{est} \approx m \tag{2.2}$$

In order to obtain initial estimates for the bias and slope values, a six-point calibration is performed. The calibration involves recording the IMU output during a sequence of sequential rotations where gravity points along and opposite to each of the three principal axes, thus obtaining a six-point calibration. Each orientation is held for a period of of at least $30s$ such that a sufficiently accurate central value and data statistics can be determined. The accelerometer data from a six-point calibration is shown in Figure 2.3

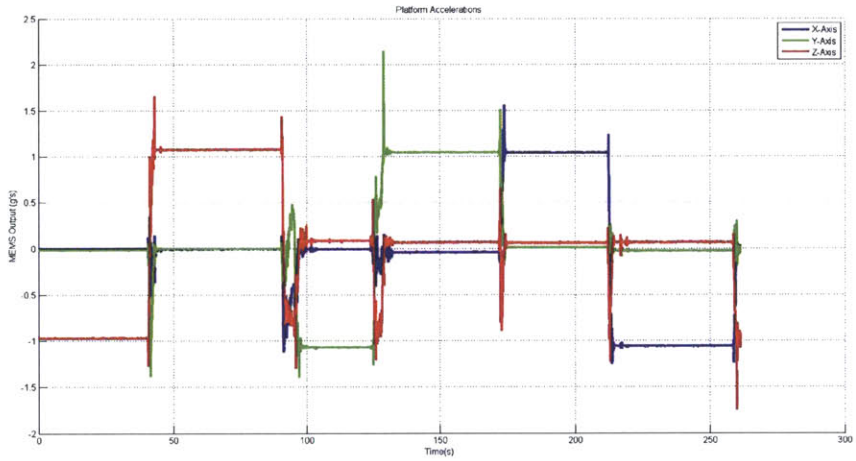


Figure 2.3: Six-Point Calibration Data

The calibration depicts the IMU output of each axis alternating between positive and negative 1g. The Locust INS board was kept as flat as possible during each rotation to improve the calibration results. The bias estimates are produced by averaging the values of each channel during the periods of time for which the gravity vector was perpendicular to measured axis. Equation 2.3 outlines the averaging process for period of time when the magnitude of the accelerometer output is less than $0.2g$, thus isolating instants in time where the IMU was not being rotated and the axis in question was not aligned with gravity. The slope was calculated by averaging the positive and negative 1g output values, called a_{out+} and a_{out-} respectively. These values are used in Equation 2.4 to find the MEMS calibration slope estimate, m_{est}.

$$b_{est} = \frac{1}{\mid U \mid} \sum_{t \in U} a_{out}(t) \quad U = \{t\ \ s.t.\ \mid a_{out}(t) \mid < .2g\} \tag{2.3}$$

$$m_{est} = \frac{\triangle a_{out}}{\triangle a_{in}} = \frac{a_{out+} - a_{out-}}{2} \tag{2.4}$$

The results of the above six-point calibration for initial accelerometer biases and slope estimates are shown in Table 2.5. The estimates are used like in Equation 2.2 to provide the INS with better accelerometer data.

Table 2.5: Six-Point Calibration Results

	X	Y	Z
Bias ($\frac{m}{s^2}$)	-0.0163	-0.0110	0.0617
Slope	1.0468	1.0554	1.0211

2.3.2 Gyroscope Rate Table Calibration

MEMS gyroscopes, like accelerometers, also usually suffer from a bias and noise problem. However, their bias signals drift considerably faster than the bias signals of accelerometers and therefore obtaining static values for gyroscope biases are not as valuable. A more important calibration for MEMS gyroscopes are full-range scale-factor errors which manifest much like the response nonlinearities described above. In order to measure the output response of the gyroscopes, a device called a rate table is used to spin the IMU at specific angular velocities along each axis. For each selected angular rate, the sensor's output can be recorded. The resulting data will yield an input-output relationship similar to the MEMS accelerometer response shown in Figure 2.2. Depending on the complexity of the response curve either a single linear model, a higher-order polynomial, or a piece-wise model can be fit to the curve for correction of the scale-factor behavior of the sensor. Alternatively a look-up table can be used to speed up the correction process. Unfortunately, a rate table was not available during the time of this research in order to calibrate the gyroscopes properly. A factory calibration will be utilized and, wherever possible, partial corrections will be extrapolated from our limited testing against similar sensors.

2.3.3 Hard Iron and Soft Iron Calibration

As previously discussed the INS uses the magnetometers for yaw/heading measurements. A properly calibrated magnetometer, in the presence of a uniform magnetic field, reports the magnetic field vector uniformly over a 3D sphere centered about the sensor. The sensor will consistently report the direction and magnitude of the magnetic field vector, regardless of orientation. As described in [16], there are two sources of corrup-

tion in the magnetometer data, known as the hard-iron and soft-iron effects. The Hard-Iron (HI) effect is a result of magnetic fields produced by permanently magnetized ferromagnetic materials in the vicinity of the sensor. Metal cages, housings, wiring, and even the printed circuit board on which the magnetometer is placed are able to produce their own magnetic fields and, therefore, cause a hard-iron disturbance in the data. Such a disturbance manifests as an offset in the 3D measurement response sphere from the origin. The Soft-Iron (SI) effect is a result of magnetic fields induced by the geomagnetic field of the Earth. Such fields deform the measurement response sphere from a sphere to an ellipsoid. Please note that the following discussion relies heavily on the use of rotation matrices and reference frames; refer to Section 3.1 and Section 3.2 for a discussion of reference frames, rotation matrices, and the uses thereof. Without the HI/SI effects, the measurement of the local magnetic field are given by the relationship expressed in Equation 2.5. In the northern hemisphere, the local magnetic field strength, B, is distributed along the northward and downward directions in the navigation or Earth frame. The navigation frame is related to the measurement obtained in the platform frame of the sensor by the rotational offset shown in Equation 2.5.

$$B_p = R_n^p B_n = R_x(\phi)R_y(\theta)R_z(\psi)B \begin{bmatrix} \cos\delta \\ 0 \\ \sin\delta \end{bmatrix} \tag{2.5}$$

The angle of inclination, δ, determines what amount of the magnetic field is directed in the north or down direction. The inclination angle varies from $\delta = 0$, at the equator to $\delta = 90$ at the North pole. The angle of declination, the location dependent offset between magnetic north and true north, is ignored in this derivation and we assume that it can be subtracted out by the INS. The platform frame magnetometer reading B_p is the product of the local magnetic field vector with the rotation matrix from navigation frame to the platform frame. The HI effect manifests as an offset in the platform frame magnetometer reading, represented by the vector V in Equation 2.6. The SI effect is modeled as a 3x3 matrix W, which distorts the measurement in the platform frame. The combined HI/SI measurement model is given by:

$$B_p = WR_n^p B_n + V = WR_x(\phi)R_y(\theta)R_z(\psi)B \begin{bmatrix} \cos\delta \\ 0 \\ \sin\delta \end{bmatrix} + V \tag{2.6}$$

Given a body/platform frame magnetometer measurement, roll ϕ, pitch θ, and an estimate of the SI

matrix W_{est} and the HI vector V_{est} , Equation 2.6 can be rearranged, as shown in Equation 2.7, to yield a measurement of yaw represented by Equation 2.8.

$$B_f = \begin{bmatrix} B_{fx} \\ B_{fy} \\ B_{fz} \end{bmatrix} = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} B \begin{bmatrix} \cos\delta \\ 0 \\ \sin\delta \end{bmatrix} = R_y(-\theta)R_x(-\phi)W_{est}^{-1}(B_p - V_{est}) \tag{2.7}$$

$$\psi = \arctan\frac{-B_{fy}}{B_{fx}} \tag{2.8}$$

An estimate of the SI matrix W and the HI vector V are needed in order to complete the calculations above. The locus of all response vectors from a magnetometer subject to combined HI/SI effects is an ellipsoid which is offset from the origin in the magnetometer measurement space. Ideally the response would be a sphere (soft-iron corrected) centered at the origin (hard-iron corrected). For more explanation about the nature of these effects see [15, 16]. The equation of an ellipsoid in quadratic form is shown in Equation 2.9.

$$(x - v)^T A(x - v) = 1 \tag{2.9}$$

$$A = PDP^{-1} = \begin{bmatrix} \uparrow & \uparrow & \uparrow \\ evec_1 & evec_2 & evec_3 \\ \downarrow & \downarrow & \downarrow \end{bmatrix} \begin{bmatrix} \frac{1}{a^2} & 0 & 0 \\ 0 & \frac{1}{b^2} & 0 \\ 0 & 0 & \frac{1}{c^2} \end{bmatrix} \begin{bmatrix} \uparrow & \uparrow & \uparrow \\ evec_1 & evec_2 & evec_3 \\ \downarrow & \downarrow & \downarrow \end{bmatrix}^{-1} \tag{2.10}$$

The matrix A in Equation 2.9, is expanded in Equation 2.10. The A matrix determines the shape of the ellipsoid. Specifically, the P matrix contains three eigenvectors which point in the directions of the three principle axes of the ellipsoid. The D matrix contains the eigenvalues of the quadratic along its diagonal, whose inverses are the squares of the radii, (a, b, c), of the ellipsoid. Therefore by altering the A matrix, we can alter the size, shape, and direction the ellipsoid points in. Expressing Equation 2.6 in a similar quadratic form yields:

$$(W^{-1}(B_p - V))^T W^{-1}(B_p - V) = (R_n^p B \begin{bmatrix} \cos\delta \\ 0 \\ \sin\delta \end{bmatrix})^T (R_n^p B \begin{bmatrix} \cos\delta \\ 0 \\ \sin\delta \end{bmatrix}) = B^2 \tag{2.11}$$

$$(B_p - V)^T (W^{-1})^T W^{-1}(B_p - V) = B^2 \tag{2.12}$$

The B^2 term can be dropped from the analysis since information about the total strength of the magnetic field is largely inconsequential, as it is the direction and not the magnitude of the field that is important for navigation. By inspection we see that $(W^{-1})^T W^{-1}$ is equivalent to the A matrix of the quadratic Equation 2.9. For reasons discussed in [15, 16] we assume that the inverse SI matrix W^{-1} is symmetric, that is $(W^{-1})^T = W^{-1}$, such that we may solve for the the matrix given the quadratic shaping matrix A.

$$W^{-1} = \sqrt{A} \tag{2.13}$$

In order to find the matrix which defines the ellipsoid's radii and direction, A, we must fit some ellipsoid to a test data set of magnetometer readings. The center of the ellipsoid data will correspond to the HI offset vector V. MATLAB was used to calculate the ellipsoid's radii and center locations. An accurate HI/SI calibration will require that a test data set records the response of the magnetometer to the geomagnetic field in a wide range orientations such that an imaginary vector pointing out of the IMU would trace out the shape of the ellipsoid of interest. Figure 2.4 depicts one HI/SI calibration data set. The green data depicts the raw magnetometer output. Notice the center offset and ovoid shape which matches the prediction of a HI and a SI effect present in the measurement process. The red data set is corrected for the HI effect, and therefore appears similar in shape to the raw measurements, but now centered about the origin. Finally the blue data set represents the combined HI/SI calibrated measurements with unity magnitude. Notice the ovoid features of the green and red data have been replaced by a spherical figure, and that the radii of the HI/SI calibrated data sphere is approximately 1 in every direction.

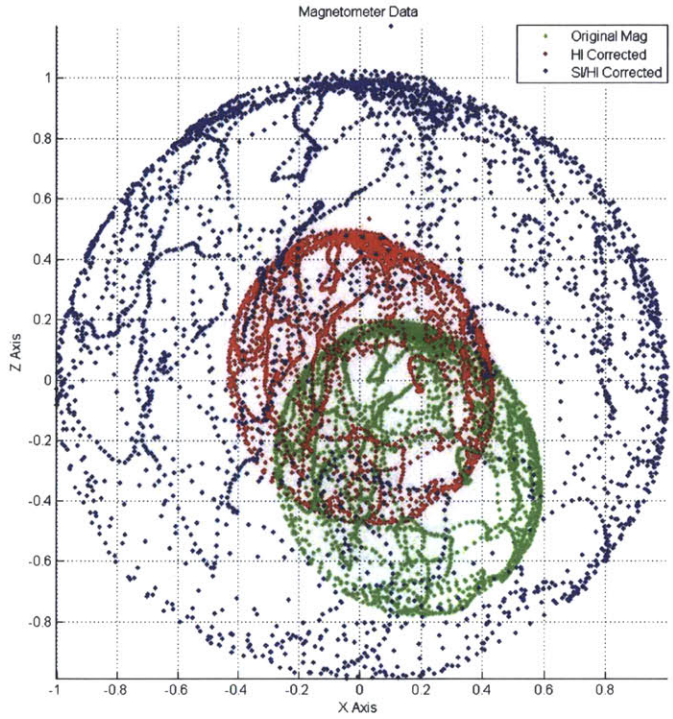


Figure 2.4: HI/SI Calibration Data

Figure 2.5 shows the same data as above, with the addition of spheroid fit which was calculated from the HI/SI data. Since the magnetometer measurements can now be projected onto a unit-sphere, for which accuracy will not vary with respect to orientation, they may be used to faithfully produce a yaw measurement.

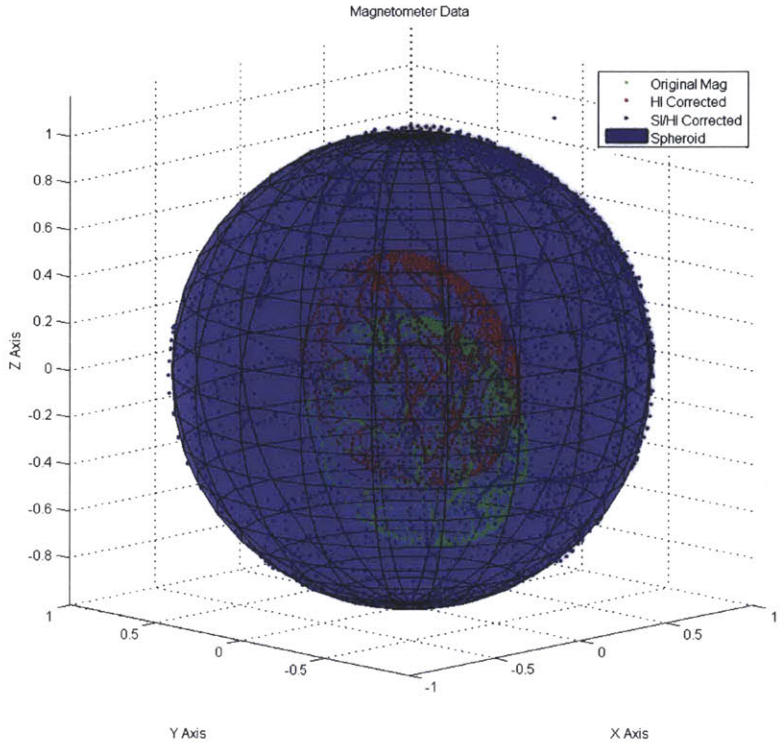


Figure 2.5: HI/SI with Spheroid Fit

Chapter 3

Inertial Navigation

This chapter describes a mathematical derivation of the key techniques and concepts used in developing an INS. A short review of reference frame definitions and rotational dynamics is given, followed by the derivation of the discrete-time recursive Kalman filter. The Kalman filter is a state estimator which generates an optimal estimate, in the minimum variance sense, of state variables when configured properly to account for system noise characteristics, and is common in many INS architectures. As mentioned above, a state estimator is useful to an INS because of its ability to mitigate the non-ideal characteristics present in some of the sensors used. Without a state estimator, integrating raw rate sensor signals subject to biases and noise would yield compounding errors in time and result in increasingly inaccurate state estimates and eventually a failure to control the aircraft. Section 3.4.1 demonstrates what can happen to a simple system utilizing a MEMS accelerometer without the use of a state estimator and aiding information. A more in depth discussion of Kalman filters can be found in [25, 18, 21, 38]. Farrell [12] and Kuipers [20] both give summaries of the important concepts of rotational dynamics and how they apply to navigation.

3.1 Coordinate Systems

In order to discuss the complexities of designing an INS, we must first define the reference frames and kinematic conventions of vehicle dynamics. Stimac describes several options for defining an Earth relative coordinate system, which are summarized below [34]. Different models are useful for different situations and for specific requirements. The relationships between these frames are important to understanding the

evolution of the vehicle's dynamics state. Specifically, it is important to be able to freely move from one reference frame to another while preserving relative state information.

1. Earth Centered Inertial (ECEI): The origin is located at the center of the earth. The z-axis points upwards through the North Pole, while the x-axis and y-axis lie perpendicular to each other on the equatorial plane, forming a right-handed plane. The x-axis is defined along some predetermined longitudinal axis during some instant in time. The rotation of the earth is neglected in this frame definition.
2. Earth Centered Fixed (ECEF): The axes are the same as ECEI, except the axes are fixed while the earth rotates about them. This frame is useful for the consideration of motion relative to the surface of the Earth.
3. Geodetic: Coordinates based on the ellipsoid shape of the Earth.
4. Geocentric: Coordinates based on a spherical approximation of the Earth.
5. North East Down (NED): A local Cartesian projection onto the Earth's surface where the x-axis points north, the y-axis points east, and the z-axis points in decreasing altitude towards the center of the Earth.
6. North East Up (ENU): A local Cartesian projection onto the Earth's surface where the x-axis points east, the y-axis points north, and the z-axis points in increasing altitude away the center of the Earth.

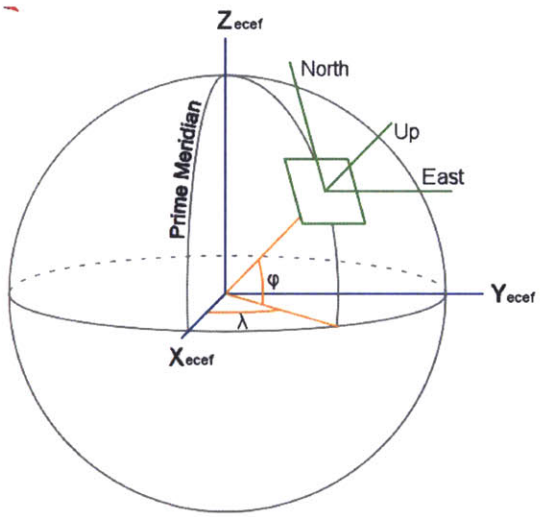


Figure 3.1: ECEF and local NEU frames[40]

Figure 3.1 depicts an ECEF frame and an ENU frame placed on the surface of the Earth. An ENU frame maps the (E,N,U) directions to the (x,y,z) axes. An alternative, as mentioned above, is the NED frame which instead has the z-axis directed downward towards the center of the Earth and maps the (N,E,D) direction to

the (x,y,z) axes. An NED frame is used in this thesis and is defined locally on the surface of the Earth near some predefined origin. This choice is convenient for analysis but will limit the accuracy of the navigation filter as the vehicle moves further away from the NED frame origin. However, due to the expectation of Locust's limited range and short operating times, these inaccuracies are anticipated to remain small.

It is also important to define a coordinate system for the aircraft. Typically, a right handed aircraft coordinate system is defined to have an origin at the vehicle center of mass of the with the x-axis pointed out the nose along the longitudinal axis of the aircraft, and the y-axis directed out the right wing along the lateral axis of the airframe. The z-axis points downwards out the belly of the aircraft. Defining wind-relative motion of the vehicle may also be useful and, therefore, we define a set of wind-axes that align with the local direction of travel and are related to the vehicle axis by an y-axis rotation by the vehicle's Angle of Attack (AOA) to the wind and a z-axis rotation by the vehicles angle of Side Slip (SS) to the local wind. Positive rotations in roll, pitch, and yaw are then defined about the respective vehicle axes: x, y, z using the right-handed convention. Subsequently from the vehicle-fixed perspective a right-handed roll, upward pitch, and nose-right yaw are considered positive. Figure 3.2 shows an illustration of the aircraft coordinate system used herein which is a common convention for aircraft rotation and translation. One quirk of this convention is that positive vehicle-relative deflection of any control surface on the trailing edge of an aerodynamic body (the most common configuration) actually yields negative rotation of the vehicle.

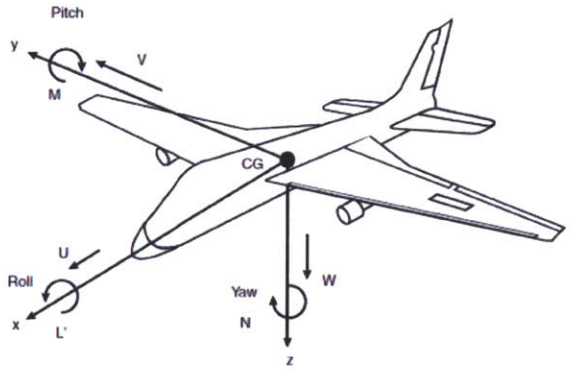


Figure 3.2: Aircraft Coordinate Frame [2]

3.2 Rotational Kinematics

One way to convert between a chosen navigation frame and the aircraft coordinate frame is through the use of a rotation matrix. A rotation matrix maps a vector from one space to a vector in another space while preserving certain aspects of the vector like magnitude and relative direction to other vectors in that space. For example, given axes x_1, y_1, and z_1, a vector $\vec{X}_1$ in that space, and a set of desired axes x_2, y_2, and z_2, it is possible to transform $\vec{X}_1$ to a vector $\vec{X}_2$ in the new set of axes. Equation 3.1 shows how to mathematically take a vector from one set of axes into another through multiplication of the rotation matrix, R_1^2. Two common approaches to define rotation matrices are Euler angles and unit quaternions. Euler angles are more intuitive to understand since they correspond to an aircraft's roll, pitch, and yaw. Quaternions are four-dimensional complex quantities which, while being numerically more complicated, have more desirable characteristics for the purposes consistent attitude determination. What follows ins a short summary of the concepts and equations relevant to Euler angles and quaternion based methods of constructing the rotation matrix.

$$\vec{X}_2 = R_1^2 \vec{X}_1 = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \vec{X}_1 \tag{3.1}$$

3.2.1 Euler Angles

The Euler angle method of coordinate transformation uses three consecutive rotations about the three principle axes to produce the rotation matrix. The first rotation is about the z_1-axis by an angle of ψ which produces a set of intermediate axes denoted as x_1', y_1', z_1'. The second rotation is about the y_1'-axis by an angle of θ which results in a new set of intermediate axes defined by x_1'', y_1'', z_1''. The final rotation is about the x_1''-axis by an angle of ϕ resulting in x_1''', y_1''', z_1''' which, if the three angles $\begin{bmatrix} \phi & \theta & \psi \end{bmatrix}$ are correct, are identical to the desired axes x_2, y_2, z_2. The three rotations are consecutive and, just like in matrix multiplication, order is extremely important. The rotation matrix R_1^2, shown in Equation 3.3, is the product of the three consecutively multiplied rotation matrices, $R_1^2 = R_x(\phi)R_y(\theta)R_z(\psi)$ multiplied in that order, generally called the aerospace sequence of the Direction Cosine Matrix (DCM).

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \quad R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \quad R_z(\psi) = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

$$R_1^2 = \begin{bmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ \sin\theta\sin\phi\cos\psi - \sin\psi\cos\phi & \sin\psi\sin\theta\sin\phi + \cos\psi\cos\phi & \sin\phi\cos\theta \\ \sin\theta\cos\phi\cos\psi + \sin\psi\sin\phi & \sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi & \cos\phi\cos\theta \end{bmatrix} \tag{3.3}$$

$$\phi = \arctan\frac{r_{23}}{r_{33}} \tag{3.4}$$

$$\theta = -\arcsin r_{13} \tag{3.5}$$

$$\psi = \arctan\frac{r_{12}}{r_{11}} \tag{3.6}$$

The rotation matrix above will transform a vector, such as a force or acceleration, from one frame (e.g. the platform frame) into another frame (e.g. the navigation frame). This is useful for integration through time by the INS. By inspection, the matrix can be used to extract the individual Euler angles which constitute the matrix, shown in Equations 3.4-3.6. The matrix itself can also be integrated through time in order to keep an accurate time-evolving representation of the relationship between the two reference frames. The matrix which transforms body/platform frame quantities to navigation frame quantities, R_p^n, cannot be propagated in time by simply integrating the body frame angular rates to produce the resulting angles in the navigation frame, which is what the Euler angles represent. This is a direct result of the fact that rotations in R^3, which form a mathematical set called the special orthogonal group or SO(3), are order dependent. For example imagine an aircraft performing a series of maneuvers in a body frame sense: first the aircraft rolls $+90°$, followed by a pitch of $+90°$, and then a roll of $-90°$. If order did not matter, and therefore rotations commuted under multiplication, then the two rolls would cancel each other out and there would only be a net pitch change. However, from the point of view of an observer in the navigation frame, the aircraft yawed $+90°$ to the right. This inconsistency is a result of the cross-coupling of rotations that happens as the orientation changes. Under a roll of $+90°$, pitch action in the body frame no longer results in pitch

action in the navigation frame, but instead is directly coupled into yaw. Equation 3.7 below describes this mapping from body frame angular rates to navigation frame Euler rates. The resultant Euler rates can then be integrated forward in time using Equation 3.8, where the indices k describe instants in time and t_{int} represents a small window in time where the body angular rates are assumed to be constant.

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{3.7}$$

$$\phi_k = \phi_{k-1} + t_{int}\dot{\phi}_{k-1} \quad \theta_k = \theta_{k-1} + t_{int}\dot{\theta}_{k-1} \quad \psi_k = \psi_{k-1} + t_{int}\dot{\psi}_{k-1} \tag{3.8}$$

Generally, the range in which the Euler angles can be represented needs to be constrained in order to eliminate multiple solutions for a given rotation matrix. For example, adding $2\pi/360°$ to any of the Euler angles will result in the same rotation matrix. Therefore, the common constraints used are that $-180° < \phi < 180°$, $-90° < \theta < 90°$ and finally $-180° < \psi < 180°$ [20]. Euler angles further suffer from a phenomenon called gimbal lock, which is a loss of a degree of rotation under certain circumstances. This problem is most easily explained by way of an example aircraft's rotation in space. Imagine an aircraft which is originally oriented so that all three Euler angles are zero. The aircraft then pitches to $\theta = +90° - \epsilon$, where ϵ is some infinitesimally small angle measure. As the aircraft pitches farther past $+90°$ something must happen to account for the increased pitch, while still keeping $\theta < 90°$. Therefore, in order to track the angle change, the common solution is to add $+180°$ to both the roll and yaw angles, which subsequently get re-normalized into their allowable ranges [20]. This problem is called a singularity and is a problem inherent in any Euler angle representation. The singularity can also be seen in Equation 3.3 where the secant terms could lead to instabilities around angles of $\pm 90°$. Quaternions, while conceptually more complicated, are a suitable alternative which eliminate the problem of singularities.

3.2.2 Quaternions

The discussion of quaternions given in this section is meant as a summary for the reader. Please refer to [20, 12] for a more thorough derivation of quaternions. Henceforth a quaternion will be defined by the symbol

q. A quaternion is a four-dimensional vector which extends the complex numbers. It contains a scalar $q_0 \in R$ and vector component $\vec{q} \in R^3$ as shown in Equation 3.9.

$$q = q_0 + \vec{q} = q_0 + iq_1 + jq_2 + kq_3 = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \tag{3.9}$$

Addition of two quaternions $p = p_0 + \vec{p}$ and $q = q_0 + \vec{q}$ works as expected where $p + q = (p_0 + q_0) + (\vec{p} + \vec{q})$. Notice how the scalar adds with the scalar and the vector adds with the vector. The vector addition works as normal vector addition in R^3 where addition is applied to individual components, i, j, k, individually. The quaternion dot/scalar product of the vector component works like normal vectors where each of the components in the vector multiply their respective counterparts, shown in Equation 3.10. The vector cross product is shown in Equation 3.11.

$$\vec{p} \cdot \vec{q} = p_1 q_1 + p_2 q_2 + p_3 q_3 \tag{3.10}$$

$$\vec{p} \times \vec{q} = \begin{vmatrix} p_2 & p_3 \\ q_2 & q_3 \end{vmatrix} i + \begin{vmatrix} p_3 & p_1 \\ q_3 & q_1 \end{vmatrix} j + \begin{vmatrix} p_1 & p_2 \\ q_1 & q_2 \end{vmatrix} k \tag{3.11}$$

Quaternion multiplication is more complicated, and the interested reader should again refer to [20, 12] for a more in depth analysis of quaternion operations. As defined by William Rowan Hamilton, multiplication of the vector components follows $i^2 = j^2 = k^2 = ifk = -1$. From this relationship, quaternion multiplication, denoted by $\otimes$, can be derived as follows in Equation 3.12.

$$p \otimes q = p_0 q_0 - \vec{p} \cdot \vec{q} + p_0 \vec{q} + q_0 \vec{p} + \vec{p} \times \vec{q} = \begin{bmatrix} p_0 & -p_1 & -p_2 & -p_3 \\ p_1 & p_0 & -p_3 & p_2 \\ p_2 & p_3 & p_0 & -p_1 \\ p_3 & -p_2 & p_1 & p_0 \end{bmatrix} \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \tag{3.12}$$

The complex conjugate of a quaternion is defined as $q^* = q_o - \vec{q}$ where the imaginary part of the quaternion has been negated. The quaternion norm defined in Equation 3.13.

$$N(q) = \sqrt{q^* \otimes q} = \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2} \tag{3.13}$$

The inverse of a quaternion is given in Equation 3.14. In the special case where the norm of the quaternion is unity, the inverse is also the conjugate, a property which will be important for rotation matrices.

$$q^{-1} = \frac{q^*}{N^2(q)} = \frac{q^*}{\left|q\right|^2} \; for \; N(q) = 1, \;\; q^{-1} = q^* \tag{3.14}$$

Unit quaternions are useful building blocks for describing rotations in R^3. Geometrically, the quaternion represents the rotation of a reference frame about a fixed axis defined by the vector $\vec{q}$. After tedious derivations and algebraic simplifications we can arrive at the equation which defines a rotation matrix equivalent to Equation 3.3, but derived instead from a quaternion. Whether or not the the matrix below or the matrix's transpose is the intended rotation matrix depends on which frame we want to rotate from, defined by how the initial quaternion was formed.

$$R_1^2 = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 + q_0q_3) & 2(q_1q_3 - q_0q_2) \\ 2(q_1q_2 - q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 + q_0q_1) \\ 2(q_1q_3 + q_0q_2) & 2(q_2q_3 - q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \tag{3.15}$$

The quaternion can also be derived from the rotation matrix, just as in Euler angles, though the sign of the solution is not unique [34]. Below are the relevant equations used to take the elements of a rotation matrix and form a unit quaternion which represents the same rotation.

$$4q_0^2 = 1 + r_{11} + r_{22} + r_{33} \tag{3.16}$$

$$4q_1^2 = 1 + r_{11} - r_{22} - r_{33} \tag{3.17}$$

$$4q_2^2 = 1 - r_{11} + r_{22} - r_{33} \tag{3.18}$$

$$4q_3^2 = 1 - r_{11} - r_{22} + r_{33} \tag{3.19}$$

The sign of the extracted quaternion elements can be determined from the following equations [34].

$$4q_0q_1 = r_{23} - r_{32} \tag{3.20}$$

$$4q_0q_2 = r_{31} - r_{13} \tag{3.21}$$

$$4q_0q_3 = r_{12} - r_{21} \tag{3.22}$$

$$4q_1q_2 = r_{12} - r_{21} \tag{3.23}$$

$$4q_2q_3 = r_{23} - r_{32} \tag{3.24}$$

$$4q_1q_3 = r_{13} - r_{31} \tag{3.25}$$

To go directly from Euler angles to quaternions, Equations 3.26-3.29 are used. Note that the first sign must be chosen for all of the quaternion components, reflecting the sign ambiguity referred to above.

$$q_0 = \pm(\cos\frac{\phi}{2}\cos\frac{\theta}{2}\cos\frac{\psi}{2} + \sin\frac{\phi}{2}\sin\frac{\theta}{2}\sin\frac{\psi}{2}) \tag{3.26}$$

$$q_1 = \pm(\sin\frac{\phi}{2}\cos\frac{\theta}{2}\cos\frac{\psi}{2} - \cos\frac{\phi}{2}\sin\frac{\theta}{2}\sin\frac{\psi}{2}) \tag{3.27}$$

$$q_2 = \pm(\cos\frac{\phi}{2}\sin\frac{\theta}{2}\cos\frac{\psi}{2} + \sin\frac{\phi}{2}\cos\frac{\theta}{2}\sin\frac{\psi}{2}) \tag{3.28}$$

$$q_3 = \pm(\cos\frac{\phi}{2}\cos\frac{\theta}{2}\sin\frac{\psi}{2} - \sin\frac{\phi}{2}\sin\frac{\theta}{2}\cos\frac{\psi}{2}) \tag{3.29}$$

The quaternion derivative is the quaternion product of the current quaternion with another quaternion formed by the three angular rates $\omega_{bq} = \begin{bmatrix} 0 & \omega_x & \omega_y & \omega_z \end{bmatrix}^T$. The resulting quaternion derivative is shown in Equation 3.30. The quaternion integration process is shown in Equation 3.31.

$$\dot{q} = .5q \otimes \omega_{bq} = .5 \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & -q_3 & q_2 \\ q_2 & q_3 & q_0 & -q_1 \\ q_3 & -q_2 & q_1 & q_0 \end{bmatrix} \begin{bmatrix} 0 \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{3.30}$$

$$q_k = q_{k-1} + t_{int}\dot{q}_{k-1} \tag{3.31}$$

Equation 3.30 can be found in alternate forms [20, 34, 39], specifically the order of the quaternion multiplication. The work described below assumes that the quaternion is initially formed from initial Euler angles (roll,pitch,yaw) information. Those same angles, if used to form a rotation matrix, would form the DCM from the navigation to the platform frame, R_n^p. Similarly, the initial quaternion would perform the same rotation as R_n^p, shown in Equation 3.32 transforming a vector in the navigation frame, v^n, to the platform frame, v^p. Therefore, the quaternion multiplication order of the conjugate followed by the quaternion will bring any vector from the navigation to the platform frame. Reversing the order would also reverse the direction of the transform, just like taking the transpose of the rotation matrix, $R_p^n = (R_n^p)^T$, shown in Equation 3.33.

$$\begin{bmatrix} 0 \\ \uparrow \\ v^p \\ \downarrow \end{bmatrix} = q^* \otimes \begin{bmatrix} 0 \\ \uparrow \\ v^n \\ \downarrow \end{bmatrix} \otimes q \tag{3.32}$$

$$\begin{bmatrix} 0 \\ \uparrow \\ v^n \\ \downarrow \end{bmatrix} = q \otimes \begin{bmatrix} 0 \\ \uparrow \\ v^p \\ \downarrow \end{bmatrix} \otimes q^* \tag{3.33}$$

3.3 The Discrete Kalman Filter

As described by Austin [3], Kriehn et al. [41] and Tischler [36] the Kalman filter and its derivatives are some of the most powerful tools for state estimation in the presence of uncertainty and they are especially useful for UAS. The Kalman filter is used in UAS for navigation, targeting, dynamic state estimation, and parameter identification. The general form of the Kalman filter process, as described by Stimac, Zacharn, and Welch et al., is a system of simultaneous equations that evolves from time step to time step in an update-prediction cycle.[34, 25, 38] The filter produces an optimal estimate in the presence of system and measurement noise, which is useful for guidance and system identification. The following is a derivation of the discrete time Kalman filter for time-steps indexed by the value k, adapted from How [18]. Assume there exists a state vector, $\vec{x}_k$, which evolves from time-step $k-1$ to k through multiplication of a state transition matrix A_d(the

d subscript denotes the discrete time state transition matrix). The system described by A_d may be controlled through some deterministic input signal $\vec{u}_k$ and also affected by some white Gaussian process noise $\vec{w}_k$. The driving signal $\vec{u}_k$ affects the system state through the matrix B_{ud}. Equation 3.34 describes how the state evolves from one time step to another subject to the system dynamics, control signals, and corrupting noise.

$$\vec{x}_k = A_d\vec{x}_{k-1} + B_{ud}\vec{u}_{k-1} + \vec{w}_{k-1} \tag{3.34}$$

$$\vec{y}_k = C_d\vec{x}_k + D_d\vec{u}_k + \vec{v}_k \tag{3.35}$$

Equation 3.35 describes the measurement of the state vector $\vec{x}_k$ through a measurement matrix C_d. The D_d matrix, often called the feed-through matrix, reflects the potential for a deterministic driving signal to manifest directly in the measurement and is often taken to be identically zero. This derivation ignores the deterministic driving signal $\vec{u}_k$,which affects the system state through the B_{ud} and D_d matrices, because their inclusion does not change the estimation process and only adds unnecessary complexity. Corrupting the measurement process is a Gaussian white noise represented by $\vec{v}_k$. An important assumption in the development of Kalman filter is that the process and measurement noises are taken to be white, normal, and uncorrelated. As this is a discrete time derivation of the Kalman filter, the noise signals denoted by $\vec{w}_k$ and $\vec{v}_k$ are representative of the instantaneous noise signal values of the discrete equivalent noises constrained by the following equations.

$$E[\vec{w}_k\vec{w}_j^T] = W_k\Delta(k-j) \tag{3.36}$$

$$E[\vec{v}_k\vec{v}_j^T] = \mathbb{R}_k\Delta(k-j) \tag{3.37}$$

$$E[\vec{w}_k] = E[\vec{v}_k] = E[\vec{w}_k\vec{v}_k^T] = 0 \ \forall k,j \tag{3.38}$$

Please note that the measurement noise covariance matrix, $\mathbb{R}_k$, has been stylized in order to prevent confusion with the symbol for a rotation matrix, R_1^2. An important note to understand is that W_k and $\mathbb{R}_k$ are covariance matrices and not spectral densities, which are their continuous time analogs. Furthermore, Equation 3.38 stipulates that for all time-steps k and j the two noise signals have no correlation and that each process is zero-mean. This requirement makes the derivation of the problem much more straightforward

and simplifies calculations during filter operation. The system matrices A_d and C_d may both vary with time but, for simplicity, remain constant in this derivation.

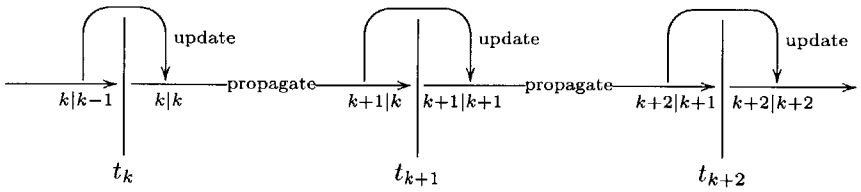


Figure 3.3: Kalman filter operation time-line[18]

Figure 3.3 depicts how the filter will work. At each time instant, t_k,t_{k+1}, and t_{k+1}, the Kalman filter incorporates a new measurement, taken at that instant, with the prior information available at that time-step. In this fashion, an a priori estimate $\hat{\vec{x}}_{k|k-1}$ is combined with new measurement data $\vec{y}_k$ to produce an optimal estimate, called the a posteriori estimate $\hat{\vec{x}}_{k|k}$. The $\hat{}$ symbol denotes the Kalman filter's estimate of the state. The Kalman filter does this blending to produce an optimal estimate in the minimum mean square error sense. At each time step we will then have an a priori estimation error $\tilde{x}_{k|k-1}$, with a related error covariance matrix $Q_{k|k-1}$, and the a posteriori equivalents given by $\tilde{x}_{k|k}$ and $Q_{k|k}$,respectively. The equations which define these quantities are given by:

$$\tilde{x}_{k|k-1} = \vec{x}_k - \hat{\vec{x}}_{k|k-1} \tag{3.39}$$

$$Q_{k|k-1} = E[\tilde{x}_{k|k-1}\tilde{x}^T_{k|k-1}] \tag{3.40}$$

$$\tilde{x}_{k|k} = \vec{x}_k - \hat{\vec{x}}_{k|k} \tag{3.41}$$

$$Q_{k|k} = E[\tilde{x}_{k|k}\tilde{x}^T_{k|k}] \tag{3.42}$$

This process happens recursively so that at each update, only the previous time step's data is needed. This recursive property makes the filter beneficial for use in systems with limited memory and computing resources. In order to complete the Kalman filter let us first assume that the blending step is a linear combination of the previous estimate and the new measurement, as shown below in Equation 3.43. The two

blending coefficients are L_k and $L_k^{'}$. Let us also assume that the first a priori estimation error is unbiased, such that $E[\tilde{x}_{k|k-1}] = 0$. We also assume that we are given an initial estimate of our state vector, $\hat{\vec{x}}_{k|k-1}$, and some sense of the quality of our initial estimate given by $Q_{k|k-1}$. The initial value of both of these quantities will not affect the stability of the filter, only the rate at which it will converge to an optimal solution. In order to recursively operate the Kalman filter, let us constrain the a posteriori estimation error to also be unbiased, $E[\tilde{x}_{k|k}] = 0$. Remember that $E[\vec{v}_k] = 0$, and that $E[\tilde{x}_{k|k-1}] = 0$.

$$\hat{\vec{x}}_{k|k} = L_k^{'}\hat{\vec{x}}_{k|k-1} + L_k\vec{y}_k \tag{3.43}$$

$$E[\tilde{x}_{k|k}] = E[\vec{x}_k - \hat{\vec{x}}_{k|k}] = E[\vec{x}_k - (L_k^{'}\hat{\vec{x}}_{k|k-1} + L_k\vec{y}_k)] = 0 \tag{3.44}$$

$$= E[\vec{x}_k - L_k^{'}\hat{\vec{x}}_{k|k-1} - L_k(C_d\vec{x}_k + v_k) = E[(I - L_kC_d)\vec{x}_k - L_k^{'}\hat{\vec{x}}_{k|k-1}] \quad \pm E[L_k^{'}\vec{x}_k] \tag{3.45}$$

$$= E[(I - L_k^{'} - L_kC_d)\vec{x}_k + L_k^{'}\vec{x}_k - L_k^{'}\hat{\vec{x}}_{k|k-1}] = E[(I - L_k^{'} - L_kC_d)\vec{x}_k] + L_k^{'}E[\tilde{x}_{k|k-1}] \tag{3.46}$$

$$E[\tilde{x}_{k|k}] = (I - L_k^{'} - L_kC_d)E[\vec{x}_k] = 0 \quad \therefore \tag{3.47}$$

$$L_k^{'} + L_kC_d \quad = \quad I \tag{3.48}$$

$$\hat{\vec{x}}_{k|k} = (I - L_kC_d)\hat{\vec{x}}_{k|k-1} + L_k\vec{y}_k = \hat{\vec{x}}_{k|k-1} + L_k(\vec{y}_k - C_d\hat{\vec{x}}_{k|k-1}) \tag{3.49}$$

Using the result from Equation 3.48 we rewrite Equation 3.43 to produce Equation 3.49. This form of the measurement update equation shows that in going from an a priori to an a posteriori estimate, we add a correction term to the a priori estimate. This correction term is the product of a coefficient L_k, and what is called the innovation. The innovation is the difference between the the measurement $\vec{y}$ at time-step k, and the predicted value of the measurement, $C_d\hat{\vec{x}}_{k|k-1}$. Assuming that all plant dynamics are modeled accurately

and that there is no measurement noise, the innovation for any time step would always be zero. In reality modeling errors and external noise will appear in the innovation. The gain L_k attenuates the innovation before being incorporated into the estimate. Intuitively, in order to produce reasonable estimates, the filter gain L_k should weight highly accurate measurements (i.e. if the determinant of the measurement noise covariance matrix $\mathbb{R}_k$ is small) more than noisy measurements. The way in which we choose this weighting matrix L_k, is therefore crucial to the proper operation of this estimator. When the gain L_k is chosen in order to minimize the error variance in a least-squares sense, it is called the Kalman gain. We, therefore, need a way to choose the Kalman gain dynamically at each time-step. Minimizing the error covariance matrix is equivalent to minimizing the sum of the squares of the estimation error vector. We define a cost function J_k whose minimization will result in the Kalman gain. Note that the inner product of a vector with itself is equal to the trace of the outer/tensor product of the same vector with itself, shown in Equation 3.50.

$$x^T x = trace[xx^T] \tag{3.50}$$

$$Q_{k|k} = E[\tilde{x}_{k|k}\tilde{x}_{k|k}^T] \qquad J_k = E[\tilde{x}_{k|k}^T\tilde{x}_{k|k}] = trace[Q_{k|k}] \tag{3.51}$$

$$L_k = argmin\, J_k \tag{3.52}$$

$$\tilde{x}_{k|k} = \vec{x}_k - \hat{\vec{x}}_{k|k} = \vec{x}_k - (I - L_k C_d)\hat{\vec{x}}_{k|k-1} - L_k(C_d\vec{x}_k + \vec{v}_k) = (I - L_k C_d)\tilde{x}_{k|k-1} - L_k\vec{v}_k \tag{3.53}$$

$$Q_{k|k} = E[\tilde{x}_{k|k}\tilde{x}_{k|k}^T] = E[((I - L_k C_d)\tilde{x}_{k|k-1} - L_k v_k)((I - L_k C_d)\tilde{x}_{k|k-1} - L_k\vec{v}_k)^T] \tag{3.54}$$

$$Q_{k|k} = (I - L_k C_d)Q_{k|k-1}(I - L_k C_d)^T + L_k\mathbb{R}_k L_k^T \tag{3.55}$$

Equation 3.53 describes how to determine the new estimation error from the previous estimation error for any gain L_k. This equation makes intuitive sense because the only change in the estimation error is

from either the measured state $L_k C_d$ or the measurement noise $L_k \vec{v}_k$. Using this equation we produce the equation for the a posteriori error covariance matrix (Eqn. 3.54). Multiplication of the cross terms $\tilde{x}_{k|k-1}$ and $L_k \vec{v}_k$ inside the expectation results in the cancellation since the prior estimation error is uncorrelated with the measurement noise at this new time step. All that remains are the squared terms which are shown in Equation 3.55. Taking the derivative of cost function J_k with respect to the gain L_k and setting the result equal to zero produces in the following expression for the Kalman gain (Eqn. 3.57)

$$\frac{\partial J}{\partial L_k} = \frac{\partial trace[Q_{k|k}]}{\partial L_k} = 0 \tag{3.56}$$

$$L_k = Q_{k|k-1} C_d^T [C_d Q_{k|k-1} C^T + \mathbb{R}_k]^{-1} \tag{3.57}$$

$$Q_{k|k} = (I - L_k C_d) Q_{k|k-1} \tag{3.58}$$

Plugging Equation 3.57 back into Equation 3.55 results in a simplified error covariance (Eqn. 3.58), though this form is less stable numerically than Equation 3.55, which is written in what is known as Joseph's form. Equation 3.55 should be used for error covariance propagation in order to avoid Q_k becoming non-symmetric (which would cause divergence in the filter) as a result of numerical precision/round off errors in calculation.

Applying the matrix inversion lemma yields the following alternate forms of the Kalman filter equations. These alternate equations can be used on measurement updates to first calculate the new error covariance matrix and then the Kalman gain. Conversely Equation 3.55 requires that the Kalman gain is calculated first using Equation 3.57.

$$Q_{k|k} = (Q_{k|k-1}^{-1} + C_d^T \mathbb{R}_k^{-1} C_d)^{-1} \tag{3.59}$$

$$L_k = Q_{k|k} C_d^T \mathbb{R}_k^{-1} \tag{3.60}$$

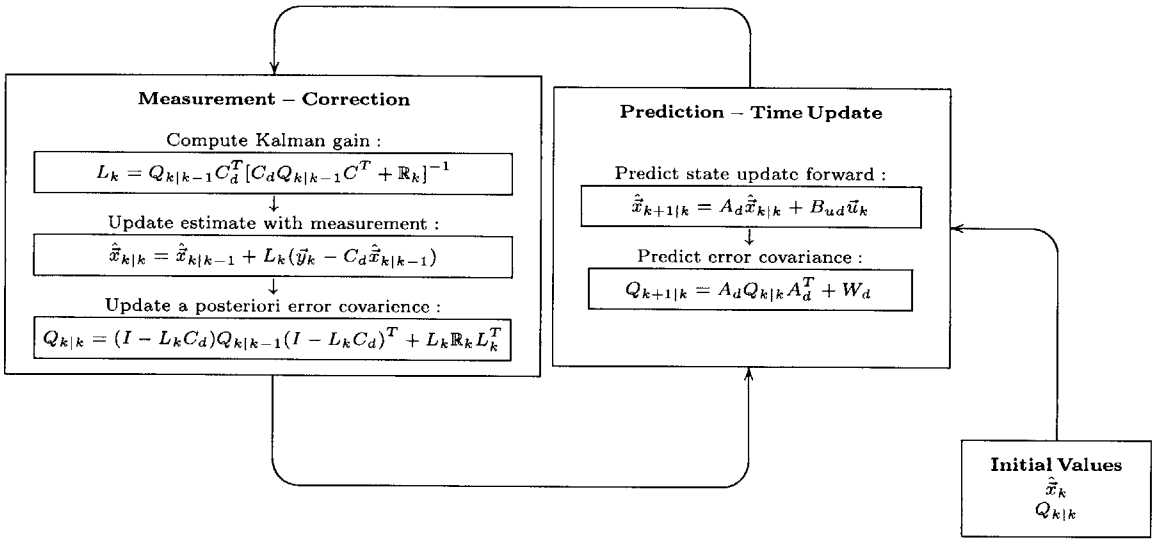


Figure 3.4: Prediction-Correction Cycle of the Kalman Filter

Figure 3.4 depicts the Kalman filter cycle starting from some initial state and then oscillating between a prediction and a correction state. Some intuition for the operation of the Kalman filter can be established by way of an analogy. Imagine the Kalman filter is tracking the velocity state of a car traveling down the road. The dynamics matrix A_d encodes the necessary information to achieve the new velocity of the car given the old one (i.e. Newton's first law, ignoring friction and other dynamics). The driving function $\vec{u}_k$ could be the accelerator pedal. Measurements $\vec{y}_k$would reflect the speedometer output with some corrupting noise $\vec{v}_k$ added. Since the car is physically traveling down a road subject to real world phenomena, we can imagine that car's process noise $\vec{w}_k$ results from gusts of wind or speed bumps in the roads. Propagation of our system's dynamics forward in time, between the speedometer measurements, is directly dependent on the accuracy of our system matrix A_d. If we know that our model is not very accurate we can increase the processes noise covariance matrix W_k in order to make the filter aware of the inaccuracies in our model. This will cause the filter to weight the measurements more heavily than the previous state during a measurement update step and increase the error covariance matrix to account for the uncertainty during propagation steps. Conversely, if the speedometer is particularly noisy and unreliable, the Kalman gain values the previous state more than the incoming information from the measurement $\vec{y}_k$.

The following list is a summary of the common assumptions used in the preceding derivation of the Kalman filter:

1. The measurement update step is linear
2. The process noise $\vec{w}_k$ and the measurement noise $\vec{v}_k$ are both Gaussian, white, and uncorrelated
3. The initial estimate is unbiased, $E[\tilde{x}_{k|k-1}] = 0$
4. Initial state $\hat{\vec{x}}_{k|k-1}$ and error covariance $Q_{k|k-1}$ are known

3.4 The Indirect State Feedback Kalman Filter

3.4.1 One Dimensional Unaided Inertial Navigation System

In order to motivate the need for a state estimation algorithm when using MEMS sensors, a simple example is given below [12]. Consider a One-Dimensional (1-D) dynamic system wherein an object is only free to move linearly on a rigid frictionless track subject to external forces in the direction of the track. In the body of the object is a single-axis accelerometer that measures acceleration, $a(t)$, in the axis of travel. First it will be shown that the accelerometer alone is not sufficient to uniquely identify position $p(t)$ and velocity $v(t)$. Subsequently the concept of position aiding will be introduced. Finally a MEMS accelerometer model will be given for which position aiding can yield a converging state estimator. The equations of motion of the 1-D constrained system are as follows:

$$\dot{p}(t) = v(t) \tag{3.61}$$

$$\dot{v}(t) = a(t) \tag{3.62}$$

The resulting unforced linear state space representation of this system, if given for the state vector $\vec{x}(t) = [p, \dot{p}, \ddot{p}]^T = [p(t), v(t), a(t)]$, is given by:

$$\dot{\vec{x}}(t) = A\vec{x}(t) + B_w\vec{w}(t) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \vec{x}(t) + B_w\vec{w}(t) \tag{3.63}$$

$$E[\vec{w}(t)] = 0 \quad E[\vec{w}(t)\vec{w}^T(\tau)] = W_c\delta(t-\tau) \tag{3.64}$$

In this derivation it is assumed that there is no corrupting process noise in the system such that $W_c = 0$. Assume that perfect acceleration measurements are available such that the system output relationship is given by Equation 3.65 for the measurement matrix $C = [0\ 0\ 1]$:

$$y(t) = C\vec{x}(t) = a(t) \tag{3.65}$$

Given the discrete nature of accelerometer measurements we will transform our system into a discrete time equivalent in which the continuous time values are sampled at regular intervals kT_s, with sampling period T_s. The discrete state space representation of the time evolution of $\vec{x}_k$ is given below in Equation 3.72. Below are the necessary equations to transform a system from continuous time to discrete. Please refer to [18, 12, 19] for other examples of this discretization. The discrete time state transition matrix is given by Equation 3.66.

$$A_d = e^{AT_s} \tag{3.66}$$

$$W_d = \int_0^{T_s} A_d B_w W_c B_w^T A_d^T d\tau \tag{3.67}$$

If there were process noise in this problem, Equation 3.67 could return the discrete time equivalent noise covariance matrix. Alternatively, the discrete equivalent state space propagation matrix A_d and noise covariance matrix W_d can be derived using the following equations [18, p7-18].

$$S = \begin{bmatrix} -A & B_w W_c B_w^T \\ 0 & A^T \end{bmatrix} \tag{3.68}$$

$$C = e^{ST_s} = \begin{bmatrix} C_{11} & C_{12} \\ 0 & C_{22} \end{bmatrix} \tag{3.69}$$

$$A_d = C_{22}^T \tag{3.70}$$

$$W_d = C_{22}^T C_{12} = A_d C_{12} \tag{3.71}$$

The matrix A is the 3x3 matrix governing the continuous time derivatives in Equation 3.63. The measurement equation of the discrete system is identical to that of the continuous time system such that $C_d = C = [0\,0\,1]$.

$$\vec{x}_{k+1} = A_d\vec{x}_k = \begin{bmatrix} 1 & T_s & \frac{T_s^2}{2} \\ 0 & 1 & T_s \\ 0 & 0 & 1 \end{bmatrix} \vec{x}_k \tag{3.72}$$

We now have a foundation on which to build an estimator for the position, velocity, and acceleration of a 1-D object given, temporarily, perfect acceleration measurements. It should be immediately evident that there is a issue with using accelerometer data to determine position and velocity. Directly integrating acceleration to determine velocity yields one constant of integration to account for the initial velocity of the system. A second integration to determine system position requires knowledge of both the previous constant and yields yet another constant of integration for the initial position. This thought experiment illustrates the point that acceleration alone is insufficient to uniquely determine, or observe, the absolute state of the system. Construction of the so-called observability matrix allows for the diagnosis of such observability problems in state space systems even when they are not evident by inspection. The observability matrix for this system is shown in Equation 3.73.

$$\mathcal{O} = \begin{bmatrix} C_d \\ C_dA_d \\ C_dA_d^2 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.73}$$

The rank of the observability matrix is, effectively, a measure of the number of states that can be uniquely observed given the state and output matrices of the system. The observability matrix in this case is rank one, which is consistent with the intuitive explanation of limited observability given above. Ultimately, the position and velocity cannot be determined form acceleration without perfect initial conditions and measurements. If instead position measurements are available the output matrix becomes $C = [1\,0\,0]$ which results in an observability matrix of full rank as shown in Equation 3.74. All three states of the system are observable from position measurements, though it can be shown that at least two position measurements will be needed to determine velocity and three will be needed to determine acceleration.

$$\mathcal{O} = \begin{bmatrix} C_d \\ C_d A_d \\ C C_d A_d \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & T_s & \frac{T_s^2}{2} \\ 1 & 2T_s & 2T_s^2 \end{bmatrix} \tag{3.74}$$

The value of using position measurements to determine system state is clear. However, a basic principle of sampling theory tells us that in order to capture and control the higher-frequency dynamics of an aircraft the system must sample and respond to the vehicle state at a rate that is twice the primary frequency of the critical aircraft dynamics [32]. For even a very stable and docile aircraft the lower bound on the sampling and control frequency can be in the tens of Hertz. For high rate dynamics, such as those of a μUAV, the rate at which position measurements would need to be taken, as determined by the sampling period T_s, will generally need to be in the hundreds of Hertz. The most common, convenient, and widespread means of position measurement for an aircraft is GPS for which most commercial solutions can only update at a rate of 1-10Hz, which is far below the sampling requirement of the system. Conversely, modern MEMS accelerometers can operate as speeds upwards of 1000Hz. Therefore an effective inertial navigation system will be required to leverage the benefits of both fast and noisy MEMS sensors and the slower and more accurate sensors such as GPS. Such a system is sometimes referred to as a complimentary filter. Madgwick, Euston, and Farrell all give examples of similar systems which leverage complimentary Kalman filters [23, 11, 12]. Madgwick develops a similar INS to the one presented here, specialized for purposes of tracking human motion. Section 3.4.2 builds such a system where the benefits of both measurement systems are combined to produce a robust estimator which accounts for real world phenomena in sensor measurements and can handle dynamic system environments.

3.4.2 Indirect Filter Derivation

The benefits of implementing a Kalman filter as described in Section 3.3 rely heavily on the accuracy of the system matrices A_d,B_{ud},C_d, and D_d. Unlike the majority of the aerospace industry, wherein significant time and effort is expended to develop increasingly accurate system models, accurate matrices to describe the dynamics of new UAS prototypes are not often available. By using the Kalman filter as described above, in its direct state form, the matrices needed to compute new estimates at each time step can become quite large. For example, at a minimum, an aircraft needs to track six degrees of freedom, three in attitude (roll, pitch, and

yaw) and another three in translation in the three spatial dimensions. From first principles, if no constraints are placed on motion, every degree of freedom adds the equivalent of a second-order differential equation of motion to the system dynamics and requires two first-order states in the state space representation. As such, the approximate lower bound on the number of states required to represent an aircraft is twelve. Moreover, even if motion constraints allow for a reduction in the number of required dynamic states, additional states will be required to track the offset biases in the sensors. Inverting a 12x12 matrix (as would be required to calculate the error covariance matrix) is a very processor intensive task. Finally, an assumption made in the mathematics of the discrete-time Kalman filter is that all of the measurements are available at the same time. The resulting filter operates at a fixed period defined by T_s, and it is therefore difficult to incorporate measurements from sensors which operate at different rates without recalculating the discrete equivalent noise and propagation matrices at each time step.

An alternative to using the full-state approach described above is the indirect/complimentary form of the Kalman filter. An indirect architecture involves the filter tracking error states as opposed to the true states of the system. For example, instead of tracking position p, an indirect filter would instead track $\delta p = p - \hat{p}$ where p is the true position and $\hat{p}$ is the estimate of position. The complicated dynamic of an aircraft get replaced by the error state dynamics which are governed by physical laws (i.e. $v = \frac{dp}{dt}$, $a = \frac{dv}{dt}$). This concept of dynamics model replacement resembles the system described in Equation 3.63. However, had the car analogy contained real phenomenon like friction, coupling of systems, aerodynamic effects, engine dynamics, etc., the state dynamics matrix A would have been much more complicated. It was simplified for the sake of the reader, but its resemblance to the idea discussed here should not be confused as anything more than a coincidence. The construction of an indirect/complimentary filter is only concerned with the error state dynamics, which will take the form of general derivatives. The difference should be made clear in the following section. An example of indirect feedback structure filter is shown in Figure 3.5.

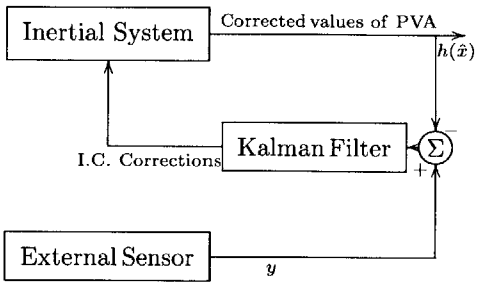


Figure 3.5: Indirect Feedback Architecture[18]

An inertial system continuously integrates high-rate sensors, like MEMS accelerometers, during $t_{k-1} < t < t_k$ to produce values for position and velocity. It keeps track of the each state's initial conditions after each integration step. At a slower periodic rate, when $t = t_k$ an external sensor provides an accurate measurement $\vec{y}_k$ of one of the lower order states, like position. At the slower measurement rate, the difference between the inertial system estimate and the external sensor measurement is presented to the Indirect Kalman Filter (IKF). The output of the IKF is finally fed into the inertial system as an initial conditions correction. This structure lends itself well to the complimentary nature of the accelerometer/GPS systems alluded to in the earlier example system. Furthermore, should the external sensor experience periods of faulty operation, the inertial system will still provide viable, if slowly drifting, dynamic estimates of the state. An alternative form is the feed-forward architecture, where the corrections to the inertial system are added to the output of the inertial system to produce the corrected result. A feed-forward IKF would produce similar results to the feed-back IKF. However, by feeding the corrections back into the inertial system the error states of the IKF will be bounded, whereas a feed-forward IKF could have unbounded error states.

Continuing the 1-D example proposed in Section 3.4.1, a model which accounts for the real world output characteristics of a MEMS sensor is incorporated into the example framework below. Equation 3.75 shows how the MEMS accelerometer measurement $a_m(t)$ of the true acceleration $a(t)$ is corrupted by some random bias signal $b(t)$ and some Gaussian white noise $v_a(t)$. The measurement noise v_a has a variance of $\sigma^2_{v_a}$. The bias signal is taken to be the result of a random walk process where $\dot{b} = \omega_b(t)$ and where ω_b is also a Gaussian white noise with its own variance of $\sigma^2_{\omega_b}$.

$$a_m(t) = a(t) - b(t) - v_a(t) \tag{3.75}$$

In order to make use of the accelerometer measurements $a_m(t)$ an estimate of the bias must be calculated

in order to attempt to cancel the offset before it is integrated into the state estimate. Therefore, the state vector given in Section 3.4.1 will be altered to track this time-varying bias signal. In the new state vector $\vec{x} = [p, v, b]^T$ the bias state replaces the acceleration state. The construction in Section 3.4.1 derived the equations necessary to describe an accelerating object, where the position, velocity, and acceleration were useful. Now that we want to develop a state estimator for that system, we must realize that keeping acceleration in the state vector doesn't make intuitive sense. There is no way for the system to predict/estimate what the acceleration will be since it is arbitrarily driven by some external force. The complimentary structure instead feeds acceleration measurement/state information to a bank of pure integrators. Given a bias estimate $\hat{b}(t)$, the best estimate of the true acceleration is given by $\hat{a}(t) = a_m(t) + \hat{b}(t)$. The resulting dynamics can be written as shown below [12]. Note that the derivative of the bias estimate is zero since it is driven by Gaussian white noise, whose expectation is zero, $E[\omega_b] = 0$.

$$\dot{\hat{p}}(t) = \hat{v}(t) \tag{3.76}$$

$$\dot{\hat{v}}(t) = \hat{a}(t) = a_m(t) + \hat{b}(t) \tag{3.77}$$

$$\dot{\hat{b}}(t) = 0 \tag{3.78}$$

The error states are defined as: $\delta p = p - \hat{p}$,$\delta v = v - \hat{v}$,$\delta b = b - \hat{b}$. The resulting full state space realization is shown below in Equation 3.79. In this system we are only using the accelerometer measurements to track position, velocity, and accelerometer bias. In order to simplify the analysis, we will again discretize the continuous system to form equivalent discrete system matrices. The discrete state transition matrix A_d is formed as shown previously in Equation 3.66. Note B_w is the continuous process noise matrix which maps the Gaussian noise vector $\vec{w}(t)$ into the state equations.

$$\delta\dot{\vec{x}}(t) = \begin{bmatrix} \delta\dot{p} \\ \delta\dot{v} \\ \delta\dot{b} \end{bmatrix} = A\delta\vec{x}(t) + B_w\vec{w}(t) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \delta p \\ \delta v \\ \delta b \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_a(t) \\ \omega_b(t) \end{bmatrix} \tag{3.79}$$

$$A_d = \begin{bmatrix} 1 & T_s & \frac{T_s^2}{2} \\ 0 & 1 & T_s \\ 0 & 0 & 1 \end{bmatrix} for\, \delta x_k = \begin{bmatrix} \delta p_k \\ \delta v_k \\ \delta b_k \end{bmatrix} \tag{3.80}$$

Given some initial error covariances (Q_{p0}, Q_{v0}, Q_{b0}) the discrete error covariance of this system is defined by Equation 3.81. This equation relates the initial state uncertainty to future state uncertainty subject to noise. In a normal system the $A_d Q_k A_d^T$ term is a contractionary term which decreases the error covariance between time-steps, assuming A_d is stable. The second integral term is an expansionary term which accounts for the noise and uncertainty introduced into the system at every time-step. Unfortunately, because of the lack of any observation of the system state, the error covariance grows with time [12, p158]. The resulting covariance matrix for any arbitrary time after initialization is shown in Equation 3.81. The error covariance of all three states grows unbounded in time since the state dynamics are not stable and the process noise continuously adds uncertainty. The initial error covariance matrix is a diagonal 3x3 matrix $Q_0 = diag(Q_{p0}, Q_{v0}, Q_{b0})$. The growth of each of these components is shown in Equations 3.82-3.84.

$$Q_{k+1} = A_d Q_k A_d^T + W_d = A_d Q_k A_d^T + \int_0^{T_s} (A_d B_w \begin{bmatrix} \sigma_{v_a}^2 & 0 \\ 0 & \sigma_{\omega_b}^2 \end{bmatrix} B_w^T A_d^T)\, d\tau \tag{3.81}$$

$$Q_p(t) = (Q_{p0} + Q_{v0} t^2 + Q_{b0} \frac{t^4}{4}) + (\frac{\sigma_{v_a}^2 t^3}{3} + \frac{\sigma_{\omega_b}^2 t^5}{20}) \tag{3.82}$$

$$Q_v(t) = (Q_{v0} t + Q_{b0} t^2) + (\sigma_{v_a}^2 t + \frac{\sigma_{\omega_b}^2 t^3}{3}) \tag{3.83}$$

$$Q_b(t) = Q_{b0} + \sigma_{\omega_b}^2 t \tag{3.84}$$

Equation 3.73 demonstrated that when tracking $[p, v, a]$ with acceleration measurements the system is not fully observable without perfect prior information and perfect measurement data. Even with initial condition information and initial uncertainty, the presence of non-ideal measurements in the form of bias and white noise causes the system to diverge in all three states. In order to ensure the error covariance remains bounded, some observation of the system state must be made, like a position measurement for example. Consider a discrete position measurement $p_m(kT_s) = p(kT_s) + v_p(kT_s)$, where v_p is a Gaussian noise process

with variance $\mathbb{R}_p$. As mentioned earlier the inertial system will continuously integrate accelerometer data at a high rate, called f_{int}, while the IKF will operate at a rate equal to that of the external sensor measurement, called $f_s = \frac{1}{T_s}$. Equation 3.85 details how the aiding measurement is formulated for the IKF. The current estimate of the position is subtracted from the new measurement to form an error measurement. The measurement matrix is given by $C_d = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$.

$$\delta y_k = p_m - \hat{p}_k = p(kT_s) + v_p(kT_s) - \hat{p}_k = \delta p_k + v_p(kT_s) = C_d \delta \hat{\vec{x}}_k + v_p(kT_s) \tag{3.85}$$

At the time of filter initialization $\hat{\vec{x}}_0 = E[\vec{x}(0)]$ therefore the initial error state is also zero, $\delta \vec{x}_{0|0} = 0$. The initial error covariance $Q_{0|0} = var(\delta \vec{x}_{0|0}) = var(\vec{x}(0))$. The first aiding measurement occurs at $k = 1$, so the error covariance and error state will need to be propagated in time as described in Figure 3.4.

$$\delta \vec{x}_{1|0} = A_d \delta \vec{x}_{0|0} = 0 \tag{3.86}$$

$$Q_{1|0} = A_d Q_{0|0} A_d^T + W_d \tag{3.87}$$

The measurement update process incorporates the new measurement δy_1 into the existing state estimate. Since the prior estimate of the state error $\delta \vec{x}_{0|0} = 0$, the equation simplifies as follows in Equation 3.88. Equation 3.89 describes the update of the true state $\hat{\vec{x}}$ with the new correction provided by $\delta \vec{x}$. The new updated state is equal to the expected value of the state, $\hat{\vec{x}}_{1|1} = E[\vec{x}(t_1)]$, after incorporating the measurement information at step $k = 1$. Therefore, after the update step, the new estimate of the state error is zero $\delta \vec{x}_{1|1} = 0$, such that the propagation of the error state will be zero again, $\delta \vec{x}_{2|1} = A_d \delta \vec{x}_{1|1} = 0$.

$$\delta \vec{x}_{1|0} = \delta \vec{x}_{0|0} + L_k(\delta y_k - C_d \delta \vec{x}_{0|0}) = L_k \delta y_k \tag{3.88}$$

$$\hat{\vec{x}}_{1|1} = \hat{\vec{x}}_{1|0} + \delta \vec{x}_{1|0} = \hat{\vec{x}}_{1|0} + L_k \delta y_k \tag{3.89}$$

The remainder of the Kalman filter equations concerning the error covariance update and propagation, as previously derived, still apply. The critical differences between the general Kalman filter and the IKF, outside of structure, are that the state propagation step becomes trivial and the measurement update step is

simplified by allowing the filter to ignore the previous error state estimate since that information has already been incorporated.

3.4.3 Simulation

Several 1-D simulations were preformed with accelerometer and gyroscope models to validate and assess the performance of the IKF. For the simulation, the rate of the IMU accelerometer was set to $f_{int} = 120\text{Hz}$ while the position/velocity aiding sensor reported data at $f_s = 5\text{Hz}$. The simulated aiding measurements are modeled on the expectation of a GPS-like update in the form of a position corrupted by white noise with $\sigma_p^2 = 3\text{m}$, and velocity corrupted by a white noise with $\sigma_v^2 = 0.0400\frac{\text{m}}{\text{s}}$ [29]. The incorporation of the velocity measurements changes the measurement matrix to $C_d = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$. The accelerometer model is based off of Equation 3.75 where $\sigma_{v_a}^2 = 2.5 \cdot 10^{-3}\frac{\text{m}^2}{\text{s}^3}$ and $\sigma_{\omega_b}^2 = 1.0 \cdot 10^{-6}\frac{\text{m}^2}{\text{s}^5}$. The true bias offset of the accelerometer is set to $0.05\frac{\text{m}}{\text{s}^2}$. The simulation tracks a particle moving, subject to sinusoidal acceleration, for 100 seconds (Fig. 3.6).

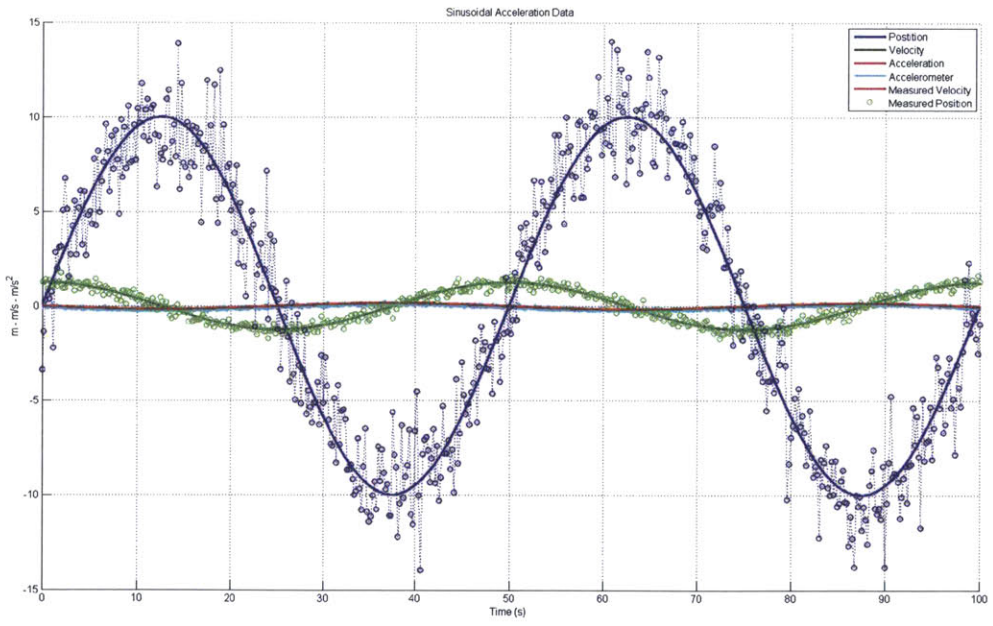


Figure 3.6: Sinusoidal Simulation Results

The corruption of the position and velocity aiding measurements can be seen in the scatter of blue and green markers, respectively, around the true signals, which are indicated by matching solid-colored lines. The accelerometer signal, plotted in cyan, can be seen to track the red acceleration data. Note that the accelerometer signal is always below the true acceleration due to the bias offset. Below are the plots of the resulting filtered data.

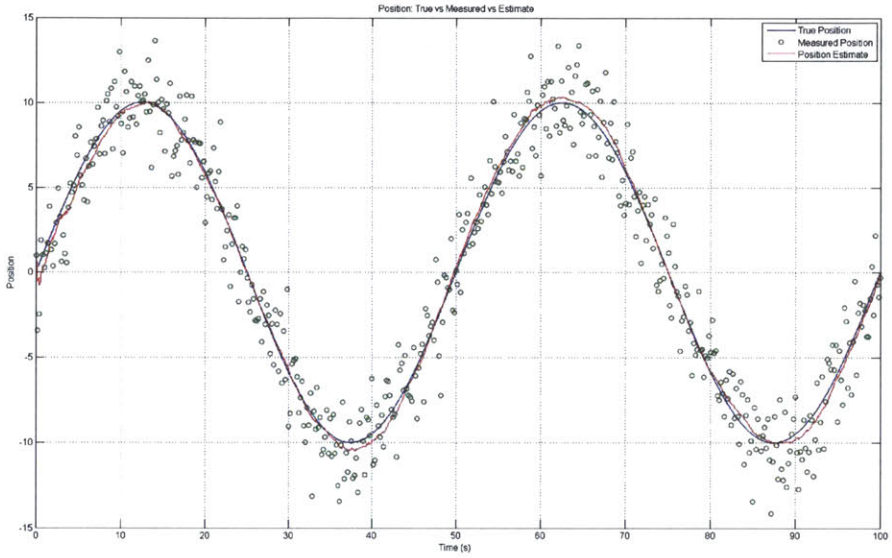


Figure 3.7: Position Results

Figure 3.7 shows the results of the IKF and how it blends the noise position and velocity data with its total state estimate. Initially, for $t \in [0, 10]s$, the filter lacks a good estimate of the accelerometer bias, so the integration of the accelerometer data results in the position plot containing an non-trivial error. This error slowly decays away as the filter's estimate of the bias improves. Figure 3.8 shows the time evolution of the bias estimate, whose true value was specified in the simulation to be $0.05\frac{\mathrm{m}}{\mathrm{s}^2}$. It is clear that after $t = 10s$ the estimate is accurate to within a few $\frac{\mathrm{mm}}{\mathrm{s}^2}$.

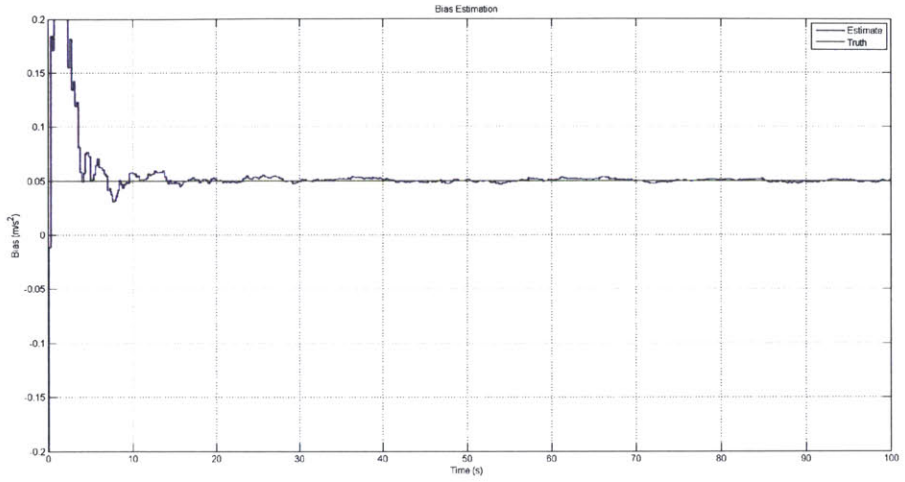


Figure 3.8: Bias Results

Figure 3.9 depicts the state errors and related one-standard deviation bounds on the the error uncertainty for each state. Both reach steady-state values after $10s$. Given the performance characteristics of the simulated sensors, the IKF is able to attain sub-half meter accuracy in position and sub-quarter meter-per-second accuracy in velocity. The bias estimation also produces an estimate with a steady state standard deviation of less than $0.025\frac{m}{s^2}$.

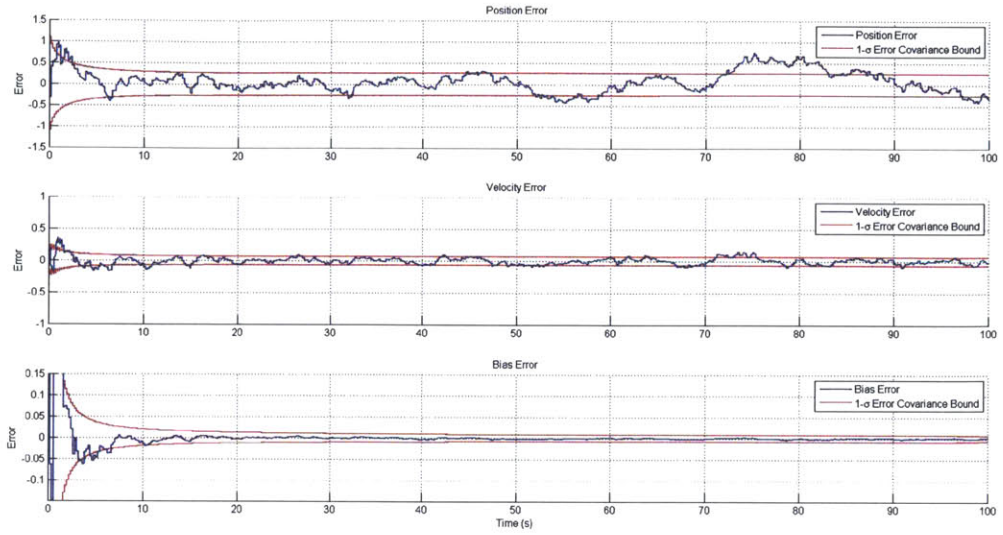


Figure 3.9: Estimate Errors and 1-σ Bounds on Error Covariance

Figure 3.10 depicts the auto-correlation plots of the position and velocity innovation processes, showing that they are both uncorrelated. This means that the filter is extracting all relevant information from each sample. This result is equivalent to having the only remaining data in the innovation be the white noise corrupting the measurement signals (note that the auto-correlation of white noise is an impulse at $t = 0$).

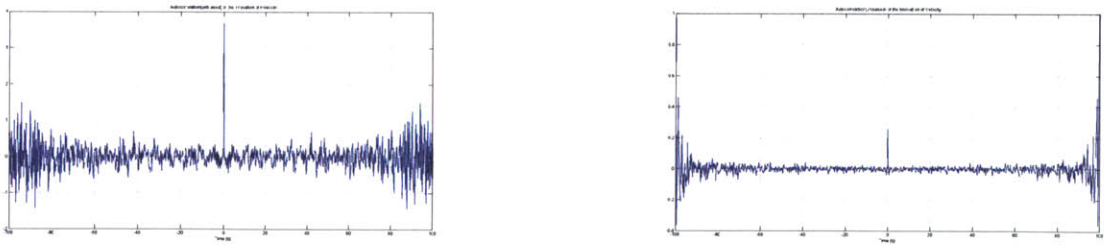


Figure 3.10: Auto-correlation of Position and Velocity Innovation Processes

A plot of the Kalman gains is shown below in Figure 3.11. The Kalman gain matrix, a 3x2 matrix, is calculated on every cycle of the IKF. Note, however, that the components of the gain matrix reach their steady state value within about six seconds of filter operation. Computational burden can be mitigated on the target platform by using a pre-computed steady state Kalman gain instead of inverting the necessary

matrices on each time step. Doing so results in a loss of accuracy and a slower transient response from poor initial conditions, but may be beneficial when considering the computational load on a system.

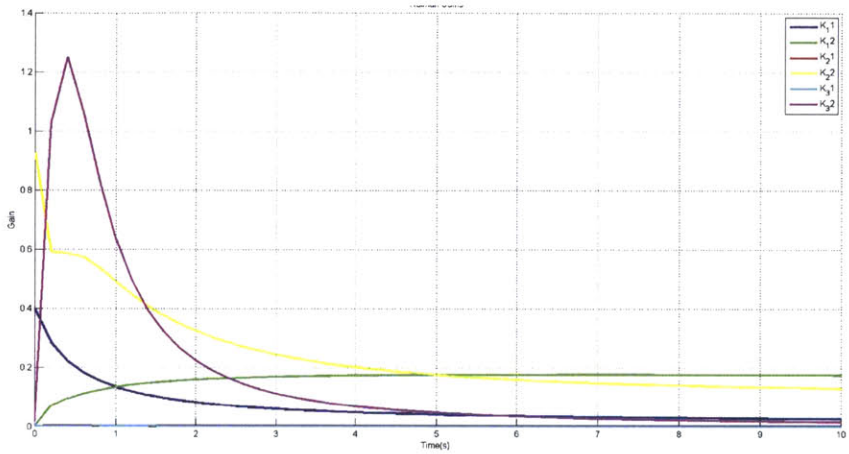


Figure 3.11: 1-D Kalman Gains

Table 3.1: 1-D Indirect Kalman Filter RMS Errors

State	RMS Error
Position (m)	0.251
Velocity ($\frac{m}{s}$)	0.049
Accelerometer Bias ($\frac{m}{s^2}$)	0.0021

The simulation was run 100 times to produce averages of the RMS errors on all three state which are shown in Table 3.1. The results reflect the average RMS errors of the IKF after the 15 second start-up transient when the initial bias estimate was produced. Given the realistic values for the noises used in the simulation, the IKF performs well given a dynamic tracking environment. This IKF forms the building block of the navigation filter inside the INS designed for the µUAS.

Chapter 4

Navigation Development and Testing

To operate autonomously a UAS needs to be able to discern its instantaneous position and orientation relative to some fixed reference frame. An important concern which drives the design of the INS in this case is the need for the INS algorithm to eventually be implemented on a processing-constrained micro-controller. Architectures explored by [34, 31, 19, 6] highlight the importance of designing the INS with this implementation detail in mind. For this reason we consider not just the effectiveness of a particular architecture but also the required computational burden. There are, of course, many different approaches to solving this navigation problem.

To start, we first make a distinction between the problems of solving for the aircraft's orientation and determining the aircraft's position and velocity in some frame. We refer to any system which handles the attitude estimation as an Attitude and Heading Reference System (AHRS) and to any system which determines the location and velocity of the UAS in the navigation coordinate system as a Navigation Filter (NF). Some INS solutions combine the NF and AHRS together into one computational block while others separate them. Korka [19] describes the implementation of a combined complimentary architecture where the AHRS and NF exist together as one filter. The filter still implements a complimentary architecture, however, wherein the inertial integration is still separate from state estimation. Stimac [34] presents a more complete architecture which models rigid-body dynamics and employs a single large estimation structure which incorporates the AHRS and NF along with the inertial integrators. The drawback of such a system is the computational burden of inverting the large matrices needed to propagate the system state. Furthermore,

the tracking of rigid-body dynamics is only practically useful in exo-atmospheric situations where external force and torque disturbances are kept to a minimum [34, p151]. Roumeliotis et al. [31, 4] and Cutler [6] both describe the implementation of INS which split the operation of the a complimentary AHRS and a complimentary NF for robustness and speed. Specifically, by keeping the integrated inertial systems isolated from the correction systems, the integrated systems are capable of operating even in the event of measurement loss, like short-term GPS blackout or magnetic field disturbances. Increased robustness and computational simplicity comes at a cost, however, to optimality and accuracy. Because the AHRS and NF are separated, and therefore share no internal information, they cannot offer aiding information to one an other, which in a combined architecture would happen naturally and generally reduce estimation errors. Another concept of INS operation, called a tightly coupled solution, exists where GPS measurements are incorporated into the state estimator [34]. A tightly-coupled solution uses the GPS raw pseudo-range measurements and constructs its own estimate of position and velocity while incorporating other sensor data. A loosely-coupled solution, which is what is presented here, takes GPS measurements only as final/whole data products.

In this section we will provide an overview of the implemented INS architecture along with discussion and testing of the separate AHRS and NF components. All of the simulations and INS results were carried out in MATLAB to minimize the development time of INS algorithms. Special attention was paid to minimizing the use of MATLAB-specific functionality and algorithms which might be difficult to port to a micro-controller. Several tools from the Aerospace toolbox, like functions to transform between the ECEF, NED, and LLA frames were used to speed up development. These algorithms are well documented and available for custom implementation in C. However, the micro-controller still needed to be configured and programmed to perform the necessary data collection and recording processes for testing.

4.1 Architecture

The architecture settled upon for the Locust INS was inspired by Farrell and by the work of the MIT Aerospace Controls Lab [6, 12, p.19]. In the interest of designing the INS with portability to a micro-controller, the INS was first split into separate AHRS and the NF sections. This splitting produces sub-optimal results compared to a single larger filter, as discussed previously, but is much easier to compute numerically due to the reduced state vector sizes. Furthermore, both the AHRS and NF were built as complimentary filters in order to take advantage of high rate MEMS sensors (>100Hz) and slower GPS and

heading aiding measurements ($\leq$ 5Hz) . The problem of solving the NF can be further simplified with the understanding that motion in all three axes are independent from one another in this framework. Therefore, the NF is further subdivided into three separate IKFs tracking motion in each of the three navigation frame axes. The AHRS cannot be subdivided in the same manner due to the inherent interdependence of rotations in R^3, as previously discussed. This problem was briefly described in Section 3.2.1 and is the subject of much research. The reader should consult [20] for a thorough exploration into the topic of rotation on the SO(3). A brief explanation of the figure below will be given, followed by discussions of the NF and AHRS in isolation.

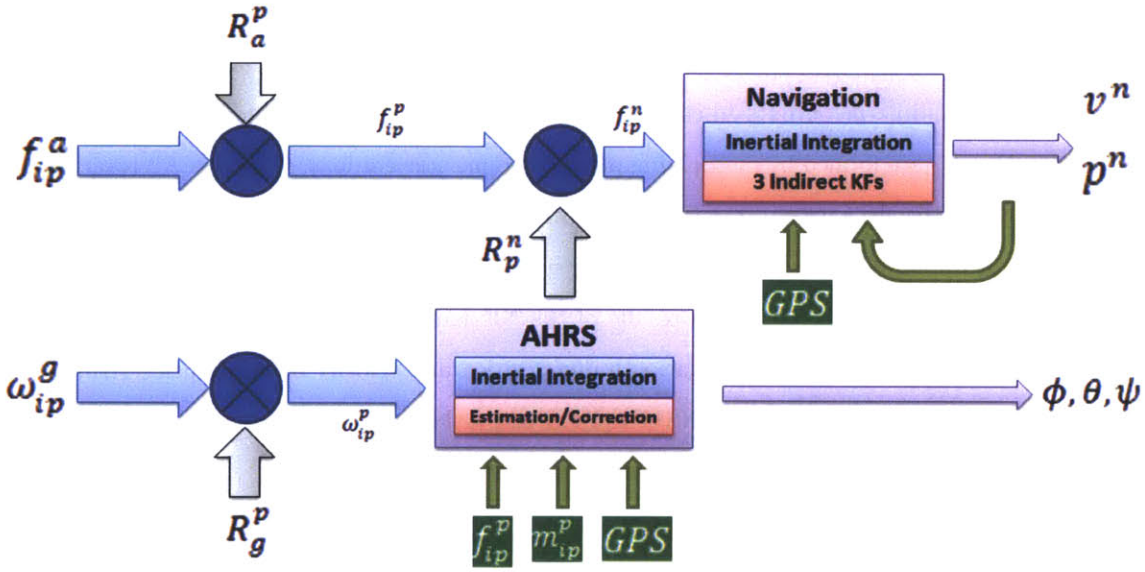


Figure 4.1: INS Block Diagram

In the above figure inertial data is represented by blue lines, rotation matrices by gray lines, aiding data by green lines, and result data by purple lines. In an effort to minimize complexity and computational load, the INS is split up into two separate segments, the Attitude and Heading Reference System (AHRS) and the Navigation Filter (NF). The NF is a type of complimentary filter which work in conjunction with inertial integrators, shown in blue, and three IKF state estimators, shown in red, and GPS to produce estimates of position and velocity. The AHRS is a simpler complimentary design which instead uses a static filter to estimate signal bias and correct the INS attitude estimates of roll, pitch, and yaw. Following the notation of Farrell, the inputs to the INS, f_{ip}^a and ω_{ip}^g, appear on the left side of Figure 4.1 [12]. The

MEMS accelerometer output f_{ip}^{a}, represents the platform acceleration relative to an inertial frame, resolved in the accelerometer frame. Similarly, the gyroscopes measure ω_{ip}^{g}, the platform rotation rate relative to an inertial frame, resolved in the gyroscope frame. The two most important frames-of-reference used in most INS are the platform frame (which is defined according to the body axes of the aircraft) and the navigation frame (which is defined by some convenient local inertial frame). The rotation matrices R_a^p and R_g^p rotate the inertial measurements from their respective sensor frames into the platform frame, accounting for installation misalignment between the IMU and the frame of the aircraft. The platform frame gyroscope readings ω_{ip}^{p} are passed into the AHRS which performs the necessary integration and correction steps on the three Euler angles/unit quaternion to produce an accurate estimate of the rotation matrix R_p^n, which relates the platform frame to the navigation frame. The AHRS system uses accelerometer data, magnetometer data, GPS, and previous estimator outputs to correct the rate-integrated vehicle attitude estimate. Navigation frame acceleration f_{ip}^{n} are formed by multiplying platform frame accelerations f_{ip}^{p} by the R_p^n rotation matrix from the AHRS. The result is passed into the NF to produce estimates of the vehicle's position and velocity in the navigation frame. Current position and velocity data along with GPS are used to correct the NF estimates. The INS outputs the combined results of the NF and the AHRS, which describe the aircraft's Euler-angle attitude and the current navigation-frame position and velocity of the vehicle. Armed with a high-confidence state estimate from the INS, the FMS can make use of flight stabilization techniques to control the aircraft.

An important note about the INS inputs is that they are assumed to be pre-compensated for scale factor/linearity errors. Typically, MEMS sensors suffer from scale factor errors, which manifest as non-unity slopes in their input/output relationship and nonlinearities over the range of sensor operation. The INS presented here is only capable of tracking the bias signals for the MEMS inertial sensors and assumes that any scale factor correction/linearization has already happened. Farrell describes how to design a more complex INS which is capable of tracking these additional corrupting effects, though there will necessarily be more equations to solve as well as a concern about observability[12]. Section 2.3.1 outlines the method used to correct for scale factor errors and predict initial bias estimates in the MEMS accelerometers. Section 2.3.2 describes how to correct for scale factor errors and in the MEMS gyroscopes. The magnetometer sensors require their own special compensation due to the fact that their behavior is a function of their proximity to other metallic, and therefore potentially magnetized materials. Section 2.3.3 describes how to handle these effects, called Hard and Soft Iron effects. Another assumption made is that all measurement corrupting noise

is uncorrelated or white.

4.2 Navigation Filter

The navigation filter is comprised of three parallel IKFs running on each channel/axis of the navigation frame. The filters, like the ones described in Section 3.4.2, each track their channel's error in position, velocity, and a navigation frame accelerometer bias. This bias error estimate is the projection of the body frame biases into the navigation frame, so in order to use the estimates they must be rotated back from navigation frame to the platform platform. The simplicity of the architecture, due to the fact that motion in one of the axes is independent to motion in another perpendicular axis, lends itself to the implementation on a μUAS.

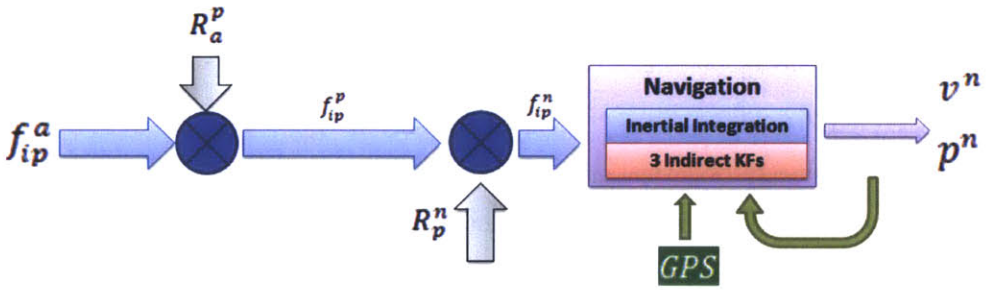


Figure 4.2: Navigation Filter Block Diagram

Using the notation provided in Section 3.4.2, the equations which govern the NF are summarized below, where the first subscript refers to the navigation frame axis/channel and the second refers to the time step. Equation 4.1 shows the process model for the North-axis, which is identical to the East and Down axes which are, therefore, not repeated.

$$\delta x_{N,k} = \begin{bmatrix} \delta p_{N,k} \\ \delta v_{N,k} \\ \delta b_{N,k} \end{bmatrix} = A_d \delta x_{N,k-1} + W_d = \begin{bmatrix} 1 & T_s & \frac{T_s^2}{2} \\ 0 & 1 & T_s \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \delta p_{N,k-1} \\ \delta v_{N,k-1} \\ \delta b_{N,k-1} \end{bmatrix} + W_d \tag{4.1}$$

$$\delta y_{N,k} = C_d \delta \hat{\vec{x}}_{N,k} + v_p(kT_s) \tag{4.2}$$

The familiar IKF measurement process for the North channel is shown in Equation 4.2, again which is

identical to the other two channels. The correction-update step of the IKFs follow the methods described in Section 3.4.2 and are not repeated here. The bias error states are calculated in the navigation frame, but they are only recorded and useful in the platform frame. Therefore, during each correction step, the last estimate of the platform frame biases must be transformed into navigation frame biases. Once the correction step has occurred the new estimates for errors in the navigation frame biases needs to be converted back into the platform frame for use on the next inertial integration step, shown in Equation 4.3. The rotation matrices in Equation 4.3 are also time-varying but their time indices are not included below for clarity in the subscript notation. Note that for the bias vector estimates, $\hat{b}^n$ and $\hat{b}^p$, the p and n superscripts denote the platform and navigation frame values. When the vector is decomposed, the superscripts are dropped in favor of the subscript notation for the N,E,D/x,y,z indices.

$$\hat{b}^n_{k-1} = \begin{bmatrix} \hat{b}_{N,k-1} \\ \hat{b}_{E,k-1} \\ \hat{b}_{D,k-1} \end{bmatrix} = R^n_p \hat{b}^p_{k-1} = R^n_p \begin{bmatrix} \hat{b}_{x,k-1} \\ \hat{b}_{y,k-1} \\ \hat{b}_{z,k-1} \end{bmatrix} \Rightarrow IKF\,Correction \Rightarrow \; b^p_k = R^p_n b^n_k \tag{4.3}$$

The inertial system integrates navigation frame accelerations in order to produce velocity and position estimates. The velocity integration step is shown below in Equation 4.4. Remember, the rotation matrices are also time-varying and in fact estimates themselves, but time index k and estimate hat $\hat{}$ symbols are omitted for clarity in understanding the frames subscripts.

$$v_k = v_{k-1} + \int_0^{t_{int}} (f^n_{ip} + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix}) d\tau = \begin{bmatrix} v_{N,k-1} \\ v_{E,k-1} \\ v_{D,k-1} \end{bmatrix} + t_{int} \left\{ R^n_p (f^p_{ip} + \hat{b}^p_{k-1}) + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \right\} \tag{4.4}$$

$$p_k = p_{k-1} + \int_0^{t_{int}} \left\{ v_{k-1} + \tau (f^n_{ip} + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix}) \right\} d\tau = \begin{bmatrix} p_{N,k-1} \\ p_{E,k-1} \\ p_{D,k-1} \end{bmatrix} + t_{int} v_{k-1} + \frac{t^2_{int}}{2} \left\{ R^n_p (f^p_{ip} + \hat{b}^p_{k-1}) + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \right\} \tag{4.5}$$

The position integration step is shown in Equation 4.2. The incorporation of acceleration in the position equation could be neglected if the integration period, t_{int}, is small since the square of a small quantity will be negligible.

4.2.1 Simulation Results

In order to begin the development of the 3-DOF navigation filter, an incremental test simulation was built in MATLAB to track the dynamics of a point mass. This simulation ignores body rotations to eliminate the need for an AHRS to track them. The simulation assumes that the MEMS accelerometers are co-aligned with the platform frame so that, $R_a^p = I$. Since there are no body rotations, the rotation matrix from the body/platform to navigation frame is the identity matrix, $R_p^n = I$ as well. Since all three degrees of freedom are perpendicular to one another, responses per axis are independent of each other. Therefore three IKF, one for each axis, and an inertial integration block comprise this prototype navigation filter. The hypothetical point mass is given an initial position above the Massachusetts Avenue bridge near MIT in Cambridge, Massachusetts. The object is subjected to step changes in accelerations, shown in Figure 4.3. Similar to in the 1-D IKF, the simulated on-board MEMS accelerometers were modeled with $\sigma_{v_a}^2 = 2.5 \cdot 10^{-3} \frac{m^2}{s^3}$ and $\sigma_{\omega_b}^2 = 1.0 \cdot 10^{-6} \frac{m^2}{s^5}$.

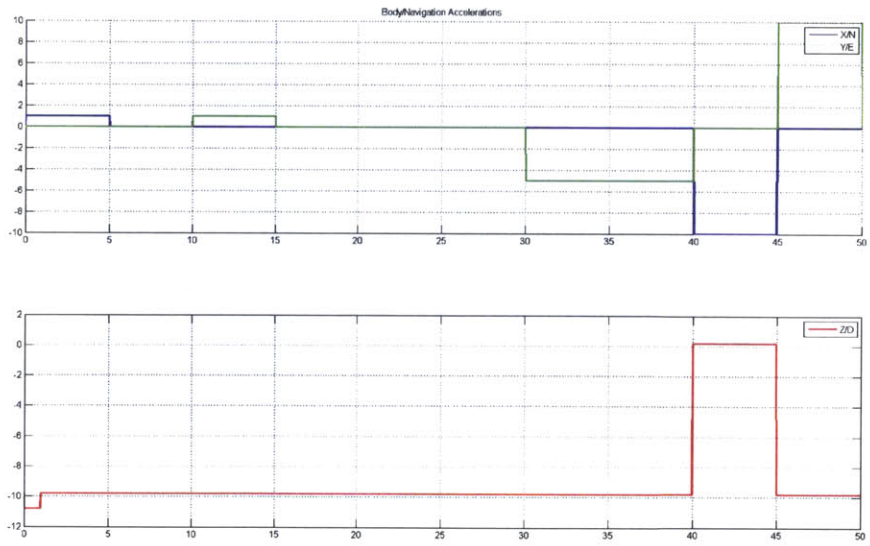


Figure 4.3: 3-DOF Simulated Acceleration Inputs (Non-Inertial)

A simulated GPS was used as a source for aiding measurements with additive white Gaussian noises of variance $\sigma_p^2 = 3m$ in position and $\sigma_v^2 = .2\frac{m}{s}$ in velocity. The accelerations are applied to the the test mass in the co-aligned body-navigation frame. The NED navigation frame was used as described in Section

3.1. In the NED frame, gravity manifests as a constant $-9.8\frac{m}{s^2}$ offset in the z/Down-axis acceleration. The contribution from gravity is removed by adding the Earth's gravitational constant to the z/Down-axis in order to obtain the inertial accelerations. For this simulation, adding $+g$ to the z/Down-axis yields a small negative acceleration during the first second of the simulation. Given the frame conventions, a negative acceleration in the down axis will result in increasing altitude. Similarly the particle will accelerate along the x/North-axis and y/East-axis during the first 15 seconds, resulting in a northeast climbing flight path. Below is the result of the simulation depicted in Google Earth.

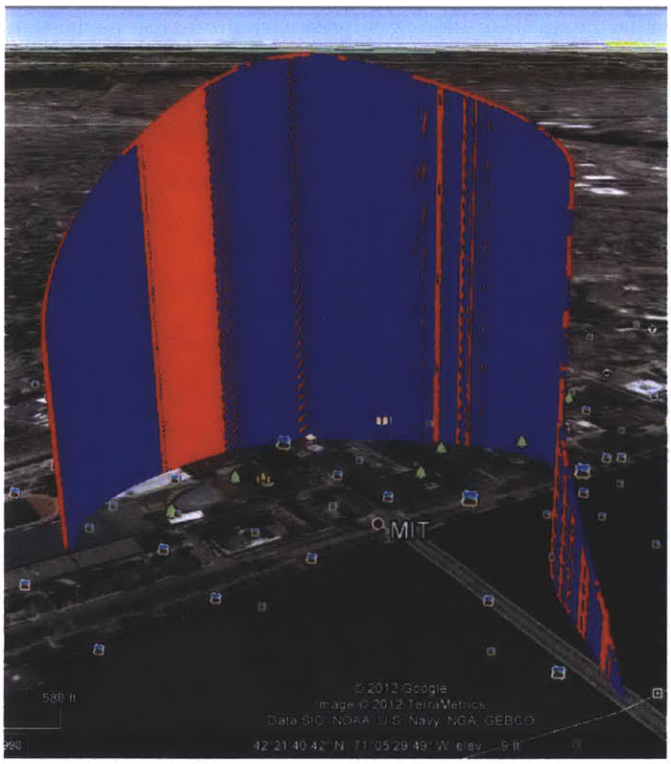


Figure 4.4: Google Earth Plot of 3-DOF Simulation

The simulation data in Figure 4.4 shows the flight path of the simulation subject to the constant accelerations shown in Figure 4.3. The blue extruded curve is the true flight path of the particle subject to perfect accelerations. The red curve is the path produced by 3-IKFs being fed noisy accelerometer and GPS data. As expected the NF does a good job of tracking the particles position and velocity in time subject to noisy position and accelerometer data. The close overlap of the two curves indicates good tracking of the

navigation filter. The average error magnitude in all 3 dimensions was $p_{err} = 0.41\text{m}$.

The performance and simplicity of this filter is greatly dependent on the assumption that the system knows, with absolute certainty, the aircraft's orientation at all times. Given correct attitude information the contribution of gravity is perfectly canceled, reducing the problem to three IKFs tracking their own dynamics in time. Imagine, however, that the initial attitude were slightly corrupted by, for example, $-2°$ in roll. Even with zero body-frame accelerations and zero initial position and velocity, gravity would couple into both the accelerometer measurements of the aircraft z/Down-axis and its y/East-axis. If the NF were unaware of this attitude when it compensated for gravity by adding $9.8\frac{\text{m}}{\text{s}^2}$ to the z/down accelerometer reading, it would compensate incorrectly, resulting in a positive downward acceleration of $f^n_{ip\,D} = 9.81(1 - \cos(-2))\frac{\text{m}}{\text{s}^2} = .0060\frac{\text{m}}{\text{s}^2}$. Furthermore, the positive y-axis coupling would cause the NF to believe that the particle were being forced to the east with an acceleration of $f^n_{ip\,E} = -\sin(-2)9.81\frac{\text{m}}{\text{s}^2} = 0.3424\frac{\text{m}}{\text{s}^2}$. This thought experiment illustrates that even a slight attitude error could be devastating for a NF that does not incorporate rotations. As such, a capable AHRS will be required in order to constantly monitor and estimate the attitude of the aircraft. This attitude sensitivity mostly impacts rotations in roll and pitch, since yaw action of the aircraft will typically only change the heading of the aircraft, and not the manner in which gravity interacts with MEMS sensors. Therefore, a yaw tracking AHRS could operate in parallel with a NF to do basic ground based tracking assuming that the roll and pitch angles are kept relatively small. The next section will build such a single-axis AHRS in conjunction with the NF filter described above.

4.2.2 Four Degree of Freedom Test Results

A Four Degree-of-Freedom (4-DoF) incremental development test involving the μUAS avionics board and several commercial IMU systems was performed in order to asses the viability of the prototype INS. The four degrees of freedom in this case were the three navigation frame axes and the yaw rotation axis. The test setup was comprised of a wheeled cart (hence only yaw rotations were possible with zero roll or pitch angles) which carried a Novatel SPAN INS connected to a Novatel LN-200 IMU, an SBG Systems IG-500N INS, and the μUAS avionics board as shown below (Fig. 4.5). The test was performed inside of a tented indoor tennis facility which provided a climate-controlled and relatively flat and level testing environment which did not interfere considerably with GPS reception. The LN200 is a tactical-grade IMU, containing fiber-optic ring gyroscopes and highly sensitive solid-state silicon accelerometers. It has been in production for several years

and continues to be used on many aircraft. The LN200 is connected to a SPAN INS module, which also receives a GPS signal, to produce position, velocity, and the three Euler angles. The SBG Systems INS is a commercially available compact INS which offers a full solution to customers in a 30mm x 30mm x 15mm footprint, shown to the left of Figure 4.6. The UAS avionics board is shown to the right of Figure 4.6. The performance results of the commercial INS solutions are shown below in Table 4.1.

Table 4.1: Commercial INS Performance

RMS Error	SPAN/LN200	SBG-500N
Horizontal Position (m)	1.2	<2
Vertical Position (m)	0.6	<2
Horizontal Velocity ($\frac{m}{s}$)	0.020	< 0.1
Vertical Velocity ($\frac{m}{s}$)	0.010	< 0.1
Roll Accuracy (degs)	0.011	1.0
Pitch Accuracy (degs)	0.011	1.0
Yaw Accuracy(degs)	0.022	1.0

The specifications listed above are for reference to qualitatively compare the performance of each system. All three systems were stacked vertically so that their z-axes were as close to co-aligned as possible and their x-y axes were parallel and co-aligned. A stackable plastic 3D-printed cage was produced to provide easy alignment of the three devices.

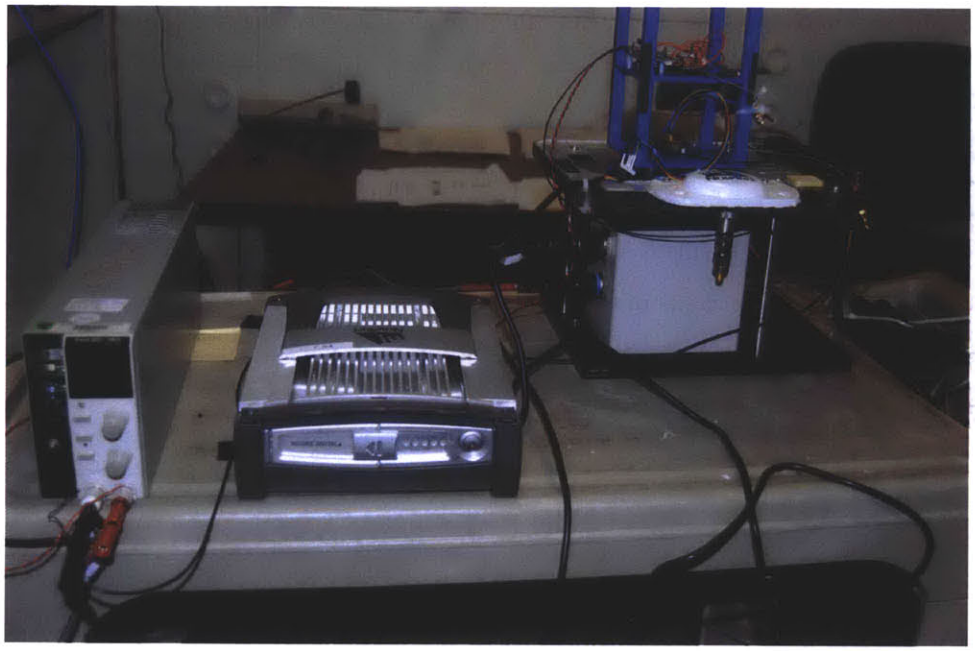


Figure 4.5: 4-DOF Test Cart

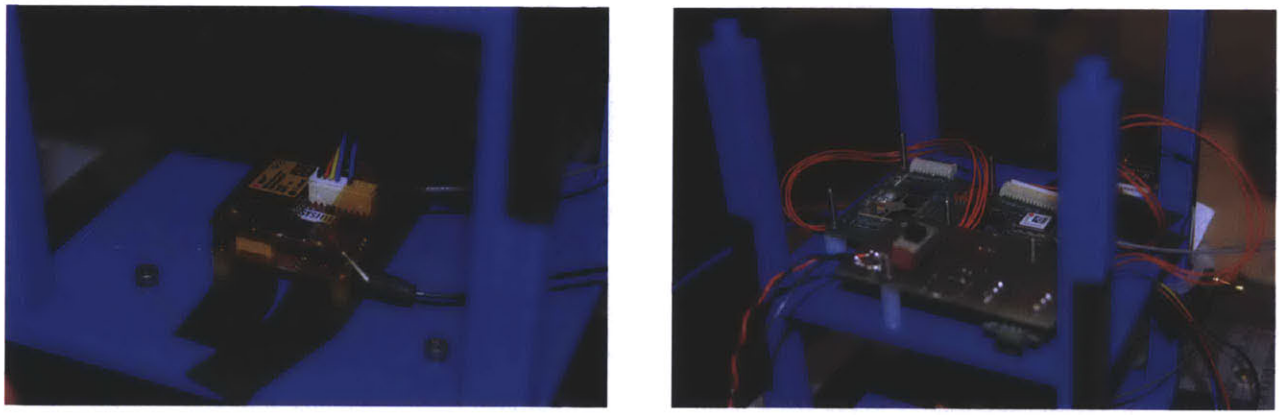


Figure 4.6: Close up of SBG Systems INS and μUAS Avionics housing

The INS shown in Figure 4.1 depicts accelerometer, magnetometer, and GPS data being fed into the AHRS in order to help produce aiding corrections. In this test, GPS velocity was the only AHRS aiding measurement available due to the preliminary nature of this test. For this purpose we assume that the heading indicated by the GPS ground velocity vector is the same as the vehicle heading as shown in Equation 4.6. This assumption, of course, does not necessarily hold in airborne platform for which aerodynamic phenomenon,

such as side slip or wind correction angle, can cause the aircraft's heading and course (direction of travel) to differ.

$$\psi \approx \psi_{GPS} = \arctan \frac{Velocity\,in\,East}{Velocity\,in\,North} \tag{4.6}$$

The path traveled by the cart inside the tennis bubble involved several overlapping square patterns along with a few circles. The results of one of the square segments which lasted for approximately 65 seconds is shown below in Figure 4.7. The test starts at the location $(0, 0)$ and progresses south-west in a clockwise manner, with right hand turns at each corner.

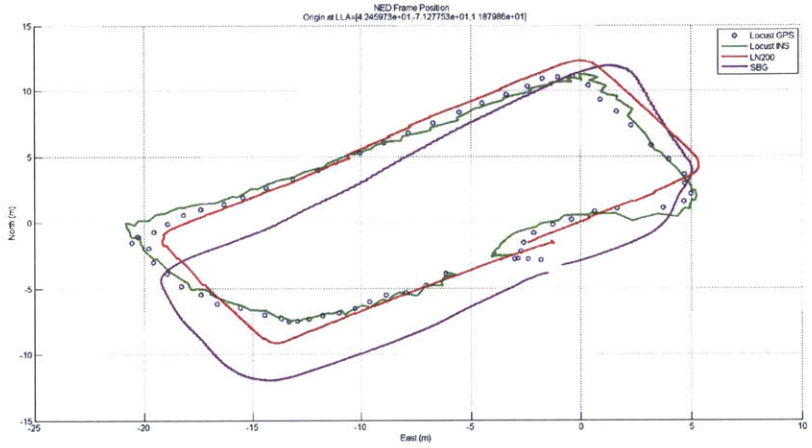


Figure 4.7: 4-DOF Test with Yaw Aiding

The red curve above is the LN200 results and is taken to be the most accurate data set to compare against. The purple curve is the SBG result which exhibits an offset in the north and east direction, which is due to non-ideal initialization of the system as well as pre-programmed antenna offsets in the commercial solutions. The blue circles are the periodic (1Hz for this test) GPS position measurements. Finally, the green path is the result of the INS running three IKFs for the navigation filter, and one IKF for yaw correction. The NF works as described in 3.4.2 and 4.2.1 where three IKFs are run in parallel to track motion. The AHRS was a one-dimensional IKF tracking the yaw rotation of the cart. The AHRS IKF is identical to the one described in Section 3.4.2 except that the state vector and measurement matrix have changed. The gyroscope is modeled as having a white Gaussian noise signal v_g with a variance of $\sigma^2_{v_g}$, and a random walk

bias signal $b(t)$ driven by white noise Gaussian noise such that $\dot{b} = \omega_{bg}$ where ω_{bg} is characterized by a variance of $\sigma^2_{\omega_{gb}}$(Eqn. 4.7). The state vector now only tracks error in yaw and error in the bias of z-axis gyroscope, as shown in Equation 4.8. The measurement matrix is altered to reflect the smaller state vector and represents the measurement of the yaw angle error, which originates from the GPS yaw measurement, shown in Equation 4.9.

$$\omega_m(t) = \omega(t) - b(t) - v_g(t) \tag{4.7}$$

$$\delta x_k = \begin{bmatrix} \delta\psi \\ \delta b_k \end{bmatrix} \tag{4.8}$$

$$C_d = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{4.9}$$

During this test the GPS yaw measurement would lag noticeably behind that actual physical rotations because of the slow 1Hz update rate and the inability of the GPS chip to predict or track sharp turns along the ground. This made clear the need for an algorithm to determine when it was appropriate to perform attitude error corrections. In other words, the measurements available to either the AHRS may not always be valuable or accurate. Incorporating erroneous information could cause the INS to diverge and report incorrect results. Therefore, in an effort to improve INS performance the GPS yaw measurement was only used during periods of time which satisfied the following requirements:

1. The GPS self reported accuracy is less than a factory defined constant.
2. The absolute value of the z-axis gyroscope was less than $15\frac{\text{degs}}{\text{s}}$
3. The INS estimated ground speed was greater than $1\frac{\text{m}}{\text{s}}$

It was found that under these conditions the 1-D AHRS performed well and tracked the yaw of the cart throughout the experiment. Figure 4.8 is a graph of the INS estimated ground speed, the z-axis gyroscope, and the Boolean flag which represents when the above conditions were met.

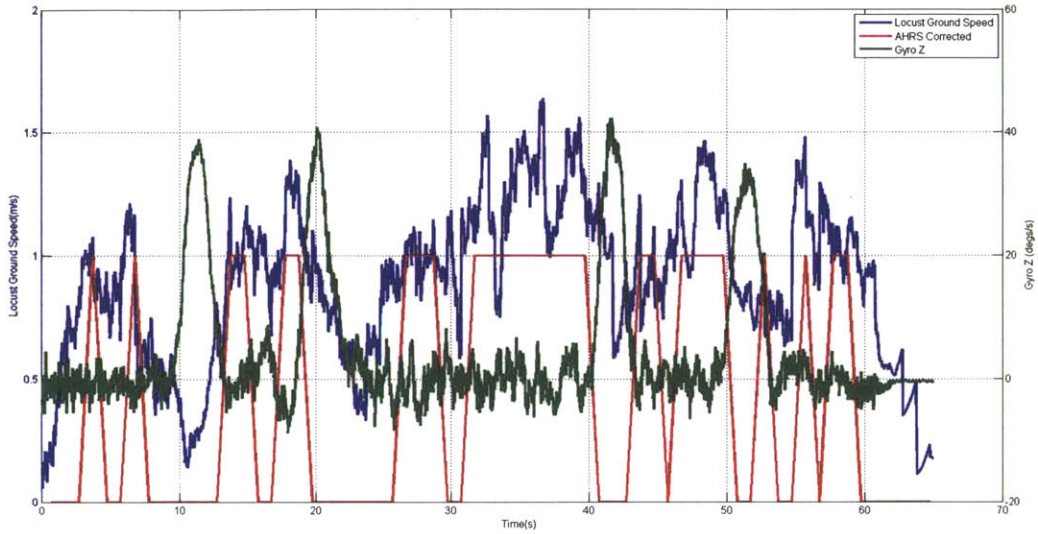


Figure 4.8: 4-DOF Boolean Flag

Figure 4.9 depicts the results of the AHRS throughout the experiment. Blue circles represent the the GPS calculated yaw measurement. Notice the divergence of measurements during turns, which occur at $t = [12, 20, 23]s$. The green curve represents the INS yaw solution and the red and purple curves represent the LN200 and SBG reported yaws, respectively. The blue line represents the Boolean flag which determines whether or not the AHRS corrections occurred during that time interval. A value of 25 represents AHRS correction while a value of 0 represents no correction, just gyroscope integration.

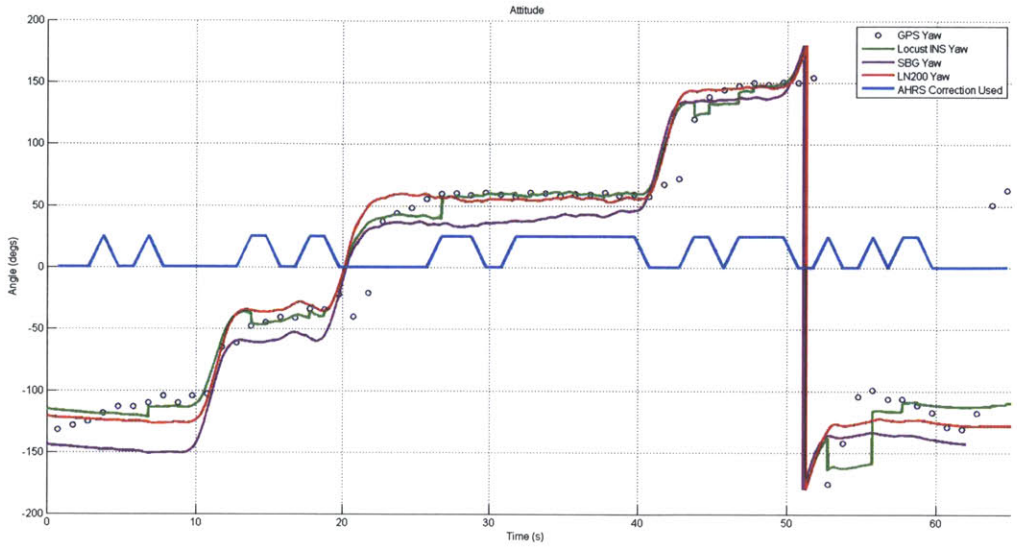


Figure 4.9: 4-DOF Yaw Results

The jump in yaw at 26s depicts the action of the AHRS, which brings the INS estimate of yaw closer to the LN200 result. Notice, however, that the correction step can cause the AHRS to diverge from the LN200 result when the GPS is trusted during instances where its data is inaccurate, such as at $t = [13, 42, 52]$. These errors can be mitigated by placing less weight on the GPS yaw measurement, in effect informing the IKF to trust the measurement less. This causes correction steps to be smaller resulting in a smoother yaw graph. The downside to this adjustment is a slower overall response to large errors in yaw. As previously mentioned, the fairly uniform offset of the SBG result from the other systems is most likely due to a poor initial calibration. The wrap around which occurs at $52s$ is an expected product of the yaw angle being restricted to the range of $\pm 180°$. The most important lesson from this experiment was the need for a decision algorithm to activate AHRS correction only when certain conditions were met. It should be noted that during each of the right angle turns, uncorrected gyroscope integration routinely undershot the true rotations by $10°$. This error is attributed to scale-factor errors in the gyroscope calibration which were not compensated for during this test. As previously discussed, the INS sensors models assume that this compensation has already been completed accurately, so no attempt was made to track or alter the gyroscope reading to correct for this effect. The INS is still capable of tracking the error produced by the gyroscope scale-factor using aiding measurements (i.e

following a valid GPS measurement) but, left unchecked, this problem could significantly degrade dynamic tracking performance on the target platform.

Table 4.2: 4-DOF Test Results

RMS Error	SBG-LN200	GPS-LN200	Locust-LN200
Horizontal Position (m)	1.8	6.7	7.5
Horizontal Velocity ($\frac{m}{s}$)	0.8	2.8	2.2
Yaw (degs)	5.4	14.7	10.4

Table 4.2 lists the approximate performance results for the 4-DOF test. The SBG INS, Locust GPS sensor, and Locust INS are all compared against the LN200 INS which was considered the truth measurement. Because the cart was traveling on a flat surface, the roll, pitch, and vertical velocity of all the INS were always close to zero and therefore omitted from the table. The only relevant measurements were horizontal position and velocity error as well as heading error. Note that the GPS and yaw offset in the SBG result has been pre-compensated for in the results so that it does not artificially increase the RMS error. The Locust NF shows considerable overshoot during turns which is likely attributed to slow response of the 1-D IKF to turn events. The result of a slow to react attitude estimate is a misdirected velocity vector in the navigation frame, producing overshoot until the system corrects itself a few seconds after the turn. Because of this overshoot, the Locust INS does worse than the GPS in horizontal position. The Locust's ability to leverage high rate sensors, however, allows it to perform slightly better in velocity than the GPS. Furthermore, it is clear that for ground based tests where the GPS reported velocity vector is small, deriving a yaw measurement from GPS produced slow and noisy results, as can be seen in Figure 4.9 and in the yaw RMS error value above. Each of the performance metrics listed above is expected to improve when using a more mature NF/AHRS and once the velocity of the IMU increases, as expected in flight.

4.3 Attitude and Heading Reference System

The AHRS is the combination an inertial integration system and a correcting filter, shown below in Figure 4.10. The complexity of handling rotations in SO(3) referenced in Section 3.2.1 will motivate the design of the correcting filter. The AHRS uses on-board gyroscopes to measure body frame angular rates ω_{ip}^{p}. However, the INS requires the navigation frame angles, also known as the Euler angles, to form a rotation matrix. In truth the INS only needs a rotation matrix to transform body frame quantities to navigation

frame quantities, so the source, whether it be three Euler angles or a quaternion, is inconsequential. The body rates transform into navigation frame rates according to Equation 3.7. This relationship describes the coupling that can occur between the three rotation axes under certain circumstances. Therefore, three independent IKFs cannot be used to track rotations. Instead, the rotation sequence as a whole needs to be tracked and corrected. This makes the process of estimating bias signals for the gyroscopes much harder to follow as all three bias signals impact the rotation output simultaneously in a coupled fashion.

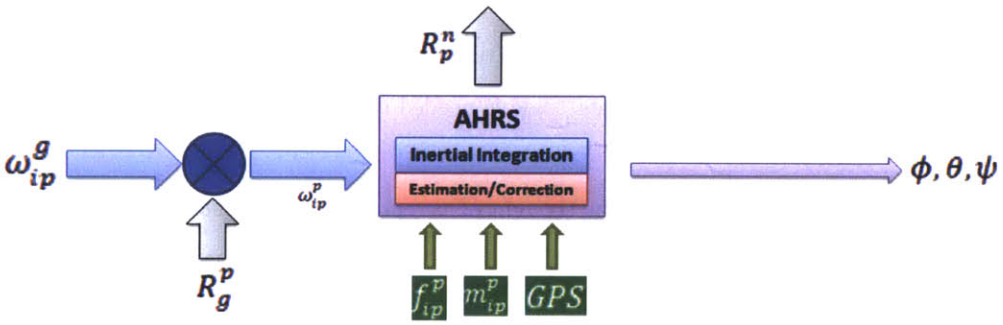


Figure 4.10: AHRS Block Diagram

Like the NF, the AHRS is split into an inertial system and a correction system. Unit quaternions are used inside the AHRS to avoid the problem of singularities associated with Euler angle representations. The inertial system integrates the attitude quaternion according to Equation 3.30 and Equation 3.31. Periodically, the attitude will be corrected in order to account for drift and bias effects. Euler angles are easier to conceptualize than quaternions, and will commonly be used by the guidance and control systems of an aircraft. The following sections will use Euler angles to form aiding measurements which eventually get converted into unit quaternions for incorporation into the correction filter. The Euler angles can be readily recovered from either a rotation matrix or a quaternion as necessary for guidance or control. A method of generating aiding measurements to correct the estimated attitude is presented in Section 4.3.1. Section 4.3.2 describes a filter which can correct the attitude estimate and provide an estimate of the gyroscope bias signals. Finally Section 4.3.3 presents AHRS test results.

4.3.1 Aiding Measurements

The most convenient way to produce measurements for all three angles is to leverage the on-board accelerometers and magnetometers. Accelerometers measure the effects of inertial accelerations as well as the reaction

force generated if the sensor is not in free-fall inside of a gravitational field. The gravity component of the measurement is useful in calculating orientation as it can constrain the roll and pitch angles of the aircraft. Because the roll and pitch calculations use the accelerometer as a gravity sensor, it can only be used when the aircraft is relatively stable and not accelerating, or $\left\| \begin{bmatrix} a_x & a_y & a_z \end{bmatrix} \right\| \approx 9.8\frac{m}{s^2}$. Given roll and pitch, heading/yaw can be calculated using the magnetometer. GPS can also be used by the AHRS to produce lower a quality measure of yaw, as shown in Section 4.2.2. The GPS sensor can produce a ground velocity vector whose arc-tangent is equivalent to a heading measurement. However, making the gross assumption that the aircraft is always pointed in the same direction of its travel can lead to divergence in flight. As was the case in Section 4.2.2 with GPS yaw measurements, logic is needed to determine when the accelerometer data is valid for use in the AHRS.

Accelerations in the platform frame can be thought of as representations of navigation frame accelerations through the R_n^p rotation matrix (Eqn. 4.10). The NED navigation frame acceleration is equal to $f_{ip}^n = \begin{bmatrix} 0 & 0 & g \end{bmatrix}'$, where $g = -9.8\frac{m}{s^2}$, only when there are no externally applied forces acting on the IMU. Platform frame accelerations measured by the MEMS accelerometers are called $f_{ip}^p = \begin{bmatrix} a_{px} & a_{py} & a_{pz} \end{bmatrix}$. Notice that $-f_{ip}^p$ is used in Equation 4.10 because gravity is taken to be negative due to the D axis pointing down in the NED frame. Following the procedure in [15], the first step in deriving the roll and pitch equations is to pre-multiply Equation 4.10 by the inverse roll and pitch rotation matrices, shown in Equation 4.11.

$$-f_{ip}^p = R_x(\phi)R_y(\theta)R_z(\psi)f_{ip}^n = R_n^p \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \tag{4.10}$$

$$-R_y(-\theta)R_x(-\phi)f_{ip}^p = R_y(-\theta)R_x(-\phi)\begin{bmatrix} -a_{px} \\ -a_{py} \\ -a_{pz} \end{bmatrix} = R_z(\psi)\begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \tag{4.11}$$

$$\begin{bmatrix} \cos\theta & \sin\theta\sin\phi & \sin\theta\cos\phi \\ 0 & \cos\phi & -\sin\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix}\begin{bmatrix} -a_{px} \\ -a_{py} \\ -a_{pz} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \tag{4.12}$$

$$\phi = atan2(-a_{py}, -a_{pz}) \tag{4.13}$$

$$\theta = atan(a_{px}, \sqrt{a_{py}^2 + a_{pz}^2}) \tag{4.14}$$

Solving for row two in Equation 4.12 yields an equation for roll (Eqn. 4.13). Solving for the first row in Equation 4.12 yields an equation for pitch (Eqn. 4.14). In practice the accelerometer values used in Equation 4.13 and Equation 4.14 are Low-Pass Filtered (LPF) versions f_{ip}^p because MEMS accelerometer data tends to be noisy. Furthermore, if the input output relationship between acceleration and accelerometer is non-unity, a unitizing calibration is needed. A calibration technique to meet this need is described in Section 2.3.1.

In order to determine yaw from the magnetometer, a Hard-Iron/Soft-Iron calibration must be performed in order to remove the effects of on-board permanent and induced magnetic fields which distort the output response. The details of the Hard-Iron and Soft-Iron calibration are offered in Section 2.3.3. The calibrated magnetometer measurement is represented by $m_{ip}^p = \begin{bmatrix} m_{px} & m_{py} & m_{pz} \end{bmatrix}$. In the northern hemisphere the Earth magnetic field points mostly north with a slight downward component which increases with latitude. This downward component is denoted by the angle of inclination, δ. Equation 4.15 describes the relationship between the magnetic field in the navigation and platform frame. Using the measured roll and pitch angles calculated from the above equations, the magnetic field measurement can be rotated to yield an intermediate tangent frame where $\phi = \theta = 0$. From there, the projected magnetic field measurements form the B_f vector which can then be used to find ψ.

$$m_{ip}^p = R_x(\phi)R_y(\theta)R_z(\psi)B \begin{bmatrix} \cos(\delta) \\ 0 \\ \sin(\delta) \end{bmatrix} = R_n^p B \begin{bmatrix} \cos(\delta) \\ 0 \\ \sin(\delta) \end{bmatrix} \tag{4.15}$$

$$B_f = \begin{bmatrix} B_{fx} \\ B_{fy} \\ B_{fz} \end{bmatrix} = R_y(-\theta)R_x(-\phi)m_{ip}^p = \begin{bmatrix} \cos\theta & \sin\theta\sin\phi & \sin\theta\cos\phi \\ 0 & \cos\phi & -\sin\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix} \begin{bmatrix} m_{px} \\ m_{py} \\ m_{pz} \end{bmatrix} = R_z(\psi)B \begin{bmatrix} \cos(\delta) \\ 0 \\ \sin(\delta) \end{bmatrix} \tag{4.16}$$

$$\begin{bmatrix} B_{fx} \\ B_{fy} \\ B_{fz} \end{bmatrix} = \begin{bmatrix} m_{px}\cos\theta + m_{py}\sin\theta\sin\phi + m_{pz}\sin\theta\cos\phi \\ m_{py}\cos\phi - m_{pz}\sin\phi \\ -m_{px}\sin\theta + m_{py}\cos\theta\sin\phi + m_{pz}\cos\theta\cos\phi \end{bmatrix} = \begin{bmatrix} B\cos\delta\cos\psi \\ -B\cos\delta\sin\psi \\ B\sin\delta \end{bmatrix} \tag{4.17}$$

$$\psi = arctan2(-B_{fy}, B_{fx}) \tag{4.18}$$

Using the first two rows of the right hand side of Equation 4.17, the calculation for the ψ angle is derived, shown in Equation 4.18. The left hand side of 4.17 defines how to generate the B_{fx} and B_{fy} coefficients given the magnetometer measurements m_{ip}^{p}, roll, and pitch.

Given favorable flight condition when $\left\| a_x \quad a_y \quad a_z \right\| \approx 9.8\frac{m}{s^2}$, and when the $\left\| \omega_x \quad \omega_y \quad \omega_z \right\| \approx 0\frac{deg}{s}$, roll and pitch can be calculated using the accelerometers. Consequently yaw can be found given the two prior angles and magnetometer information. In order to reduce attitude measurement noise, the accelerometer, gyroscope, and magnetometer data are low-pass filtered by averaging their values over the window of $\sim 1s$. During especially noisy conditions, this averaging window can be widened. Note, that the aiding measurements will lag behind the true dynamics due to the averaging, though the effect is minimal compared to the potential noise errors which are avoided. These three angles can then be converted into a quaternion following Equations 4.19-4.22, repeated here for convenience.

$$q_0 = \pm(\cos\frac{\phi}{2}\cos\frac{\theta}{2}\cos\frac{\psi}{2} + \sin\frac{\phi}{2}\sin\frac{\theta}{2}\sin\frac{\psi}{2}) \tag{4.19}$$

$$q_1 = \pm(\sin\frac{\phi}{2}\cos\frac{\theta}{2}\cos\frac{\psi}{2} - \cos\frac{\phi}{2}\sin\frac{\theta}{2}\sin\frac{\psi}{2}) \tag{4.20}$$

$$q_2 = \pm(\cos\frac{\phi}{2}\sin\frac{\theta}{2}\cos\frac{\psi}{2} + \sin\frac{\phi}{2}\cos\frac{\theta}{2}\sin\frac{\psi}{2}) \tag{4.21}$$

$$q_3 = \pm(\cos\frac{\phi}{2}\cos\frac{\theta}{2}\sin\frac{\psi}{2} - \sin\frac{\phi}{2}\sin\frac{\theta}{2}\cos\frac{\psi}{2}) \tag{4.22}$$

There is the matter of keeping track of the sign for consistency, since either the positive of negative quaternion can be used to represent the same rotation. Code will check the new computed quaternion's sign with the previous estimates and if the result is a mismatch in sign, the measured quaternion is negated. Furthermore, in order to ensure stability in the quaternion calculation, the estimated quaternion was periodically re-normalized onto the unit sphere so that the quaternion norm was unity.

4.3.2 Quaternion Filter

IMU gyroscopes report ω_m, which is a measure of the true body frame angular rates, ω_T, corrupted by the true bias signal b_T and a white Gaussian noise v_g, shown in Equation 4.23. The noise in each axis is characterized by a variance of $\sigma^2_{v_g}$. The noise signals, however, do not have an impact on the derivation of the quaternion filter since the integration of the gyroscopes results in a time averaging of the signals. The time average of the Gaussian white noise component in the gyroscope measurement is zero. The INS will need to produce an estimate of this bias signal, b_{est}, in order to prevent potential drift errors in attitude. Given an estimate of the bias, the estimated angular rates can be represented as shown in Equation 4.24. Rearranging Equation 4.23 and adding a positive and negative b_{est} yields Equation 4.25.

$$\omega_m = \omega_T - b_T - v_g = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} - \begin{bmatrix} b_{Tx} \\ b_{Ty} \\ b_{Tz} \end{bmatrix} + \begin{bmatrix} v_{gx} \\ v_{gy} \\ v_{gz} \end{bmatrix} \tag{4.23}$$

$$\omega_{est} = \omega_m + b_{est} \tag{4.24}$$

$$\omega_T = \omega_m + b_T + v_g \pm b_{est} = (\omega_m + b_{est}) + (b_T - b_{est}) + v_g = \omega_{est} + b_{res} + v_g \tag{4.25}$$

Equation 4.25 informs us that the true body frame angular rates are the linear combination of our best estimate of rotation ω_{est} given our bias estimate, the error or residue of our bias estimate, b_{res}, and some Gaussian white noise. The gyroscope data is used to propagate the quaternion, which represents the aircraft's attitude state, forward in time according to Equation 3.31. Numbered subscripts represent consecutive instants in time of length t_{int}, over which angular rates are assumed to be constant. Ignoring the measurement noise v_g and given an accurate initial quaternion of q_{0T} and the true rotation rates ω_T, the quaternion is propagated as follows:

$$q_{1T} = q_{0T} + \frac{1}{2} t_{int} (q_{0T} \otimes \begin{bmatrix} 0 \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}) = q_{0T} + \frac{1}{2} t_{int} (q_{0T} \otimes \begin{bmatrix} 0 \\ \omega_T \end{bmatrix}) \tag{4.26}$$

$$q_{1m} \approx q_{1T} \tag{4.27}$$

The second term in Equation 4.26 represents the quaternion derivative, defined in Equation 3.30. It is assumed that the measured quaternion q_m, formed from the methods discussed in Section 4.3.1, is approximately equal to the true propagated quaternion since, aside from zero-mean noise corruption, these two quaternions should be in agreement (Eqn. 4.27). The quaternion represented in Equation 4.26 represents the optimal result of the INS, assuming that there were no bias signals which disturbed the measurement process. The INS, however, has to deal with these problems, and instead be integrates its best estimate of the angular rates, ω_{est}. The INS quaternion propagation equation is shown in Equation 4.28. Note that both the theoretically ideal propagation and the real world INS propagation start from the same initial attitude, q_{0T}.

$$q_{1est} = q_{0T} + \frac{1}{2} t_{int} (q_{0T} \otimes \begin{bmatrix} 0 \\ \omega_{est} \end{bmatrix}) = q_{0T} + \frac{1}{2} t_{int} (q_{0T} \otimes \begin{bmatrix} 0 \\ \omega_T - b_{res} \end{bmatrix}) \tag{4.28}$$

$$q_{1est} = q_{0T} + \frac{1}{2} t_{int} (q_{0T} \otimes \begin{bmatrix} 0 \\ \omega_T \end{bmatrix}) - \frac{1}{2} t_{int} (q_{0T} \otimes \begin{bmatrix} 0 \\ b_{res} \end{bmatrix}) \tag{4.29}$$

$$q_{1est} = q_{1T} - \frac{1}{2} t_{int} (q_{0T} \otimes \begin{bmatrix} 0 \\ b_{res} \end{bmatrix}) \tag{4.30}$$

Equation 4.30 highlights an important aspect of the AHRS, which is that the estimate quaternion is the combination of the true result and an error quaternion formed by the bias signal. Following the procedure described by Cutler, we form the residue quaternion, B_r, shown in Equation 4.31 [6, p60]. In the following equations the $\frac{1}{2} t_{int}$ terms will be replaced with the symbol γ. Recall the quaternion product of a quaternion with its conjugate is $q^* \otimes q = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}^T$.

$$B_r = q^*_{1est} \otimes q_{1m} \approx q^*_{1est} \otimes q_{1T} \tag{4.31}$$

$$B_r \approx \left\{ q_{1T} - \gamma(q_{0T} \otimes \begin{bmatrix} 0 \\ b_{res} \end{bmatrix}) \right\}^* \otimes q_{1T} = \left\{ q_{1T}^* - \gamma(\begin{bmatrix} 0 \\ b_{res} \end{bmatrix}^* \otimes q_{0T}^*) \right\} \otimes q_{1T} \tag{4.32}$$

$$B_r \approx \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} - \gamma \begin{bmatrix} 0 \\ b_{res} \end{bmatrix}^* \otimes q_{0T}^* \otimes q_{1T} \tag{4.33}$$

The simplified form of B_r appears in Equation 4.33. The goal in this AHRS process is to be able to estimate what b_{res} is in order to correct the gyroscopes appropriately. Equation 4.33, however, requires knowledge about the true propagated quaternion in order to solve for b_{res}. Here, we make the small angle assumption in order to claim that over the small windows of integration, determined by t_{int}, the change in attitude will be small, and therefore the quaternions between two consecutive time steps will be similar. In other words we assume that $q_{1T} \approx q_{0T}$. Therefore, the property of quaternion self-conjugation still applies and results in $q_{0T}^* \otimes q_{1T} \approx \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}$. Therefore the bias resides can be directly calculated from the quaternion product of the estimated inertial quaternion and the measured AHRS quaternion, shown in Equation 4.34, where the approximately equal signs have been dropped.

$$B_r = \begin{bmatrix} B_{r0} \\ B_{rx} \\ B_{ry} \\ B_{rz} \end{bmatrix} = q_{1est}^* \otimes q_{1m} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \gamma \begin{bmatrix} 0 \\ b_{res} \end{bmatrix} = \begin{bmatrix} 1 \\ \gamma b_{resx} \\ \gamma b_{resy} \\ \gamma b_{resz} \end{bmatrix} \tag{4.34}$$

$$b_{xk} = b_{xk-1} + \frac{K_b}{\gamma} B_{rx} \tag{4.35}$$

The residues found through Equation 4.34 can be added to previous bias estimates in order to correct bias states. Equation 4.35 outlines this correction step for the x-axis gyroscope. The integration constant, γ, appears in the denominator of the B_{rx} coefficient in order to cancel out the internal γ of the residue vector. The bias gain K_b was experimentally tuned, and found to work well at a magnitude less than 0.01.

The only remaining issue to discuss is the correction of the actual quaternion which is tracking the INS motion. The solution proposed here is a LPF for the INS quaternion with a constant gain. The simplicity of

this design does not detract from its performance, although more optimal solutions do exist. Cutler uses a similar architecture to correct for drift in the flight of quad-rotors [6]. Stimac describes an extended Kalman filter to perform small Euler angle corrections [34]. Roumeliotis et al. implements an IKF structure which tracks the small error quaternions in order to estimate the attitude of an object [31]. The AHRS may migrate to the structure defined by Roumeliotis if it is found that the LPF does not suffice for stability. The LPF is in practice identical to a Kalman filter which has reached steady state operation, that is the Kalman gains, which are analogous to the LPF pole locations, are fixed. The LPF used is a simple discrete LPF defined by a single gain, shown in Equation 4.36

$$q_{est} = q_{est} - K_q(q_{est} - q_m) = (1 - K_q)q_{est} + K_q q_m \tag{4.36}$$

The algorithm low-pass filters changes in the estimated quaternion, q_{est}, that result from the incorporation of data from the measured quaternion q_m. This technique works well for small angle corrections in the attitude. The LPF gain $K_q \approx 0.07$, is kept intentionally small in order to mitigate noise from the AHRS aiding measurements. Section 4.3.3 presents test results of the AHRS measurement and correction algorithms described above.

An important note about implementing a quaternion based attitude correction system is the possibility of quaternion flipping. Specifically, the quaternion which represents a single rotation R_1^2, can be either q or $-q$. The negation of the entire quaternion still contains the same rotation information, but when operating on the quaternion as above, the sign can matter. Therefore, the AHRS software checks the sign of the measured and estimated quaternion and negates appropriately, as shown in Equation 4.37. Note the dot product operation between the two quaternions instead of the quaternion product. The dot product operates on quaternions just like vectors so that each component in one quaternion multiplies its respective component in the other quaternion.

$$if\,(q_{est} \bullet q_m < 0)\ then\ q_m = -q_m \tag{4.37}$$

4.3.3 AHRS Test Results

The AHRS measurements of Section 4.3.1 produce values for roll, pitch, and yaw measurements. A measurement quaternion q_m is formed from these measurements according to Equations 4.19-4.22. The measurement

quaternion is used by Equations 4.36 and 4.34 to correct the quaternion estimate. A series of benchtop tests were performed in order to assess the performance of the measurement and correction process. Because a rate table was unavailable, the only calibration technique used was a six-point accelerometer calibration as discussed in Section 2.3.1. The resulting inertial gyroscope integrations yield attitude angles that are close but typically over/under-shoot their true values. The AHRS is able to correct the result by incorporating aiding information when appropriate. The inertial system operates at 120Hz. The rotation sequence is given below, followed by the AHRS attitude results in Figure 4.11.

1. Initialize the Locust board with zero roll and pitch and a yaw of $-110°$. This is called the test's start position.
2. Rotate $+40°$ about the platform x-axis/roll axis.
3. Rotate $-40°$ about the platform x-axis/roll axis, returning to the start position.
4. Rotate $+45°$ about the platform y-axis/pitch axis.
5. Rotate $-45°$ about the platform y-axis/pitch axis, returning to the start position.
6. Rotate $+90°$ about the platform z-axis/yaw axis.
7. Rotate $-90°$ about the platform z-axis/yaw axis, returning to the start position.
8. Combine $+45°$ about the x, and then the y axis, followed by $+90°$ about the z-axis
9. Return to the start position.

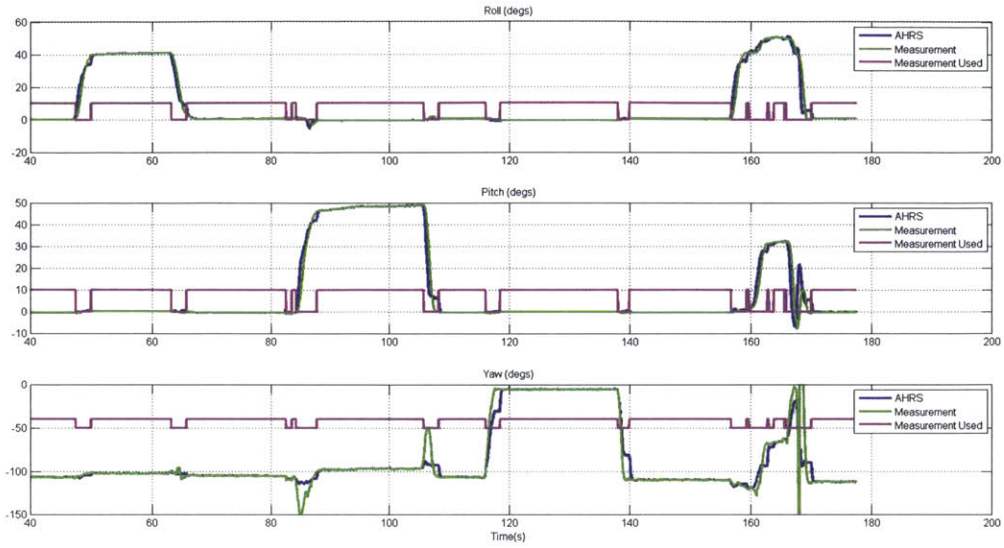


Figure 4.11: AHRS Test Rotation Results

The blue curves represent the combined state estimate of the three Euler angles. The green curves represent the aiding measurements which are being continuously generated from the accelerometer and magnetometer data. The purple curve represents a Boolean flag which indicates when the aiding measurements are used to correct the AHRS result. The flag is true when both the magnitude of the low-pass filtered gyroscope readings are less than $4\frac{\text{degs}}{\text{s}}$, and when the magnitude of the low-pass filtered inertial accelerations are less than $0.2\frac{\text{m}}{\text{s}^2}$. The gyroscope constraint prevents the AHRS from attenuating the change in angle from the integrated gyroscopes by incorporating LPF aiding information which is slow to respond to changes. The accelerometer constraint prevents the AHRS from using the accelerometers to produce roll and pitch estimates during times of inertial acceleration, which would corrupt the gravity estimate. The uncompensated scale-factor errors of the gyroscopes manifest themselves as integration errors during rotations. For example, the inertial integration system only captures $\sim 34°$ out of the total $40°$which the INS experienced, which can be seen at $t = 50s$. Similar integration errors can be seen during other rotations throughout the test. However, the AHRS is able to correct these errors once the system detects that the IMU has stopped rotating and has stabilized, which is represented by the purple curve rising to its high state.

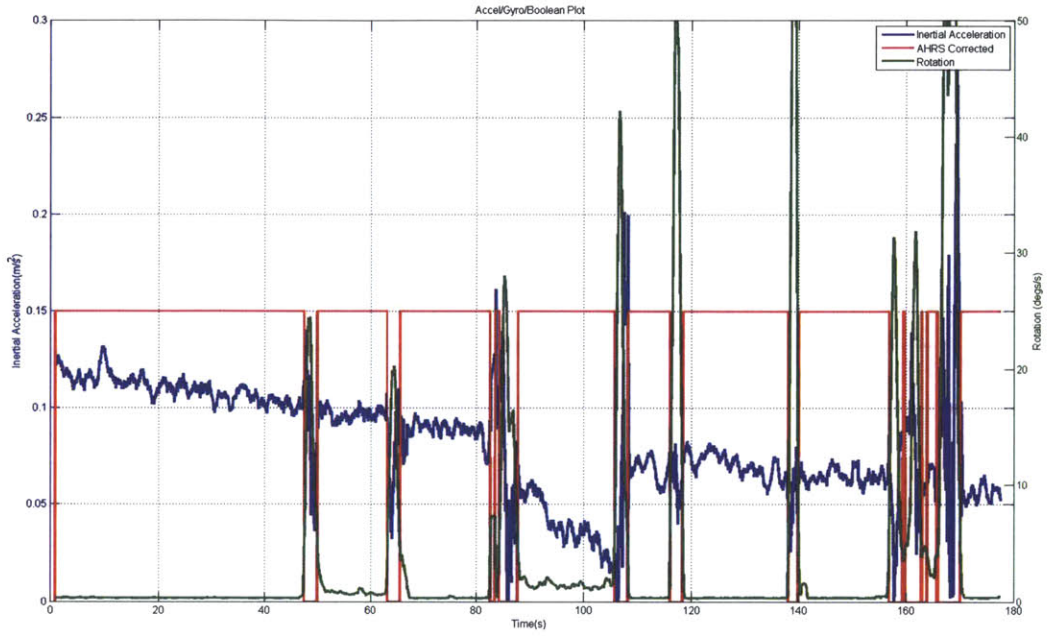


Figure 4.12: AHRS Boolean Flag

Figure 4.12 depicts the evolution of the Boolean flag in response to the rotations described above. Rotations are indicated by large spikes in the LPF rotation magnitude, in green. The blue curve depicts the inertial accelerations experienced by the INS. Inertial accelerations spike every time the Locust board was moved during the bench-top tests. Figure 4.13 depicts the bias estimation process of the AHRS. The results show good tracking of the bias signal which matches the initial bias estimate. Because the AHRS assumes that the gyroscopes are compensated for any scale-factor errors, the bias estimates experience discrete changes during rotations in order to compensate for mismatches in the angle estimate and the measurement.

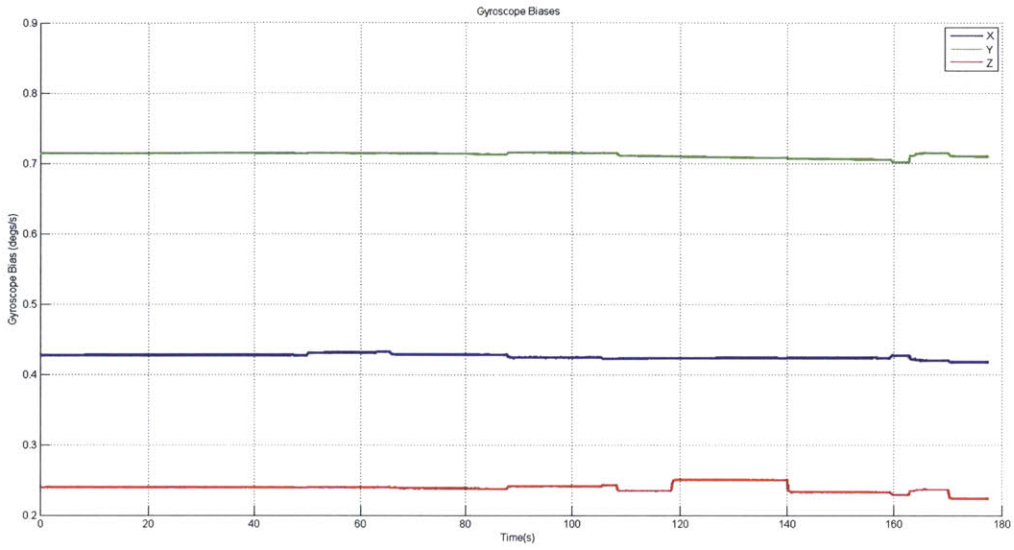


Figure 4.13: AHRS Bias Estimation

The above test exhibits good tracking performance of the AHRS under rotations in all three axes. Bias estimates converge on nominal values and measurement logic is able to dynamically asses the validity of measurements. The dynamic performance motivates the need for an accurate scale-factor calibration for all three gyroscopes. However, even in the presence of such corrupting errors, the AHRS is able to faithfully track the orientation of the avionics board. The measurement LPF window size should be sized appropriately for application of the INS. For example, a slowly rotating system would benefit from a wide window in order to effectively remove noise errors but a faster system would suffer with a wide window because of the lag in the LPF data. Ultimately, If the window is too large, the measurement data would be too outdated to be useful in correcting the attitude estimate.

Chapter 5

INS Test Aircraft Results

An in situ test involving the μUAV airframe was not possible during the time of the work presented here, as a repeatable/reliable testbed was not yet available. Furthermore, the space and weight constraints of the µUAS platform meant that it was impossible to include any sort of truth measurement system along with the embedded avionics. Therefore, to test the viability of the INS architecture it was flown inside of a Grumman Gulfstream II (GII) private aircraft, which was equipped with its own LN-200 INS, similar to the one shown in Figure 4.5. Figure 5.1 depicts the INS flight results from shortly after takeoff to landing, 75 minutes later. The Locust INS result, shown in blue, tracks the flight path well throughout the data set. The INS begins operation shortly after takeoff in order to avoid divergence problems experienced while on the ground. Currently the NF filter weighs the value of each of its measurements statically during operation. The GPS sensor is configured to track the dynamics representative of a flying vehicle, specifically speeds greater than $10\frac{\mathrm{m}}{\mathrm{s}}$. Therefore, the INS does a very poor job of estimating velocity and position of a slowly moving/still object on the surface of the Earth because the GPS measurements are not optimal. Incorporating poor measurements into an INS which relies heavily on their accuracy would result in divergence. This problem could be circumvented by dynamically reprogramming the GPS for ground-based and then airborne operation, and by integrating logic into the INS which could dynamically alter measurement noise covariances, but these methods were not explored for this thesis.

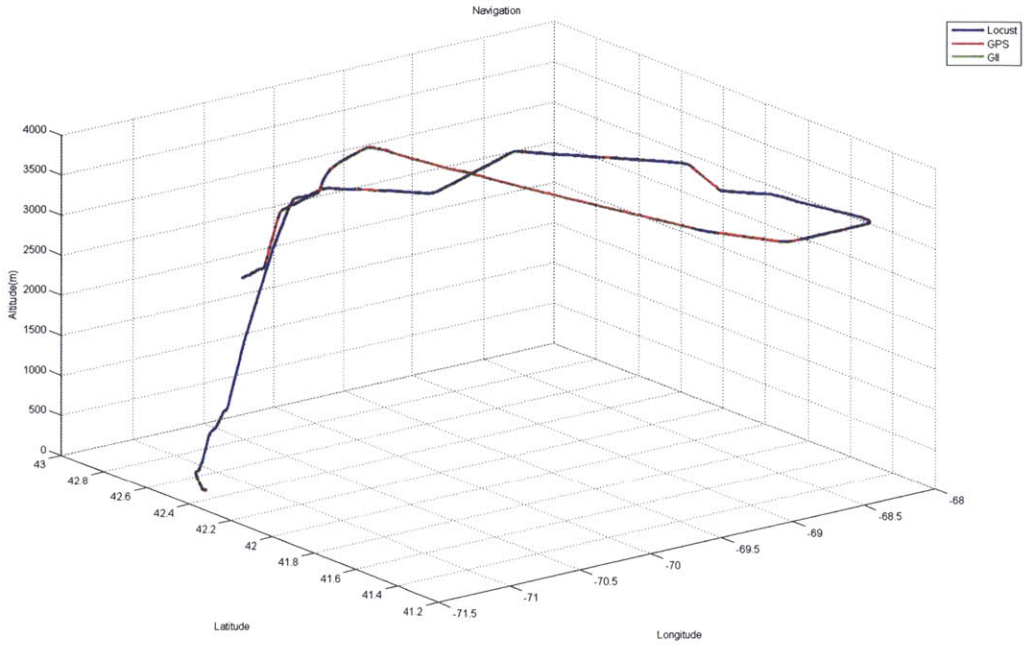


Figure 5.1: GII Test Results

The data set analyzed below, however, is a small segment of the total flight which was chosen for its balance of analytical manageability and sufficient dynamic content. For the sake of INS convergence mid-flight, the filter was initialized with the correct values for velocity in the North, East, and Down, directions as well as initial attitude information. Furthermore, because the avionics board was placed within the airframe of the GII aircraft, the magnetometer data was rendered unusable. It was infeasible to perform the required rotations to attain a proper HI/SI calibration with the INS board mounted inside the aircraft. Instead the GPS was used to produce a yaw measurement from the velocity vector, as described in Section 4.2.2. Since the Locust is designed to be deployed from an airborne platform, the INS should only use information available to it during initialization to start the INS solution, specifically initial 3D GPS position and velocity, and attitude from the AHRS sensors. No other initialization is necessary to attain an INS which converged on the correct solution.

5.1 Flat and Level Flight Segment

In order to effectively analyze the INS performance during the GII flight, a relatively flat and stable flight section was isolated. The latitude, longitude, altitude plot of this flight segment are shown below in Figure 5.2.

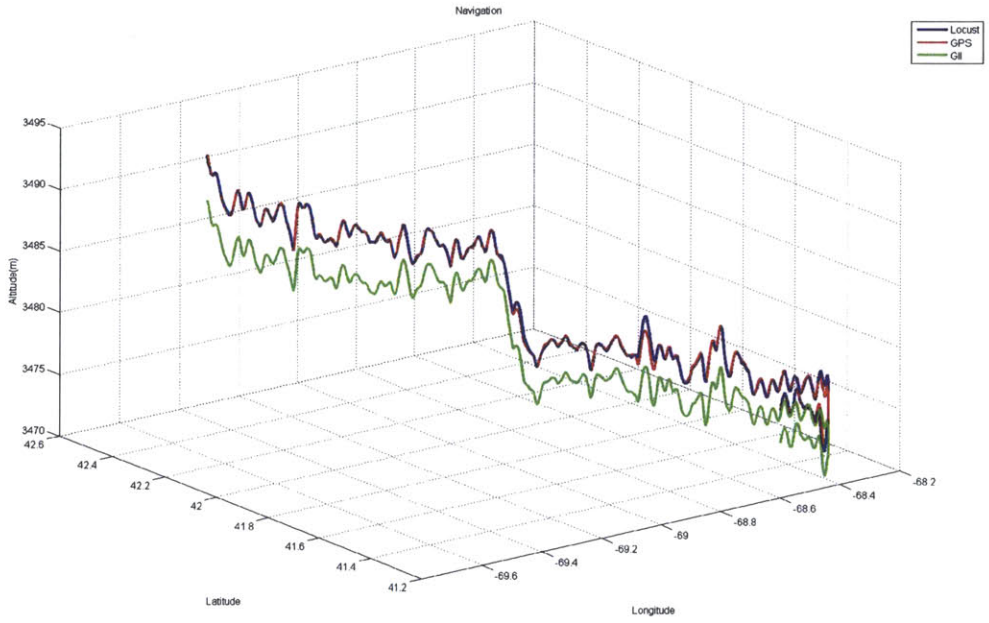


Figure 5.2: Flat and Level Flight

This data set consists of the aircraft moving in a southeast direction with a heading of 160° followed by a turn to the east and then another turn to the northeast. The plot shows good tracking of the three NF channels aside from an offset in vertical position. This offset it not a product of the INS solution, but rather the GPS sensor used by the INS. Two factors can cause an offset in GPS position when comparing two sensors. The first source of offset is the trigonometry of the antenna setup. Here, both the GII SPAN/LN200 INS and the Locust INS shared the same GPS antenna signal so this phenomenon had no effect on the position offset seen in graph above. Had the two platforms incorporated their own antenna's, there would have been an additional noticeable offset corresponding to the physical separation of the two antenna systems. The second source offset is any pre-programmed lever-arm compensation inside of the SPAN/LN200 INS. It was verified that the SPAN/LN200 INS performs its own lever-arm compensation to mathematically translate the effective GPS antenna position to the location of the LN200 IMU. This pre-programmed offset causes

the 3m shift in reported altitude seen in Figure 5.2. The velocity estimates are shown below in Figure 5.3.

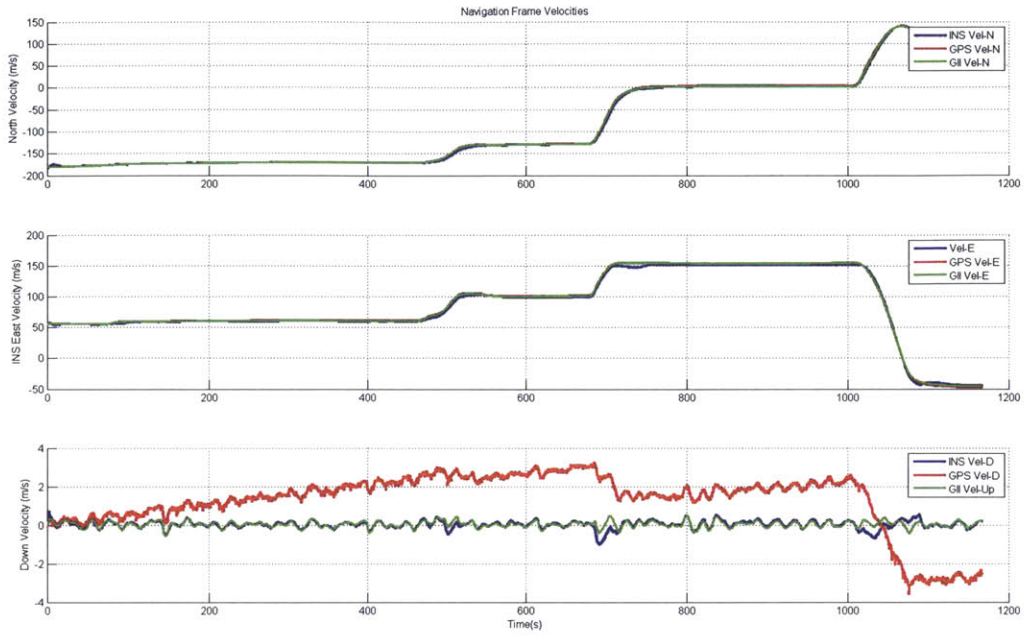


Figure 5.3: Flat and Level Flight Velocity

Note that the North and East channels display good tracking of the velocity. The Down channel displays a divergence in the GPS reported velocity with the GII result. This error is attributed to poor GPS performance for vertical velocity estimation, which is not uncommon for GPS as the vertical axis is the least informed in the implicit satellite-triangulation problem . The INS was informed of the poor GPS velocity measurements by increasing the noise covariance for that signal. The result is that the INS eastimate, shown in blue, generated its own estimate of the vertical velocity through GPS position measurements instead. The mechanisms of the IKF are inherently capable of producing position, velocity, and bias error estimates with only position aiding. Therefore, the removal of velocity information does not prevent the NF from converging on a vertical velocity estimate. The attitude performance of the INS is depicted in Figure 5.4.

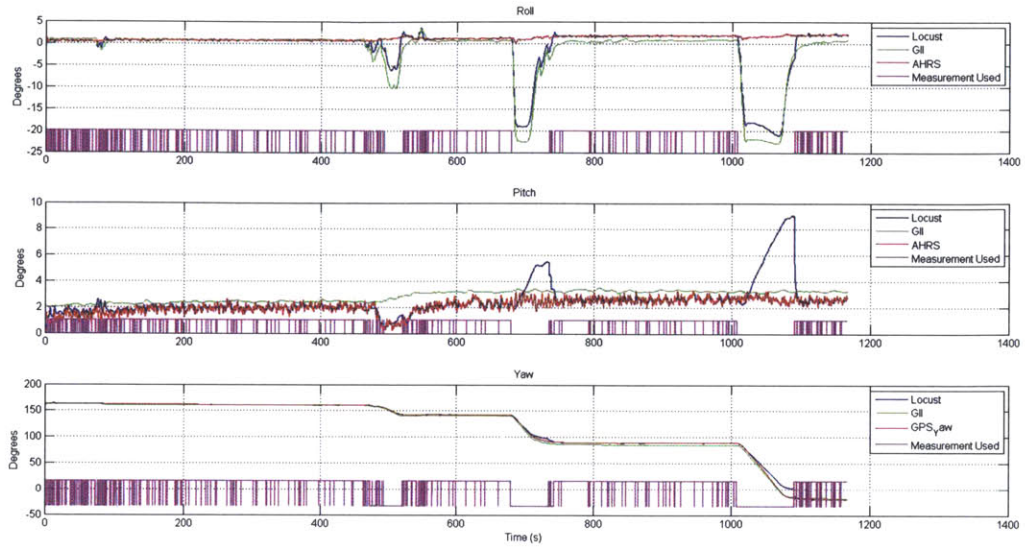


Figure 5.4: Flat and Level Flight Attitude

The purple line represents a Boolean flag similar to the one described in 4.3.3, which is true when the following conditions are met:

1. $|\omega_{ip}^{p}| < 0.8\frac{\text{degs}}{\text{s}}$: Magnitude of the instantaneous angular velocity vector must be below a certain threshold. This disables aiding during fast turns and attitude changes.

2. $LPF(|\omega_{ip}^{p}|) < 0.5\frac{\text{degs}}{\text{s}}$: Magnitude of the low-pass filtered angular velocity vector must be below a certain threshold. This removes the effects of noise on the gyro measurements when the aircraft is flying steady. Note, however, the LPF introduces a phase delay in the signal so using this metric alone would cause sluggish response to dynamics. The LPF was a two second window average.

3. $|LPF(|f_{ip}^{p}|) - 9.81\frac{\text{m}}{\text{s}^2}| < 0.4\frac{\text{m}}{\text{s}^2}$: Magnitude of the low-pass filtered inertial acceleration vector must be below a certain threshold. This disables the AHRS from correcting the attitude during periods of time when the gravity estimate would be corrupted by inertial accelerations.

Like the 4-DOF test and AHRS test, the gyroscopes exhibit scale-factor errors during turns. Notice, however, that unlike in the AHRS tests where aiding measurements were used after a rotation was performed to correct for the under-integration, AHRS aiding was not used after initial attitude changes because the aircraft was

experiencing inertial accelerations and rotations in all three axes. Furthermore, the roll and pitch channels, also exhibited slight inconsistencies with the GII data during turns. The source of this problem comes from the way in which the GII performed turns. Skilled pilots usually couple together roll, pitch, and yaw in order to perform what is called a coordinated turn. The result is that to an observer in the aircraft, the gravity vector feels as if it is pointed out the bottom of the aircraft, even though the aircraft is rolled and pitched. The coordinated turns corrupt the AHRS aiding measurements (red curves) by keeping them at their static level values, as seen at $t = \{490 - 520, 680 - 740, 1010 - 1090\}s$. By monitoring instantaneous gyroscope values, which are not phase/time lagged, the INS is able to detect rotation and instead depend only on the gyroscopes. If the gyroscopes were more accurately calibrated, the attitude error seen during these turns would decrease. A second artifact, which bares closer inspection, is the increase in pitch that manifests during turns in the μUAS INS but not in the GII INS. A first-principle evaluation of flight dynamics tells us that in steady-level flight the lift produced by an aircraft must be equal to the weight of the aircraft. If an aircraft in steady-level flight subsequently enters a banked-turn, we know that a portion of the lift will be directed horizontally into the turn (proportional to the sine of the bank angle). The vehicle must compensate for the reduced vertical component of lift either by accelerating or increasing the AOA by pitching up, or it will begin to lose altitude. The GII flight data shown in Figure 5.5, indicates neither a marked decent nor increase in airspeed, so the GII must have increased it's AOA by pitching up slightly during turns. Pilots know this as "applying back-pressure" by the application of nose-up elevator pitch.

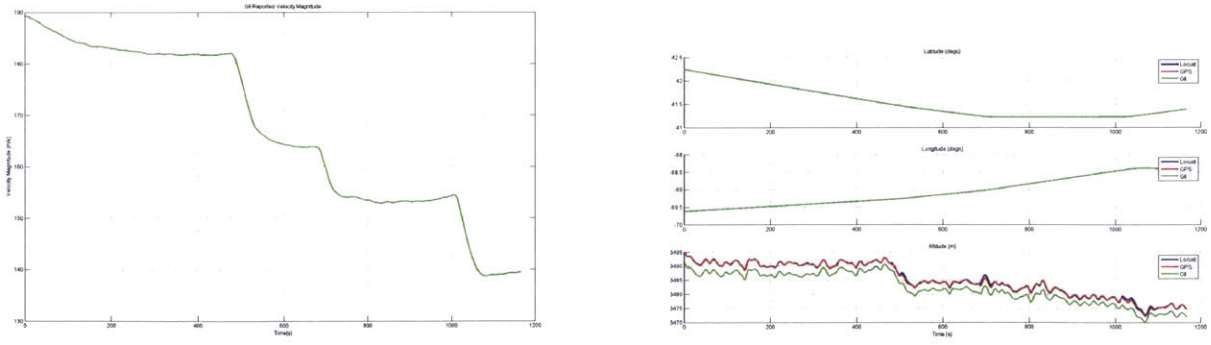


Figure 5.5: GII Reported Ground Speed(L) and Altitude(R)

From this we conclude that some increase in the apparent body-frame pitch angle is, in fact, reasonable and expected. This leaves only the question of the appropriate magnitude of pitch change for which, it is reasonable to conclude, the μUAS INS response is almost certainly too pronounced. It is believed that

the inaccuracies in the factory calibration of the gyroscopes (particularly consistent undershoot in roll and potential over-shoot in pitch) and any subsequent errors in AHRS estimation are exacerbated, in this case, when the total angular velocity is integrated disproportionately to the pitch axis. Regarding the divergence from the GII data, we believe that there is some inconsistency between the true and reported configuration of the data products provided by the GII instrumentation. For example, that the pitch field might actually express a pitch relationship in a rotated frame or some relation to the flight-path-angle (the instantaneous angle between the path of flight and a locally-level flight path). Further investigation is being conducted but initial feedback suggests that an increase in pitch is, in fact, expected by the operators despite the fact that it does not manifest very clearly in the raw data.

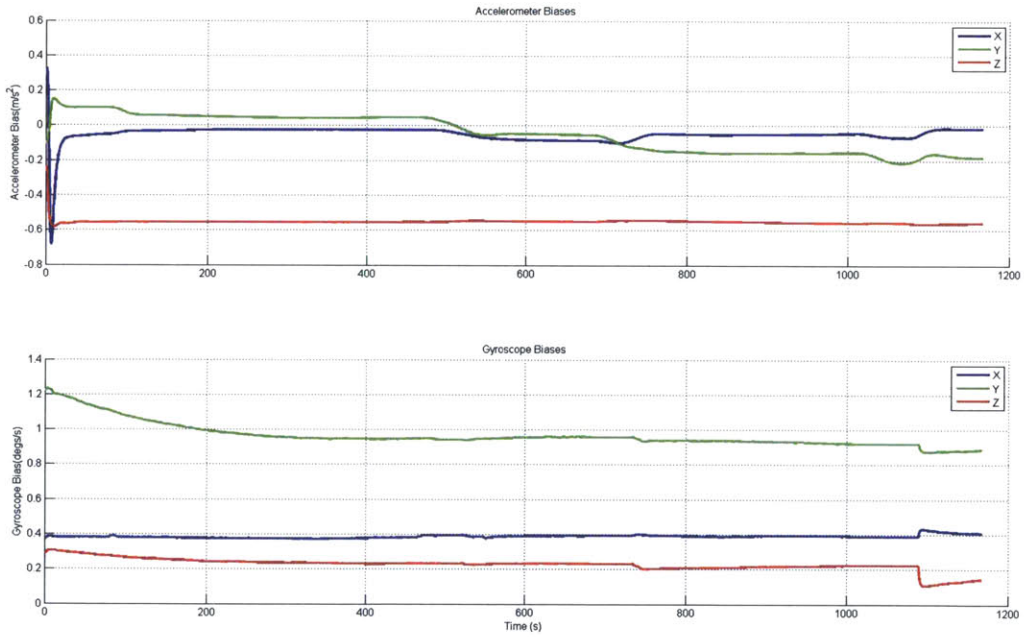


Figure 5.6: Flat and Level Flight Bias Estimation

The bias signals for the gyroscopes converge onto their nominal values with step variations during turns. This step phenomenon was explored in the AHRS section and described as the errors which accumulated due to the scale-factor errors present in the gyroscope measurements. The accelerometer bias signals experience a spike during the first few seconds of INS operation due to the NF's Kalman gains not yet having reached

steady state values. A close up of the initial moments of the accelerometer bias estimation process is shown in Figure 5.7.

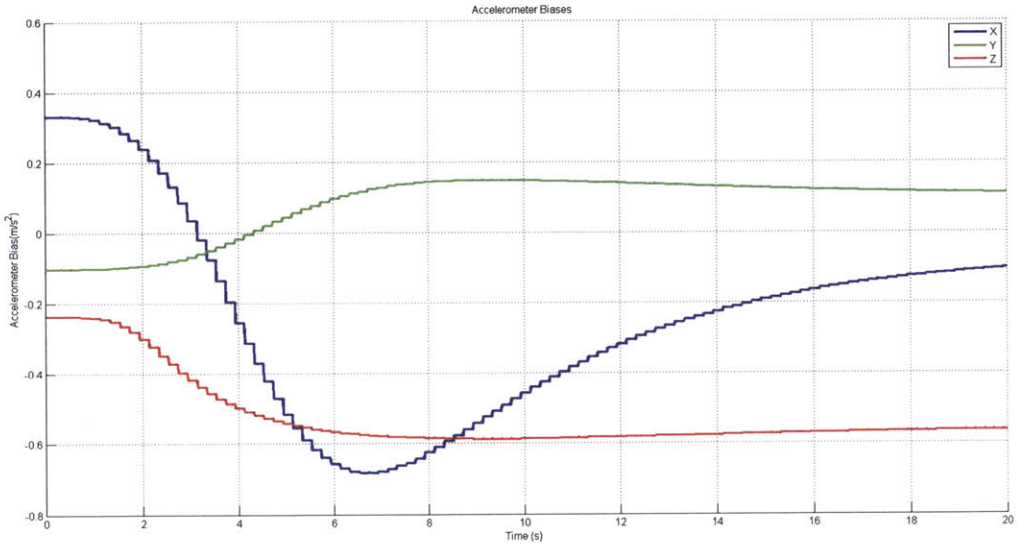


Figure 5.7: Flat and Level Flight Accelerometer Bias Initialization

All three bias signals diverge from their initial estimates to values which persist for approximately 400 seconds, at which point they change at $t = 490s$ and then again at $t = 700s$. In order to understand what is happening to the bias signals it is important to understand how the accelerometer bias is coupled into the INS. First, the bias compensated accelerometers are used by the AHRS to produce an estimate for the roll and pitch of the aircraft. Each milli-g of accelerometer bias error results in $1mrad = 0.05°$ in attitude error [12, p356]. Because the accelerometers are part of the measurement process and not part of the process being tracked in the AHRS, namely the gyroscope processes, there is no direct feedback coupling from the attitude to the accelerometer biases. Instead, there exists an indirect coupling in the NF. The attitude produced by the AHRS produces a rotation matrix R_p^n, which rotates platform frame accelerations into the navigation frame. Ideally, the navigation frame accelerations represent the true accelerations felt by the IMU, directed about the North, East, and Down directions. Integration of the North and East accelerations result in the corresponding North and East trajectories. The Down acceleration must be compensated for gravity before integrating. The entire process works well, so long as the rotation matrix R_p^n can faithfully

rotate the IMU into the navigation frame. Errors in the rotation cause the kinds of problems discussed in the end of Section 4.2.1, where the subtraction of the gravity vector along a misaligned Down axis results in residual acceleration in potentially all three axes. These residual accelerations would be detected by the NF because the integrated results would not match the GPS aiding information. The result is the NF attributing the errors to a significant change in the bias of the accelerometers as opposed to an error in rotation. This is one of the potential pitfalls of implementing an INS which separates the AHRS and NF components. A more complex INS might be able to better discern between a change in accelerometer bias and an attitude estimation error. One way to mitigate the effects of this coupling is to inform the NF that the white noise driving the bias process is much smaller than it actually is. In effect, this causes the bias signal to react much slower to correction steps of the NF.

Another problem associated with the accelerometer biases was the issue of generating initial estimates. A ground-based six-point calibration was not performed before the avionics board mounted in the GII. Therefore, initial bias estimates could only be generated by examining favorable portions of the flight. Favorable, in this case, meant periods of time without drastic changes in velocity in any axis and little/no rotation. The above INS performance assumed that the aircraft's pitch during cruise conditions was zero, which was likely incorrect by at least some small factor. On closer inspection of the GII pitch results, the aircraft cruises with relatively constant pitch of 2.25°. Similar analysis of GII roll data showed a cruising roll of 0.8°. The roll during cruise is most likely an offset error in the GII's INS. The pitch during cruise, however, was more likely a real phenomenon of the flight. As mentioned above, the initial accelerometer bias estimates were produced by averaging output data during these cruise windows. Ignoring the initial pitch caused the initial bias estimates for the INS x and z axes to be corrupted since the gravity vector was being distributed along two axes instead of one.

The auto-correlations of the NF innovation processes are shown below in Figure 5.8.

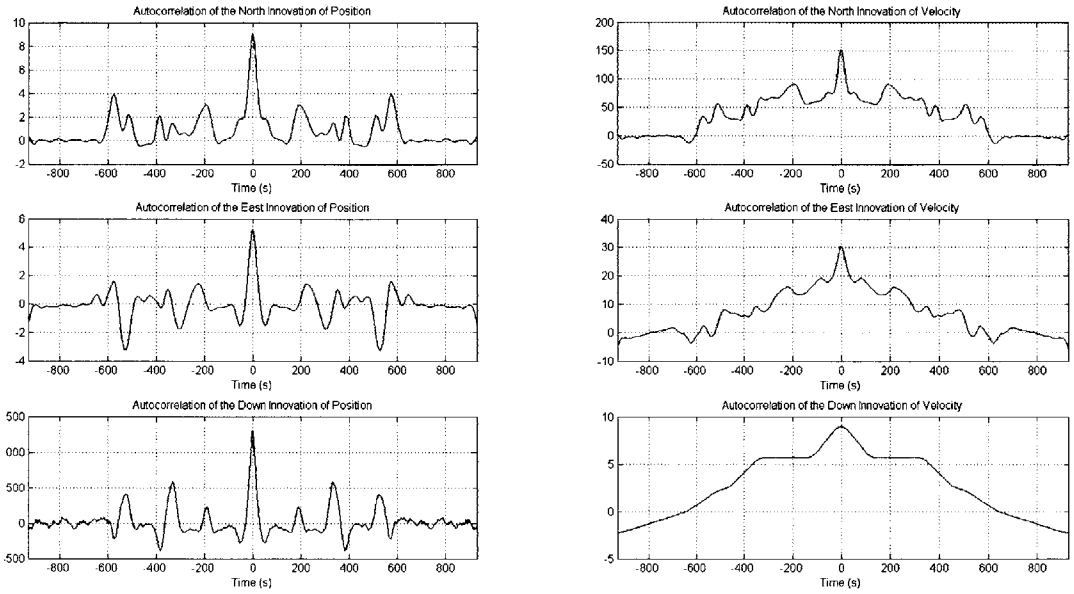


Figure 5.8: NF Innovation Auto-correlation

The whiteness of the innovations are a measure of how well the NF is extracting information from the measurement process. If the innovation is pure white noise, depicted by an impulse auto-correlation plot at t=0, then all relevant information has been incorporated into the state estimate. The NF does a fair job of extracting most of the information from each of the measurements, though because of the sub-optimality inherent in a split AHRS/NF architecture, there is some periodicity to the innovations. Furthermore, a sub-optimal calibration could result in non-whiteness in the innovations. The time evolution of the Kalman gains is shown in Figure 5.9. The important features of this chart are not the values the signals take on, but the time period during which they evolve. Notice that the time axis only extends to five seconds. This shows that the NF Kalman filters reach their steady state of operation almost immediately. Because there is no feedback into the process and measurement noise covariances, once the Kalman gains reach their steady states they will stay there for the rest of the flight. Changing the noise covariances will change how long they take to reach steady state, as well as their steady state value. This result suggests that it may be more beneficial to run the NF with constant Kalman gains, pre-calculated offline in order to avoid inverting the 3x3 matrices inside each channel's Kalman filter during operation.

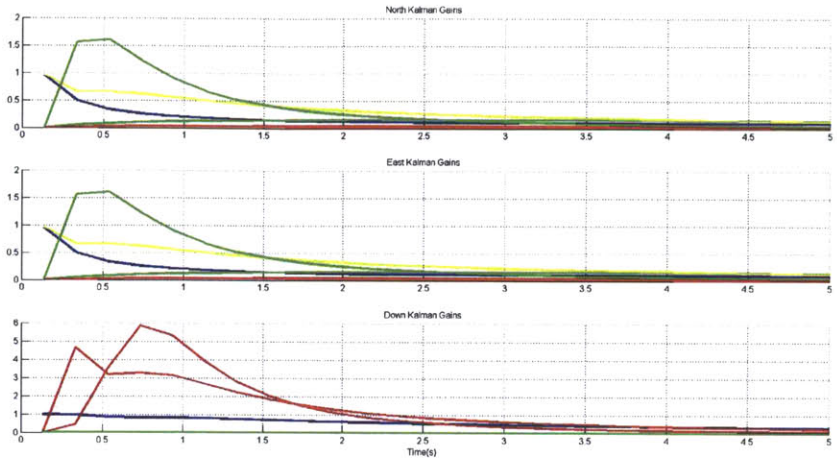


Figure 5.9: Kalman Gains

5.2 GPS Blackout

As a test of the INS ability to operate on inertial sensing alone, an artificial GPS blackout was imposed on the INS during the Flat and Level flight segment. GPS aiding was turned off, leaving the INS with only AHRS aiding and inertial integration for state propagation. Two separate GPS blackouts were imposed, each lasting 50s. The first was at t=200s at which time the IMU experienced no rotations and little inertial accelerations. The second blackout was during the second turn at roughly t=680s. The results of the first GPS blackout is shown below. The only considerable deviation was in the altitude estimate which deviated approximately 2m during the 50s blackout. The attitude estimate was not affected by the blackout because there was little rotation during the event.

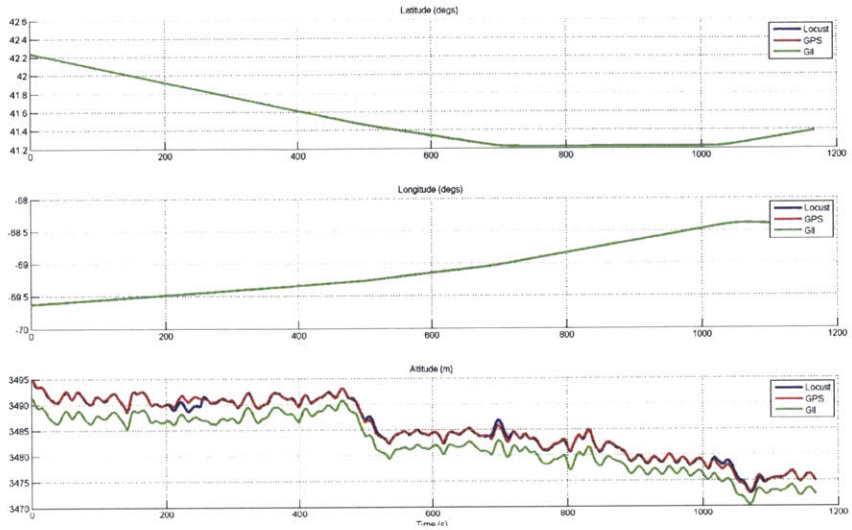


Figure 5.10: GPS Blackout t=200s LLA

The results of the GPS blackout at t=680 is shown below in Figure 5.11. The blackout resulted in an altitude error of $\sim$ 70m over the 50 second GPS blackout. The increased altitude error as compared to the t=200s blackout is because of the rotations which occurred during the GPS blackout, making it harder for the INS to determine the correct position, velocity, and altitude estimates.

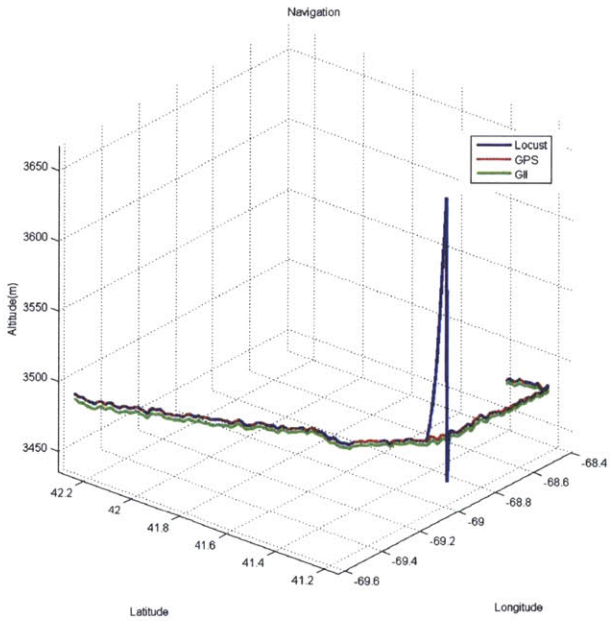


Figure 5.11: GPS Blackout t=680 3D Plot

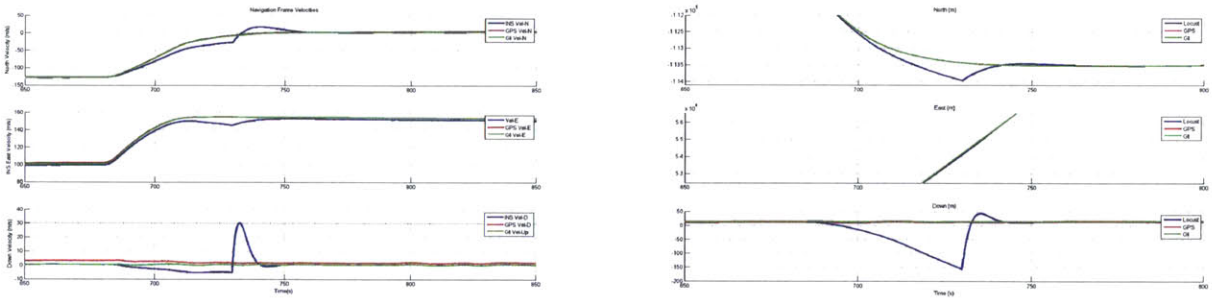


Figure 5.12: Velocity Estimate(L) and LLA(R)

The rotations during the GPS blackout produced uncertainty in the gravity estimate, which in turn caused the estimate of the rotation matrix from the platform to the navigation frame R_p^n to become inaccurate. This leads to velocity estimate errors of $18\frac{\mathrm{m}}{\mathrm{s}}$,$10\frac{\mathrm{m}}{\mathrm{s}}$, $5\frac{\mathrm{m}}{\mathrm{s}}$ in the N, E, and D channels, respectively. Therefore, the position estimates necessarily diverged as well, as shown to the right of Figure 5.12. The poor performance of the INS during GPS blackouts can be improved by performing a viable and accurate gyroscope and accelerometer calibration which would enable to INS to track inertial phenomenon much better without aiding information. Alternatively, performance can be improved by tuning the noise parameters to better

suit the more uncertain situation.

5.3 Results

Ultimately, the INS was able to attain favorable performance in navigation and attitude determination (Tab. 5.1). Performance during the flat and level flight is described under two situations, nominal (<0.5g acceleration) and dynamic/maneuvering (>1g acceleration). For each of these scenarios the following statistical metrics were calculated: root-mean-squared error, average absolute error, and standard deviation of error. Table 5.1 lists the nominal performance results of the INS. Note that the 3m vertical offset between in the Locust and GII GPS signals, inherent to the test configuration, was removed prior to computing the error statistics.

Table 5.1: Nominal Locust INS Performance Statistics

Value/Metric	RMS Error	Avg. Error	Error Std. Dev.
Horizontal Position (m)	1.73205	1.374	1.630
Vertical Position (m)	0.55	0.49	0.363
Horizontal Velocity ($\frac{m}{s}$)	0.84	0.79	0.21
Vertical Velocity ($\frac{m}{s}$)	0.05	0.04	0.04
Roll (degs)	0.67	0.63	0.19
Pitch (degs)	0.67	0.64	0.19
Yaw (degs)	2.34	2.34	0.1352

The statistics for nominal attitude errors are generally very good, though the yaw result is clearly less accurate overall. Closer inspection of the test data reveals that this discrepancy is at least partly due to underlying flight test limitation. Because a full 3D magnetometer HI/SI calibration was impractical to perform in a large aircraft, the magnetometer yaw aiding measurements were replaced with a GPS psuedo-yaw measurement. Figure 5.13 illustrates the resulting source of yaw error in the INS. The GPS yaw measurement, shown in red below, diverges from the GII reported yaw, especially during turns of the aircraft. The INS uses the GPS yaw measurement under the assumption that the aircraft's heading and ground velocity vector are co-aligned. Prevailing winds, side slip, turbulence, and other aircraft phenomena can invalidate this assumption. The GPS yaw error changes on every turn, corresponding to a different environment for the aircraft which affects the heading/ground speed vector relationship. Because the Locust INS uses the GPS yaw measurement for AHRS aiding, it too diverges from the GII reported yaw, shown in the blue curve below. The overshoots in yaw seen at $t = \{500, 700\}s$ are most likely due to the aforementioned uncalibrated

scale-factor error in the gyroscopes. The overshoot at $t = 1000s$ is a combination of the scale-factor error and the wrap around in angle which occurs at $\pm 180°$of yaw.

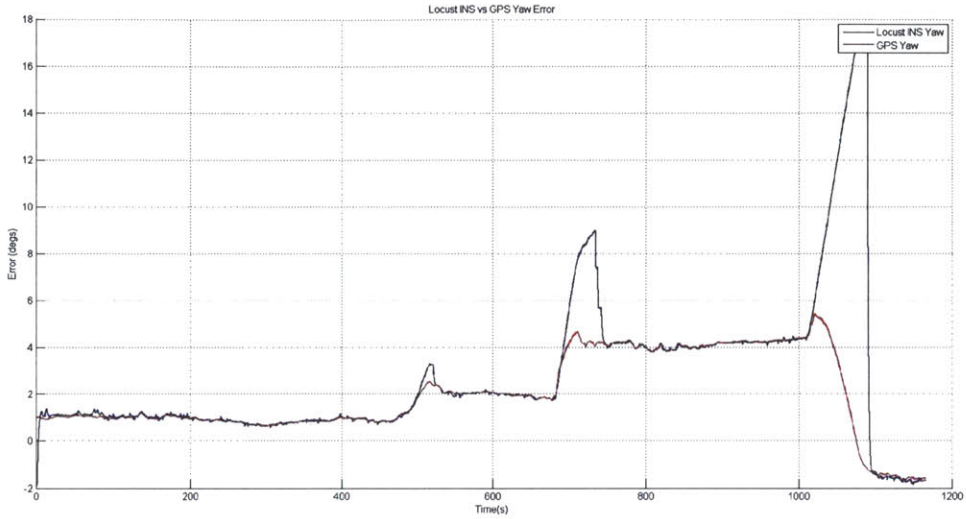


Figure 5.13: Locust INS Yaw and GPS Yaw Errors

Table 5.2 depicts the dynamic performance statistics of the INS. There is a marked increase in horizontal position and velocity errors, while the vertical errors remain relatively low. The errors in horizontal position and velocity are results of coupled errors from the AHRS. Any delay or error in the estimated attitude of the aircraft will result in inaccurate flight direction estimates, and accumulate position errors in the wrong velocity vector direction. For the nominal flight case, the NF and AHRS were able to track the slowly changing dynamics of the aircraft and maintain low error statistics. Because of the AHRS under-reporting of changes in attitude during turns, the navigation filter's results suffer. Improving the dynamic performance of the the AHRS will improve the dynamic performance of the overall INS system.

Table 5.2: Dynamic Locust INS Performance Statistics

Value/Metric	RMS Error	Avg. Error	Error Std. Dev.
Horizontal Position (m)	9.05	7.69	7.65
Vertical Position (m)	0.488	0.416	0.465
Horizontal Velocity ($\frac{m}{s}$)	2.59	2.13	1.62
Vertical Velocity ($\frac{m}{s}$)	0.29	0.22	0.22
Roll (degs)	2.56	2.28	1.09
Pitch (degs)	1.84	1.59	1.20
Yaw (degs)	6.50	6.14	2.13

The performance of the Locust INS, while favorable, can still be optimized by tuning the noise parameters of the NF and the characteristics of the AHRS quaternion low-pass filter. A table which shows the values used for the measurement and process noise covariances is shown below (Tab. 5.3). The North and East position variance is artificially high in order to improve filter performance. The variables in Table 5.3 can be altered to value one performance metric over another, depending on the application.

Table 5.3: INS Test Noise Values

Measurement Noise	Variance		Process Noise/LPF Gains	
$R_{p(N/E)}$	$(3 \cdot 10\text{m})^2$		$\sigma^2_{v_a}$	$0.02\frac{\text{m}^2}{\text{s}^3}$
$R_{p,D}$	$(3\text{m})^2$		$\sigma^2_{\omega_b}$	$0.0603\frac{\text{m}^2}{\text{s}^5}$
$R_{v,(N/E)}$	$(15\frac{\text{m}}{\text{s}})^2$		κ_b/γ	.0050
$R_{v,D}$	$(50\frac{\text{m}}{\text{s}})^2$		K_q	.007

Chapter 6

Conclusions and Future Work

This thesis presented the theoretical foundation, mathematical framework, simulated results, and airborne test results for an inertial navigation system. Kalman state estimation and quaternion attitude dynamics were leveraged to combine GPS and MEMS accelerometer, gyroscope, and magnetometer sensor data. A theoretical derivation presented the details of implementing an indirect Kalman filter, whose main benefit was the ability to replace an aircraft's complex dynamics with simpler error-state dynamics. This concept of dynamic model replacement proved invaluable in developing an INS for a platform whose design was in constant flux.

The INS developed for the Locust μUAS exhibits the basic navigation functionality needed in order to allow a FMS to autonomously stabilize and control an aircraft. The design focused on building a system which could be easily implemented and executed on a SWaP-constrained micro-controller. The decision to separate the AHRS and NF proved to be successful, although the optimality of the total solution necessarily suffered because of it. This is a tradeoff in performance which will be reexamined once the full INS and FMS are ready to be implemented on the Locust μUAS for flight tests.

The error statistics seen in Table 5.1, specifically a nominal average position error of less than 1.5m and an average attitude (roll and pitch) error of less than 1°, indicates a viable foundation for a final solution. However, these performance values are only indicative of a specific flight envelope and, therefore, are likely to not be representative of the INS performance in every situation. Table 5.2 is proof of this distinction and further proof that improvements need to be made in order to stabilize a real aircraft in flight. The most

pressing concern for the successful implementation of the INS is the dynamic performance of the AHRS during turns. Being able to accurately estimate the aircraft's attitude through changing orientations will be crucial to the success of the INS, and therefore the FMS as a whole. To this end, a thorough calibration will be performed for the gyroscopes which will solve the chronic problem of over/undershooting turns. Furthermore, because the flight segment analyzed above varied with less than 10m in altitude, it is likely that the dynamic vertical performance of the INS is artificially accurate.

Some of the above performance metrics reflect limitations of the separate AHRS/NF INS architecture. Notably, the relationship between attitude and accelerometer bias errors seen in the above analysis may motivate future development of a more complete architecture which could allow information to be shared by the NF and AHRS. The decision to implement complementary filters greatly simplified the approach of the NF, and can be further used to develop an IKF to track the attitude quaternion. A quaternion based IKF is the most likely immediate extension to the INS, along with more detailed noise models of the MEMS sensors. The complimentary filter architecture is a powerful tool for INS because it allows complex aircraft dynamics to be replaced by simpler kinematic models based on error relationships. While the development of the Locust airframe is ongoing, which therefore means that the target platform is unavailable for testing, other solutions will be available in the near future to test the INS in a more representative fashion. For example, captive carries of the Locust board on small RC piloted aircraft will produce dynamics more indicative of the Locust airframe than that of a Gulfstream private jet.

This thesis stands as a foundation for the future development of the Locust μUAS flight management system. Development of the larger flight management system will depend on the success and accuracy of the INS to achieve autonomy. The results presented above indicate the successful implementation of the necessary functionalities needed to provide a FMS with a faithful representation of the state of a micro-scale unmanned aerial vehicle.

Bibliography

[1] Dava J Neaman Amir. *Interactive Aerospace Engineering and Design: Introduction to Aircraft Stability and Control.* McGraw-Hill, 2002.

[2] Dava J. Newman Amir R. Amir. *Interactive Aerospace Engineering and Design.* McGraw-Hill, 2002.

[3] Reg Austin. *Unmanned Aircraft Systems: UAVs Design, Development and Deployment.* Wiley, 2010.

[4] Stergios I. Roumeliotis Gaurav S. Sukhatme George A. Bekey. Circumventing dynamic modeling: Evaluation of the error-state kalman filter applied to mobile robot localization. In *IEEE*, 1999.

[5] Stergios I. Roumeliotis Gaurav S. Sukhatme George A. Bekey. Smoother based 3d attitude estimation for mobile robot localization. Technical report, University of Southern California, 1999.

[6] Mark Johnson Cutler. Design and control of an autonomous variable-pitch quadrotor helicopter. Master's thesis, MIT, 2012.

[7] Analog Devices. Adxl345. Rev. D.

[8] DOD. Dod vehicle budget request and summary justification "major weapons systems", february 2011. Technical report, DOD, 2011.

[9] Ph.D Dr. J.O. Miller Capt Carl R. Pawling, USAF Maj Stephen P. Chambal. Modeling the u.s. military intelligence process.

[10] Billur Barshan Hugh F. Durrant-Whyte. Inertial navigation systems for mobile robots. In *IEEE*, 1995.

[11] M. Euston, P. Coote, R. Mahony, Jonghyuk Kim, and T. Hamel. A complementary filter for attitude estimation of a fixed-wing uav. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 340 345, 2008.

[12] Jay A. Farrell. *Aided Navigation GPS with High Rate Sensors*. McGraw Hill, 2008.

[13] Phillip Finnegan. Unmanned aerial vehicles. Technical report, Teal Group, 6-12 2012.

[14] Freescale-Semiconductor. An4248 rev.3, 1 2012.

[15] Freescale-Semiconductor. An3461 rev.6 tilt sensing using a three axis accelerometer, 3 2013.

[16] Freescale-Semiconductor. Calibrating an ecompass in the presence of hard and soft-iron interference, 4 2013.

[17] Honeywell. Hmc5883l datasheet, 02 2013.

[18] Jonathan How. 16.322 class notes mit fa2012.

[19] David A. Korka. Kalman filtering for an aided inertial navigation system. Master's thesis, Massachusetts Institute of Technology, 1999.

[20] Jack B. Kuipers. *Quatermions and Rotation Sequences*. Princeton University Press, 1999.

[21] KyuCheol Park Dohyoung Chung Hakyoung Chung Jang Gyu Lee. Dead reckoning navigation of a mobile robot using and inindi kalman filter. In *IEEE*, 1996.

[22] Sebastian O.H. Madgwick. An efficient orientation filterlter for inertial and inertial/magnetic sensor arrays. Technical report, N/A, 2010.

[23] S.O.H. Madgwick, A. J L Harrison, and R. Vaidyanathan. Estimation of imu and marg orientation using a gradient descent algorithm. In *Rehabilitation Robotics (ICORR), 2011 IEEE International Conference on*, pages 1–7, 2011.

[24] R. Mahony, T. Hamel, and Jean-Michel Pflimlin. Nonlinear complementary filters on the special orthogonal group. *Automatic Control, IEEE Transactions on*, 53(5):1203–1218, 2008.

[25] Paul Zacharn Howard Musoff. *Fundamentals of Kalman Filtering: A Practical Approach*. American Institute of Aeronautics and Astronautics, 2009.

[26] Vikas Kumar N. Integration of inertial navigation system and global positioning system using kalman filtering. Master's thesis, DEPARTMENT OF AEROSPACE ENGINEERING INDIAN INSTITUTE OF TECHNOLOGY, BOMBAY MUMBAI, 2004.

[27] NAVSTAR. Navstar gps user equipment introduction. Technical report, NAVSTAR, 1996.

[28] MIT OCW. 16.322 stochastic estimation and control 2004.

[29] U.S.A Department of Defense GPS NAVSTAR Organisation. Global positioning system standard positioning service performance standard. Technical report, U.S.A Department of Defense GPS NAVSTAR Organisation, 2008.

[30] Roger Pratt. *Flight Control Systems.* American Institute of Aeronautics and Astronautics, 2000.

[31] Nikolas Trawny Stergios I. Roumeliotis. Indirect kalman filter for 3d attitude estimation. Technical report, Department of Computer Science & Engineering University of Minnesota, 2005.

[32] C. E. Shannon. Communication in the presence of noise. *Proc. Institute of Radio Engineers*, 37(1):10–21, January 1949.

[33] ST. St l3g4200d, 12 2010.

[34] Andrew K Stimac. Precision navigation for aerospace applications. Master's thesis, Massachusetts Institute of Technology, 2004.

[35] Tony S. Tao. Design and development of a high-altitude, in-flight-deployable micro-uav. Master's thesis, Massachusetts Institute of Technology, 2012.

[36] Mark B. Tischler. System identification methods for aircraft flight control development and validation. Technical report, National Aeronautics and Space Administration, Aeroflightdynamics Directorate, U.S. Army ATCOM, Ames Research Center, Moffett Field, California, 1995.

[37] u blox. Max-6q.

[38] Gary Bishop Greg Welch. An introduction to the kalman filter, May 2012.

[39] David Titterton John Weston. *Strapdown Inertial Navigation Technology, 2nd Edition.* The IEE, 2004.

[40] Wikipedia. Image ecef enu longitude latitude relationships.

[41] Gregory R Kriehn Jonathan Flerchinger Alfred Lopez Victor Urbieta Eryn Combs Tom Pittenger Tim Courrejou Jason Cook Charles Yam. Unmanned aerial vehicle laser targeting system. Technical report, AUVSI, 2010.