ANALYZING AND MODELING LOW-COST MEMS IMUS FOR USE IN AN INERTIAL NAVIGATION SYSTEM

by

Justin Michael Barrett

A Thesis

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

in partial fulfillment of the requirements for the

Degree of Master of Science

in

Robotics Engineering

May 2014

APPROVED:

_____

Professor Michael A. Gennert, Major Advisor, Head of Program

_____

Professor William R. Michalson

_____

Dr. Julian L. Center, Jr.

Abstract

Inertial navigation is a relative navigation technique commonly used by autonomous vehicles to determine their linear velocity, position and orientation in three-dimensional space. The basic premise of inertial navigation is that measurements of acceleration and angular velocity from an inertial measurement unit (IMU) are integrated over time to produce estimates of linear velocity, position and orientation. However, this process is a particularly involved one. The raw inertial data must first be properly analyzed and modeled in order to ensure that any inertial navigation system (INS) that uses the inertial data will produce accurate results.

This thesis describes the process of analyzing and modeling raw IMU data, as well as how to use the results of that analysis to design an INS. Two separate INS units are designed using two different micro-electro-mechanical system (MEMS) IMUs. To test the effectiveness of each INS, each IMU is rigidly mounted to an unmanned ground vehicle (UGV) and the vehicle is driven through a known test course. The linear velocity, position and orientation estimates produced by each INS are then compared to the true linear velocity, position and orientation of the UGV over time. Final results from these experiments include quantifications of how well each INS was able to estimate the true linear velocity, position and orientation of the UGV in several different navigation scenarios as well as a direct comparison of the performances of the two separate INS units.

Acknowledgements

First and foremost I would like to thank my family and friends for their unwavering support over the last eight years of my academic career at WPI and the many years preceding it. You have always been confident in my abilities even when I wasn't, and without you this thesis simply would not have been possible.

Second, I would like to thank my research advisor Professor Michael A. Gennert and the other two members of my thesis committee Professor William R. Michalson and Dr. Julian L. Center, Jr. for consistently keeping me and my research on the right track. Thank you for constantly answering all of my questions and for providing me with praise when I earned it and constructive criticism when I needed it.

Third, I would like to thank Autonomous Exploration, Inc. and NASA for providing the funding for this research. Without their financial contributions I would have missed out on this valuable research experience that has taught me so much. I would also like to thank Bryan W. Welch of the NASA Glenn Research Center for being our team's NASA contact and for periodically providing us with advice on how to troubleshoot our technical problems.

Finally, I would like to thank my fellow team members James F. Kirk and Michael D. Audi for their contributions to the larger research project that this thesis is a part of. Your contributions to the project made my life significantly easier, and working with you was always an enjoyable and worthwhile experience.

Table of Contents

List of Figures

List of Tables

Section 1 – Introduction

As robotics technology continues to become more prevalent in modern society, the development and production of autonomous vehicles continues to expand to meet the increasing demand for vehicles that can safely explore treacherous, inhospitable or hard to reach locations. One of the most crucial components of any autonomous vehicle, especially one tasked with exploring and/or mapping unknown environments, is its navigation system. Without the ability to accurately determine its linear velocity, position and orientation with respect to some larger frame of reference, an autonomous vehicle would be drastically limited in terms of the tasks it could accurately perform. Therefore, the implementation of an accurate navigation system on any autonomous vehicle is an important aspect of the system design and is imperative to the overall ability of the vehicle to function properly.

One type of navigation system commonly implemented on autonomous vehicles is called an inertial navigation system (INS). Inertial navigation systems measure the accelerations and angular velocities that an autonomous vehicle experiences (typically in three dimensions) and integrate those measurements over time to produce estimates of the linear velocity, position and orientation of the vehicle. Inertial navigation is particularly useful because it can be implemented on almost any type of vehicle and in almost any type of environment. In comparison, many absolute navigation sensors such as GPS receivers, Digital Magnetic Compasses (DMCs), etc. have a much more limited range of usefulness and are typically most effective when used in relatively spacious outdoor environments.

The measurements of acceleration and angular velocity that are required for inertial navigation are produced by a sensor package called an inertial measurement unit (IMU). There are several different varieties of IMUs, but one of the more common varieties is the micro-electro-mechanical-system (MEMS) IMU. MEMS IMUs are very inexpensive to manufacture, which makes them a popular choice among robotics hobbyists. However, as is the case with any type of sensor, IMUs can never produce perfect measurements of acceleration and angular velocity. Since the measured accelerations and angular velocities returned by the IMU are imperfect, it follows that any estimates of linear velocity, position and orientation based on those measurements will be imperfect as well. The end result is that after a short period of time, the estimates of linear velocity, position and orientation produced by an INS very quickly become inaccurate and unusable.

To compensate for the imperfections in the IMU measurements, an INS typically has one or more secondary navigation sensors that provide direct measurements of the linear velocity, position and/or orientation of a vehicle. These secondary navigation sensors could be anything from stereo vision systems, to GPS receivers, to digital magnetic compasses (DMCs) or any other type of sensor that could

be used to measure linear velocity, position or orientation. The information from these secondary navigation sensors is incorporated into the INS using an Extended Kalman Filter (EKF). The EKF produces correction terms that are used to adjust the initial estimates of linear velocity, position and orientation calculated from the imperfect IMU measurements. Adding secondary navigation sensors into an INS greatly increases its ability to produce accurate estimates of the linear velocity, position and orientation of a vehicle over long periods of time. The block diagram in Figure 1.1 illustrates the basic operation of a typical INS.

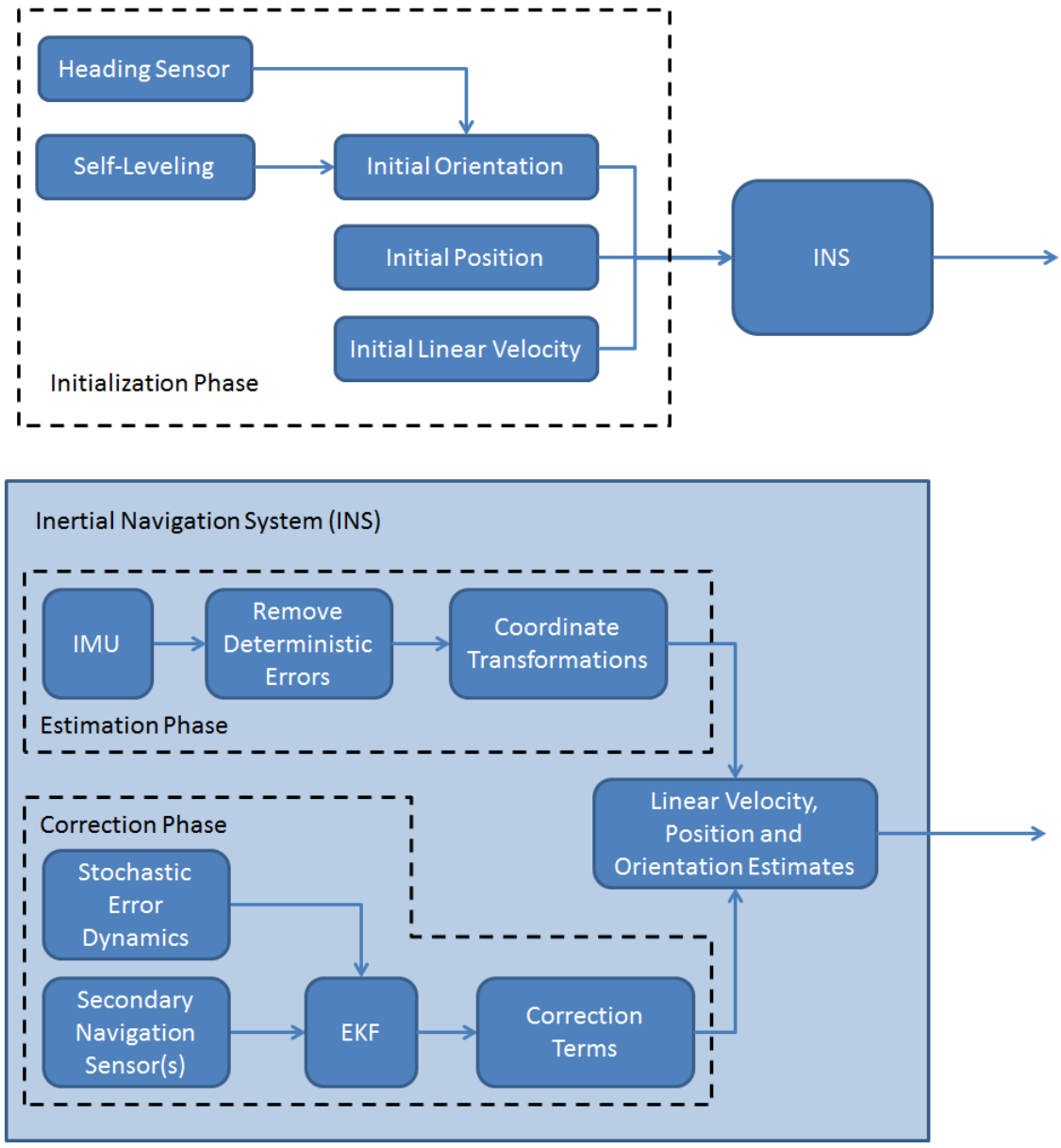

Figure 1.1: A block diagram illustrating the basic operation of a typical INS.

The goal of this thesis is to design, implement and test two separate INS units based around two low-cost MEMS IMUs. The results of the testing procedures are then used to determine how well each individual INS performs as well as to determine which IMU is better suited for implementation in an INS. This goal is achieved by following these basic steps:

- Each IMU is analyzed and modeled using a series of laboratory experiments designed to isolate and characterize the behavior of several well-documented types of errors common to MEMS IMUs.
- An inertial navigation algorithm is designed that incorporates the information and error models produced by the laboratory experiments.
- To collect test data for the inertial navigation algorithm, the two IMUs are rigidly mounted to an unmanned ground vehicle (UGV) and are configured to log inertial data at 25 Hz while the UGV is driven via remote control through a predefined test course. The test course is designed such that the true linear velocity, position and orientation of the UGV can be accurately measured at periodic intervals along the course. This truth data will serve as the secondary navigation information that gets fed into the EKF component of the inertial navigation algorithm.
- The inertial information recorded from each IMU during the navigation tests is then post processed using the inertial navigation algorithm. The resulting navigation information is analyzed and compared to the true linear velocity, position and orientation of the UGV to determine which INS was able to more accurately estimate the linear velocity, position and orientation of the UGV over the course of the navigation tests.

The procedures described above are implemented using two low-cost MEMS IMUs. The first is a hobbyist-grade IMU from the website SparkFun.com, and the second is a commercial-grade IMU from the company Analog Devices, Inc. (ADI). The two IMUs can be seen in Figure 1.2. Both the SparkFun and ADI IMUs contain a three-axis accelerometer, a three-axis gyroscope and an embedded temperature sensor that is capable of measuring the internal temperature of the IMU package in degrees Celsius. Some performance specifications from the SparkFun and ADI IMU datasheets are listed in Table 1.1.

As Table 1.1 illustrates, the specifications for the ADI IMU are superior to the specifications for the SparkFun IMU in some categories, but not all of them. Given that the ADI IMU costs four times as much as the SparkFun IMU, it is intuitive to think that the navigation information produced using the ADI IMU will be much better than the navigation information produced using the SparkFun IMU. However, the relative quality of two IMUs cannot be determined just by looking at the differences in their

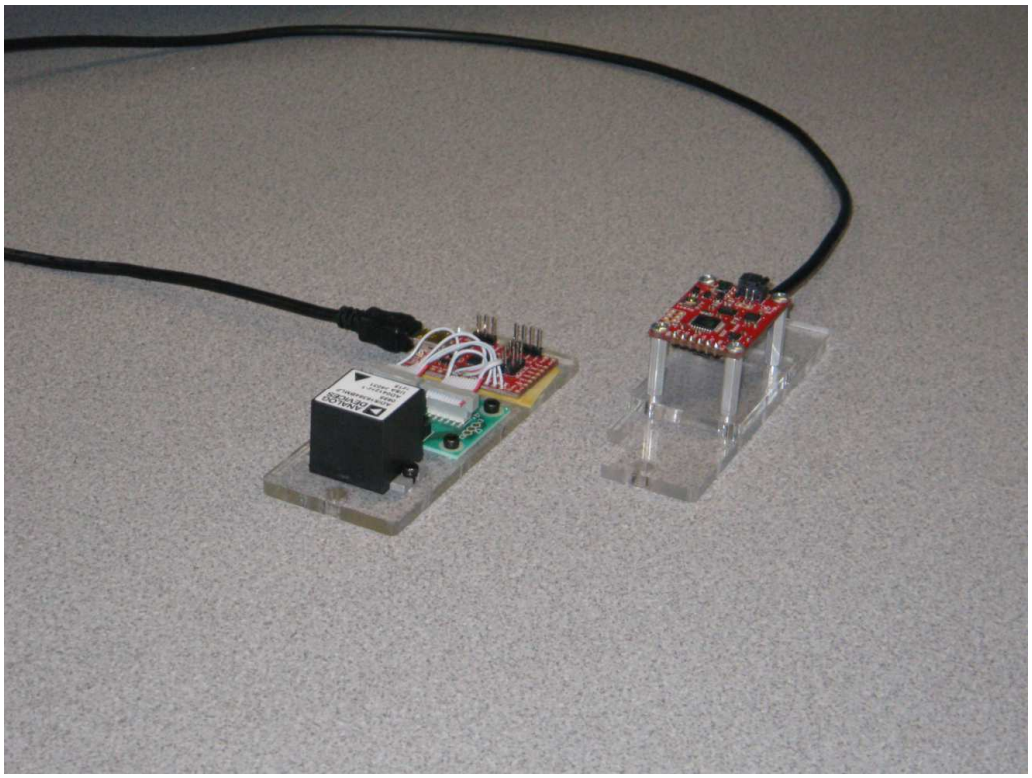

Figure 1.2: The ADI IMU (on the left) and the SparkFun IMU (on the right).

	SparkFun IMU	**ADI IMU**
Unit Information		
Model #	SEN-10736	ADIS16364
Cost Per Unit	$125	$500
Accelerometers		
Range	± 2 g	± 5 g
Resolution	3.9 mg/LSB	1 mg/LSB
0g Bias Level	± 40 mg (X and Y axis) ± 80 mg (Z axis)	8 mg
Noise Performance	< 3.9 mg RMS	5 mg RMS
Gyroscopes		
Range	± 2000 °/sec	± 350 °/sec
Resolution	0.0696 °/sec/LSB	0.05 °/sec/LSB
0g Bias Level	± 40 °/sec	± 3°/sec
Noise Performance	0.38 °/sec RMS	0.8 °/sec RMS

Table 1.1: Selected performance specifications for the SparkFun and ADI IMUs [1-3].

price tags. Only after performing the experiments outlined in this thesis can it be definitively said that one IMU outperforms the other in terms of the ability to provide accurate navigation information.

The remaining sections of this thesis are briefly summarized in the bulleted list below:

- Section 2 presents the background research performed for this thesis. Each of the five papers presented represents some previous research that has been conducted in the field of IMU error modeling, IMU error calibration or inertial navigation.
- Section 3 discusses the two main categories of IMU errors (deterministic and stochastic) in detail and breaks both categories down into several subcategories. The mathematical representations of each type of IMU error are also discussed.
- Section 4 explains the laboratory experiments designed to isolate and quantify each of the IMU errors discussed in Section 3. Each experiment is described in detail, including its purpose, how it is set up and the end results that are produced.
- Section 5 describes the inertial navigation algorithm in more detail. A brief explanation of the basic principles of inertial navigation is provided, the assumptions made by the algorithm are stated and each individual component of the inertial navigation algorithm is thoroughly explained.
- Section 6 outlines the INS testing process. The predefined test course for the UGV is presented along with a description of how the secondary navigation information for the EKF component of the inertial navigation algorithm is recorded.
- Section 7 presents and discusses the results obtained from the UGV navigation testing. The results are presented visually, displaying the estimated position and orientation of the UGV and the true position and orientation of the UGV on the same graph. A comparison of the two INS units is also presented to determine which INS was able to more accurately estimate the linear velocity, position and orientation of the UGV over the course of the navigation testing.
- Section 8 lists some potential ideas for future work, and Section 9 restates the goal of this thesis and summarizes the general conclusions that can be made about the results that were obtained.
- The Appendix contains supplemental figures detailing the complete results from all of the UGV navigation testing trials.

Section 2 – Background Research

The field of inertial navigation has been extensively researched and well documented over the course of the last fifty years. There are many published papers that document a variety of different ways to model the various types of errors that commonly appear in inertial data. This section presents five of these papers and explains how they are related to the work demonstrated in this thesis.

The first paper is titled "Calibration of Accelerometer Triad on an IMU with Drifting Z-Accelerometer Bias" [4]. The goal of this paper was to design a test that would quantify a series of error parameters (bias, scale factor, dis-symmetry and non-orthogonality errors) for a three-axis accelerometer with the prior knowledge that the Z-axis accelerometer bias varies with time. Normally, this type of test would be conducted under the assumption that all of the error parameters being estimated remain constant during the duration of the test. However, because the Z-axis accelerometer bias is known to vary with time, this assumption cannot be made. A method for estimating the Z-axis accelerometer bias is proposed that takes advantage of the correlation between the Z-axis accelerometer bias and the non-orthogonality errors of all three accelerometer axes. If the non-orthogonality errors are assumed to be constant and negligibly close to zero, then any non-zero values that appear in the estimated non-orthogonality terms must be a result of the drift in the Z-axis accelerometer bias. This relationship between the non-orthogonality errors and the Z-axis accelerometer bias drift can be used to estimate the steady-state value of the Z-axis accelerometer bias. The paper proves the validity of this concept through the use of a computer simulation.

This paper proved to be a useful resource because it describes a multi-angle experiment that is very similar to the one described in Section 4.3. The assertion that the Z-axis accelerometer bias drift can cause inaccuracies in all of the other error parameter estimates seems difficult to believe, although that assertion may be based on how long the IMU is left in each orientation. The Multi-Angle Experiment conducted in this thesis records five minutes worth of data at each orientation, so the amount of bias drift that occurs at any given orientation is most likely small enough to be considered negligible. If the bias drift is assumed to be negligible, then the experiment becomes much simpler because it can be assumed that all of the error parameters remain constant over the duration of the test.

The second paper is titled "Research on Auto Compensation Technique of Strap-Down Inertial Navigation Systems" [5]. The goal of this paper was to show that the heading estimates produced by an INS could be improved by mounting it to a vehicle using a platform that constantly rotates about the gravity vector. This is opposed to the more conventional method of mounting an INS rigidly to the body of a vehicle. As the platform rotates, it generates a constant angular velocity. This constant angular

velocity is measured by the INS and appears in the data as a change in one of the gyroscope biases. This additional bias term from the rotating platform ensures that even when the heading of the vehicle is constant, there is always some non-zero component of angular velocity that the INS can measure. Also, because the angular velocity of the rotating platform is known, error calibration routines can be run on the INS at all times. This results in more accurate heading estimates over longer periods of time. The paper goes on to prove this concept through the use of a computer simulation that models the implementation of this type of INS mounting strategy on a vehicle travelling with a fixed heading over a twenty four hour time period.

The rotating INS platform described in this paper is very similar to the Rate Table Experiment described in Section 4.4. Although the method described in the paper is for a continuous calibration process, the core idea of exposing the gyroscopes to a constant angular velocity in order to determine scale factor and misalignment errors can be adapted for use in a more comprehensive laboratory experiment. This paper provides some useful information on how to structure the Rate Table Experiment, and how to interpret the results that it produces.

The third paper is titled “Indoor 3D Position Estimation Using Low-Cost Inertial Sensors and Marker-Based Video-Tracking” [6]. The goal of this paper was to develop an indoor object-tracking system using a low-cost IMU and an external outside-looking-in video tracking system. The paper states that an IMU can be used to provide a rough estimate of the position, velocity, and orientation of an object, but that it is also subject to errors caused by biases and bias drifts. In order to compensate for these errors, an external video tracking system is used to provide corrections for the IMU estimates. The data returned from the IMU and the video tracking system is synchronized and fused together using an EKF, which results in a more accurate estimate of the detected object’s position, velocity, and orientation. The system was first tested on a set of experimental data to provide a proof of concept test for the system design, and then was subsequently tested on an actual object using the system hardware to provide proof that the system was capable of tracking real world objects. For the real world test, a test subject’s hand was used as the object to be tracked by the system. The IMU was strapped to the subject’s wrist, and a fiducial marker cube was attached to the subject’s arm to act as a target for the video tracking system to track. The results of both the simulated and real world experiments showed that the system was capable of accurately estimating the three-dimensional position and orientation of a detected object, and the position estimates were accurate to within a few centimeters of the true position values. The results also showed that the combination of the IMU and video tracking sensors resulted in a more robust and accurate system than if either of the two sensors were used on their own.

The primary reason for including this paper is because the structure of the EKF-based navigation system is very similar to the inertial navigation algorithm that is presented in this thesis. And while the

paper didn't mention any other IMU calibration techniques other than the bias tracking and compensation mechanism of the EKF, it is still a valuable resource to have because the navigation system that it presents is similar in design to the inertial navigation algorithm that this thesis is based around.

The fourth paper is titled "Thermal Calibration of MEMS Inertial Sensors for an FPGA-Based Navigation System" [7]. This paper focused on both the development and calibration of a custom IMU that uses an FPGA-based processor to produce real-time navigation information. It briefly discussed the process of developing the IMU which contained low-cost accelerometers and gyroscopes from Analog Devices, Inc., and the FPGA-based processor which was used for both data collection and navigation calculations. However, the majority of the paper discussed the calibration routines that are required to compensate for the deterministic and stochastic errors in the IMU data. Error sources such as biases, bias drifts, scale factors, misalignment errors, and thermal sensitivity were all discussed, and methods for compensating for them were explained. Because thermal sensitivity affects all of the other error parameters, all of the IMU calibration testing was performed under the umbrella of temperature regulated testing. The paper concluded by presenting the results of the calibration experiments, and by showing how accurately they were able to compensate for the IMU errors.

This paper contains useful information about IMU calibration in general (Allan Variance analysis, multi-angle experiments, etc.), but the primary reason for including it is because of the focus that it puts on thermal sensitivity and how it can affect IMU calibration. Because MEMS sensors are so sensitive to changes in temperature, it is important to realize that any calibration routines performed on an IMU will only yield results that will be usable at the temperature that the test was performed at. Therefore, in order to accurately compensate for IMU errors at different temperatures, calibration routines must be performed at a variety of different temperatures so that a model can be constructed that accurately describes how each of the different IMU errors changes with respect to temperature. This paper is a valuable reference to have because of the information that it provides regarding the effects of temperature variation on the behavior of IMU errors.

The fifth and final paper is titled "Estimation of Deterministic and Stochastic IMU Error Parameters" [8]. This paper breaks down the problems of IMU calibration into two main components: deterministic errors and stochastic errors. Experiments designed to quantify deterministic errors include multi-angle and rate table experiments, which expose the IMU to a series of controlled accelerations and angular velocities in order to determine how much the IMU measurements differ from the true accelerations and angular velocities that the IMU is actually experiencing. Experiments designed to model stochastic errors include estimation techniques such as Kalman Filtering, which produces estimates of the states of a dynamic system by using a model of the system and imperfect measurements of the system states. Stochastic errors are typically more difficult to compensate for due to their inherently random

nature. The paper uses both simulated and real IMU data to test the proposed calibration methods, and presents the successful results that the researchers were able to obtain. Overall, the results of this paper showed that it is possible to improve the performance of an IMU by quantifying and modeling the various deterministic and stochastic errors in the IMU data and then using that information to help compensate for the negative effects of the errors.

The primary reason for including this paper is that it is very similar to this thesis in terms of what it sets out to achieve and the methodology that it uses to achieve it. It attempts to estimate a very similar group of IMU error parameters using a very similar series of laboratory experiments, and it also utilizes many of the same mathematical concepts (primarily least-squares estimation and Kalman Filtering) that are used for performing error analyses in this thesis. The only major topic from this thesis that the paper doesn't cover is temperature sensitivity testing. Overall, this paper is a valuable reference to have because it describes several common types of IMU errors and how they behave, and it also provides a lot of insight into how to design experiments that will both isolate and quantify those errors.

Section 3 – IMU Errors and Error Models

There are some IMUs that are capable of producing incredibly accurate measurements of acceleration and angular velocity. However, there is no IMU that can produce perfect measurements of acceleration and angular velocity. No matter how good an IMU is, there will always be imperfections in the inertial data that it returns. These imperfections are caused by several different kinds of errors, and each has a different effect on the data. IMU errors can be broken up into two primary categories: deterministic errors and stochastic errors. Each of these categories can then be further broken down into a series of subcategories, one for each different kind of error. By understanding each of these errors and modeling their behavior mathematically, it becomes possible to compensate for them and improve the overall quality of the IMU data.

It should be noted that there are many different types of IMU errors, and only a subset of them are discussed in this thesis. A substantial amount of time can be devoted to precisely modeling and understanding the behavior of all of the different types of IMU errors that have been previously documented. However, after a certain extent the benefit from doing so becomes less and less. Because of this, a specific group of IMU errors have been selected that are deemed to be the most important and are thought to have the largest effect on the integrity of the IMU data. Those are the IMU errors that are explored in the following subsections.

Section 3.1 – Deterministic Errors

The first category of IMU error is deterministic errors. Deterministic errors can be precisely determined and represented using either scalar numbers or matrices. Their values either remain constant over the course of time or vary in a way that can be exactly mathematically modeled. Once a deterministic error has been accurately modeled, its model will typically remain the same regardless of how long the IMU is allowed to run or whether power is cycled on the IMU. The following subsections explain each type of deterministic error in detail and present the mathematical models that are used to characterize their behavior.

Section 3.1.1 – Static Biases

The first type of deterministic error is static biases. A static bias is simply a positive or negative constant that offsets data by a specific amount. An example of a static bias can be seen in Figure 3.1. Static biases in IMU data are modeled as a pair of 3x1 vectors as shown below:

$$b_a = \begin{bmatrix} b_a^x \\ b_a^y \\ b_a^z \end{bmatrix} \qquad b_g = \begin{bmatrix} b_g^x \\ b_g^y \\ b_g^z \end{bmatrix} \qquad \text{Eq. 3.1 \& 3.2}$$

where b_a represents the static biases for the three-axis accelerometer and b_g represents the static biases for the three-axis gyroscope. Normally these vectors would be enough to completely model the static biases. However, due to the temperature sensitive nature of MEMS IMUs, the values of the static biases change with respect to temperature. The MEMS gyroscopes in particular are more sensitive to temperature than the MEMS accelerometers, but both sensors must be analyzed for the sake of completeness. Because of this relationship between the static biases and temperature, the model of the static biases changes from a single 3x1 vector pair to a series of 3x1 vector pairs arranged in a look-up table based on temperature. The look-up table takes in a specific input temperature and produces the static bias vectors that correspond to that temperature.

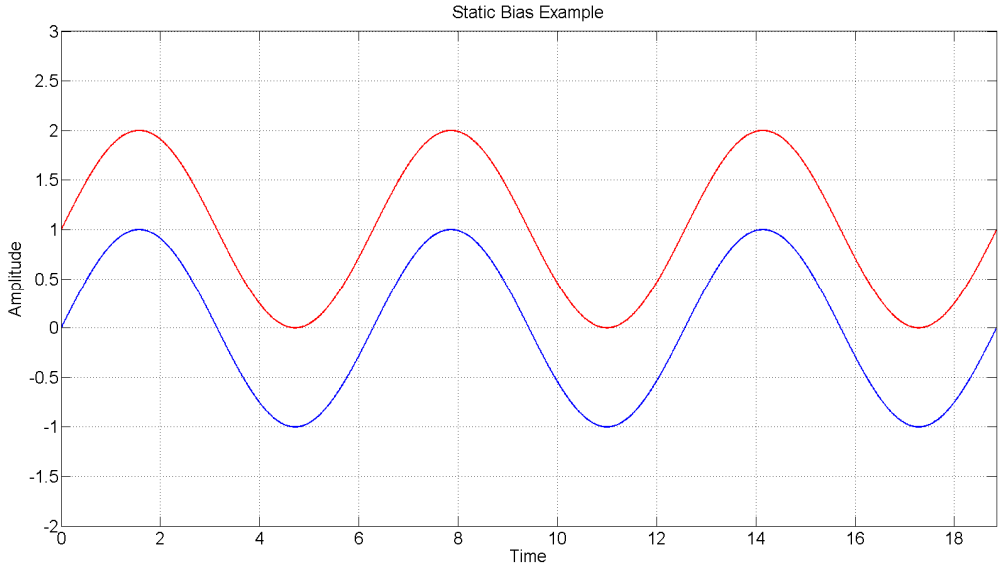

Figure 3.1: An example of an unbiased sine wave (blue) and a biased sine wave with a positive static bias of 1 (red).

Section 3.1.2 – Scale Factors and Misalignment Errors

The second and third types of deterministic errors are scale factors and misalignment errors. Scale factors will be examined first. Scale factors are similar to static biases in that they offset data by a specific amount. However, instead of offsetting data by an additive constant, scale factors offset data by a multiplicative constant. An example of a scale factor can be seen in Figure 3.2. Scale factors in IMU data are modeled as a pair of 3x3 diagonal matrices as shown below:

$$S_a = \begin{bmatrix} S_a^x & 0 & 0 \\ 0 & S_a^y & 0 \\ 0 & 0 & S_a^z \end{bmatrix} \qquad S_g = \begin{bmatrix} S_g^x & 0 & 0 \\ 0 & S_g^y & 0 \\ 0 & 0 & S_g^z \end{bmatrix} \qquad \text{Eq. 3.3 \& 3.4}$$

where S_a represents the scale factors for the three-axis accelerometer and S_g represents the scale factors for the three-axis gyroscope. As was the case with the static biases in Section 3.1.1, scale factors are also sensitive to temperature. Therefore, instead of being modeled as a single 3x3 matrix pair, scale factors must be modeled as a series of 3x3 matrix pairs arranged in a look-up table based on temperature. The look-up table takes in a specific input temperature and produces the scale factor matrices that correspond to that temperature.

Now misalignment errors will be examined. Misalignment errors occur when the axes of an IMU's sensors are not properly aligned with the X, Y and Z directions defined by the IMU package. An

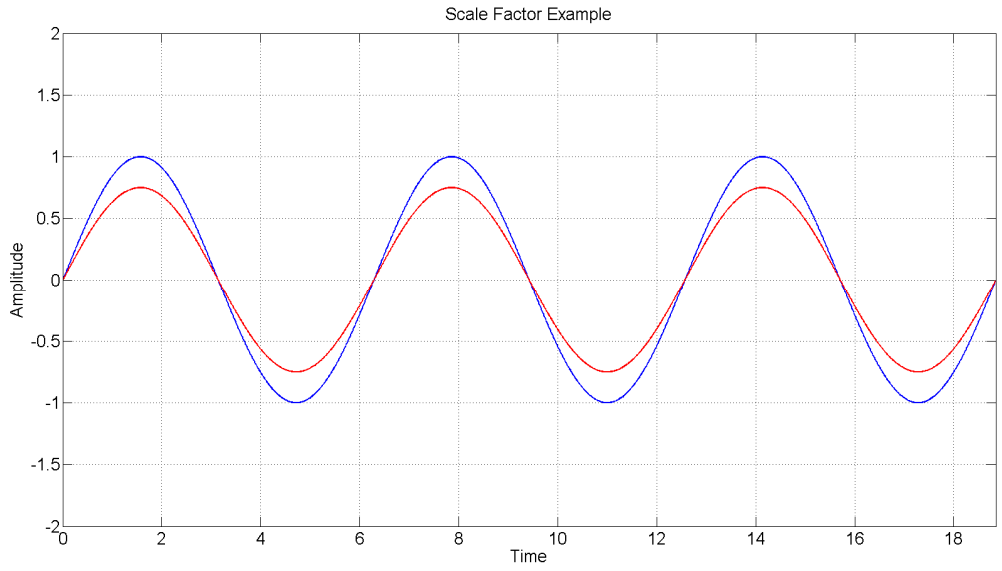

Figure 3.2: An example of a non-scaled sine wave (blue) and a scaled sine wave with a positive scale factor of 0.75 (red).

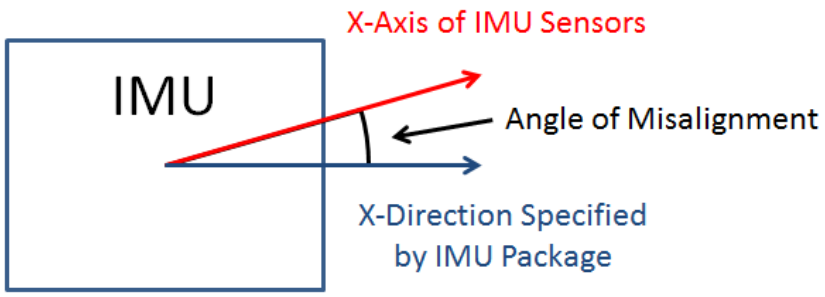

Figure 3.3: An example of a misalignment between the X direction specified by the IMU package and the X-axis of the IMU sensors.

example of an axis misalignment can be seen in Figure 3.3.

Since IMUs are typically constructed in square or rectangular formats, the X, Y and Z directions are usually defined by the unit vectors that are exactly perpendicular to the faces of the IMU package. Ideally, the axes of the IMU should coincide with the specified X, Y and Z directions exactly. However, it is very unlikely that this is actually the case. If an axis is not properly aligned with its corresponding direction, then that axis will measure portions of the accelerations and angular velocities that occur in the other two directions.

Misalignment errors are particularly noticeable when dealing with acceleration due to gravity. Suppose that an IMU is sitting on a flat and level lab bench recording data. The axis parallel to the gravity vector will measure a very large acceleration compared to the other two axes in the plane perpendicular to the gravity vector (which should each measure close to zero acceleration). If one of those two axes is misaligned, even by a small amount, it will measure a component of the gravity vector that is proportional to the amount of misalignment. For example, a misalignment of just 1° with respect to the plane perpendicular to the gravity vector would cause a measurement of $9.80665 * \sin(1°) = 0.1711$ meters per second squared in the misaligned axis. This amount of acceleration error may not seem like much of a problem, but it could be enough to drown out or heavily distort the true acceleration being experienced in that axis. More importantly, as that acceleration error propagates through the inertial navigation algorithm, it will cause the estimates of linear velocity, position and orientation to drift significantly within a matter of minutes or even seconds. In fact, over the course of just five minutes the constant acceleration error caused by a 1° misalignment would result in a final velocity error of 51.3449 meters per second and a final position error of 7701.7 meters.

Misalignment errors in IMU data are modeled as a pair of 3x3 matrices as shown below:

$$M_a = \begin{bmatrix} m_a^{xx} & m_a^{xy} & m_a^{xz} \\ m_a^{yx} & m_a^{yy} & m_a^{yz} \\ m_a^{zx} & m_a^{zy} & m_a^{zz} \end{bmatrix} \qquad M_g = \begin{bmatrix} m_g^{xx} & m_g^{xy} & m_g^{xz} \\ m_g^{yx} & m_g^{yy} & m_g^{yz} \\ m_g^{zx} & m_g^{zy} & m_g^{zz} \end{bmatrix} \qquad \text{Eq. 3.5 \& 3.6}$$

where M_a represents the axis misalignments in the three-axis accelerometer and M_g represents the axis misalignments in the three-axis gyroscope. Each of the terms in the M_a and M_g matrices specifies how accelerations and angular velocities in X, Y and Z directions are measured by the X, Y and Z IMU axes. For example, the m_a^{xy} term specifies the component of acceleration in the Y direction that is measured by the X-axis of the IMU. Ideally, all of the diagonal terms in the M_a and M_g matrices would be equal to one and all of the other terms would be equal to zero, making both M_a and M_g equal to the identity matrix I. That would mean that the IMU axes are perfectly aligned with the IMU package and each axis is only ever measuring acceleration and angular velocity along or about its own axis. In reality this is never the case, but the closer the M_a and M_g matrices are to the identity matrix I the smaller the misalignment errors will be and the better the IMU data will be.

Scale factors and misalignment errors both affect IMU data in very similar ways, and therefore the mathematical models that define their behavior are also very similar. Because their mathematical models are so similar, the individual effects of the two errors are difficult to differentiate from one another. To deal with this issue, the two errors are modeled together as one joint error term. This does not present a problem in the overall analysis of the IMU data because the effects of the scale factors and misalignment errors do not need to be known independently. Both of the errors can be compensated for simultaneously if the combination of their effects is known. The combination of scale factors and misalignment errors in the IMU data is modeled using a pair of 3x3 matrices as shown below:

$$C_a = M_a S_a \qquad C_g = M_g S_g \qquad \text{Eq. 3.7 \& 3.8}$$

where C_a represents the combination of the scale factors and misalignment errors in the three-axis accelerometer and C_g represents the combination of the scale factors and misalignment errors in the three-axis gyroscope. The temperature sensitive nature of the scale factors described earlier in this section carries over into the C_a and C_g matrices as well. Therefore, as was the case with the S_a and S_g matrices, the C_a and C_g matrices must be modeled as a series of 3x3 matrix pairs arranged in a look-up table based on temperature instead of as a single static 3x3 matrix pair. The look-up table takes in a specific input temperature and produces the combination of scale factors and misalignment errors that corresponds to that temperature.

Section 3.2 – Stochastic Errors

The second category of IMU error is stochastic errors. Unlike deterministic errors, which are inherently stable and repeatable, the behavior of stochastic errors is based around random processes. The random nature of stochastic errors means that their values change constantly over the course of time and there is no way to predict what the exact value of a specific error term will be at any given point in time. It also means that if power is cycled on the IMU the values of the stochastic errors will be completely different from any previous time that the IMU was used. Because of their inherently random nature the only way to model stochastic errors is by using random variables and probabilistic models. However, it is assumed that the probability distributions used to model the stochastic errors remain the same over the course of time. The following subsections explain each type of stochastic error in detail and present the mathematical models that are used to characterize their behavior.

Section 3.2.1 – Measurement Noise

The first type of stochastic error is measurement noise. Measurement noise is a zero-mean random process that appears in sensor data and obscures its true nature. An example of measurement noise can be seen in Figure 3.4. In order to accurately model measurement noise, two properties of the measurement noise must be understood. The first property is the color of the measurement noise, and the

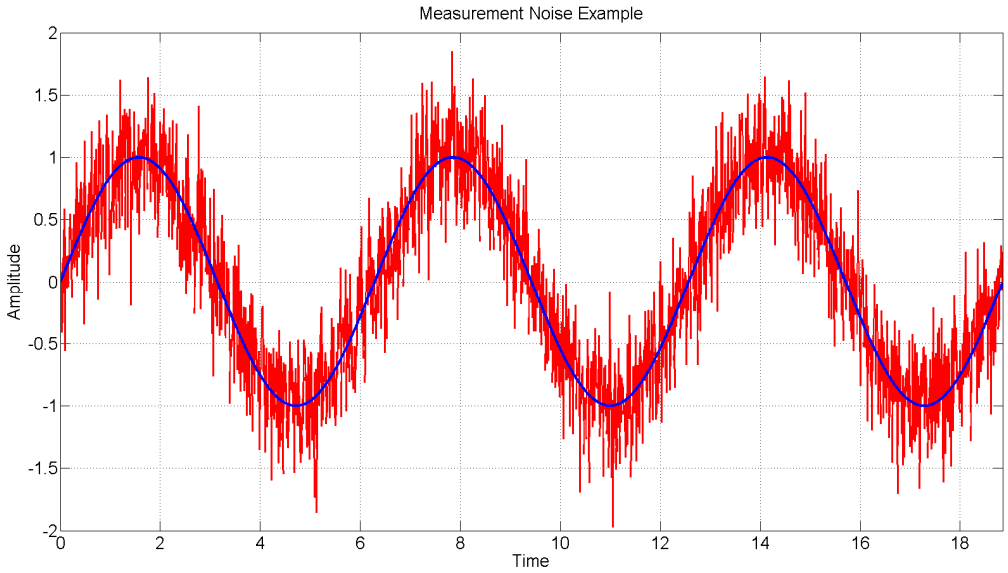

Figure 3.4: An example of a sine wave without noise (blue) and a sine wave with a white Gaussian noise process ($\mu = 0$, $\sigma^2 = 0.1$) added to it (red).

second property is the probability density function (PDF) that the measurement noise samples are drawn from.

The first property used to characterize measurement noise is called the noise color. A noise process is said to have a certain color based on the relationship between its power spectrum and frequency. For example, white noise has a power spectrum that is flat or equal valued at all noise frequencies. Pink noise has a power spectrum that is inversely proportional to frequency ($1/f$), and Brownian or random walk noise has a power spectrum that is inversely proportional to the square of the frequency ($1/f^2$). Some noise colors are relatively easy to model, and some noise colors are not. White noise and Brownian noise are very simple to model and are also very easy to work with mathematically. Pink noise, on the other hand, cannot be represented using a finite model and can only ever be approximated to within a certain degree of accuracy. White noise is one of the more common noise colors found in both natural and man-made systems. Because of this, noise processes will sometimes be assumed to be white noise if no other information about the noise process seems to suggest otherwise. However, the more appropriate method for determining the color of a noise process is to perform a mathematical analysis such as an Allan Variance or Power Spectral Density (PSD) analysis to get a definitive characterization of the true color of the noise. These analyses are described in more detail in Sections 4.1.1 and 4.1.2 respectively.

The second property used to characterize measurement noise is called the noise probability density function (PDF). A PDF is a function that describes how samples from a random noise process are distributed. As was the case with noise color, there are several different categories of noise PDFs that are represented by different families of functions. Some examples of PDF families include Gaussian PDFs, Uniform PDFs, Exponential PDFs, Poisson PDFs, Rayleigh PDFs, and many more. One of the most commonly used PDF families is the family of Gaussian PDFs (also referred to as Normal PDFs). Gaussian PDFs are defined by the equation:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \qquad \text{Eq. 3.9}$$

where μ is the mean of the distribution and σ^2 is the variance of the distribution. The Gaussian PDF has several properties that make it easier to work with than most PDFs. First, the behavior of many natural and man-made random processes can be accurately modeled using Gaussian PDFs. Second, a Gaussian PDF can be completely defined using only two parameters: the mean of the distribution μ and the variance of the distribution σ^2. And third, if a linear operation is performed on a Gaussian distributed random variable, the resulting random variable is also Gaussian distributed. This makes Gaussian

distributed random variables very easy to work with in linear systems because any Gaussian distributed random variable that goes into the system will produce a random variable at the output of the system that is also Gaussian distributed, but with a potentially different mean and variance.

Because Gaussian distributed noise occurs so frequently and is so desirable mathematically, it is common practice to initially assume that an unknown noise process is Gaussian distributed if no other information about the PDF of the noise is available (this assumption can be justified based on the Central Limit Theorem [9]). However, care must be taken to ensure that this assumption is not made recklessly or without proper justification. The actual PDF of a noise process should always be verified if possible. Verifying the PDF of a noise process is described in more detail in Section 4.1.3.

Section 3.2.2 – Turn-On to Turn-On Bias Variation

The second type of stochastic error is turn-on to turn-on bias variation. In Section 3.1.1, static biases were discussed and classified as a type of deterministic error because their values remain constant over the course of time. However, that is only true if the static biases are being observed during a single continuous power cycle. If the static biases are being observed over multiple continuous power cycles, then there will be slight variations in their values from power cycle to power cycle.

Suppose for example that an IMU is powered on to collect a sample of data, then powered down after the data collection is complete and then powered on again to perform a second data collection. When the IMU powers on again to perform the second data collection, the values of the static biases will be slightly different from the values that were observed during the first data collection. This variation in the values of the static biases between power cycles is what the turn-on to turn-on bias variation models. The turn-on to turn-on bias variation for each of the six static biases is modeled as a random process with a mean and a variance. These variances are used in the initialization of the EKF component of the inertial navigation algorithm to provide initial covariance values for the bias correction terms in the EKF's state vector.

Section 3.2.3 – Drifting Biases

The third type of stochastic error is drifting biases. A drifting bias is a bias that starts off at a specific value and then randomly drifts away from that value slowly over time (this type of behavior is commonly referred to as a "random walk" behavior). An example of a drifting bias taken from a sample of gyroscope data can be seen in Figure 3.5. Unlike static biases, drifting biases contain a random

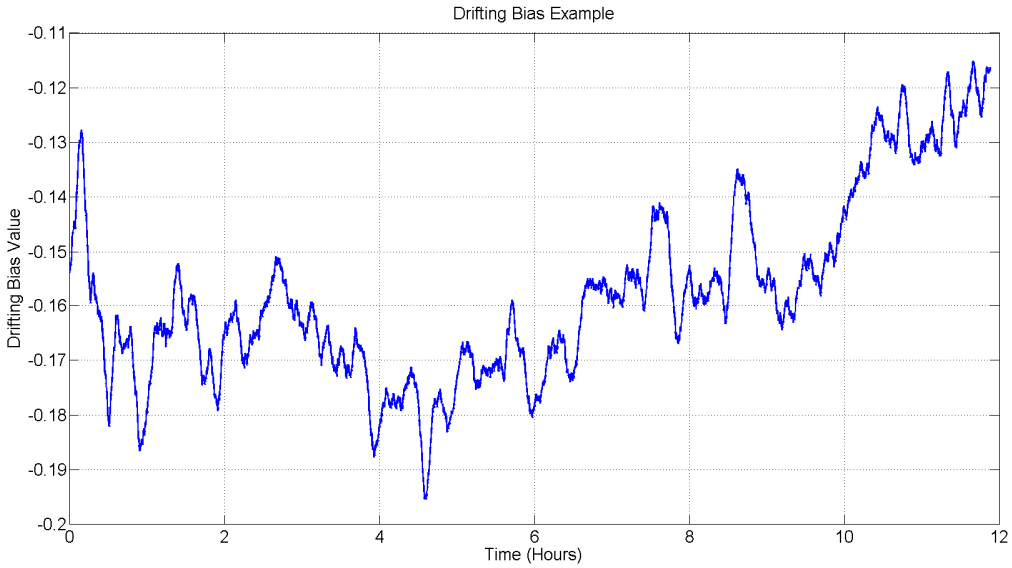

Figure 3.5: An example of a drifting bias in a sample of IMU gyroscope data.

component that dictates how its value will change over the course of time. This random component must be taken into account by whatever model is used to approximate the behavior of the drifting bias. The model chosen to approximate the behavior of the drifting bias is the first-order Markov process model. The drifting bias model is described by the recursive equation:

$$b_d(i) = \varphi b_d(i-1) + \varepsilon(i) \qquad \text{Eq. 3.10}$$

where $b_d(i)$ is the value of the drifting bias at time i, φ is a scale factor, and ε is a zero-mean white Gaussian noise (WGN) process with unknown variance σ_{bd}^2. During each iteration of the equation, the current value of the drifting bias ($b_d(i)$) is correlated with the previous value ($b_d(i-1)$) through the scale factor (φ), and a sample from the WGN process ($\varepsilon(i)$) is added in to simulate the random component of the drifting bias. One of the benefits of using the first-order Markov process model to approximate the behavior of the drifting biases is that because it is a recursive model that is driven by a WGN process, it is very simple to integrate into the equations of an EKF. In terms of the inertial navigation algorithm as a whole this means that it is also very simple to compensate for the effects of the drifting biases, which will improve the overall quality of the linear velocity, position and orientation estimates that the inertial navigation algorithm produces.

Section 4 – Laboratory Experiments

In Section 3, several types of IMU errors were discussed and the mathematical models for characterizing their behavior were developed. This section describes the laboratory experiments that are used to determine values for the parameters of the generic IMU error models from Section 3. Each of the experiments is designed to isolate the effects of one or more of the IMU errors so that their behavior can be studied independently of the effects of the other IMU errors. Then, by controlling the accelerations and angular velocities that the IMU experiences during an experiment, the specific error being studied can be accurately quantified. By the end of this section, each of the IMU errors will have a specific mathematical model that is tailored to the individual SparkFun and ADI IMUs being used in this thesis.

Each of the following subsections describes a specific experiment in detail, including what the goal of the experiment is, how the experiment is set up, how the experiment is run, the type of data that is recorded and how that data is used to calculate the desired IMU error parameters.

Section 4.1 – 12-Hour Static IMU Data Experiment

The goal of the 12-Hour Static IMU Data Experiment is to gather a twelve hour sample of static IMU data so that the behavior of the measurement noise and drifting biases can be analyzed. Because the IMU is kept static (meaning that the IMU experiences no accelerations other than those caused by the Normal force opposing gravity and no angular velocities other than those caused by the rotation of the Earth) during the duration of the experiment, the only errors that affect the data are measurement noise, drifting biases, static biases and misalignment errors. In order to isolate the effects of the measurement noise and drifting biases, it will be assumed that subtracting the average value of the data from the entire data sample will be enough to sufficiently remove the effects of the static biases and misalignment errors. The experiment is run for twelve hours to ensure that the drifting biases have enough time to accrue a significant amount of drift (ten to twelve hours should be enough time for the drifting biases in a typical IMU to drift off by a noticeable amount). Once the measurement noise and drifting biases have been isolated, their behaviors can be properly analyzed.

The 12-Hour Static IMU Data Experiment is conducted by following the series of steps in the bulleted list below:

- The IMU is securely mounted on a test fixture that can be adjusted to ensure that the IMU is completely level (See Figure 4.1).

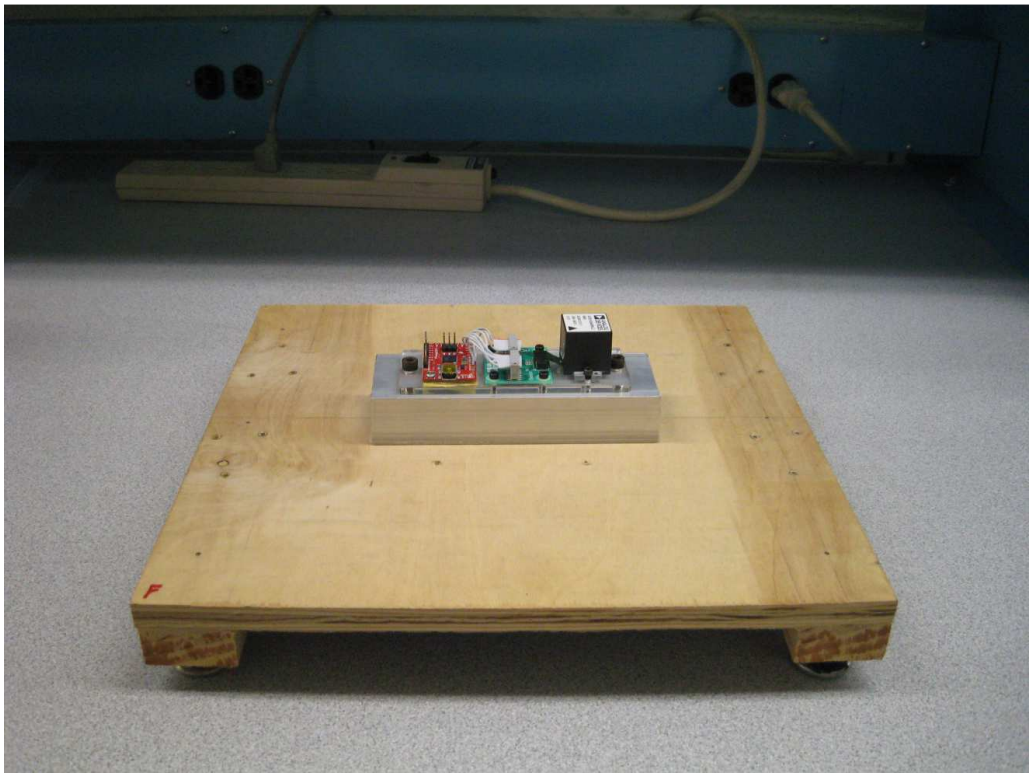

Figure 4.1: The test fixture for the 12-Hour Static IMU Data Experiment.

- The test fixture is then enclosed in a hybridization incubator to prevent the internal temperature of the IMU from changing radically during the course of the test (See Figure 4.2). Because this test lasts for so long, it is inevitable that the internal temperature of the IMU will change over time as the temperature of the room changes. Keeping the IMU in the incubator keeps the temperature fluctuations to a minimum.
- The IMU is then powered on and allowed to come up to temperature (the initial powering on of the IMU causes its internal temperature to rise approximately 1-2°C).
- Once the IMU has reached a stable temperature, the data logging begins. The data logging lasts for twelve hours. It is important that the IMU remains completely motionless during this time. Ideally, the IMU should be left alone in a room by itself with no people around it to ensure that no unintended vibrations or other disturbances appear in the recorded data.
- Once the data has been recorded, it is filtered to remove outliers and other anomalies. For each data point ($x(n)$) in the sample, the difference between $x(n)$ and the mean value of the sample ($\bar{x}$) is calculated. If the absolute value of that difference is above a certain threshold (typically 1 meter per second squared for the accelerometers and 1 degree per second for the gyroscopes) then $x(n)$ is replaced with the median value of the vector $[x(n-1) \quad x(n) \quad x(n+1)]$. If $x(n)$ is the

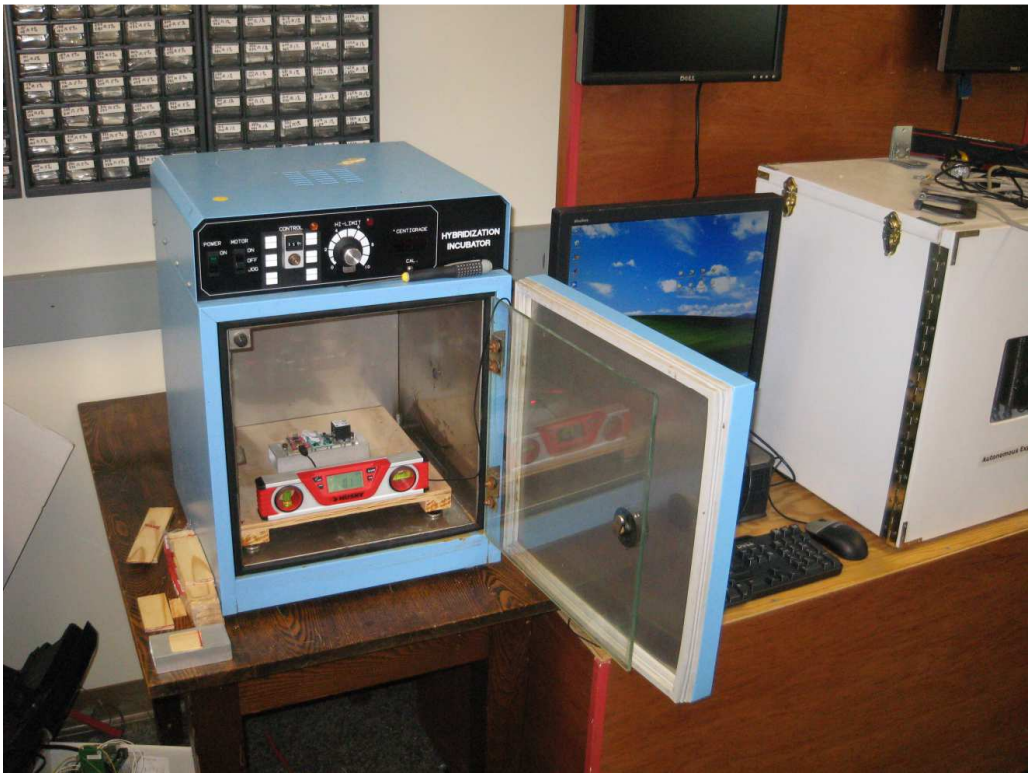

Figure 4.2: The test fixture for the 12-Hour Static IMU Data Experiment enclosed in a hybridization incubator.

first or the last data point in the sample, then it is replaced with the median value of the vector $[\bar{x} \quad x(n) \quad x(n+1)]$ or the median of the vector $[x(n-1) \quad x(n) \quad \bar{x}]$ respectively. This process removes all of the outliers and other anomalies from the raw inertial data.

- After the data has been filtered, the average value of the recorded data from each individual sensor axis is subtracted from every point in its respective data sample to remove the effects of the static biases and misalignment errors. The resulting data samples represent only the effects of the measurement noise and the drifting biases.

The following subsections describe the various methods that are used to analyze the data gathered from this experiment.

Section 4.1.1 – Allan Variance Analysis

The first analysis performed on the static IMU data is called an Allan Variance analysis. Allan Variance was originally developed by Dr. David W. Allan as a method for analyzing the frequency stability of precision oscillators [10]. However, as mentioned previously in Section 3.2.1, Allan Variance

can also be used to analyze a noise process and determine the color or combination of colors that best models its behavior.

The method for calculating Allan Variance is as follows. Suppose there exists a time series of discrete data points $x(t)$ that is N elements long and every element is separated by a constant amount of time Δt. An integer n is chosen and $x(t)$ is divided into m segments of n data points. If there are any points left over at the end of $x(t)$ that don't fill up an entire segment then they are truncated. A time $\tau = n\Delta t$ is computed that represents the amount of time spanned by each segment, and the time average of each of the m segments is computed to produce a series of averages denoted by $(a_1(\tau), a_2(\tau) \dots a_m(\tau))$. The averages are then used to compute the Allan Variance $(\sigma^2(\tau))$ for a given value of τ using the equation

$$\sigma^2(\tau) = \frac{1}{2(m-1)} \sum_{i=1}^{m-1} (a_{i+1}(\tau) - a_i(\tau))^2 \qquad \text{Eq. 4.1}$$

This process is then repeated for values of n ranging between 1 and $(N/10)$. The upper limit of $(N/10)$ is imposed because if there are fewer than nine averages used in the calculation of the Allan Variance, then the results begin to lose their significance [11]. Once the Allan Variance has been calculated over a sufficient range of n values, the square root of the Allan Variance (called the Allan Deviation) is plotted against the corresponding values of τ on a set of log-log axes. This final result is called the Allan Deviation function.

Once the Allan Deviation function of a noise process has been calculated and plotted, some insights can be gained about the color of that noise process. Each type of noise color produces a specific type of Allan Deviation function that is unique to that noise color. For example, the Allan Deviation function of white noise is a line with a slope of -½ (as seen in Figure 4.3). The Allan Deviation function of pink noise is a line with a slope of 0, and the Allan Deviation function of Brownian or random walk noise is a line with a slope of +½. Using this unique relationship between the Allan Deviation function and noise color, it is possible to identify the color of any noise process by calculating its Allan Deviation function and comparing it to the Allan Deviation functions of each noise color. Therefore, the color of the IMU measurement noise can be determined by computing the Allan Deviation function of the static IMU data (which only contains the effects of the measurement noise and drifting biases) and comparing it to the Allan Deviation functions of each of the noise colors. The Allan Deviation functions of the static IMU data can be seen in Figures 4.4 and 4.5.

The Allan Deviation functions in Figures 4.4 and 4.5 all have very similar characteristics. Initially, when τ is small, the Allan Deviation functions are all linearly decreasing. These portions of the

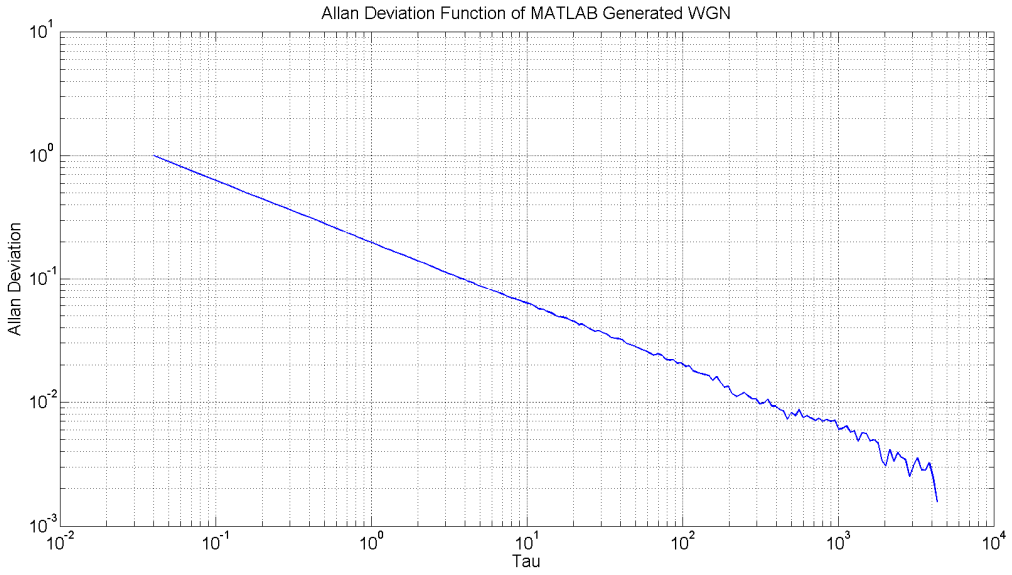

Figure 4.3: The Allan Deviation function of a sample of WGN generated by MATLAB.

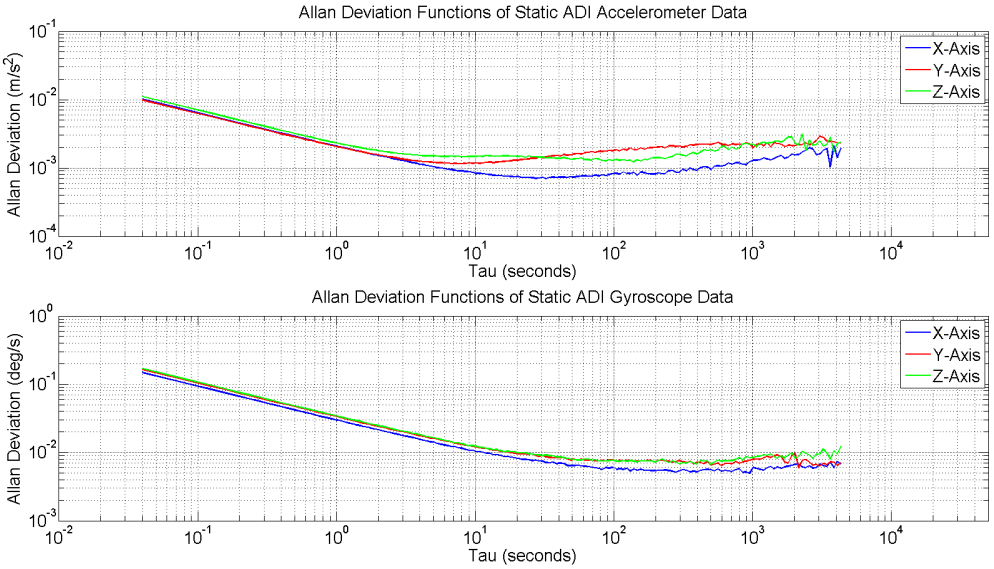

Figure 4.4: The Allan Deviation functions of static data from the ADI IMU accelerometers and gyroscopes.

Allan Deviation functions reflect the behavior of the measurement noise. Because of the slow-moving nature of the drifting biases, when τ is small the effects of the measurement noise overpower the effects of the drifting biases. However, as τ gets larger, the Allan Deviation functions all start to flatten out and curve back up towards a slope value of +½. These portions of the Allan Deviation functions reflect the behavior of the drifting biases. As τ gets larger, the effects of the drifting biases start to become more

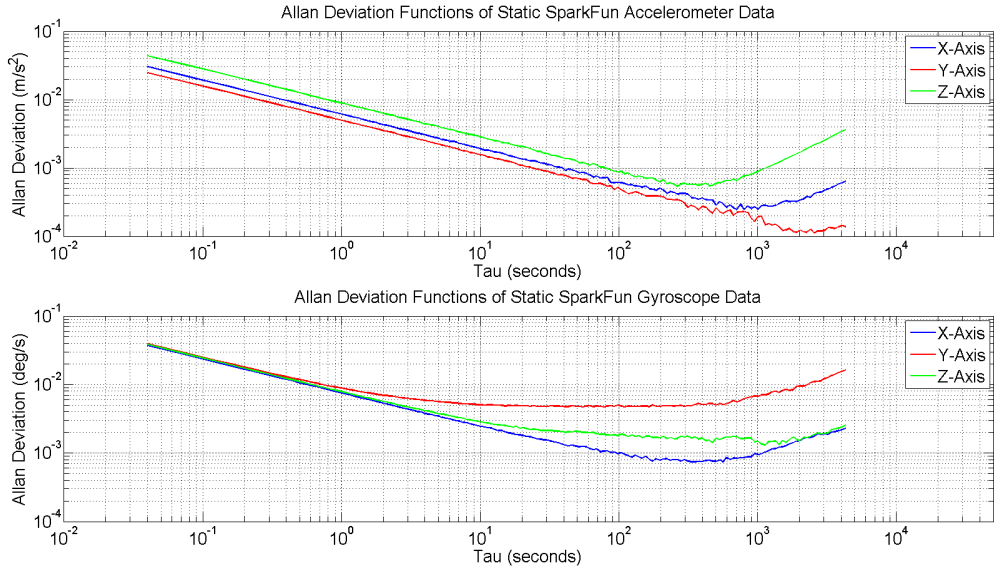

Figure 4.5: The Allan Deviation functions of static data from the SparkFun IMU accelerometers and gyroscopes.

ADI IMU	Allan Deviation Function Slope	SparkFun IMU	Allan Deviation Function Slope
X-Axis Accelerometer	-0.4968 $(m/s^2)/s$	X-Axis Accelerometer	-0.5023 $(m/s^2)/s$
Y-Axis Accelerometer	-0.4941 $(m/s^2)/s$	Y-Axis Accelerometer	-0.5012 $(m/s^2)/s$
Z-Axis Accelerometer	-0.4954 $(m/s^2)/s$	Z-Axis Accelerometer	-0.4951 $(m/s^2)/s$
X-Axis Gyroscope	-0.4980 (°/s)/s	X-Axis Gyroscope	-0.4977 (°/s)/s
Y-Axis Gyroscope	-0.4955 (°/s)/s	Y-Axis Gyroscope	-0.4832 (°/s)/s
Z-Axis Gyroscope	-0.4943 (°/s)/s	Z-Axis Gyroscope	-0.4978 (°/s)/s

Table 4.1: Measurement noise Allan Deviation function slope values for the ADI and SparkFun IMUs.

prevalent, and eventually they overpower the effects of the measurement noise. Since the ultimate goal of this analysis is to determine the color of the IMU measurement noise, the later portions of the Allan Deviation functions will be ignored for now and the rest of the analysis will focus on the early parts of the Allan Deviation functions where the effects of the measurement noise are the strongest. The later portions of the Allan Deviation functions will be discussed more in Section 4.1.4.

Upon inspection, the portions of the Allan Deviation functions that reflect the behavior of the measurement noise are very similar to the Allan Deviation function of white noise. To verify that the Allan Deviation functions of the measurement noise do truly match up with the Allan Deviation function of white noise, the value of their slopes must be determined. The calculated slopes of the measurement noise Allan Deviation functions are listed in Table 4.1.

As Table 4.1 shows, all of the slopes are very close to the value of -½ that is indicative of white noise. This confirms that the IMU measurement noise can be accurately modeled using the white noise model.

Section 4.1.2 – Power Spectral Density (PSD) Analysis

The second analysis performed on the static IMU data is called a Power Spectral Density (PSD) analysis. Like the Allan Variance analysis described earlier, a PSD analysis provides another method for determining the color of a noise process. PSD analyses are commonly used in the field of signals analysis to determine the amount of power that a signal has at specific frequencies. Its most basic mathematical form is defined as the Fourier transform of the autocorrelation function of the signal being analyzed

$$S(f) = F\big(R(\tau)\big) \qquad \text{Eq. 4.2}$$

where $S(f)$ is the PSD and $R(\tau)$ is the autocorrelation function of the signal. However, there are many variations of PSD and many different ways in which it can be calculated. The method used in this thesis is called the Welch method. The Welch method was developed by Peter Welch in 1967 as a more computationally efficient way of estimating the PSD of a signal by calculating the Fourier transforms of small, overlapping segments within the signal and averaging them together to produce an estimate of the PSD of the entire signal [12].

The Welch method is briefly described below, and a more detailed description can be found in Welch's paper [12]. Suppose there exists a time series of discrete data points $X(j)$ that is N elements long and every element is separated by a constant amount of time Δt. First, $X(j)$ is divided up into K fixed-length, overlapping segments such that the segments cover the entire length of $X(j)$ as illustrated in Figure 4.6. Each of the K segments is then filtered by subtracting the average value of the segment from each of the data points within the segment to remove any biases that may be in the data [13]. Then, each

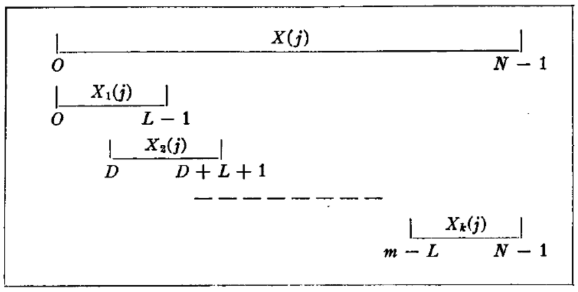

Figure 4.6: The division of a time series $X(j)$ into a group of K fixed-length overlapping segments [12].

of the K segments is multiplied by a windowing function (any standard windowing function such as Gaussian, Hamming, Hann, etc. can be used) and its Fourier transform is calculated using the fast Fourier transform (FFT) algorithm. This results in K Fourier transforms that are then averaged together to produce an estimate of the PSD of the entire time series $X(j)$. The Welch PSD analyses performed in this thesis were performed using a Gaussian windowing function of size 16384 and an overlap of 50% [12].

As was the case with Allan Variance, there is also a unique relationship between the color of a noise process and its PSD. These power spectrum to frequency relationships were described earlier in Section 3.2.1, but they will be briefly restated here for convenience. White noise has a power spectrum that is flat or equal valued at all noise frequencies (as seen in Figure 4.7), pink noise has a power spectrum that is inversely proportional to frequency ($1/f$), and Brownian or random walk noise has a power spectrum that is inversely proportional to the square of the frequency ($1/f^2$). Using these unique relationships between PSD and noise color, it is possible to identify the color of any noise process by calculating its PSD and comparing it to the PSDs of each noise color. Therefore, the color of the IMU measurement noise can also be determined by taking the PSD of the static IMU data and comparing it to the PSDs of each of the noise colors. The PSDs of the static IMU data can be seen in Figures 4.8 and 4.9.

As Figures 4.8 and 4.9 show, the PSDs are all essentially flat, which is indicative of white noise. However, in the range of frequencies less than 1 Hz, the PSDs start to stray away from the flat spectrum model. This behavior is caused by the drifting biases. As the frequency gets lower the effects of the drifting biases begin to overpower the effects of the measurement noise and the PSDs start to stray away from the flat model. But when the frequency gets higher, the effects of the measurement noise begin to

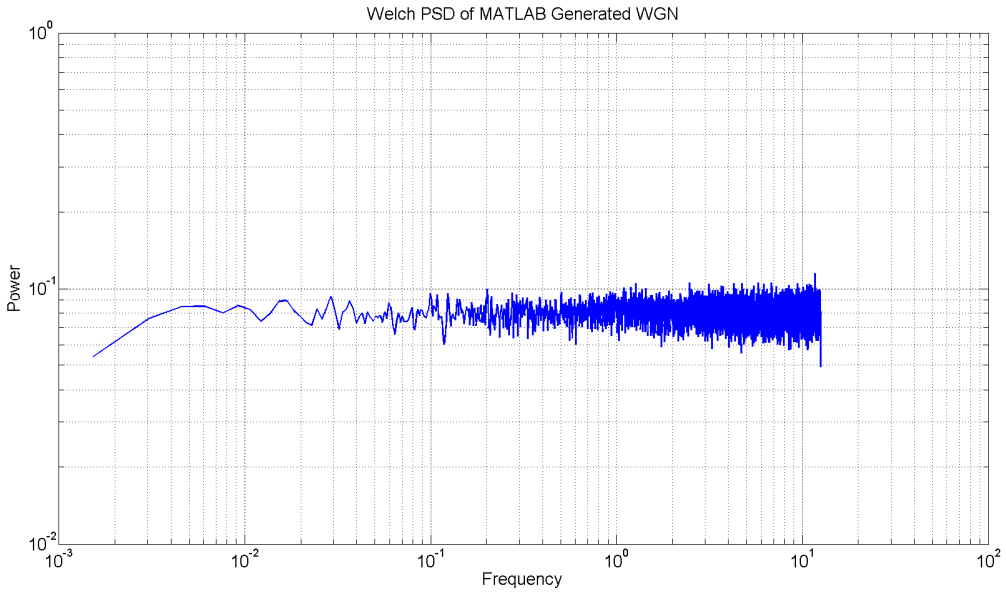

Figure 4.7: The Welch PSD of a sample of WGN generated by MATLAB.

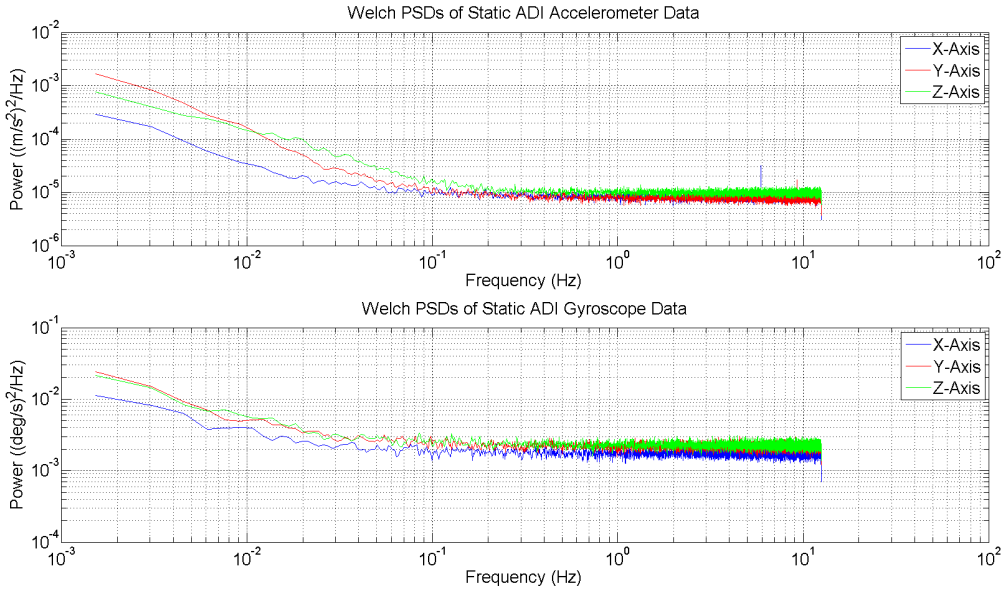

Figure 4.8: The Welch PSDs of static data from the ADI IMU accelerometers and gyroscopes.

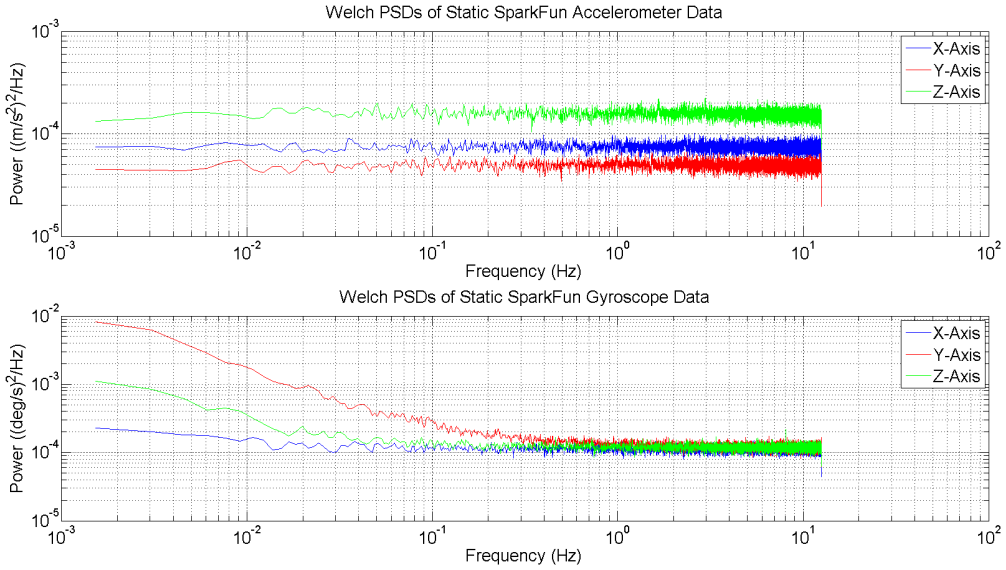

Figure 4.9: The Welch PSDs of static data from the SparkFun IMU accelerometers and gyroscopes.

overpower the effects of the drifting biases, and the PSDs fit very closely with the flat model. As was the case with the Allan Deviation functions, the lower frequency portions of the PSDs will be ignored for now and the primary focus will be on the portions of the PSDs above the 1Hz frequency.

ADI IMU	PSD Slope
X-Axis Accelerometer	$8.3417 * 10^{-6}$ $(m/s^2)^2/Hz^2$
Y-Axis Accelerometer	$8.5088 * 10^{-6}$ $(m/s^2)^2/Hz^2$
Z-Axis Accelerometer	$1.0463 * 10^{-5}$ $(m/s^2)^2/Hz^2$
X-Axis Gyroscope	$1.7595 * 10^{-3}$ $(°/s)^2/Hz^2$
Y-Axis Gyroscope	$2.1995 * 10^{-3}$ $(°/s)^2/Hz^2$
Z-Axis Gyroscope	$2.2977 * 10^{-3}$ $(°/s)^2/Hz^2$
SparkFun IMU	**PSD Slope**
X-Axis Accelerometer	$7.4451 * 10^{-5}$ $(m/s^2)^2/Hz^2$
Y-Axis Accelerometer	$4.9229 * 10^{-5}$ $(m/s^2)^2/Hz^2$
Z-Axis Accelerometer	$1.5559 * 10^{-4}$ $(m/s^2)^2/Hz^2$
X-Axis Gyroscope	$1.1222 * 10^{-4}$ $(°/s)^2/Hz^2$
Y-Axis Gyroscope	$1.3046 * 10^{-4}$ $(°/s)^2/Hz^2$
Z-Axis Gyroscope	$1.2000 * 10^{-4}$ $(°/s)^2/Hz^2$

Table 4.2: Measurement noise PSD slope values for the ADI and SparkFun IMUs.

To verify that the PSDs of the measurement noise do truly match up with the PSD of white noise, the value of their slopes is calculated. The calculated slopes of the measurement noise PSDs are listed in Table 4.2.

As Table 4.2 shows, all of the slopes of the measurement noise PSDs are very close to zero, which indicates that the measurement noise PSDs are all essentially flat. This further confirms that the IMU measurement noise can be accurately modeled using the white noise model.

Based on the results from both the Allan Variance and PSD analyses, the white noise model is clearly the model that best fits the IMU measurement noise. Now that the color of the measurement noise has been determined, all that remains is to determine the PDF of the measurement noise to complete the measurement noise model.

Section 4.1.3 – Probability Density Function (PDF) Analysis

The third analysis performed on the static IMU data is called a Probability Density Function (PDF) analysis. The process of determining the PDF of a noise process is relatively simple. The noise samples are arranged in histograms that display the number of times a specific value appears in the noise process. Curves can then be fitted to the general shape of these histograms to see which family of PDFs most accurately models the noise distribution. Figure 4.10 shows the histogram of a sample of WGN with a Gaussian distribution superimposed on top of it.

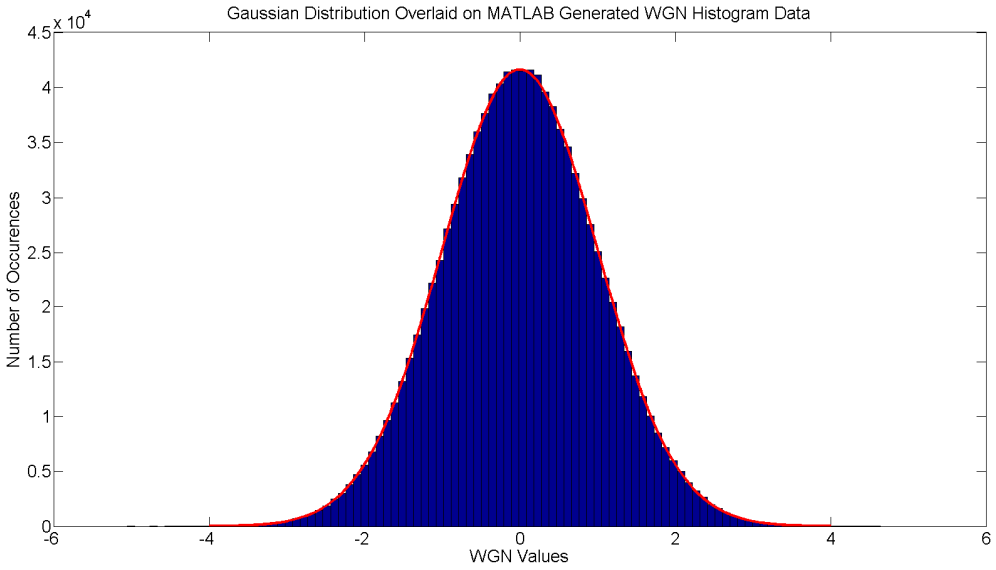

Figure 4.10: A histogram of a sample of WGN generated by MATLAB with a Gaussian distribution fitted to it.

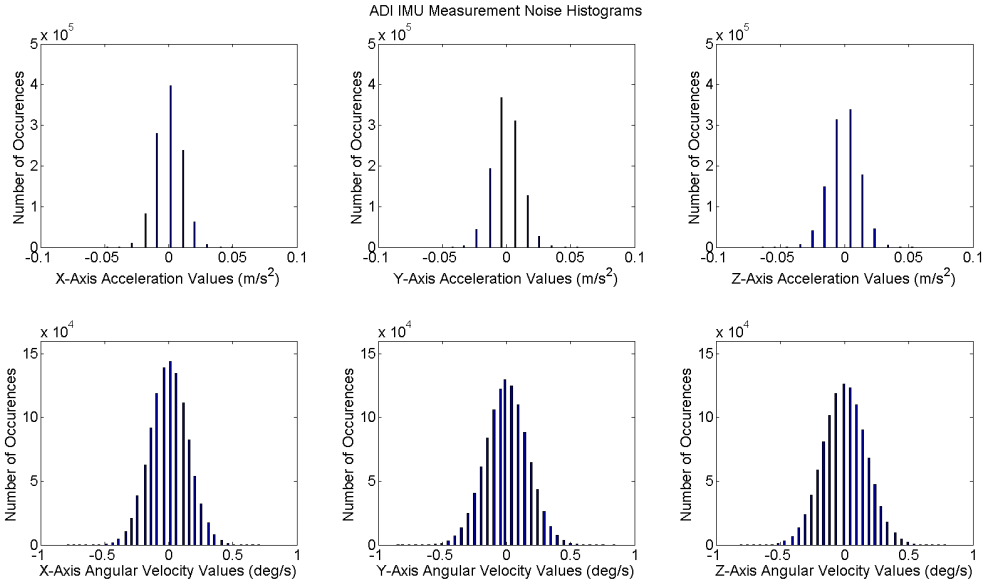

Figure 4.11: A histogram of the measurement noise processes from the ADI IMU.

This is the method that will be used to estimate the PDFs of the IMU measurement noise. The histograms of the IMU measurement noise can be seen in Figures 4.11 and 4.12. The gaps in the histograms occur because the continuous accelerations and angular velocities being measured by the IMU are quantized and converted into a digital format by an analog to digital converter (ADC) so that they can

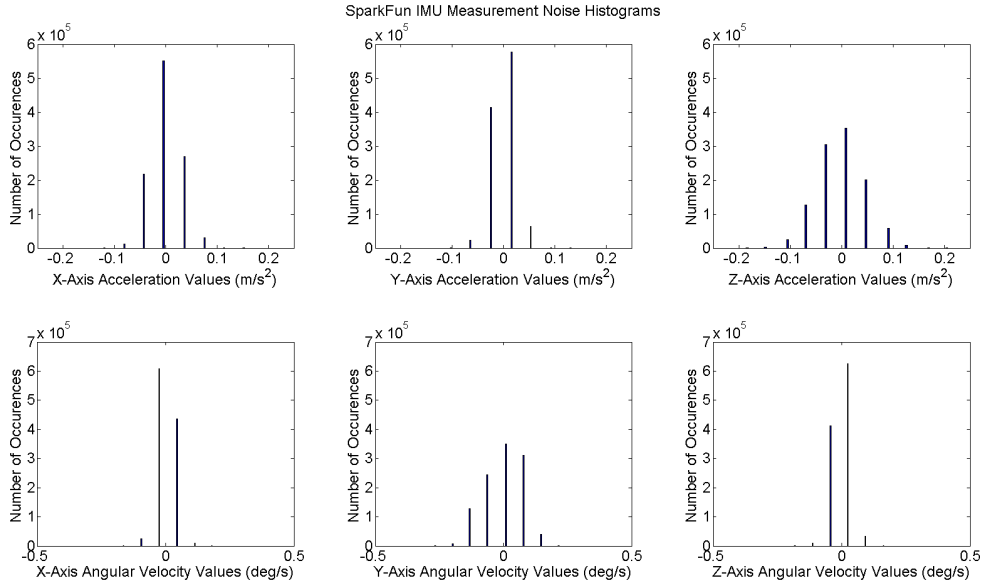

Figure 4.12: A histogram of the measurement noise processes from the SparkFun IMU.

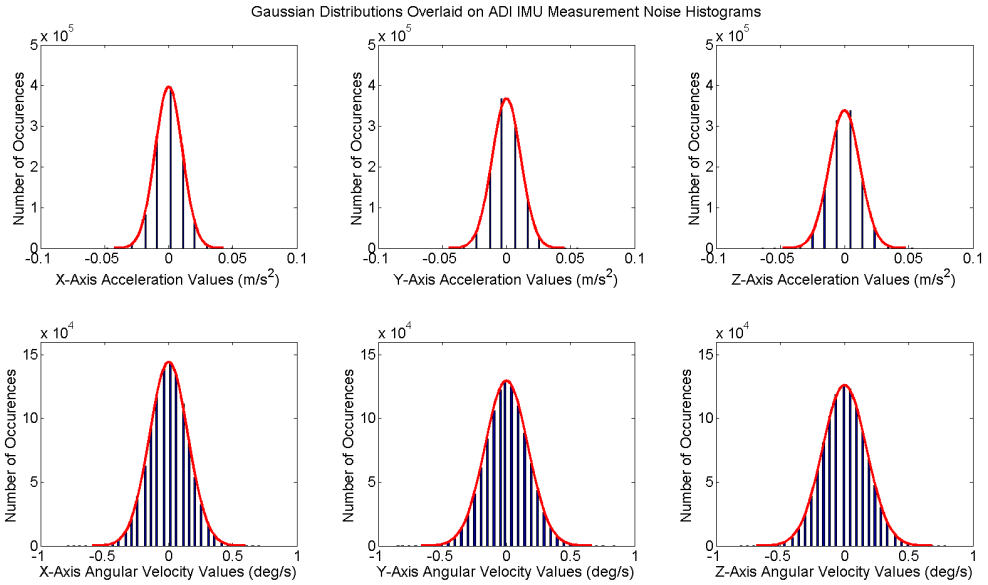

Figure 4.13: The ADI IMU measurement noise histograms with Gaussian distributions fit to them.

be transmitted and processed digitally. The general shape of the histograms seems to suggest that Gaussian PDFs would accurately model the measurement noise distributions. Because Gaussian distributions can be completely defined using only a mean and a variance and each of the measurement noise processes is zero-mean by definition, the sample variance of each measurement noise process can

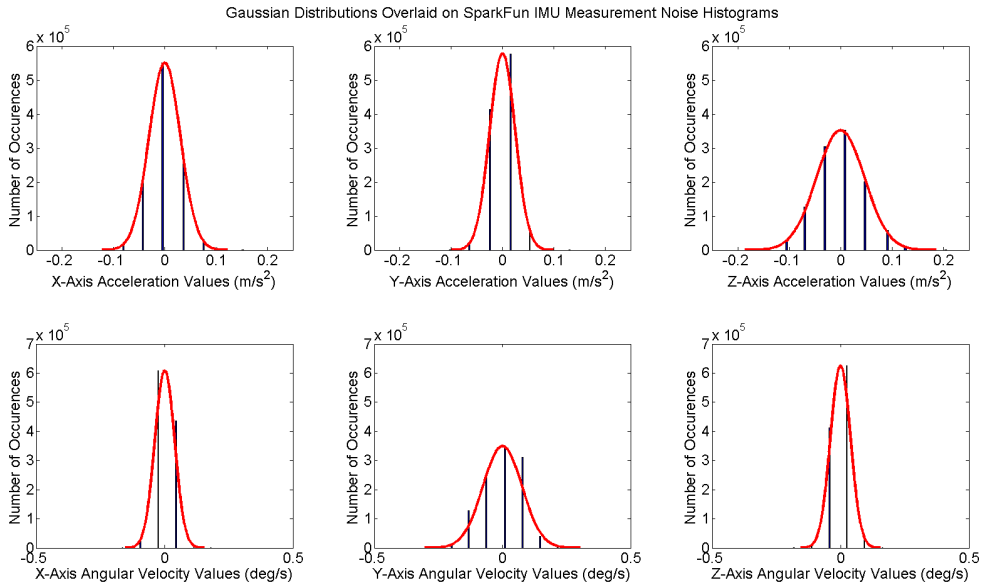

Figure 4.14: The SparkFun IMU measurement noise histograms with Gaussian distributions fit to them.

ADI IMU	Sample Variance	SparkFun IMU	Sample Variance
X-Axis Accelerometer	$1.1267 * 10^{-4}$ $(m/s^2)^2$	X-Axis Accelerometer	$9.3624 * 10^{-4}$ $(m/s^2)^2$
Y-Axis Accelerometer	$1.2580 * 10^{-4}$ $(m/s^2)^2$	Y-Axis Accelerometer	$6.1601 * 10^{-4}$ $(m/s^2)^2$
Z-Axis Accelerometer	$1.4302 * 10^{-4}$ $(m/s^2)^2$	Z-Axis Accelerometer	$2.1651 * 10^{-3}$ $(m/s^2)^2$
X-Axis Gyroscope	$2.2280 * 10^{-2}$ $(°/s)^2$	X-Axis Gyroscope	$1.4673 * 10^{-3}$ $(°/s)^2$
Y-Axis Gyroscope	$2.7662 * 10^{-2}$ $(°/s)^2$	Y-Axis Gyroscope	$5.7326 * 10^{-3}$ $(°/s)^2$
Z-Axis Gyroscope	$2.9154 * 10^{-2}$ $(°/s)^2$	Z-Axis Gyroscope	$1.5098 * 10^{-3}$ $(°/s)^2$

Table 4.3: The sample variances of the IMU measurement noise processes.

be used to create a family of Gaussian distributions that should accurately model each of the measurement noises processes.

The sample variances used to compute the Gaussian distributions can be seen in Table 4.3. Figures 4.13 and 4.14 show the Gaussian distributions superimposed on top of the measurement noise histograms to provide a visual representation of how accurately they model the measurement noise distributions. These histograms are only a visual confirmation of the measurement noise PDFs. It could be proven that the PDFs are Gaussian through hypothesis testing, but visual confirmation is considered to be enough for this analysis because the Gaussian distributions very closely match the shape of the measurement noise histograms. Therefore, the PDFs of the IMU measurement noises will be modeled using Gaussian PDFs parameterized by the sample variances of the measurement noises.

Combining the results obtained from the Allan Variance, PSD and PDF analyses, it is clear that the IMU measurement noise can be most accurately modeled using the WGN model. This is the model that will be used to represent the IMU measurement noise in the EKF component of the inertial navigation algorithm.

Section 4.1.4 – Drifting Bias Analysis

The fourth and final analysis performed on the static IMU data is called the Drifting Bias analysis. Recall from Section 3.2.2 that the behavior of the drifting bias is modeled using a first-order Markov process (see Equation 3.10 from Section 3.2.3), which falls under the family of random walk models. Therefore, the drifting biases should produce Allan Deviation functions with slopes of +½. This was discussed briefly back in Section 4.1.1, where it was shown that at the tail end of the Allan Deviation functions the slopes started to swing from -½ to +½. This reflects the fact that as the length of the time average parameter τ gets larger and larger, the effects of the drifting biases start to overpower the effects of the measurement noise, which is why the effects of the drifting biases show up more predominantly at the tail ends of the Allan Deviation functions.

The process of estimating the parameters for the first-order Markov processes that characterize the drifting biases begins by first isolating the drifting biases from a sample of static IMU data. As mentioned previously, the sample of static IMU data should be at least twelve hours in length to ensure that the drifting biases have an adequate amount of time to diverge from their initial values. The drifting biases can be isolated from a sample of static IMU data by first subtracting the sample means of the data samples from each individual sensor axis to remove the effects of the static biases and then performing a moving average on the data to remove the effects of the measurement noise. The remaining portions of the IMU data represent the drifting biases.

The parameters of the first-order Markov process can be estimated by exploiting a couple of mathematical properties of the first-order Markov process model. There are two parameters that must be estimated for each drifting bias: a scale factor φ and a noise variance σ_{bd}^2 that is the variance of the WGN process that drives the first-order Markov process. The parameter φ can be found by analyzing the autocorrelation function (ACF) of the drifting bias. The value of the ACF of a first-order Markov process at a lag value (τ) equal to one represents the amount of correlation that exists between any two data points in the first-order Markov process that are separated by one timestep. Referring back to the discussion of the drifting biases in Section 3.2.3, the parameter φ was also defined as the amount of correlation that exists between any two data points in a drifting bias that are separated by one timestep. Therefore, since

ADI IMU	$\boldsymbol{\varphi}$	$\boldsymbol{\sigma_{bd}^2}$
X-Axis Accelerometer	0.9887	$1.9239 * 10^{-7}$ $(m/s^2)^2$
Y-Axis Accelerometer	0.9912	$3.5379 * 10^{-7}$ $(m/s^2)^2$
Z-Axis Accelerometer	0.9907	$2.4626 * 10^{-7}$ $(m/s^2)^2$
X-Axis Gyroscope	0.9923	$4.3040 * 10^{-6}$ $(°/s)^2$
Y-Axis Gyroscope	0.9789	$8.2511 * 10^{-6}$ $(°/s)^2$
Z-Axis Gyroscope	0.9908	$8.1647 * 10^{-6}$ $(°/s)^2$
SparkFun IMU	$\boldsymbol{\varphi}$	$\boldsymbol{\sigma_{bd}^2}$
X-Axis Accelerometer	0.9907	$1.0827 * 10^{-7}$ $(m/s^2)^2$
Y-Axis Accelerometer	0.9905	$3.8738 * 10^{-9}$ $(m/s^2)^2$
Z-Axis Accelerometer	0.9907	$3.9915 * 10^{-6}$ $(m/s^2)^2$
X-Axis Gyroscope	0.9907	$1.1606 * 10^{-6}$ $(°/s)^2$
Y-Axis Gyroscope	0.9809	$1.5477 * 10^{-4}$ $(°/s)^2$
Z-Axis Gyroscope	0.9906	$2.1167 * 10^{-7}$ $(°/s)^2$

Table 4.4: The estimated values of the drifting bias first-order Markov process parameters.

these two values represent the same statistical quantity, the value of φ for each drifting bias can be estimated by simply calculating the ACF of the drifting bias data and using the value of the ACF at a lag value of one [14].

Once the φ parameter has been estimated, there is another mathematical property of the first-order Markov process that can be exploited to produce an estimate of the σ_{bd}^2 parameter. The variance of a first-order Markov process is defined as

$$var(b_d) = \frac{\sigma_{bd}^2}{1 - \varphi^2} \qquad \text{Eq. 4.3}$$

where $var(b_d)$ is the variance of the first-order Markov process (in this case the drifting bias). Therefore, an estimate of the noise variance σ_{bd}^2 can be produced by taking the variance of the drifting bias data and multiplying it by the quantity $(1 - \varphi^2)$. Now that both the φ and σ_{bd}^2 parameters have been estimated, the first-order Markov process that represents the behavior of the drifting bias is complete. Table 4.4 shows the values that were obtained for all of the drifting bias first-order Markov process parameters [14].

The results displayed in Table 4.4 show that on average the drifting biases in the gyroscopes tend to drift more noticeably than drifting biases in the accelerometers do. The results also show that the values of φ for both IMUs are very close to one and the noise variances are all very small. These values suggest that the amount of bias drift is fairly minimal, and it shouldn't have a large impact on the overall integrity of the raw inertial data.

Section 4.2 – Power Cycle Experiment

The goal of the Power Cycle Experiment is to determine the amount of variance in the values of the accelerometer and gyroscope static biases. This experiment is similar to the 12-Hour Static IMU Data Experiment, except the length of the data sample will be much shorter (approximately five minutes) and a group of static data samples will be taken in between power cycles of the IMU in order to determine the turn-on to turn-on bias variance. A power cycle is defined as the act of turning the IMU off and then back on again at a later time. As was the case with the 12-Hour Static IMU Data Experiment, the only errors that affect the data during data collection are measurement noise, drifting biases, static biases and misalignment errors. In order to isolate the effects of the static biases, each data sample is limited to a length of five minutes, over which time the effects of the drifting biases are considered to be negligible. Each data sample is then averaged together to produce a single set of average values for the data sample. The measurement noise in the data cannot be perfectly removed, but its effects are dramatically reduced to the point of being negligible (i.e. variances on the order of 10^{-7} to 10^{-8} $(m/s^2)^2$ for the accelerometers and 10^{-6} to 10^{-7} $(°/s)^2$ for the gyroscopes) once the data sample is averaged together because the measurement noise is zero-mean. The effects of the misalignment errors are also considered to be negligible for the purposes of this experiment. Once the static biases have been isolated, their behavior in

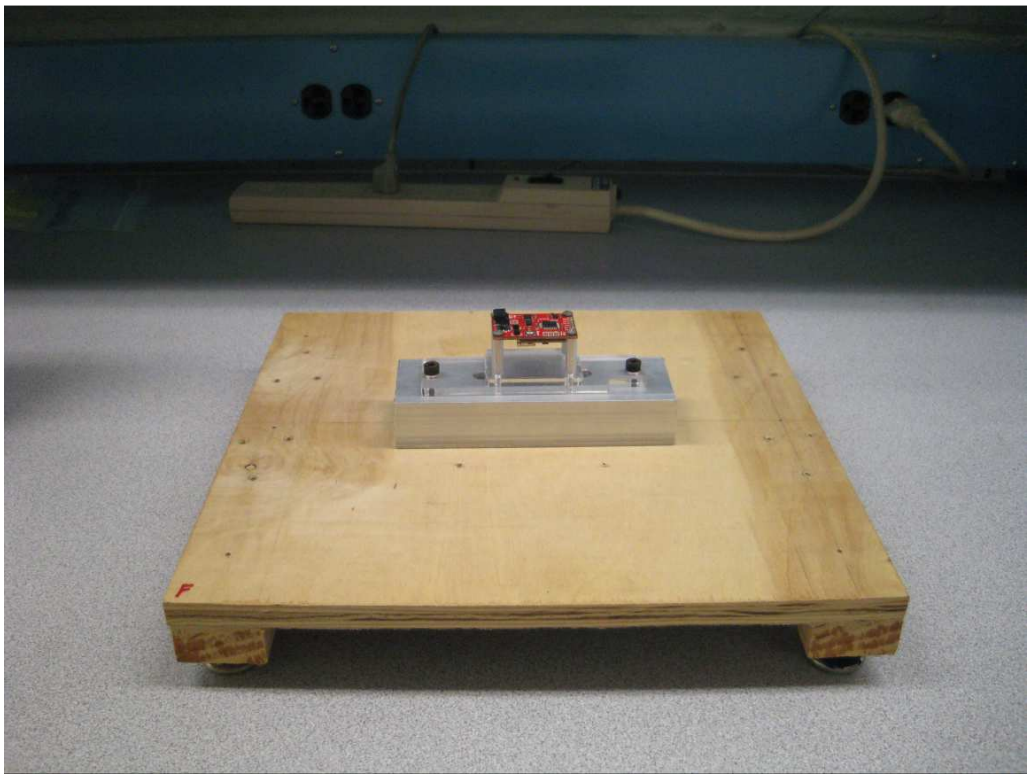

Figure 4.15: The test fixture for the Power Cycle Experiment.

Figure 4.16: The test fixture for the Power Cycle Experiment enclosed in a hybridization incubator.

between power cycles of the IMU can be properly analyzed.

The Power Cycle Experiment is conducted by following the series of steps in the bulleted list below:

- The IMU is securely mounted on a test fixture that can be adjusted to ensure that the IMU is completely level (See Figure 4.15). The test fixture is then enclosed in a hybridization incubator to minimize fluctuations in the internal temperature of the IMU during the course of the test (See Figure 4.16).
- The IMU is then powered on and allowed to come up to temperature (the initial powering on of the IMU causes its internal temperature to rise approximately 1-2°C).
- Once the IMU has reached a stable temperature, a five minute data sample is taken. It is important that the IMU remains completely motionless during this time.
- After the data sample is taken, the IMU is unplugged and powered off for five minutes. After five minutes, the IMU is plugged back in and powered on. The IMU is allowed to come up to temperature again and once the internal temperature of the IMU has stabilized, another five minute data sample is taken. Then the IMU is unplugged and powered off for another five

ADI IMU	Turn-On to Turn-On Bias Variation
X-Axis Accelerometer	$6.7203 * 10^{-6} (m/s^2)^2$
Y-Axis Accelerometer	$8.7258 * 10^{-6} (m/s^2)^2$
Z-Axis Accelerometer	$4.2737 * 10^{-6} (m/s^2)^2$
X-Axis Gyroscope	$7.2805 * 10^{-5} (°/s)^2$
Y-Axis Gyroscope	$1.9517 * 10^{-4} (°/s)^2$
Z-Axis Gyroscope	$4.4230 * 10^{-4} (°/s)^2$
SparkFun IMU	**Turn-On to Turn-On Bias Variation**
X-Axis Accelerometer	$1.1761 * 10^{-6} (m/s^2)^2$
Y-Axis Accelerometer	$1.5348 * 10^{-6} (m/s^2)^2$
Z-Axis Accelerometer	$4.3348 * 10^{-5} (m/s^2)^2$
X-Axis Gyroscope	$2.1586 * 10^{-4} (°/s)^2$
Y-Axis Gyroscope	$2.4880 * 10^{-3} (°/s)^2$
Z-Axis Gyroscope	$5.1009 * 10^{-5} (°/s)^2$

Table 4.5: The calculated values of the turn-on to turn-on bias variation for the ADI and SparkFun IMUs.

minutes before the next data sample is taken. This process repeats until twenty distinct data samples have been collected.

- Once the data has been recorded, it is filtered to remove outliers and other anomalies (see Section 4.1 for a more detailed description of the filtering process). Then the inertial data in each individual data sample is averaged together to dramatically reduce the effects of the measurement noise. The resulting averaged data sets represent only the static biases of each of the individual sensor axes. The twenty data points for each individual sensor axis are then grouped together and the sample variances of those twenty data points are calculated to determine the turn-on to turn-on bias variation for each of the individual sensor axes.

After performing the Power Cycle Experiment in the manner described above, specific values for the turn-on to turn-on bias variations for both the ADI and SparkFun IMUs were calculated. Table 4.5 shows the results.

Section 4.3 – Multi-Angle Experiment

The goal of the Multi-Angle Experiment is to determine the values of the static biases, scale factors and misalignment errors for an IMU's accelerometer as well as the extent to which those values are affected by temperature. This experiment is based on the fact that gravity is a known source of constant acceleration. By fixing the IMU in a known orientation it is possible to calculate the amount of acceleration due to gravity that each IMU axis should experience. These ideal accelerations can then be

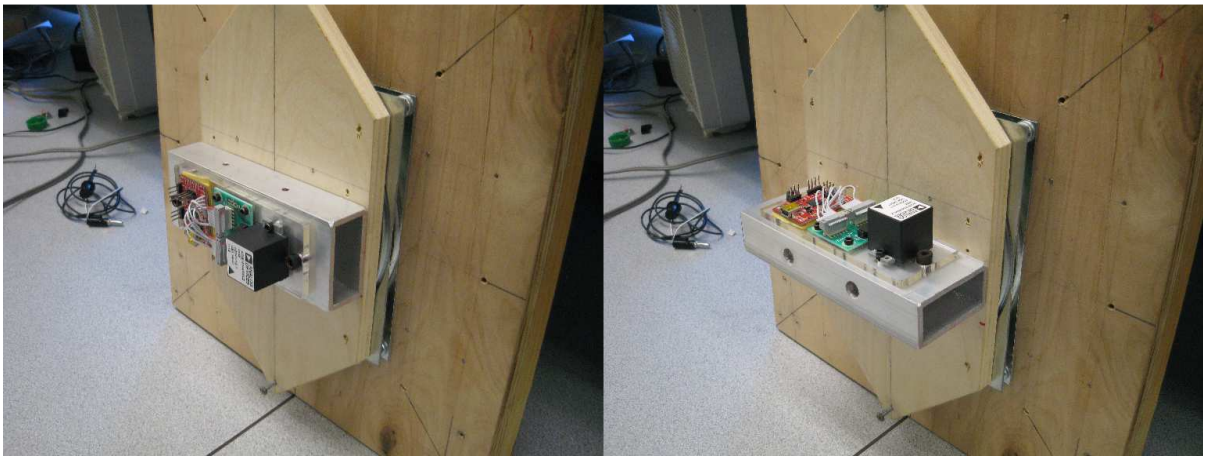

Figure 4.17: The Multi-Angle Experiment test fixture with the IMU mounted to it in both configurations.

Figure 4.18: The Multi-Angle Experiment test fixture enclosed in the hybridization incubator. The PVC pipes are used to ensure that the test fixture remains stationary and level during the course of the experiment.

compared to the measured accelerations produced by the IMU and based on that comparison the values of the static biases, scale factors and misalignment errors for the accelerometer can be estimated.

Due to the temperature sensitive nature of the static biases and scale factors, this experiment must be conducted over a range of different temperatures. A look-up table can then be created using the results from each temperature. Since the ADI IMU is already temperature calibrated by the manufacturer, only the SparkFun IMU is tested over multiple temperatures. The static biases, scale factors and misalignment errors for the ADI IMU accelerometer will be represented by a single set of parameters.

The Multi-Angle Experiment is conducted by following the series of steps in the bulleted list below:

- Before conducting the experiment, an appropriate test fixture must first be created. An image of the test fixture can be seen in Figure 4.17. The test fixture consists of a fixed wooden frame with a rotating arm mounted to it such that the axis of rotation is perpendicular to the gravity vector. The IMU is rigidly mounted to the rotating arm such that one of its axes is aligned with the test fixture's axis of rotation and the other two axes are exposed to gravity in varying degrees based on the rotating arm's orientation. The test fixture has several pre-drilled holes in it that mark increments of 45° all the way around the test fixture. There is a screw embedded in the rotating arm that can be screwed into these holes, which allows the rotating arm to be fixed at increments of 45° all the way around the test fixture.
- In order to determine how sensitive the static biases and scale factors are to temperature, data will be collected from an array of different temperatures (Room Temperature, 35°C, 40°C, 45°C, 50°C, 55°C and 60°C). To maintain these target temperatures during the course of the experiment, the test fixture is enclosed in a hybridization incubator (as shown in Figure 4.18). A digital level is used to make sure that the fixture is level inside the incubator and that its axis of rotation is as perpendicular to gravity as possible.
- Once the test fixture has been placed inside the incubator, the orientation of the IMU with respect to gravity is measured using the digital level and recorded.
- The incubator is then allowed to saturate at the first target temperature. Once the incubator has saturated, the IMU is powered on and its internal temperature is allowed to saturate at the target temperature.
- After the IMU has come up to temperature, a five minute static data sample is taken. After the data sample has been collected, the IMU is unplugged and the incubator is set for the next target temperature. Once the incubator saturates at the new target temperature, the IMU is powered on and allowed to saturate at the new target temperature. Once the IMU has come up to temperature, another five minute static data sample is taken. Then the IMU is unplugged, the incubator is set for the next target temperature and the process repeats until all of the target temperatures have been covered.
- After covering all of the target temperatures, the test fixture is rotated to the next 45° increment. The orientation of the IMU is measured again using the digital level and recorded. The data logging procedure is then repeated at each of the target temperatures with the IMU in its new orientation.
- This process is repeated until all eight orientations on the test fixture have been tested. Then the IMU is removed from the test fixture and remounted such that the previously untested IMU axis (the axis in-line with the rotation of the test fixture) will be exposed to gravity. The eight

orientations of the test fixture are tested again in the same manner as before, resulting in a total of sixteen orientations.

- After the experiment is over, each of the five minute data samples is filtered to remove outliers and other anomalies (see Section 4.1 for a more detailed description of the filtering process). Then the average of each data sample is calculated to produce the average X, Y and Z acceleration for each data sample. Because the data samples are only five minutes long, the effects of the drifting biases are considered to be negligible. However, since the IMU is power cycled in between data samples, turn-on to turn-on bias variation must be accounted for in the data analysis. The measurement noise in the data cannot be perfectly removed, but its effects are dramatically reduced to the point of being negligible (i.e. variances on the order of 10^{-7} to 10^{-8} $(m/s^2)^2$ for the accelerometers and 10^{-6} to 10^{-7} $(°/s)^2$ for the gyroscopes) once the data sample is averaged together because the measurement noise is zero-mean.
- The final result is that for each target temperature there should be an average X, Y and Z acceleration for each of the sixteen distinct IMU orientations. This data can then be processed using a least squares estimation algorithm to determine the static biases, scale factors and misalignment errors for the accelerometer at each of the target temperatures. A lookup table can then be created by interpolating between the target temperatures to determine the static biases, scale factors and misalignment errors for any given input temperature.

Section 4.3.1 – Linear Least Squares Estimation Analysis

For each target temperature, the Multi-Angle Experiment produces an average measured X, Y and Z acceleration for each of the sixteen IMU orientations. Also, because the actual orientation of the IMU at each orientation set point was recorded using a digital level, the actual accelerations that were experienced by the IMU at each of the sixteen orientations can be calculated as well. These values represent the true accelerations that would have ideally been measured by the accelerometer if it could produce perfect measurements. To estimate the values of the static biases, scale factors and misalignment errors for the accelerometer, the measured accelerations will be compared to the true accelerations through a process called linear least squares estimation (LLSE). LLSE is an estimation technique that applies to linear systems with Gaussian distributed noise in the form of

$$y = Hx + v \qquad \text{Eq. 4.4}$$

where x is some unknown state vector to be estimated, H is a known observation matrix that performs an operation on the x parameters, y is a vector of measurements made of the x parameters after being operated on by H, and v is a zero-mean WGN process with a covariance matrix R that obscures the measurement data y. LLSE provides a method for optimally estimating (in terms of minimum error variance) the value of x based on the imperfect measurements y. This optimal estimate can be produced by using the equation

$$\hat{x} = (H^T R^{-1} H)^{-1} H^T R^{-1} y \qquad \text{Eq. 4.5}$$

where $\hat{x}$ is the optimal estimate of the state vector x [15].

These equations can be interpreted to fit the data from the Multi-Angle Experiment and produce an estimate of the static biases, scale factors and misalignment errors for the accelerometer. The relationship between the measured accelerations and true accelerations, incorporating the effects of the static biases, scale factors and misalignment errors, is

$$a_m = C_a a_t + b_a + w_a \qquad \text{Eq. 4.6}$$

where a_m is the measured acceleration, a_t is the true acceleration, and w_a is the IMU measurement noise. This equation can be rearranged to fit the form of Equation 4.4. The averaged X, Y and Z acceleration measurements are used to populate the y vector, the true X, Y and Z acceleration values are used to populate the H matrix, the static bias, scale factor and misalignment error parameters to be estimated are used to populate the x vector and the IMU measurement noise is used to populate the v vector. The new linear vector equation is

$$\begin{bmatrix} a_{mx} \\ a_{my} \\ a_{mz} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & a_{tx} & a_{ty} & a_{tz} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & a_{tx} & a_{ty} & a_{tz} & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & a_{tx} & a_{ty} & a_{tz} \end{bmatrix} \begin{bmatrix} b_a^x \\ b_a^y \\ b_a^z \\ c_a^{xx} \\ c_a^{xy} \\ c_a^{xz} \\ c_a^{yx} \\ c_a^{yy} \\ c_a^{yz} \\ c_a^{zx} \\ c_a^{zy} \\ c_a^{zz} \end{bmatrix} + \begin{bmatrix} w_a^x \\ w_a^y \\ w_a^z \end{bmatrix} \qquad \text{Eq. 4.7}$$

$$y_1 = H_1 x + v_1 \quad \text{Eq. 4.8}$$

This equation is mathematically the same as Equation 4.6, but it is in a form that is more conducive to LLSE. Equation 4.8, however, is only enough to represent data from a single orientation. In order to use the data from all sixteen orientations in the LLSE algorithm, sixteen of these matrix equations are created, each one using the data from one of the individual IMU orientations. The individual equations are then stacked together to produce one combined matrix equation as shown in Equation 4.9.

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{16} \end{bmatrix} = \begin{bmatrix} H_1 \\ H_2 \\ \vdots \\ H_{16} \end{bmatrix} x + \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_{16} \end{bmatrix} \rightarrow y = Hx + v \quad \text{Eq. 4.9}$$

The last matrix that needs to be populated is the covariance matrix R. The R matrix is populated

ADI IMU	$\boldsymbol{b_a}$	$\boldsymbol{C_a}$
Room Temperature	$\begin{bmatrix} -0.0259 \\ -0.0763 \\ 0.0148 \end{bmatrix}$ (m/s^2)	$\begin{bmatrix} 1.0002 & -0.0085 & -0.0008 \\ 0.0100 & 1.0003 & -0.0048 \\ -0.0005 & -0.0005 & 1.0001 \end{bmatrix}$
SparkFun IMU	$\boldsymbol{b_a}$	$\boldsymbol{C_a}$
Room Temperature	$\begin{bmatrix} 0.2663 \\ -0.1030 \\ -0.6025 \end{bmatrix}$ (m/s^2)	$\begin{bmatrix} 1.0529 & -0.0160 & 0.0040 \\ 0.0124 & 1.0630 & 0.0093 \\ 0.0039 & -0.0051 & 1.0167 \end{bmatrix}$
35°C	$\begin{bmatrix} 0.2211 \\ -0.1137 \\ -0.5437 \end{bmatrix}$ (m/s^2)	$\begin{bmatrix} 1.0521 & -0.0175 & 0.0030 \\ 0.0139 & 1.0665 & 0.0100 \\ 0.0047 & -0.0054 & 1.0186 \end{bmatrix}$
40°C	$\begin{bmatrix} 0.2213 \\ -0.1248 \\ -0.5848 \end{bmatrix}$ (m/s^2)	$\begin{bmatrix} 1.0528 & -0.0176 & 0.0031 \\ 0.0141 & 1.0673 & 0.0102 \\ 0.0048 & -0.0062 & 1.0174 \end{bmatrix}$
45°C	$\begin{bmatrix} 0.2258 \\ -0.1419 \\ 0.6261 \end{bmatrix}$ (m/s^2)	$\begin{bmatrix} 1.0539 & -0.0176 & 0.0034 \\ 0.0139 & 1.0684 & 0.0109 \\ 0.0034 & -0.0065 & 1.0172 \end{bmatrix}$
50°C	$\begin{bmatrix} 0.2316 \\ -0.1561 \\ -0.6583 \end{bmatrix}$ (m/s^2)	$\begin{bmatrix} 1.0546 & -0.0177 & 0.0034 \\ 0.0140 & 1.0691 & 0.0112 \\ 0.0038 & -0.0076 & 1.0154 \end{bmatrix}$
55°C	$\begin{bmatrix} 0.2383 \\ -0.1723 \\ -0.6905 \end{bmatrix}$ (m/s^2)	$\begin{bmatrix} 1.0556 & -0.0178 & 0.0036 \\ 0.0137 & 1.0700 & 0.0121 \\ 0.0027 & -0.0077 & 1.0154 \end{bmatrix}$
60°C	$\begin{bmatrix} 0.2427 \\ -0.1887 \\ -0.7264 \end{bmatrix}$ (m/s^2)	$\begin{bmatrix} 1.0564 & -0.0178 & 0.0041 \\ 0.0138 & 1.0706 & 0.0123 \\ 0.0027 & -0.0084 & 1.0135 \end{bmatrix}$

Table 4.6: The estimated values of the static biases, scale factors and misalignment errors for the ADI and SparkFun IMU accelerometers.

by repeating the three accelerometer turn-on to turn-on bias variances (discussed in Section 4.2) along the diagonal of the matrix as shown in Equations 4.10 and 4.11. The *R* matrix is populated with the turn-on to turn-on bias variances because the IMU is powered cycled in between individual data samples. The turn-on to turn-on bias variances account for the slight variations in the accelerometer static biases that occur over multiple independent data samples as a result of this power cycling.

$$R' = \begin{bmatrix} \sigma_{ax}^2 & 0 & 0 \\ 0 & \sigma_{ay}^2 & 0 \\ 0 & 0 & \sigma_{az}^2 \end{bmatrix} \qquad R = \begin{bmatrix} R' & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & R' \end{bmatrix}_{48x48} \qquad \text{Eq. 4.10 \& 4.11}$$

Now that all of the required matrices are known, they can be plugged into Equation 4.5 to produce estimates of the static biases, scale factors and misalignment errors for the accelerometer.

After performing the Multi-Angle Experiment in the manner described above, specific values for the static biases, scale factors and misalignment errors for both the ADI and SparkFun accelerometers were estimated for all of the target temperatures. Table 4.6 shows the results.

Section 4.4 – Rate Table Experiment

The goal of the Rate Table Experiment is to determine the values of the static biases, scale factors and misalignment errors for an IMU's gyroscope as well as the extent to which those values are affected by temperature. This experiment is very similar in design to the Multi-Angle Experiment described in Section 4.3. In the Multi-Angle Experiment, gravity was used as a known source of constant acceleration to determine the static biases, scale factors and misalignment errors for an accelerometer. Unfortunately, there is no convenient naturally occurring source of constant angular velocity that can be exploited in this experiment to determine the static biases, scale factors and misalignment errors for a gyroscope. Therefore, a stepper motor is used to create a known and controllable angular velocity for the gyroscope to measure. By mounting the IMU to the stepper motor and rotating it at a known and constant angular velocity, it is possible to calculate the amount of angular velocity that each IMU axis should experience. These ideal angular velocities can then be compared to the measured angular velocities produced by the IMU and based on that comparison the values of the static biases, scale factors and misalignment errors for the gyroscope can be estimated.

As was the case with the Multi-Angle Experiment, the temperature sensitive nature of the static biases and scale factors means that the Rate Table Experiment must also be conducted over a range of different temperatures. A look-up table can then be created using the results from each temperature.

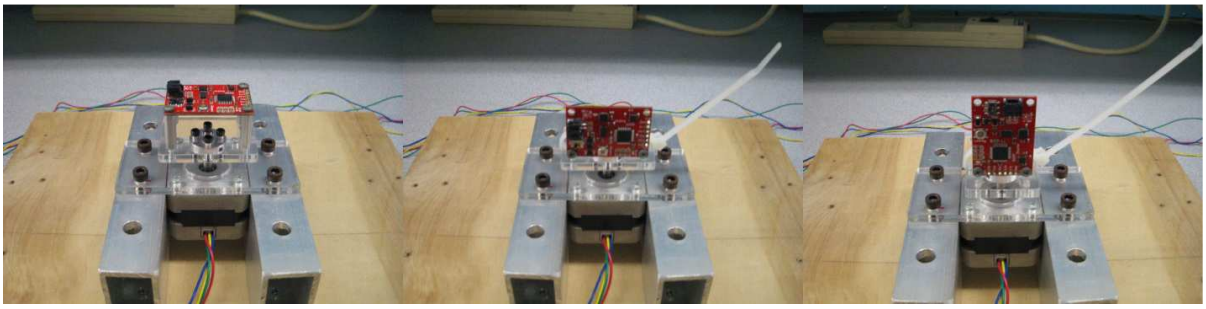

Figure 4.19: The Rate Table Experiment test fixture with the IMU mounted to it in all three configurations.

Again, since the ADI IMU is already temperature calibrated, only the SparkFun IMU is tested over multiple temperatures. The static biases, scale factors and misalignment errors for the ADI IMU gyroscope will be represented by a single set of parameters.

The Rate Table Experiment is conducted by following the series of steps in the bulleted list below:

- Before conducting the experiment, an appropriate test fixture must first be created. An image of the test fixture can be seen in Figure 4.19. The test fixture consists of a stepper motor with a rotating platform attached to the shaft via a shaft collar. The IMU is mounted to the rotating platform such that one of its axes is in line with the axis of rotation of the stepper motor. The stepper motor is controlled using a microcontroller, which sends digital signals to a stepper motor controller at a constant frequency. These signals determine when the motor should step and in what direction. The frequency of the signals coming from the microcontroller determines the angular velocity of the stepper motor. The stepper motor is mounted to a wooden base to ensure that it doesn't move during the duration of the testing. The wooden base has feet that can be adjusted to ensure that the stepper motor platform is level and the axis of rotation of the stepper motor is parallel to gravity. The test fixture is powered using an external 12V power source to ensure that the stepper motor has enough power to rotate at a constant velocity.
- Unlike many of the other experiments performed in this thesis where the IMU remains stationary for the duration of the experiment, in this experiment the IMU is constantly rotating. This means that data from the IMU cannot be logged onto a computer using a direct USB connection because the USB cable would very quickly become tangled around the test fixture and could potentially damage the test fixture, the IMU or itself. To overcome this problem the IMU was outfitted with a 3.3V external lithium polymer battery to supply it with power and a data logger that reads the inertial data coming out of the IMU over its serial port and records the data directly into a text file

that is stored on an on-board 2GB microSD card (see Figure 4.20). The microSD card can then be removed after the experiment has concluded and the data from the experiment can be transferred from the microSD card to a computer.

- In order to determine how sensitive the static biases and scale factors are to temperature, data will be collected from an array of temperatures (Room Temperature, 35°C, 40°C, 45°C and 50°C because the lithium polymer battery used in this experiment is only rated for temperatures up to 50°C). To maintain these target temperatures during the course of the experiment, the test fixture is enclosed in a hybridization incubator (as shown in Figure 4.21). A digital level is used to make sure that the test fixture is level inside the incubator and that the axis of rotation of the stepper motor is as parallel to gravity as possible. Since there is no direct USB connection between the IMU and a computer during this experiment, monitoring the internal temperature of the IMU is more difficult. To overcome this problem, a second identical IMU is placed inside the incubator to act as a stationary temperature reference. The stationary IMU is connected to a computer with a direct USB connection and its internal temperature is observed in real time during the course of the experiment. Because the two IMUs are identical, the unknown internal temperature of the rotating IMU can be inferred from the known internal temperature of the stationary IMU.
- The incubator is then allowed to saturate at the first target temperature. Once the incubator has

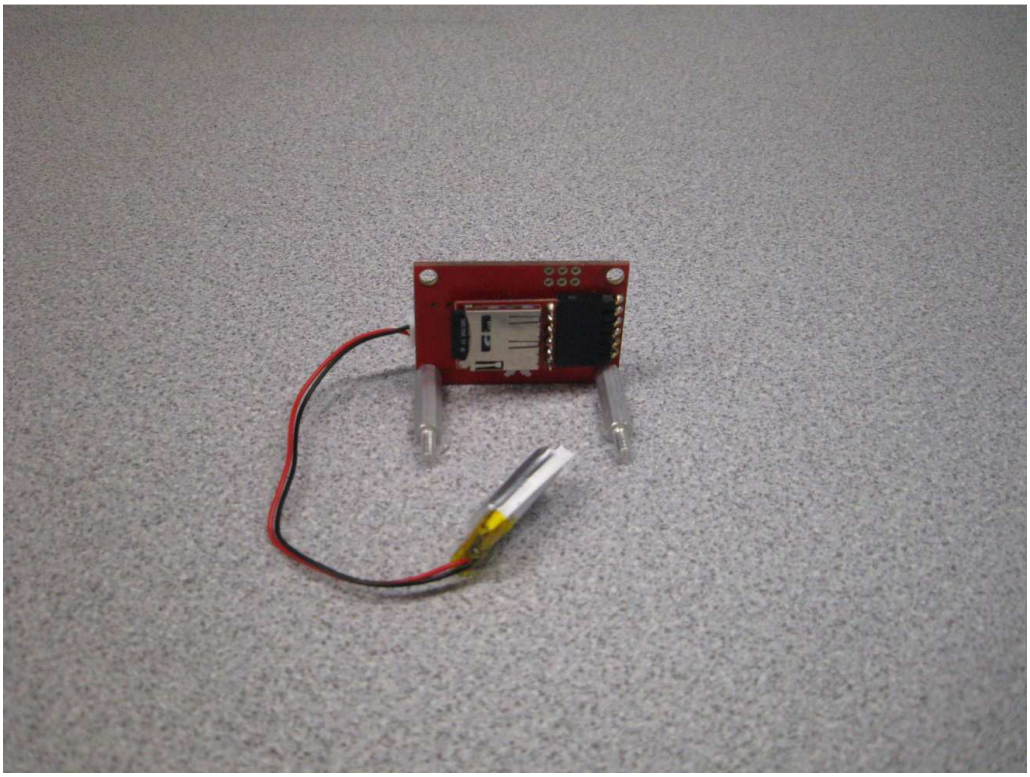

Figure 4.20: The IMU data logging peripherals.

Figure 4.21: The Rate Table Experiment test fixture enclosed in the hybridization incubator.

saturated, the IMU is powered on and its internal temperature is allowed to saturate at the target temperature.

- After the IMU has come up to temperature, the stepper motor circuitry is powered on and the IMU begins to rotate at a specific angular velocity. A five minute data sample is taken while the IMU is rotating. After the data sample has been collected the stepper motor circuitry is powered off, the IMU is unplugged and the incubator is set for the next target temperature. Once the incubator saturates at the new target temperature, the IMU is powered on and allowed to saturate at the new target temperature. Once the IMU has come up to temperature, the stepper motor circuitry is powered on again and another five minute data sample is taken while the IMU is rotating. Then the stepper motor circuitry is powered off, the IMU is unplugged, the incubator is set for the next target temperature and the process repeats until all of the target temperatures have been covered.
- After covering all of the target temperatures, the data logging procedure is repeated at each of the target temperatures again using a different angular velocity for the stepper motor. This process is repeated for five different angular velocities (0 degrees per second, ±90 degrees per second and ±180 degrees per second). Then the IMU is removed from the test fixture and remounted such

that one of the previously untested IMU axes (one of the axes perpendicular to the stepper motor's axis of rotation) will be exposed to the rotation of the stepper motor. The five angular velocities are tested again in the same manner as before for the two remaining IMU axes, resulting in a total of fifteen angular velocities.

- After the experiment is over, each of the five minute data samples is filtered to remove outliers and other anomalies (see Section 4.1 for a more detailed description of the filtering process). Then the average of each data sample is calculated to produce the average X, Y and Z angular velocity for each data sample. Because the data samples are only five minutes long, the effects of the drifting biases are considered to be negligible. However, since the IMU is power cycled in between data samples, turn-on to turn-on bias variation must be accounted for in the data analysis. The measurement noise in the data cannot be perfectly removed, but its effects are dramatically reduced to the point of being negligible (i.e. variances on the order of 10^{-7} to 10^{-8} $(m/s^2)^2$ for the accelerometers and 10^{-6} to 10^{-7} $(°/s)^2$ for the gyroscopes) once the data sample is averaged together because the measurement noise is zero-mean.
- The final result is that for each target temperature there should be an average X, Y and Z angular velocity for each of the fifteen distinct IMU configurations with the stepper motor. This data can be processed using the same least squares estimation algorithm that was used in the Multi-Angle Experiment to determine the static biases, scale factors and misalignment errors for the gyroscope at each of the target temperatures. A lookup table can then be created by interpolating between the target temperatures to determine the static biases, scale factors and misalignment errors for any given input temperature.

Section 4.4.1 – Linear Least Squares Estimation Analysis

For each target temperature, the Rate Table Experiment produces an average measured X, Y and Z angular velocity for each of the fifteen IMU configurations with the stepper motor. Also, the actual angular velocities that were experienced by the IMU in each of the fifteen configurations are known. These values represent the true angular velocities that would have ideally been measured by the gyroscope if it could produce perfect measurements. To estimate the values of the static biases, scale factors and misalignment errors for the gyroscope, the measured angular velocities will be compared to the true angular velocities using the same LLSE algorithm that was used to analyze the data from the Multi-Angle Experiment (see Section 4.2.1 for more information on how the LLSE algorithm works and how to convert the data into the proper matrices that are required by the algorithm). The parameters for Equations 4.6 and 4.7 remain the same with the exception that angular velocities are used in place of

ADI IMU	$\boldsymbol{b_g}$	$\boldsymbol{C_g}$
Room Temperature	$\begin{bmatrix} -0.0746 \\ 0.0325 \\ -0.2106 \end{bmatrix}$ (°/s)	$\begin{bmatrix} 1.0010 & -0.0007 & -0.0005 \\ 0.0017 & 1.0006 & -0.0117 \\ -0.0057 & -0.0218 & 1.0039 \end{bmatrix}$
SparkFun IMU	$\boldsymbol{b_g}$	$\boldsymbol{C_g}$
Room Temperature	$\begin{bmatrix} -0.9049 \\ 0.0824 \\ 1.3259 \end{bmatrix}$ (°/s)	$\begin{bmatrix} 0.9849 & -0.0059 & -0.0159 \\ 0.0057 & 0.9897 & 0.0043 \\ 0.0047 & 0.0152 & 1.0008 \end{bmatrix}$
35°C	$\begin{bmatrix} -1.0624 \\ 1.4117 \\ 1.3279 \end{bmatrix}$ (°/s)	$\begin{bmatrix} 0.9908 & 0.0130 & -0.0033 \\ 0.0051 & 0.9925 & 0.0014 \\ 0.0115 & 0.0102 & 0.9979 \end{bmatrix}$
40°C	$\begin{bmatrix} -1.1308 \\ 2.2828 \\ 1.3470 \end{bmatrix}$ (°/s)	$\begin{bmatrix} 0.9907 & 0.0129 & -0.0032 \\ 0.0051 & 0.9926 & 0.0012 \\ 0.0112 & 0.0102 & 1.0019 \end{bmatrix}$
45°C	$\begin{bmatrix} -1.2068 \\ 3.3385 \\ 1.3662 \end{bmatrix}$ (°/s)	$\begin{bmatrix} 0.9902 & 0.0125 & -0.0031 \\ 0.0045 & 0.9894 & 0.0010 \\ 0.0106 & 0.0104 & 1.0018 \end{bmatrix}$
50°C	$\begin{bmatrix} -1.2657 \\ 4.1985 \\ 1.3921 \end{bmatrix}$ (°/s)	$\begin{bmatrix} 0.9902 & 0.0125 & -0.0031 \\ 0.0046 & 0.9887 & 0.0009 \\ 0.0107 & 0.0103 & 0.9995 \end{bmatrix}$
55°C	$\begin{bmatrix} -1.3078 \\ 5.0648 \\ 1.4165 \end{bmatrix}$ (°/s)	*N/A
60°C	$\begin{bmatrix} -1.3621 \\ 6.0811 \\ 1.4481 \end{bmatrix}$ (°/s)	*N/A

*NOTE: Gyroscope scale factor matrices (C_g) could not be calculated for 55°C and 60°C because the lithium polymer battery used for the Rate Table Experiment was only rated for temperatures up to 50°C. The gyroscope bias matrices (b_g) for the 55°C and 60°C temperatures were calculated from the corresponding Multi-Angle Experiment data sets.

Table 4.7: The estimated values of the static biases, scale factors and misalignment errors for the ADI and SparkFun IMU gyroscopes.

accelerations. Once all of the appropriate matrices have been calculated and organized into the proper format (as dictated by Equations 4.8 and 4.9), they can be plugged into Equation 4.5 to produce an estimate of the static biases, scale factors and misalignment errors for the gyroscope.

After performing the Rate Table Experiment in the manner described above, specific values for the static biases, scale factors and misalignment errors for both the ADI and SparkFun gyroscopes were estimated for all of the target temperatures. Table 4.7 shows the results.

Section 5 – The Inertial Navigation Algorithm

Now that all of the IMU error sources have been properly analyzed and quantified, the next step is to develop the inertial navigation algorithm that will produce estimates of the linear velocity, position and orientation of a vehicle based on the raw IMU data and the secondary navigation information. The following subsections provide a basic overview of the entire inertial navigation algorithm as a whole, and more detailed descriptions of the individual components of the inertial navigation algorithm as well.

Section 5.1 – Coordinate Frames

The first step in understanding the inertial navigation algorithm is to understand the various coordinate frames that are used to describe how specific pieces of information are interpreted. The inertial navigation algorithm presented in this thesis uses four coordinate frames: the Body frame, the Navigation frame, the Earth frame and the Inertial frame. Figure 5.1 shows the four coordinate frames and how they are related to each other.

Each of the four coordinate frames used by the inertial navigation algorithm will now be explained in more detail. The first coordinate frame in the chain is the Body frame. The origin of the Body frame is fixed to the center of the vehicle (or another more applicable reference point) and its axes are oriented such that the X-axis points out through the front of the vehicle, the Z-axis points down through the bottom of the vehicle and the Y-axis points out through the right side of the vehicle. The Body frame translates and rotates with the vehicle as it moves through three-dimensional space, and the origin of the Body frame is used as the reference point for the (x, y, z) position of the vehicle. The Body

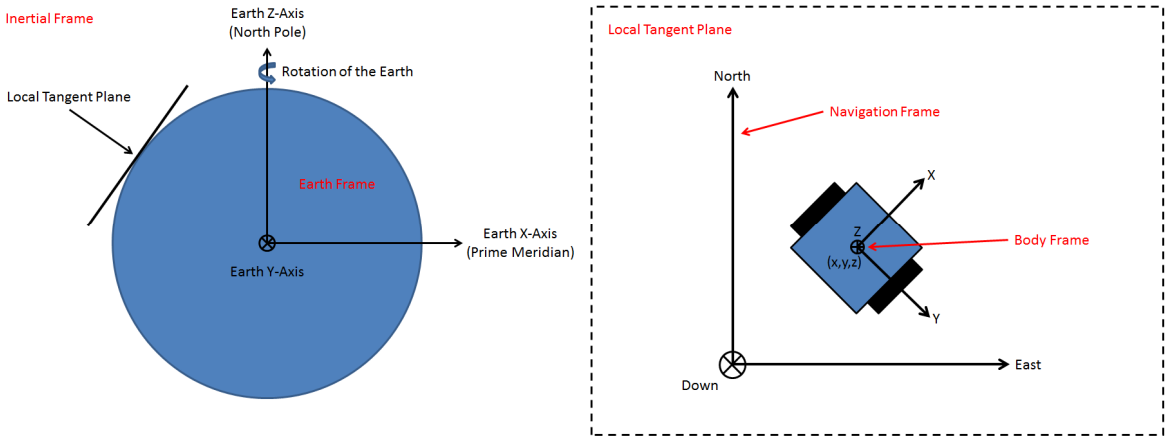

Figure 5.1: The four coordinate frames used by the inertial navigation algorithm.

frame is also used to define the positive acceleration and angular velocity conventions for the system. Positive accelerations are defined as accelerations that occur along the positive axes of the Body frame, and positive angular velocities are defined as shown in Figure 5.2.

The next coordinate frame up the chain from the Body frame is the Navigation frame. The origin of the Navigation frame is initially set at the starting location of the vehicle and its axes are oriented such that the X-axis points due North, the Z-axis points straight down towards the center of the Earth and the Y-axis points due East (this is also commonly referred to as the North, East, Down or NED convention). Unlike the Body frame, the Navigation frame does not rotate with the vehicle. Its orientation is permanently fixed according to the NED convention. The Navigation frame may or may not translate with the vehicle depending on how far the vehicle travels. In order to fully understand how the Navigation frame is defined, the next coordinate frame up the chain must first be introduced.

The next coordinate frame up the chain from the Navigation frame is the Earth frame. The origin of the Earth frame is fixed at the center of the Earth and its axes are oriented such that the X-axis points out of the Earth along the prime meridian, the Z-axis points out of the Earth through the North Pole and the Y-axis is aligned to complete the right-handed orthogonal coordinate system. The Earth frame rotates about its Z-axis along with the Earth (at a rate of approximately fifteen degrees per hour) so that the X-axis of the Earth frame is always aligned with the Prime Meridian.

In navigation applications where the vehicle is navigating over a small area, the origin of the

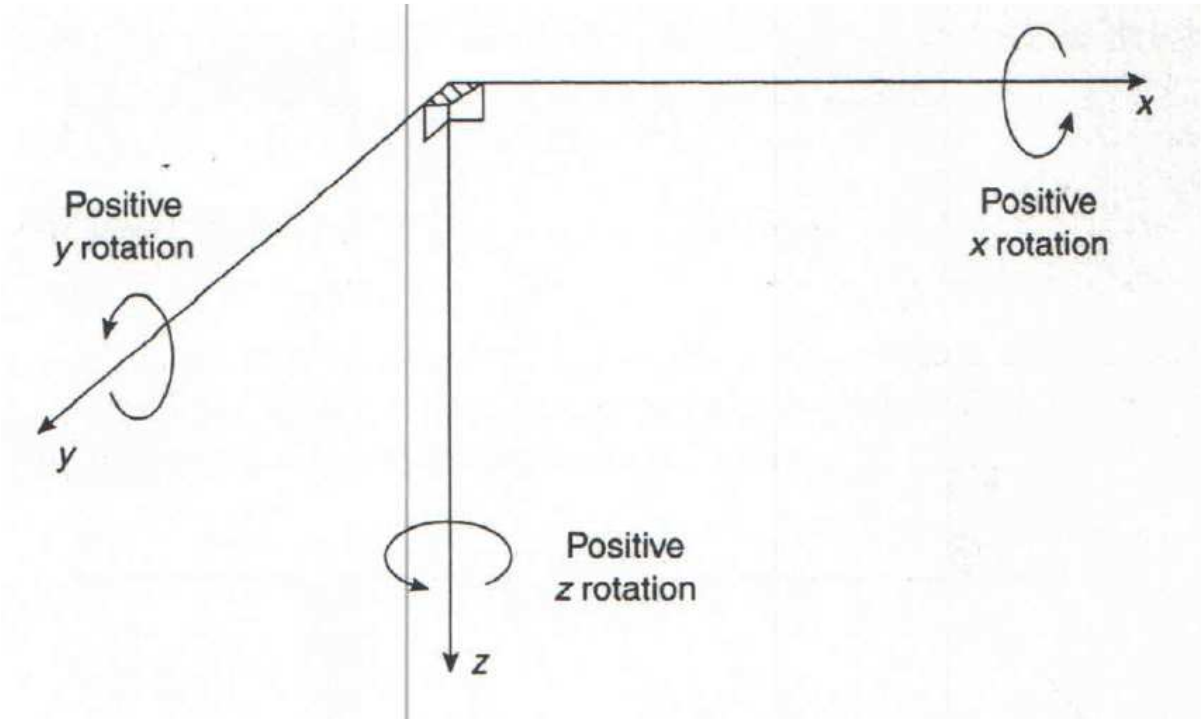

Figure 5.2: The positive angular velocity conventions defined by the Body frame.

Navigation frame remains permanently fixed at the same location on the Earth's surface (the starting location of the vehicle) and does not translate with the vehicle. If the vehicle is navigating over a small enough area, then the curvature of the Earth's surface is so slight that it can be considered negligible. Because the curvature of the Earth's surface is negligible in this scenario, the navigation of the vehicle can be constrained to a single planar surface instead of a curved surface. This surface is called the Local Tangent Plane (LTP), and it is defined as the plane created by the X- and Y-axes of the Navigation frame. The LTP is a perfectly flat surface that is tangent to the surface of the Earth at a single point (the starting location of the vehicle). Using the LTP greatly simplifies many aspects of the inertial navigation algorithm, but the downside is that it is only accurate over small areas. Using the LTP to define the position of the vehicle becomes more and more inaccurate as the vehicle travels further and further away from its starting location.

In navigation applications where the vehicle is navigating over a large area (i.e. navigating on a more global scale like an airplane would), the origin of the Navigation frame is moved to coincide with the origin of the Body frame and it translates with the vehicle as it moves across the Earth's surface. In these global-scale navigation scenarios, the curvature of the Earth becomes non-negligible and therefore the assumptions made by the LTP become unrealistic. The vehicle must be modeled as navigating on the curved surface of the Earth instead of the planar surface of the LTP. This means that the position of the vehicle must be represented in terms of spherical coordinates (latitude, longitude, altitude) instead of planar coordinates (NED). Although, because the Earth is not perfectly spherical in shape, an oblate spheroid model of the Earth is used to more accurately model the true shape of the Earth. The Global Positioning System (GPS) uses the WGS84 model of the Earth to produce more accurate measurements of the latitude, longitude and altitude of a GPS receiver on the surface of the Earth.

The last coordinate frame in the chain is the Inertial frame. The Inertial frame is a coordinate frame whose origin is fixed at some point in outer space far away from the surface of the Earth. For the purposes of this thesis, the inertial frame is simply treated as a reference frame that can be used to observe the rotation of the Earth.

Section 5.2 – Assumptions

This subsection lists the assumptions that are made by the inertial navigation algorithm presented in this thesis. Each of the assumptions is described in the bulleted list below.

- It is assumed that the vehicle is always navigating over a small enough area that the LTP assumptions are always valid (i.e. the origin of the Navigation frame remains fixed at a single

point on the Earth's surface (the starting location of the vehicle) and doesn't translate with the vehicle). Therefore, the position of the vehicle is referenced to the Navigation frame as opposed to the Earth frame and is denoted using NED planar position coordinates.

- It is assumed that the inertial data returned by the IMU conforms to the conventions of positive acceleration and positive angular velocity defined by the Body frame (seen Section 5.1 for details). If this is not the case, then the IMU data must be appropriately transformed to match the positive acceleration and positive angular velocity conventions defined by the Body frame. It is also assumed that any external misalignments between the IMU package and the vehicle are small enough that their effects can be considered negligible. Therefore, the orientation of the IMU and the orientation of the vehicle are considered to be the same at all times.
- It is assumed that when the inertial navigation algorithm begins, the initial linear velocity of the vehicle is always zero (a.k.a. the vehicle starts at a standstill), the initial position of the vehicle is always at the origin of the Navigation frame (0, 0, 0) and the initial heading of the vehicle is always zero degrees (a.k.a. aligned with the X-axis of the Navigation frame). It is also assumed that the heading of the vehicle is referenced to the local environment as opposed to the Earth's magnetic North Pole. Because the heading of the vehicle is locally referenced, its initial heading can be set to any desired starting value and a value of zero degrees is chosen for the sake of simplicity.
- It is assumed that the inertial data recorded at a specific time t is constant over the length of a single timestep Δt, meaning that the acceleration and angular velocity of the vehicle is assumed to be constant from time t until immediately before the next timestep $t + \Delta t$ when a new sample of inertial data is read from the IMU.

Section 5.3 – Inertial Navigation Algorithm Overview

This subsection provides a high-level overview of the inertial navigation algorithm. A basic block diagram of the inertial navigation algorithm can be seen in Figure 5.3. As Figure 5.3 shows, the inertial navigation algorithm is broken up into three phases. The first phase of the inertial navigation algorithm is the initialization phase. The initialization phase occurs before the vehicle begins to move. During the initialization phase, the initial linear velocity, position and orientation of the vehicle are determined. Assumptions are made about the initial linear velocity and position of the vehicle, but determining the initial orientation of the vehicle is more difficult. The initial roll and pitch of the vehicle are determined using a process called self-leveling, where measurements from the accelerometers are used to determine

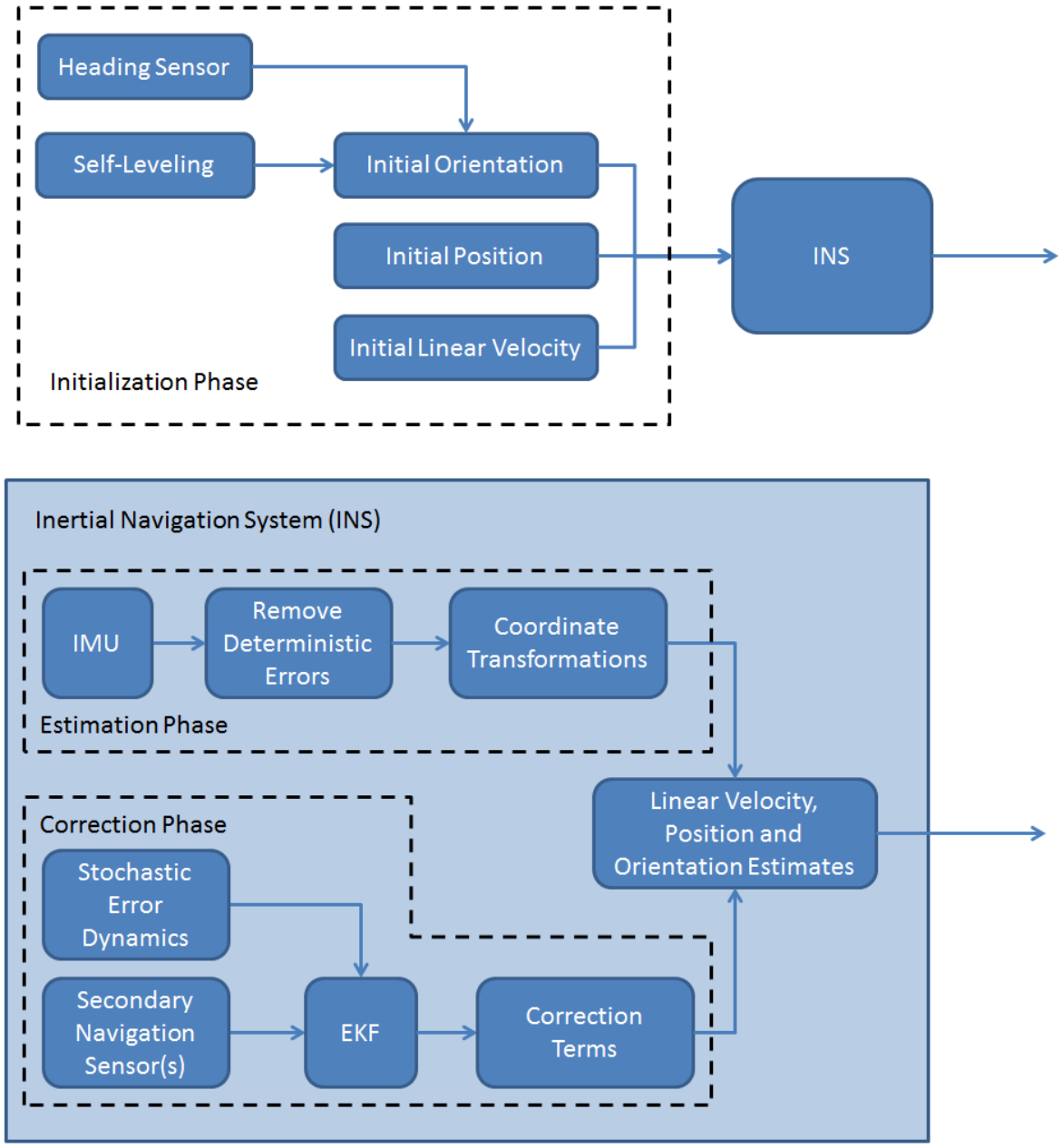

Figure 5.3: A high-level block diagram of the inertial navigation algorithm.

how the IMU (and by extension, the vehicle) is oriented with respect to gravity. In outdoor environments, the initial heading of the vehicle can be determined by using an external sensor system (such as a digital magnetic compass (DMC) or a three-axis magnetometer) to measure the global heading of the vehicle with respect to the Earth's magnetic North Pole. In indoor environments or other environments where

globally referenced heading information isn't strictly necessary, the initial heading of the vehicle can be measured with respect to other features or local landmarks within the environment.

After the initialization phase ends and the vehicle begins to move, the algorithm enters the estimation phase. During the estimation phase, inertial data from the IMU is used to produce initial estimates of the linear velocity, position and orientation of the vehicle as it moves through its environment. The effects of the static biases, scale factors and misalignment errors are removed from the inertial data as well as the effects of other external factors such as the angular velocity caused by the rotation of the Earth. Then, the corrected angular velocity data is used to estimate the orientation of the vehicle. Once the orientation of the vehicle has been determined, the corrected acceleration data is converted from the Body frame into the Navigation frame, the effects of gravity are removed and the corrected acceleration data is integrated over time using Runge-Kutta numerical integration to estimate the linear velocity and position of the vehicle. The end result of the estimation phase is initial estimates of the linear velocity, position and orientation of the vehicle.

After the estimation phase, the algorithm enters the correction phase. In the correction phase, an EKF is used to combine the dynamics of the stochastic errors (i.e. measurement noise processes, drifting biases, etc.) and secondary navigation information to produce correction terms for the initial linear velocity, position and orientation estimates produced in the estimation phase. These correction terms are applied to the initial estimates from the estimation phase to produce the final estimates of linear velocity, position and orientation for a specific timestep. However, the EKF can only generate correction terms if secondary navigation information is available. So if there is a timestep where no new secondary navigation information is available, then the estimation phase of the algorithm will be executed and the estimation-error covariances of the EKF will be propagated, but no correction terms will be produced by the EKF. In these instances, the initial estimates from the estimation phase of the algorithm become the final estimates of linear velocity, position and orientation for that timestep.

After the correction phase, the algorithm loops back to the estimation phase and the process described above is repeated using a new set of inertial measurements from the IMU and any new secondary navigation information. The estimation and correction phases of the algorithm are repeated indefinitely in this manner until the vehicle is either powered off or another goal condition is reached.

The following subsections will provide a more detailed description of each of the three phases of the inertial navigation algorithm.

Section 5.4 – The Initialization Phase

This subsection explores the initialization phase of the inertial navigation algorithm in more detail and explains how the initial linear velocity, position and orientation of the vehicle are determined. A basic block diagram of the initialization phase can be seen in Figure 5.3. At the beginning of the algorithm (time t_0), the vehicle has some known initial linear velocity, position and orientation. As mentioned previously in Section 5.2, it is assumed that the initial linear velocity of the vehicle is zero, the initial position of the vehicle is at the origin of the Navigation frame (0, 0, 0) and the initial heading of the vehicle is zero degrees.

Before the vehicle begins to move, there is a sixty second period of time where the vehicle remains stationary. During this time, the inertial navigation algorithm performs a process called self-leveling. Self-leveling is very similar to the Multi-Angle Experiment described earlier in Section 4.3 in that it uses gravity as a source of known and constant acceleration to determine the initial roll and pitch of the IMU, and by extension the vehicle. These initial measurements of roll and pitch are used along with the initial heading of the vehicle to create a direction cosine matrix (DCM) that describes the initial three-dimensional orientation of the vehicle. The self-leveling routine is performed using a recursive linear least squares estimation process that is very similar to one used in the Multi-Angle Experiment to estimate the combined scale factor and misalignment error matrices for the accelerometers (C_a). Refer to Section 4.3.1 for more details.

Section 5.5 – The Estimation Phase

This subsection explores the estimation phase of the inertial navigation algorithm in more detail and explains how raw inertial measurements from the IMU are used to produce initial estimates of the linear velocity, position and orientation of the vehicle. A detailed block diagram of the estimation phase of the inertial navigation algorithm is shown in Figure 5.4.

The purpose of the estimation phase is to calculate an initial estimate of what the linear velocity, position and orientation of the vehicle will be at the future timestep $t + \Delta t$. At the current timestep t the vehicle has some linear velocity, position and orientation, and the future linear velocity, position and orientation of the vehicle at time $t + \Delta t$ is estimated using inertial information read from the IMU at the current timestep t. As stated previously in Section 5.2, the inertial data recorded at a specific time t is assumed to be constant over the length of a single timestep Δt.

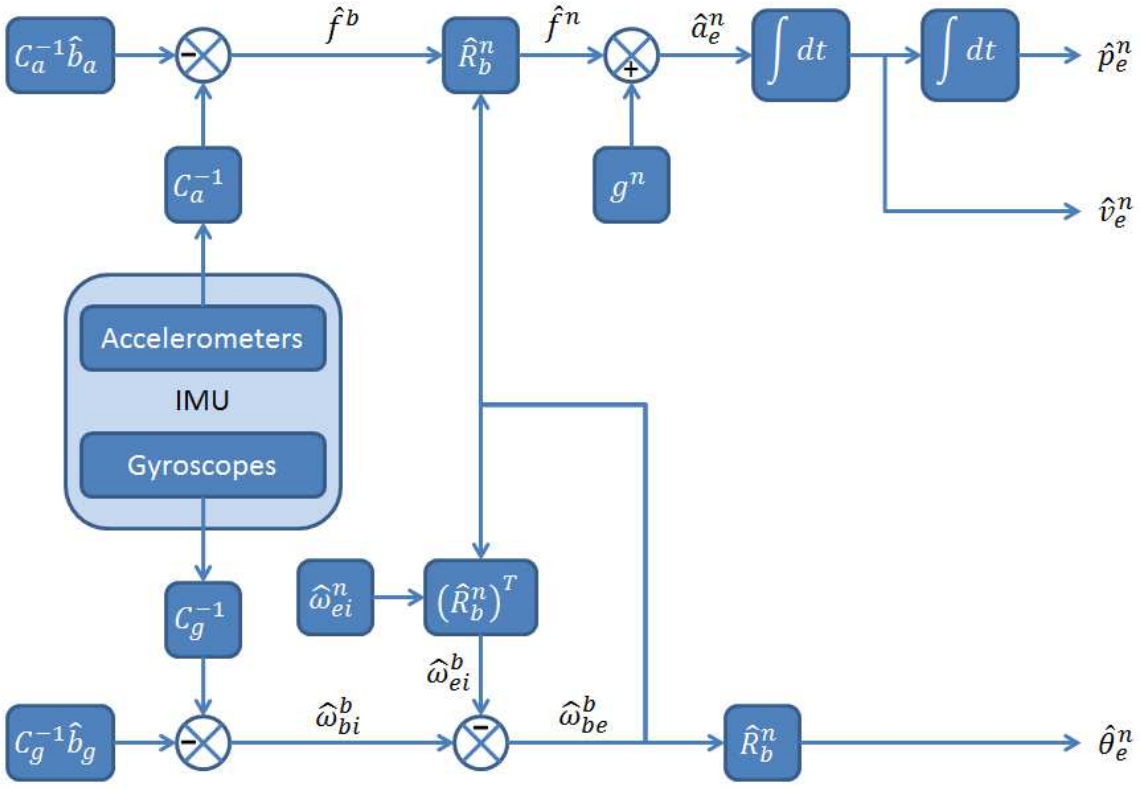

Figure 5.4: A detailed block diagram of the estimation phase of the inertial navigation algorithm.

The estimation process begins by reading a set of raw inertial measurements (three-dimensional acceleration, three-dimensional angular velocity and temperature) from the IMU. Based on the temperature measurement, the corresponding static biases and scale factors for the accelerometers and gyroscopes are retrieved from a look-up table and used to calculate compensation terms as shown in Equations 5.1 and 5.2.

$$y = Cx + b \qquad \text{Eq. 5.1}$$

$$x = C^{-1}y - C^{-1}b \qquad \text{Eq. 5.2}$$

where x is the true acceleration or angular velocity, y is the measured value of the acceleration or angular velocity, C^{-1} is the scale factor and misalignment error compensation term and $C^{-1}b$ is the static bias compensation term. These compensation terms are then applied to the raw inertial data to remove the effects of the static biases, scale factors and misalignment errors. The compensated acceleration and angular velocity terms are denoted by the variables $\hat{f}^b$ and $\hat{\omega}_{bi}^b$.

Next, another compensation term ($\hat{\omega}_{ei}^{b}$) is applied to the angular velocity data to remove the effects of the rotation of the Earth. The rotation of the Earth creates a very small angular velocity that is constantly measured by the gyroscopes. Over time this angular velocity will propagate through the inertial navigation algorithm and cause the estimated orientation of the vehicle to drift. The angular velocity of the Earth is originally referenced to the Z-axis of the Earth frame, but it can be transformed into the Navigation frame if the latitude, longitude and altitude of the vehicle are known. And because the vehicle is assumed to be navigating on the LTP (see Section 5.2), the origin of the Navigation frame remains fixed at a single point on the Earth's surface and therefore the transformation between the Earth frame and the Navigation frame always remains the same. This means that the angular velocity of the Earth can be expressed in the Navigation frame as a constant three-dimensional angular velocity term. The quantity $\hat{\omega}_{ei}^{b}$ is calculated by transforming the angular velocity of the Earth from the Navigation frame ($\hat{\omega}_{ei}^{n}$) into the Body frame using the transpose of the DCM that represents the current orientation of the vehicle at time t, which is $\left(\hat{R}_{b}^{n}(t)\right)^{T}$. The quantity $\hat{\omega}_{ei}^{b}$ is then subtracted from the compensated angular velocity data ($\hat{\omega}_{bi}^{b}$) to produce the quantity $\hat{\omega}_{be}^{b}$.

The next step in the process is to calculate what the orientation of the vehicle will be at the future timestep $t + \Delta t$. This is done by propagating the $\hat{R}_{b}^{n}(t)$ matrix forward through time based on the fully compensated angular velocity data $\hat{\omega}_{be}^{b}$. Equations 5.3 through 5.6 define how a generic DCM is propagated forward through time [16,17].

$$\hat{R}_{b}^{n}(t + \Delta t) = \hat{R}_{b}^{n}(t) \begin{bmatrix} 1 & -\Delta\theta_z & \Delta\theta_y \\ \Delta\theta_z & 1 & -\Delta\theta_x \\ -\Delta\theta_y & \Delta\theta_x & 1 \end{bmatrix} \quad \text{Eq. 5.3}$$

$$\Delta\theta_x = \omega_x \Delta t \qquad \Delta\theta_y = \omega_y \Delta t \qquad \Delta\theta_z = \omega_z \Delta t \quad \text{Eq. 5.4 – 5.6}$$

The $\hat{R}_{b}^{n}(t + \Delta t)$ matrix is then used to calculate the initial estimates of the roll, pitch and heading of the vehicle at the future timestep $t + \Delta t$.

Now that the angular velocity data has been processed, the next step is to process the acceleration data. First, the compensated acceleration data ($\hat{f}^{b}$) is converted from the Body frame into the Navigation frame using the DCM that represents the current orientation of the vehicle ($\hat{R}_{b}^{n}(t)$) and the effects of gravity are removed as shown in Equation 5.7. This results in the fully compensated acceleration term $\hat{a}_{e}^{n}(t)$. This fully compensated acceleration term is then integrated over time using Runge-Kutta

numerical integration to calculate the initial estimates of the linear velocity and position of the vehicle at the future timestep $t + \Delta t$ as shown in Equations 5.8 and 5.9 [18].

$$\hat{a}_e^n(t) = \hat{R}_b^n(t)\hat{f}^b + g^n \quad \text{Eq. 5.7}$$

$$\hat{v}(t + \Delta t) = \hat{v}(t) + \frac{\Delta t}{6}\left(\hat{a}_e^n(t) + 4\hat{a}_e^n\left(t + \frac{\Delta t}{2}\right) + \hat{a}_e^n(t + \Delta t)\right) \quad \text{Eq. 5.8}$$

$$\hat{p}(t + \Delta t) = \hat{p}(t) + \hat{v}(t)\Delta t + \frac{\Delta t^2}{6}\left(\hat{a}_e^n(t) + 2\hat{a}_e^n\left(t + \frac{\Delta t}{2}\right)\right) \quad \text{Eq. 5.9}$$

The initial estimates of linear velocity, position and orientation produced in this phase of the inertial navigation algorithm accurately model the true linear velocity, position and orientation of the vehicle over short periods of time, but imperfections and drifts in the IMU data will eventually cause the estimates to diverge from the true linear velocity, position and orientation of the vehicle. In the next phase of the algorithm, the correction phase, the dynamics of the stochastic IMU errors and secondary navigation information will be incorporated into an EKF to produce correction terms for these initial estimates. The correction terms generated by the EKF will help to correct the errors and drifts that build up in the initial estimates of linear velocity, position and orientation due to the imperfections and drifts in the inertial data returned by the IMU.

Section 5.6 – The Correction Phase

This subsection explores the correction phase of the inertial navigation algorithm in more detail and explains how the dynamics of the stochastic IMU errors and secondary navigation information are combined in an EKF to produce correction terms for the initial estimates of linear velocity, position and orientation from the estimation phase of the algorithm. A block diagram of the correction phase of the inertial navigation algorithm is shown in Figure 5.5.

The purpose of the correction phase is to generate correction terms for the initial estimates of linear velocity, position and orientation that are produced in the estimation phase of the inertial navigation algorithm. Specifically, there are fifteen correction terms that need to be generated: three each for the accelerometer and gyroscope biases (denoted as Δb_a and Δb_g respectively), three for the linear velocity estimates (denoted as Δv_e^n), three for the position estimates (denoted as Δp_e^n) and three for the orientation estimates (denoted as $\Delta\theta_{bn}^b$). These correction terms are generated using an EKF. An EKF is a recursive

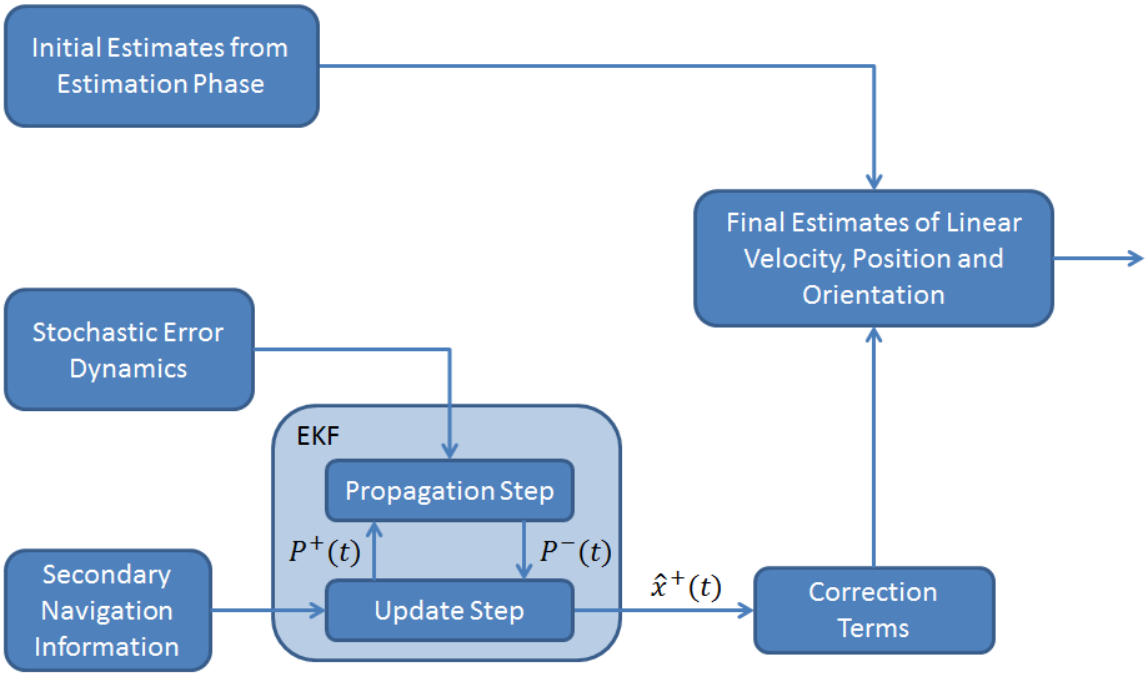

Figure 5.5: A detailed block diagram of the correction phase of the inertial navigation algorithm.

Bayesian estimator that estimates one or more dynamic states over the course of time. The fifteen-element state vector of the EKF is shown in Equation 5.10.

$$x(t) = \begin{bmatrix} \Delta b_a & \Delta v_e^n & \Delta p_e^n & \Delta b_g & \Delta \theta_{bn}^b \end{bmatrix}^T \quad \text{Eq. 5.10}$$

The basic operation of a generic EKF is as follows. An EKF is implemented with three main steps: an initialization step, a propagation step and an update step. During the initialization step, an initial estimate of the state vector ($\hat{x}^+(0)$) and an initial estimation-error covariance matrix ($P^+(0)$) are determined based on any knowledge that is available about the initial state of the system. During the propagation step an a priori estimate of the state vector is generated based on the system model that defines how the states behave and the estimation-error covariance matrix is updated based on the system model as well. During the update step an a posteriori estimate of the state vector is generated based on the a priori state estimate from the propagation step and either direct or indirect measurements of some or all of the states in the state vector. The estimation-error covariance matrix is also updated based on the measurement model that defines the relationship between the measurement data and the state vector. The propagation and update steps are then repeated during each subsequent iteration of the EKF to produce new estimates of the dynamic state vector over the course of time.

The EKF presented in this thesis follows a slightly different approach than a generic EKF. The initialization step occurs the same way, and an initial estimate of the state vector ($\hat{x}^+(0)$) and an initial estimation-error covariance matrix ($P^+(0)$) are determined based on any knowledge that is available about the initial state of the system (as shown in Equations 5.11 and 5.12).

$$\hat{x}^+(0) = [0 \quad 0 \quad 0 \quad 0 \quad 0]^T \qquad \text{Eq. 5.11}$$

$$P^+(0) = \begin{bmatrix} \sigma_{ba}^2 I & 0 & 0 & 0 & 0 \\ 0 & I & 0 & 0 & 0 \\ 0 & 0 & I & 0 & 0 \\ 0 & 0 & 0 & \sigma_{bg}^2 I & 0 \\ 0 & 0 & 0 & 0 & I \end{bmatrix} \qquad \text{Eq. 5.12}$$

The initial estimate of the state vector ($\hat{x}^+(0)$) is set to zero because during the first timestep the linear velocity, position and orientation of the vehicle are already known so no correction terms are necessary. The initial estimation-error covariance matrix ($P^+(0)$) is set up as a diagonal matrix with Identity matrices along its diagonal except for the positions that correspond to the bias correction states. These positions are populated with the turn-on to turn-on bias variation terms calculated in Section 4.2 because they represent the initial variability in the values of the static biases.

After the initialization step, the EKF moves on to the propagation step. During the propagation step, the dynamics of the stochastic IMU errors are used to generate the a priori estimation-error covariance matrix, but no a priori state estimates are generated. The a priori estimate of the state vector is simply set to contain all zeroes.

The equations for implementing the propagation step of the EKF are shown in Equations 5.13 through 5.16 [19,20].

$$F(t) = \begin{bmatrix} -\varphi_a I & 0 & 0 & 0 & 0 \\ -\hat{R}_b^n(t) & 0 & 0 & 0 & \hat{R}_b^n(t)\left(-\hat{f}^b(t)\times\right) \\ 0 & I & 0 & 0 & 0 \\ 0 & 0 & 0 & -\varphi_g I & 0 \\ 0 & 0 & 0 & -I & \left(-\hat{\omega}_{ei}^b(t)\times\right) \end{bmatrix} \qquad \text{Eq. 5.13}$$

$$Q = \begin{bmatrix} (\sigma_{bd}^a)^2 I & 0 & 0 & 0 & 0 \\ 0 & \sigma_a^2 I & 0 & 0 & 0 \\ 0 & 0 & \sigma_p^2 I & 0 & 0 \\ 0 & 0 & 0 & \left(\sigma_{bd}^g\right)^2 I & 0 \\ 0 & 0 & 0 & 0 & \sigma_g^2 I \end{bmatrix} \qquad \text{Eq. 5.14}$$

$$A(t) = I + F(t)\Delta t \qquad \text{Eq. 5.15}$$

$$P^{-}(t) = A(t)P^{+}(t - \Delta t)A(t)^{T} + Q\Delta t \quad \text{Eq. 5.16}$$

The terms in the state transition matrix ($F(t)$) show how each of the states in the state vector ($x(t)$) are related to each other at the current timestep. The variance terms along the diagonal of the process noise covariance matrix (Q) represent the inherent randomness of each of the states in the state vector ($x(t)$). The first and fourth variance terms along the diagonal represents the variances of the WGN processes that drive the drifting biases of the accelerometers and gyroscopes respectively. The second and fifth variance terms along the diagonal represents the variances of the measurement noises for the accelerometers and gyroscopes respectively. Finally, the third variance term along the diagonal represents the variances of the position correction states in the state vector (Δp_e^n).

Equations 5.15 and 5.16 define how the a priori estimation-error covariance matrix for the current timestep ($P^{-}(t)$) is calculated based on the state transition matrix for the current timestep ($F(t)$), the a posteriori estimation-error covariance matrix from the previous timestep ($P^{+}(t - \Delta t)$) and the process noise covariance matrix (Q).

After the propagation step, the EKF moves on to the update step. During the update step, the EKF uses the a priori estimation-error covariance matrix from the propagation step and any available secondary navigation information to generate a set of correction terms for a specific timestep. As mentioned previously, the update step of the EKF is only executed when new secondary navigation information is available. If there is no new secondary navigation information for a specific timestep, then correction terms cannot be generated and only the propagation step of the EKF is executed. Typically, this secondary navigation information is produced by a separate sensor system such as GPS or a visual odometry system. The secondary navigation information for the inertial navigation algorithm presented in this thesis is generated using a combination of zero velocity updates (ZUPTs), position fixes and orientation fixes. A ZUPT is a period of time when the vehicle is known to have both a linear and angular velocity of zero. If the ZUPT occurs at a known location, then the position and orientation of the vehicle can be recorded as well. These accurate measurements of the true position and orientation of the vehicle are referred to as position and orientation fixes respectively.

The equations for implementing the update step of the EKF are shown in Equations 5.17 through 5.22 [19,20].

$$y(t) = \begin{bmatrix} \Delta v_e^n & \Delta p_e^n & \Delta b_g & \Delta \theta_{bn}^b \end{bmatrix}^T \quad \text{Eq. 5.17}$$

$$H = \begin{bmatrix} 0 & I & 0 & 0 & 0 \\ 0 & 0 & I & 0 & 0 \\ 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & 0 & I \end{bmatrix} \quad \text{Eq. 5.18}$$

$$R = \begin{bmatrix} \sigma_{\Delta v}^2 I & 0 & 0 & 0 \\ 0 & \sigma_{\Delta p}^2 I & 0 & 0 \\ 0 & 0 & \sigma_{\Delta b}^2 I & 0 \\ 0 & 0 & 0 & \sigma_{\Delta \theta}^2 I \end{bmatrix} \quad \text{Eq. 5.19}$$

$$K(t) = P^-(t)H^T(HP^-(t)H^T + R)^{-1} \quad \text{Eq. 5.20}$$

$$\hat{x}^+(t) = K(t)y(t) \quad \text{Eq. 5.21}$$

$$P^+(t) = (I - K(t)H)P^-(t)(I - K(t)H)^T + K(t)RK(t)^T \quad \text{Eq. 5.22}$$

The terms in the measurement data vector ($y(t)$) represent the twelve secondary navigation information terms that are recorded when the vehicle undergoes a ZUPT at a specific time t. The terms in the measurement matrix (H) show how the terms in the measurement data vector ($y(t)$) are related to the terms in the state vector ($x(t)$). The variance terms along the diagonal of the measurement noise covariance matrix (R) represent the variances of each of the measurements in the measurement data vector ($y(t)$). The first variance term along the diagonal represents the variances of the measurements that are made of the velocity correction states (Δv_e^n). The second term along the diagonal represents the variances of the measurements that are made of the position correction states (Δp_e^n). The third term along the diagonal represents the variances of the measurements that are made of the gyroscope bias correction states (Δb_g). And finally, the fourth term along the diagonal represents the variances of the measurements that are made of the orientation correction states ($\Delta \theta_{bn}^b$).

Equation 5.20 defines how the optimal gain matrix for the current timestep ($K(t)$) is calculated based on the a priori estimation-error covariance matrix for the current timestep ($P^-(t)$), the measurement matrix (H) and the measurement noise covariance matrix (R). Equation 5.21 defines how the a posteriori estimate of the state vector for the current timestep ($\hat{x}^+(t)$) is calculated based on the optimal gain matrix for the current timestep ($K(t)$) and the measurement data vector for the current timestep ($y(t)$). And finally, Equation 5.22 defines how the a posteriori estimation-error covariance matrix for the current timestep ($P^+(t)$) is calculated based on the a priori estimation-error covariance matrix for the current timestep ($P^-(t)$), the optimal gain matrix for the current timestep ($K(t)$), the measurement matrix (H) and the measurement noise covariance matrix (R).

After the update step, assuming that secondary navigation information was available during the current timestep and the update step was successfully performed, the fifteen correction terms generated by the EKF are used to update the quantities that they correspond to within the inertial navigation algorithm.

The equations for applying the correction terms generated by the EKF are shown in Equations 5.23 through 5.27 [20].

$$\hat{b}_a = \hat{b}_a + \Delta b_a \quad \text{Eq. 5.23}$$

$$\hat{v}_e^n = \hat{v}_e^n + \Delta v_e^n \quad \text{Eq. 5.24}$$

$$\hat{p}_e^n = \hat{p}_e^n + \Delta p_e^n \quad \text{Eq. 5.25}$$

$$\hat{b}_g = \hat{b}_g + \Delta b_g \quad \text{Eq. 5.26}$$

$$\hat{\theta}_e^n = \hat{\theta}_e^n + \Delta \theta_{bn}^b \quad \text{Eq. 5.27}$$

Once the fifteen quantities in the above equations have been updated, the correction phase of the inertial navigation algorithm is over. The inertial navigation algorithm moves on to the next timestep and the estimation and correction phases are repeated using a new set of inertial data from the IMU and a new set of secondary navigation information (if any is available). The estimation and correction phases of the inertial navigation algorithm are repeated indefinitely in this manner until the vehicle is either powered off or another goal condition is reached.

Section 6 – UGV Navigation Test Plan

Once the inertial navigation algorithm has been completely developed and coded, it must be tested using inertial data from a real-world navigation scenario. To acquire this data, both the SparkFun and ADI IMUs were rigidly mounted to a UGV, which was then driven through a predefined test course. The UGV was driven through the test course three times (starting and ending in the same place each time), resulting in three independent sets of inertial data for each IMU. The inertial data recorded from both IMUs during these tests was used to determine how accurately the inertial navigation algorithm can track the linear velocity, position and orientation of a moving vehicle. This section will explain the overall methodology behind the UGV testing, including how inertial data was logged from the two IMUs, how the test course was set up, how the vehicle was driven and how the true linear velocity, position and orientation of the vehicle was determined.

Section 6.1 – The Test Course

The UGV test course was set up on the third floor of the Atwater Kent building on the Worcester Polytechnic Institute campus. This location was chosen for several reasons. Indoor environments (and hallways in particular) are typically very structured and flat with few obstructions. This means that any vehicle driving through an indoor environment is essentially navigating along a single planar surface that is also level with gravity. Deviations from this ideal surface (i.e. uneven or un-level patches of the floor) are expected to be fairly minimal and shouldn't have much of an effect on the vehicle's ability to maneuver through the environment. Indoor environments are also simple to measure and map, which is essential for accurately placing position markers to gather truth data for the inertial navigation algorithm.

The test course itself is rectangular in shape and measures 22 meters long by 16.5 meters wide. A rectangular test course was chosen so that the vehicle could travel in a loop and finish at the same location that it started at. This loop closure is a useful metric to have because after one complete loop of the test course, the vehicle should ideally be at the exact same location that it started at. Any discrepancies between the estimated starting and finishing location of the vehicle can be used as a metric to assess the accuracy of the inertial navigation algorithm.

To generate secondary navigation information for the EKF component of the inertial navigation algorithm (as described in Section 5.6), the vehicle must be periodically stopped at specific known locations along the test course so that ZUPTs, position fixes and orientation fixes can be recorded. In order to mark the locations where the vehicle should be stopped, position markers were laid out at two

Figure 6.1: The position markers outlining the locations of ZUPTs along the length of the test course.

meter intervals along the entire length of the test course as seen in Figure 6.1 (with the exception of two position markers that had to be laid out at two and a half meter intervals due to the 16.5 meter length of two of the hallways). The position markers were laid out directly in the center of the hallways, and also served as convenient waypoints to help keep the vehicle from straying off of the predefined test course. Since the position markers were laid out at known intervals, the position and orientation of the vehicle can be accurately determined when it is stopped on top of a specific marker.

In a more elaborate INS, the secondary navigation information would come from a physical sensor system (GPS for example) at a frequency on the order of 1Hz or faster. However, due to the spacing of the position markers and the relatively low top speed of the vehicle, secondary navigation information was only recorded at approximately 0.1Hz (approximately once every ten seconds). Because of the long stretches in between secondary navigation information readings, the drifts and other imperfections in the inertial data have a longer amount of time to accumulate.

Section 6.2 – Configuring and Driving the UGV

The vehicle used to navigate the test course was a Husky A100 UGV from the company Clearpath Robotics. A picture of the Husky A100 UGV can be seen in Figure 6.2. The Husky A100 UGV is a six-wheeled, skid-steered vehicle, which makes it ideal for outdoor environments with rough terrain. However, this wheel configuration can present some problems when trying to maneuver on the carpeted surfaces typically found in many indoor environments. Because of the high-traction, all-terrain treads of the vehicle's wheels and the skid-steering nature of how the vehicle turns there is a lot of friction that builds up between the wheels of the vehicle and the carpet. This friction results in turning movements that are jerky and non-fluid. To get around this issue, each of the vehicle's six wheels was coated in a single layer of duct tape (as seen in Figure 6.3) which significantly reduced the traction of the vehicle's tires as well as the coefficient of friction between the vehicle's wheels and the carpet. This reduction in friction allowed the vehicle to make turns that were much more fluid and continuous.

Before each test run, both the SparkFun and ADI IMUs were rigidly mounted to the vehicle as shown in Figure 6.4. At the beginning of each test run, the vehicle was kept stationary for sixty seconds while the two IMUs recorded inertial data. The data from this initial stationary period was used to

Figure 6.2: The Husky A100 UGV.

Figure 6.3: The wheels of the Husky A100 UGV coated in duct tape to reduce the coefficient of friction between the wheels and the carpeted floor.

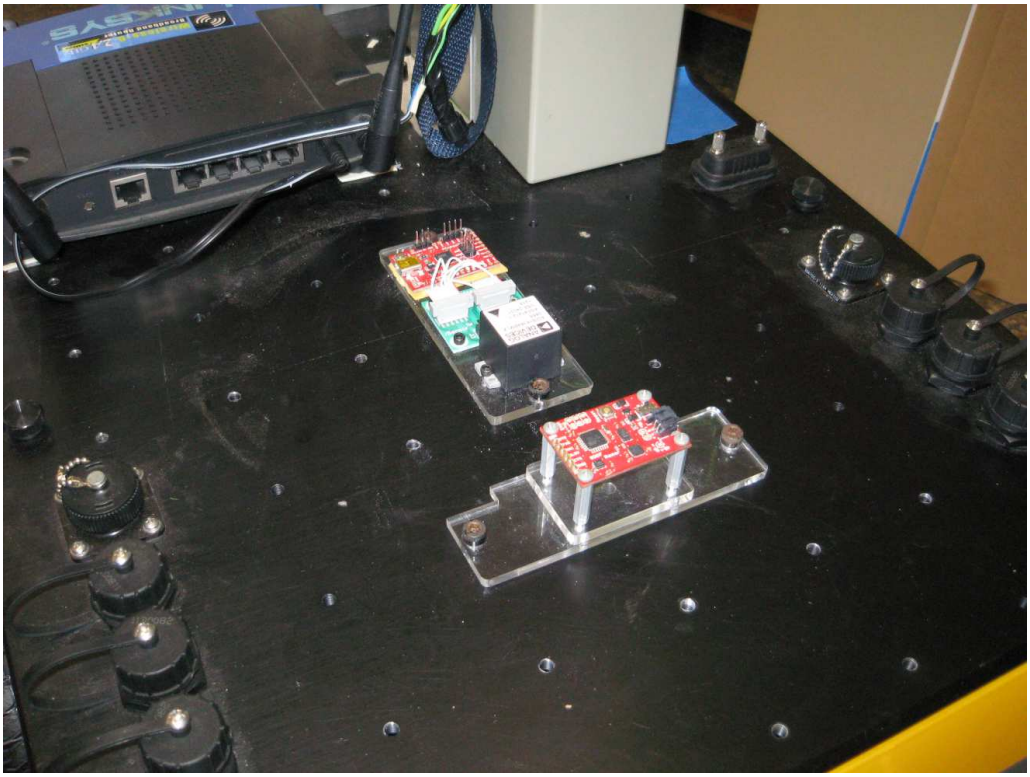

Figure 6.4: The SparkFun and ADI IMUs rigidly mounted to the Husky A100 UGV.

Figure 6.5: The Husky A100 UGV positioned at the beginning of the test course.

perform the self-leveling routine discussed in Section 5.4. The vehicle initially began each test run at one of the corners of the test course as shown in Figure 6.5. It was then manually driven through a single complete loop of the test course, stopping for ten seconds at each one of the designated position markers to record ZUPTs, position fix and orientation fix data, until it arrived back at its original starting location. Throughout the test run, inertial data was continuously logged from both the SparkFun and ADI IMUs at a rate of 25Hz.

Another feature of the Husky A100 UGV is that it has a built in speed control algorithm, which means that it can be commanded to drive forwards, backwards or turn in either direction at a specific rate. The forward velocity of the vehicle was kept at a constant rate of 0.5 meters per second whenever possible during the course of a test run. The vehicle only accelerated or decelerated when it was either starting from or slowing to a stop respectively. Likewise, the turning velocity of the vehicle was also kept at a constant rate of approximately 0.4 rad/s throughout the course of any turning maneuvers, and it would only accelerate or decelerate when it was either just starting or just ending a turning maneuver respectively.

Section 7 – Results and Discussion

This section presents and discusses the results obtained from the UGV navigation testing described in Section 6. All of the results presented in this section are from the same navigation testing trial (Trial #3). The complete results from all three of the navigation testing trials are located in the Appendix. The first three subsections contain the results obtained from running the two INS units through some specific navigation scenarios and the corresponding discussion of those results. The final subsection contains a direct comparison of the two INS units and an evaluation of which INS was able to produce the most accurate estimates of the linear velocity, position and orientation of the UGV over the length of the test course.

Section 7.1 – Inertial Data Only

The results presented in this subsection illustrate how the SparkFun-based and ADI-based INS units perform when they are limited to using only inertial data from the IMUs and no secondary navigation information. Figure 7.1 shows the estimates of the position and orientation of the vehicle over time produced by the two INS units versus the true position and orientation of the vehicle over time.

Figure 7.1 shows just how important secondary navigation information is to the success of the inertial navigation algorithm. With no secondary navigation information to help compensate for the

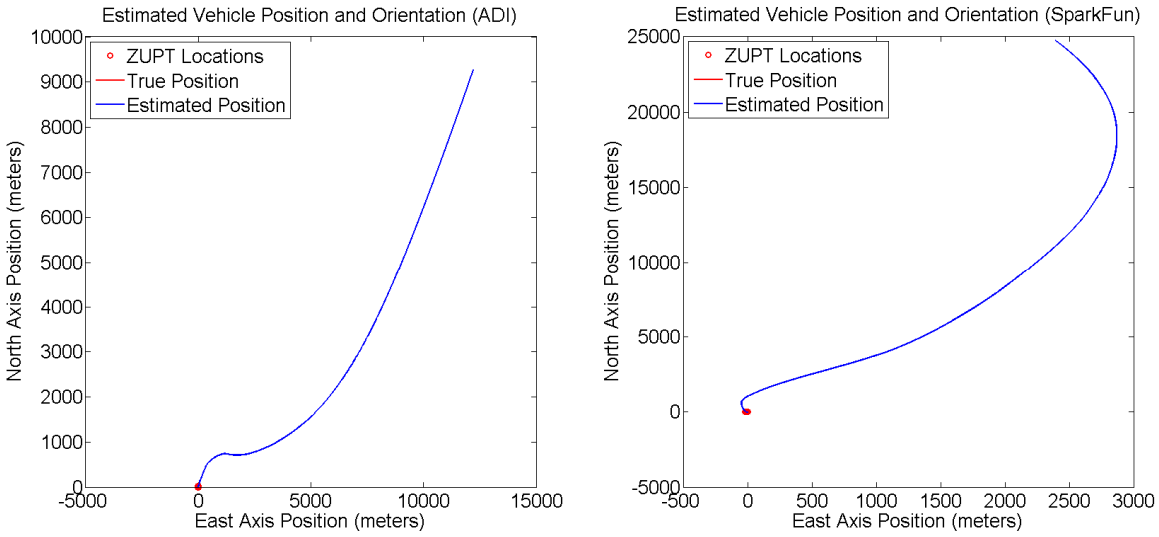

Figure 7.1: Estimated position and orientation of the vehicle versus true position and orientation of the vehicle (Inertial data only).

inaccuracies and drifts in the inertial data, the errors in the estimated linear velocity, position and orientation of the vehicle grow unbounded over time. If left unchecked these errors become very large very quickly, which is why accurately compensating for the drifts and other imperfections in the inertial data using secondary navigation information is a huge part of successfully navigating using inertial sensors.

Section 7.2 – Inertial Data and ZUPTs

The results presented in this subsection illustrate how the SparkFun-based and ADI-based INS units perform when they are limited to using only inertial data from the IMUs and velocity fixes (i.e. ZUPTs). Figure 7.2 shows the estimates of the position and orientation of the vehicle over time produced by the two INS units versus the true position and orientation of the vehicle over time.

Compared to the results from Section 7.1, the results in Figure 7.2 clearly show that the linear velocity information obtained from periodically performing ZUPTs helps to significantly limit the errors in the estimated linear velocity, position and orientation of the vehicle over time. The secondary navigation information obtained from these ZUPTs allows the EKF to generate correction terms for the initial linear velocity estimates based on the inertial data from the IMU. Applying these correction terms to the initial linear velocity estimates produces final estimates of the linear velocity, position and orientation of the vehicle that are much more accurate.

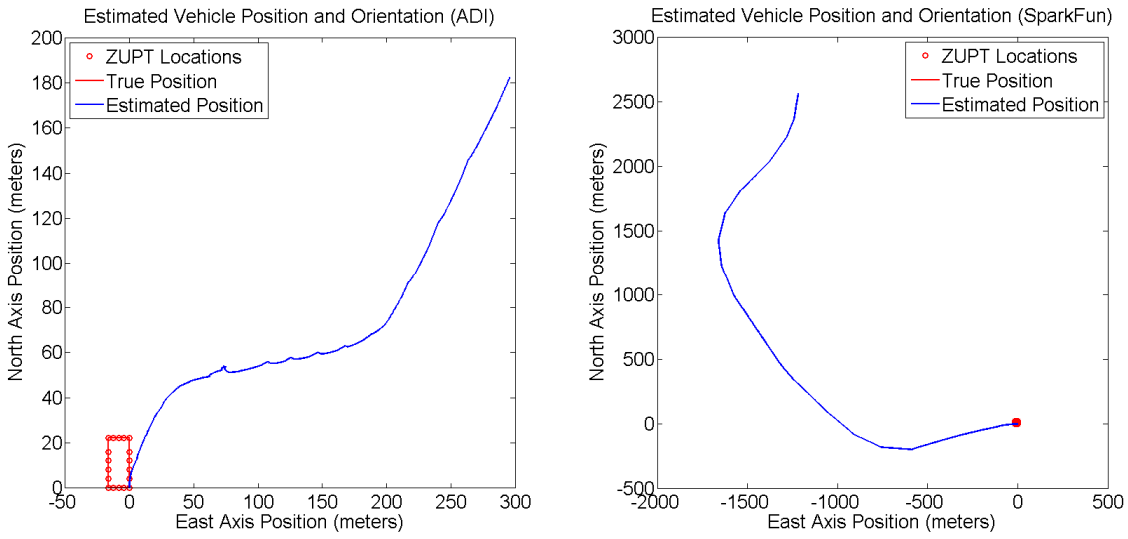

Figure 7.2: Estimated position and orientation of the vehicle versus true position and orientation of the vehicle (Inertial data and ZUPTs only).

Section 7.3 – Inertial Data, ZUPTs and Position and Orientation Fixes

The results presented in this subsection illustrate how the SparkFun-based and ADI-based INS units perform when they are free to use all of the information that is available to them (i.e. inertial data from the IMUs, velocity fixes (ZUPTs) and position and orientation fixes). These results represent the navigation capabilities of the fully completed INS units. Figure 7.3 shows the estimates of the position and orientation of the vehicle over time produced by the two INS units versus the true position and orientation of the vehicle over time.

As shown previously in Section 7.2, the addition of secondary linear velocity information into the system greatly increased the accuracy of the estimates produced by both INS units. Now, by adding secondary position and orientation information into the system as well, the accuracy of the estimates is increased even further. Figure 7.3 shows how an inertial navigation system can produce accurate linear velocity, position and orientation estimates when it is provided with enough secondary navigation information to sufficiently compensate for the drifts and other imperfections in the IMU data. The estimates are still not perfect, but the secondary navigation information for these experiments was collected at a relatively low frequency (approximately once every ten seconds). In most commercial, real-time INS units, secondary navigation information is gathered much more quickly (approximately once per second or faster), so EKF corrections are typically generated at a much higher frequency. And as EKF corrections are generated more and more frequently, the estimates produced by the INS become more and more accurate.

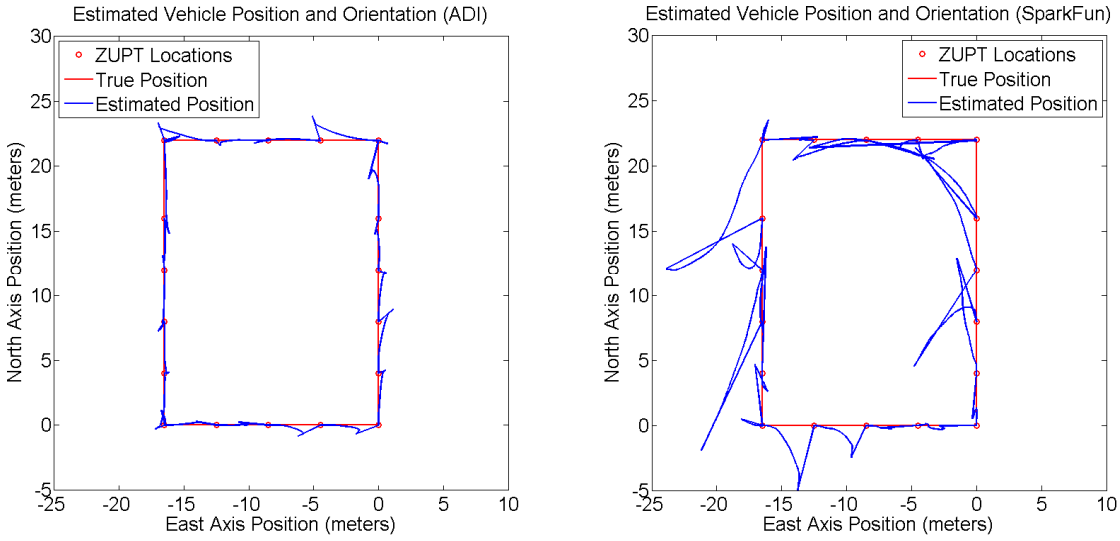

Figure 7.3: Estimated position and orientation of the vehicle versus true position and orientation of the vehicle (Inertial data, ZUPTs and position and orientation fixes).

Section 7.4 – SparkFun and ADI IMU Comparison

The results presented in this final subsection provide a direct comparison of the navigation results generated by the SparkFun-based INS and the ADI-based INS. For the purposes of this comparison, each INS had access to the inertial data from its respective IMU and all of the available secondary navigation information collected during the UGV navigation tests (i.e. ZUPTs, position fixes and orientation fixes).

The two INS units are compared in two ways. The first comparison is a visual comparison of how well each INS was able to track the linear velocity, position and orientation of the vehicle over time as it drove around the predefined test course. The second comparison is a numeric comparison of the estimation errors from each of the INS units (i.e. the differences between the estimated linear velocity, position and orientation from the INS units and the true linear velocity, position and orientation of the vehicle over time). These comparisons are shown in Figures 7.4 through 7.7. Figure 7.4 displays the estimated position and orientation of the vehicle over time from each of the INS units on the same graph versus the true position and orientation of the vehicle over time. Figure 7.5 displays the estimated linear velocity of the vehicle over time from each of the INS units on the same graph versus the target velocities for the vehicle over time. And finally, Figures 7.6 and 7.7 display the errors between the linear velocity, position and orientation estimates produced by each of the INS units and the true linear velocity, position and orientation of the vehicle over time.

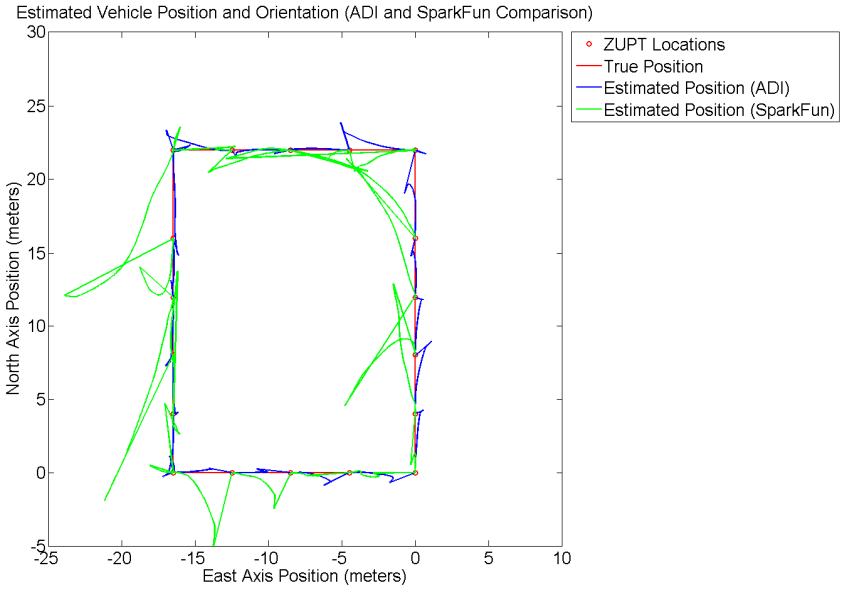

Figure 7.4: A visual comparison of the position and orientation estimates produced by the SparkFun-based and ADI-based INS units.

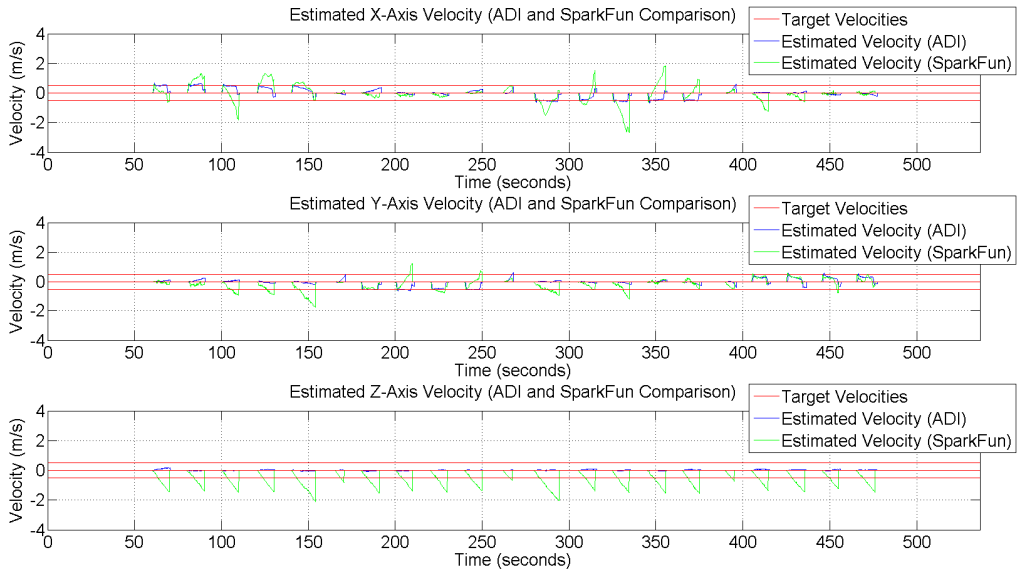

Figure 7.5: A visual comparison of the linear velocity estimates produced by the SparkFun-based and ADI-based INS units.

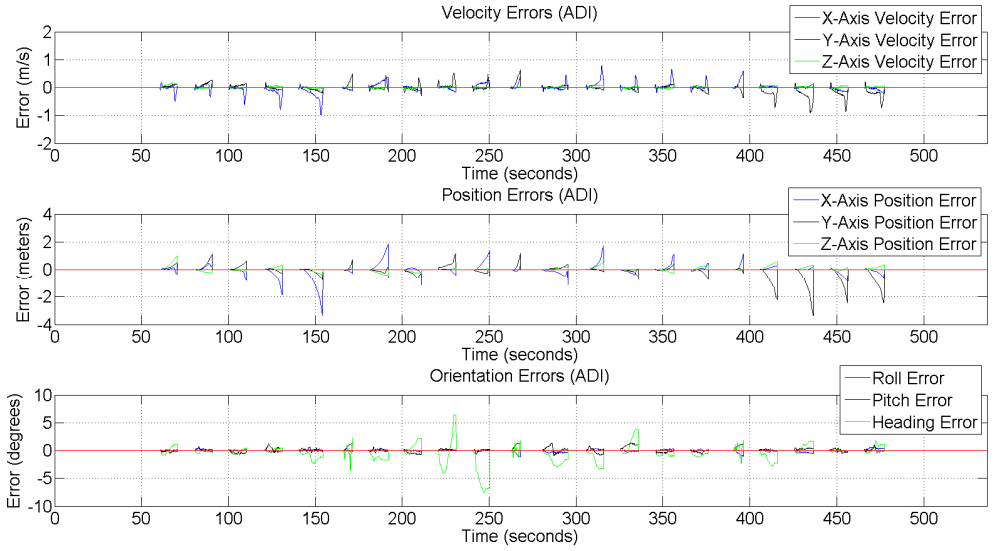

Figure 7.6: Errors in the estimated linear velocity, position and orientation of the vehicle over time (ADI).

As Figures 7.4 through 7.7 show, the navigation data from the ADI INS is clearly much more accurate than the navigation data from the SparkFun INS is in terms of both the visual comparison and the numerical comparison. This result is intuitively satisfying given the cost differential between the two IMUs, and it supports the common-sense hypothesis that as the cost per unit of a MEMS IMU increases, the quality of the IMU and of the inertial data that it returns increases proportionally.

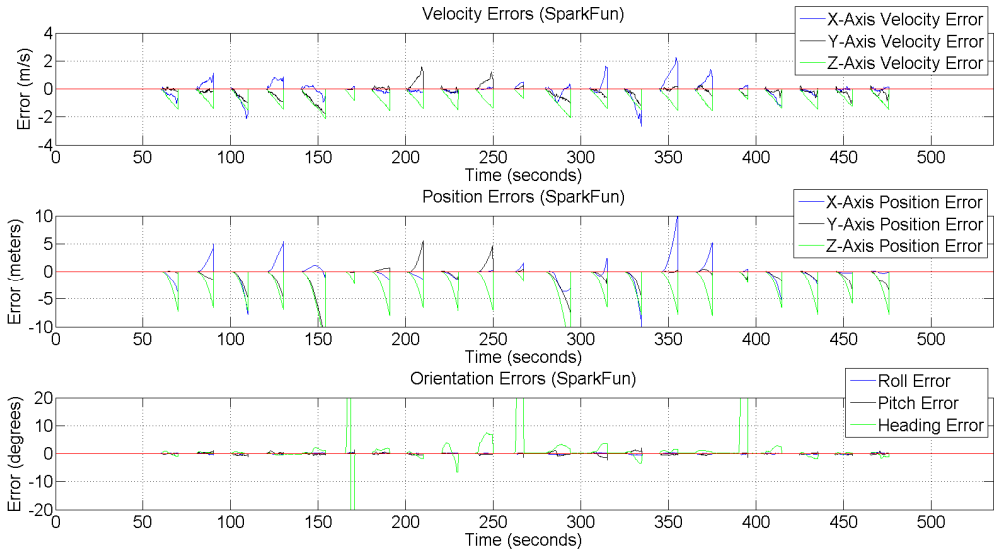

Figure 7.7: Errors in the estimated linear velocity, position and orientation of the vehicle over time (SparkFun).

The primary conclusion to take away from these results is that the analysis and modeling of typical MEMS IMU error sources did make the inertial data from the SparkFun IMU better overall, and the performance gap between the SparkFun and ADI IMUs did decrease, but the inertial data from the SparkFun IMU is still not as high-quality as the inertial data from the ADI IMU is. The inertial data from the ADI IMU is still much better overall, due largely in part to how the internal mechanisms of the two IMUs are set up. While both the SparkFun and ADI IMUs are constructed using MEMS technology, the ADI IMU has several sophisticated internal mechanisms (temperature calibration, low-pass filtering, etc.) that operate on the inertial data before it is received by the user. By comparison, the SparkFun IMU is much simpler and contains less sophisticated internal mechanisms for preprocessing the raw data from its inertial sensors. Since the inertial data from the ADI IMU is already cleaned up to some extent, it is inherently more accurate than the inertial data from the SparkFun IMU is. Because of this, the ADI IMU has a distinct advantage over the SparkFun IMU in terms of producing accurate navigation information.

Furthermore, because each of the INS units uses the exact same secondary navigation information to calculate their EKF correction terms, any differences in the final navigation results are entirely related to the quality of the inertial data produced by the IMUs. Therefore, the suitability of each IMU for navigation applications can be assessed by observing which INS produced the better overall navigation results. And given the results displayed in Figures 7.4 through 7.7, it can be concluded that the ADI IMU definitely produces better overall inertial data than the SparkFun IMU does, and is therefore much better suited for inertial navigation applications than the SparkFun IMU is.

Section 8 – Future Work

This section lists several new ideas that could potentially be implemented in future iterations of this research to either improve the quality of the results obtained from the IMU calibration experiments or the inertial navigation algorithm or add a new component or functionality to the existing research that has already been accomplished.

- Perform unit to unit testing. Testing multiple different IMUs of the same model number would help to determine how widely errors vary in between different units. Unit to unit testing would determine whether or not each individual IMU must be separately analyzed or if the results from a single analysis would be enough to model the errors for all IMUs of a specific model number.
- Automate the IMU laboratory experiments. Many of the experiments used to estimate the IMU error parameters require the same action to be performed over and over again. These experiments could be greatly improved and simplified by automating the process.
- Analyze the performance of the EKF in more depth. There are many aspects of Kalman Filtering not touched upon in this thesis, and they could be explored in more detail to further improve the performance of the EKF implemented in the inertial navigation algorithm. Some examples include analyzing the estimation-error covariance matrices (P_k) and analyzing the EKF innovations ($y_k - H\hat{x}_k^-$).
- Find a new testing location and/or testing vehicle. Using a six-wheeled, skid-steered vehicle on a high-friction carpeted surface doesn't make for the best testing environment. In future iterations of this research or similar research, it would make sense to find a vehicle with a smoother steering mechanism as well as a testing location with a lower coefficient of friction. This will hopefully result in smoother vehicle motions during the duration of navigation testing.
- Improve the test course. The test course that was used for the navigation tests performed in this thesis proved to be adequate. However, in order to test navigation performance with a higher degree of accuracy, a more structured test course with a higher degree of overall precision would be required. An improved method for gathering secondary navigation information for the inertial navigation algorithm would also be required.
- Use real secondary navigation information. Secondary navigation information could be obtained much more frequently from a real sensor system (GPS, stereo camera systems, etc.). More frequent secondary navigation information would lead to more frequent EKF correction terms, which in turn would greatly increase the accuracy of the navigation testing results.

Section 9 – Conclusion

The goal of this thesis was to design, implement and test two separate INS units based around two low-cost MEMS IMUs. The results of the testing procedures were then used to determine how well each individual INS performed and which IMU is better suited for implementation in an INS. To achieve this goal, each IMU was first analyzed and modeled using a series of laboratory experiments designed to isolate and characterize the behavior of several well-documented types of MEMS IMU errors. Then, an inertial navigation algorithm was designed that incorporated the information and error models produced by the laboratory experiments. The inertial navigation algorithm was tested by rigidly mounting the two IMUs to a UGV and then driving the UGV via remote control through a predefined test course. The inertial information recorded from each IMU during the navigation tests was then post processed using the inertial navigation algorithm. Finally, the resulting navigation information was analyzed and compared to the true linear velocity, position and orientation of the UGV to determine which INS was able to more accurately track the linear velocity, position and orientation of the UGV over time.

There are several primary results to take away from this thesis. The first result is that the analysis and modeling of typical MEMS IMU error sources did make the inertial data from the SparkFun IMU better overall, but it was not enough to bring the SparkFun IMU up to the performance level of the ADI IMU. The ADI IMU proved to ultimately be much better for inertial navigation applications than the SparkFun IMU was. The second result is that while inertial data is the primary component of an INS, secondary navigation information plays an equally important role in the overall success of the INS. As secondary navigation information is obtained more frequently, the results from the INS will improve proportionally. The third and final result is that if high quality inertial data is essential to the success of a system, the best way to obtain that data is most likely to invest in a high quality IMU with internal mechanisms that preprocess the raw data from the inertial sensors. Low-cost, hobbyist-grade IMUs such as the SparkFun IMU are better suited for applications where high quality inertial data is not essential, such as impulse recognition, sensor fusion with a second type of sensor, wheel slippage detection, unintended vehicle motion detection and self-leveling or orientation detection.

Overall, this thesis showed that it is possible to analyze and model several types of common MEMS IMU errors and design an INS that uses those error models along with inertial data and secondary navigation information to estimate the linear velocity, position and orientation of a UGV over time to within a certain degree of accuracy. However, in order for the INS to be successful, the IMU supplying the inertial data to the system must be high quality enough to produce accurate measurements of the accelerations and angular velocities that the vehicle experiences over time.

Appendix

The Appendix displays all of the results generated by both the SparkFun-based and ADI-based INS units for all three of the navigation testing trials. Both of the INS units were allowed to use the inertial data from their respective IMUs and all of the secondary navigation information that was available to them (i.e. ZUPTs, position fixes and orientation fixes). Each of the following subsections displays the full results from a specific UGV navigation testing trial. The complete results include graphs displaying the following information:

- The estimated linear velocity of the vehicle versus the desired target velocities for the vehicle
- The estimated position and orientation of the vehicle versus the true position and orientation of the vehicle
- The errors between the linear velocity, position and orientation estimates produced by the two INS units and the true linear velocity, position and orientation of the vehicle over time

Section A.1 – Full Results of UGV Navigation Testing (Trial #1)

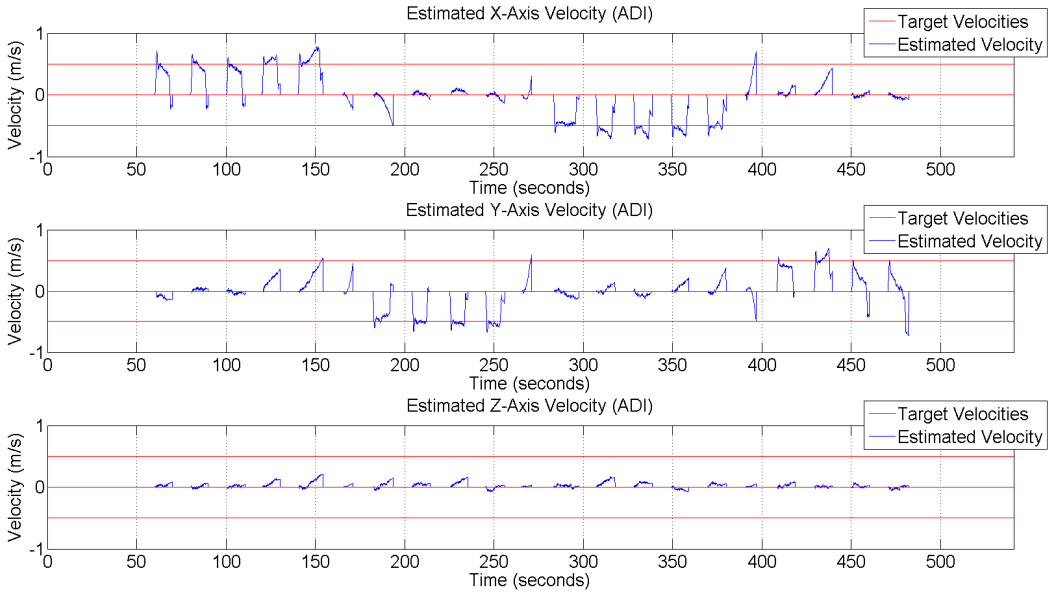

Figure A.1: Estimated linear velocity of the vehicle versus target velocities (ADI).

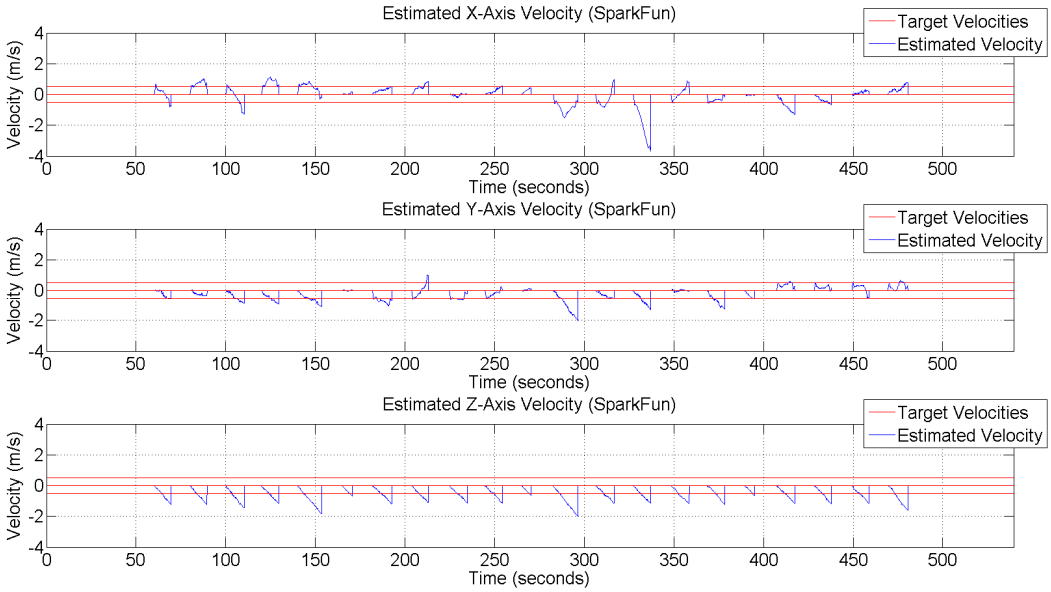

Figure A.2: Estimated linear velocity of the vehicle versus target velocities (SparkFun).

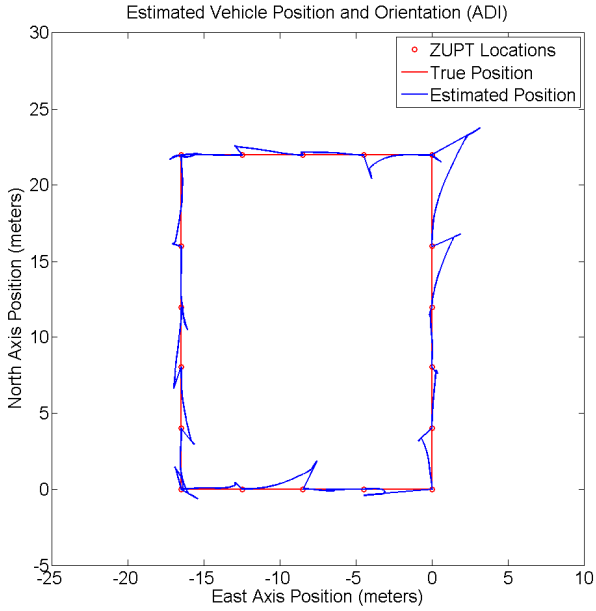

Figure A.3: Estimated position and orientation of the vehicle versus true position and orientation of the vehicle (ADI).

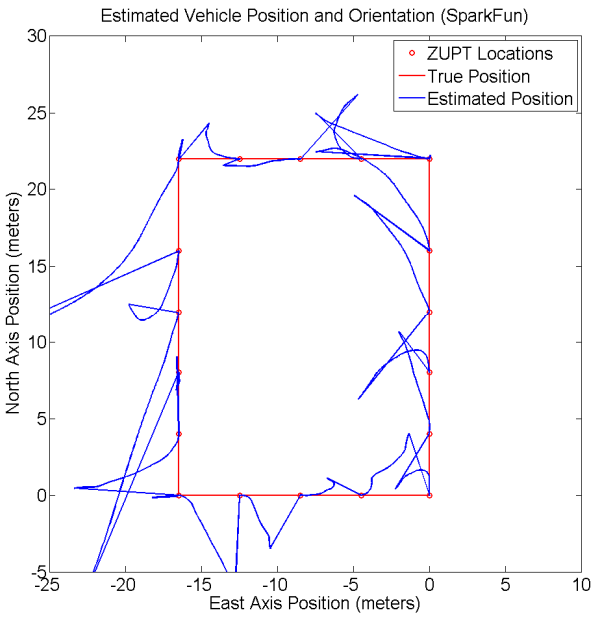

Figure A.4: Estimated position and orientation of the vehicle versus true position and orientation of the vehicle (SparkFun).

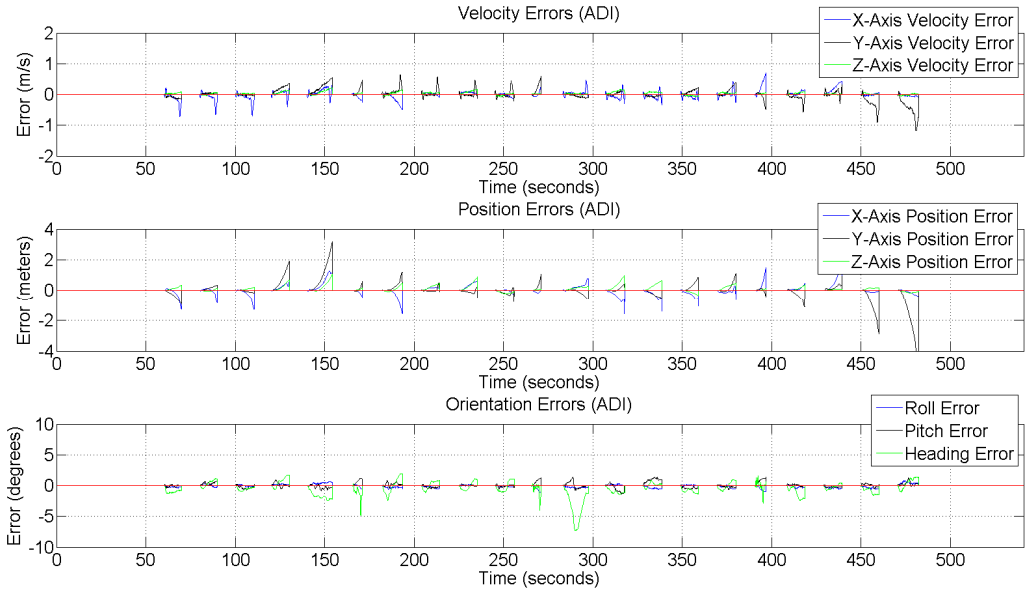

Figure A.5: Linear velocity, position and orientation estimation errors over time (ADI).

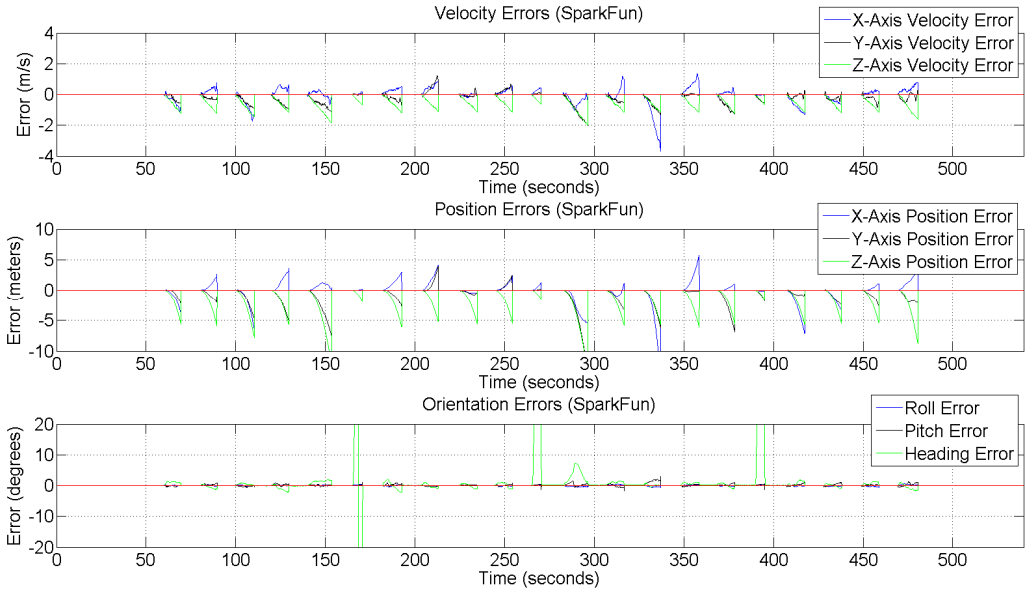

Figure A.6: Linear velocity, position and orientation estimation errors over time (SparkFun).

Section A.2 – Full Results of UGV Navigation Testing (Trial #2)

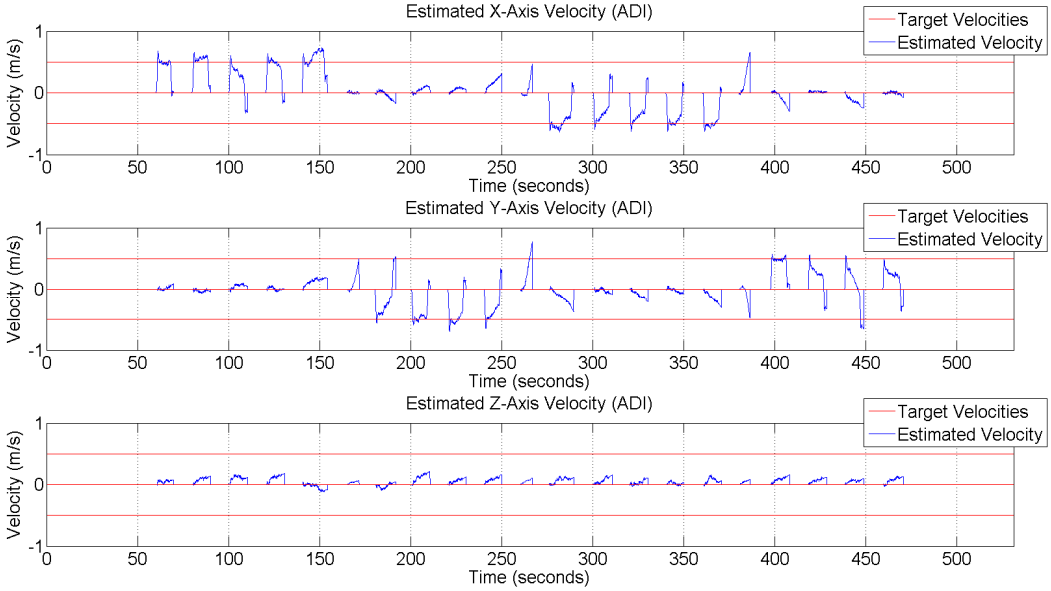

Figure A.7: Estimated linear velocity of the vehicle versus target velocities (ADI).

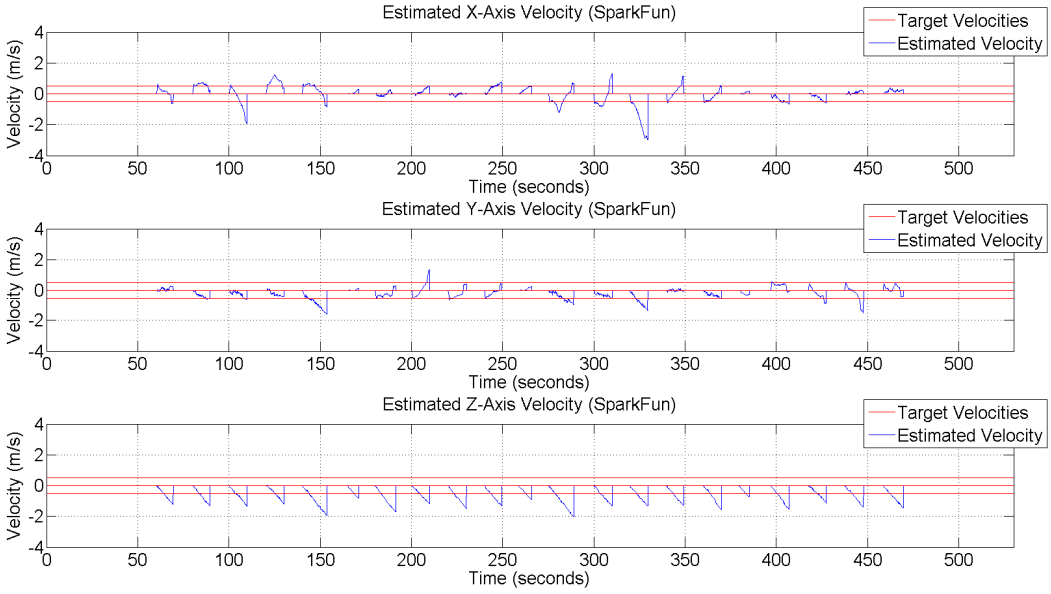

Figure A.8: Estimated linear velocity of the vehicle versus target velocities (SparkFun).

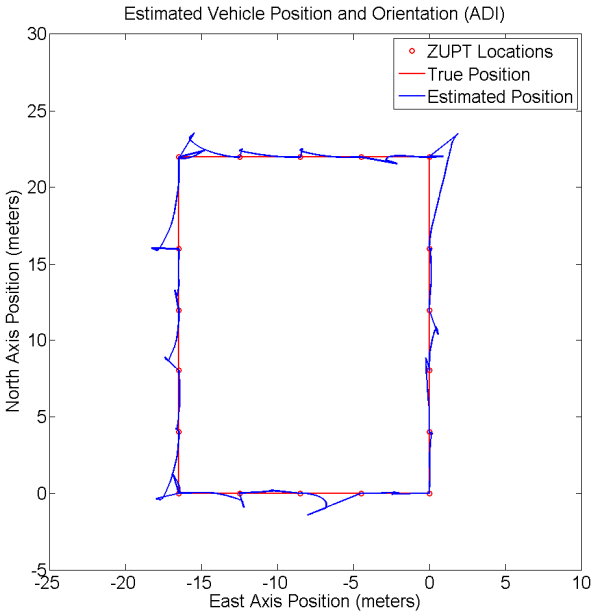

Figure A.9: Estimated position and orientation of the vehicle versus true position and orientation of the vehicle (ADI).

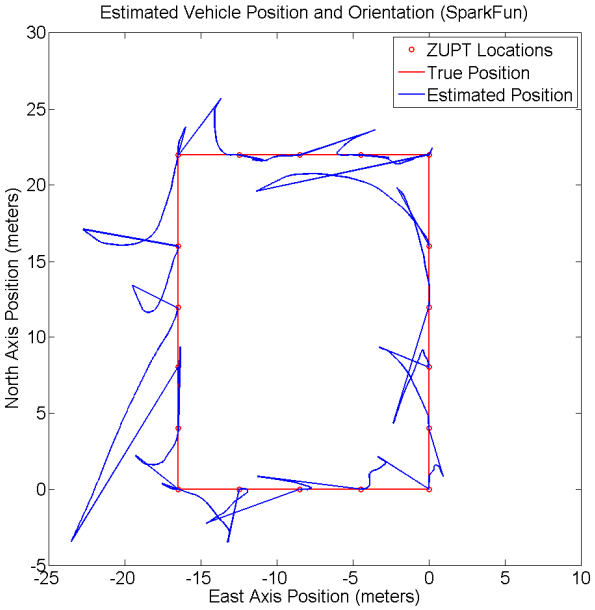

Figure A.10: Estimated position and orientation of the vehicle versus true position and orientation of the vehicle (SparkFun).

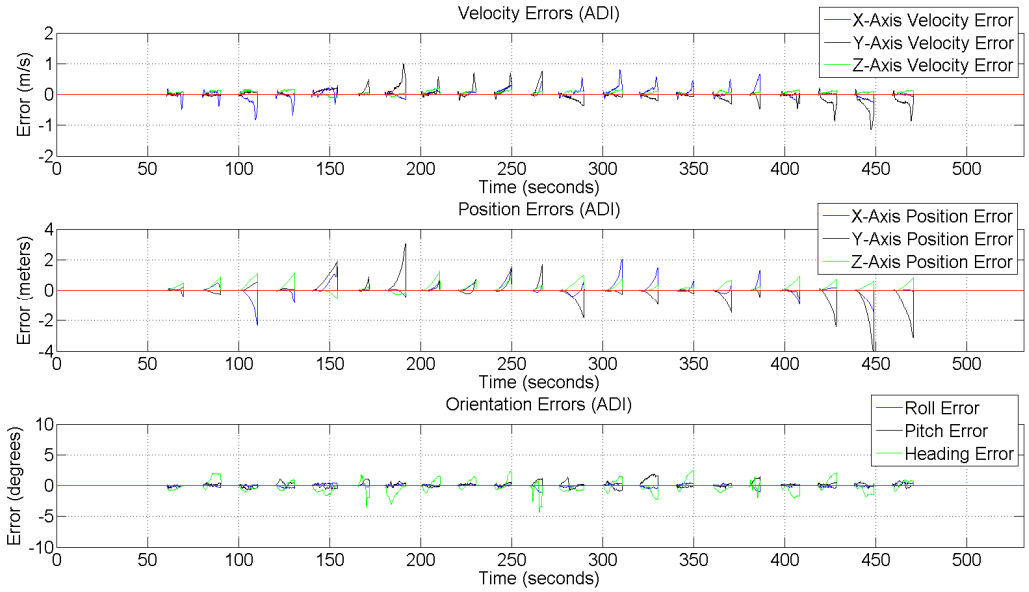

Figure A.11: Linear velocity, position and orientation estimation errors over time (ADI).

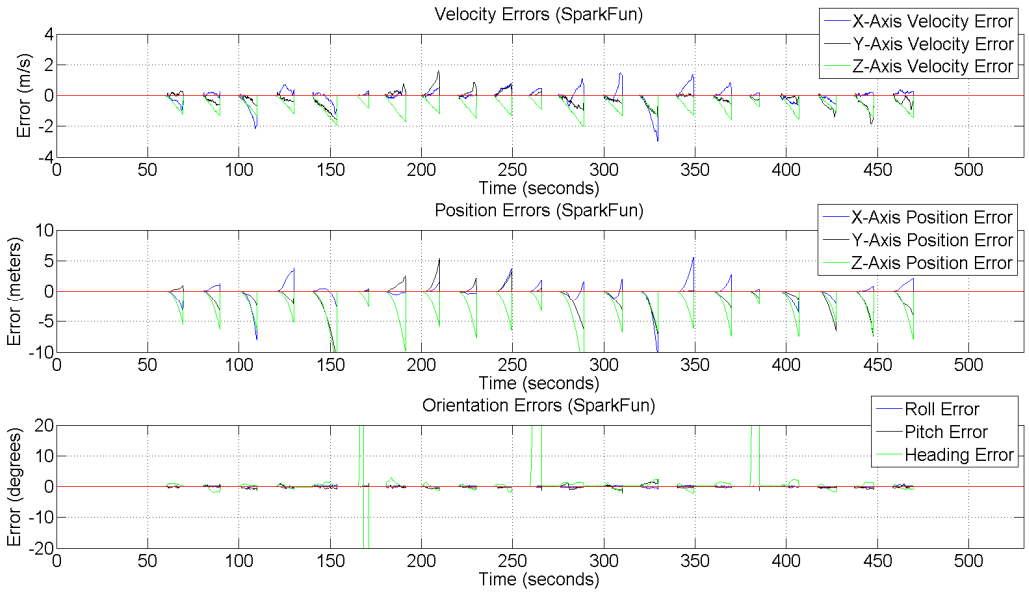

Figure A.12: Linear velocity, position and orientation estimation errors over time (SparkFun).

Section A.3 – Full Results of UGV Navigation Testing (Trial #3)

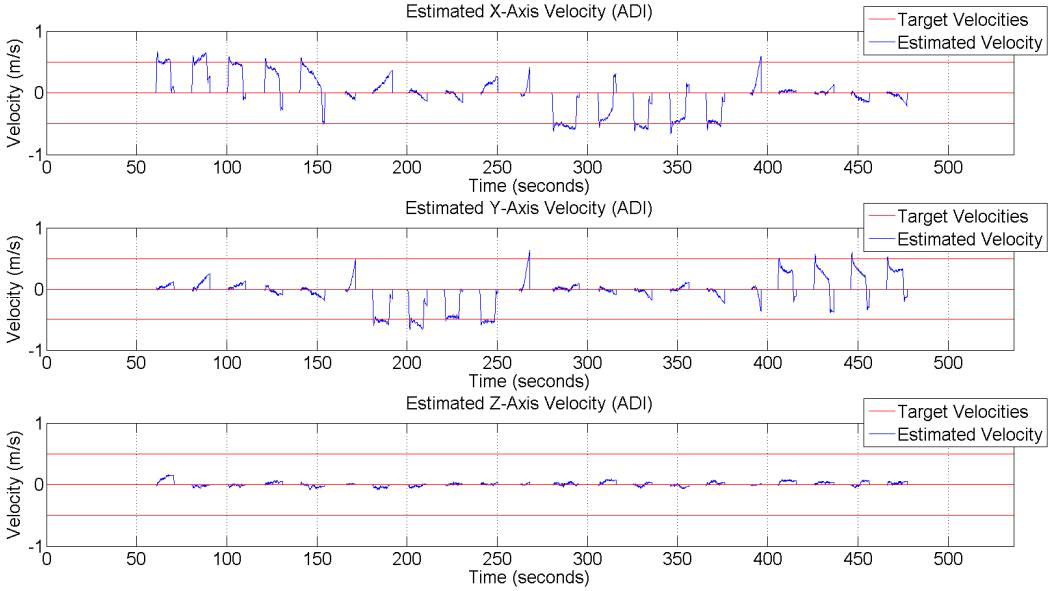

Figure A.13: Estimated linear velocity of the vehicle versus target velocities (ADI).

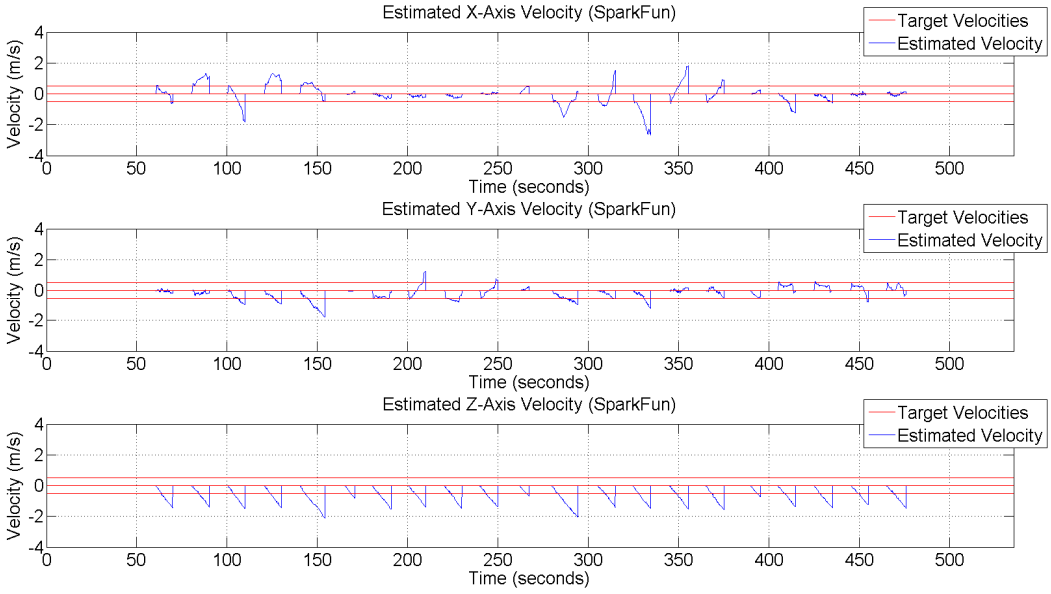

Figure A.14: Estimated linear velocity of the vehicle versus target velocities (SparkFun).

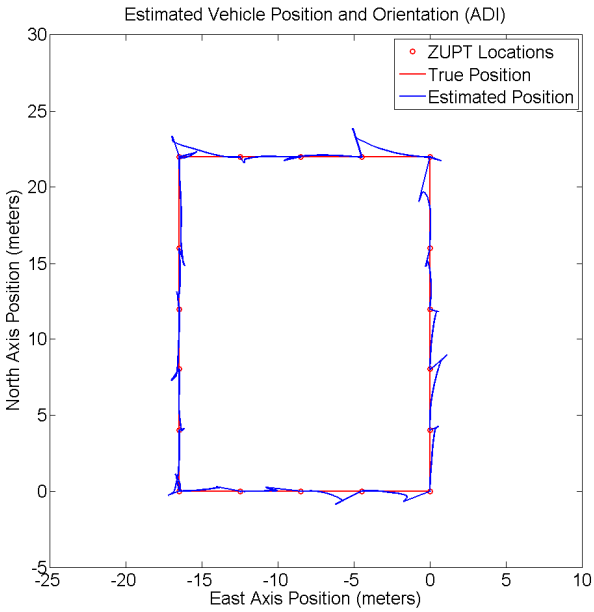

Figure A.15: Estimated position and orientation of the vehicle versus true position and orientation of the vehicle (ADI).

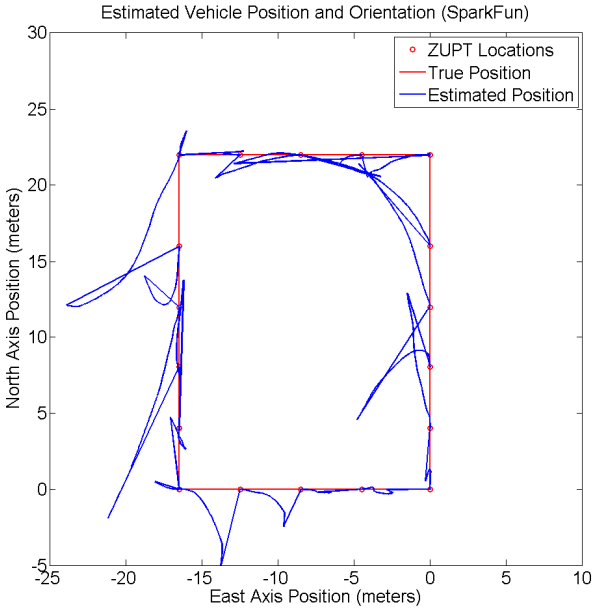

Figure A.16: Estimated position and orientation of the vehicle versus true position and orientation of the vehicle (SparkFun).

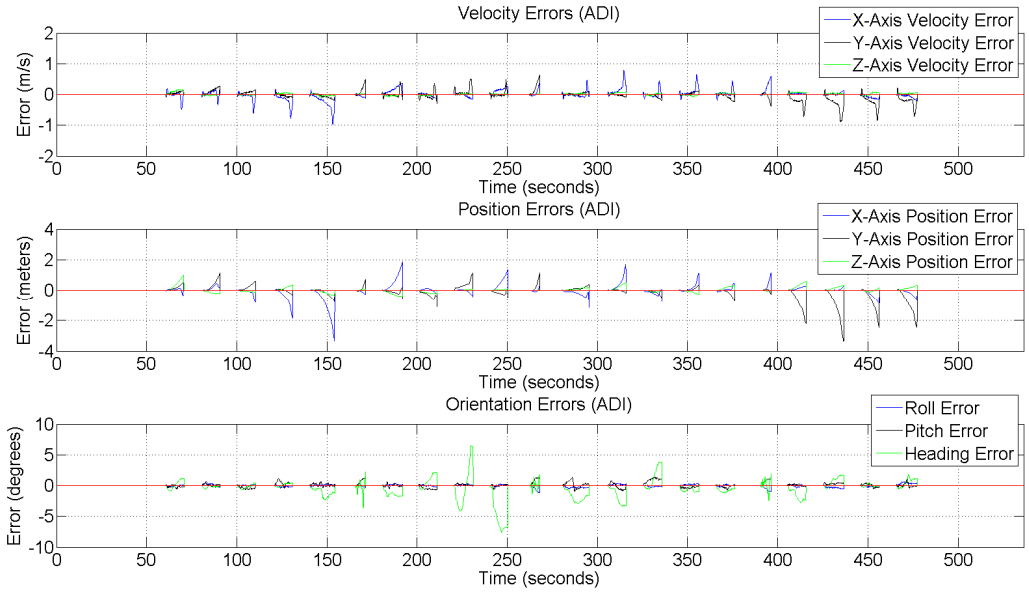

Figure A.17: Linear velocity, position and orientation estimation errors over time (ADI).

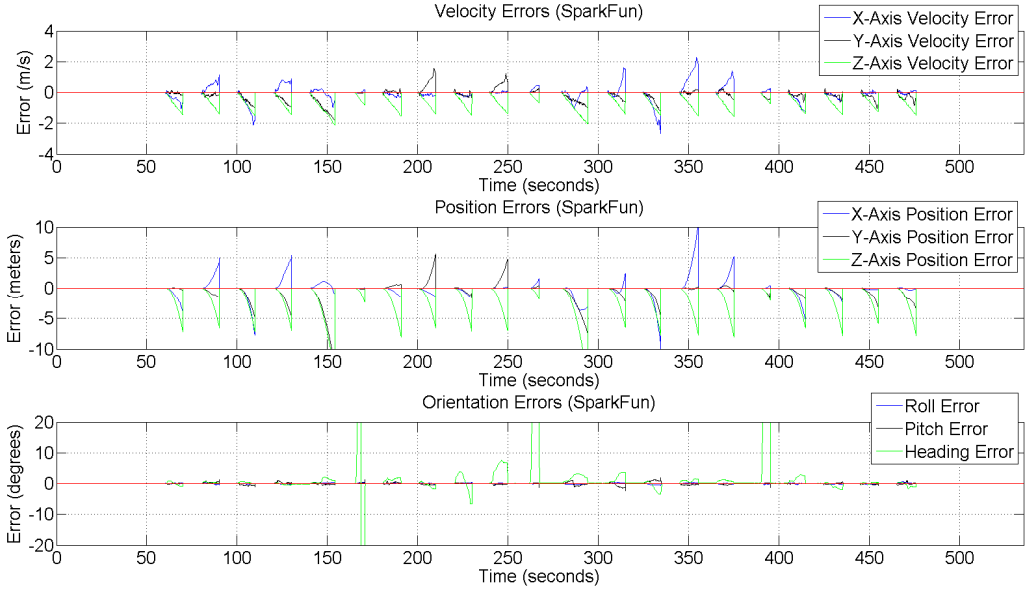

Figure A.18: Linear velocity, position and orientation estimation errors over time (SparkFun).

References

[1] Analog Devices, Inc., "3-Axis, ±2g/±4g/±8g/±16g Digital Accelerometer," ADXL345 datasheet, 2009.

[2] InvenSense, Inc., "ITG-3200 Product Specification Revision 1.4," ITG-3200 datasheet, March 2010.

[3] Analog Devices, Inc., "Six Degrees of Freedom Inertial Sensor," ADIS16364 datasheet, 2009-2011.

[4] J. C. Hung, J. R. Thacher, H. V. White, "Calibration of Accelerometer Triad of an IMU With Drifting Z-Accelerometer Bias," In Proc. IEEE 1989 National Aerospace and Electronics Conference, 1989, pp. 153-158 Vol. 1.

[5] L. Zhang et al., "Research on Auto Compensation Technique of Strap-Down Inertial Navigation Systems," International Asia Conference on Informatics in Control, Automation and Robotics, 2009, pp. 350-353.

[6] B. Hartmann, N. Link, G. F. Trommer, "Indoor 3D Position Estimation Using Low-Cost Inertial Sensors and Marker-Based Video-Tracking," 2010 IEEE/ION Position Location and Navigation Symposium (PLANS), 2010, pp. 319-326.

[7] L. Wang et al., "Thermal Calibration of MEMS Inertial Sensors for an FPGA-Based Navigation System," 3rd International Conference on Intelligent Networks and Intelligent Systems (ICINIS), 2010, pp. 139-143.

[8] D. Unsal, K. Demirbas, "Estimation of Deterministic and Stochastic IMU Error Parameters," 2012 IEEE/ION Position Location and Navigation Symposium (PLANS), 2012, pp. 862-868.

[9] J. J. Higgins, S. Keller-McNulty, *Concepts in Probability and Stochastic Modeling*. Boston, MA, 1995, pg. 330, Theorem 8.2-1.

[10] D. W. Allan, "Time and Frequency (Time-Domain) Characterization, Estimation, and Prediction of Precision Clocks and Oscillators," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 34, no. 6, pp. 647-654, November, 1987. [Online serial]. Available: http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=1539968&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D1539968. [Accessed 2011].

[11] O. J. Woodman, "An Introduction to Inertial Navigation," University of Cambridge Computer Laboratory, Cambridge, United Kingdom, Tech. Report. UCAM-CL-TR-696, Aug. 2007.

[12] P. D. Welch, "The Use of Fast Fourier Transform for the Estimation of Power Spectra: A Method Based on Time Averaging Over Short, Modified Periodograms," *IEEE Transactions on Audio and Electroacoustics*, vol. 15, no. 2, pp. 70-73, June, 1967. [Online serial]. Available:

http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=1161901&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D1161901. [Accessed 2012].

[13] S. Vitale et al., "Recommendation for an algorithm for Power Spectral Density estimation for LISA Pathfinder," Università Degli Studi Di Trento, Trento, Italy, Tech. Report. S2-UTN-TN-3040, Sept. 2006. [Online]. Available: http://www.rssd.esa.int/SP/LISAPATHFINDER/docs/Data_Analysis/PSD_Recipe.pdf. [Accessed 2012].

[14] Penn State. STAT 510. Class Lecture, Topic: "Applied Time Series Analysis." Department of Statistics, Eberly College of Science, Pennsylvania State University, University Park, State College, PA, 2013. [Online]. Available: https://onlinecourses.science.psu.edu/stat510/?q=node/60. [Accessed 2013].

[15] S. M. Kay, "Chapter 4 – Linear Models," in *Fundamentals of Statistical Signal Processing (Volume I: Estimation Theory)*. Upper Saddle River, NJ, 1993, pp-83-100.

[16] D. Titterton, J. L. Weston, "Strapdown Attitude Representations," in *Strapdown Inertial Navigation Technology, 2nd Edition*. IET, 2004, pp-36-44.

[17] W. Premerlani, P. Bizard, *Direction Cosine Matrix IMU: Theory*, May 2009.

[18] J. L. Center, Jr., *Integration for Local Tangent Navigation*, Autonomous Exploration, Inc., Aug. 2013.

[19] D. Simon, *Optimal State Estimation*. Hoboken, NJ, 2006.

[20] J. L. Center, Jr., *Information Filter Signal Flow*, Autonomous Exploration, Inc., 2011.