Design and Characterization of a Low Cost MEMS IMU Cluster for Precision Navigation

Daniel R. Greenheck
Marquette University

DESIGN AND CHARACTERIZATION OF A LOW COST MEMS IMU CLUSTER FOR PRECISION NAVIGATION

by

Daniel R. Greenheck, B.Sc.

A Thesis Submitted to the Faculty of the Graduate School
Marquette University,
in Partial Fulfillment of the Requirements for
the Degree of Master of Science

Milwaukee, Wisconsin

August 2015

ABSTRACT

DESIGN AND CHARACTERIZATION OF A LOW COST MEMS IMU CLUSTER FOR PRECISION NAVIGATION

Daniel R. Greenheck, B.Sc.

Marquette University, 2015

The fast paced development of micro-electromechanical systems (MEMS) technology in recent years has resulted in the availability of low cost gyroscopes and accelerometers in commercial markets. These sensors can be integrated into a single device known as an inertial measurement unit (IMU). An IMU is capable of tracking and navigating a vehicle for a short period of time in the absence of external position and attitude updates. The precision of the manufacturing techniques used to fabricate commercial MEMS sensors as well as their mechanical nature result in noise and errors that limit their performance. It has been mathematically shown that combining many MEMS sensors into a single device results in improved performance characteristics which are unattainable using a single MEMS sensor. The primary aim of this thesis is to design and validate the performance of a prototype IMU composed of a cluster of individual MEMS IMUs. The secondary aim of this thesis is to derive and validate a set of computationally inexpensive coning and sculling algorithms to mitigate dead-reckoning errors resulting from high frequency motion of the vehicle.

ACKNOWLEDGEMENTS

Daniel R. Greenheck, B.Sc.

First and foremost, I would like to thank my adviser Dr. Robert H. Bishop for providing me with the opportunity to undertake this research. Without his support and help, I would not have found my passion for the work I do. Secondly, I would like to thank my committee members for providing me with guidance during the entire process. I would like to thank Dr. Dean Jeutter especially for providing me with valuable insight into my circuit board design and assisting me in the fabrication process.

I would also like to express gratitude to my collaborators Dr. John Christian and Drew Bittner from West Virginia University for their willingness to answer my endless questions; it has been a pleasure working with them on this project.

Lastly– but certainly not least– I would like to thank my family and friends for their love, support and wisdom throughout my entire graduate experience. It was not without hardship and they have always stood by my side.

This work was made possible by NASA cooperative agreement NNX13AQ79A with Marquette University under subcontract to West Virginia University.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

ARW	**A**ngle **R**andom **W**alk
AWGN	**A**dditive **W**hite **G**aussian **N**oise
CDH	**C**ommand and **D**data **H**andling
DCM	**D**irection **C**osine **M**atrix
IMU	**I**nertial **M**easurement **U**nit
INS	**I**nertial **N**avigation **S**ystem
LEO	**L**ow **E**arth **O**rbit
LSB	**L**east **S**ignificant **B**it
MEMS	**M**icro **E**lectro-**M**echanical **S**ystem
MIMUC	**M**EMS **I**nertial **M**easurement **U**nit **C**luster
PCB	**P**rinted **C**ircuit **B**oard
PSD	**P**ower **S**pectral **D**ensity
RND	**R**ate **N**oise **D**ensity
SPI	**S**erial **P**eripheral **I**nterface
VRW	**V**elocity **R**andom **W**alk

Chapter 1

Introduction

1.1 Background and Motivation

Developments in micro electro-mechanical systems, or MEMS, technology within the last decade has resulted in significant advancements in sensor technology. MEMS sensors have the advantages of low power consumption and small mass, in addition to low cost as a result of high-volume production. One type of MEMS sensor that has benefited from these advancements is the inertial measurement unit, or IMU. A typical 6-axis IMU has a three-axis gyroscope and three-axis accelerometer integrated into a single package. The gyroscope measures the angular rates and the accelerometer measures non-gravitational accelerations, known as specific force. An important application of the IMU is vehicle navigation. In a strap-down navigation system under consideration here, the IMU is mounted directly to the chassis of the vehicle, rather than mounted to a gimbal. Gimbaled IMUs are mechanically much more complex, but the associated navigation algorithms are much simpler. However, with the advent of powerful and inexpensive embedded computers, the increased algorithmic complexity for strapdown navigation systems is not an issue. The outputs of the IMU can be utilized to determine the position of the vehicle at future times given a known initial position, velocity and attitude. This is done without the aid of external navigation sources, such as GPS. Navigation without external updates is referred to as *dead reckoning*. The objective is thesis is to utilize a cluster of MEMS IMUs to create a tactical grade equivalent IMU (at significantly reduced cost, power, and mass) to support as precise a dead reckoning navigation solution as possible without requiring external measurement updates. Applications include precise navigation of nanosatellites under thrusting conditions (where star tracker updates are difficult

to achieve) and automated rovers in GPS-denied environments (such as in dense urban settings).

The concept behind dead reckoning is straightforward. The accelerometer in the IMU provides a measure of the non-gravitational accelerations, also known as specific force, that can be integrated once (in an inertial reference frame) to determine the velocity and once more to determine the position relative to the starting location. However, if the vehicle frame in which the IMU is attached is a non-inertial rotating frame, the accelerations must first be transformed to an inertial reference frame for integration utilizing the gyroscopes that provide a measure of the angular rates. In this way, the accelerometers and gyros provide the necessary information to the dead reckoning navigation algorithm on the embedded computer.

In practice there are considerable challenges in obtaining a precise navigation solution with MEMS devices. All IMU sensors, regardless of technology or quality, exhibit both systematic errors and stochastic errors. Commercial-grade MEMS sensors in particular exhibit considerable bias drift and noise, making them unsuitable for precision navigation applications without significant calibration and additional architecture design modifications and algorithmic support. To illustrate why this is a problem, consider an accelerometer at rest with a constant bias error. Integrating to obtain the velocity results in an error that grows linearly over time. Integrating once more to introduces a quadratic error in position. Very small errors in the gyro or accelerometer outputs result in rapidly growing errors in velocity, position and attitude. To eliminate these errors, an external navigation update is required, such as GPS. Without external updates, these errors cannot be compensated for and eventually the error will be outside of the allowed threshold specified for a particular mission.

1.2 Objectives

The aim of this thesis is to develop a prototype IMU by incorporating many inexpensive MEMS IMUs on a single circuit board (referred to as a MEMS IMU Cluster, or MIMUC). The objective of this task is to design an IMU using inexpensive consumer grade MEMS IMUs suitable for precision navigation applications. By averaging many independent measurements of the angular rate and acceleration, the MIMUC will be able to provide a more accurate navigation solution for a longer period of time compared with just a single MEMS IMU. This navigation system is aimed at applications placed under strict power, mass and cost restrictions.

A secondary aim of this thesis is to propose a set of computationally efficient algorithms to counteract the effects of coning and sculling. Certain motions can result in the accumulation of errors in the navigation solution independent of the systematic and stochastic errors of the IMUs. These errors are a result of sampling the angular rate and accelerations at discrete times. If the vehicle is undergoing high frequency angular or linear accelerations, the IMUs may fail to completely quantify the motion. Naively increasing the update rate of the navigation algorithms may place an intolerable burden on the computational hardware. Previous work by Bortz [5] and Savage [30][31] develops a method that allows the gyros and accelerometers to be independently sampled at a much faster rate than the rate at which the navigation equations are updated. By doing so, high frequency motion can be more accurately tracked and errors resulting from coning and sculling motions can be mitigated. This thesis aims builds on that previous work by developing and simulating a set of algorithms that can be executed with limited computational resources.

1.3 Applications

The driving application for this research is the navigation, pointing and rendezvous of nanosatellites in Low Earth Orbit (LEO). Nanosatellites (also known as CubeSats) are a classification of satellites with the following restrictions: 1) a mass no greater than 1.33 kg, 2) a volume of 1000 cm^3 (a cube with side lengths of 10 cm), 3) power consumption under 1 W. Fig. 1.1 shows a typical nanosatellite frame. In addition, the available power for on-board electronics is inherently scarce due to the limited surface area for solar panels. These strict criteria have limited the use of precision IMUs in nanosatellites. For example, the Northrup Grumman LN200S, a space-rated tactical-grade IMU with a rich space heritage, has a mass of 0.75kg, volume of 574 cm^3, and nominal power consumption of 12 W [12]. The inclusion of an IMU of this type is not feasible for nanosatellites.

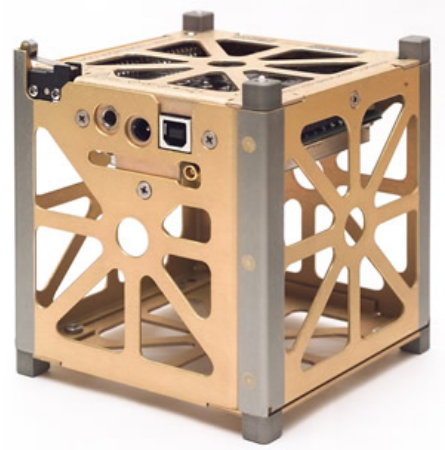

FIGURE 1.1: 1U CubeSat chassis [35]

These restrictions are not limited to nanosatellites; many other mobile vehicles such as UAVs (unmanned aerial vehicles) and autonomous submarines have strict requirements on power, mass and volume. The properties of MEMS sensors make them suitable for these applications: they are inexpensive, consume little power, and are available in small form factors. Despite these advantages, the noise characteristics of consumer-grade MEMS IMUs are too poor to be used in precision inertial navigation systems. It is the goal of this thesis to explore the

concept of creating an IMU consisting of a cluster of individual MEMS IMUs with decreased output noise compared with a single MEMS IMU.

The end goal of this technology is to enable nanosatellites to perform orbital maneuvers as well as rendezvous of multiple satellites in orbit. Rendezvous would allow the formation of swarms of nanosatellites in orbit. At the 2014 AIAA CubeSat Developers Conference, a team from NASA Ames Research center presented the concept of the EDSN Interstellar Communications Architecture [19]. The concept is to send multiple nanosatellites into orbit capable of creating a communications network on demand. A particular satellite is assigned to be the *captain* based on some fitness assessment. The rest of the satellites– referred to as *lieutenants*– relay their data to the captain at some regular interval. When the captain is in sight of the ground station, it will download all of the data it has collected from the lieutenants.

There are several advantages to this type of system. Single satellites are vulnerable to single-unit failures. As a result, redundant systems are incorporated into the satellite to mitigate these types of scenarios. This redundancy introduces extra cost and complexity. Satellite swarms do not need redundant systems because the failure of one or more satellites can be tolerated. The EDSN architecture outlines the procedures if the captain were to fail. The lieutenants would detect the failure of the captain and a new captain would be reassigned to another satellite.

1.4 Literature Review

The idea of integrating many MEMS IMUs into a single IMU cluster is not a new concept. Chang, et al. [6] investigated the concept of combining several gyroscopes to create a synthetic gyroscope with higher accuracy outputs. It was shown that the bias stability of such a device- with appropriate filtering- substantially improves the bias drift, an error measure tied closely with overall IMU

performance. Guerrier [18] explored the concept of improving the accuracy of navigation systems using redundant MEMS IMU sensors with integrated GPS. It was shown that such a configuration of sensors reduces the noise proportional to the square root of the number of sensors. Crain, et al. [12] extended this work by comparing the performance of a MEMS IMU cluster to a tactical, navigation and strategic grade IMU. The performance of a ten IMU cluster was compared with a navigation grade IMU during simulation of lunar entry return navigation. It was determined that although the navigation grade device exceeded the performance of the IMU cluster, the cluster offered considerable advantages in terms of power, volume and mass. Martin et al. [22] investigated several topics within the broader topic of MEMS IMU clusters such as the physical arrangement of sensors and improving performance by combining IMUs with differing dynamic ranges. Colomina, et al. [10] proposed using redundant MEMS IMUs to assess the quality of navigation parameters and determine estimates of the system noise in real-time. The effect of different geometries of sensor configurations was examined by Osman, et al. [27]. The data showed that an orthogonal configuration of three sensors had improved performance upon three sensors in a skew-symmetric configuration. Not enough data was provided to generalize this statement. Bancroft [3] presented results of using an IMU cluster to navigate a vehicle during 30 second GPS outages. The use of two, three, four and five redundant IMUs resulted in performance improvements of 25%, 29%, 32%, and 34%, respectively, compared with a single IMU.

The type of design proposed in this thesis has been attempted and tested before by Skog, et al. [33, 34] and Nilsson, et al. [24]. Skog, et al. [33] constructed a MIMUC using 18 individual MEMS IMUs. A novel communication interface was proposed that allowed data to be acquired from all IMUs simultaneously over an I2C bus. An error analysis based on static measurements showed an improvement roughly equivalent the square root of the number of sensors, supporting Guerrier's results. This system was proposed as a possible method of pedestrian tracking

by placing it in the sole of a shoe. However, a test of the system in Skog, et al. [34] revealed that simple averaging of the IMU outputs was not an optimal solution for pedestrian tracking; little improvement was seen in the dead reckoning performance of an IMU cluster versus a single MEMS sensor.

1.5 Contributions

Although the concept of a MEMS IMU cluster has been explored by previous researchers, to the author's knowledge it has not been applied to the area of satellite navigation. This thesis proposes a design for a prototype MEMS IMU cluster suitable for use in a nanosatellite. A circuit board containing a cluster of sixteen MEMS IMUs is proposed, as well as a set of computationally efficient algorithms for coning and sculling compensation. The performance of such as system is evaluated and compared with a single MEMS IMU. The coning and sculling algorithms are simulated in MATLAB evaluated with respect to standard trapezoidal integration.

1.6 Thesis Organization

Chapter 2 presents the mathematical foundation behind the error models used in both calibrating the MEMS IMUs as well as predicting the performance of the MEMS IMU cluster. In addition, a derivation for the coning and sculling algorithms is presented. Chapter 3 covers the both the hardware and software design of the MIMUC. The procedure for calibrating the MIMUC is also presented. Finally, a set of experiments are proposed to evaluate the performance of the MIMUC compared with a single MEMS IMU as well as a virtual MIMUC. In addition, a set of experiments for evaluating the performance of the coning and sculling algorithms are proposed. Chapter 4 presents the results of the experiments. Chapter 5 discusses the results presented in Chapter 4. The MIMUC is also compared with several other IMUs to give some notion of the performance

of the MIMUC relative to existing technology. Chapter 6 summarizes the results within this thesis, as well as examining future directions for this work.

Chapter 2

Mathematical Foundations

2.1 IMU Error Models

2.1.1 Background

Consumer grade MEMS IMUs exhibit a variety of stochastic and deterministic errors. The stochastic errors can be handle through a variety of filtering techniques which will not be discussed in this thesis. The deterministic errors can be accounted for through calibration. Although the IMUs are pre-calibrated at the factory, the calibration is not accurate enough for precision navigation applications. Detailed testing and characterization of each individual sensor is a costly and time-consuming process, therefore making it prohibitive for high volume production.

Proper calibration requires an understanding of the different errors that effect the output of both the gyroscope and accelerometer. These error sources have been extensively investigated in literature. The IEEE Standards [25, 26] provide complete error models for Coriolis Vibratory Gyros and Non-Gyroscopic Linear Accelerometers (the type of sensors used in the MIMUC). The full error models include several dozen error terms. Including all of these terms is impractical due to limited laboratory equipment and computational power available to apply the corrections in real-time on the MIMUC microcontroller. The error models used in the characterization of the MIMUC have been truncated to include the most significant error terms as identified in existing literature. Both Ramalingam, et al. [29], Titterton [39], and Flenniken, et al. [14] were used as a reference for the deterministic components of the error model. Gebre-Egziabher [16] was used a reference for the stochastic components of the error model.

2.1.2 Gyroscope Error Model

The measured output of the rate gyro $K_g\tilde{\boldsymbol{\omega}}$ is given by

$$K_g\tilde{\boldsymbol{\omega}} = (\boldsymbol{I} + \boldsymbol{S}_g + \boldsymbol{M}_g)\boldsymbol{\omega} + \boldsymbol{b}_g + \boldsymbol{G}_g\boldsymbol{a} + \boldsymbol{T}_g\Delta T + \boldsymbol{\eta}_g + \boldsymbol{\epsilon}_g \tag{2.1}$$

where $\tilde{\boldsymbol{\omega}}$ is the digital output of the gyros (LSB), $\boldsymbol{\omega}$ is the true angular rate (deg/s), $\boldsymbol{a}$ is the true specific force (g), K_g is the gyro sensitivity scale factor (deg/s/LSB), $\boldsymbol{b}_g$ is the gyro bias error (deg/s), $\boldsymbol{S}_g$ is the gyro scale factor matrix (Unitless) defined as

$$\boldsymbol{S}_g = \begin{pmatrix} S_{g,x} & 0 & 0 \\ 0 & S_{g,y} & 0 \\ 0 & 0 & S_{g,z} \end{pmatrix},$$

$\boldsymbol{M}_g$ is the skew-symmetric gyro cross-coupling matrix (Unitless) defined as

$$\boldsymbol{M}_g = \begin{pmatrix} 0 & M_{g,xy} & M_{g,xz} \\ -M_{g,xy} & 0 & M_{g,yz} \\ -M_{g,xz} & -M_{g,yz} & 0 \end{pmatrix},$$

$\boldsymbol{T}_g$ is the gyro temperature coefficient (deg/s/°C), ΔT is the difference between the measured temperature and the temperature the sensors were calibrated at (°C), $\boldsymbol{G}_g$ is the g-sensitivity matrix (deg/s/g) defined as

$$\boldsymbol{G}_g = \begin{pmatrix} G_{g,xx} & G_{g,xy} & G_{g,xz} \\ G_{g,yx} & G_{g,yy} & G_{g,yz} \\ G_{g,zx} & G_{g,zy} & G_{g,zz} \end{pmatrix},$$

$\boldsymbol{\eta}_g$ is zero-mean white noise process (deg/s), $\boldsymbol{\epsilon}_g$ is a random walk sequence driven by a white noise process (deg/s). The inclusion of the sensitivity scale factor K_g is necessary to convert the gyro digital output from a signed 16 bit integer to deg/s.

This value is found in the manufacturer datasheet [20].

The MIMUC on-board calibration algorithms require solving for $\boldsymbol{\omega}$ since this quantity is used in the navigation algorithms. Analysis of the deterministic terms in Eq. (2.1) results in the following expression

$$\boldsymbol{\omega} = (\boldsymbol{I} + \boldsymbol{S}_g + \boldsymbol{M}_g)^{-1}(K_g\tilde{\boldsymbol{\omega}} - \boldsymbol{b}_g - \boldsymbol{G}_g\boldsymbol{a} - \boldsymbol{T}\Delta T) \tag{2.2}$$

The inversion of the scale factor/cross-coupling error matrix is computed offline to minimize the calculations required by the MIMUC.

2.1.3 Accelerometer Error Model

The measured output of the accelerometer $K_a\tilde{\boldsymbol{a}}$ is given by

$$K_a\tilde{\boldsymbol{a}} = (\boldsymbol{I} + \boldsymbol{S}_a + \boldsymbol{M}_a)\boldsymbol{a} + \boldsymbol{b}_a + \boldsymbol{T}_a\Delta T + \boldsymbol{\eta}_a + \boldsymbol{\epsilon}_a \tag{2.3}$$

where $\tilde{\boldsymbol{a}}$ is the digital output of the accelerometers (LSB), $\boldsymbol{a}$ is the true specific force (g), K_a is the accelerometer sensitivity scale factor (g/LSB), $\boldsymbol{b}_a$ is the accelerometer bias error (g), $\boldsymbol{S}_a$ is the accelerometer scale factor matrix (Unitless) defined as

$$\boldsymbol{S}_a = \begin{pmatrix} S_{a,x} & 0 & 0 \\ 0 & S_{a,y} & 0 \\ 0 & 0 & S_{a,z} \end{pmatrix},$$

$\boldsymbol{M}_a$ is the skew-symmetric accelerometer cross-coupling matrix (Unitless) defined as

$$\boldsymbol{M}_a = \begin{pmatrix} 0 & M_{a,xy} & M_{a,xz} \\ -M_{a,xy} & 0 & M_{a,yz} \\ -M_{a,xz} & -M_{a,yz} & 0 \end{pmatrix},$$

$\boldsymbol{T}_a$ is the accelerometer temperature coefficient (g/°C), ΔT is the difference between the measured temperature and the temperature the sensors were calibrated at (°C), $\boldsymbol{\eta}_a$ is zero-mean white noise process (deg/s), $\boldsymbol{\epsilon}_a$ is a random walk sequence driven by a white noise process (deg/s). The inclusion of the sensitivity scale factor K_a is necessary to convert the accelerometer digital output from a signed 16 bit integer to g. This value is found in the manufacturer datasheet [20].

The MIMUC on-board calibration algorithms require solving for $\boldsymbol{a}$ since this quantity is used in the navigation algorithms. Analysis of the deterministic terms in Eq. (2.3) results in the following expression

$$\boldsymbol{a} = (\boldsymbol{I} + \boldsymbol{S}_a + \boldsymbol{M}_a)^{-1}(K_a\tilde{\boldsymbol{a}} - \boldsymbol{b}_a - \boldsymbol{T}\Delta T) \tag{2.4}$$

The inversion of the scale factor/cross-coupling error matrix is computed offline to minimize the calculations required by the MIMUC.

2.1.4 Sources of Error

2.1.4.1 Calibration Errors

Calibration errors result from inaccurate measuring or estimating of the bias, scale factor, and misalignment errors of a gyroscope/accelerometer. These errors are due to imperfections in the manufacturing process and flaws of the sensor design itself. The purpose of calibration is to measure or estimate these errors and remove them. Bias error is the deviation of the gyro/accelerometer output from zero when no input is applied. Scale factor errors result from improperly measuring the linear input-output relationship of the sensor. It is also possible that input-output relationship is non-linear. The error model used in this thesis does not account for any non-linearities. The MPU-6000 datasheet specifies a non-linearity of 0.2% over the gyro full scale range and 0.5% over the accelerometer full scale range. Misalignment errors result from flaws in the manufacturing process. The

axes of the gyro/accelerometer may not be exactly orthogonal with one another, or are misaligned with respect to the package. The MPU-6000 product sheet specifies a cross-axis sensitivity of ±2% for both the gyros and accelerometers [20].

These errors are manifested as constant bias errors in the calibrated IMU output [41]. Integration of a constant bias in angular rate results in an attitude error that grows linearly over time. Likewise, a bias in the specific force measurement results in a linear error in velocity and a quadratic error in position.

2.1.4.2 Temperature Dependent Bias

One disadvantage of MEMS IMUs is that they are highly sensitive to changes in temperature [29]. The MPU-6000 includes an internal temperature sensor, allowing this type of error to be accounted for through calibration. The MPU-6000 data sheet specifies that the gyro scale factor typically varies by ±2% and the zero rate output varies by ±20 deg/s over the temperature range -40°C to +85°C. The accelerometer scale factor varies by ±0.02%/°C and the zero force output varies by ±35 mg for the X and Y axes and ±70 mg for the Z axis over the same temperature range [20]. The model used in this thesis only accounts for linear effects of the temperature on the bias.

2.1.4.3 Gyroscope g-Dependent Bias

The MPU-6000 gyro senses angular rotations by measuring vibrations induced by the Coriolis force with a capacitive pickoff [20]. Although care is taken in the sensor design to reject non-Coriolis forces, the gyros still exhibit some sensitivity to linear accelerations. The MPU-6000 datasheet specifies the sensitivity of the gyro output to linear accelerations as 0.1 deg/s/g [20]. The exact dependence can be found by applying different linear accelerations to the gyro (e.g. gravity) and measuring the relationship between the two. During operation, the specific

force measurement from the accelerometers is used in the gyroscope calibration to cancel out this effect.

2.1.4.4 Random Noise

The output of MEMS gyros and accelerometers exhibit random noise. The main source of this noise is thermo-mechanical fluctuations within the sensor [41]. This noise is generally modeled as additive white Gaussian noise (AWGN) with zero-mean and variance σ_w^2 [14]. The MPU-6000 datasheet specifies the rate noise density (RND) of the gyro as 0.005 deg/s/$\sqrt{\text{Hz}}$ and the power spectral density (PSD) of the accelerometer as 400 µg/$\sqrt{\text{Hz}}$ [20]. It is helpful to express the noise density in terms more applicable to inertial navigation to facilitate comparison with other IMUs.

A common measurement used to quantify the random noise of a sensor is angle random walk (ARW) for gyros, expressed in units of deg/$\sqrt{\text{hr}}$, and velocity random walk (VRW) for accelerometers, expressed in units of m/s/$\sqrt{\text{hr}}$. To illustrate the meaning of these terms, consider the following example. Using the gyroscope as an example, it is often of interest to determine how random noise present in the angular rate measurement will affect the attitude measurement. The attitude angles are found by integrating the gyro output. Integrating zero-mean AWGN results in a random walk in the angle. The expected standard deviation of the random walk at time t can be found by multiplying the ARW by $\sqrt{t}$. If a gyro has an ARW of 0.1 deg/$\sqrt{\text{hr}}$, the integrated angular rate will be a random variable with zero-mean and standard deviation of $0.1\sqrt{t}$ deg at time t. Similarly, if an accelerometer has a VRW of 1 m/s/$\sqrt{\text{hr}}$, the velocity will be a random variable with zero-mean and standard deviation of $1\sqrt{t}$ m/s at time t.

ARW and VRW can be determined from the noise density using the following relations [41]

$$\text{ARW } [\text{deg}/\sqrt{\text{hr}}] = 60\frac{\sqrt{\text{s}}}{\sqrt{\text{hr}}} \times \text{RND } [\text{deg/s}/\sqrt{\text{Hz}}] \tag{2.5}$$

$$\text{VRW } [\text{m/s}/\sqrt{\text{hr}}] = \frac{9.81 \text{ m/s}^2}{10^6 \text{ µg}} \times 60\frac{\sqrt{\text{s}}}{\sqrt{\text{hr}}} \times \text{PSD } [\text{µg}/\sqrt{\text{Hz}}] \tag{2.6}$$

Using these relations, the MPU-6000 has a predicted ARW of 0.3 deg/$\sqrt{\text{hr}}$ and VRW of 0.235 m/s/$\sqrt{\text{hr}}$.

2.1.4.5 Bias Random Walk

The bias in MEMS IMUs tends to slowly vary over long time scales, a phenomenon known as bias random walk. The source of this error is electronic flicker noise which exhibits a 1/f spectrum [41]. Several models have been proposed to accurately model this noise. Gebre-Egiziabher [16] proposed a first-order Markov process driven by white noise. Woodman [41] proposed that the bias walk be modeled as a random walk sequence ϵ_k where

$$\epsilon_k = \sum_{i=1}^{k} N_i \tag{2.7}$$

where N_i are zero-mean, AWGN sequences. This model is not entirely accurate since as $t \to \infty$, the standard deviation of the random walk will grow without bound. However, this model is sufficient for analysis since calibration errors and random noise dominate the dead-reckoning errors (as shown in Chapter 5). Manufacturers typically specify the bias random walk in terms of two numbers [39]: the in-run bias stability and a time at which the bias stability measurement was taken. The bias stability has units of deg/hr for gyros and mg for accelerometers. For example, if the bias stability of a gyro is specified as 1 deg/hr with $\tau = 100$ s and the bias b is known at time t, then the bias of the sensor at $(t + 100)$ seconds can be modeled as random variable $x(t)$ where $E[x(t)] = b$ and $\sigma_x = 1$ deg/hr [41].

2.2 Performance Predictions

For analysis, the error models for the gyros and accelerometers can be combined and simplified into the following form [14]

$$\boldsymbol{y}_i = x + \boldsymbol{\nu}_i, \qquad i = 1, \ldots, N \tag{2.8}$$

where $\boldsymbol{y}_i$ is the measured angular rate/specific force for a given axis, x is the true angular rate/specific force and $\boldsymbol{\nu}_i$ is the output noise with and $E[\boldsymbol{\nu}_i] = 0$ and $E[\boldsymbol{\nu}_i^2] = \sigma_\nu^2 \; \forall \; i$. Bold quantities denote random variables. The subscript i denotes the sensor index. Both the white noise and noise due to bias random walk are lumped into $\boldsymbol{\nu}_i$. The output of the MIMUC, $\boldsymbol{z}$, can then be found by averaging the output of each IMU

$$\boldsymbol{z} = \frac{1}{N} \sum_{i=1}^{N} x + \boldsymbol{\nu}_i \tag{2.9}$$

It is assumed that the noise sources for each sensor are independent such that $E[\boldsymbol{\nu}_i \boldsymbol{\nu}_j] = 0$. In reality, there may be mechanical coupling between sensors (since each contains a vibrating element) or common-mode noise present on the output of each sensor due to fluctuations in the power supply. Care has been taken in the MIMUC design to properly filter all power supply pins to limit the latter effect.

The variance of the MIMUC output is given by

$$\sigma_z^2 = E[\boldsymbol{z}^2] - E[\boldsymbol{z}]^2 \tag{2.10}$$

$$= E\Big[\Big(\frac{1}{N} \sum_{i=1}^{N} x + \boldsymbol{\nu}_i\Big)^2\Big] - E\Big[\frac{1}{N} \sum_{i=1}^{N} x + \boldsymbol{\nu}_i\Big]^2 \tag{2.11}$$

$$= E\Big[\Big(x + \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{\nu}_i\Big)^2\Big] - x^2 \tag{2.12}$$

$$= \frac{\sigma_\nu^2}{N} \tag{2.13}$$

This important result shows the variance of the output noise of the MIMUC is inversely proportional to the number of sensors, i.e. the standard deviation of the output noise scales as $1/\sqrt{N}$. Since the MIMUC has sixteen individual IMUs, the standard deviation of the output noise is expected to be reduced by a factor of four when compared with a single IMU. The expected performance improvements between the MIMUC and a single IMU are given in Table 2.1. Since the MPU-6000 datasheet does not specify a bias stability for the gyro or accelerometer, this parameter was determined empirically using an Allan Variance plot and computing the average bias stability of all the sensors.

Performance Metric	Units	MPU-6000	MIMUC
Angle Random Walk	$\text{deg}/\sqrt{\text{hr}}$	0.30	0.08
Velocity Random Walk	$\text{m/s}/\sqrt{\text{hr}}$	0.24	0.06
Gyro Bias Stability ($\tau = 500$ s)	deg/hr	4.6	1.2
Accelerometer Bias Stability ($\tau = 800$ s)	µg	36	9

TABLE 2.1: Comparison of performance metrics between IMU cluster and MPU-6000

2.3 Determining Calibration Coefficients

Calibration is the process of determining the coefficients of the deterministic error sources defined in the previous section: temperature independent/dependent bias error, scale factor error, misalignment error, and g-dependent bias error for the gyros. These error coefficients are used in conjunction with the error models to determine the true angular rate and specific force given the measured angular rate and specific force. The procedure used to calibrate the accelerometers and gyros is outlined in the IEEE Standards [25, 26]. The coefficients will be derived only for the x-axis; the y-axis and z-axis formulas will be given as they can be found easily via change of variables.

The calibration procedure requires the MIMUC to be placed in six orthogonal orientations (each axis parallel/anti-parallel with the gravity vector) and rotated at three different rates ($\pm\boldsymbol{\omega}_\alpha$ and zero rate), for a total of 18 datasets. For

each combination of rate and orientation, data is collected for one minute and then time averaged to eliminate the stochastic components of the signal ($E[\boldsymbol{\eta}] = 0$ and $E[\boldsymbol{\epsilon}] = 0$). One minute was chosen as the acquisition time to sufficiently average out the random noise but short enough that fluctuations due to bias random walk are small. Each individual IMU sensor on the MIMUC is calibrated individually. For ease of presentation, the calibration coefficients for each axis are determined independently rather than in matrix form.

2.3.1 Gyro Bias

Suppose $\boldsymbol{\omega} = \mathbf{0}$, $a_y = a_z = 0$, and $\Delta T = 0$. Applying the time average operator to Eq. (2.1) yields

$$K_g \langle \tilde{\omega}_x \rangle = b_{g,x} + G_{xx} \langle a_x \rangle \tag{2.14}$$

Define $\tilde{\omega}_{x,+ax}$ to be the X axis output of the gyro with the its X axis anti-parallel to the gravitational vector ($\langle a_x \rangle$ = 1 g). Similarly, define $\tilde{\omega}_{x,-ax}$ to be the output of the gyro X axis with its X axis parallel to the gravitational vector ($\langle a_x \rangle$ = -1 g). It follows that

$$\frac{K_g}{2}\big(\langle \tilde{\omega}_{x,+ax} \rangle + \langle \tilde{\omega}_{x,-ax} \rangle\big) = \frac{(b_{g,x} + G_{xx}(1\ g)) + (b_{g,x} + G_{xx}(-1\ g))}{2} = b_{g,x} \tag{2.15}$$

Similarly,

$$b_{g,y} = \frac{K_g}{2}\big(\langle \tilde{\omega}_{y,+ay} \rangle + \langle \tilde{\omega}_{y,-ay} \rangle\big) \tag{2.16}$$

$$b_{g,z} = \frac{K_g}{2}\big(\langle \tilde{\omega}_{z,+az} \rangle + \langle \tilde{\omega}_{z,-az} \rangle\big) \tag{2.17}$$

This procedure is repeated for each axis of all N gyros, for a total of $3N$ gyro bias calibrations.

2.3.2 Gyro Scale Factor

Suppose $\omega_y = \omega_z = 0$, $a_y = a_z = 0$, and $\Delta T = 0$. Applying the time average operator to Eq. (2.1) yields

$$K_g\langle\tilde{\omega}_x\rangle = (1 + S_{g,x})\langle\omega_x\rangle + b_{g,x} + G_{xx}\langle a_x\rangle \tag{2.18}$$

Define $\tilde{\omega}_{x,+\beta x}$ to be the X axis output of the gyro when $\langle\omega_x\rangle = +\beta$ and the X axis is anti-parallel to the gravity vector ($\langle a_x\rangle = 1$ g). The input rate β should be chosen to be well within the gyro operating range to avoid sensor non-linearities at extreme rates. Similarly, define $\tilde{\omega}_{x,-\beta x}$ to be the X axis output of the gyro when $\langle\omega_x\rangle = -\beta$ and the X axis is anti-parallel to the gravity vector ($\langle a_x\rangle = 1$ g). It follows that

$$\begin{aligned}&\frac{K_g}{2\beta}\big(\langle\tilde{\omega}_{x,+\beta x}\rangle + \langle\tilde{\omega}_{x,-\beta x}\rangle\big) - 1 \\ &= \frac{(1 + S_{g,x})\beta + b_{g,x} + G_{xx}(1\ g) - (1 + S_{g,x})(-\beta) - b_{g,x} - G_{xx}(1\ g)}{2\beta} - 1 = S_{g,x}\end{aligned} \tag{2.19}$$

Similarly,

$$S_{g,y} = \frac{K_g}{2\beta}\big(\langle\tilde{\omega}_{y,+\beta y}\rangle - \langle\tilde{\omega}_{y,-\beta y}\rangle\big) - 1 \tag{2.20}$$

$$S_{g,z} = \frac{K_g}{2\beta}\big(\langle\tilde{\omega}_{z,+\beta z}\rangle - \langle\tilde{\omega}_{z,-\beta z}\rangle\big) - 1 \tag{2.21}$$

This procedure is repeated for each axis of all N gyros, for a total of $3N$ gyro scale factor calibrations.

2.3.3 Gyro Misalignment

Suppose $\omega_x = \omega_z = 0$, $a_x = a_z = 0$, and $\Delta T = 0$. Applying the time average operator to Eq. (2.1) yields

$$K_g\langle\tilde{\omega}_x\rangle = M_{g,xy}\langle\omega_y\rangle + b_{g,x} + G_{xy}\langle a_y\rangle \tag{2.22}$$

Define $\tilde{\omega}_{x,+\beta y}$ to be the X axis output of the gyro when $\langle\omega_y\rangle = +\beta$ and the Y axis is anti-parallel to the gravity vector ($\langle a_y\rangle = 1$ g). β should be chosen to be well within the gyro operating range to avoid sensor non-linearities at extreme rates. Similarly, define $\tilde{\omega}_{x,-\beta y}$ to be the X axis output of the gyro when $\langle\omega_y\rangle = -\beta$ and the Y axis is anti-parallel to the gravity vector ($\langle a_y\rangle = 1$ g). It follows that

$$\begin{aligned}&\frac{K_g}{2\beta}\big(\langle\tilde{\omega}_{x,+\beta y}\rangle - \langle\tilde{\omega}_{x,-\beta y}\rangle\big)\\ &= \frac{(M_{g,xy}\beta + b_{g,x} + G_{xy}(1\ g)) - (M_{g,xy}(-\beta) + b_{g,x} + G_{xy}(-1\ g))}{2\beta} = M_{g,xy}\end{aligned} \tag{2.23}$$

Similarly,

$$M_{g,xz} = \frac{K_g}{2\beta}\big(\langle\tilde{\omega}_{x,+\beta z}\rangle - \langle\tilde{\omega}_{x,-\beta z}\rangle\big) \tag{2.24}$$

$$M_{g,yz} = \frac{K_g}{2\beta}\big(\langle\tilde{\omega}_{y,+\beta z}\rangle - \langle\tilde{\omega}_{y,-\beta z}\rangle\big) \tag{2.25}$$

This procedure is repeated for each of the N gyros, for a total of $3N$ gyro misalignment calibrations.

2.3.4 Gyro g-Sensitivity

Suppose $\boldsymbol{\omega} = \mathbf{0}$, $a_y = a_z = 0$, and $\Delta T = 0$. Applying the time average operator to Eq. (2.1) yields

$$K_g\langle\tilde{\omega}_x\rangle = b_{g,x} + G_{xx}\langle a_x\rangle \tag{2.26}$$

Define $\tilde{\omega}_{x,+ax}$ to be the X axis output of the gyro with the its X axis anti-parallel to the gravitational vector ($\langle a_x \rangle$ = 1 g). Similarly, define $\tilde{\omega}_{x,-ax}$ to be the output of the gyro X axis with its X axis parallel to the gravitational vector ($\langle a_x \rangle$ = -1 g). It follows that

$$\frac{K_g}{2}\big(\langle\tilde{\omega}_{x,+ax}\rangle - \langle\tilde{\omega}_{x,-ax}\rangle\big) = \frac{(b_{g,x} + G_{xx}(1\ g)) - (b_{g,x} - G_{xx}(-1\ g))}{2} = G_{xx} \quad (2.27)$$

Similarly,

$$G_{ij} = \frac{K_g}{2}\big(\langle\tilde{\omega}_{i,+aj}\rangle - \langle\tilde{\omega}_{i,-aj}\rangle\big) \quad (2.28)$$

This procedure is repeated for each of the N gyros, for a total of $9N$ gyro g-sensitivity calibrations.

2.3.5 Accelerometer Bias

Suppose $a_y = a_z = 0$ and $\Delta T = 0$. Applying the time average operator to Eq. (2.3) yields

$$K_a\langle\tilde{a}_x\rangle = (1 + S_{a,x})\langle a_x\rangle + b_{a,x} \quad (2.29)$$

Define $\tilde{a}_{x,+ax}$ to be the X axis output of the accelerometer with its X axis anti-parallel to the gravitational vector ($\langle a_x \rangle$ = 1 g). Similarly, define $\tilde{a}_{x,-ax}$ to be the X axis output of the accelerometer with its X axis parallel to the gravitational vector ($\langle a_x \rangle$ = -1 g). It follows that

$$\frac{K_a}{2}\big(\langle\tilde{a}_{x,+gx}\rangle + \langle\tilde{a}_{x,-gx}\rangle\big) = \frac{b_{a,x} + (1 + S_{a,x})(1\ g) + b_{a,x} + (1 + S_{a,x})(-1\ g)}{2} = b_{a,x} \quad (2.30)$$

where g is the units of standard gravity on Earth's surface. Similarly,

$$b_{a,y} = \frac{K_a}{2}\big(\langle\tilde{a}_{y,+ay}\rangle + \langle\tilde{a}_{y,-ay}\rangle\big) \tag{2.31}$$

$$b_{a,z} = \frac{K_a}{2}\big(\langle\tilde{a}_{z,+az}\rangle + \langle\tilde{a}_{z,-az}\rangle\big) \tag{2.32}$$

This procedure is repeated for each axis of all N gyros, for a total of $3N$ accelerometer bias calibrations.

2.3.6 Accelerometer Scale Factor

Suppose $a_y = a_z = 0$ and $\Delta T = 0$. Applying the time average operator to Eq. (2.3) yields

$$K_a\langle\tilde{a}_x\rangle = (1 + S_{a,x})\langle a_x\rangle + b_{a,x} \tag{2.33}$$

Define $\tilde{a}_{x,+ax}$ to be the X axis output of the accelerometer when $\langle a_x\rangle = 1\ g$. Similarly, define $\tilde{a}_{x,-ax}$ to be the X axis output of the gyro when $\langle a_x\rangle = -1\ g$. It follows that

$$\frac{K_a}{2g}\big(\langle\tilde{a}_{x,+ax}\rangle - \langle\tilde{a}_{x,-ax}\rangle\big) - 1 = \frac{b_{a,x} + (1 + S_{a,x})(1\ g) - b_{a,x} - (1 + S_{a,x})(-1\ g)}{2g} - 1 = S_{a,x} \tag{2.34}$$

where g is the units of standard gravity on Earth's surface. Similarly,

$$S_{a,y} = \frac{K_a}{2g}\big(\langle\tilde{a}_{y,+ay}\rangle + \langle\tilde{a}_{y,-ay}\rangle\big) - 1 \tag{2.35}$$

$$S_{a,z} = \frac{K_a}{g2}\big(\langle\tilde{a}_{z,+az}\rangle + \langle\tilde{a}_{z,-az}\rangle\big) - 1 \tag{2.36}$$

This procedure is repeated for each axis of all N gyros, for a total of $3N$ accelerometer scale factor calibrations.

2.3.7 Accelerometer Misalignment

Suppose $a_x = a_z = 0$ and $\Delta T = 0$. Applying the time average operator to Eq. (2.3) yields

$$K_a \langle \tilde{a}_x \rangle = M_{a,xy} \langle a_y \rangle + b_{a,x} \tag{2.37}$$

Define $\tilde{a}_{x,+ay}$ to be the X axis output of the accelerometer when $\langle a_y \rangle = 1\ g$. Similarly, define $\tilde{a}_{x,-ay}$ to be the X axis output of the gyro when $\langle a_y \rangle = -1\ g$. It follows that

$$\frac{K_a}{2g}\big(\langle \tilde{a}_{x,+ay} \rangle - \langle \tilde{a}_{x,-ay} \rangle\big) = \frac{(b_{a,x} + M_{a,xy}(1\ g)) - (b_{a,x} - M_{a,xy}(-1\ g))}{2g} = M_{a,x} \tag{2.38}$$

where g is the units of standard gravity on Earth's surface. Similarly,

$$M_{a,xz} = \frac{K_a}{2g}\big(\langle \tilde{a}_{x,+az} \rangle + \langle \tilde{a}_{x,-az} \rangle\big) \tag{2.39}$$

$$M_{a,yz} = \frac{K_a}{2g}\big(\langle \tilde{a}_{y,+az} \rangle + \langle \tilde{a}_{y,-az} \rangle\big) \tag{2.40}$$

This procedure is repeated for each of the N gyros, for a total of $3N$ accelerometer misalignment calibrations.

2.3.8 Temperature-dependent Bias

A thermal chamber can be used to measure the temperature dependence of the bias. The IMU is placed inside the thermal chamber and the temperature is cycled over (some subset of) the operating range of the sensor. The thermal chamber is cycled between the minimum and maximum temperature several times. Data is recorded during the entire process. The digital output of each sensor is then plotted versus the temperature recorded by the IMU's internal temperature sensor. A linear trend line is fit to the data using least squares approximation.

The temperature drift coefficient (LSB/°C) of the bias is equivalent to the slope of the trend line.

2.4 Dead Reckoning

Dead-reckoning is a method of navigation that relies solely on measuring changes in the inertia of the vehicle. Given a known initial position, velocity and attitude, estimates of current velocity and attitude can be used to propagate position. An IMU is able to measure the changes in angular and linear inertia, providing measures of both non-gravitational acceleration and angular ate. The specific force can be integrated with respect to time to determine the velocity, and once more to determine the position. The angular rate can be integrated to determine the attitude. The specific force cannot be integrated directly, however, since they are measured in vehicle frame (or body frame), which is generally a non-inertial reference frame due to the vehicle having a non-zero angular rate. They must first be transformed to inertial frame.

One method of transforming the specific forces from the body frame to an inertial frame is the *direction cosine matrix* (DCM). The DCM is denoted by $\boldsymbol{C}_b^i$, where the subscript b and superscript i indicate the direction of the transformation. The specific force in the inertial frame is given by [39]

$$\boldsymbol{f}_i(t) = \boldsymbol{C}_b^i(t)\boldsymbol{f}_b(t)$$

The propagation of the DCM with time can be written as [39]

$$\dot{\boldsymbol{C}}_b^i = \boldsymbol{C}_b^i\boldsymbol{\Omega}_b^i$$

where

$$\boldsymbol{\Omega}_b^i = \begin{pmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{pmatrix}$$

The solution of $\dot{\boldsymbol{C}}_b^i$ may be solved on a digital computer via numerical integration.

2.5 Coning

2.5.1 Background

Coning errors arise due to motion of the angular rate vector in the vehicle frame. Since the angular rate measurement is only available at discrete time instants, it must be assumed that the angular rate vector remains fixed over the integration interval. If a vehicle is undergoing rapid angular motion, this assumption does not hold. Failure to account for this motion will result in errors in the attitude computation due to the non-commutativity of finite rotations. It is emphasized that this "coning error" is a result of measuring the angular rate at discrete times; the error would appear even with an ideal gyroscope.

The naive approach to reducing this error is reducing the sampling time between attitude updates. The conventional method for updating the attitude is to compute the direction cosine matrix (DCM) which relates the specific force measurements of the accelerometer from the vehicle frame to an inertial frame [39]. Computing the DCM is a non-trivial task; increasing the attitude update rate may not be possible due to limitations in computational hardware.

To circumvent this issue, Bortz [5] developed a method where the attitude is represented as a *rotation vector*. A series of algorithms allow the rotation vector to be updated at a much higher rate than the DCM update rate. Updating the rotation vector is computationally inexpensive compared to updating the DCM.

This effectively increases the sampling rate to allow high frequency motion of the rotation vector to be captured without overwhelming the microprocessor. What follows is a derivation of the algorithm used to update the rotation vector, also referred to as the *coning algorithm*. For further discussion on the source of coning errors, refer to Flynn [15].

2.5.2 Derivation of the Coning Equations

The attitude of the vehicle is represented using Euler's rotation theorem. This theorem states that any rotation can be expressed as a rotation of an angle ϕ about some axis $\boldsymbol{e}$ [39]. Let the rotation vector parameterization of attitude be defined as

$$\boldsymbol{\phi} = \phi \boldsymbol{e} \tag{2.41}$$

This representation was introduced by Bortz to account for the non-commutativity term of the attitude. The kinematics of the vector are given by Bortz' equation [5]

$$\dot{\boldsymbol{\phi}}(t) = \boldsymbol{\omega}(t) + \frac{1}{2}\boldsymbol{\phi}(t) \times \boldsymbol{\omega}(t) + \frac{1}{\phi(t)^2}\Big(1 - \frac{\phi(t)}{2}\cot\frac{\phi(t)}{2}\Big)\boldsymbol{\phi}(t) \times \boldsymbol{\phi}(t) \times \boldsymbol{\omega}(t) \tag{2.42}$$

Savage[30] presented a method for simplifying the above expression to second order accuracy. By replacing $\cot(\phi(t)/2)$ with its Taylor series (up to second order terms), the scalar multiplier of $\boldsymbol{\phi}(t) \times \boldsymbol{\phi}(t) \times \boldsymbol{\omega}(t)$ can be approximated as

$$\frac{1}{\phi(t)^2}\left[1 - \frac{\phi(t)}{2}\Big(\frac{2}{\phi(t)} - \frac{\phi(t)}{6}\Big)\right] \approx \frac{1}{12} \tag{2.43}$$

Assuming a small coning motion ($\phi(t)$ is small), the remaining terms can be approximated to second order accuracy as [30]

$$\frac{1}{2}\boldsymbol{\phi}(t) \times \boldsymbol{\omega}(t) + \frac{1}{12}\boldsymbol{\phi}(t) \times \boldsymbol{\phi}(t) \times \boldsymbol{\omega}(t) \approx \frac{1}{2}\boldsymbol{\theta}(t) \times \boldsymbol{\omega}(t) \tag{2.44}$$

where

$$\boldsymbol{\theta}(t) \triangleq \int_{t_{k-1}}^{t} \boldsymbol{\omega}(\tau) d\tau \tag{2.45}$$

Eq. (2.42) can then be approximated to second order accuracy as

$$\dot{\boldsymbol{\phi}}(t) = \boldsymbol{\omega}(t) + \frac{1}{2}\boldsymbol{\theta}(t) \times \boldsymbol{\omega}(t) \tag{2.46}$$

Consider the time interval $[t_{k-1}\ t_k]$ where $t_k = t_{k-1} + T_k$ and $T_k = 1/f_k$ with f_k given. The change in the rotation vector over this time interval is given by integrating Eq. (2.46)

$$\Delta\boldsymbol{\phi}(t_k) = \int_{t_{k-1}}^{t_k} \Big(\boldsymbol{\omega}(t) + \frac{1}{2}\boldsymbol{\theta}(t) \times \boldsymbol{\omega}(t)\Big) d\tau \tag{2.47}$$

This integral can be split into two parts. The first part is dependent on $\boldsymbol{\omega}$(t) and is due to the inertially measurable motion. This component is measured by the gyro. The second part due to the non-inertially measurable motion that cannot be measured by the gyro. Let the integral be rewritten as

$$\Delta\boldsymbol{\phi}(t_k) = \boldsymbol{\theta}(t_k) + \boldsymbol{\beta}(t_k) \tag{2.48}$$

with

$$\boldsymbol{\theta}(t) = \int_{t_{k-1}}^{t} \boldsymbol{\omega}(\tau) d\tau \qquad \text{and} \qquad \boldsymbol{\beta}(t) = \frac{1}{2}\int_{t_{k-1}}^{t} \boldsymbol{\theta}(\tau) \times \boldsymbol{\omega}(\tau) d\tau$$

$\boldsymbol{\theta}(t)$ can be calculated by numerical integrating the output of the gyros. $\boldsymbol{\beta}(t)$, also known as the *non-commutativity* term, must be written in a more suitable form for computation. This term contains the errors due to non-inertial movement of the orientation vector. Recall that the object is to measure the output of the gyros at a much higher rate to better track high frequency movement of the rotation

vector. During the DCM update, the non-commutative effects can be accounted for.

We begin by subdividing the time interval $[t_{k-1}\ t_k]$ into N subdivisions. Define the attitude update rate as f_m where $T_m = 1/f_m$. The attitude update rate must be an integer multiple of the body update rate such that $f_m = Nf_k$. Define t_m as the m^{th} subdivision of the k^{th} interval, such that $t_m = t_{k-1} + mT_m$ where $m = 0, 1, ..., N$.

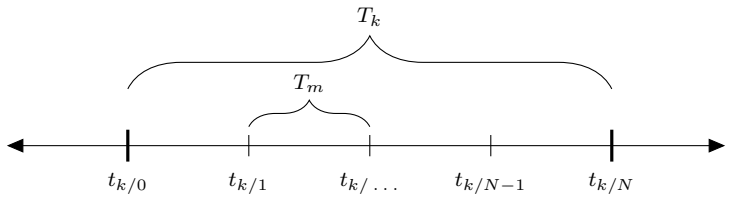

FIGURE 2.1: Subdivision of time interval $[t_{k-1}\ t{-}_k]$ for coning update

Applying this subdivision to $\boldsymbol{\theta}(t)$ and $\boldsymbol{\beta}(t)$ yields

$$\boldsymbol{\theta}(t) = \boldsymbol{\theta}(t_{m-1}) + \Delta\boldsymbol{\theta}(t) \tag{2.49}$$

$$\boldsymbol{\beta}(t) = \boldsymbol{\beta}(t_{m-1}) + \Delta\boldsymbol{\beta}(t) \tag{2.50}$$

where

$$\Delta\boldsymbol{\theta}(t) = \int_{t_{m-1}}^{t} \boldsymbol{\omega}(\tau)d\tau \qquad \text{and} \qquad \Delta\boldsymbol{\beta}(t) = \frac{1}{2}\int_{t_{m-1}}^{t} \boldsymbol{\theta}(\tau) \times \boldsymbol{\omega}(\tau)d\tau$$

Since measurements of $\boldsymbol{\theta}(t)$ and $\boldsymbol{\beta}(t)$ are only available at discrete sampling instants, it is useful to restate Eq. (2.40) and Eq. (2.41) in a discrete form.

$$\boldsymbol{\theta}_m = \boldsymbol{\theta}_{m-1} + \Delta\boldsymbol{\theta}_m \tag{2.51}$$

$$\boldsymbol{\beta}_m = \boldsymbol{\beta}_{m-1} + \Delta\boldsymbol{\beta}_m \tag{2.52}$$

where

$$\boldsymbol{\theta}_m = \int_{t_{k-1}}^{t_m} \boldsymbol{\omega}(\tau)d\tau \qquad \Delta\boldsymbol{\theta}_m = \int_{t_{m-1}}^{t_m} \boldsymbol{\omega}(\tau)d\tau$$

$$\boldsymbol{\beta}_m = \frac{1}{2}\int_{t_{k-1}}^{t_m} \boldsymbol{\theta}(\tau) \times \boldsymbol{\omega}(\tau)d\tau \qquad \Delta\boldsymbol{\beta}_m = \frac{1}{2}\int_{t_{m-1}}^{t_m} \boldsymbol{\theta}(\tau) \times \boldsymbol{\omega}(\tau)d\tau$$

The $\Delta\boldsymbol{\beta}_m$ can be rearranged

$$\begin{aligned}
\Delta\boldsymbol{\beta}_m &= \frac{1}{2}\int_{t_{m-1}}^{t_m} \boldsymbol{\theta}(\tau) \times \boldsymbol{\omega}(\tau)d\tau \\
&= \frac{1}{2}\int_{t_{m-1}}^{t_m} \big(\boldsymbol{\theta}_{m-1} + \Delta\boldsymbol{\theta}(\tau)\big) \times \boldsymbol{\omega}(\tau)d\tau \\
&= \frac{1}{2}\Big[\boldsymbol{\theta}_{m-1} \times \int_{t_{m-1}}^{t_m} \boldsymbol{\omega}(\tau)d\tau\Big] + \frac{1}{2}\int_{t_{m-1}}^{t_m} \Delta\boldsymbol{\theta}(\tau) \times \boldsymbol{\omega}(\tau)d\tau \\
&= \frac{1}{2}\big(\boldsymbol{\theta}_{m-1} \times \Delta\boldsymbol{\theta}_m\big) + \frac{1}{2}\int_{t_{m-1}}^{t_m} \Delta\boldsymbol{\theta}(\tau) \times \boldsymbol{\omega}(\tau)d\tau
\end{aligned} \tag{2.53}$$

The integral can be simplified by approximating $\boldsymbol{\omega}(t)$ as a first-order power series on the domain $t \in [t_{m-1}\ t_m]$

$$\boldsymbol{\omega}(t) \approx \boldsymbol{\omega}_0 + \boldsymbol{\omega}_1(t - t_{m-1}) \tag{2.54}$$

The constants $\boldsymbol{\omega}_0$ and $\boldsymbol{\omega}_1$ can then be found in terms of $\Delta\boldsymbol{\theta}_{m-1}$ and $\Delta\boldsymbol{\theta}_m$. Begin by substituting $\boldsymbol{\omega}$(t) with its first order approximation in the definition of $\Delta\boldsymbol{\theta}_m$

yielding

$$\Delta\boldsymbol{\theta}_{m-1} = \int_{t_{m-2}}^{t_{m-1}} \boldsymbol{\omega}_0 + \boldsymbol{\omega}_1(\tau - t_{m-1})d\tau$$
$$= \boldsymbol{\omega}_0(t_{m-1} - t_{m-2}) + \frac{\boldsymbol{\omega}_1}{2}(t_{m-1} - t_{m-2})^2 = \boldsymbol{\omega}_0 T_m + \frac{1}{2}\boldsymbol{\omega}_1(T_m)^2 \quad (2.55)$$

and (2.56)

$$\Delta\boldsymbol{\theta}_m = \int_{t_{m-1}}^{t_m} \boldsymbol{\omega}_0 + \boldsymbol{\omega}_1(\tau - t_{m-1})d\tau$$
$$= \boldsymbol{\omega}_0(t_m - t_{m-1}) - \frac{\boldsymbol{\omega}_1}{2}(t_m - t_{m-1})^2 = \boldsymbol{\omega}_0 T_m - \frac{1}{2}\boldsymbol{\omega}_1(T_m)^2 \quad (2.57)$$

yielding

$$\boldsymbol{\omega}_0 = \frac{1}{2T_m}\big(\Delta\boldsymbol{\theta}_m + \Delta\boldsymbol{\theta}_{m-1}\big) \quad (2.58)$$
$$\boldsymbol{\omega}_1 = \frac{1}{(T_m)^2}\big(\Delta\boldsymbol{\theta}_m - \Delta\boldsymbol{\theta}_{m-1}\big) \quad (2.59)$$

Replacing $\boldsymbol{\omega}(t)$ in Eq. (2.59) with the first order approximation yields

$$\frac{1}{2}\int_{t_{m-1}}^{t_m} \Delta\boldsymbol{\theta}(\tau) \times \boldsymbol{\omega}(\tau)d\tau = \frac{1}{2}\int_{t_{m-1}}^{t_m} \big(\boldsymbol{\omega}_0(\tau - t_{m-1}) + \frac{1}{2}\boldsymbol{\omega}_1(\tau - t_{m-1})^2\big)$$
$$\times \big(\boldsymbol{\omega}_0 + \boldsymbol{\omega}_1(\tau - t_{m-1})\big)d\tau$$
$$= \frac{1}{2}\int_{t_{m-1}}^{t_m} (\boldsymbol{\omega}_0 \times \boldsymbol{\omega}_1)(\tau - t_{m-1})^2 d\tau = \frac{1}{6}(\boldsymbol{\omega}_0 \times \boldsymbol{\omega}_1)(T_m)^3$$
$$= \frac{1}{12}(\Delta\boldsymbol{\theta}_m + \Delta\boldsymbol{\theta}_{m-1}) \times (\Delta\boldsymbol{\theta}_m - \Delta\boldsymbol{\theta}_{m-1})$$
$$= \frac{1}{6}(\Delta\boldsymbol{\theta}_{m-1} \times \Delta\boldsymbol{\theta}_m) \quad (2.60)$$

Finally,

$$\begin{aligned}\Delta\boldsymbol{\beta}_m &= \frac{1}{2}(\boldsymbol{\theta}_{m-1} \times \Delta\boldsymbol{\theta}_{m-1}) + \frac{1}{6}(\Delta\boldsymbol{\theta}_{m-1} \times \Delta\boldsymbol{\theta}_m) \\ &= \frac{1}{2}(\boldsymbol{\theta}_{m-1} + \frac{1}{6}\Delta\boldsymbol{\theta}_{m-1}) \times \Delta\boldsymbol{\theta}_m \end{aligned} \tag{2.61}$$

The pertinent equations are summarized

$$\boldsymbol{\omega}_m = \text{gyro output at } t_m \text{ during k-th update cycle} \tag{2.62}$$

$$\boldsymbol{\theta}_m = \boldsymbol{\theta}_{m-1} + \Delta\boldsymbol{\theta}_m \tag{2.63}$$

$$\boldsymbol{\beta}_m = \boldsymbol{\beta}_{m-1} + \Delta\boldsymbol{\beta}_m \tag{2.64}$$

$$\Delta\boldsymbol{\theta}_m = \frac{T_m}{2}(\boldsymbol{\omega}_m + \boldsymbol{\omega}_{m-1}) \tag{2.65}$$

$$\Delta\boldsymbol{\beta}_m = \frac{1}{2}(\boldsymbol{\theta}_{m-1} + \frac{1}{6}\Delta\boldsymbol{\theta}_{m-1}) \times \Delta\boldsymbol{\theta}_m \tag{2.66}$$

The algorithm is executed at the body update rate f_k. The batch of N gyro measurements over the time interval $[t_{k-1}\ t_k]$ are run through Eq. (2.67)-(2.71) to compute $\boldsymbol{\theta}_k$ and $\boldsymbol{\beta}_k$. These quantities can be substituted into Eq. (2.52) to determine the change in rotation vector over the time interval $[t_{k-1}\ t_k]$.

2.5.3 Rotation Vector Update

A brief introduction to quaternions is required before discussing how the previously derived results can be used to update the rotation vector. A *quaternion* is one of many ways to represent attitude. Quaternions are advantageous over Euler angles since they do not run into the problem of singularities at $\theta = \pm 90$ deg [39]. A quaternion is a four element vector defined as

$$\bar{\boldsymbol{q}} = \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} \cos(\mu/2) \\ (\mu_x/\mu)\sin(\mu/2) \\ (\mu_y/\mu)\sin(\mu/2) \\ (\mu_z/\mu)\sin(\mu/2) \end{pmatrix} \tag{2.67}$$

where $\boldsymbol{\mu}$ is the axis of rotation and the rotation angle is given by $|\boldsymbol{\mu}| = \mu$ [21]. A quaternion represents a rotation about a given axis $\boldsymbol{\mu}$ by some angle μ. The attitude quaternion is defined such that if the inertial reference frame is rotated by μ about $\boldsymbol{\mu}$, it will coincide with the vehicle frame. This directly coincides with the definition of the rotation vector $\boldsymbol{\phi}$. Therefore, the attitude quaternion is the quaternion with $\boldsymbol{\mu} = \boldsymbol{\phi}$.

The change in the rotation vector $\Delta\phi$ defines the angle and axis that the current attitude quaternion should be rotated by. Let $\bar{\boldsymbol{q}}_{k-1}$ be the attitude quaternion at time t_{k-1} and $\Delta\phi_k$ the change in the rotation vector over the time interval $[t_{k-1}\ t_k]$. The attitude quaternion $\bar{\boldsymbol{q}}_k$ at time t_k is given by

$$\bar{\boldsymbol{q}}_k = \bar{\boldsymbol{q}}_{k-1} \times \begin{bmatrix} \sin\left(\frac{\Delta\phi_k}{2}\right)\frac{\Delta\boldsymbol{\phi}_k}{\Delta\phi_k} \\ cos\left(\frac{\Delta\phi_k}{2}\right) \end{bmatrix} \tag{2.68}$$

where $\times$ represents the quaternion product. If $\bar{\boldsymbol{p}}$ and $\bar{\boldsymbol{r}}$ are two arbitrary quaternions, let their product be defined as

$$\bar{\boldsymbol{r}}_k \times \bar{\boldsymbol{p}}_k = \begin{pmatrix} p_a r_a - p_b r_b - p_c r_c - p_d r_d \\ p_a r_b + p_b r_a + p_c r_d - p_d r_c \\ p_a r_c + p_c r_a - p_b r_d + p_d r_b \\ p_a r_d + p_d r_a + p_b r_c - p_c r_b \end{pmatrix} \tag{2.69}$$

A pseudocode implementation of the coning algorithm can be found in Appendix A.

2.6 Sculling

2.6.1 Background

Sculling errors arise when the vehicle is subjected to certain translational and angular oscillatory motions. An example of this type of motion is the motion of an oar behind a boat, referred to as *sculling*. The oar is driven side to side in a

linear motion (Z axis) while the oar is rotated back and forth along its axis (Y axis) to generate forward thrust. The process of sampling the gyro and accelerometer at discrete time instants results in an error in the X component of the specific force with magnitude $0.5a\omega A\cos(\phi)$ [39]. The idea behind the sculling algorithm is similar to the coning algorithm. The object is to separate out the velocity update into a computationally inexpensive algorithm. This update can be carried out at a fast rate compared to the body update rate to capture high frequency oscillatory linear and angular motion.

2.6.2 Derivation of the Sculling Equations

Savage[31] presents a method to account for the sculling motion in the navigation equations. We begin by considering the equation of motion governing the evolution of the inertial velocity

$$\dot{\boldsymbol{v}}(t) = \boldsymbol{g}(\boldsymbol{r}_{imu}(t)) + \boldsymbol{T}^T(t)\boldsymbol{a}(t) \tag{2.70}$$

where $\boldsymbol{v}$(t) is the velocity, $\boldsymbol{g}(\boldsymbol{r}_{imu}(t))$ is the local gravity vector at the position of the IMU, $\boldsymbol{T}^T$(t) is the rotation matrix, and $\boldsymbol{a}$(t) is the non-gravitational acceleration (e.g. thrust). The rotation matrix $\boldsymbol{T}$ is required because $\boldsymbol{a}$ is typically measured in the IMU case reference frame and $\boldsymbol{r}$ and $\boldsymbol{v}$ are typically represented in an inertial reference frame. Therefore, $\boldsymbol{T}$ transforms the accelerations into an inertial reference frame. Discretization of the time interval $[t_{k-1}\ t_k]$ yields

$$\boldsymbol{v}(t_k) = \boldsymbol{v}(t_{k-1}) + \int_{t_{k-1}}^{t_k} \boldsymbol{g}(r(\tau))d\tau + \int_{t_{k-1}}^{t_k} \boldsymbol{T}^T(\tau)\boldsymbol{a}(\tau)d\tau \tag{2.71}$$

Define changes in velocity due to gravitational accelerations and non-gravitational accelerations as

$$\Delta \boldsymbol{v}_g(t) = \int_{t_{k-1}}^{t} \boldsymbol{g}(\boldsymbol{r}(\tau))d\tau \tag{2.72}$$

$$\Delta \boldsymbol{v}_{ng}(t) = \int_{t_{k-1}}^{t} \boldsymbol{T}^T(\tau)\boldsymbol{a}(\tau)d\tau \tag{2.73}$$

With these definitions, we write the velocity propagation as

$$\boldsymbol{v}(t_k) = \boldsymbol{v}(t_{k-1}) + \Delta \boldsymbol{v}_g(t_k) + \Delta \boldsymbol{v}_{ng}(t_k) \tag{2.74}$$

Assuming the gravitational acceleration is constant over the time interval $[t_{k-1}\ t_k]$ $(T_k = t_k - t_{k-1})$, we have

$$\Delta \boldsymbol{v}_{g,k} = \boldsymbol{g}(\boldsymbol{r}(t_{k-1}))T_k \tag{2.75}$$

The next step is to find an appropriate approximation for the change in velocity due to non-gravitational accelerations. The rotation matrix $\boldsymbol{T}^T(t)$ can be approximated to first-order utilizing the rotation vector

$$\boldsymbol{T}^T(t) \approx \boldsymbol{T}^T_{k-1} + \boldsymbol{T}^T_{k-1}[\boldsymbol{\phi}(t)\times] \tag{2.76}$$

Assuming a small coning effect, we can replace $\boldsymbol{\phi}(t)$ with $\boldsymbol{\theta}(t)$ [30]. Substituting this result back into the integral expression for $\Delta\boldsymbol{v}_{ng,k}$, we have

$$\Delta\boldsymbol{v}_{ng}(t_k) = \int_{t_{k-1}}^{t_k} \big(\boldsymbol{T}_{k-1}^T + \boldsymbol{T}_{k-1}^T[\boldsymbol{\theta}(\tau)\times]\big)\boldsymbol{a}(\tau)d\tau \tag{2.77}$$

$$= \boldsymbol{T}_{k-1}^T \int_{t_{k-1}}^{t_k} \boldsymbol{a}(\tau)d\tau + \boldsymbol{T}_{k-1}^T \int_{t_{k-1}}^{t_k} \boldsymbol{\theta}(\tau)\times\boldsymbol{a}(\tau)d\tau \tag{2.78}$$

$$= \boldsymbol{T}_{k-1}^T \left[\boldsymbol{v}(t_k) + \int_{t_{k-1}}^{t_k} \boldsymbol{\theta}(\tau)\times\boldsymbol{a}(\tau)d\tau\right] \tag{2.79}$$

where

$$\boldsymbol{v}(t_k) = \int_{t_{k-1}}^{t_k} \boldsymbol{a}(\tau)d\tau \tag{2.80}$$

A fundamental limitation to the accuracy is the assumption made by replacing the rotation matrix with its first order approximation. We can improve the quality of the solution by noting that

$$\frac{d}{dt}\big(\boldsymbol{\theta}(t)\times\boldsymbol{v}(t)\big) = \boldsymbol{\theta}(t)\times\dot{\boldsymbol{v}}(t) - \boldsymbol{v}(t)\times\dot{\boldsymbol{\theta}}(t) \tag{2.81}$$

which, when rearranged, yields

$$\boldsymbol{\theta}(t)\times\dot{\boldsymbol{v}}(t) = \frac{d}{dt}\big(\boldsymbol{\theta}(t)\times\boldsymbol{v}(t)\big) + \boldsymbol{v}(t)\times\dot{\boldsymbol{\theta}}(t) \tag{2.82}$$

We can also write

$$\begin{aligned}\boldsymbol{\theta}(t)\times\dot{\boldsymbol{v}}(t) &= \frac{1}{2}\boldsymbol{\theta}(t)\times\dot{\boldsymbol{v}}(t) + \frac{1}{2}\boldsymbol{\theta}(t)\times\dot{\boldsymbol{v}}(t)\\ &= \frac{1}{2}\boldsymbol{\theta}(t)\times\dot{\boldsymbol{v}}(t) + \frac{1}{2}\frac{d}{dt}\big(\boldsymbol{\theta}(t)\times\boldsymbol{v}(t)\big) + \frac{1}{2}\boldsymbol{v}(t)\times\dot{\boldsymbol{\theta}}(t)\\ &= \frac{1}{2}\frac{d}{dt}\big(\boldsymbol{\theta}(t)\times\boldsymbol{v}(t)\big) + \frac{1}{2}\big(\boldsymbol{\theta}(t)\times\dot{\boldsymbol{v}}(t) + \boldsymbol{v}(t)\times\dot{\boldsymbol{\theta}}(t)\big)\end{aligned} \tag{2.83}$$

Utilizing the fact that $\dot{\boldsymbol{\theta}}(t) = \boldsymbol{\omega}(t)$ (Eq. 2.45) and $\dot{\boldsymbol{v}}(t) = \boldsymbol{a}(t)$, we have

$$\boldsymbol{\theta}(t) \times \boldsymbol{a}(t) = \frac{1}{2}\frac{d}{dt}\big(\boldsymbol{\theta}(t) \times \boldsymbol{v}(t)\big) + \frac{1}{2}\big(\boldsymbol{\theta}(t) \times \boldsymbol{a}(t) + \boldsymbol{v}(t) \times \boldsymbol{\omega}(t)\big) \tag{2.84}$$

Therefore, $\Delta\boldsymbol{v}_{ng}(t_k)$ can be rewritten as

$$\Delta\boldsymbol{v}_{ng}(t_k) = \boldsymbol{T}_{k-1}^{T}\Big(\boldsymbol{v}(t_k) + \frac{1}{2}\int_{t_{k-1}}^{t_k}\Big[\frac{d}{dt}\big(\boldsymbol{\theta}(\tau) \times \boldsymbol{v}(\tau)\big) + \boldsymbol{\theta}(t) \times \boldsymbol{a}(\tau) + \boldsymbol{v}(\tau) \times \boldsymbol{\omega}(\tau)\Big]d\tau\Big) \tag{2.85}$$

Evaluating the term involving the integral of the derivative yields

$$\Delta\boldsymbol{v}_{ng}(t_k) = \boldsymbol{T}_{k-1}^{T}\Big(\boldsymbol{v}(t_k) + \frac{1}{2}\boldsymbol{\theta}(t_k) \times \boldsymbol{v}(t_k) + \frac{1}{2}\int_{t_{k-1}}^{t_k}\Big(\boldsymbol{\theta}(\tau) \times \boldsymbol{a}(\tau) + \boldsymbol{v}(\tau) \times \boldsymbol{\omega}(\tau)\Big)d\tau\Big) \tag{2.86}$$

Define a change in velocity due to rotation as

$$\Delta\boldsymbol{v}_{rot}(t_k) = \frac{1}{2}\big(\boldsymbol{\theta}(t_k) \times \boldsymbol{v}(t_k)\big) \tag{2.87}$$

and a change in velocity due to sculling as

$$\Delta\boldsymbol{v}_{scul}(t) = \frac{1}{2}\int_{t_{k-1}}^{t} \boldsymbol{\theta}(\tau) \times \boldsymbol{a}(\tau) + \boldsymbol{v}(\tau) \times \boldsymbol{\omega}(\tau)d\tau \tag{2.88}$$

The above definitions give the following relation for the change in non-gravitational velocity over a single time step

$$\Delta\boldsymbol{v}_{ng}(t_k) = \boldsymbol{v}(t_k) + \Delta\boldsymbol{v}_{rot}(t_k) + \Delta\boldsymbol{v}_{scul}(t_k) \tag{2.89}$$

Applying the same subdivision technique to the time interval $[t_{k-1}\ t_k]$ as shown in Section 2.5 and using previous results yields the following

$$\boldsymbol{v}(t) = \boldsymbol{v}(t_{m-1}) + \Delta\boldsymbol{v}(t) \tag{2.90}$$

$$\Delta\boldsymbol{v}_{scul}(t) = \Delta\boldsymbol{v}_{scul}(t_{m-1}) + \delta\boldsymbol{v}(t) \tag{2.91}$$

where

$$\Delta\boldsymbol{v}(t) = \int_{t_{m-1}}^{t} \boldsymbol{a}(\tau)d\tau \tag{2.92}$$

$$\delta\boldsymbol{v}(t) = \int_{t_{m-1}}^{t} \boldsymbol{\theta}(\tau) \times \boldsymbol{a}(\tau) + \boldsymbol{v}(\tau) \times \boldsymbol{\omega}(\tau)d\tau \tag{2.93}$$

Discretization of the above equations results in a more compact notation suitable for implementation on a microprocessor

$$\boldsymbol{v}_m = \boldsymbol{v}_{m-1} + \Delta\boldsymbol{v}_m \tag{2.94}$$

$$\Delta\boldsymbol{v}_{scul,m} = \Delta\boldsymbol{v}_{scul,m-1} + \delta\boldsymbol{v}_{scul,m} \tag{2.95}$$

where

$$\boldsymbol{v}_m = \int_{t_{k-1}}^{t_m} \boldsymbol{a}(\tau)d\tau \qquad \text{and} \qquad \Delta\boldsymbol{v}_m = \int_{t_{m-1}}^{t_m} \boldsymbol{a}(\tau)d\tau$$

$$\Delta\boldsymbol{v}_{scul,m} = \frac{1}{2}\int_{t_{k-1}}^{t_m} \boldsymbol{\theta}(\tau) \times \boldsymbol{a}(\tau) + \boldsymbol{v}(\tau) \times \boldsymbol{\omega}(\tau)d\tau$$

$$\delta\boldsymbol{v}_{scul,m} = \frac{1}{2}\int_{t_{m-1}}^{t_m} \boldsymbol{\theta}(\tau) \times \boldsymbol{a}(\tau) + \boldsymbol{v}(\tau) \times \boldsymbol{\omega}(\tau)d\tau$$

The expression for $\delta \boldsymbol{v}_{scul,m}$ is reformulated to be computationally feasible. Begin by substituting Eq. (2.53) and Eq. (2.95) into $\delta \boldsymbol{v}_{scul,m}$, yielding

$$\begin{aligned}
\delta \boldsymbol{v}_{scul,m} &= \frac{1}{2} \int_{t_{m-1}}^{t_m} \boldsymbol{\theta}(\tau) \times \boldsymbol{a}(\tau) + \boldsymbol{v}(\tau) \times \boldsymbol{\omega}(\tau) d\tau \\
&= \frac{1}{2} \int_{t_{m-1}}^{t_m} \big(\boldsymbol{\theta}_m + \Delta\boldsymbol{\theta}(\tau)\big) \times \boldsymbol{a}(\tau) + \big(\boldsymbol{v}_m + \Delta\boldsymbol{v}(\tau)\big) \times \boldsymbol{\omega}(\tau) d\tau \\
&= \frac{1}{2}(\boldsymbol{\theta}_{m-1} \times \boldsymbol{v}_m + \boldsymbol{v}_{m-1} \times \Delta\boldsymbol{\theta}_m) + \frac{1}{2} \int_{t_{m-1}}^{t_m} (\Delta\boldsymbol{\theta}(\tau) \times \boldsymbol{a}(\tau) + \Delta\boldsymbol{v}(\tau) \times \boldsymbol{\omega}(\tau)) d\tau
\end{aligned}$$

Assume that $\boldsymbol{a}(t)$ may be approximated as a first-order power series on the domain $t \in [t_{m-1}\ t_m]$ as

$$\boldsymbol{a}(t) \approx \boldsymbol{a}_0 + \boldsymbol{a}_1(t - t_{m-1}) \tag{2.96}$$

The constants $\boldsymbol{a}_0$ and $\boldsymbol{a}_1$ can be solved for using a procedure similar to the one used to derive Eq. (2.63) and Eq. (2.64), yielding

$$\boldsymbol{a}_0 = \frac{1}{2T_m}\big(\Delta\boldsymbol{v}_m + \Delta\boldsymbol{v}_{m-1}\big) \tag{2.97}$$

$$\boldsymbol{a}_1 = \frac{1}{(T_m)^2}\big(\Delta\boldsymbol{v}_m - \Delta\boldsymbol{v}_{m-1}\big) \tag{2.98}$$

Substituting the first order approximations of $\boldsymbol{\theta}(t)$ and $\boldsymbol{a}(t)$ into the integral in $\delta \boldsymbol{v}_{scul,m}$ yields

$$\begin{aligned}
\delta \boldsymbol{v}_{scul,m} = \frac{1}{2} \int_{t_{m-1}}^{t_m} &\big[\boldsymbol{\omega}_0(\tau - t_{m-1}) + \frac{1}{2}\boldsymbol{\omega}_1(\tau - t_{m-1})^2\big] \times \big[\boldsymbol{a}_0 + \boldsymbol{a}_1(\tau - t_{m-1})\big] + \\
&\big[\boldsymbol{a}_0(\tau - t_{m-1}) + \frac{1}{2}\boldsymbol{a}_1(\tau - t_{m-1})^2\big] \times \big[\boldsymbol{\omega}_0 + \boldsymbol{\omega}_1(\tau - t_{m-1})\big] d\tau
\end{aligned}$$

Expanding the cross products and combining like terms yields

$$\delta \boldsymbol{v}_{scul,m} = \frac{1}{4} \int_{t_{m-1}}^{t_m} (\boldsymbol{a}_0 \times \boldsymbol{\omega}_1)(\tau - t_{m-1})^2 + (\boldsymbol{\omega}_0 \times \boldsymbol{a}_1)(\tau - t_{m-1})^2 d\tau$$

Lastly, the integral is evaluated and expressed in terms of $\Delta \boldsymbol{v}_m$, $\Delta \boldsymbol{v}_{m-1}$, $\Delta \boldsymbol{\theta}_m$, $\Delta \boldsymbol{\theta}_{m-1}$

$$\begin{aligned}
\delta \boldsymbol{v}_{scul,m} &= \frac{1}{12}\big((\boldsymbol{a}_0 \times \boldsymbol{\omega}_1)(t_m - t_{m-1})^3 + (\boldsymbol{\omega}_0 \times \boldsymbol{a}_1)(t_m - t_{m-1})^3\big) \\
&= \frac{\big(T_m\big)^3}{12}\big((\boldsymbol{a}_0 \times \boldsymbol{\omega}_1) + (\boldsymbol{\omega}_0 \times \boldsymbol{a}_1)\big) \\
&= \frac{1}{24}\big((\Delta \boldsymbol{v}_m + \Delta \boldsymbol{v}_{m-1}) \times (\Delta \boldsymbol{\theta}_m - \Delta \boldsymbol{\theta}_{m-1}) + \\
&\qquad\qquad (\Delta \boldsymbol{\theta}_m - \Delta \boldsymbol{\theta}_{m-1}) \times (\Delta \boldsymbol{v}_m + \Delta \boldsymbol{v}_{m-1})\big) \\
&= \frac{1}{12}\big(\Delta \boldsymbol{v}_{m-1} \times \Delta \boldsymbol{\theta}_m + \Delta \boldsymbol{\theta}_{m-1} \times \Delta \boldsymbol{v}_m\big)
\end{aligned}$$

Finally, $\delta \boldsymbol{v}_{scul,m}$ may be expressed as

$$\delta \boldsymbol{v}_{scul,m} = \frac{1}{2}\Big[\big(\boldsymbol{\theta}_{m-1} + \frac{1}{6}\Delta \boldsymbol{\theta}_{m-1}\big) \times \Delta \boldsymbol{v}_m + \big(\boldsymbol{v}_{m-1} + \frac{1}{6}\Delta \boldsymbol{v}_{m-1}\big) \times \Delta \boldsymbol{\theta}_m\Big] \quad (2.99)$$

The sculling equations are summarized

$$\boldsymbol{\omega}_m = \text{gyro output at time } t_m \text{ during } k^{th} \text{ update cycle} \tag{2.100}$$

$$\boldsymbol{a}_m = \text{accelerometer output at time } t_m \text{ during } k^{th} \text{ update cycle} \tag{2.101}$$

$$\boldsymbol{\theta}_m = \boldsymbol{\theta}_{m-1} + \Delta\boldsymbol{\theta}_m \tag{2.102}$$

$$\boldsymbol{v}_m = \boldsymbol{v}_{m-1} + \Delta\boldsymbol{v}_m \tag{2.103}$$

$$\Delta\boldsymbol{v}_{scul,m} = \Delta\boldsymbol{v}_{scul,m-1} + \delta\boldsymbol{v}_{scul,m} \tag{2.104}$$

$$\Delta\boldsymbol{v}_m = \frac{T_m}{2}(\boldsymbol{a}_m + \boldsymbol{a}_{m-1}) \tag{2.105}$$

$$\Delta\boldsymbol{\theta}_m = \frac{T_m}{2}(\boldsymbol{\omega}_m + \boldsymbol{\omega}_{m-1}) \tag{2.106}$$

$$\delta\boldsymbol{v}_{scul,m} = \frac{1}{2}\Big[\big(\boldsymbol{\theta}_{m-1} + \frac{1}{6}\Delta\boldsymbol{\theta}_{m-1}\big) \times \Delta\boldsymbol{v}_m + \big(\boldsymbol{v}_{m-1} + \frac{1}{6}\Delta\boldsymbol{v}_{m-1}\big) \times \Delta\boldsymbol{\theta}_m\Big] \tag{2.107}$$

$$\Delta\boldsymbol{v}_{ng,k} = \boldsymbol{v}_k + \frac{1}{2}(\boldsymbol{\theta}_k \times \boldsymbol{v}_k) + \Delta\boldsymbol{v}_{scul,k} \tag{2.108}$$

The algorithm is executed at the body update rate f_k. The batch of N gyro and accelerometer measurements over the time interval $[t_{k-1}\ t_k]$ are run through Eq. (2.102)-(2.110) to compute $\Delta\boldsymbol{v}_{scul,k}$. This quantity is then used in Eq. (2.110) to compute the change in velocity due to non-gravitational accelerations over the time interval $[t_{k-1}\ t_k]$. The coning and sculling algorithms are combined into the same update routine. The implementation of the algorithm is presented in Appendix A.

Chapter 3

Hardware Design and Implementation

3.1 MIMUC Design

3.1.1 Overview

The MIMUC consists of two circuit boards, the inertial measurement unit (IMU) board and the command and data handling (CDH) board. These two boards are stacked on top of each other to provide a complete inertial navigation system. The IMU cluster contains sixteen IMUs that measure specific force and angular rate. This data is collected by a 16-bit processor on the IMU board before being passed to the CDH board for processing. The CDH board calibrates the data, averages the angular rates/specific forces and outputs the average value over several different interfaces. The power consumption of the unit is 1 W. The mass of the IMU board is 40 g and the mass of the CDH board is 58 g. The volume of the unit is 9 cm x 9.5 cm x 3.2 cm, or 275 cm^3. The author emphasizes that these numbers are representative only of the prototype version and not the final production version.

3.1.2 IMU Circuit Board Design

The IMU board has sixteen MEMS IMUs integrated onto a single PCB. A 16-bit microcontroller communicates with each of the IMUs over an SPI bus. Angular rate, specific force and temperature data are collected from each sensor and combined into a single data packet. This data packet is sent to the CDH board over an I2C bus for calibration and processing. The IMU board also contains a low-noise 3.3 V linear voltage regulator to supply power to the sensors and microcontroller.

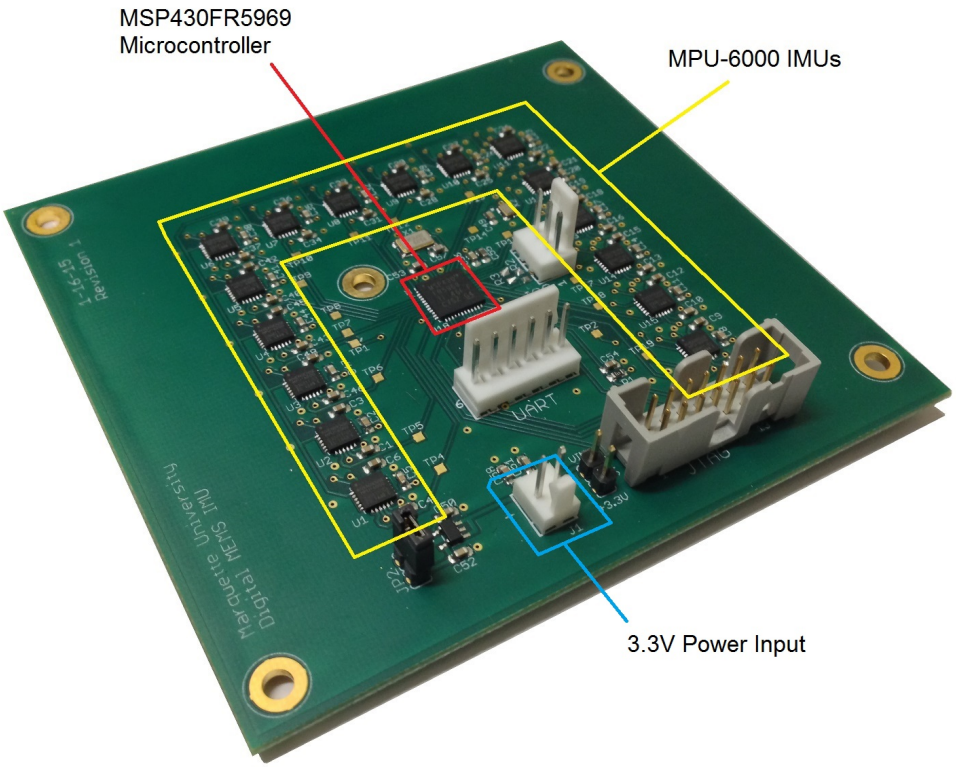

FIGURE 3.1: MIMUC PCB

3.1.2.1 Board Layout and Fabrication

The IMU circuit board layout was designed in EagleCAD 6.6.0 as shown in Fig. 3.2. The board consists of four separate layers of copper. From top to bottom, the layers are: signal, ground, power and a bottom signal layer. A four layer board was chosen to minimize the length and complexity of the traces. Following layout and verification of the design, the files were sent to a PCB manufacturer for fabrication. The PC/104 form factor of the PCB is based on the PC/104 standard. The form factor was designed to standardize dimensions in stackable embedded computer systems [11]. It is commonly used in CubeSats to allow interchangeability of components between different satellites. The dimensions of the circuit board are 9 cm x 9.6 cm, yielding a total surface area of 86.4 cm^2.

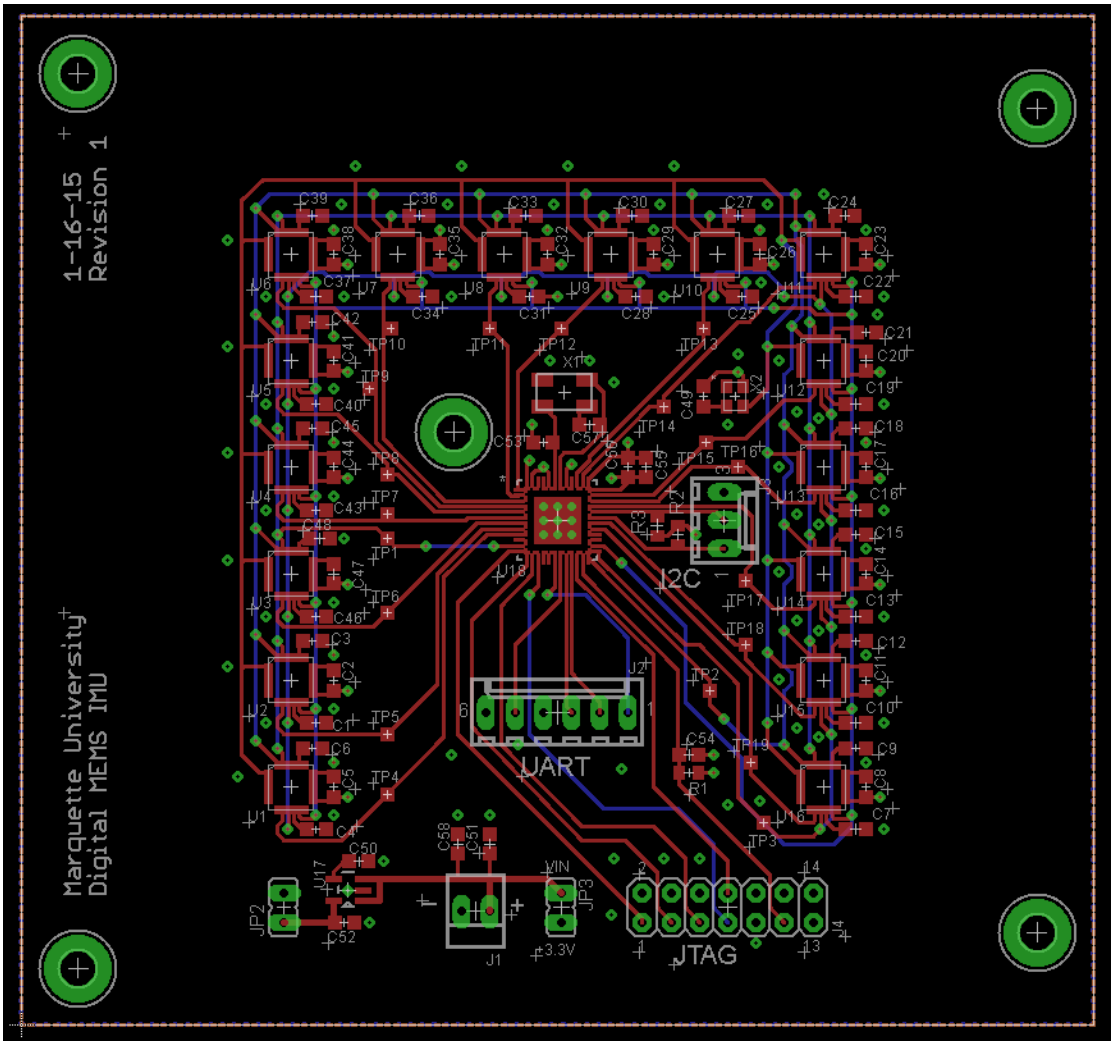

FIGURE 3.2: EagleCAD layout of MIMUC PCB

3.1.2.2 MEMS IMU

The heart of the MIMUC is the MPU-6000 IMU manufactured by Invensense. The MPU-6000 was chosen for its low noise, small form factor and low cost ($6 per sensor at the time of publication). The IMUs can be seen in Fig. 3.2 in a U-shaped configuration. The MPU-6000 contains three single-axis accelerometers and three single-axis gyros [20]. The maximum bandwidth of the MPU-6000 is 256 Hz for the gyro and 260 Hz for the accelerometer. Angular rate/specific force is measured by the gyroscope/accelerometer and converted into an analog voltage. This voltage is converted into a 16-bit digital value which is stored in the sensor registers. The MPU-6000 also has a built-in 16-bit temperature sensor. Data from

this sensor is incorporated into the calibration to compensate for bias drift due to changes in temperature.

The MPU-6000 supports two modes of serial communication, I2C and SPI. SPI is a bi-directional communication bus capable of very high throughput. SPI was chosen over I2C as it allows the sensor registers to be read at speeds up to 20 MHz (8 MHz was used in practice, a limitation of the microcontroller). The I2C bus is limited to a speed of 400 kHz. Since the SPI clock and data signals are high frequency, several design techniques were used to limit noise coupling and cross-talk between traces. A grounded copper fill was included in the top signal layer. Vias connecting the top layer copper fill and the ground layer were placed along the SPI signal lines to reduce coupling between traces carrying high frequency signals.

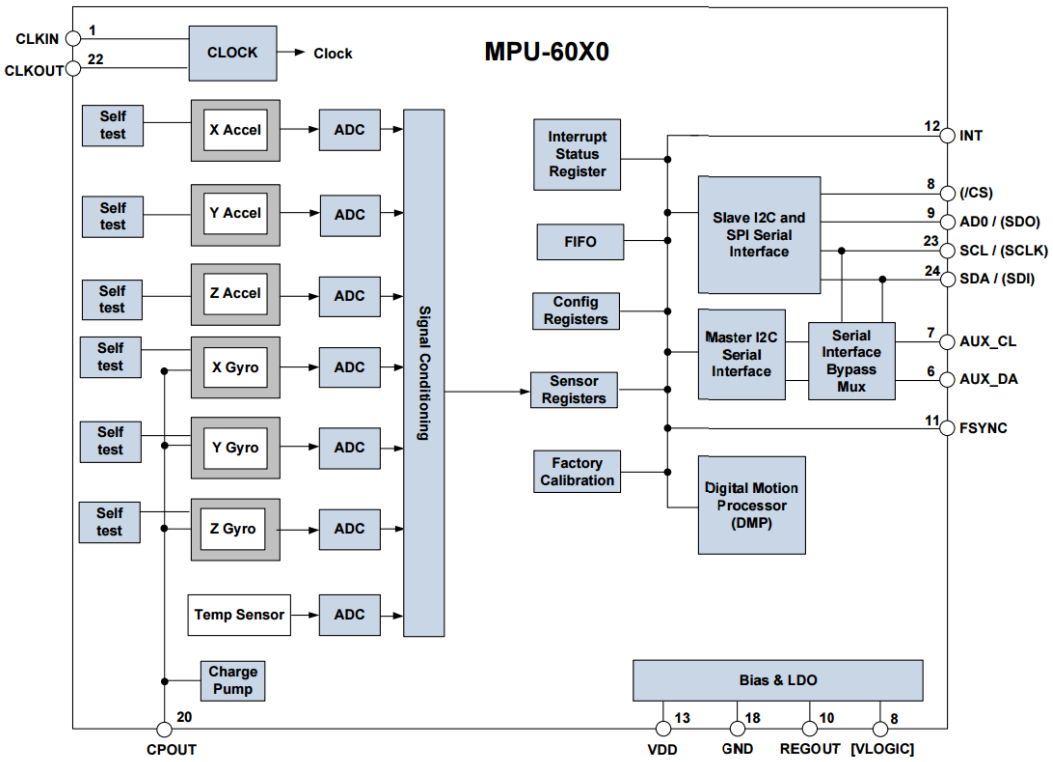

FIGURE 3.3: MPU-6000 block diagram [20]

Other features of the MPU-6000 include a built-in low pass filter, configurable full scale range of both the accelerometer and gyro, configurable sampling

rate as well as a variety of power-saving options. The complete block diagram of the sensor internals are shown in Fig. 3.3.

3.1.2.3 Microcontroller

The microcontroller used on the IMU board is the Texas Instruments MSP430FR5969, shown in Fig. 3.1. The MSP430 is an ultra-low power 16-bit microcontroller that operates at 16 MHz clock frequency [37]. This microcontroller was chosen because of its low power consumption–one of the key design goals of the MIMUC. The MSP430FR5969 has built in interfaces for UART, I2C and SPI. The SPI was used to communicate with the sensors while the I2C interfaces was left exposed to transfer data to the CDH board. A JTAG header was also added to allow reprogramming of the MSP430.

In hindsight, the MSP430FR5969 was a poor choice as it did not have enough processing power to handle the floating point calculations used in the calibration algorithm at the desired data rates. The discovery of this fact was the impetus for incorporating a second processing board containing a 32-bit processor into the PCB stack. In future design iterations, the 16-bit processor will be replaced with a 32-bit processor with sufficient power to handle the floating point calculations.

3.1.2.4 Voltage Regulation

Voltage regulation is handled by the Texas Instruments TPS73133. The TPS73133 is an LDO (low-dropout) linear voltage regulator [38]. The board receives power via a two-pin Molex header, shown in Fig. 3.1). The allowed input voltage is between 3.4 V and 5 V. A linear voltage regulator was chosen over a switching voltage regulator due to the concern that the switching regulator may introduce noise into the IMU outputs. Although switching regulators operate at frequencies

much higher than the bandwidth of the IMUs, the electromechanical elements within the IMU oscillate at frequencies on the order of 30-40 kHz [20]. The true impact of a switching voltage regulator on the IMU noise merits further study.

3.1.2.5 Other Features

The PCB also incorporates two jumpers, JP1 and JP2. JP1 allows the voltage regulator to be bypassed in the event that the voltage regulator fails or is not desired. The board must be provided with a low-noise, regulated 3.3 V in this case. JP2 connects the output of the voltage regulator to the power plane of the board. This allows the output of the voltage regulator to be verified before connecting it to the rest of the board components.

3.1.2.6 Software Design

The software on the MSP430 serves two purposes: communication with the sixteen IMUs and offloading the collected data to the CDH board. When IMU board is powered on, the MSP430 initializes each MPU-6000 with the proper settings. Steps in this initialization process include setting the reference clock to the on-board 19.2 MHz crystal oscillator and setting the sampling rate, digital low-pass filter frequency response and full scale range of both the gyroscope and accelerometer. A sampling rate of 1 kHz was chosen as this is the maximum sampling rate for the accelerometers (the gyroscopes can be sampled up to 8 kHz). The low-pass filter was set to the highest allowable bandwidth, which corresponds to a corner frequency of 260 Hz for the accelerometers and 256 Hz for the gyroscopes. The full scale ranges were set to the maximum values– $\pm$ 250 deg/s for the gyroscopes and $\pm$ 2 g for the accelerometers.

After initialization, the IMU board continuously acquires data from the sensors over the SPI bus. The MSP430 starts by reading the following values from

IMU 1 (in order): x, y, z-axis specific force, temperature, x, y, z-axis angular rate. Each value spans two 8 bit registers for a total of 14 bytes per IMU. Since there are a total of 16 IMUs, the total amount of data transferred per cycle is 224 bytes. The data is stored in the *primary* buffer. Once data has been acquired from all sixteen IMUs, a flag is raised indicating that the primary buffer has been filled. The primary buffer is then swapped with the *secondary* buffer to await transfer to the CDH board via I2C. The purpose of having two buffers is to prevent incoming data from the sensors from overwriting the outgoing data to the CDH board.

3.1.3 Command and Data Handling (CDH) Circuit Board Design

The purpose the CDH board is two-fold. The first task is to calibrate the data in real-time. The calibration coefficients for all sensors are determined offline and loaded into memory. These calibration coefficients are derived from the error models defined in Chapter 2. Secondly, the CDH board also handles communication with exporting the data over UART or storing it on an onboard SD card. The software can be configured to allow the output of raw sensor data or calibrated values. The CDH board is shown in Fig. 3.4.

3.1.3.1 Board Layout and Design

The CDH board was built using a PC/104 prototyping board. Unlike the MEMS IMUs, the performance of the CDH components are not sensitive to variations in board design. Prototyping board was used instead of a PCB to minimize costs. Board layout involved rearranging the components on the board until a desirable configuration was found.

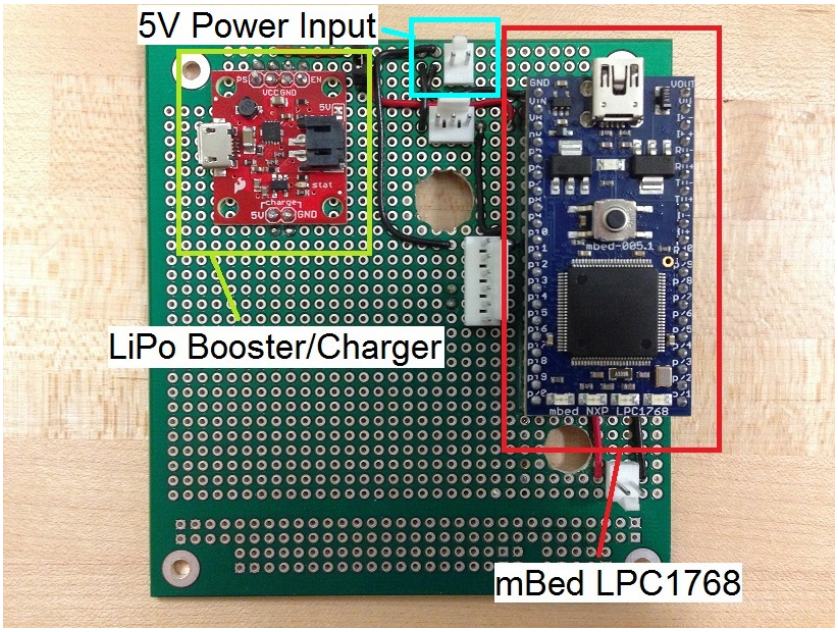

FIGURE 3.4: Command and data handling PCB

3.1.3.2 Microcontroller

The heart of the CDH board is the mBed NXP LPC1768 development platform. The LPC1768 includes a 32-bit ARM Cortex-3 core running at 96 MHz. Features include a real-time clock, three UART interfaces, two I2C buses and a USB interface for programming.

3.1.3.3 Power Supply

The CDH board incorporates a two-position jumper that allows switching between on board battery power or external 5V power. The on-board battery is a 2000 mA·hr 3.7 V Lithium Polymer (LiPo) battery. A fully charged battery will power the board for approximately 6-7 hours. The 5 V power can be provided via a two-pin Molex connector at the top of the board. In addition, the board also incorporates a combination LiPo booster and charging circuit purchased from SparkFun Electronics. The primary task is to boost the voltage of the battery from 3.7 V to 5 V, the required supply voltage for the LPC1768. It also allows

charging via micro USB. A second two-pin Molex connector, shown in Fig. 3.4, provides power to the IMU circuit board.

3.1.3.4 External Data Connections

Two external data connections are present on the CDH board. The first connection is an I2C bus which is connected directly to the IMU circuit board. Data is collected from the sensors and transmitted over this connection. Secondly, a UART connection is also available. This provides a direct connection to the mBed. It is primarily used for data transfer but also allows adjusting the MIMUC settings via command line. Both raw sensor data and calibrated sensor data can be transmitted at a user-selectable baud rate. The CDH board also includes a microSD card slot for long-term data storage. Data is transferred to the SD card via a SPI bus.

3.1.3.5 Software Design

Software for the mBed is written in C++ using the online mBed Compiler. This online service provides access to a number of libraries created by the user community and mBed developers for accessing the UART/I2C/SPI interfaces, configuring the power settings of the mBed, and using the real time clock and system timers.

Upon power-up, the CDH board loads the settings from a configuration file stored on the SD card. These settings allow the communication speeds and output modes to be adjusted without reprogramming. A second configuration file containing the calibration coefficients for the MIMUC is loaded, concluding the initialization stage.

The data acquisition process is done using an timer interrupt to ensure a constant data rate. The timer interrupt triggers a subroutine every X seconds; the desired data rate determines the value of X. The subroutine contains code which reads one data packet from the IMU board. This data packet contains the

angular rate and specific force measurements as well as the internal temperature of each sensor. The data is stored in a first-in first-out (FIFO) buffer to await transmission over one of the output interfaces. The main loop of the program calibrates the data and transmits it over the UART interface or stores it on the on-board SD card. The timer interrupt is critical when storing data on the SD card as SD write times may range from 1 ms to 250 ms long. The FIFO buffer therefore must be large enough to store 250 ms worth of incoming data from the IMU board. If the buffer happens to overflow, the most recent data is thrown out.

The CDH board can be configured to output either calibrated or uncalibrated data. For the uncalibrated output mode, the raw data from the IMU is encoded using COBS encoding [8]. For the calibrated data output mode, the data is first calibrated using Eq. (2.2) and Eq. (2.4). The data is then transferred over the UART interface and/or stored on the on-board SD card. At this time, coning and sculling algorithms have not yet been implemented in hardware.

The CDH board also has a command line interface which allows many of the software settings to be changed without reprogramming. The UART baud rate, output modes and calibration coefficients can all be modified. The command line interface also allows the user to read the software version number and print the calibration coefficients for any particular sensor.

3.2 Calibration

3.2.1 Testing Materials

To test the hardware design, a suitable apparatus is required to apply known specific forces and angular rates to the MIMUC. The output of the MIMUC can then be compared with the inputs to determine the error. The testing materials used for calibration and testing were a single-axis rate table (Trio-Tech Model 1102) with an adjustable mounting pivot (Thor Labs AP180 and Thor

Labs RP01). The complete testing apparatus is shown in Fig. 3.5. The calibration was performed in a laboratory at Marquette University. The rate table enables the MIMUC to rotate at a very accurate angular rates. This is required in order to solve for the gyroscope calibration coefficients. The normal force is used as the specific force input during calibration since the local gravity vector can be determined to high precision. The Thor Labs mounting hardware allows the IMU to be tilted between 0 deg and 90 deg with respect to gravity vector, as well as rotated from 0 deg to 360 deg about the Z axis of the MIMUC. The only challenge with the mounting hardware is that the board must be removed and mounted upside-down to test the negative Z axis of the MIMUC.

3.2.2 Calibration Procedure

Accurate calibration of inertial instrumentation is a difficult and complex task. Great care must be taken to ensure that accurate inputs are applied to the test article and that variables not under test remain constant (e.g. room temperature, external vibrations, etc.) The calibration procedures used for the gyros and accelerometers are defined in the IEEE Standards [26][25]. The MIMUC is mounted in six different orientations such that each principal axis is aligned both parallel and anti-parallel with the gravitational vector. At each orientation, data is acquired from the MIMUC at three different rates: $\pm\omega$ and zero rate. The input angular ω is chosen to be ± 40 deg/s, well within the operating range of the gyros. The length of data acquisition was two minutes. The error coefficients for the gyros and accelerometers can be solved as described in Section 2.4. Table 3.1 presents the testing matrix used during calibration. The calibration coefficients are given in Appendix B.

3.2.3 Temperature Calibration

Temperature calibration was performed in a Ransco Model SD-60 thermal chamber. The MIMUC was initially cooled down to a temperature of -10°C. The

Test #	Input Rate (deg/s)	Direction	Up-Axis	Duration (s)
1	0	-	+X	120
2	0	-	-X	120
3	0	-	+Y	120
4	0	-	-Y	120
5	0	-	+Z	120
6	0	-	-Z	120
7	40	CW	+X	120
8	40	CW	-X	120
9	40	CW	+Y	120
10	40	CW	-Y	120
11	40	CW	+Z	120
12	40	CW	-Z	120
13	40	CCW	+X	120
14	40	CCW	-X	120
15	40	CCW	+Y	120
16	40	CCW	-Y	120
17	40	CCW	+Z	120
18	40	CCW	-Z	120

TABLE 3.1: MIMUC calibration test matrix

cooling source was removed and data collection was initiated. Data was collected for 15 minutes as the MIMUC warmed up to room temperature. The output of each sensor was then plotted versus temperature and a linear regression was fit to the data to determine the temperature coefficient of each sensor.

3.3 Experiment Design

A set of experiments was designed to compare the performance of the MIMUC with a single IMU in terms of standard performance metrics (discussed below) as well as performance in a dead-reckoning scenario. The error models from Section 2.1 were also simulated in MATLAB to provide a baseline comparison. The coning and sculling algorithms performance was also compared with trapezoidal integration at the body rate.

The IMU performance was quantified by measuring bias stability (for both gyro and accelerometer), ARW and VRW. These parameters can be determined from the Allan deviation plot of the gyro/accelerometer outputs. The MIMUC

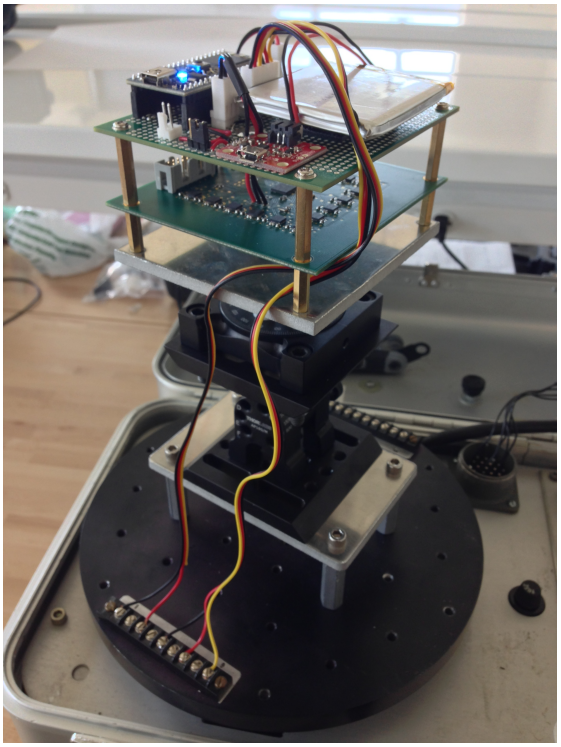

FIGURE 3.5: Calibration setup– MIMUC mounted to single-axis rate table

FIGURE 3.6: Ransco Model SD-60 thermal chamber

was placed on a level surface and data was collected for 8 hours with the unit at rest. The sampling rate was 76.3 Hz, the maximum allowed rate when sending uncalibrated data over UART. The author emphasizes that during real operation, the IMU would be outputting calibrated data, allowing a much higher data rate due much less data being transferred (34 bytes per packet vs. 230 bytes per

packet). The Z axis was aligned anti-parallel with the gravity vector. The room temperature was controlled to stay within 1°C during the duration of the test. The 8 hour acquisition time was chosen to accurately characterize the bias stability of the gyros and accelerometers.

To measure the dead-reckoning performance, the MIMUC was placed on a level surface and data was collected for five minutes with the unit at rest. The Z axis was aligned anti-parallel with the gravity vector. The IMU outputs were integrated using trapezoidal integration to determine the velocity, position and attitude over the five minute interval. If the experiment was performed with an ideal IMU (no calibration errors, no noise, perfect bias stability), the expected output would be zero velocity, zero position and zero attitude (excluding the rotation of the Earth) for all time. Since the MIMUC is not an ideal system, these quantities will drift from zero over time. The velocity, position, and attitude errors were compared between the three separate sources: (1) the MIMUC, (2) a single MPU-6000, (3) a "virtual" MIMUC.

The virtual MIMUC was generated in MATLAB via simulation. Woodman [41] provides a model for simulating a gyro/accelerometer using the ARW/VRW and the bias stability. The ARW/VRW and bias stability values from Table 2.1 for the Single IMU column were used to generate the output of sixteen separate virtual IMU sensors. The sample rate of 76.3 Hz was chosen to match the actual MIMUC sample rate. The output of each of these virtual sensors was then averaged together to create a virtual MIMUC. Since the MPU-6000 is a digital sensor, the calibrated output is a quantized value. The simulated outputs were quantized to match the MPU-6000. Calibration errors were not included in the simulated MIMUC. The objective is to use the simulated model as the performance benchmark under ideal calibration.

The coning and sculling algorithms were tested via simulation in MATLAB. The specific forces and angular velocities that result in the coning/sculling motion

were generated in simulation. For the coning simulation, the coning angle and frequency of oscillation were varied to determine the effect of each of these parameters on the norm of the attitude error. For the sculling simulation, the angular oscillation amplitude, linear oscillation amplitude and oscillation frequency were varied to determine the effect of each of these parameters on the error in position. Each scenario was simulated using both the coning and sculling algorithms, as well as trapezoidal integration. The errors between the two methods were compared.

Chapter 4

Results

4.1 Performance Comparison

The main objective of this thesis is to compare the performance of a MEMS IMU cluster versus a single MEMS IMU. Several metrics commonly used for quantifying the performance of IMUs are utilized. The first metric is the bias of the sensor. This value is the average output of the sensor when no input is applied. As mentioned previously, these errors typically arise due to inaccuracies of the calibration process. The second metric is bias stability, which provides a measure of the bias drift over a certain period of time. The final metric is the ARW/VRW, which provides a measure of the random noise present at the output of the sensor.

4.1.1 Bias Error

The bias error was determined by taking the time average of five minutes of data with the MIMUC at rest. Tables 4.1 and 4.2 present the bias errors of a single MPU-6000 compared with the MIMUC. The bias error for the single MPU-6000 was determined by choosing the sensor with the largest magnitude bias error for each axis.

Sensor Axis	MPU-6000 (deg/s)	MIMUC (deg/s)
X	−0.151	−0.067
Y	+0.124	+0.009
Z	−0.340	−0.053

Table 4.1: Gyro bias errors

Sensor Axis	MPU-6000 (mg)	MIMUC (mg)
X	+4.319	+3.012
Y	−3.351	−1.795
Z	−6.015	−2.194

Table 4.2: Accelerometer bias errors

The simulated MIMUC has zero bias errors since no calibration errors were included in the simulation model.

4.1.2 Bias Stability

The bias stability is determined by taking the long term static output of the gyro/accelerometer and generating the Allan deviation plot. For a discussion on how the Allan deviation plot is generated, refer to Appendix C. The bias stability is the minimum of the plot and the averaging time τ is that time at which the minimum occurs [13]. The Allan deviation plot for the MIMUC was computed from the ensemble average of the sixteen IMUs. The Allan deviation for a single MPU-6000 was determined by taking the Allan deviation plot for each MPU-6000 and averaging the plots together. Fig. 4.1 and 4.3 show the Allan deviation plot for the gyros and accelerometers, respectively. The Allan deviation for the single MPU-6000 is shown by the thin traces while the MIMUC is shown with thick traces.

In addition to the Allan deviation plots of the experimentally gathered data, the virtual MIMUC Allan deviation was plotted as well (Fig. 4.2 and 4.4). Tables 4.3 and 4.4 summarize the bias stability results for the gyros and accelerometers, respectively, for the single MPU-6000, MIMUC and simulated MIMUC.

Axis	MPU-6000	Virtual MPU-6000	MIMUC	Virtual MIMUC
X	4.86	1.72	1.17	0.40
Y	5.28	1.73	1.30	0.39
Z	3.90	1.74	0.96	0.46

TABLE 4.3: Gyroscope bias stability (deg/hr)

Axis	MPU-6000	Virtual MPU-6000	MIMUC	Virtual MIMUC
X	26.8	23.1	6.0	4.6
Y	24.3	21.8	6.2	4.5
Z	56.6	22.2	26.5	5.8

TABLE 4.4: Accelerometer bias stability (µg)

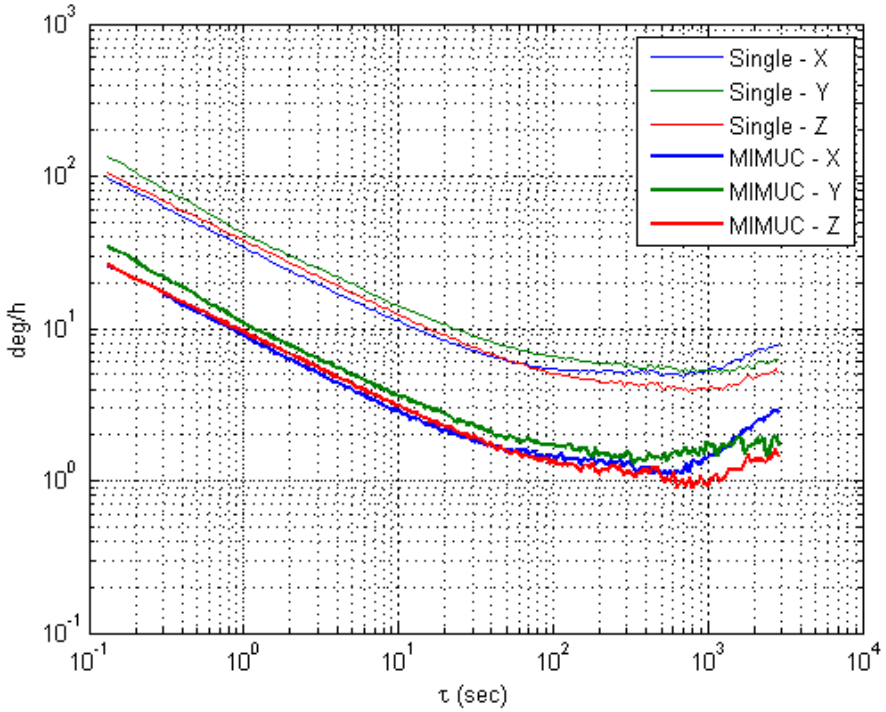

FIGURE 4.1: Gyro Allan deviation

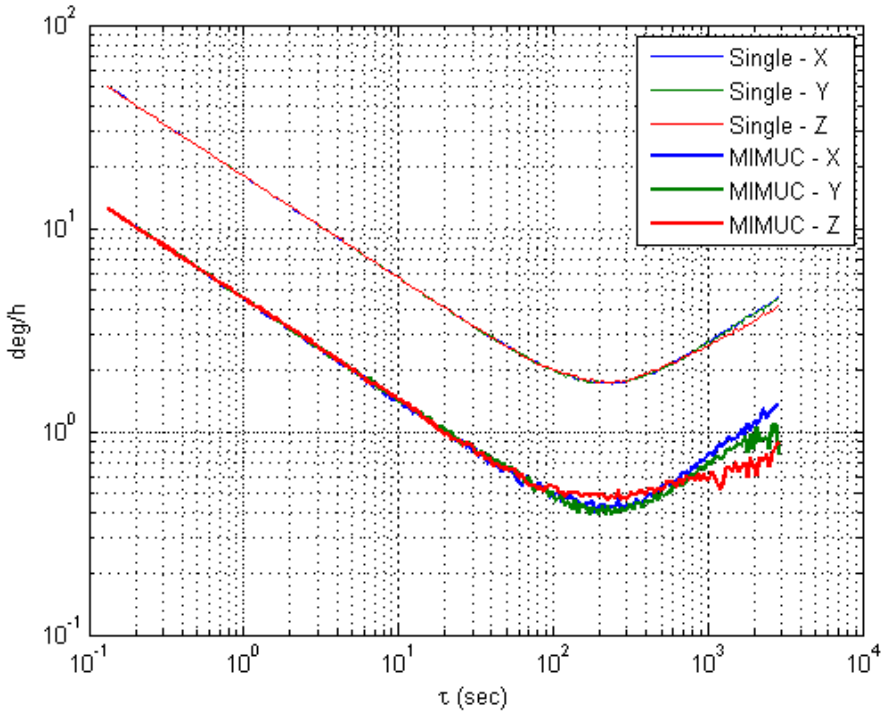

FIGURE 4.2: Virtual gyro Allan deviation

4.1.3 Angle/Velocity Random Walk

The ARW/VRW is determined by reading off value of the Allan deviation plot at $\tau = 1$ s [13]. Tables 4.5 and 4.6 summarize the angle random walk and

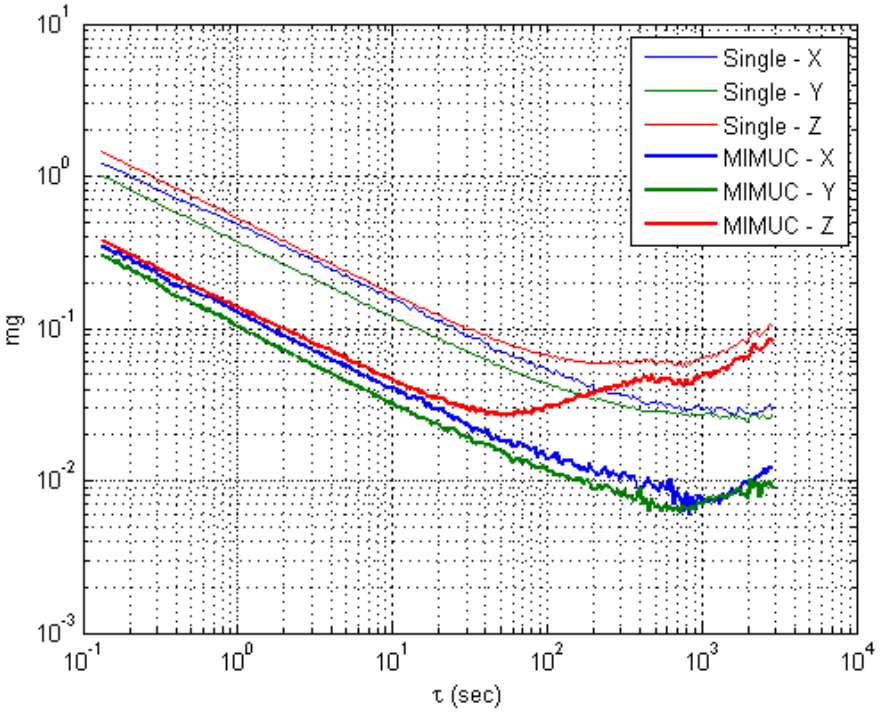

FIGURE 4.3: Accelerometer Allan deviation

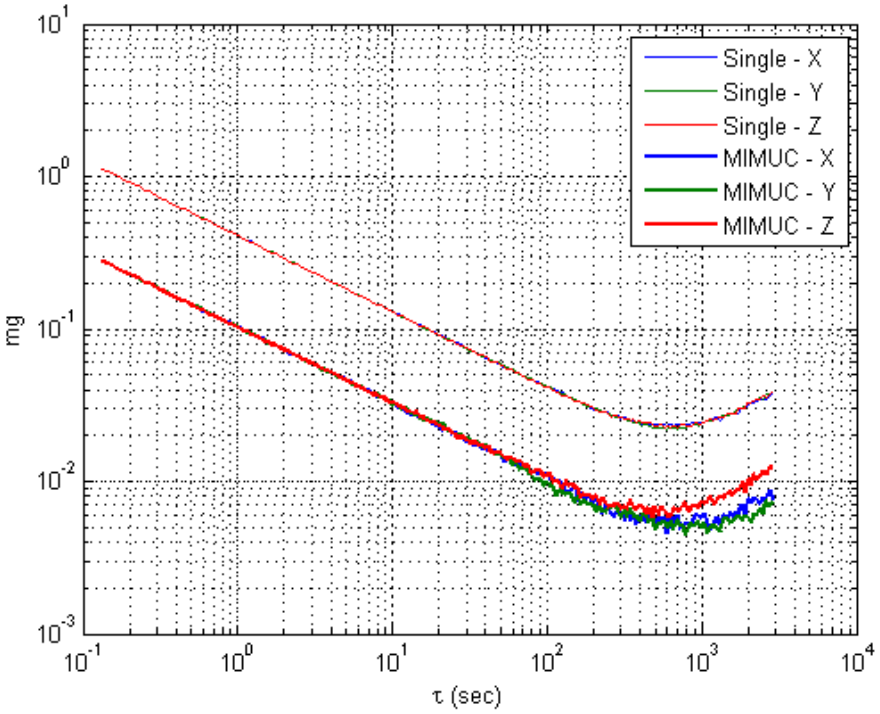

FIGURE 4.4: Virtual accelerometer Allan deviation

velocity random walk, respectively. The values were determined from the same Allan deviation figures used in Section 4.1.2 (Fig. 4.1, 4.2, 4.3, 4.4).

Axis	MPU-6000	MIMUC	Virtual MPU-6000	Virtual MIMUC
X	0.631	0.303	0.166	0.075
Y	0.776	0.303	0.201	0.076
Z	0.693	0.303	0.173	0.076

TABLE 4.5: Gyro ARW ($\deg/\sqrt{\text{hr}}$)

Axis	MPU-6000	MIMUC	Virtual MPU-6000	Virtual MIMUC
X	0.309	0.243	0.083	0.060
Y	0.239	0.242	0.069	0.061
Z	0.341	0.242	0.089	0.061

TABLE 4.6: Accelerometer VRW ($\text{m/s}/\sqrt{\text{hr}}$)

4.1.4 Dead Reckoning

The dead-reckoning errors over the five minute navigation period are plotted in Fig. 4.5 and 4.6. The leftmost graphs show the measured MIMUC output over the 5 minute dead-reckoning period. The other graphs indicate the attitude errors (Fig. 4.5) and velocity and position errors (Fig. 4.6). Each MPU-6000 is plotted as a dashed red line; the MIMUC is plotted as a bold blue line; the virtual MIMUC is plotted as a bold green line. The errors in velocity, position and attitude after the five minute period are presented in Table 4.7. Acceleration due to gravity was subtracted from the Z axis accelerometer output before running the dead-reckoning algorithm. The author would like to reiterate that the simulated MIMUC does not include any calibration errors (i.e. the gyro and accelerometer outputs have zero bias offset).

The first part of Table 4.7 summarizes the attitude errors at the end of the navigation period. The velocity and position errors along each axis are listed below the attitude errors. For the single IMU case, the MPU-6000 with the largest error was chosen from each axis.

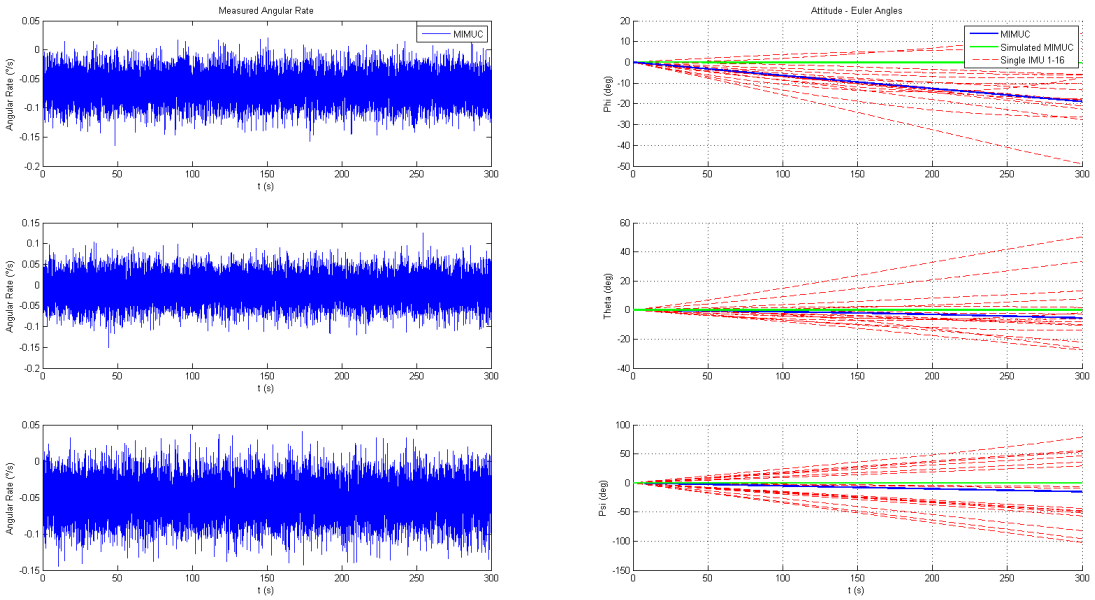

FIGURE 4.5: Dead reckoning attitude errors

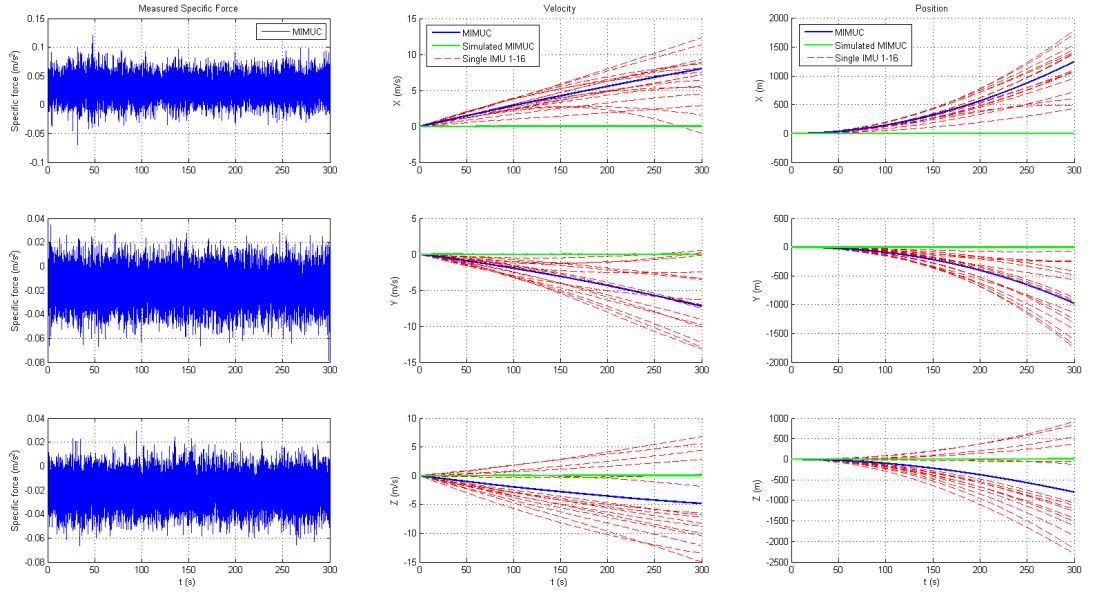

FIGURE 4.6: Dead reckoning velocity and position errors

4.2 Simulation

4.2.1 Coning

The coning algorithms were evaluated via simulation. The angular rate vector that produces a coning motion is given by

$$\boldsymbol{\omega}(t) = \begin{pmatrix} \omega_c \sin\theta_c \cos\omega_c t \\ -\omega_c \sin\theta_c \sin\omega_c t \\ \omega_c(1-\cos\theta_c) \end{pmatrix} \tag{4.1}$$

Attitude Error (deg)	ϕ	θ	ψ
MPU-6000 (Max)	48.9	50.2	102.4
MIMUC	18.9	5.5	15.1
Virtual MIMUC	0.1	0.1	0.1
Velocity Error (m/s)	X	Y	Z
MPU-6000 (Max)	12.3	13.1	15.0
MIMUC	8.0	7.2	4.8
Virtual MIMUC	∼0.0	∼0.0	∼0.0
Position Error (m)	X	Y	Z
MPU-6000 (Max)	1794.5	1748.2	2291.7
MIMUC	1251.7	982.7	801.5
Virtual MIMUC	3.9	2.1	4.8

TABLE 4.7: Comparison of dead reckoning attitude, velocity and position errors for MPU-6000, MIMUC and virtual MIMUC

where ω_c is the coning frequency in rad/s and θ_c is the coning angle in radians. The angular rate was integrated using two different integration schemes: coning vs. trapezoidal integration. Three separate scenarios were evaluated to test the efficacy of the coning algorithm under varying coning angle and frequency (Table 4.8). The coning update rate was chosen to be 1 kHz as this is the maximum possible output rate of the MPU-6000. The body rate was arbitrarily chosen to be 50 Hz. The simulations were run for 10 seconds. Fig. 4.7, 4.8 and 4.9 show the results of the simulation. Table 4.9 shows the norm of the attitude error of the two different integration schemes for the various coning angles/frequencies.

Scenario		1	2	3
Coning Angle	deg	1.0	10.0	1.0
Coning Frequency (f_c)	Hz	1.0	1.0	10.0

TABLE 4.8: Coning scenario parameters

Scenario	Coning (deg)	Trapezoid (deg)
1	3.7878E-6	0.0014
2	3.7499E-4	0.1431
3	3.7934E-3	1.3360

TABLE 4.9: Attitude error for coning scenarios

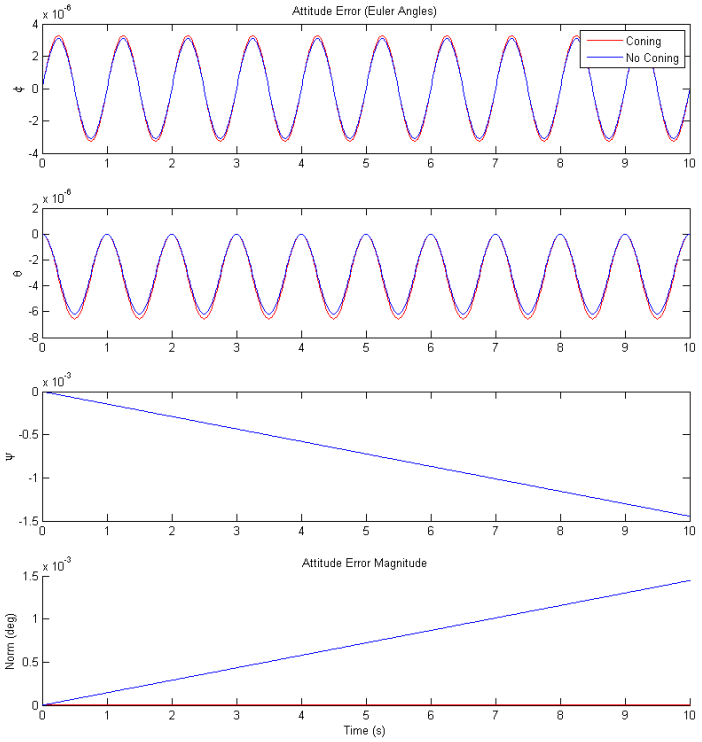

FIGURE 4.7: Attitude errors for coning scenario 1

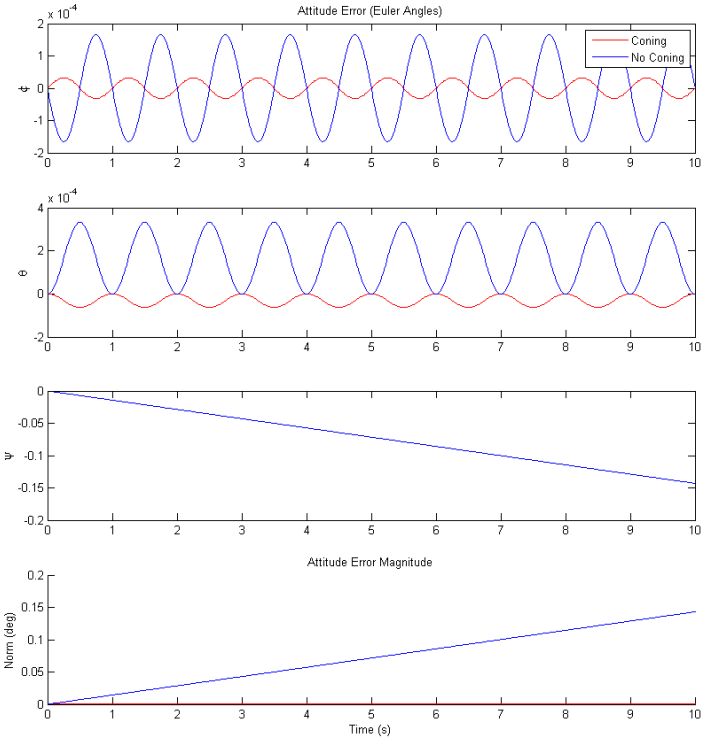

FIGURE 4.8: Attitude errors for coning scenario 2

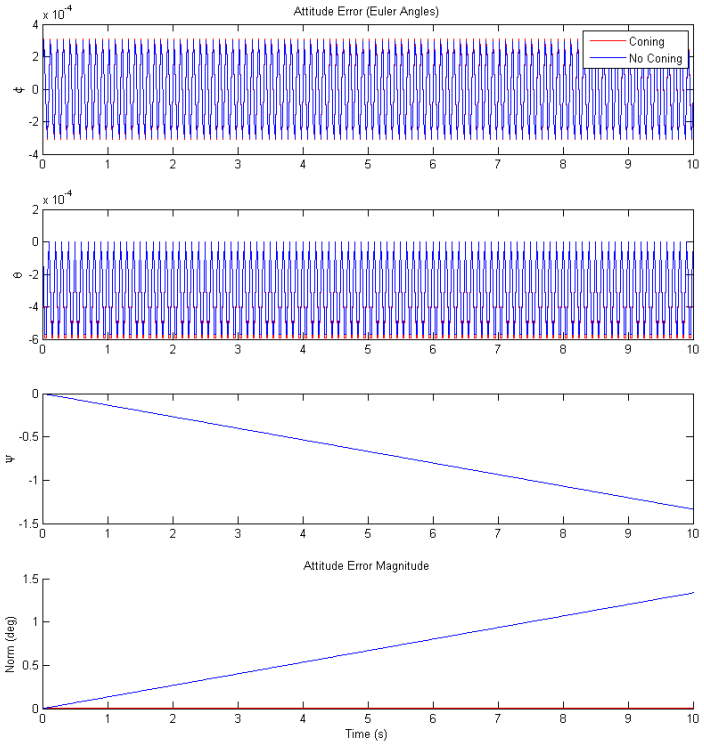

FIGURE 4.9: Attitude errors for coning scenario 3

4.2.2 Sculling

The sculling algorithms were evaluated via simulation. Four separate scenarios were evaluated to test the efficacy of the algorithm under varying angular and linear oscillation amplitude and oscillation frequency. The angular rate vector and acceleration vector that produces the sculling motion is defined by

$$\boldsymbol{a}(t) = \begin{pmatrix} 0 \\ 0 \\ A\sin(\omega_s t) \end{pmatrix} \qquad \text{and} \qquad \boldsymbol{\omega}(t) = \begin{pmatrix} 0 \\ a\omega_s \sin \omega_s t \\ 0 \end{pmatrix}$$

where A is the magnitude of the specific force oscillation (g), a is the magnitude of the angular oscillation (rad/s) and ω_s is the frequency of the sculling motion (rad/s). The following scenarios were tested

Scenario	Units	1	2	3	4
Angular Amplitude	deg	0.1	1.0	0.1	0.1
Linear Amplitude	g	1.0	1.0	10.0	1.0
Frequency (f_s)	Hz	1.0	1.0	1.0	10.0

TABLE 4.10: Sculling scenario parameters

The body update rate was chosen to be 50 Hz and the sculling update frequency 1 kHz. The sculling update rate was chosen to be 1 kHz as this is the maximum possible output rate of the MPU-6000. The body rate was chosen arbitrarily. The simulations were run for 10 seconds. Two different integration schemes are plotted: one with sculling, the other with trapezoidal integration at the body rate. Fig. 4.10, 4.11, 4.12 and 4.13 show the results of the simulation. Table 4.11 shows the norm of the position error of the two different integration schemes for the various sculling scenarios.

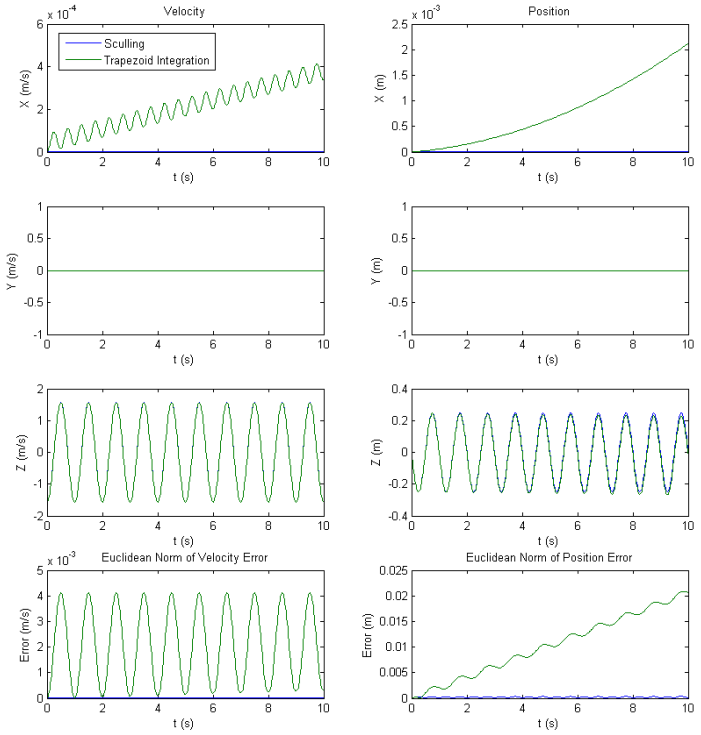

FIGURE 4.10: Velocity and position errors for sculling scenario 1

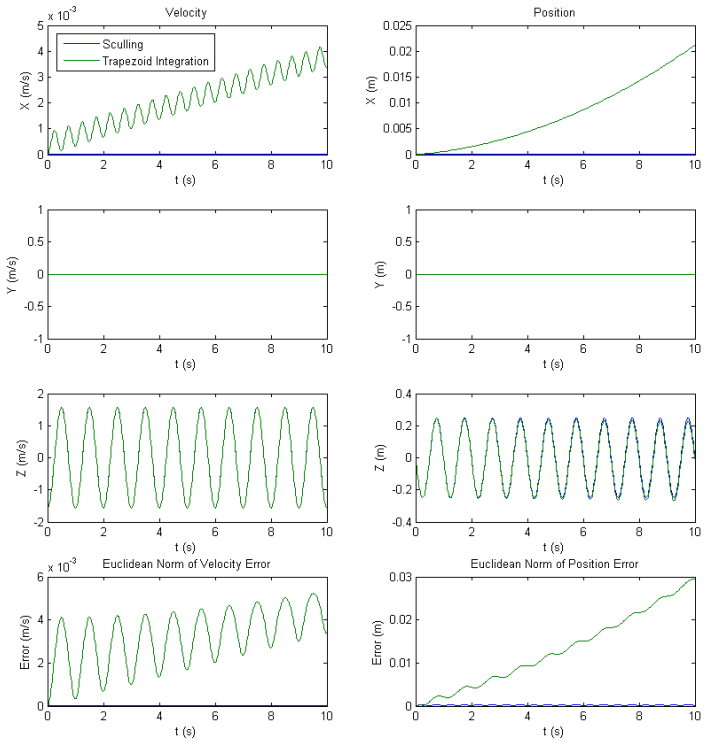

FIGURE 4.11: Velocity and position errors for sculling scenario 2

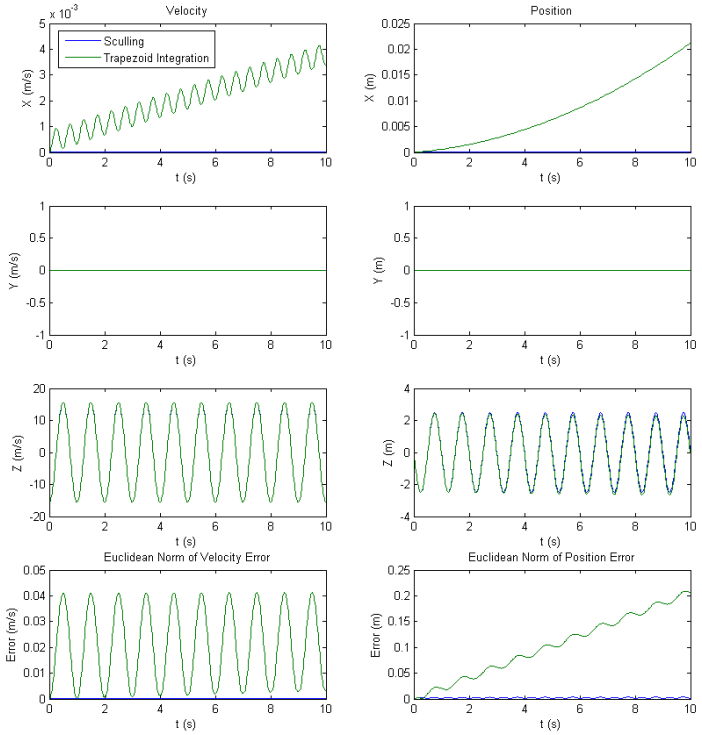

FIGURE 4.12: Velocity and position errors for sculling scenario 3

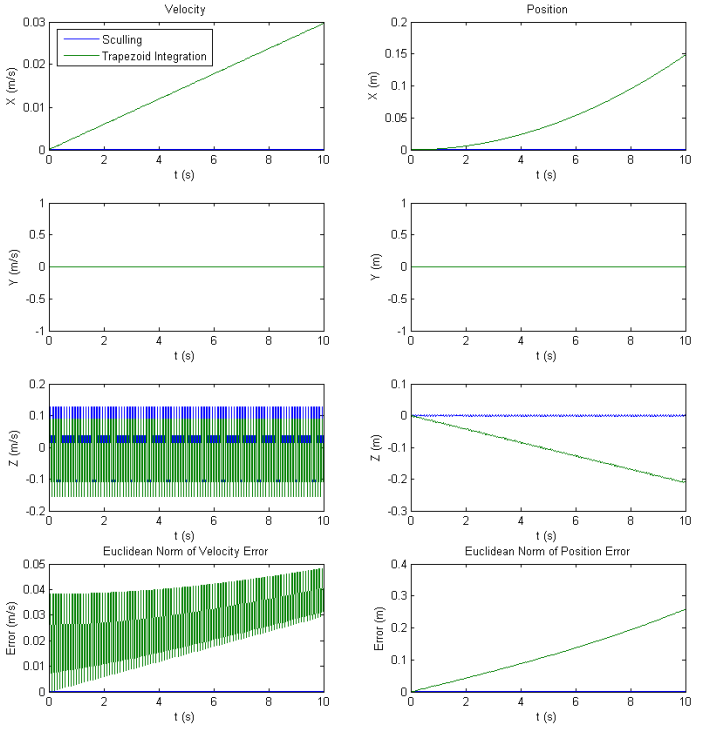

FIGURE 4.13: Velocity and position errors for sculling scenario 4

	Norm of Velocity Error (m/s)		Norm of Position Error (m)	
Scenario	Sculling	Trapezoid	Sculling	Trapezoid
1	5.915E-7	3.378E-4	5.141E-5	0.021
2	5.916E-6	3.378E-3	5.570E-5	0.030
3	5.916E-6	3.378E-3	5.141E-4	0.207
4	5.919E-5	0.0296	5.924E-4	0.258

TABLE 4.11: Velocity and position error for sculling scenarios

Chapter 5

Discussion of Results

5.1 MIMUC Performance

5.1.1 Bias Error

Because the bias error is a result of the calibration process and not a stochastic error, it is not expected that the bias error will be reduced by the square root of the number of sensors. Since all of the IMUs were calibrated simultaneously, any errors in the calibration apparatus would appear on each IMU. Table 5.1 presents the X axis accelerometer bias errors for all sixteen MPU-6000 sensors. Recall the X axis bias error of the MIMUC was 3.0123 mg. The bias errors below this value have been bolded. From the table, it is evident that the bias errors are not distributed about zero. Averaging of the bias errors will cause the MIMUC to take on the mean of the distribution of the single sensor bias errors. In order to improve the MIMUC bias error, the calibration process must be made more accurate such that the mean of the distribution of the single sensor bias errors is approximately zero.

Sensor #	1	2	3	4	5	6	7	8
Bias (mg)	**1.1347**	**1.7007**	**2.1947**	**2.7754**	3.0334	3.7556	4.0203	4.2385
	9	10	11	12	13	14	15	16
	4.1449	4.3190	3.2053	3.1572	**2.7575**	**2.7322**	**2.4309**	**2.5967**

TABLE 5.1: Accelerometer bias errors (X axis)

5.1.2 Bias Stability

The bias stability of the sensors saw a significant improvement through averaging. In Section 2.2, it was predicted that averaging N sensors would reduce the bias stability by $1/\sqrt{N}$. Since sixteen MPU-6000 IMUs were used, a four times

reduction was expected. Beginning with the gyro bias stability, the performance ratio was 4.153, 4.06 and 4.06 for the X, Y and Z axes, respectively. The *performance ratio* is defined as the ratio of the single IMU bias stability and the MIMUC bias stability. This result confirms the predicted performance improvement.

The simulated gyro bias stability is a factor of two lower than the experimental value (Table 4.3). Since the MPU-6000 datasheet did not specify the bias stability of the IMU, it is difficult to determine if the IMU is performing as expected or if the simulation model is optimistic. The simulated gyro performance ratio was 4.30, 4.44 and 3.78 for the X, Y and Z axes, respectively. This is consistent with the experimental results.

For the accelerometers, the bias stability performance ratio was 4.47, 3.92 and 2.13 for the X, Y and Z axes, respectively. The X and Y axes show the expected improvement in performance. However, the Z axis shows a considerably smaller improvement in the bias stability. Recalling that the MIMUC was placed with the Z axis aligned anti-parallel with the gravity vector during the static test, this result indicates that the expected performance improvement does not hold when an axis is under load. In addition to a smaller reduction in bias stability through averaging, the Z axis also shows $\sim$2.3 times increase in bias stability as compared to the X and Y axes. The exact mechanism which causes this increase is not currently understood. One possible explanation is that placing the accelerometer under load induces a preference in the direction of the bias drift.

The simulated accelerometers more accurately predict the true accelerometer bias stability than the gyros. The simulated accelerometer slightly outperform the experimental data. This is expected since the simulation model does not account for all possible error/noise sources. The ratio of the simulated MPU-6000 and simulated MIMUC accelerometer bias stability was 5.02, 4.84, and 3.82 for the X, Y and Z axes, respectively.

5.1.3 Angle/Velocity Random Walk

The ARW/VRW also saw a significant improvement from averaging. In Section 2.2, it was predicted that averaging N sensors would scale the ARW/VRW of the MIMUC by $1/\sqrt{N}$. Since sixteen MPU-6000 IMUs were used, a four times reduction in the output noise was expected. Beginning with ARW, the improvement for the X, Y and Z axes was 3.80, 3.86, and 4.01, respectively. This result also suggests that there is little to no correlation in the ARW between individual sensors. To confirm this hypothesis, the X axis output of each sensor was correlated with the other fifteen sensors. The process was repeated for the Y and Z axes. For all pairs of sensors, the correlation coefficient ρ did not exceed 0.04, indicating no statistical correlation between any of the sensors.

The ARW of the simulated MPU-6000 exactly matched the predicted ARW of 0.3 deg/$\sqrt{\text{hr}}$ (Section 2.2). The simulated MIMUC also showed a four times reduction in the ARW compared to a single MPU-6000, consistent with both the prediction and the experimental data. One interesting observation from Table 4.5 is that the ARW of the MPU-6000 is more than double the expected value. One possible explanation is that the datasheet specifies the rate noise density of the gyroscopes for 10 Hz. This figure therefore may not accurately describe the noise of the gyro across all frequencies.

The VRW performance ratios are 3.72, 3.46, and 3.83 for the X, Y, and Z axes, respectively. The VRW showed slightly less improvement via averaging when compared with the ARW. This suggests that the random noise at the output of each accelerometer is slightly correlated with the other sensors. Unlike the bias stability, the VRW of the Z axis accelerometer did not seem to be affected by the axis being under load.

Similar to the gyros, the correlation coefficients were computed for each pair

of accelerometers for each axis. The correlation coefficients a small but statistically significant correlation. The X axis gyroscopes exhibited positive correlation coefficients that ranged from $\rho = 0.14$ to $\rho = 0.24$. Similarly, the Y axis gyroscopes also exhibited positive correlation coefficients that ranged from $\rho = 0.10$ to $\rho = 0.17$. The Z axis accelerometers, however, showed no statistical correlation, with $\rho < 0.02$ for all pairs of accelerometers. This is an interesting result, indicating the resistance of the accelerometer bias stability to averaging is not due to correlation between accelerometers. The correlation coefficients do support the hypothesis that the output noise of the accelerometers is somewhat correlated.

The VRW of the simulated MPU-6000 exactly matched the expected VRW of 0.024 m/s/$\sqrt{\text{hr}}$ predicted in Section 2.2. The simulated MIMUC also showed a four times reduction in the VRW compared to a single MPU-6000, consistent with both the predicted result and the experimental data.

5.1.4 Dead Reckoning

Fig. 4.5 4.6 show the accumulation in attitude, velocity and position error during 300 s of dead-reckoning navigation. Table 4.7 summarizes the errors at the end of the five minute period. Beginning with the attitude error, the single MPU-6000 accumulated significant errors during the five minute period- the attitude errors were 48.9 deg, 50.2 deg, and 102.4 deg for the X, Y, and Z axes, respectively. In comparison, the MIMUC exhibited significant errors as well but substantially improved over the single IMU case- 12.3 deg, 13.1 deg, and 15.0 deg for the X, Y, and Z axes, respectively. The improvement seen due to averaging was highly dependent on the accuracy of the calibration. In Fig. 4.5, the X axis gyros all show a negative linear trend, suggesting a negative bias error common to all gyros. Contrarily, the Y and Z gyros have a more even distribution of positive and negative biases. This resulted in the MIMUC having a constant bias error near zero. The attitude errors of the virtual MIMUC were near zero relative to the

MPU-6000 and the true MIMUC- no single axis error exceeded 0.074 deg. This suggests that the major contributor to attitude error is calibration errors. These numbers also give some indication into the type of performance possible with the MIMUC.

The velocity and position errors were consistent with the attitude errors. After 300 s, the MPU-6000 exhibited position errors greater than 1.7 km on each axis (Table 4.7). Averaging was insufficient in reducing the errors to an acceptable level; the MIMUC exhibited position errors of 1.25 km, 983 m, and 802 m for the X, Y, and Z axes, respectively. As per the case with the attitude errors, the dominant source of error was due to bias errors in the accelerometer output. The simulated MIMUC confirms this result- after five minutes of dead-reckoning, none of the position errors exceeded 4.8 m.

To more precisely illustrate the impact of each error source (bias error, random noise, bias random walk) on the dead-reckoning performance, the virtual MIMUC was simulated by independently removing each error source and performing dead-reckoning navigation for 300 s. The bias errors of the virtual MIMUC were set equal to the constant bias errors of the MIMUC (Tables 4.1 and 4.2). Fig. 5.1 presents the comparison with the various error sources removed. As expected, the constant bias error is by far the largest contributor to the dead-reckoning error. The norm of the position error with the bias removed was 5.864 m, 1967.4 m with the random noise removed, 1967.0 m with the bias walk removed, and 1965.8 m with all error sources included.

The dead-reckoning experiments demonstrate that the most significant source of error by far is bias errors resulting from improper calibration. Calibration requires knowing the input rates and forces to a high degree of precision. For example, a gyro with a bias on the order of 0.01 deg/hr requires mounting accuracies on the order of ten arcseconds [39]. That being said, the focus of this thesis was not to create a precision calibration apparatus.

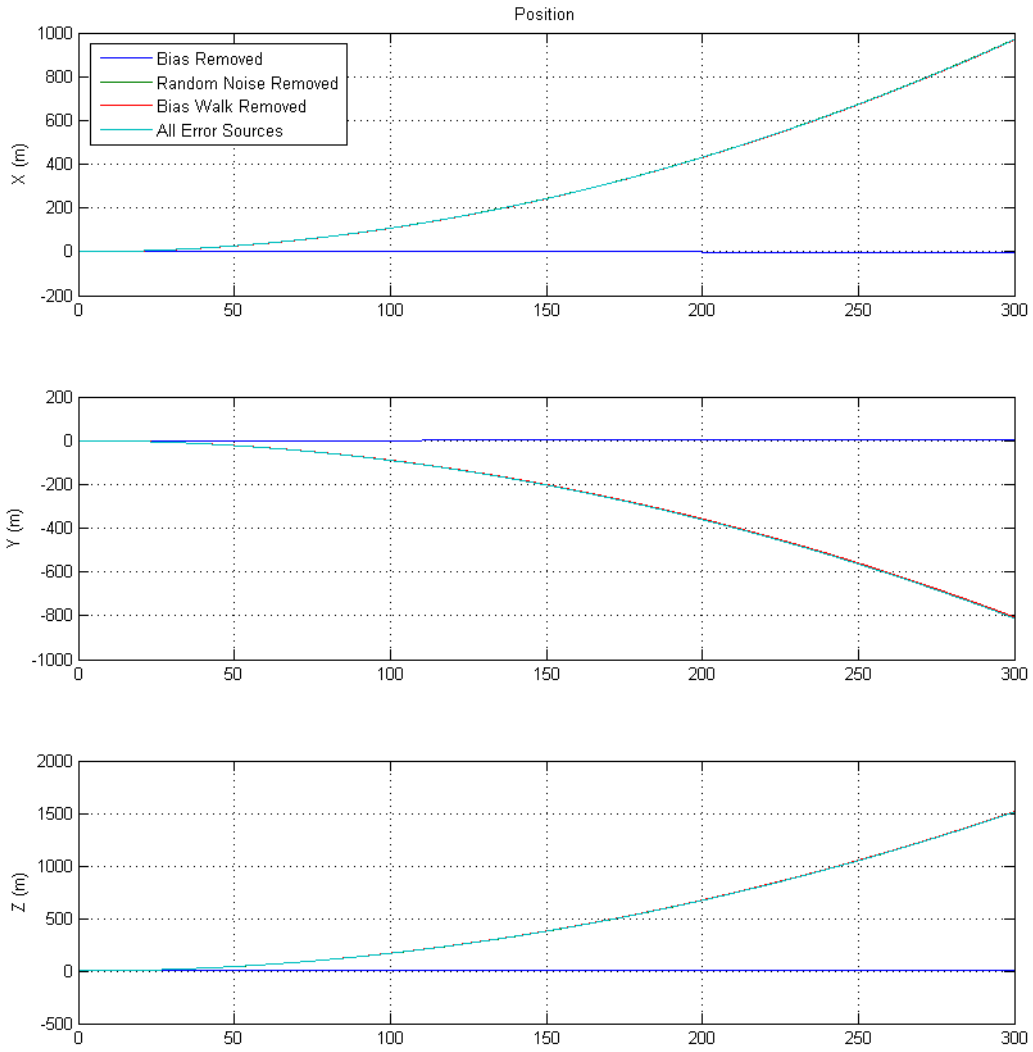

FIGURE 5.1: Comparison of virtual MIMUC dead reckoning position error with selected error sources removed

5.1.5 Thresholding

Apart from improving the accuracy of the calibration process, sensor errors and noise can be mitigated using other methods. In practical applications of dead-reckoning, bias errors and other noise sources are often dealt with by thresholding the accelerometer outputs [4]. Using this method, if the sensed accelerations do not exceed the threshold value, they are assumed to be zero and not integrated. Care must be taken in setting the threshold value. Too large of a threshold will result in the accelerometers being insensitive to true motion of the vehicle. Too low of a threshold will not sufficiently mitigate bias errors and sensor noise. The value

is often set to the 3σ bias error of the accelerometer. Unfortunately, the MPU-6000 does not come with an accurately calibrated zero bias so the 3σ bias error is unknown. Therefore, the 3σ bias error must be determined empirically from many MPU-6000 IMUs calibrated using the methods described in this thesis. Only sixteen sensors were available which is an unsuitable number to draw statistics from.

Although this thesis did not address the issue of creating navigation algorithms using the MIMUC output, it is worth examining how thresholding might improve the dead-reckoning performance. Since the 3σ bias error of the MPU-6000 accelerometers is currently unknown, several threshold values were chosen from Bishop [4]. The threshold values were 0.01 m/s^2, 0.05 m/s^2, 0.1 m/s^2 and no thresholding. The dead-reckoning scenario from the previous section was simulated with the different thresholds applied to the accelerometers. Fig. 5.2 shows a plot of the position errors along each axis for the various thresholds. Table 5.2 lists the position errors for each axis at each threshold level.

Thresholding Level	X-axis Error (m)	Y-axis Error (m)	Z-axis Error (m)
None	1251.3	-982.6	-801.5
0.01 m/s^2	1241.3	-944.0	-775.1
0.05 m/s^2	390.8	-54.1	-12.0
0.1 m/s^2	1.9	-0.1	0.02

TABLE 5.2: Impact of accelerometer threshold level on MIMUC dead reckoning position errors

The threshold level of 0.01 m/s^2 was too low to have a significant impact on the dead-reckoning errors, with less than 5% improvement seen for any axis. Increasing the threshold level to 0.05 m/s^2 saw significant improvements in the dead-reckoning error, although the X axis still showed a significant error of 390.8 m. This indicates that the bias errors of the X axis accelerometers are too large for the threshold value. Increasing the threshold to 0.1 m/s^2 reduced the position errors to acceptable levels. It has been shown that sufficient thresholding of the

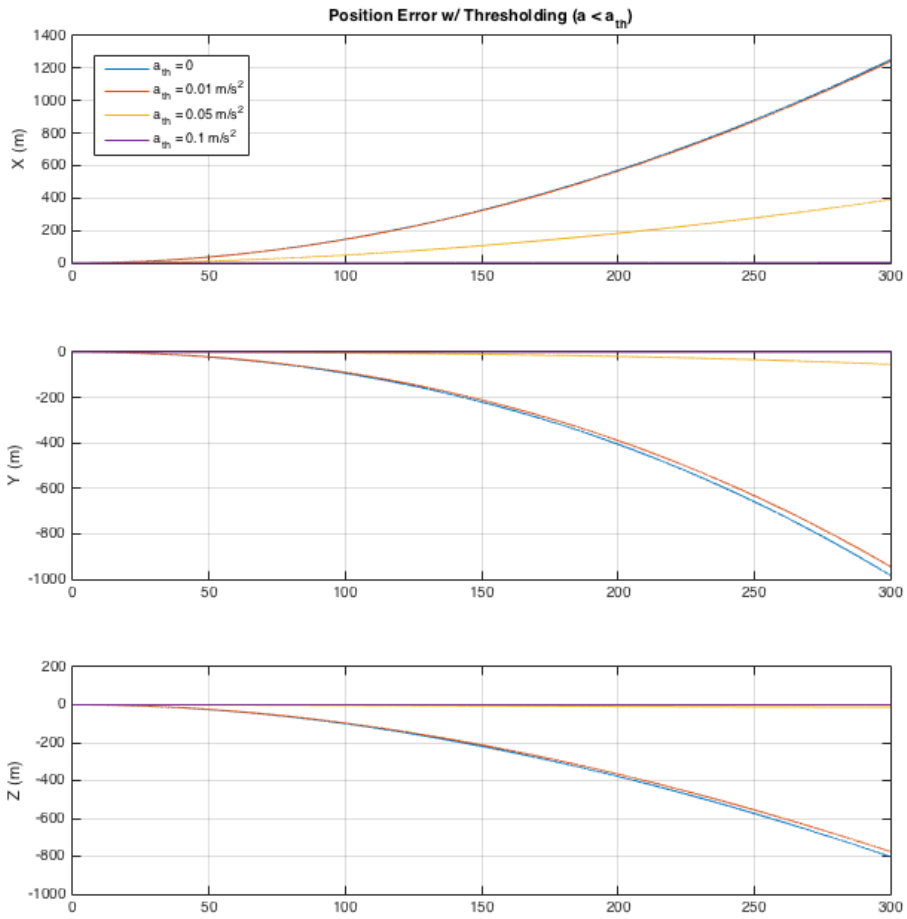

FIGURE 5.2: MIMUC dead reckoning position errors with accelerometer thresholding

accelerometer output significantly reduces dead-reckoning navigation errors. Determining the correct threshold level to mitigate sensor bias/noise while capturing true motion is a subject of future study.

5.2 Coning and Sculling Algorithms

5.2.1 Coning

The coning algorithm was successful in mitigating errors resulting from high frequency motion of the rotation vector between body updates. The algorithm allows the gyro outputs to be integrated at a high rate independently of computing the attitude direction cosine matrix/quaternion. The angular rate resulting from a

coning motion was generated in simulation for 10 s. The rate was then integrated using two methods: the coning algorithm at 1 kHz and trapezoidal integration 50 Hz. Trapezoidal integration was performed at a much lower speed since updating the DCM is computationally expensive.

Three separate scenarios were simulated to examine the effect of the size of the coning angle and frequency of the coning motion on the errors (Table 4.8). Beginning with Scenario 1 (4.7), a coning motion is simulated with an oscillation amplitude of 1 deg at 1 Hz. The attitude error for the coning algorithm is shown red while the attitude error resulting from trapezoidal integration is shown in blue. After 10 seconds, the norm of the attitude error when using the coning algorithm is 3.7878E-6 deg compared to 0.0014 deg for trapezoidal integration. Scenario 2 (Fig. 4.8) increases the amplitude of the coning motion to 10 deg. The norm of the attitude error is 3.7549E-4 deg using the coning algorithm and 0.1431 deg using trapezoidal integration. The attitude errors for Scenario 2 are 100 times larger when compared with Scenario 1 for both integration methods, suggesting that the norm of the attitude error scales with the square of the coning angle. Scenario 3 (Fig. 4.9) simulates a coning motion of 1 deg at 10 Hz. The coning algorithm resulted in an error of 0.0038 deg while trapezoid integration had an error of 1.336 deg. The attitude errors for Scenario 3 are 1000 times larger when compared with Scenario 1 for both integration methods, suggesting the norm of the attitude error appears to be proportional to the cube of the coning frequency.

In all three scenarios, the coning algorithm outperformed trapezoid integration. The attitude error using the coning algorithm was approximately 350 times less than trapezoidal integration for all three cases. This indicates that the improvement is independent of both the coning angle and the coning frequency. The performance improvement is dependent on the relative sampling frequency of the two integration methods. Performing the simulations with a lower sampling frequency for the coning algorithm resulted in larger attitude errors.

The simulations have also shown that the errors resulting from a coning motion are most sensitive to the frequency of the motion. This is an intuitive result as a higher coning rate results in the rotation vector is changing more rapidly between sample times. Fig. 4.7-4.9 illustrate the errors accumulate quickly over time.

5.2.2 Sculling

The sculling algorithm was successful in mitigating errors resulting from a combination of linear and angular oscillations. The algorithm allows the accelerometer outputs to be integrated at a high rate independently of updating the vehicle state. The angular rate and specific force resulting from a sculling motion was generated in simulation for 10 s. Both quantities were integrated using two methods: the coning/sculling (CS) algorithm at 1 kHz and trapezoidal integration at 50 Hz.

Three separate scenarios were simulated to examine the effect of the size of the coning angle and frequency of the coning motion on the errors (Table 4.8). Beginning with Scenario 1 (4.7), a coning motion is simulated with an oscillation amplitude of 1 deg at 1 Hz. The attitude error for the coning algorithm is shown red while the attitude error resulting from trapezoidal integration is shown in blue. After 10 seconds, the norm of the attitude error when using the coning algorithm is 3.7878E-6 deg compared to 0.0014 deg for trapezoidal integration. Scenario 2 (Fig. 4.8) increases the amplitude of the coning motion to 10 deg. The norm of the attitude error is 3.7549E-4 deg using the coning algorithm and 0.1431 deg using trapezoidal integration. The attitude errors for Scenario 2 are 100 times larger when compared with Scenario 1 for both integration methods, suggesting that the norm of the attitude error scales with the square of the coning angle. Scenario 3 (Fig. 4.9) simulates a coning motion of 1 deg at 10 Hz. The coning algorithm resulted in an error of 0.0038 deg while trapezoid integration had an

error of 1.336 deg. The attitude errors for Scenario 3 are 1000 times larger when compared with Scenario 1 for both integration methods, suggesting the norm of the attitude error appears to be proportional to the cube of the coning frequency.

Four separate scenarios were simulated to examine the effect of the angular oscillation amplitude, linear oscillation amplitude and oscillation frequency of the sculling motion on the errors (Table 4.11). Beginning with Scenario 1, a sculling motion resulted in 5.141E-5 m error when using the CS algorithms while trapezoidal integration yielded an error of 0.021 m. Scenario 2 increased the angular oscillation amplitude by a factor of 10 which had minimal impact on the error. Scenario 3 simulated a 10 g linear oscillation amplitude. This had the effect of increasing the position error by an order of magnitude. Scenario 4 increased the sculling frequency from 1 Hz to 10 Hz. This increased the error by greater than an order of magnitude relative to Scenario 1. The different scenarios indicate that the integration algorithms are most sensitive to an increase in the frequency of oscillation. This result is intuitive as the vehicle state is changing more rapidly between state updates. Since the CS algorithms are able to integrate the accelerometer and gyro outputs at a 50X higher rate than trapezoidal integration, it is better able to track the high frequency motion.

Titterton [39] states that a bias appears on the output of the X axis accelerometer due to inaccuracies of the resolution process. Therefore it is expected that this error will propagate as a linear error in velocity and a quadratic error in position. Fig. 4.10-4.13 confirm these expectations. Therefore, the CS algorithms prove to be extremely important as any quadratic error in position will quickly render the estimate of the position unusable. As an illustration, Scenario 4 was repeated for a 300 s simulation to illustrate how the errors would accumulate during a dead-reckoning simulation. As Fig. 5.3 shows, using trapezoidal integration results in a position error of 133m after 300 sec. Contrarily, the error using CS algorithms is only 0.25 m.

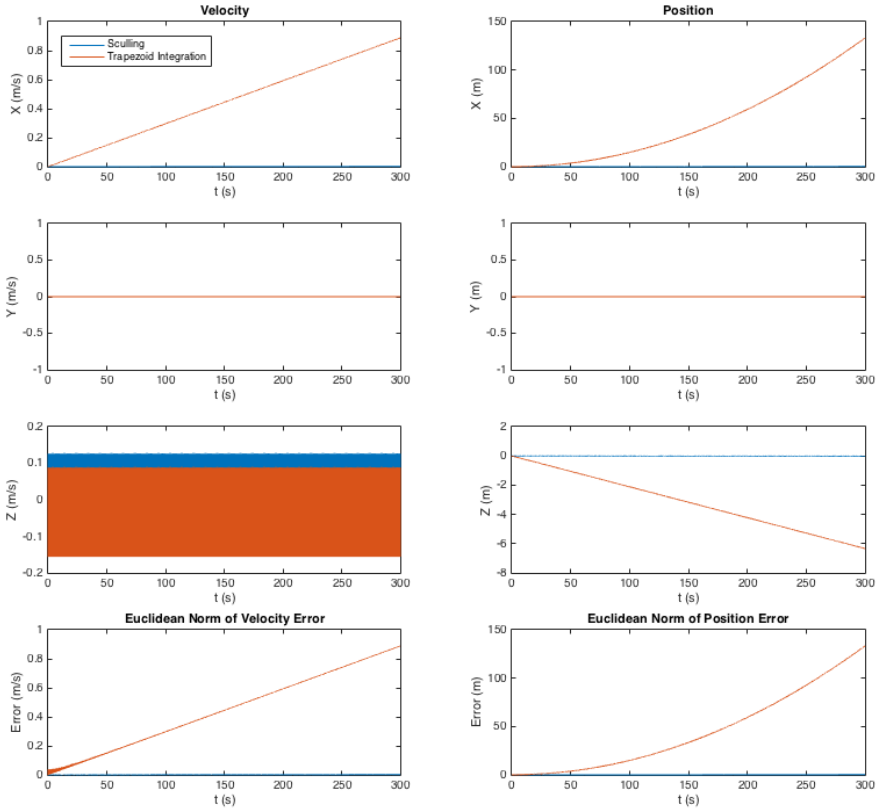

FIGURE 5.3: Velocity and position errors for sculling scenario 4 (300 seconds)

5.3 Comparison with Other IMUs

To further quantify the performance of the MIMUC, several tactical grade MEMS IMUs were chosen for comparison. The following IMUs were chosen for comparison: Analog Devices ADIS16488A, Moog Crossbow ANAV200A-341, Systron Donner SDI500-AB00, and the Sensonor STIM300. Four critical performance criteria were chosen for comparison: gyro bias stability, accelerometer bias stability, ARW and VRW. A summary of these parameters for each IMU can be found in Table 5.3. The MIMUC parameters were chosen by taking the worst values from each axis in Tables 4.3, 4.4, 4.5, 4.6. A visual comparison of these parameters is shown in Fig. 5.4 and 5.5. These figures allow a comparison of two important measures of IMU accuracy: bias stability and ARW/VRW. Points closer to the origin (lower-left of the plot) represent more accurate IMUs as both the bias

stability and random walk is minimal.

	Units	MIMUC	ADIS16488A	SDI500	STIM300	ANAV200A
Gyro Bias Stability	deg/hr	1.3	5.1	2	0.5	3
Angle Random Walk	deg/$\sqrt{\text{hr}}$	0.20	0.26	0.03	0.15	0.10
Accel. Bias Stability	mg	0.028	0.07	0.1	0.05	0.075
Velocity Random Walk	m/s/$\sqrt{\text{hr}}$	0.089	0.029	0.07	0.07	0.026
Power	W	1	0.8	5	2	1.25
Volume	cm^3	275	29	311	33	408
Mass	kg	0.1	0.048	0.59	0.055	0.409

TABLE 5.3: Comparison of MIMUC with commercial tactical grade IMUs [2, 23, 32, 36]

As shown in Fig. 5.4, the MIMUC gyroscope performs well compared with other commercial IMUs. The bias stability is 1.3 deg/hr which is outperformed only by the STIM300 and SDI500. The ARW is 0.20 deg/$\sqrt{\text{hr}}$, outperforming only the ADIS16488A. The SDI500 has an ARW of 0.02 deg/$\sqrt{\text{hr}}$ which is near navigation-grade. Fig. 5.5 compares the accuracy of the IMU accelerometers. The MIMUC outperforms all other IMUs with a bias stability of 0.028 mg. The next best bias stability is from the STIM300 at 0.05 mg. Contrarily, the MIMUC has the worst VRW. However, an examination of Table 4.6 shows that the Y-axis has a VRW equivalent to the SDI500 and STIM300.

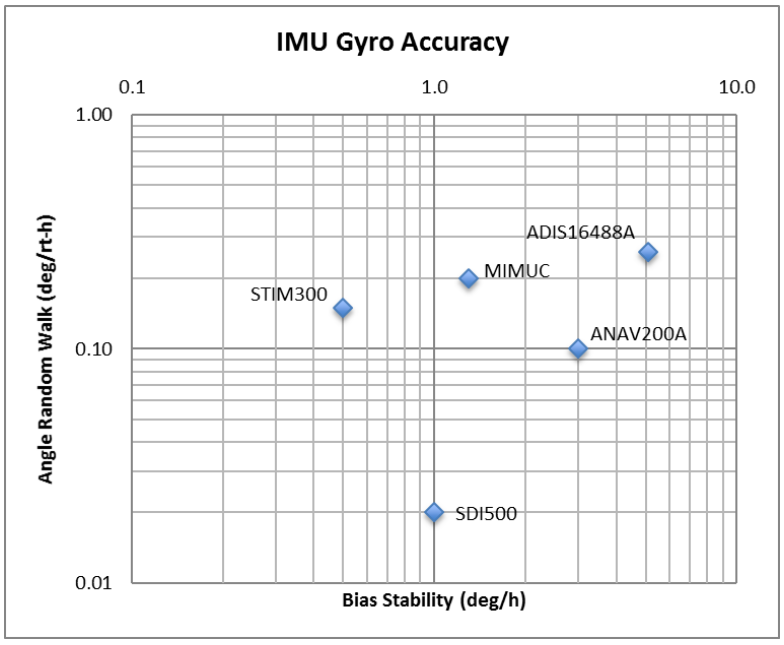

FIGURE 5.4: IMU gyro accuracy

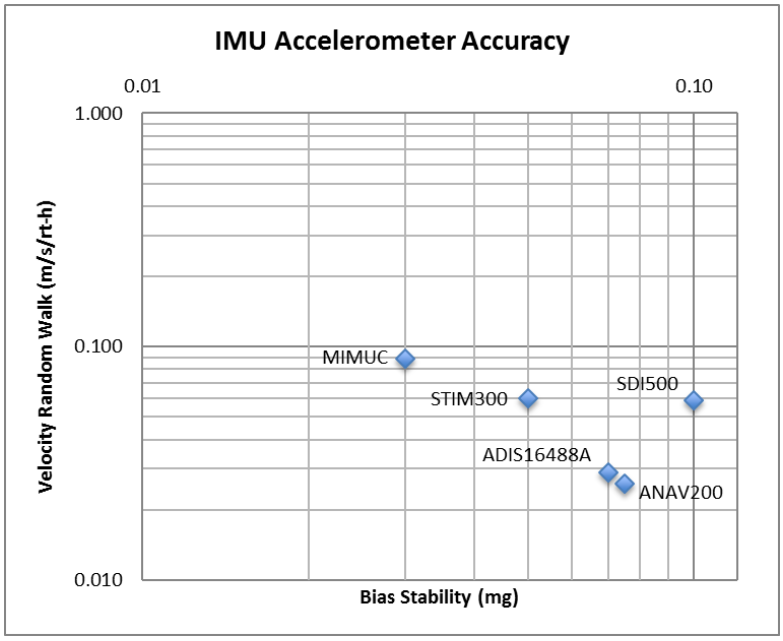

FIGURE 5.5: IMU accelerometer accuracy

Although the bias stability and ARW/VRW are important metrics for measuring performance, it is not the only factor when deciding on an IMU for a particular application. The objective of this thesis was to design an IMU for applications that have tight constraints on cost, power, mass and volume. Therefore, it is worthwhile to take these factors into account when comparing the MIMUC with other commercial IMUs. To illustrate this comparison, Fig. 5.6 and 5.7 show the bias stability and random walk of the gyros and accelerometers, respectively, scaled by the power consumption of each respective IMU. Similarly, Fig. 5.8 and 5.9 show the bias stability and random walk of the gyros and accelerometers, respectively, scaled by the mass of each respective IMU.

When looking at the overall gyro accuracy scaled by power, the overall accuracy of the MIMUC outperforms all other IMUs (the data point is closest to the origin). The bias stability is only outperformed by the STIM300 and the ARW is within a factor of two of the best performing IMU. The MIMUC has the 2nd lowest power consumption out of all the IMUs, with the ADIS16488A having the lowest power consumption. The MIMUC accelerometer accuracy also improves when accounting for power consumption. The bias stability remains the best out

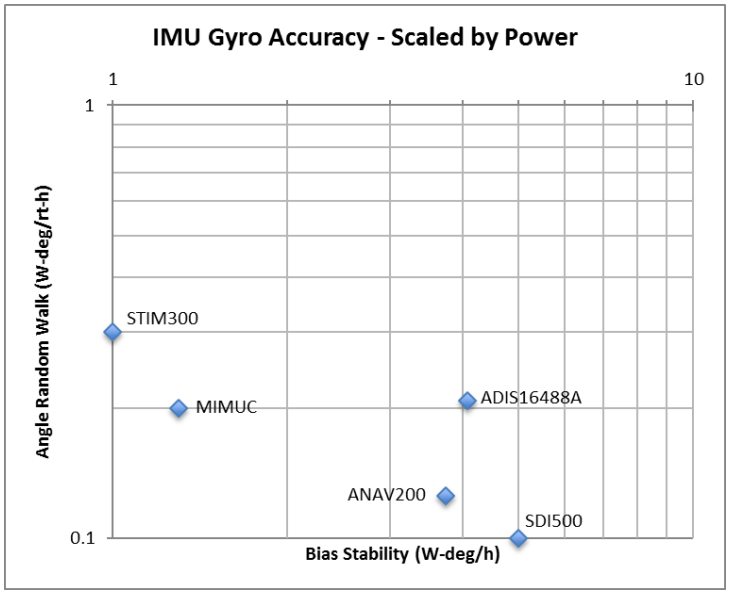

FIGURE 5.6: IMU gyro accuracy scaled by power

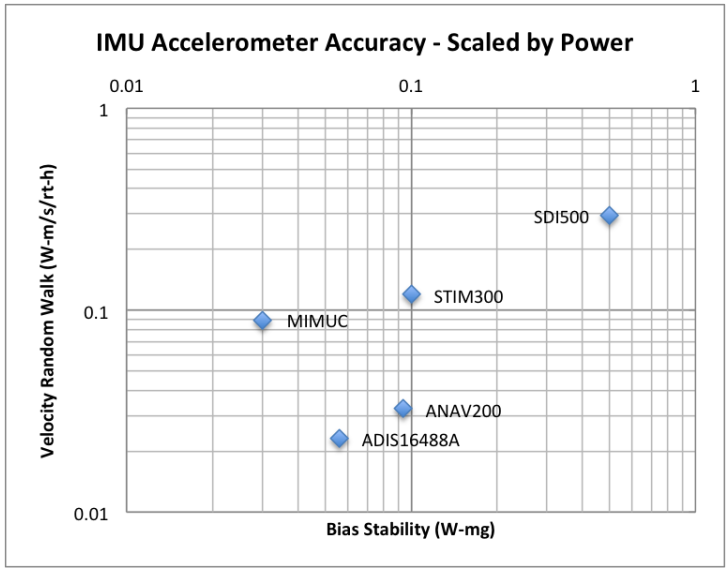

FIGURE 5.7: IMU accelerometer accuracy scaled by power

of the five IMUs while the VRW improves to the 3rd best.

Scaling the gyro and accelerometer accuracy by the mass has the effect of reducing the accuracy of the MIMUC compared with other IMUs. This can be attributed to the fact that the prototype MIMUC consists of two circuit boards. Consolidating the MIMUC to a single circuit board instead of two stacked boards would further improve the mass-scaled accuracy and especially the volume-scaled accuracy.

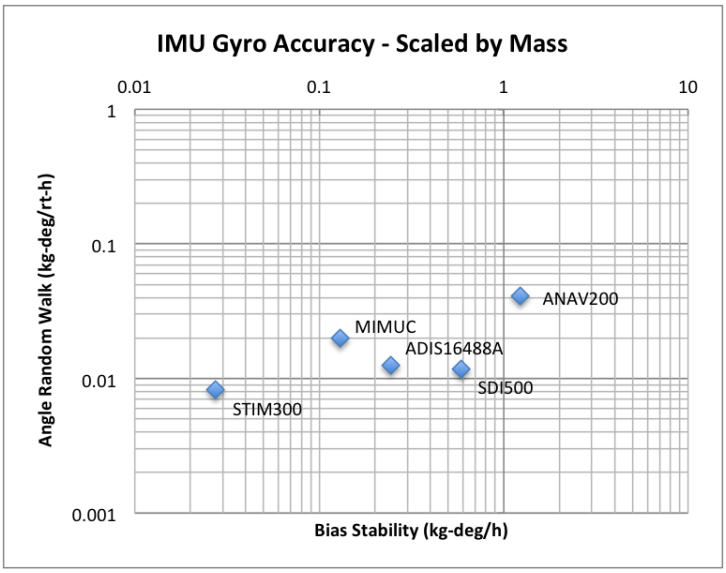

FIGURE 5.8: IMU gyro accuracy scaled by mass

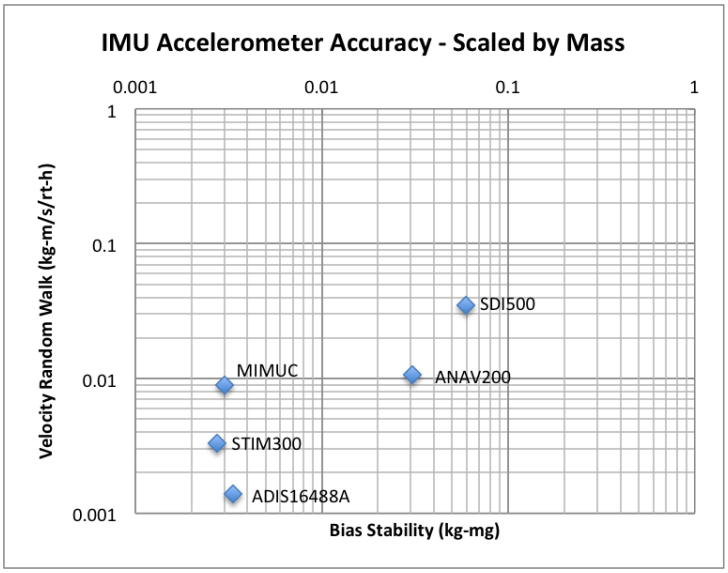

FIGURE 5.9: IMU accelerometer accuracy scaled by mass

In its current stage, the MIMUC has shown to be a competitive with other commercial tactical grade IMUs. Scaling the accuracy of the IMUs by power/-mass has shown that relative to other IMUs, the MIMUC has acceptable power consumption and mass. The power, mass and volume can be reduced in further iterations by combining the CDH and IMU circuit boards, eliminating the 16-bit processor altogether and reducing overall board size.

Chapter 6

Summary and Conclusion

6.1 Summary of Results

6.1.1 Conclusion

The main aim of this thesis was to develop a prototype MEMS IMU cluster for precision navigation. In Chapter 2, it was theorized that a MEMS IMU consisting of N individual MEMS IMUs would decrease the output noise compared to a single MEMS IMU by $\sqrt{N}$. The results of the experiments proposed in Chapter 3 set to test this hypothesis. Chapter 4 presented results that proved that averaging does result in a $\sqrt{N}$ decrease in the output noise. The gyro bias stability, accelerometer bias stability, ARW, and VRW of the MIMUC were approximately reduced by a factor of 4 relative to a single MPU-6000 IMU. The only exception to this result was that the bias stability of the Z axis only saw a decrease by a factor of 2 through averaging. This suggests that if the axis of the accelerometers are under load, averaging the sensors outputs has limited benefit.

The dead-reckoning performance of the MIMUC was much better than a single MEMS IMU, but is still unsuitable for precision navigation applications. After five minutes of dead-reckoning, position errors exceed 1 km. These errors were a result of inaccurate calibration of the gyro and accelerometer bias errors. A more accurate calibration process must be developed to minimize these errors. An alternative solution was proposed by thresholding the outputs of the accelerometers. With a threshold level of 0.1 m/s^2, the dead-reckoning position errors did not exceed 2 m for any axis, with two axes not exceed 10 cm.

The secondary aim of this thesis was to develop a set of coning and sculling algorithms to more accurately track high frequency motion of a vehicle. A set of

equations was derived, simulated and compared with standard trapezoidal integration. The results of the simulations showed that the coning and sculling algorithms substantially reduce errors due to coning and sculling motions relative to trapezoidal integration. For a coning motion, the proposed coning algorithm exhibited attitude errors approximately 350 times less than trapezoidal integration at the body rate for all of the scenarios tested. For a sculling motion, the proposed sculling algorithm exhibited position errors approximately 400 times less than trapezoidal integration at the body rate over all the scenarios tested. Therefore, these algorithms have been proven to provide a substantial improvement in accuracy over trapezoidal integration by allowing the angular rates and specific forces to be integrated at a high rate without burdening the computational hardware.

6.1.2 Future Work

Several topics of research could be pursued in furthering developing this technology. The first would be to incorporate more sensors in the IMU cluster to further enhance the performance characteristics. Adding more sensors simply requires placing more sensors in the schematic. Each sensor is then connected to the existing SPI bus and connected to the microcontroller. At some point, adding more sensors will result in diminishing returns since the performance scales with the square root of the number of sensors. The number of sensors is also limited by the maximum bandwidth of the SPI bus; with too many sensors on the bus, the desired data rates may not be attainable with a single microcontroller.

The design would also benefit by replacing the 16-bit processor with solely 32-bit processor. This would allow both the IMU and CDH functionality to be incorporate into a single circuit board. Overall mass, volume and power consumption of the system would be greatly reduced, as well as improving device performance by eliminating the I2C bottleneck between the IMU and CDH circuit

boards. This would make the MIMUC a more competitive solution compared to other products on the market.

Beyond redesigning the board, the performance of the MIMUC would greatly benefit from a more accurate calibration apparatus. Having a computer controlled rate table would allow more precise control of the rotation rate. This rate would show up as a bias error in calibrated gyro output. The calibration would also benefit from an auto-leveling table. The Trio-Tech rate table had a simple bubble level which did not allow a very accurate indication of how level the rate table was. If the accelerometer axis under test is not perfectly aligned with the gravity vector, not only is it not measuring the true gravitational acceleration (via the normal force), the other axes are measuring non-zero accelerations. Therefore it is paramount that the accelerometers be calibrated on a flat, level surface.

Appendix A

Coning and Sculling Algorithms

The pseudocode for the combined coning and sculling algorithm is given below. The corrections are applied over the k-th body update interval ($t_k - t_{k-1} = T_k$). It is assumed that over each body update interval there is set of N gyro outputs and N accelerometer outputs available ($T_m = T_k/N$). The outer loop handles the update of the attitude quaternion and velocity vector while the inner loop executes the high frequency computation of the change in the rotation vector and velocity vector. (Note: care must be taken to avoid division-by-zero errors in the quaternion update.)

```
for k = 1, 2, ...
  θ = 0,  β = 0, v = 0
  Δv_scul = 0, Δθ_prev = 0, Δv_prev = 0
  for m = 1 to N
    i = (k-1) + m
    % Integrate gyro and accelerometer outputs (trapezoidal integration)
    Δθ = (T_m/2)*(ω_i + ω_{i-1})
    Δv = (T_m/2)*(a_i + a_{i-1})
    % Compute coning and sculling compensation
    Δβ = (1/2)*(θ + (1/6)*Δθ_prev) × Δθ
    δv_scul = (1/2)*((θ + (1/6)*Δθ_prev) × Δv + (v_prev + (1/6)*Δv) × Δθ)
    % Update attitude and velocity
    θ += Δθ
    β += Δβ
    v += Δv
    Δv_scul += δv_scul
    % Store current values for use in next iteration
    Δθ_prev = Δθ
```

```
    Δv_prev = Δv
  end
  % Compute total change in orientation vector over k-th update cycle
  Δϕ = θ + β
  % Update attitude quaternion
  q_i = q_{i-1} × [cos(ϕ/2)  sin(ϕ/2)Δϕ/ϕ]^T
  % Compute rotation matrix from quaternion
  T = quat2dcm(q_i)
  % Component of velocity due to rotation
  v_rot = (1/2)*(θ × v);
  % Update IMU velocity
  v_{imu,k} = v_{imu,k-1} + T * (v + v_rot + v_scul);
end
```

```
function T = quat2dcm(q) % Convert quaternion to direction cosine matrix
  q0 = q(1); q1 = q(2); q2 = q(3); q3 = q(4);
  T = zeros(3,3);
  T(1,1) = q0^2 + q1^2 - q2^2 - q3^2;
  T(1,2) = 2*(q1*q2 - q0*q3);
  T(1,3) = 2*(q1*q3 + q0*q2);
  T(2,1) = 2*(q1*q2 + q0*q3);
  T(2,2) = q0^2 - q1^2 + q2^2 - q3^2;
  T(2,3) = 2*(q2*q3 - q0*q1);
  T(3,1) = 2*(q1*q3 - q0*q2);
  T(3,2) = 2*(q2*q3 + q0*q1);
  T(3,3) = q0^2 - q1^2 - q2^2 + q3^2;
end
```

Appendix B

MIMUC Calibration Coefficients

The following two tables list the calibration coefficients for each individual gyroscope and accelerometer on the MIMUC. The coefficients were determined using methods outlined in Section 2.4.

TABLE B.1: Accelerometer calibration coefficients

Sensor #	$b_{a,x}$	$b_{a,y}$	$b_{a,z}$	$S_{a,x}$	$S_{a,y}$	$S_{a,z}$	$M_{a,xy}$	$M_{a,xz}$	$M_{a,yz}$	$T_{a,x}$	$T_{a,y}$	$T_{a,z}$
1	0.0110	0.0069	0.0121	0.0018	0.0032	0.0119	0.0039	0.0079	0.0037	0.0004	0.0001	-0.0021
2	0.0152	0.0003	0.0057	0.0090	-0.0051	0.0024	0.0106	0.0019	0.0057	0.0003	0.0001	-0.0011
3	0.0121	0.0020	-0.0240	0.0065	-0.0044	0.0046	0.0067	-0.0026	-0.0029	0.0003	-0.0002	-0.0021
4	0.0050	-0.0047	0.0065	0.0062	-0.0023	0.0093	0.0044	0.0079	0.0024	0.0003	-0.0000	-0.0023
5	0.0123	0.0045	0.0046	-0.0006	0.0029	-0.0003	0.0098	0.0002	0.0057	0.0004	0.0000	-0.0020
6	0.0100	0.0072	-0.0210	-0.0017	-0.0041	-0.0026	0.0064	0.0021	0.0030	0.0002	0.0000	-0.0018
7	0.0153	-0.0000	-0.0127	-0.0035	-0.0026	0.0025	0.0030	0.0048	0.0010	0.0002	0.0000	-0.0018
8	0.0028	0.0051	-0.0132	0.0031	0.0047	0.0077	0.0043	-0.0011	0.0035	0.0002	-0.0001	-0.0029
9	0.0074	-0.0058	0.0045	-0.0031	0.0084	0.0027	0.0012	-0.0058	-0.0001	0.0001	0.0000	-0.0020
10	0.0114	0.0060	0.0170	-0.0035	0.0056	0.0019	0.0085	-0.0040	0.0013	0.0002	0.0001	-0.0016
11	-0.0028	0.0027	-0.0158	-0.0004	0.0033	0.0004	0.0037	-0.0003	0.0014	0.0002	0.0001	-0.0019
12	0.0106	0.0025	0.0212	0.0035	0.0066	0.0057	0.0108	-0.0039	-0.0004	0.0003	-0.0001	-0.0018
13	0.0131	0.0125	0.0151	0.0051	0.0033	0.0138	0.0204	-0.0047	-0.0024	0.0003	0.0001	-0.0019
14	-0.0007	0.0007	-0.0241	-0.0023	0.0034	0.0020	0.0103	0.0016	-0.0019	0.0004	0.0001	-0.0019
15	0.0063	-0.0020	0.0139	-0.0005	0.0068	0.0083	0.0118	-0.0040	0.0049	0.0002	-0.0000	-0.0016
16	0.0127	0.0042	-0.0048	0.0020	-0.0055	-0.0003	0.0009	0.0029	0.0112	0.0003	0.0000	-0.0015

TABLE B.2: Gyroscope calibration coefficients

Sensor #	$b_{a,x}$	$b_{a,y}$	$b_{a,z}$	$S_{a,x}$	$S_{a,y}$	$S_{a,z}$	$M_{a,xy}$	$M_{a,xz}$	$M_{a,yz}$
1	-2.3182	1.4541	-0.8227	0.0002	0.0013	0.0022	0.0007	-0.0129	0.0079
2	-2.2711	-0.0195	-0.0435	0.0031	-0.0032	-0.0055	0.0088	-0.0009	-0.0051
3	-1.9024	-0.6230	-0.5995	-0.0000	-0.0017	0.0062	0.0040	-0.0059	0.0121
4	-0.7252	0.9209	-0.5083	0.0023	-0.0003	-0.0012	-0.0000	-0.0125	0.0009
5	-0.7186	0.6271	0.7876	0.0080	0.0046	0.0022	0.0087	-0.0057	0.0006
6	-1.5609	1.6386	0.0783	0.0069	-0.0048	0.0069	0.0039	-0.0029	-0.0041
7	-1.1919	-0.1875	0.5903	0.0099	0.0057	0.0088	-0.0016	-0.0005	0.0057
8	-0.5615	1.2005	0.5977	0.0081	0.0082	0.0018	0.0017	-0.0029	0.0081
9	-0.6081	0.6966	-0.1049	-0.0005	-0.0028	0.0075	-0.0022	0.0018	0.0109
10	-0.5511	-0.1847	0.2416	0.0019	0.0042	0.0075	0.0073	0.0045	0.0021
11	0.6897	1.1271	0.2485	-0.0029	0.0042	-0.0025	0.0003	-0.0119	0.0006
12	-3.1843	1.3909	0.3855	0.0057	0.0098	-0.0022	0.0083	-0.0179	-0.0016
13	-1.6669	1.0371	0.4497	0.0091	0.0039	0.0006	0.0206	0.0031	-0.0000
14	-0.3208	1.2372	0.0550	0.0118	-0.0006	-0.0023	0.0074	-0.0152	0.0066
15	-1.6241	-0.0422	0.6120	-0.0024	0.0021	-0.0002	0.0074	-0.0077	0.0173
16	-1.3522	0.0874	-0.1093	0.0110	-0.0032	0.0091	-0.0019	-0.0081	-0.0030

TABLE B.3: Gyroscope calibration coefficients (cont.)

Sensor #	G_{xx}	G_{xy}	G_{xz}	G_{yx}	G_{yy}	G_{yz}	G_{zx}	G_{zy}	G_{zz}	$T_{a,x}$	$T_{a,y}$	$T_{a,z}$
1	0.0032	0.0055	0.0238	-0.0022	-0.0027	-0.0243	0.0037	-0.0100	0.0833	-0.0085	0.0087	-0.0002
2	0.0027	0.0049	0.1336	-0.0024	0.0067	0.0118	0.0010	-0.0031	-0.0029	-0.0208	-0.0271	-0.0088
3	0.0069	0.0000	0.0327	0.0011	0.0055	0.0111	0.0020	-0.0083	0.0200	0.0013	-0.0185	0.0086
4	0.0020	0.0055	0.0826	-0.0027	-0.0019	-0.0664	0.0023	-0.0085	-0.0368	-0.0110	0.0097	-0.0005
5	0.0047	0.0019	0.1209	-0.0014	-0.0007	0.0018	0.0012	0.0046	-0.1253	-0.0118	-0.0060	-0.0095
6	0.0067	0.0059	0.0950	-0.0043	-0.0018	0.0111	0.0012	-0.0023	-0.2576	-0.0164	-0.0019	0.0052
7	0.0052	0.0017	0.0649	-0.0015	0.0044	0.0139	0.0047	0.0023	-0.2555	-0.0076	0.0195	-0.0020
8	0.0021	0.0040	0.0823	-0.0036	0.0059	-0.0071	0.0062	0.0024	-0.1854	-0.0268	-0.0082	-0.0088
9	0.0019	0.0003	0.0020	-0.0037	0.0099	0.0117	0.0011	0.0015	-0.0944	-0.0001	-0.0288	0.0152
10	0.0023	-0.0019	0.0139	0.0011	0.0036	0.0013	-0.0012	0.0008	-0.0011	-0.0058	-0.0024	0.0006
11	0.0037	-0.0018	0.0412	-0.0013	0.0029	-0.0204	-0.0043	0.0044	0.0785	0.0104	0.0154	0.0190
12	0.0042	0.0010	0.0444	-0.0013	0.0034	0.0076	-0.0045	0.0064	0.0049	-0.0279	-0.0173	0.0152
13	0.0051	-0.0001	0.0428	0.0030	-0.0005	0.0179	-0.0075	0.0066	-0.0487	-0.0241	-0.0095	0.0172
14	0.0038	-0.0001	0.0399	-0.0021	0.0017	-0.0047	-0.0091	0.0101	-0.0754	-0.0016	-0.0041	0.0030
15	0.0005	-0.0001	0.0052	-0.0013	0.0029	0.0002	-0.0082	0.0022	-0.0567	-0.0006	-0.0133	0.0111
16	0.0019	-0.0014	0.0274	-0.0005	0.0014	-0.0054	-0.0108	-0.0049	-0.0194	-0.0263	-0.0076	-0.0086

Appendix C

Allan Deviation

The Allan deviation plot is a method of graphing the various error sources of a time-series of data on a single plot. The method was first introduced by David Allan in 1966 to measure the frequency stability of clocks and oscillators. The technique is useful for inertial navigation systems since it allows both the angle/velocity random walk and bias stability of the sensors to be determined in a single plot.

To compute the Allan deviation for a time series of data x_i, begin by splitting the data series into bins of size n where N is the number of resulting bins. Let y_i be the average of bin i where $i = 1, \ldots, N$. The Allan variance of x_i is given by

$$\sigma^2(nT) = \frac{1}{2(N-1)} \sum_{1}^{N-1} (y_{i+1} - y_i)^2 \tag{C.1}$$

where T is the time between consecutive samples in x_i. The Allan deviation then is found by taking the square root of the Allan variance. For interpretation of the Allan deviation plot, please refer to [13, 25, 26].

Bibliography

[1] Hector D.E. Alvarez. Geometrical Configuration Comparison of Redundant Inertial Measurement Units. In *AAS 11-258, AAS Space Flight Mechanics Meeting*, 2010.

[2] Analog Devices. ADIS16488A datasheet. URL `http://www.analog.com/media/en/technical-documentation/data-sheets/ADIS16488A.pdf`.

[3] Jared B. Bancroft. Multiple IMU Integration for Vehicular Navigation. *Proceedings of ION GNSS 2009*, 2009.

[4] Robert H. Bishop, Dilmurat Azimov, and Timothy Crain. Sensitivity of Mars Entry Navigation Errors to Sensed Acceleration Threshold. In *2003 AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2003.

[5] John E. Bortz. A New Mathematical Foundation for Strapdown Inertial Navigation. *Aerospace and Electronic Systems, IEEE Transactions on*, 1: 61–66, 1970.

[6] Honglong Chang, Liang Xue, Wei Qin, Guangmin Yuan, and Weizheng Yuan. An Integrated MEMS Gyroscope Array with Higher Accuracy Output. *Sensors*, 8(4):2886–2899, 2008.

[7] Averil B.C. Chatfield. *Fundamentals of High Accuracy Inertial Navigation*. Progress in astronautics and aeronautics. American Institute of Aeronautics & Astronautics, 1997. ISBN 9781600864278.

[8] Stuart Cheshire and Mary Baker. *Consistent Overhead Byte Stuffing*, volume 27. ACM, 1997.

[9] Seong Yun Cho and Chan Gook Park. A Calibration Technique for a Redundant IMU Containing Low-Grade Inertial Sensors. *ETRI journal*, 27(4): 418–426, 2005.

[10] I. Colomina, M. Giménez, J.J. Rosales, M. Wis, A. Gomez, and P. Miguelsanz. Redundant IMUs for Precise Trajectory Determination. In *Proceedings of the 20th ISPRS Congress, Istanbul, Turkey*, pages 12–23, 2004.

[11] PC/104 Embedded Consortium. PC/104 Specification Version 2.5. *San Francisco: PC/104 Embedded Consortium*, 2003.

[12] Timothy P. Crain, Robert H. Bishop, and Tye Brady. Shifting the Inertial Navigation Paradigm with MEMS Technology. *Advances in the Astronautical Sciences*, 137(25):2010, 2010.

[13] Naser El-Sheimy, Haiying Hou, and Xiaoji Niu. Analysis and Modeling of Inertial Sensors using Allan Variance. *Instrumentation and Measurement, IEEE Transactions on*, 57(1):140–149, 2008.

[14] Warren S. Flenniken, John H. Wall, and David M. Bevly. Characterization of Various IMU Error Sources and the Effect on Navigation Performance. In *ION GNSS*, pages 967–978, 2005.

[15] D.J. Flynn. A Discussion of Coning Errors Exhibited by Inertial Navigation Systems. Technical report, DTIC Document, 1984.

[16] Demoz Gebre-Egziabher. *Design and Performance Analysis of a Low-Cost Aided Dead Reckoning Navigator*. PhD thesis, Stanford University, 2004.

[17] Daniel R. Greenheck, Robert H. Bishop, Eric M. Jonardi, and John A. Christian. Design and Testing of a Low-Cost MEMS IMU Cluster for SmallSat Applications. In *Proceedings of the 28th Annual AIAA/USU Conference on Small Satellites*, 2014. URL `http://digitalcommons.usu.edu/smallsat/2014/AdvTechI/5/`.

[18] Stéphane Guerrier. Improving Accuracy with Multiple Sensors: Study of Redundant MEMS-IMU GPS Configurations. In *Proceedings of the 22nd International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS 2009)*, pages 3114–3121, 2009.

[19] John Hanson, James Chartres, Hugo Sanchez, and Ken Oyadomari. The EDSN Intersatellite Communications Network. In *Proceedings of the 28th Annual AIAA/USU Conference on Small Satellites*, 2014. URL `http://digitalcommons.usu.edu/smallsat/2014/Workshop/1/`.

[20] Invensense. MPU-6000 and MPU-6050 Product Specification Revision 3.4. URL `http://www.invensense.com/mems/gyro/documents/PS-MPU-6000A-00v3.4.pdf`.

[21] Jack B. Kuipers. *Quaternions and Rotation Sequences*, volume 66. Princeton university press Princeton, 1999.

[22] H.F.S. Martin, P.D. Groves, M. Newman, and R. Faragher. A New Approach to Better Low-Cost MEMS IMU Performance Using Sensor Arrays. 2013.

[23] Moog Crossbow. ANAV200A-301 datasheet. URL `http://www.moog-crossbow.com/Literature/Data_Sheets/ANAV200_Data_Sheet.pdf`.

[24] John Olof Nilsson, Isaac Skog, and Peter Händel. Aligning the Forces Eliminating the Misalignments in IMU Arrays. 2014.

[25] Institute of Electrical and Electronics Engineers. IEEE Standard Specification Format Guide and Test Procedure for Coriolis Vibratory Gyros. *IEEE Std 1431-2004*, pages 1–78, Dec 2004. doi: 10.1109/IEEESTD.2004.95744.

[26] Institute of Electrical and Electronics Engineers. IEEE Standard Specification Format Guide and Test Procedure for Linear, Single-Axis, Non-Gyroscopic Accelerometers. *IEEE Std 1293-1998 (R2008)*, pages 1–249, July 2011. doi: 10.1109/IEEESTD.2011.5960745.

[27] Abdallah Osman, Bruce Wright, Aboelmagd Noureldin, and Naser El-Sheimy. Multi-sensor Inertial Navigation Systems Employing Skewed Redundant Inertial Sensors. In *Proceedings of the 19th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS 2006)*, pages 2202–2207, 2001.

[28] Athanasios Papoulis and S. Unnikrishna Pillai. *Probability Random Variables and Stochastic Processes*. Tata McGraw-Hill Education, 2002.

[29] R. Ramalingam, G. Anitha, and J. Shanmugam. Microelectromechnical systems inertial measurement unit error modelling and error analysis for low-cost strapdown inertial navigation system. *Defence Science Journal*, 59(6):650–658, 2009.

[30] Paul G. Savage. Strapdown Inertial Navigation Integration Algorithm Design Part 1: Attitude Algorithms. *Journal of Guidance, Control, and Dynamics*, 21(1):19–28, 1998.

[31] Paul G. Savage. Strapdown Inertial Navigation Integration Algorithm Design Part 2: Velocity and Position Algorithms. *Journal of Guidance, Control, and Dynamics*, 21(2):208–221, 1998.

[32] Sensonor. STIM300 datasheet. URL `http://www.sensonor.com/media/91313/ts1524.r8%20datasheet%20stim300.pdf`.

[33] Isaac Skog, John-Olof Nilsson, and Peter Händel. An Open Source Multi Inertial Measurement Unit MIMU Platform. In *Inertial Sensors and Systems (ISISS), 2014 International Symposium on*, pages 1–4. IEEE, 2014.

[34] Isaac Skog, John-Olof Nilsson, and Peter Händel. Pedestrian Tracking Using an IMU Array. *Proc. IEEE CONECCT*, 2014.

[35] Clyde Space. 1U CubeSat Chassis Walls, 2015. URL `http://www.clyde-space.com/cubesat_shop/structures/1u_structures/115_1u-chassis-walls`.

[36] Systron Donner. SDI500 datasheet. URL `http://www.systron.com/sites/default/files/sdi500_pr_073009.pdf`.

[37] Texas Instruments. MSP430FR5969 datasheet, . URL `http://www.ti.com/lit/ds/symlink/msp430fr5969.pdf`.

[38] Texas Instruments. TPS73133 LDO Voltage Regulator datasheet, . URL http://www.ti.com/lit/ds/symlink/tps73133-ep.pdf.

[39] David Titterton and John Weston. *Strapdown Inertial Navigation Technology, 2nd Edition*. Institution of Engineering and Technology, 2004. ISBN 9780863413582.

[40] Adrian Waegli, Jan Skaloud, Stéphane Guerrier, Maria Eulàlia Parés, and Ismael Colomina. Noise Reduction and Estimation in Multiple Micro-Electromechanical Inertial systems. *Measurement Science and Technology*, 21(6):065201, 2010.

[41] Oliver J. Woodman. An Introduction to Inertial Navigation. *University of Cambridge, Computer Laboratory, Tech. Rep. UCAMCL-TR-696*, 14:15, 2007.