INVESTIGATING MEMS ACCELEROMETER CALIBRATION TECHNIQUES

Senior Honors Capstone Project
submitted in partial fulfillment of requirements
for the University Scholars Program

by
William Ward

University of Central Arkansas

Conway, Arkansas

Spring 2023

Capstone Project Committee

Advisor: _______________________
Dr. William Slaton
Professor and Engineering Physics Coordinator
Department of Physics & Astronomy

Reader: _______________________
Dr. Lin Zhang
Assistant Professor
Department of Physics & Astronomy

Dean: _______________________
Dr. Patricia Smith
Dean of the Honors College

Abstract

Triaxial microelectromechanical systems (MEMS) accelerometers are inertial sensors that measure linear acceleration along three orthogonal axes. In many robotics applications, accelerometers help track a robot's relative displacement. However, displacement estimates derived from an accelerometer's raw measurements drift significantly over time from ground truth values during the derivation process due to accumulated errors in acceleration measurements. While accelerometers can be combined with other sensors to reduce the effect of displacement estimation drifting, calibrating accelerometers themselves before use is still essential, especially for very low-cost applications. This study investigates methods of calibrating a low-cost MEMS accelerometer using least-squares fitting, primarily concentrated on removing systematic error from acceleration signals. All code written for this project is publicly available on GitHub at https://github.com/willward20/UCA-Honors-Capstone.

Table of Contents

Chapter 1: Introduction …………………………………………………………………. 4

Chapter 2: Literature Review ………………………………………………………… 8

Introduction ……………………………………………………………………... 8

Calibrating Systematic Error ……………………………………………………. 9

Zero-State Detection ………………………………………………….………. 9

Accelerometer Error Modeling ……………………………………………. 11

Velocity and Displacement Error Models ………………………………. 13

Least-Squares Optimization of Parameters ……………………………... 14

Data Collection Procedures ……………………………………………….. 15

Chapter 3: Methodology ……………………………………………………………… 17

Hardware and Software Setup …………………………………………………... 17

Preliminary Characterization Tests ……………………………………………... 19

Statistical Analysis ……………………………………………………………..... 20

Experimental Data Collection …………………………………………………... 21

Optimizing Accelerometer Model Parameters ……………………………….. 22

Final Calibration Trial …………………………………………………………... 24

Chapter 4: Results …………………………………………………………………….. 25

Preliminary Characterization Tests ……………………………………………... 25

Six-Position Calibration ………………………………………………………… 29

Final Calibration Trial …………………………………………………………... 31

Chapter 5: Conclusions ……………………………………………………………….. 34

Sources of Error ………………………………………………………………. 35

Future Work …..……………………………………………………………………. 37

Appendix A …………………………………………………………………………... 38

References ……………………………………………………………………………. 42

Chapter 1: Introduction

An accelerometer is an inertial sensor that measures linear acceleration along three orthogonal axes in the 3-D Cartesian space. For example, Figure 1 shows an accelerometer as it is pushed across a flat surface by a constant force *F*. Along the x-axis, the accelerometer measures a linear acceleration of a_x, due to the force *F*, and along the z-axis, the accelerometer measures the acceleration due to Earth's gravity (approximately 9.797 m/s^2 in Conway, AR [1]). An ideal accelerometer, whose data is completely free of error, could be used to precisely track an object's displacement, $s(t)$, from an initial starting position, s_0, by performing double integration on the measured linear acceleration, $a(t)$ [2]:

$$s(t) = \int_{t_0}^{t} \left(\int_{t_0}^{t} a(t)dt \right) dt + v_0(t - t_0) + s_0 \tag{1}$$

In (1), t_0 is the initial time, and v_0 is the initial velocity.

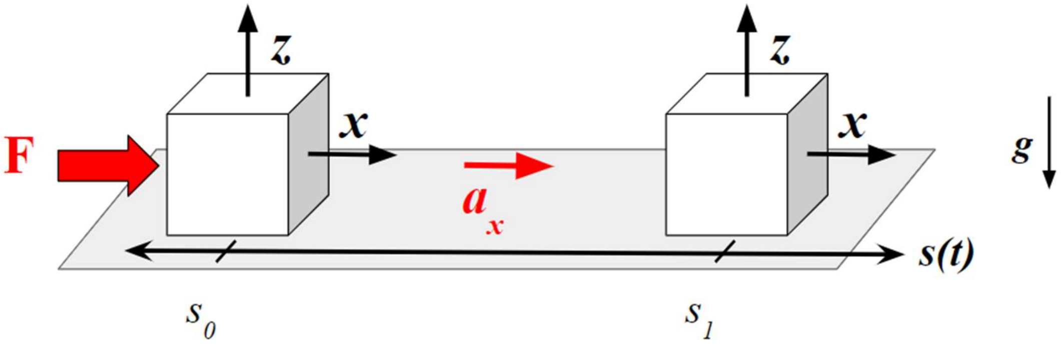


Figure 1: Accelerometer moving across a flat table due to a force F in the x direction.

Unfortunately, even the most accurate accelerometers, used for navigating autonomous vehicles or tracking tactical missiles, contain small errors in acceleration measurements that accumulate into a massive drift in displacement calculations over time [3]. For example, consider an accelerometer that is sitting at rest on a table. Since the sensor is not moving, it should measure a constant linear acceleration of 0 m/s^2 over time (after removing gravitational acceleration). Letting the acceleration be a constant value, a, and letting the initial time, velocity, and position be equal to zero, the displacement of the accelerometer after a time t, given by (1) becomes

$$s(t) = \int_0^t \left(\int_0^t a \, dt \right) dt = \frac{1}{2} a t^2 \tag{2}$$

If a = 0 m/s^2, then the displacement after any time t is equal to 0 meters. However, if the acceleration measurement is offset by some constant bias error, then the displacement over time drifts away from the expected value (0 meters). Table 1 illustrates how small errors in acceleration lead to large displacement errors, even after a short time.

Table 1: Displacement Drift Over Time $\left[s(t) = \frac{1}{2} a t^2 \right]$ for Different Acceleration Errors

	***t* = 10 seconds**	***t* = 30 seconds**	***t* = 60 seconds**
***a* = 0.001 m/s^2**	0.05 meters	0.45 meters	1.8 meters
***a* = 0.01 m/s^2**	0.5 meters	4.5 meters	18 meters
***a* = 0.1 m/s^2**	5 meters	45 meters	180 meters

An accelerometer's ability to accurately track an object's displacement over time is highly dependent on the amount of uncertainty in its measurements and the total time over which displacement tracking occurs. Therefore, in many engineering applications where precise, long-term position tracking is required, relying solely on accelerometers is unacceptable. Nevertheless, accelerometers are still commonly incorporated into complex navigational systems due to several key advantages: size, weight, and speed.

Advances in silicone technology led to the development of micro electromechanical systems (MEMS) sensors, devices that are cheap, small, lightweight, and can contain multiple sensors in one unit [3, 4]. A MEMS accelerometer (with three axes to measure acceleration in the x, y, and z directions) is often packaged in combination with MEMS gyroscopes (inertial sensors that measure the angular rate of rotation) in Inertial Measurement Units (IMUs) [3]. MEMS IMUs, while on the low end of sensor accuracy, are ideal for electronics applications because they are sold on breakout boards by electronics manufacturers at a very low cost [3], and they have solderable pin connections that allow easy integration with other electronic devices. In addition, MEMS IMUs are small and lightweight, making them advantageous for a variety of robotics applications, such as the navigation of autonomous Unmanned Aerial Vehicles [5], Personal Dead-Reckoning devices [6, 7, 8], and controlling robotic manipulators [9].

In robotics, IMUs are commonly used for odometry: tracking a robot's movement (displacement and rotation) over time [10]. An odometry system that relies solely on an IMU is plagued with displacement estimation drift, but combining an IMU with other types of sensors, such as GPS, radar, LiDAR, or vision, can create a reliable odometry system where the strengths of one sensor balance the flaws of another [5, 10]. For example, in visual-inertial odometry, a camera is combined with an IMU so that the camera corrects the IMU's accumulated inaccuracies, and the IMU compensates for the camera's slow measurement rate [5, 10].

Even though accelerometers are often used in combination with other sensors to compensate for the accumulated error, it is still important to calibrate accelerometers before use to optimize their measurement accuracy. In robotics, situational awareness is key. The more uncertain a robot is about its environment, the less accurately a robot behaves. In an emergency, where a robot must suddenly rely on an accelerometer for displacement tracking, the sensor must provide the best possible measurements to minimize the robot’s position uncertainty. One way to improve an accelerometer’s measurement accuracy is by calibrating the sensor. In this study, calibration methods are tested on a low-cost MEMS accelerometer to explore how well they reduce error in displacement calculations over time, primarily focusing on removing constant bias from acceleration signals. Calibration methods are chosen that require minimal equipment and are computationally simple to execute. All code written for this project is publicly available on GitHub at https://github.com/willward20/UCA-Honors-Capstone.

Chapter 2: Literature Review

Introduction

The following literature review provides background information about characterizing and correcting errors in accelerometer measurements and subsequent velocity and displacement calculations. Discrepancies between expected and actual measurements are attributed to two types of error: random error and systematic error [11]. Random error is unpredictable, but it can be characterized statistically, and it is often modeled as white noise that contributes to displacement drift through random walk [12, 13, 14, 15]. On the other hand, systematic error persists throughout the entire dataset and is often attributed to an inherent flaw in a sensor or in an experimental procedure [11]. When acceleration data is integrated twice to calculate displacement estimates, both types of sensor error contribute to the displacement drifting away from true values.

As it will be seen in Chapter 5: Results, systematic uncertainty accounts for *significantly* more drift in displacement calculations than random uncertainty when using a MEMS MPU-9250 accelerometer. Therefore, while calibrating random errors is still an important area of research, this literature review focuses only on removing systematic uncertainty from sensor measurements. In the future, methods of calibrating random uncertainty, such as using a Kalman Filter [12], an autocorrelation function [14], Allan variance analysis [15], or wavelet analysis [16] could be investigated in more detail and incorporated into the MEMS accelerometer calibration system to create a more robust system.

Calibrating Systematic Error

Zero-State Detection

One method of removing systematic errors from accelerometers is to manually adjust measurements to 0 m/s^2 when the sensor is known (or approximated) to be at rest. Yu et al. [17] and Irfan et al. [18] both determine when an accelerometer is at rest by comparing live acceleration measurements to a predetermined threshold. The threshold behaves like a filter that is designed to catch and correct measurements that meet a certain condition while letting the others pass through unchanged.

One simple threshold, described by Irfan et al. [18], is an interval of acceleration values that sorts measurements into two categories: data points that fall within the interval and data points that fall outside the interval. A measurement that falls within the interval indicates that the sensor is most likely at rest, so the measurement is manually corrected to 0 m/s^2. On the other hand, measurements that fall outside the interval indicate that the sensor is most likely in motion, so the measurements should not be forced to zero [18]. While Irfan et al. [18] do not describe how their threshold was chosen, it is challenging to determine the best value. An interval that is too small might miss values that ought to be corrected to zero, but an interval that is too large might lose important information about linear acceleration, compromising displacement calculations.

Building on this idea, Yu et al. [17] choose a threshold based on the average mean and standard deviation of accelerometer measurements when the sensor is at rest. Then, as new data is collected, a zero-state adaptive compensation algorithm analyzes the data in small batches of 20 data points each. The mean and standard deviation of each batch are calculated, and depending on how they compare with the threshold values, every measurement in the batch is either forced to zero or passed unmodified. The algorithm continuously analyzes new batches of data and determines whether the sensor is at rest or in motion [17]. This approach considers the deviation in acceleration measurements over a short period of time, not just isolated instantaneous measurements like Irfan et al. [18] do. Incorporating standard deviation into the threshold algorithm gives a better representation of the sensor's current state by including a second point of reference to compare new data with known static behavior.

Unfortunately, while this calibration method is simple, relying solely on a zero-state threshold to calibrate acceleration measurements and effectively reduce drift in displacement calculations is unrealistic for many robotics applications. One major issue with zero-state threshold filtering is that calibration only takes place after the algorithm detects that the sensor is at rest. When a robot is in motion, such as a UAV in mid-flight or an autonomous vehicle driving on the interstate, the acceleration measurements are not calibrated, and the robot's odometry drifts significantly from the ground truth.

In response to this problem, Yu et al. [17] describe an additional step to remove errors in velocity calculations accumulated during movement by modeling the error as a linear function and subtracting it from the calculations. However, this method still does not resolve the problem because measurements taken during motion are not calibrated *until* the motion stops. This means that a robot would have to stop frequently during an operation to correct its odometry, and that is not always feasible. For some applications, such as tracking a person's position while walking or running [6, 8], this is highly effective. An IMU mounted to a person's boot will instantaneously come to rest each time a person takes a step, allowing for frequent calibration of the sensor.

However, for a robot that drives on wheels or soars through the air, a more general way of eliminating accelerometer error is needed.

Accelerometer Error Modeling

A more general way of calibrating an accelerometer is to devise a mathematical equation that relates true acceleration to measured acceleration. In the literature, several equations of varying complexity have been proposed to model errors in the accelerometer. In each model, the estimated ground truth acceleration is manipulated by several matrices that are composed of unknown parameters. By collecting data while the sensor is at rest and employing least-square fitting, the parameters in an accelerometer model can be optimized so that the equation relationship between true and measured acceleration fits the data.

Niu et al. [2] describes one of the simplest accelerometer models where acceleration measurements, $\mathbf{a}'$, and ground truth acceleration, $\mathbf{a}$, are three dimensional vectors containing *x, y,* and *z* components of acceleration:

$$\begin{bmatrix} a'_x \\ a'_y \\ a'_z \end{bmatrix} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} + \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} \tag{3}$$

In this model, the acceleration measured along each axis equals the true acceleration plus an unknown constant DC bias, **b**. A bias term is a source of systematic error that exists in every measurement, shifting the entire data set above or below the ground truth by a constant value. When graphing the measured accelerations versus true accelerations of each axis, the bias terms represent the intercepts of the functions. Since the only difference between the true and measured acceleration values in this model is the constant bias, the model assumes that the proportional relationship between the values is one-to-one. In other words, the model cannot adjust the slope of the fitting equation to better fit the data, if necessary.

A slightly more complex accelerometer model, described by Aggarwal et al. [15] does not assume that the ratio between measured and true acceleration is one-to-one. In addition to the bias terms, the model includes a matrix of scale factors (S_x, S_y, S_z) that is multiplied by the ground truth acceleration:

$$\begin{bmatrix} a'_x \\ a'_y \\ a'_z \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & S_z \end{bmatrix} \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} + \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} \tag{4}$$

When fitting the collected $\mathbf{a}'$ and $\mathbf{a}$ data to the model, the scale factors are optimized so that the slope of the fit function for each acceleration component (*x, y,* z) matches the slope of the data for each component. Compared with the model described by Niu et al. [2], the scale factor model adds an additional dimension to the fit equation. If there exists a non-one-to-one ratio between true and measured acceleration, the equation described by Aggarwal et al. [15] will give a more accurate calibration because it can model any slope.

The accelerometer error models discussed so far assume that the axes of the accelerometer are perfectly orthogonal to (rotated 90° relative to) each other. However, due to limitations in manufacturing precision, it is unlikely that the accelerometer axes are perfectly aligned. This is important to consider because the ground truth acceleration input into the model depends on the transformation matrix of the accelerometer axis relative to the reference frame of gravity. If the axes are misaligned, then each accelerometer axis could be measuring components from one or more ground truth acceleration components. Aggarwal et al. [15], Frosio et al. [19], Sipos et al. [20], and Särkkä et al. [21] all account for misalignment by modifying the 3x3 scale factor matrix, where the diagonal components represent the scale factors from the previous model, and the non-diagonal components represent misalignments between the three sensor axes. The general twelve-parameter model is given below:

$$\begin{bmatrix} a'_x \\ a'_y \\ a'_z \end{bmatrix} = \begin{bmatrix} S_{xx} & S_{xy} & S_{xz} \\ S_{yx} & S_{yy} & S_{yz} \\ S_{zx} & S_{zy} & S_{zz} \end{bmatrix} \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} + \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} \tag{5}$$

In the literature, different variations of the 3x3 scale factor model are described. For example, Frosio et al. [19] presents a modified version of this model, assuming that $S_{xy} = S_{yx}$, $S_{xz} = S_{zx}$, and $S_{yz} = S_{zy}$. Aggarwal et al. [15] and Särkkä et al. [21], on the other hand, do not make this distinction. Sipos et al. [20] take a different approach, setting $S_{yx} = S_{xz} = S_{yz} = 0$.

Velocity and Displacement Error Models

Errors (in the form of drift) can also be approximated in velocity and displacement calculations. In each accelerometer model presented in the previous section, a constant bias term propagates through the signal. When integrating the acceleration data over time, the velocity and displacement calculations can be approximated as the true values plus a polynomial error term that results from the DC bias in the acceleration [2, 18]:

$$\mathbf{v}' = \mathbf{v} + \mathbf{p_1}t + \mathbf{p_0} \tag{6}$$

$$\mathbf{d}' = \mathbf{d} + \frac{1}{2}\mathbf{q}_2 t^2 + \mathbf{q}_1 t + \mathbf{q}_0 \tag{7}$$

If acceleration signals are collected when the accelerometer is at rest, then **v** and **d** are equal to zero, and any non-zero velocity and displacement calculations are due to accumulated drift. (The model only directly accounts for error due to constant bias in the acceleration signal). The parameters **p** and **q** can be approximated using polynomial fitting, and the resulting calibration equations can be used to reduce the amount of error in velocity and displacement [2, 18].

Least-Squares Optimization of Parameters

The advantage of error models, like those presented in the previous sections, is that precise information about the sensor does not need to be known. Instead, data can be collected (with minimal equipment) and compared with theoretical expectations to fit a model to the data and determine unknown model parameters. Nui et al. [2], Aggarwal et al. [15], Frosio et al. [19], and Särkkä et al. [21] all use a form of least-squares optimization to estimate parameters in an error model.

In least squares fitting, a chosen trend function with one or more unknown parameters is fit to a set of data points by minimizing the sum of the square differences between the measured data points and the trend function [11, 22]. The minimum is found by optimizing the unknown parameters in the trend function for a given dataset. For a simple example, in Niu et al. [2] the error in velocity calculations (due to a DC bias in acceleration measurements) is modeled as a linear function $\mathrm{p}_1 t + \mathrm{p}_0$. To determine the coefficients p_1 and p_0, they minimize the sum of the squared differences between each velocity data point and the function [2]:

$$\sum_i [v_i' - (\mathrm{p}_1 t_i + \mathrm{p}_0)]^2 \tag{8}$$

Aggarwal et al. [15], Frosio et al. [19], and Särkkä et al. [21] all use a form of least squares fitting to determine the unknown parameters in their respective error models for the accelerometer. Depending on the complexity of the model, the number of parameters that must be optimized changes. In each calibration method, accelerometer data is collected while the sensor is at rest and in different orientations relative to the reference gravity vector.

Data Collection Procedures

Aggarwal et al. [15], Frosio et al. [19], Sipos et al. [20], and Särkkä et al. [21] all collect data for error model parameter optimization by rotating the sensor to different orientations and recording acceleration experienced by each axis when the sensor is at rest. Aggarwal et al. [15] collected data at six unique orientations on an approximately level surface so that the ground truth direction and magnitude of gravity were always known. In each collection position, a different face of the calibration cube was pointed up so that each axis experienced +1 g, 0 g, and -1 g at least once during the calibration process, as shown in Figure 2.

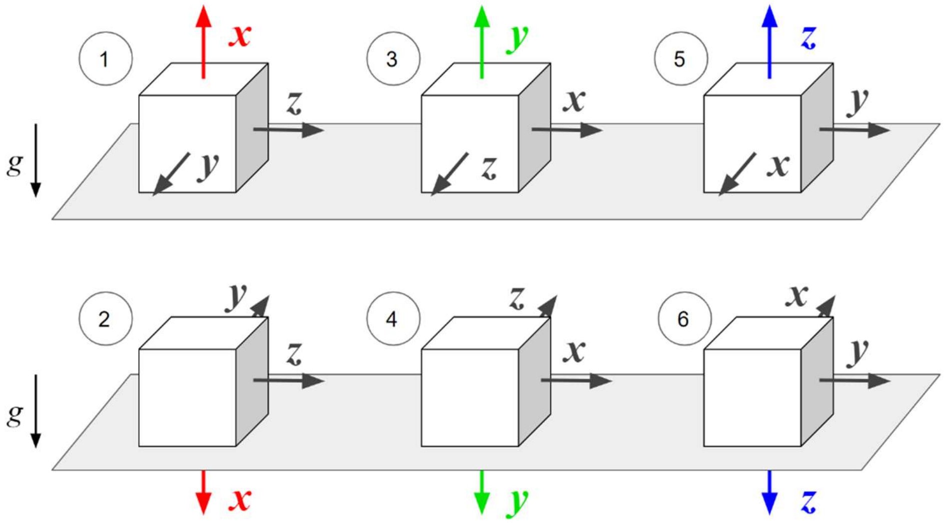


Figure 2: Six position data collection procedure

Särkkä et al. [21] expand on the previous collection process by placing the sensor, in its calibration cube, on an inclined plane and rotating the accelerometer to 24 unique orientations. For each of the six orientations used by Aggarwal et al. [15], Särkkä et al. [21] added three additional orientations by rotating the sensor 90° about the vertical axes. In addition, using an inclined plane, rather than a flat surface, exposes the accelerometer to more variety in acceleration magnitudes. The rotation angles of the platform are measured manually, but they are also included in the accelerometer error model to be optimized during least-squares fitting. This provides a point of reference to determine how accurate the model fitting is [21].

Similarly, Sipos et al. [20] suggest collecting accelerometer data at 36 positions while the sensor sits on a rotated platform. For three orientations (x-axis up, y-axis up, and z-axis up), Sipos et al. [20] suggest rotating the sensor 12 times about the vertical axis to achieve variety in the magnitudes of acceleration experienced by the other two axes. The angles of the rotating platform are measured directly for analyzing the accuracy of error model parameter optimization [20]. By contrast to the previous three studies, Frosio et al. [19] collected data at between 35 and 72 random sensor orientations using a rotating platform. Multiple sensors were tested, and the number of orientations varied for each. A motion capture system was used to measure the true angles of rotation for calibration analysis.

Chapter 3: Methodology

This chapter is divided into several sections. First, the hardware and software setup, including equipment used, circuit wiring, and data reading, are presented. Second, preliminary characterization tests are discussed, in which data is collected to learn more about the specific sensor used in this study. Third, data collection procedures used for calibration are presented, followed by an explanation of the calibration process.

Hardware and Software Setup

This study uses an MPU-9250 IMU breakout-board sensor, shown in Figure 3, that includes a three-axis accelerometer, gyroscope, and magnetometer. It is a very low-cost sensor package (less than \$20.00 US) that is manufactured by HiLetgo (hiltego.com) and sold by Amazon (amazon.com) [23, 24]. The MPU-9250 sensor itself (a small integrated circuit chip located on the breakout board) is manufactured by InvenSense, which provides a product data sheet detailing the characteristics and inner workings of the sensor array [25]. The datasheet indicates that the IMU is configured to register accelerations between -2 and +2 g's (±19.6 m/s^2), and that the "initial tolerance" of the sensor is ± 3% [25, p. 9].

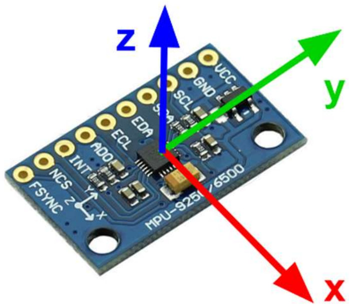


Figure 3: HiLetgo MPU-9250 IMU Sensor with coordinate axes labeled [24].

Accelerometer measurements are transmitted to a Raspberry Pi 4 for recording and processing through an I^{2}C connection. Table 2 labels the pin connections made between the IMU and the Pi [26]. The IMU is mounted inside a 3D-printed calibration cube that was sanded on all sides after printing to ensure approximate levelness [27]. The cube allows easy 90° rotation of the IMU during calibration. The full setup of the IMU connected to the Raspberry Pi is shown in Figure 4. All data collection, recording, and analysis are performed by scripts written in the programming language Python.

Table 2: Raspberry Pi and IMU Pin Connections

IMU Breakout Board Pins	Raspberry Pi 4 GPIO Pins
VCC	3V3 Power
GND	Ground
SCL	GPIO 3 (SCL)
SDA	GPIO 2 (SDA)

Before data collection begins, the accelerometer sensor is initialized and configured using the *mpu9250_i2c.py* program written by Joshua Hrisko and made publicly available in the *mpu92-calibration* repository on GitHub [28]. In addition, during data collection, functions in Hrisko's program are called to read raw bits from the sensor and convert them to acceleration in g. During an experiment, data is collected in a continuous loop. In each iteration, an acceleration data point (in units of g) for each axis is retrieved and immediately appended to a new line in a CSV file, along with a corresponding time stamp. Although saving data points one at a time may slow down the collection rate, it avoids appending data to a list, which risks data loss in the event of a power shortage. The average data collection frequency is 180 Hz (data points per second).

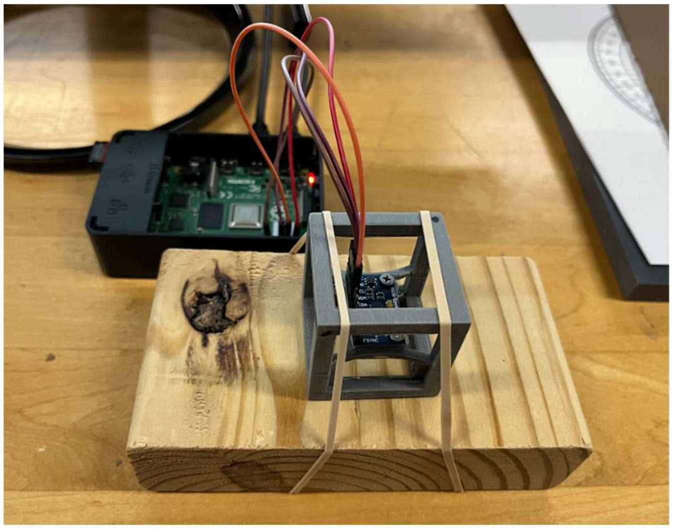


Figure 4: IMU mounted in a calibration cube and connected to a Raspberry Pi

Preliminary Characterization Tests

Before planning and performing lengthy data collection and calibration procedures, initial tests are performed while the sensor is at rest to better understand the accelerometer's limitations. First, triaxial accelerometer data is collected at 180 Hz for one hour, generating a dataset of 645,714 measurements for each axis (x, y, and z). Then, the data is analyzed using statistical analysis to understand whether systematic error in the measurements is consistent. Determining how much the systematic error changes over time in acceleration measurements is important to understand before collecting calibration data. If measurements taken over a small period (30-60 seconds) do not have a consistent mean when the accelerometer is at rest, then deriving calibration model parameters using multi-position calibration techniques will result in a calibration model with high uncertainty. This calibration technique relies on precise mean measurements at different orientations, so imprecise measurements will lead to greater uncertainty in the optimized parameters.

Second, data is collected in 60 individual trials that last one minute each. After each trial, the data collection program is restarted, re-initializing the IMU. After every tenth trial, the Raspberry Pi was rebooted. Finally, after the 30th trial, the Pi was shut down for two hours before performing the last 30 trials. Each trial is analyzed statistically and compared with the others. The purpose of this experiment is to determine whether systematic error (characterized by the mean of a set of measurements) changes between collection trials. If systematic error in the accelerometer changes significantly every time the sensor is turned off and on again, then the sensor needs to be calibrated before every use.

Finally, the preliminary data collected through these experiments are also used to determine how much of the calculated displacement error is caused by noise (random uncertainty) and how much of the displacement error is caused by bias (systematic uncertainty). If all systematic uncertainty is eliminated from a set of accelerometer measurements, any leftover displacement drift error is a result of noise. It is important to understand the contribution of noise in displacement error before analyzing the success of systematic error calibration because it sets a limit on how much displacement error the systematic calibration technique can realistically remove.

Statistical Analysis

Statistical analysis is a valuable tool for interpreting the preliminary data and the calibration data. In particular, statistical analysis can reveal important information about random and systematic uncertainty. Given a set of N measurements, x_i, of a quantity x, the mean, $\bar{x}$, of the set of measurements represents the best estimate of the quantity [11]

$$\bar{\mathrm{x}} = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{9}$$

In some cases, the best estimate of systematic uncertainty is given by the mean, representing a constant offset above or below the true value of the quantity.

If systematic uncertainty is negligible (or constant) in a set of measurements, then the standard deviation, σ_x of the data approximates the random uncertainty [11].

$$\sigma_x = \sqrt{\frac{1}{N-1}\sum(x_i - \bar{x})^2} \tag{10}$$

Any one measurement from the data set, then, will have a value of $x_i \pm \sigma_x$. As a side note, the quantity σ_x^2 is called the variance of the variable x [29]. In addition, the uncertainty of the mean of the whole data set is given by the standard deviation of the mean $\sigma_{\bar{x}}$ (SDOM) [11]

$$\sigma_{\bar{x}} = \frac{\sigma_x}{\sqrt{N}} \tag{11}$$

The mean of the data can be written as $\bar{x} \pm \sigma_{\bar{x}}$.

Experimental Data Collection

Accelerometer data collection is performed based on the method described by Aggarwal et al. [15], in which the accelerometer is rotated to six different orientations on a flat, level surface. The sensor is rotated to each orientation, and acceleration measurements are collected for 30 seconds (180 Hz) while the device is at rest. The data recorded at each orientation is then averaged and saved to a CSV file, along with a ground truth label that indicates the actual gravity components experienced by each axis. If the surface that the sensor rests on is truly level, then the direction of gravity, relative to the axes of the accelerometer, is always known. Having a reliable ground truth estimate is essential for accurate least-squares fitting. In the lab, a level surface is approximated using a flat plastic plank and several metal washers. The washers are slid underneath the corners of the plank until the surface is level, approximately verified by five different bubble levels. Figure 5 shows the experimental setup. To ensure that the IMU is always stationary during data collection, a 100 gram weight rests on top of the calibration cube.

Next, the six-position acceleration data is used to optimize parameters in three different accelerometer output models, giving calibration equations. After all orientations in the data collection procedure are recorded, additional static data at one orientation is recorded to a CSV file for one minute. Each measurement is paired with a time stamp. This dataset of acceleration over time is used to test the accuracy of the calibration models that are optimized using the multi-orientation data. The process of collecting six-position data, optimizing the error models, and testing them on static data is repeated over three consecutive trials.

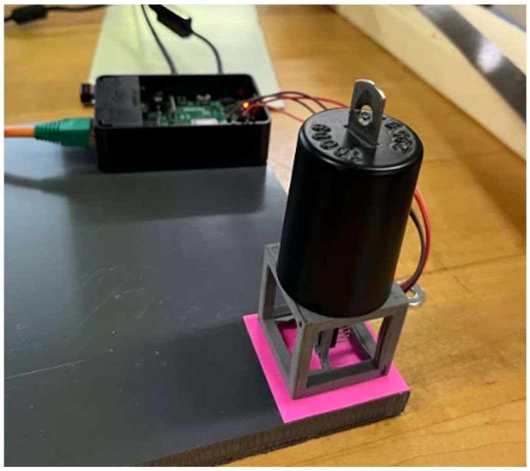


Figure 5: Six-Position data collection experimental setup

Optimizing Accelerometer Model Parameters

After collecting calibration data, three accelerometer output models from the literature (Equations 3, 4, and 5) are fit to the data using the *curve_fit* function from the python package *scipy.optimize*. The inputs into the *curve_fit* function are a fitting equation, an array of true acceleration values, a corresponding array of measured acceleration values, an array of uncertainty values in the measurements, and initial guesses of the equation parameters [30]. The outputs of the *curve_fit* function are the optimized parameters and a covariance matrix that approximates the uncertainty in the parameters [30]. Matrix members on the diagonal of the matrix represent the variances (standard deviation squared) of each parameter, while the off-diagonal components represent the covariances between pairs of parameters [31]. When analyzing the optimized parameters, the square root of the variances in the matrix characterizes the uncertainty of the optimized parameters. For each data collection method, the three accelerometer models are optimized in order of complexity, and the optimized parameters of one model are used as initial guesses in the next model. Three models are listed below in order of complexity:

$$\text{Model 1:} \quad \begin{bmatrix} a'_x \\ a'_y \\ a'_z \end{bmatrix} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} + \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} \text{ [2]} \tag{3}$$

$$\text{Model 2:} \quad \begin{bmatrix} a'_x \\ a'_y \\ a'_z \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & S_z \end{bmatrix} \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} + \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} \text{ [15]} \tag{4}$$

$$\text{Model 3:} \quad \begin{bmatrix} a'_x \\ a'_y \\ a'_z \end{bmatrix} = \begin{bmatrix} S_{xx} & S_{xy} & S_{xz} \\ S_{yx} & S_{yy} & S_{yz} \\ S_{zx} & S_{zy} & S_{zz} \end{bmatrix} \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} + \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} \text{ [15, 19, 20, 21]} \tag{5}$$

First, the bias vector in Model 1 is optimized. Then, the optimized bias vector becomes the initial guess of the bias in Model 2, increasing the speed of the fitting. Next, the bias and scale factor vectors optimized in Model 2 become the initial guesses in Model 3. Finally, the biases, scale factors, and misalignments are optimized in Model 3. After optimizing all three accelerometer models, each model is used to calibrate the static test data collected during the trial. The mean and standard deviation of the uncalibrated data are compared with the mean and standard deviation of the data after calibrating with each model. Then, each set of data is integrated twice over time to calculate the measured displacement of the sensor. The displacement drift is compared between each calibration model to analyze their calibration accuracy and determine which model is best.

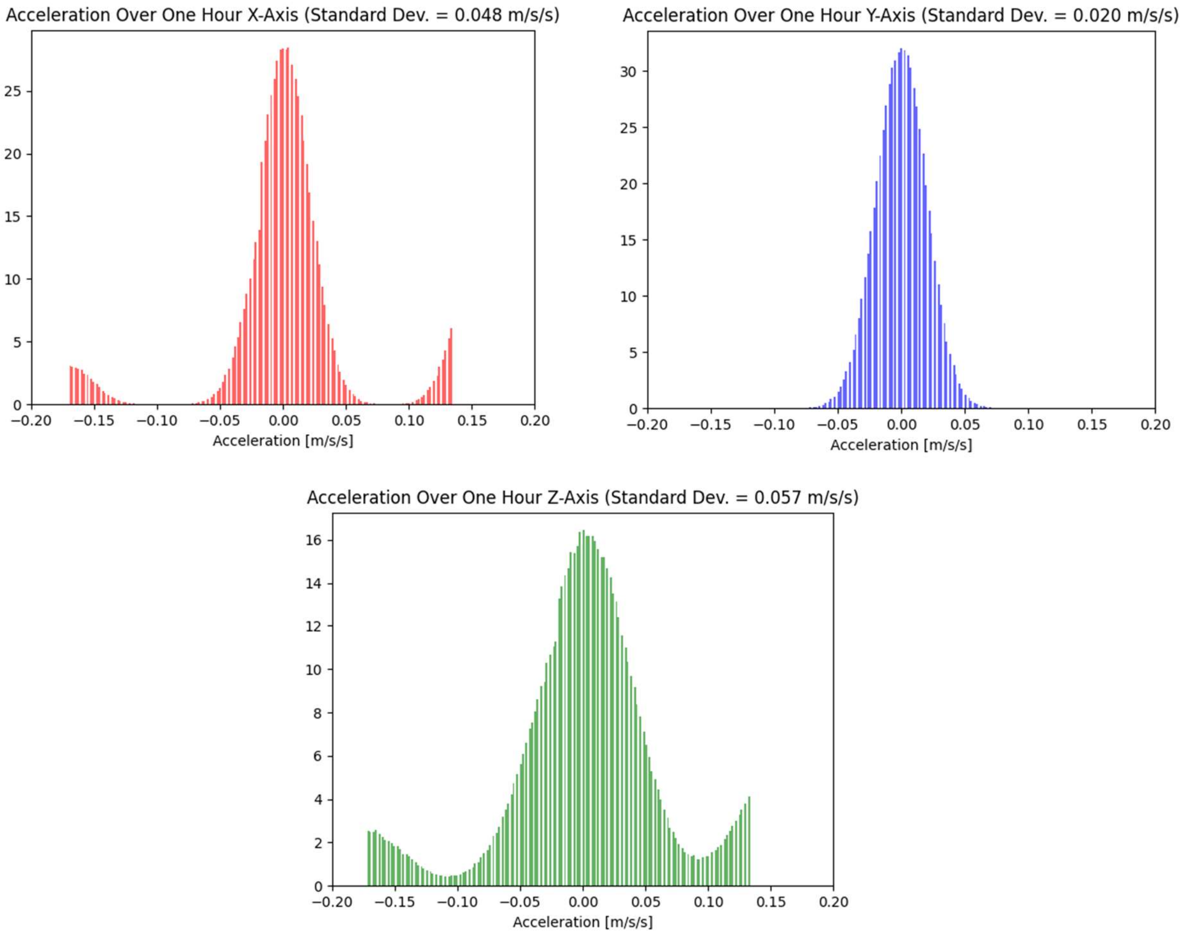
Final Calibration Trial

After determining the best accelerometer calibration model, one final calibration trial is performed that incorporates the displacement error modeling technique described by Niu et al. and Irfan et al. [2, 18]. First, six-position data collection is performed as discussed previously, followed by two rounds of 60 second data collection while the senor is on the level surface with the z-axis pointed vertically. Next, the parameters in Model 3 are optimized using the six-position data. Model 3 is used to calibrate the first set of 60 second static data, and the acceleration is integrated twice to calculate the displacement over time of the sensor. Then, least-squares optimization is used to fit Equation 7 (see Chapter 2) to the displacement data. This function models the remaining error in displacement after calibrating the acceleration with Model 3. Finally, the second set of 60 second static data is calibrated using Model 3, integrated to calculate the displacement, and calibrated using the displacement error model. The goal of this process is to introduce a second round of calibration that could further reduce the amount of drift in displacement calculations.

Chapter 4: Results

Preliminary Characterization Tests

The first question that the preliminary characterization tests attempt to answer is whether systematic error in accelerometer measurements changes significantly over time when the sensor is at rest in a controlled environment. One hour of static test data was collected at a rate of 180 Hz while the sensor was mounted to a stationary, unlevel table. Figure 6 shows histograms of the data from each axis (x, y, and z). Before plotting the histograms, the mean of the data was calculated and subtracted from the entire dataset to center the data around 0 m/s^2. Plotting in this way makes it easier to visualize the spread of the data.



Acceleration Over One Hour X-Axis (Standard Dev. = 0.048 m/s/s)

Acceleration Over One Hour Y-Axis (Standard Dev. = 0.020 m/s/s)

Acceleration Over One Hour Z-Axis (Standard Dev. = 0.057 m/s/s)

Figure 6: Histograms of each axis (x, y, and z) of the one-hour accelerometer data

The mean, standard deviation, and standard deviation of the mean (SDOM) of the one-hour data are calculated and displayed in Table 3. **Note that only the order of magnitude of the mean, not the value of the mean itself, is important in this analysis.** (Data was collected on an unlevel surface). When compared with the magnitude of the mean, the SDOM is very small, indicating high certainty in the mean of the data, even though the uncertainty in one measurement (given by the standard deviation) is somewhat large. This is an important result because it assures that the systematic error in acceleration measurements is consistent when the sensor is held at rest at one orientation. Therefore, when calibration models are derived from measurements, it is reasonable to assume that the model's accuracy does not significantly change while it is used to calibrate new data for a long time *if the new data is collected immediately after deriving the model*. In calibration experiments, as described in the Methodology section, static test data is collected immediately after collecting the multi-position data that is used for parameter optimization.

Table 3: Statistical Analysis of One Hour Accelerometer Static Data (unlevel surface)

	Mean (m/s^2)	Standard Deviation (m/s^2)	SDOM (m/s^2)
x	1.08800	0.05	0.00006
y	0.53304	0.02	0.00002
z	-2.27746	0.06	0.00007

The histograms for the x and z axes in Figure 6 exhibit unusual behavior in the distribution. Both graphs have "wings" to the left and right of the mean where the frequency of values starts to increase when moving away from the center. Strangely, at a certain distance away from the mean, the frequency suddenly drops to zero, making the plot look like it was cut off. The cause of this behavior is uncertain, but based on the SDOM's small size, it does not largely impact the mean of the data.

Next, 60 one-minute data collection trials were performed while the sensor was at rest and attached to a stationary wooden block. The mean of each trial was calculated and graphed, as shown in Figure 7. The means deviate between trials, and the most significant change occurred when the Raspberry Pi was shut off for two hours between the 30th and 31st trials. To analyze the results of this test, first, the SDOM of each one-minute trial was calculated. The average SDOMs of the x, y, and z axes were 0.0003, 0.0002, and 0.0004, respectively. Since these values are much smaller than the mean sensor measurements shown in Figure 7, it is reasonable to assume that the mean reported in each trial is accurate.

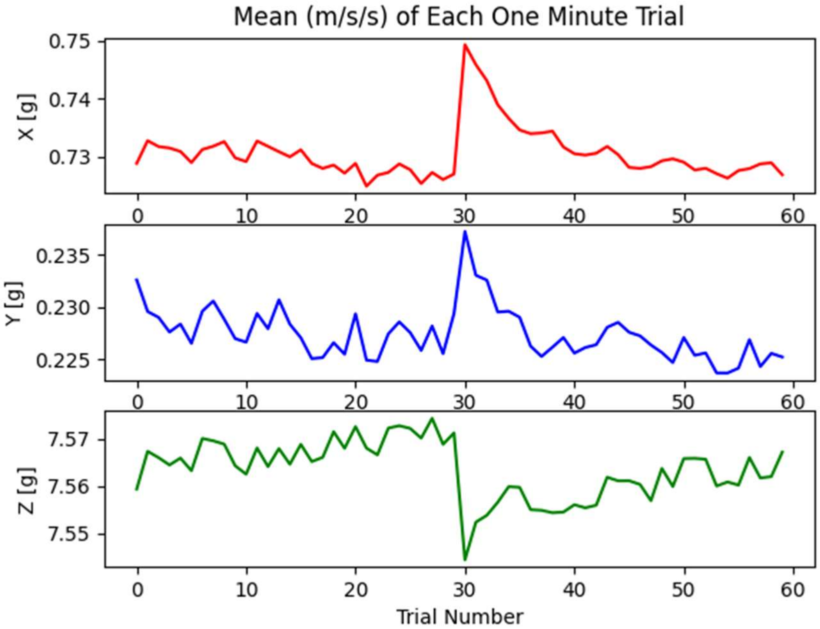


Figure 7: Mean of all 60 one-minute trials.

However, it is clear from the graph that the means between trials still deviate significantly. This quantity is approximated by taking the standard deviation of the entire set of trials. The standard deviations for the x, y, and z axes (0.004, 0.002, and 0.006, respectively), represent the uncertainty in the mean of each trial, relative to the entire set of trials, which are one order of magnitude larger than SDOM uncertainty reported by each trial, relative to itself.

These results imply that the systematic uncertainty of the sensor is not consistent when the device and/or the Raspberry Pi are rebooted. Therefore, as discussed in Chapter 3, when calibration methods are tested, test data is collected immediately after calibration data is collected to rule out this source of error.

Finally, the effect of noise on displacement calculation errors is investigated using data from one of the 60 one-minute trials. First, the measurements for each axis are converted to units of m/s^2, and the acceleration due to gravity (9.797 m/s^2) is subtracted from the z-axis because it is vertically oriented on the approximately level platform). Next, the measurements are numerically integrated twice over time to calculate the displacement, as shown in Figure 8. After only one minute of data collection, the displacement along each axis exceeds an absurdly large magnitude of 1000 meters. Since the sensor is known to be at rest, the calculated displacement should be zero meters. Therefore, there is a significant error in the calculated displacement when the sensor is not calibrated.

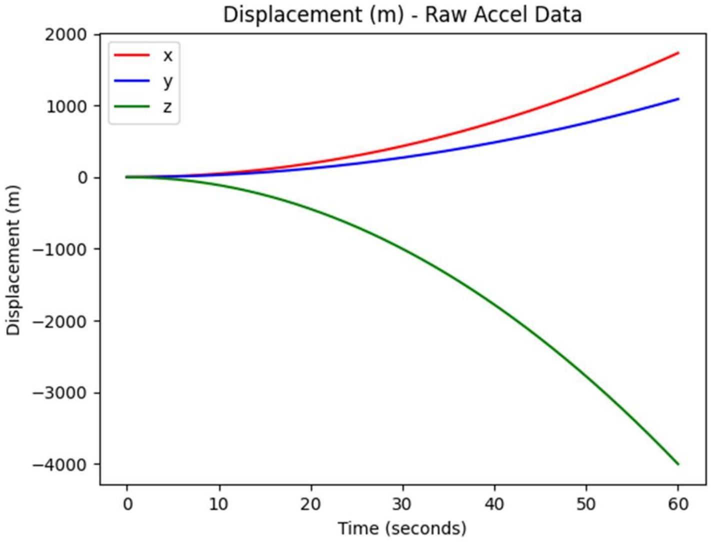


Figure 8: Displacement over time calculated from raw accelerometer data.

Next, the mean of each axis is calculated and subtracted from the data to remove systematic error (constant bias) from each axis. As with the data collected over one hour, the standard deviation of the mean is sufficiently small to justify assuming a near-constant bias over the one minute of data collection. The bias-corrected acceleration data is integrated twice to calculate the new displacement over time, as shown in Figure 9. With bias removed, the displacement after one minute does not exceed 1 meter. This dramatic improvement from the displacement error in the raw data (1000 meters) clearly indicates that the effect of noise on the displacement error (when the sensor is at rest) is insignificant compared with the effect of the bias.

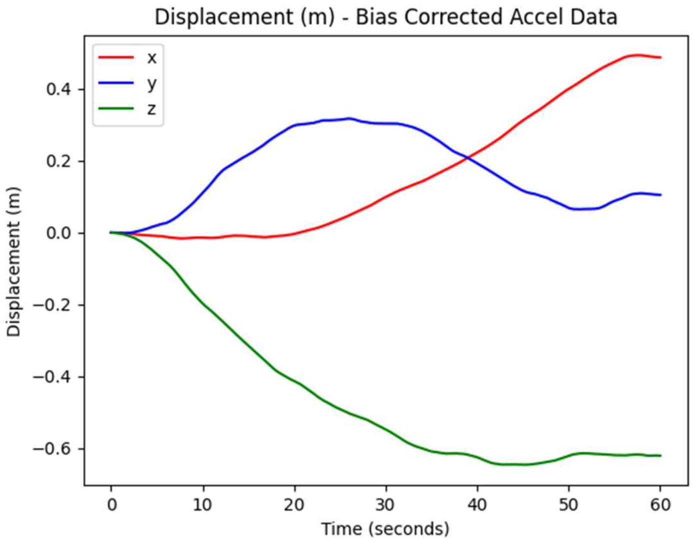


Figure 9: Displacement calculated from accelerometer data after bias is removed.

Six-Position Calibration

Six-position data was collected over three trials, and the data from each trial was used to optimize the parameters in the three accelerometer output models (see Table A-3 in Appendix A for a complete table of optimized parameters). As discussed in the Methodology chapter, one minute of static test data was recorded after every data collection trial, along with time stamps.

After optimizing the three accelerometer models, each model was used to calibrate the static test data to compare the effectiveness of the models. Table 4 shows the mean of each axis of the test data in Trial 1 before and after calibrating with each model. (Trials 2 and 3 are not included in Table 4 because they are very similar to Trial 1, but the full results from all three trials are found in Table A-1 in Appendix A). In Table 4, the uncertainty percentage of the mean was derived from the SDOM of the test data.

Table 4: Mean of Trial 1 Static Test Data Before and After Calibration

Calibration Method Used	x mean (m/s^2)	y mean (m/s^2)	z mean (m/s^2)
No Calibration	0.9617 ± 0.03%	0.6050 ± 0.09%	-2.2238 ± 0.019%
Model 1: Bias	0.0638 ± 0.5%	0.1411 ± 0.4%	0.2296 ± 0.18%
Model 2: Add Scale Factors	0.0638 ± 0.5%	0.1411 ± 0.4%	0.0140 ± 3%
Model 3: Add Misalignments	-0.0252 ± 1.2%	0.0367 ± 2.2%	0.0139 ± 3%

Since the data was collected while the sensor was at rest, the expected mean acceleration of each axis is zero (after gravity is removed from the z-axis). Therefore, non-zero mean values in Table 4 represent an error in the mean acceleration measurement. Looking at Table 4, every method of calibration improved the mean of the test data, compared to the original uncalibrated mean. Interestingly, some calibration models had more effect on the mean than others. For example, in Trial 1, the means of the x and y axes after calibrating with Model 1 are nearly identical to the means after calibrating with Model 2. On the other hand, the means in both axes improved by 0.04 and 0.1 m/s^2 between Model 2 and Model 3. Similarly, the mean of the z-axis after calibrating with Model 2 and Model 3 are comparable, while the improvement between Model 1 and Model 2 is approximately 0.21 m/s^2. Overall, calibrating with Model 3 resulted in the least amount of mean error across all three axes and all three trials (see Appendix A).

The results from Table 4 are verified further by taking the calibrated and uncalibrated data in each trial and integrating them over time (using the recorded time stamps). Table 5 shows the results from Trial 1, and Table A-2 in Appendix A contains the results of all three trials. (Again, the results of Trials 2 and 3 are comparable to Trial 1). With no calibration, the displacement along each axis drifts by more than 1000 meters after 60 seconds (sitting at rest on the table). After calibrating with Model 1, the displacement decreases by approximately one order of magnitude. Finally, after calibrating with Model 3, the drift in each axis is reduced below 100 meters.

Table 5: Final Integrated Displacement of Trial 1 Static Test Data (60 seconds)

Calibration Method Used	**x (m)**	**y (m)**	**z (m)**
No Calibration	1731	1089	-4002
Model 1: Bias	115	254	413
Model 2: Add Scale Factors	115	254	25
Model 3: Add Misalignments	-45	66	25

Final Calibration Trial

In the final calibration trial, another set of six-position data was collected, along with two 60 second static test data sets. After collecting data, the parameters in Model 3 were optimized, and the model was used to calibrate the first 60 second static test data set. After integrating the calibrated acceleration to calculate the displacement, the parameters in the displacement model were optimized to fit the model to the displacement curve. Table A-4 in Appendix A gives all the derived parameters in this trial. Finally, the second set of test data was calibrated using Model 3, integrated for displacement, and calibrated a second time using the displacement model. Table 6 compares the final calculated displacement of the second set of test data after 60 seconds when no calibration, acceleration calibration, and both acceleration and displacement calibration are used.

Table 6: Displacement of Final Trial Static Test Data Set Two (60 seconds)

Calibration Method Used	x (m)	y (m)	z (m)
No Calibration	1760	1061	-4014
Model 3 (Acceleration Only)	-35	96	25
Model 3 and Displacement Model	-1	-1	-7

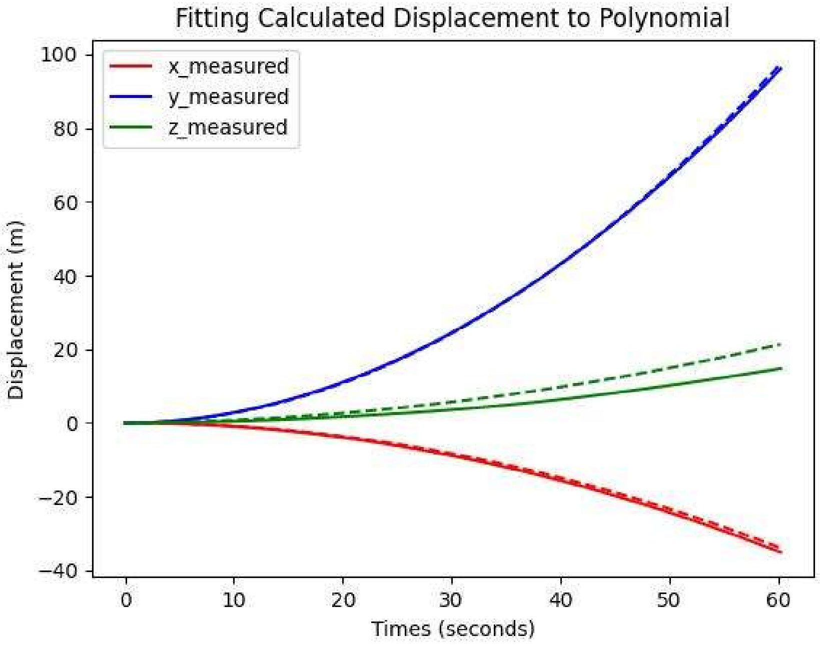


Figure 10: Displacement error model fitted to new displacement data.

Additionally, Figure 10 shows how well the displacement model fit to the second set of displacement data (after calibrating the acceleration and integrating). The fit is well aligned with the x and y displacement data (corresponding with a very small displacement drift along those axes in Table 6), but the fit begins to diverge significantly from the z displacement data, resulting in a larger accumulated displacement drift after 60 seconds. Overall, when compared with no calibration and acceleration calibration only, calibrating with the acceleration and displacement models significantly improves the displacement results, even though significant drift still occurs.

Chapter 5: Conclusions

MEMS accelerometers are compact, light, and cheap electronic sensors that might seem ideal for tracking a robot's odometry. However, inherent flaws in the sensor's linear acceleration measurements make them impossible to use effectively for position tracking without the aid of calibration or additional sensors. In this project, a mathematical modeling approach was used to calibrate the accelerometer's output data and attempt to reduce the effect of displacement drifting to the point where the sensor could be used practically in robot odometry. Of the three accelerometer models presented, Model 3 (which accounts for bias, scaling, and misalignments) performed the best, dramatically reducing the amount of drift in displacement calculations from the scale of 1000 to the scale of 100. Additionally, calibrating the displacement calculations using a polynomial error model reduced the displacement drift further.

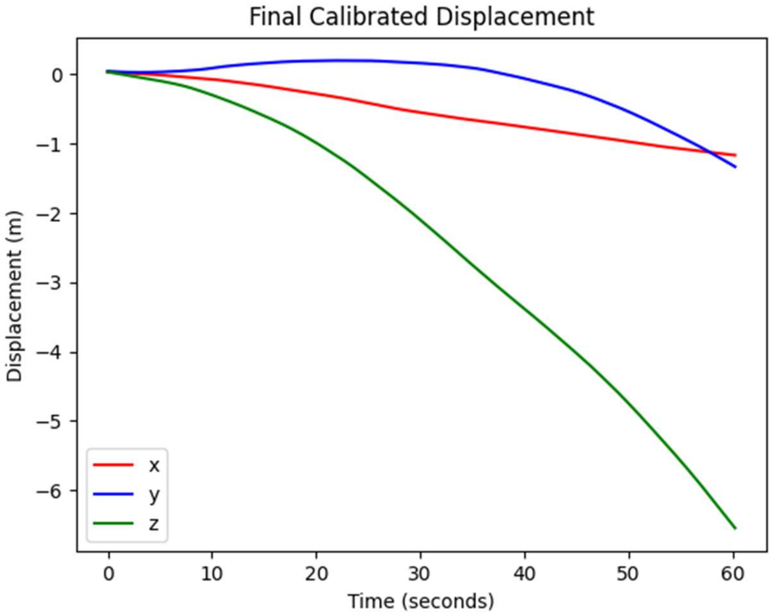


Figure 11: Final calibrated displacement (using accelerometer Model 3 and the displacement error model)

Figure 11 shows a graph of the displacement over time after calibrating the data with the accelerometer model and the displacement model. Even though the drift in displacement was reduced significantly compared with the uncalibrated results, the displacement functions shown in Figure 11 illustrate how unstable the system is. Trying to calibrate an accelerometer so that it can be used by itself for displacement odometry is like swimming against a strong current. No matter how much the data accuracy improves, small amounts of error in acceleration will always exist and accumulate over time in displacement calculations. Depending on how long a robot needs accurate robot odometry, using an accelerometer by itself may or may not be feasible. In the future, further testing should be performed to investigate how well the accelerometer approximates distance traveled while attached to a moving robot.

Sources of Error

Although the acceleration calibration models (optimized using six-position data collection) improved the displacement approximations, even the smallest amount of drift after employing this calibration model, 25 meters achieved in Trial 1, is still unacceptably large, considering that the sensor only collected data for 60 seconds. 25 meters (approximately 82 feet) is much, much larger than the size of the IMU breakout board used in the experiments (2.5 by 1.5 cm), so an error in displacement of that size makes any odometry calculation completely useless. Therefore, while the six-position calibration method using least-squares fitting improves the sensor's measurements, it is not enough by itself for practical, long-term robotics odometry tracking. On the other hand, from the results of the Preliminary Characterization Tests, we know that it *is* possible to remove systematic uncertainty directly from acceleration measurements when the sensor is at rest and produce far more accurate odometry. This raises the question of why the six-position calibration method was so ineffective.

One source of error in the experimental system that contributes to drift in the displacement results is noise (random uncertainty) in the measurements. However, in the preliminary characterization tests, random uncertainty only contributed to 0.4 to 0.6 meters of drift, which is much smaller than the total drift in the three six-position calibration trials, even after calibrating with Model 3. Therefore, more significant sources of error must exist in the system.

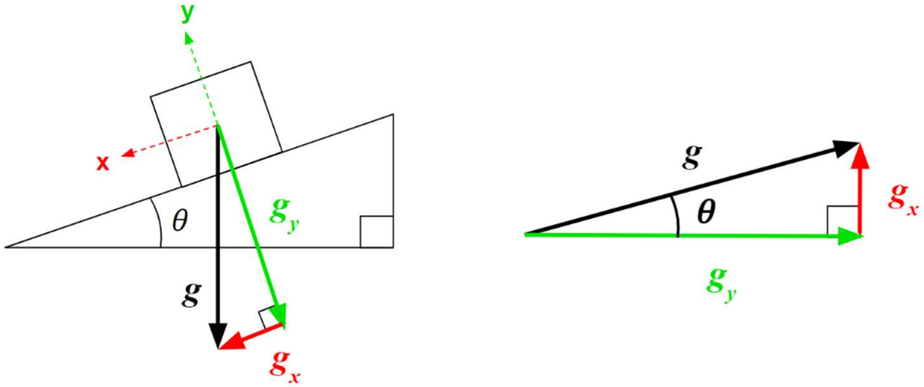


Figure 12: Calculating the angle of the table, relative to the direction of gravity.

One possibility could be that the platform that the accelerometer rested on during the data collection is not as level as was assumed. This is tested by using the accelerometer x- and y-axis mean measurements from Table 4 to calculate the approximate lateral angles of the accelerometer relative to a perfectly level surface. Figure 12 shows a diagram for calculating the approximate angles using equations 12 and 13.

$$\theta_x = \sin^{-1}\frac{a_x}{g} \tag{12}$$

$$\theta_y = \sin^{-1}\frac{a_y}{g} \tag{13}$$

When the x and y acceleration measurements are uncalibrated, the calculated angles are 5.6° and 3.5°, respectively. Setting up a simple protractor with suspended mass on the edge of the table used for data collection shows that the calculated angles are far too large. The protractor measures an angle between 0 ± 1°. Next, when the x and y acceleration measurements are calibrated using Model 3, the calculated angles are -0.15° and 0.21°, respectively. These calculated angles fall within the uncertainty bars of the measured angle of the table, so they are reasonable estimates.

It is unclear why the six-position calibration method was unable to remove more uncertainty from acceleration measurements than reported. However, a more detailed analysis of the sensor, its limitations, and the limitations of the experimental setup/calibration model could reveal the sources of error. It is highly likely that the sensor itself is often inconsistent in the values it reports between trials, leading to slightly different acceleration values that accumulate into large error when calculating displacement.

Future Work

In the future, the calibration methods discussed in this project (accelerometer Model 3 and the displacement error model) should be tested on multiple IMU sensors to investigate the consistency in results. Additionally, the methods should be tested on a moving robot to simulate real-world applications in which odometry estimations are needed. Finally, while these calibration methods are simple and quick to implement, more advanced calibration methods could be researched and tested to investigate whether there is a more effective way of using accelerometers for displacement tracking.

Appendix A: Six-Position Data Collection Full Results

Table A-1: Mean of Trial Test Data Before and After Calibration

Trial 1			
Calibration Method Used	**x (m/s^2)**	**y (m/s^2)**	**z (m/s^2)**
No Calibration	0.9608 ± 0.03%	0.6045 ± 0.09%	-2.2217 ± 0.018%
Model 1: Bias Only	0.0638 ± 0.4%	0.1410 ± 0.4%	0.2294 ± 0.17%
Model 2: Add Scale Vector	0.0637 ± 0.4%	0.1410 ± 0.4%	0.0140 ± 3%
Model 3: Add Misalignments	-0.0252 ± 1.0%	0.0367 ± 1.4%	0.0139 ± 3%
Trial 2			
Calibration Method Used	**x (m/s^2)**	**y (m/s^2)**	**z (m/s^2)**
No Calibration	0.9476 ± 0.04%	0.6004 ± 0.08%	-2.2026 ± 0.02%
Model 1: Bias Only	0.0572 ± 0.6%	0.1420 ± 0.4%	0.2408 ± 0.17%
Model 2: Add Scale Vector	0.0572 ± 0.6%	0.1420 ± 0.4%	0.0261 ± 1.6%
Model 3: Add Misalignments	-0.0194 ± 1.8%	0.0354 ± 1.4%	0.0261 ± 1.6%
Trial 3			
Calibration Method Used	**x (m/s^2)**	**y (m/s^2)**	**z (m/s^2)**
No Calibration	0.9452 ± 0.04%	0.5940 ± 0.08%	-2.2057 ± 0.19%
Model 1: Bias Only	0.0621 ± 0.6%	0.1383 ± 0.3%	0.2246 ± 0.19%
Model 2: Add Scale Vector	0.0621 ± 0.6%	0.1383 ± 0.3%	0.0056 ± 7%
Model 3: Add Misalignments	-0.0165 ± 2%	0.0334 ± 1.4%	0.0056 ± 7%

Table A-2: Final Integrated Displacement of Trial 1 Static Test Data

Trial 1			
Calibration Method Used	**x (m)**	**y (m)**	**z (m)**
No Calibration	1729	1088	-3999
Model 1: Bias	115	254	412
Model 2: Add Scale Factors	115	254	25

Model 3: Add Misalignments	-45	66	24
Trial 2			
Calibration Method Used	**x (m)**	**y (m)**	**z (m)**
No Calibration	1705	1080	-3961
Model 1: Bias	103	255	433
Model 2: Add Scale Factors	103	255	47
Model 3: Add Misalignments	-35	63	46
Trial 3			
Calibration Method Used	**x (m)**	**y (m)**	**z (m)**
No Calibration	1702	1071	-3970
Model 1: Bias	113	251	404
Model 2: Add Scale Factors	113	251	10
Model 3: Add Misalignments	-29	62	10

Table A-3: Model Parameters with Uncertainty			
Model 1: Bias Only			
Parameter	**Trial 1**	**Trial 2**	**Trial 3**
b_x	0.093 ± 3%	0.090 ± 3%	0.089 ± 3%
b_y	0.043 ± 7%	0.043 ± 7%	0.043 ± 7%
b_z	-0.251 ± 3%	-0.248 ± 2%	-0.248 ± 2%
Model 2: Bias and Scale Factor Vector			
Parameter	**Trial 1**	**Trial 2**	**Trial 3**
b_x	0.093 ± 4%	0.090 ± 3%	0.089 ± 3%
b_y	0.043 ± 7%	0.043 ± 7%	0.043 ± 7%
b_z	-0.251 ± 1.2%	-0.250 ± 1.2%	-0.249 ± 1.2%
S_{xx}	1.000 ± 1.3%	1.000 ± 0.9%	1.000 ± 0.7%
S_{yy}	1.000 ± 1%	1.000 ± 0.8%	1.000 ± 0.8%

S_{zz}	1.022 ± 0.5%	1.022 ± 0.5%	1.023 ± 0.5%
Model 3: Bias and Scale Factor Matrix with Misalignments			
Parameter	**Trial 1**	**Trial 2**	**Trial 3**
b_x	0.0916 ± 0.5%	0.0907 ± 0.3%	0.0903 ± 0.7%
b_y	0.046 ± 4%	0.0461 ± 4%	0.0459 ± 4%
b_z	-0.2501 ± 0.3%	-0.2491 ± 0.2%	-0.2480 ± 0.3%
S_{xx}	1.0001 ± 0.17%	0.9999 ± 0.07%	0.9999 ± 0.14%
S_{yy}	1.000 ± 0.5%	1.000 ± 0.4%	1.000 ± 0.4%
S_{zz}	1.0220 ± 0.11%	1.0220 ± 0.1%	1.0225 ± 0.13%
S_{xy}	0.0080 ± 9%	0.0063 ± 6%	0.0066 ± 14%
S_{xz}	0.0086 ± 9%	0.0078 ± 5%	0.0078 ± 17%
S_{yx}	-0.007 ± 30%	-0.006 ± 30%	-0.007 ± 30%
S_{yz}	0.006 ± 50%	0.007 ± 40%	0.007 ± 40%
S_{zx}	-0.0100 ± 11%	-0.0095 ± 11%	-0.0094 ± 14%
S_{zy}	-0.0047 ± 30%	-0.0047 ± 23%	-0.0042 ± 33%

Table A-4: Final Calibration Trial Model Parameters with Uncertainty	
Model 3: Bias and Scale Factor Matrix with Misalignments	
Parameter	**Optimized Value**
b_x	0.0922 ± 0.8%
b_y	0.0462 ± 4%
b_z	-0.2496 ± 0.25%
S_{xx}	0.9997 ± 0.24%
S_{yy}	1.001 ± 0.4%
S_{zz}	1.0228 ± 0.09%
S_{xy}	0.0055 ± 18%
S_{xz}	0.0088 ± 12%

S_{yx}	-0.0068 ± 40%
S_{yz}	0.008 ± 40%
S_{zx}	-0.0092 ± 11%
S_{zy}	-0.0040 ± 30%
Displacement Model (Polynomial Fitting)	
Parameter	**Optimized Value**
$q_{0,x}$	-0.0328 ± 1.2%
$q_{1,x}$	0.00880 ± 0.3%
$q_{2,x}$	-0.018965 ± -0.005%
$q_{0,y}$	-0.0480 ± 1.0%
$q_{1,y}$	0.01138 ± 0.3%
$q_{2,y}$	0.053362 ± 0.002%
$q_{0,z}$	-0.0366 ± 1.6%
$q_{1,z}$	0.02727 ± 0.17%
$q_{2,z}$	0.010863 ± 0.014%

References

[1] National Geodetic Survey. (2022, August 16). *Surface Gravity Prediction*. NGS Surface Gravity Prediction. Retrieved March 25, 2023, from https://www.ngs.noaa.gov/cgi-bin/grav_pdx.prl.

[2] W. Niu, F. Li-qing, Z. Qi, and D. Guo, "Small Displacement Measuring System Based on MEMS Accelerometer," *Mathematical Problems in Engineering*, vol. 2019, pp. 1–7, Oct. 2019.

[3] K. Borodacz, C. Szczepanski, and S. Popowski, "Review and selection of commercially available IMU for a short time inertial navigation," *Aircraft Engineering and Aerospace Technology*, vol. 94, no. 1, pp. 45–59, May 2021.

[4] D. Shaeffer, "MEMS Inertial Sensors: A Tutorial Overview," *IEEE Communications Magazine*, vol. 51, no. 4, pp. 100–109, Apr. 2013.

[5] G. Balamurugan, J. Valarmathi, and V. Naidu, "Survey on UAV navigation in GPS denied environments," presented at the 2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES), Jun. 2017.

[6] Q. Guo, W. Deng, O. Bebek, C. Cavusoglu, C. Mastrangelo, and D. Young, "Personal inertial navigation system employing MEMS wearable ground reaction sensor array and interface ASIC achieving a position accuracy of 5.5m over 3km walking distance without GPS," presented at the 2018 IEEE International Solid - State Circuits Conference - (ISSCC), Mar. 2018.

[7] A. R. Jimenez, F. Seco, C. Prieto, and J. Guevara, "A comparison of Pedestrian Dead-Reckoning algorithms using a low-cost MEMS IMU," presented at the 2009 IEEE International Symposium on Intelligent Signal Processing, Budapest, Hungary, Oct. 2009.

[8] L. Ojeda and J. Borenstein, "Personal Dead-reckoning System for GPS-denied Environments," presented at the 2007 IEEE International Workshop on Safety, Security and Rescue Robotics, Rome, Italy, Nov. 2007.

[9] M. Quigley, R. Brewer, S. Soundararaj, V. Pradeep, Q. Le, and A. Ng, "Low-cost accelerometers for robotic manipulator perception," presented at the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, Dec. 2010.

[10] S. Mohamed, M.-H. Haghbayan, T. Westerlund, J. Heikkonen, H. Tenhunen, and J. Plosila, "A Survey on Odometry for Autonomous Navigation Systems," *IEEE Access*, vol. 7, pp. 97466–97486, Jul. 2019.

[11] J. Taylor, *An Introduction to Error Analysis: The Study of Uncertainties in Physical Measurements*, 2nd ed. University Science Books, 1996.

[12] G. Pang and H. Liu, “Evaluation of a Low-cost MEMS Accelerometer for Distance Measurement,” *Journal of Intelligent and Robotic Systems*, vol. 30, pp. 249–265, Mar. 2001.

[13] Y. Thong, M. Woolfson, J. Crowe, B. Hayes-Gill, and D. Jones, “Numerical double integration of acceleration measurements in noise,” *Measurement*, vol. 36, pp. 73–92, 2004.

[14] M. Park and Y. Gao, “Error Analysis and Stochastic Modeling of Low-cost MEMS Accelerometer,” *Journal of Intelligent and Robotic Systems*, vol. 46, pp. 27–41, Jul. 2006.

[15] P. Aggarwal, Z. Syed, X. Niu, and N. El-Sheimy, “A Standard Testing and Calibration Procedure for Low Cost MEMS Inertial Sensors and Units,” *The Journal of Navigation*, vol. 61, no. 2, pp. 323–336, Apr. 2008.

[16] S. Shen, C. Chen, and H. Huang, “A New Calibration Method For Low Cost MEMS Inertial Sensor Module,” *Journal of Marine Science and Technology*, vol. 18, no. 6, pp. 819–824, 2010.

[17] Y. Yu, W. Lin, and Z. Yu, “Research on motion track error detection and compensation algorithm based on MEMS sensor,” presented at the 2022 2nd International Conference on Computer, Communication, Control, Automation and Robotics (CCCAR2022), Jun. 2022, vol. 47.

[18] Q. Irfan, M. Ciarcia, and G. Hatfield, “Inertia Measurement Unit-Based Displacement Estimation via Velocity Drift Compensation Using Ordinary Least Squares Method,” presented at the 2022 IEEE International Conference on Electro Information Technology (eIT), Jul. 2022.

[19] I. Frosio, F. Pedersini, and N. A. Borghese, “Autocalibration of MEMS Accelerometers,” *IEEE Transactions on Instrumentation and Measurement*, vol. 58, no. 6, pp. 2034–2041, Oct. 2008.

[20] M. Sipos, P. Paces, J. Rohac, and P. Novacek, “Analyses of Triaxial Accelerometer Calibration Algorithms,” *IEEE Sensors Journal*, vol. 12, no. 5, pp. 1157–1165, Sep. 2011.

[21] O. Särkkä, T. Nieminen, S. Suuriniemi, and L. Kettunen, “A Multi-Position Calibration Method for Consumer-Grade Accelerometers, Gyroscopes, and Magnetometers to Field Conditions,” *IEEE Sensors Journal*, vol. 17, no. 11, Apr. 2017.

[22] Q. Kong, T. Siauw, and B. Alexandre, *Python Programming and Numerical Methods*, 1st ed. Academic Press, 2020.

[23] HiLetgo, “MPU9250/6500 9-axis 9 DOF 16 bit gyroscope acceleration magnetic sensor IIC/SPI,” *HiLetgo*. [Online]. Available: http://www.hiletgo.com/ProductDetail/1953399.html. [Accessed: 01-Mar-2023].

[24] Amazon.com, “Hiletgo MPU9250 GY-9250 9-axis 9 dof 16 bit gyroscope acceleration ...,” *Amazon.com*. [Online]. Available: https://www.amazon.com/HiLetgo-Gyroscope-Acceleration-Accelerator-Magnetometer/dp/B01I1J0Z7Y. [Accessed: 01-Mar-2023].

[25] InvenSense, “PS-MPU-9250A-01.” InvenSense Inc., San Jose, CA, 20-Jun-2016. Available: https://invensense.tdk.com/wp-content/uploads/2015/02/PS-MPU-9250A-01-v1.1.pdf. [Accessed: 01-Mar-2023].

[26] J. Hrisko, “Calibration of an Inertial Measurement Unit (IMU) with Raspberry Pi - Part I,” *Maker Portal*, 21-Jan-2022. [Online]. Available: https://makersportal.com/blog/calibration-of-an-inertial-measurement-unit-with-raspberry-pi. [Accessed: 01-Mar-2023].

[27] A. Zhuravlev, “Gy-9250 (MPU-9250) calibration cube by Cynepkot,” *Thingiverse*, 28-May-2019. [Online]. Available: https://www.thingiverse.com/thing:3659438. [Accessed: 01-Mar-2023].

[28] J. Hrisko, “mpu9250_i2c.py.” Maker Portal, San Francisco, CA, 28-Dec-2020. Available: https://github.com/makerportal/mpu92-calibration/blob/main/mpu9250_i2c.py. [Accessed: 01-Mar-2023].

[29] J. Filliben and A. Heckert, “1.3.5.6. measures of scale,” *e-Handbook of Statistical Methods*, Apr-2012. [Online]. Available: https://www.itl.nist.gov/div898/handbook/eda/section3/eda356.htm#VARIANCE. [Accessed: 01-Mar-2023].

[30] SciPy, “scipy.optimize.curve_fit,” *scipy.optimize.curve_fit - SciPy v1.10.1 Manual*. [Online]. Available: https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.curve_fit.html. [Accessed: 01-Mar-2023].

[31] J. Prins, “6.5.4.1. Mean Vector and Covariance Matrix,” *e-Handbook of Statistical Methods*, Apr-2012. [Online]. Available: https://www.itl.nist.gov/div898/handbook/pmc/section5/pmc541.htm. [Accessed: 01-Mar-2023].