ECE167 Sensing and Sensor Technology

Lab3

James Huang

Collaborator: AI

Checkoff TA: various time/various TA

Commit: a49015073568657ec8a26d5420317f6757b8e2ff

Table of contents:

1. Project Introduction

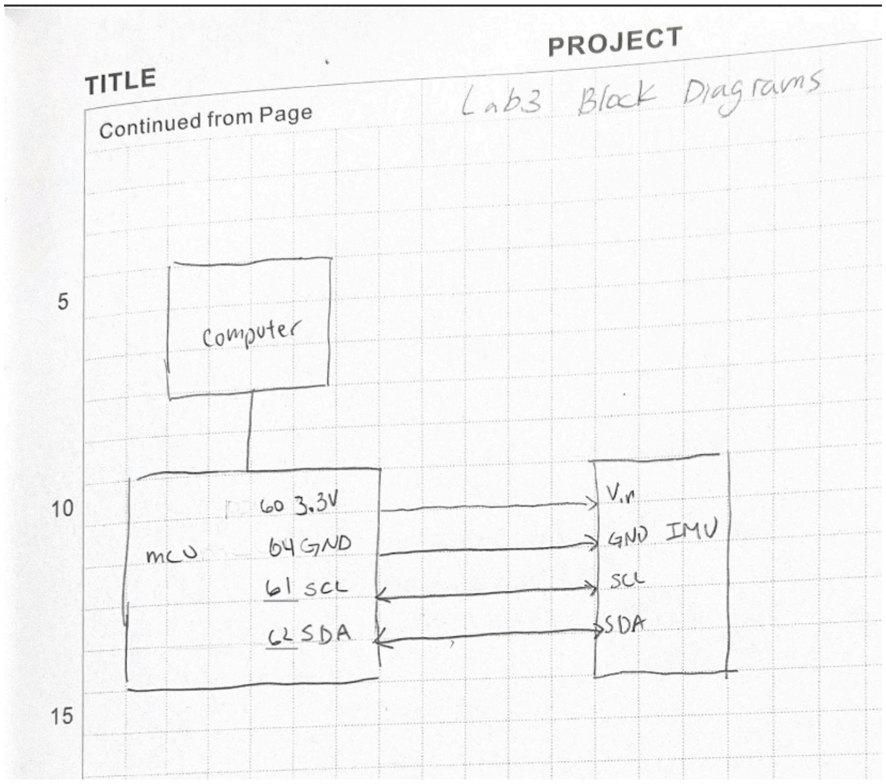
This lab focused on calibration of the IMU, the accelerometer, magnetometer, and the gyroscope. The first part of the lab introduces a simple 2D point cloud calibration generated by the MATLAB code given. Another set of code will be able to do the same but for a 3D point cloud.

We begin by calibrating the accelerometer, which measures acceleration along three axes. The raw accelerometer readings are collected in multiple orientations to estimate biases and scale factors. These corrections are then applied to generate calibrated data. A point cloud visualization in MATLAB is used to compare raw vs. calibrated data, where an ideal accelerometer should produce a sphere with a radius of 1g.

Next, the magnetometer is calibrated to have the X and Y axis as the horizontal density, or 23,233.9nT, when pointed North. The Z axis is to be calibrated to vertical comp, or 41,237.2nT, when facing up. This would result in an egg-shaped point cloud with the calibrated data, as the scale for Z is much higher.

Finally, the gyroscope, which measures angular velocity, is calibrated to read degrees as the IMU is being turned. The sensor is first measured for drift or bias by collecting raw data for 10 minutes, and averaging. Then the scale factor is measured by seeing the raw sensor value as it is being turned.

2. System Block Diagram



3. Hardware Components

The Bosch BNO055 is a 9-axis absolute orientation sensor that integrates an accelerometer, magnetometer, and gyroscope. The BNO055 supports communication via I^{2}C and UART, providing flexibility for different applications. The I^{2}C setup, including initialization and data retrieval, is handled by the provided firmware, simplifying the process. The given firmware only reads the raw values for each part of the IMU. The sensor operates with a logic level of 1.8V to 3.3V and a power supply range of 2.4V to 3.6V. In the lab, 3.3V power is used, and SCL and SDA lines are connected to the microcontroller's corresponding pins.

4. Software Components

The given MATLAB code, EllipseXYData.m, in the repo is able to produce random 2D data, which needs to be calibrated and corrected by the two given code, CalibrateEllipseData2D.m and CorrectEllipseData2D.m. A simple MATLAB code is needed to take random data and feed it to the two functions to be calibrated and plotted. The same thing can be done for 3D data, using CreateTumbleData.m, and again feed through the respective 3D calibration code.

The firmware of the IMU is very simple, there is a function to read raw sensor data for each axis, x, y, z, for accelerometer, magnetometer, and gyroscope. In order to calibrate the accelerometer to read ±1g for all axes, we need to find the bias and a scale factor for each axis. To achieve this, a 2 point calibration method is used, where average raw sensor data readings at two points (facing up/down, right/left) is collected and used in the following formula to find scale factor and bias.

$$scale\ factor = \frac{raw_{down} - raw_{up}}{-2},\ bias = \frac{raw_{up} + raw_{down}}{2}$$

The same formula is applied for all axes, but instead of facing up or down it is right or left, forward or backward respectively, these raw values can be stored using #define at the top, for easy re-calibration. Then to find the final calibrated data, the following formula is used.

$$calibrated\ data = (live\ raw\ sensor\ data - bias)\ /\ scale\ factor$$

Next, in order to calibrate the magnetometer similar method is followed, using the 2 point calibration. However the formulas are slightly different, we need to first set the expected values we want our sensor to read, so 23,233.9nT, for X and Y axis when its pointed north, and 41,237.2nT for when Z axis, and negative if point 180 degree the other way. The following formula is used.

$$scale\ factor = \frac{expected_{down} - expected_{up}}{raw_{down} - raw_{up}},\ bias = expected_{up} - (scale\ factor * raw_{up})$$

$$calibrated\ data = (live\ raw\ sensor\ data * scale\ factor) + bias$$

Again, these variables refer to the Z axis, facing up and down, but the same formula applies for the X and Y axis.

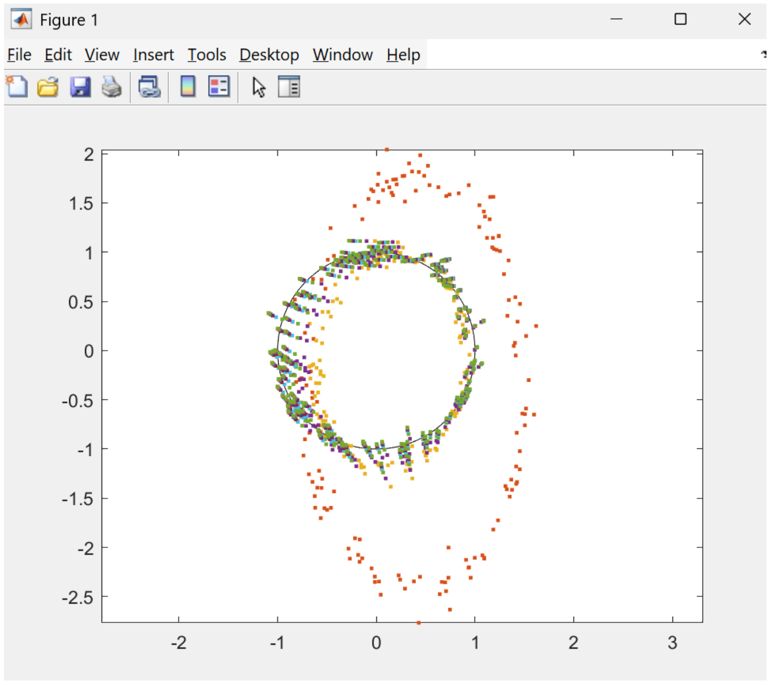
Lastly for the gyroscope, a 20ms delay is required to sample at the rate of 50Hz, and the angles can be found by first finding the bias. This is done by running the sensor for 10 minutes without movement, and seeing the drift and averaging all the data points. Then the scale factor can be found by rotating the gyroscope 180 degree in one axis, and recording the raw sensor data spike. This data can then be integrated in MATLAB using the cumtrapz() function to see a clear spike and max value when rotating the sensor. The max value will be the scale factor. With both the bias and scale factor, the following formula is used to convert the raw sensor readings to degrees.

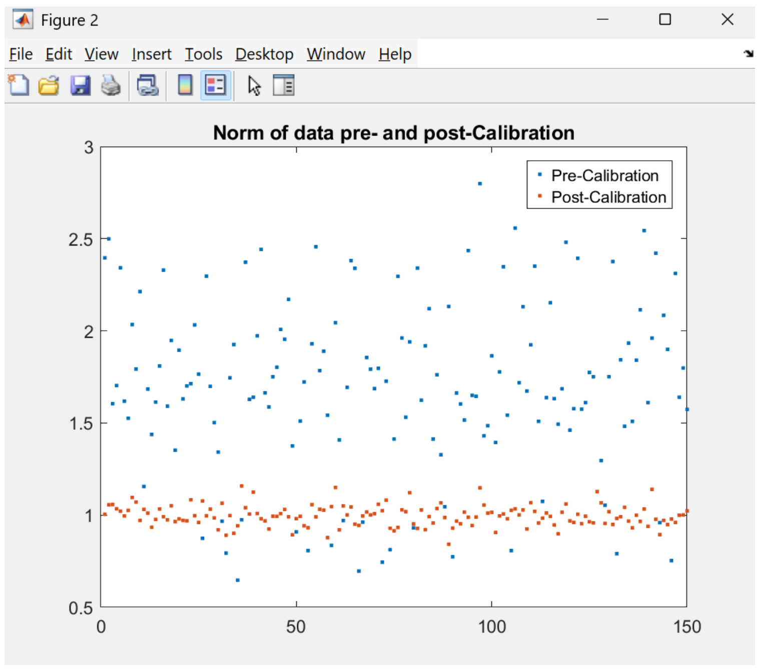
$$degree\ += ((live\ raw\ data - bias)/scale\ factor) * dt * 180$$

The dt corresponds to our sample rate of 50Hz, or 20ms, so in this case would be 0.02, and the 180 corresponds to the 180 degree we turned to to find the scale factor. If you chose to rotate 90 degrees, it would be 90 instead. All of the data collection that is done in this lab is done using CoolTerm, a serial terminal that can generate .csv files with incoming data. This .csv can then be used in MATLAB for data calibration and plotting, these files can be found in the Lab3 folder under matlab in the repo.

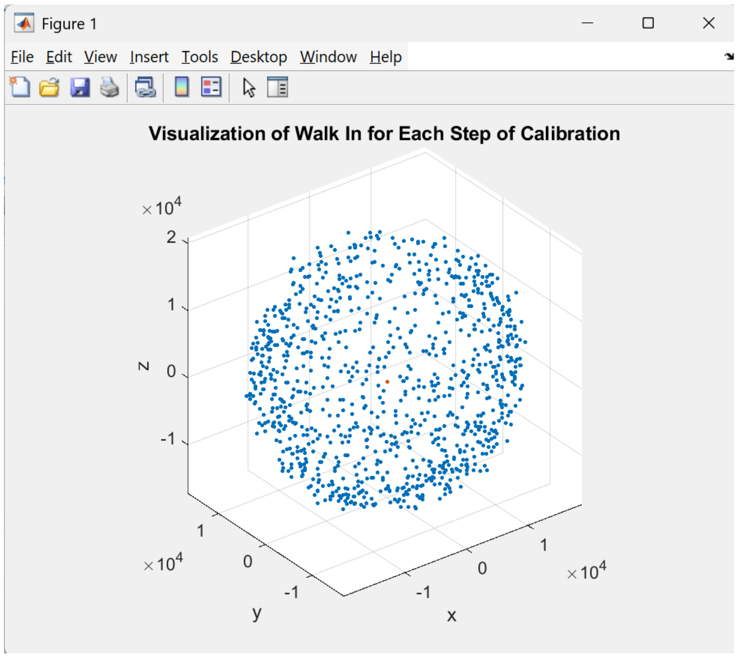
5. Testing and Result

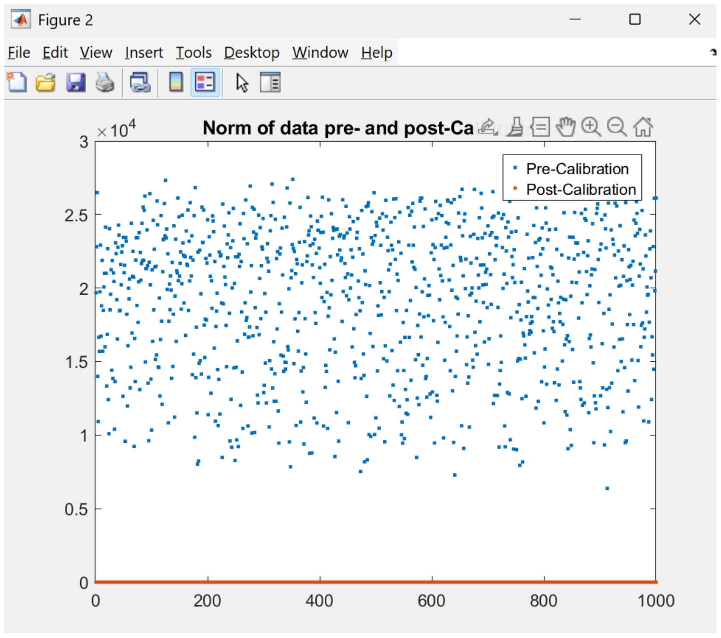
The first set MATLAB 2D calibration with given random data generated the following plots

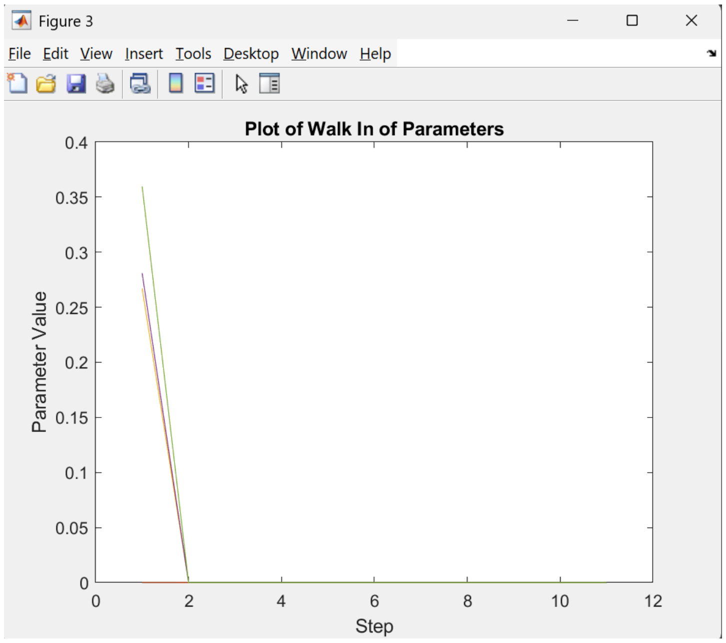


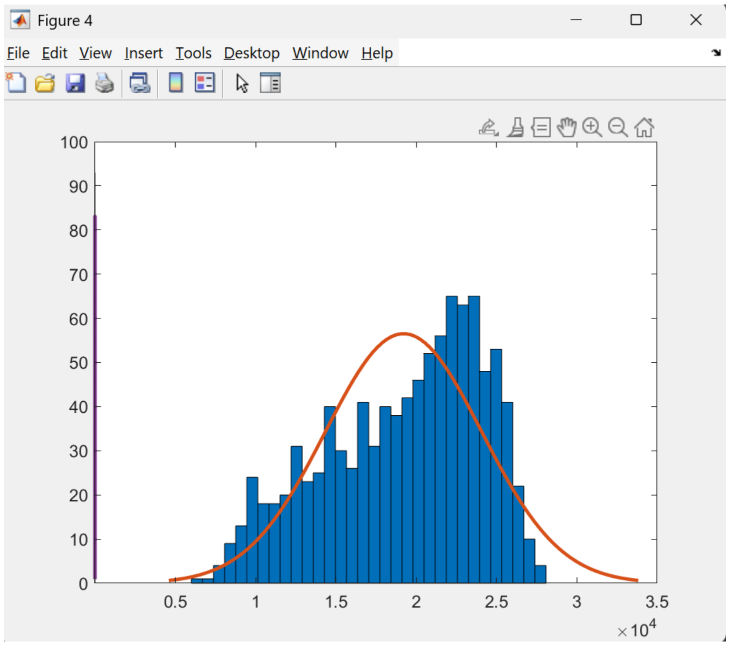


As for the 3D plots with given data and functions, these are generated.



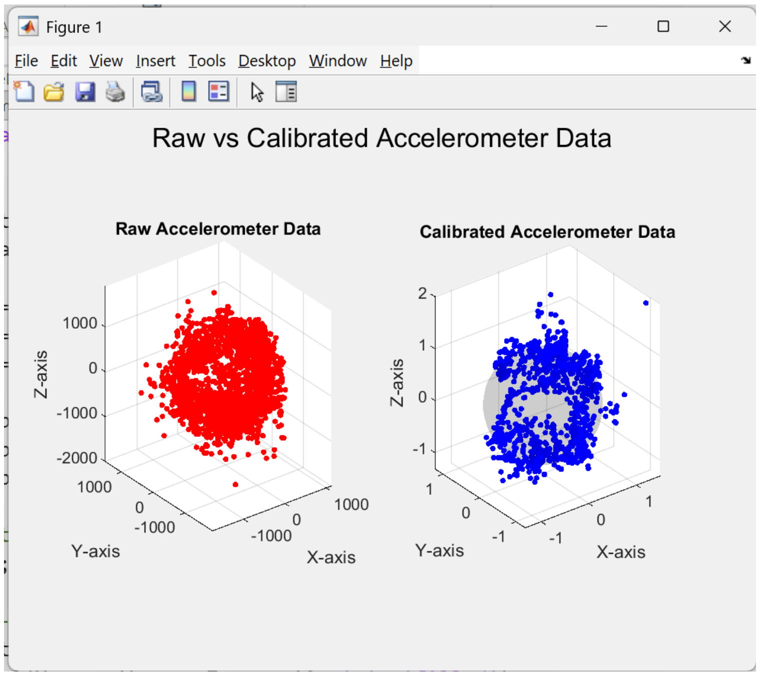




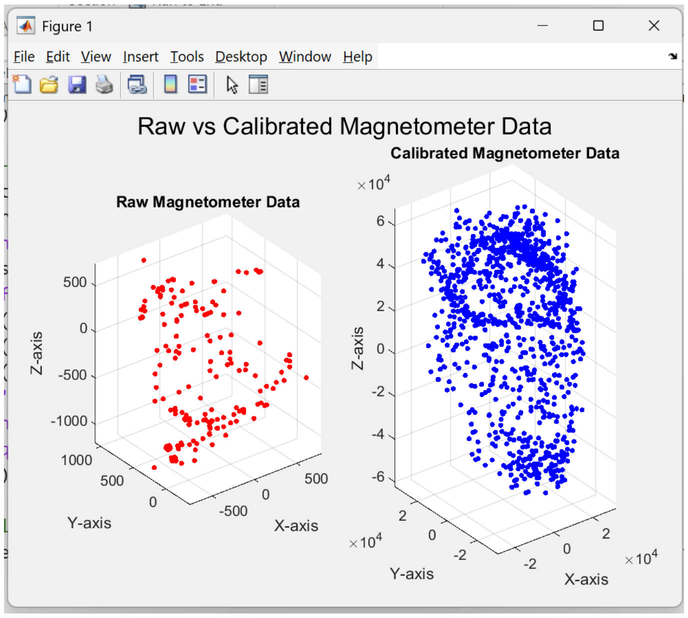


As seen from the plots, the red calibrated data is focused at the center of the plot.

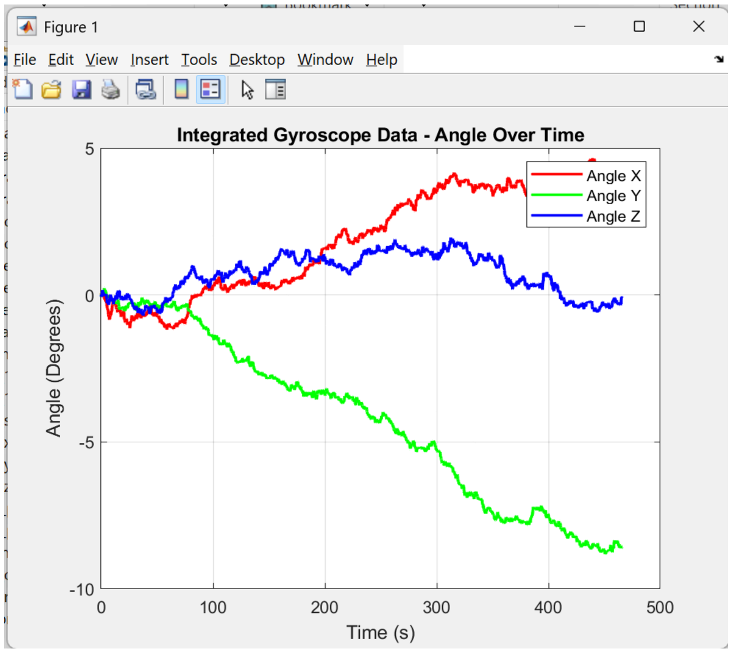
In order to calculate the accelerometer, the 2 points must be taken and saved for calculations, by flipping the IMU in different directions facing different axes. I collected raw sensor reading from my C code, as well as calibrated data, with two sets of data. I plotted the following to show the difference in MATLAB. The raw data sits at a scale of 1000, while the calibrated data sits at scale of 1, which matches the 1g readings we calibrated to. The scale factor and bias came out to be x: -998.00, -8.00 y: -995.00, -75.00 z: 998.00, -18.00 for each axis respectively.



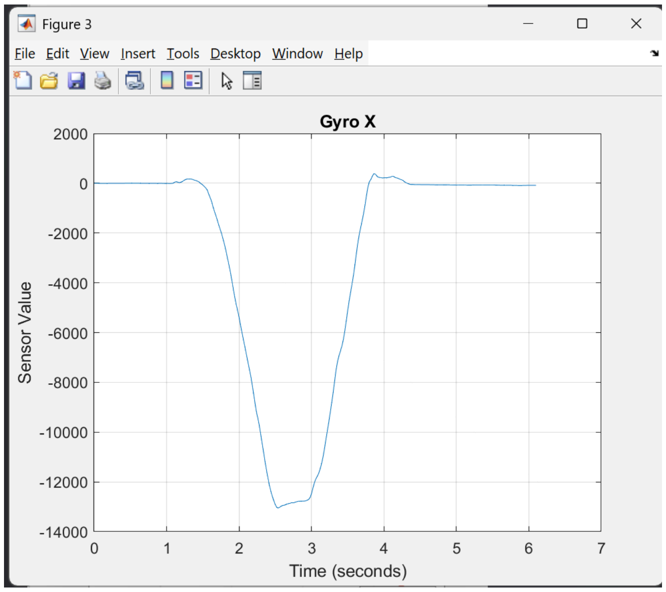
The same process is done for the magnetometer, raw and calibrated data is collected from my C code, although the magnetometer is much more sensitive and harder to calibrate. Any metal objects in the vicinity will affect the reading, it is best to calibrate the sensor in an open area with consistent objects nearby. Again, the calibrated data will look more like an egg shape due to the Z axis being set to a much higher number, almost double what the X and Y axis is being set to. The scale factor and bias came out to be The scale factor and bias came out to be x: -35.39, 5078.54 y: -41.86, 14861.32 z: -73.64, -7363.78 for each axis respectively. However, these numbers depend on the 2 point calibration raw sensor readings based on the formula provided, so they change every time you calibrate. For me these are the numbers I got based on my 2 point sensor readings, which can be found at the top of my C code in #defines, same can be said for the accelerometer.



Lastly, for the gyroscope, I took raw sensor data reading, converted to degree in MATLAB to see the degree drift over time, which can be seen in a large data set, in the 10 minutes one. The average drift I got for each axis for the 10 minutes data set is Average X: -13.14 Average Y: -19.25 Average Z: 5.68, these are raw sensor data, not degree.



Then the scale factor calibration, by taking raw sensor data as we rotate, we can capture the spike in angular velocity, the data is then plotted in MATLAB. These values can either be positive or negative, depending on which way I have rotated the sensor, but in the end it doesn't really matter. The gyroscope can be difficult to calibrate, as the longer it has been on, the more drift it has, so it is a good idea to power off and restart the sensor to calibrate on the spot.



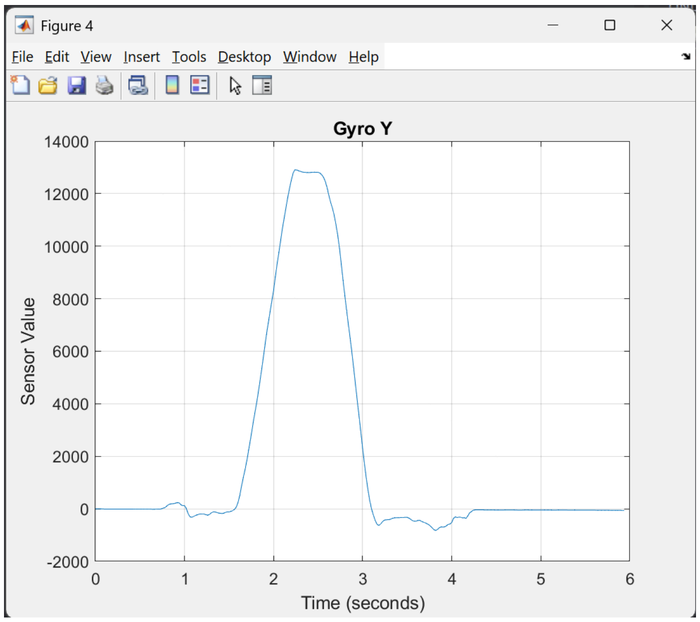


Figure 4
File Edit View Insert Tools Desktop Window Help
Gyro Y
Sensor Value
Time (seconds)
14000
12000
10000
8000
6000
4000
2000
0
-2000
0
1
2
3
4
5
6

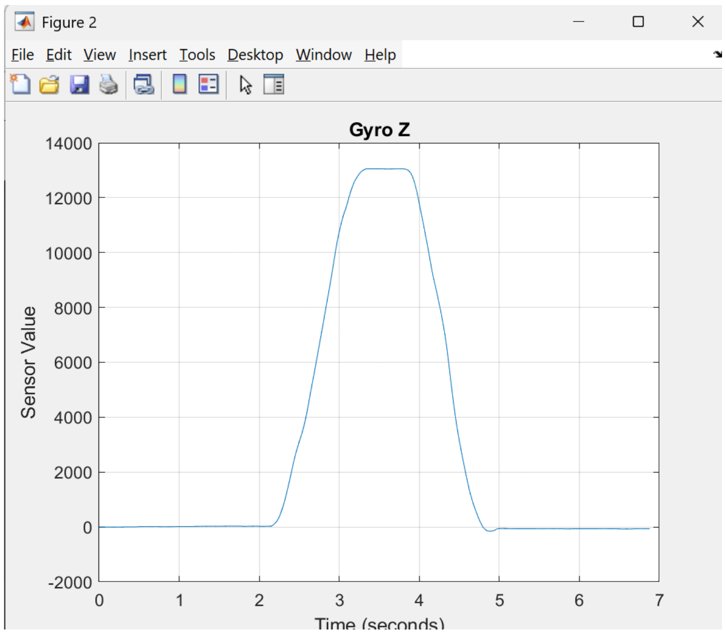


Figure 2
File Edit View Insert Tools Desktop Window Help
Gyro Z
Sensor Value
Time (seconds)
14000
12000
10000
8000
6000
4000
2000
0
-2000
0
1
2
3
4
5
6
7

6. conclusion

In conclusion, this lab is most difficult in figuring out how to calculate bias and scale factor for the accelerometer, magnetometer, and gyroscope. It is also incredibly difficult to collect good data, as our cheap sensor is noisy, not very accurate, and can shift a lot between different calibrations, which can mess up our calculated output. The lab document is TERRIBLE for this lab. The procedures are confusing and useless, I would even say the MATLAB code is useless, the only thing I see about the MATLAB code is to see what a point cloud looks like, and what raw vs calibrated data look like. The lab documentation seems out of order, part 4 is the tumble test which relates to checkoff for accelerometer and magnetometer in part 2, which should just be in part 2. I would restructure the lab document to be part 1: simulated data using given MATLAB code for both 2D and 3D (to see what point cloud is supposed to look like). Then in part 2: accelerometer and magnetometer calibration and using our own raw and calibrated data to generate a point cloud, using our own MATLAB code. Lastly, part 3: gyroscope calibration and sensor drift graph. The procedure in the lab doc is also very confusing, for example the way to calibrate the magnetometer is completely useless. It should be clear, the Z axis can be calibrated by having the sensor facing up, and X and Y axes can be calibrated facing North.