ECE 445

SENIOR DESIGN LABORATORY

FINAL REPORT

Gesture-Based Turn Signaling System

Team #20

EDAN ELAZAR
(eelazar@illinois.edu)
KAYLAN WANG
(kaylanw4@illinois.edu)
SULTAN ALNUAIMI
(saltana2@illinois.edu)

TA: Sanjana Pingali

May 1, 2024

Abstract

This report details the design, development, and verification of a gesture-based turn signaling system intended to enhance the visibility and safety of cyclists, skateboarders, and scooter riders. Utilizing inertial measurement units (IMUs) integrated into a jacket, the system processes arm gestures to activate corresponding LED signals for turns and stops. Our development focused on the design of four main subsystems: sensors, control, power, and LEDs. Through testing, the device demonstrated over 90% accuracy in gesture recognition, effectively translating these gestures into visible LED signals.

Contents

1 Introduction

1.1 Problem Statement

Cyclists, skateboarders, and scooter riders often face challenges in signaling their intentions to drivers, especially in low-light conditions. According to the CDC, 1,000 cyclists die and 130,000 are injured every year on the road in the United States [1]. These numbers don't include other riders sharing the road on things like skateboards and scooters. There are many interventions in place to prevent these accidents, such as fluorescent or retro-reflective clothing, or active lighting on the bicycle (required by law in most states) [1], but the traditional method of using hand signals is not always visible or practical, particularly at night or during adverse weather conditions. This lack of clear communication can lead to dangerous situations on the road, as other motorists may fail to recognize the cyclist's intended maneuvers, or if an accident occurs.

1.2 Proposed Solution

To address this issue, we propose the development of a gesture recognition-based turn signaling system for cyclists and scooter riders. This system will utilize IMUs containing 9 degrees of freedom (3-axis accelerometer, 3-axis gyroscope, 3-axis magnetometer), integrated into a jacket. Then the data from the sensors will be processed to identify the specific arm gesture made by the rider and activate corresponding LED signals. For example, if the rider extends their arm straight to the left, the left turn signal is activated, or if the rider indicates a stop (arm out and forearm down), then the brake light is activated, and so on. Additionally, the sensors will be able to detect when the rider has had an accident or a crash, and activate a hazard signal on the LEDs.

We propose placing an IMU below the on the left wrist, and another on the waist. The microprocessor will then receive and process the data from the IMU, and determine what kind of movement has been made in real time. Then, depending on the movement, it will output a specific signal to the LEDs to display on the front and back of the wearable.

The final product achieved all 3 high level requirements. The device is able to correctly detect predefined arm gestures (raising right/left arm for turn signals, forearm down for slowing down) with an accuracy of 90%. In addition, the device is able to correctly map the arm gestures into the different indications on the LEDs. The turn signal will be indicated by either the left or right side LED flashing red, while the brake/slow down signal will be indicated by all LEDs turning red. A crash or accident activates the hazard light, indicated by all LEDs flashing red. Lastly, the turn signals, brake lights, and hazard signals were all visible and easily identifiable from a distance of at least 250 feet to ensure that they are clearly visible at both day and night. All of these functionalities ensure that hand gestures are more visible to drivers and other people, thereby increasing the safety of the rider.

1.3 Visual Aid

Figure 1 is a mockup of the wearable device that we created where the dotted lines indicate the connections are inside the jacket. The device will be powered by a battery and will have 2 IMUs to detect the arm gestures. The system will be controlled by an ESP-32 microcontroller that will process the data from the IMUs and send a signal to the LEDs to turn on when the arm gestures are detected.

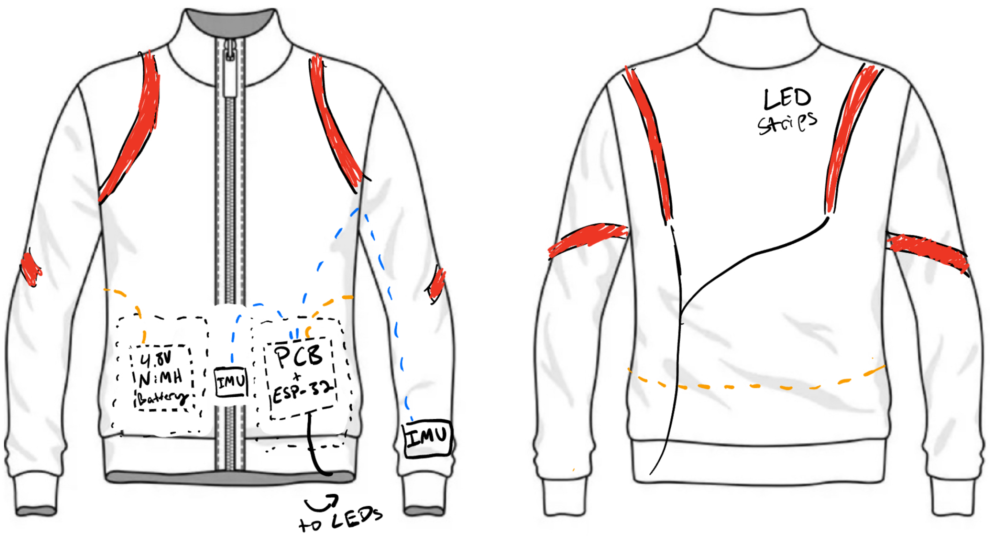


Figure 1: Visual Aid mockup of the wearable [2]

1.4 Block Diagram

As can be seen from the block diagram in figure 2, the device is split into 4 subsystems: Sensors, Control, Power, and LEDs. The power subsystem provides 4.8V to the LEDs through a BJT that gets a signal from the ESP-32. The Control Subsystem and the Sensor Subsystem receive 3.3 volts from the battery through a voltage regulator. The Control Subsystem receives real-time data from the 2 IMUs through I2C, processes the data, then sends a signal to the gate to turn the LEDs on when the IMU data is within a certain range.

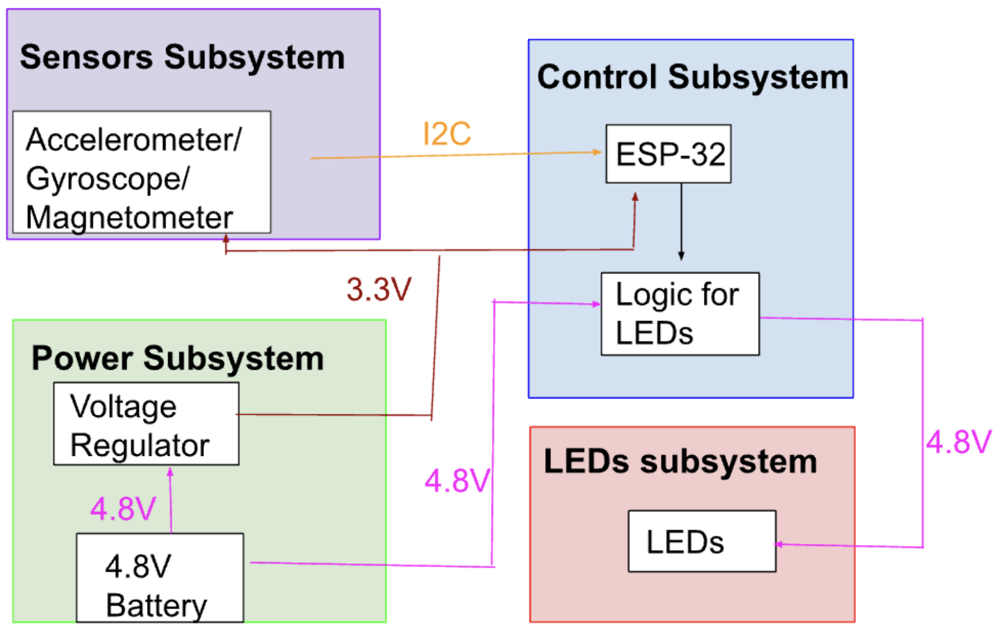


Figure 2: Block Diagram of the system

2 Design

2.1 Design Procedure

Our design consists of four different subsystems, each with their own functions and requirements. At the center of the system is the control subsystem consisting of the ESP-32 microcontroller along with some simple logic for controlling the LEDs. The goal of this subsystem is to establish communication with the sensors, receive the necessary data from them, determine which hand gesture is being activated, and activate the corresponding LEDs. The main design decisions that we had to make for the control subsystem were software related as the circuit is quite simple. We decided to simplify the algorithm for detecting the gestures due to some time constraints with the PCB order waves along with a problematic ESP-32 DevKit. We originally wanted to devise an algorithm that utilized all three sensors on the IMU, but ultimately decided that it would be too difficult to program, and we found that the magnetometer was all that was necessary to accurately detect the gestures.

The original plan was to use an IMU on each arm, but this approach proved to be difficult in terms of programming. Instead, we used one on the waist, which ensures that one of the IMUs is always stable. Then, we just take the difference in angles between the two IMUs, and set ranges for each hand gesture. We also had to discard our PCBs for our IMUs as we were unsuccessful in soldering the small IMUs onto the board. Therefore we decided to continue with the breakout boards.

For the power subsystem, we decided that the system should last at least one hour. We determined the current required to power the system to be 1864.2A total, so we needed a battery that could supply 5V with a capacity larger than 1864mAh, in addition to being rechargeable. The battery we used for the power subsystem was a 3000mAh 4.8V NiMH battery that provided ample voltage and current to power the LEDs. During the Design review, using an LED Driver was recommended to get a constant current to the LEDs. We researched several drivers and found one that fits the requirements. However, we also tested the LEDs without the driver, and the brightness was sufficient. The plan was to add the driver to get slightly brighter LEDs, but the driver only works with a PCB, and we were unable to get the LED drivers to function from the first and third order PCBs. Ultimately we decided to discard the LED driver in order to simplify the PCB and increase the odds of it functioning properly. We decided that a working PCB was more important than a slightly brighter LED.

2.2 Design Details

The LM317 datasheet [3] provides us with equation (1) which relates the output voltage to the resistances of R_1 and R_2 when using the schematic in figure 3

$$V_0 = V_{REF}\left(1 + \frac{R_2}{R_1}\right) + I_{ADJ} \times R_2 \quad [3] \tag{1}$$

Where R_1 and R_2 control the output voltage of the linear voltage regulator, and V_{REF} and I_{ADJ} are given to be 1.25V and 50μ A respectively. Using this equation, we were able to find the resistor values necessary to achieve an output of 3.3V, which are $R_1 = 314.2\Omega$ and $R_2 = 512\Omega$. Note that we also have a linear voltage regulator which outputs 1.8V, which we added with the intent of making our own IMU PCB, but we ended up using a breakout board which takes in 3.3V.

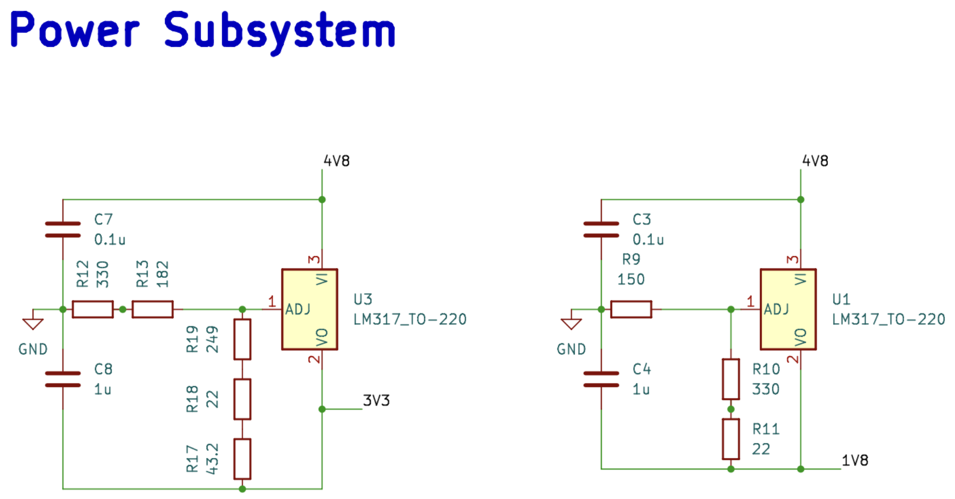


Figure 3: Schematic of the power subsystem

The control unit in figure 4 consists of the ESP32-S3 [4] which we use to control our entire system. We needed to add a programming circuit for it and ensure it has the proper connections to all the other subsystems.

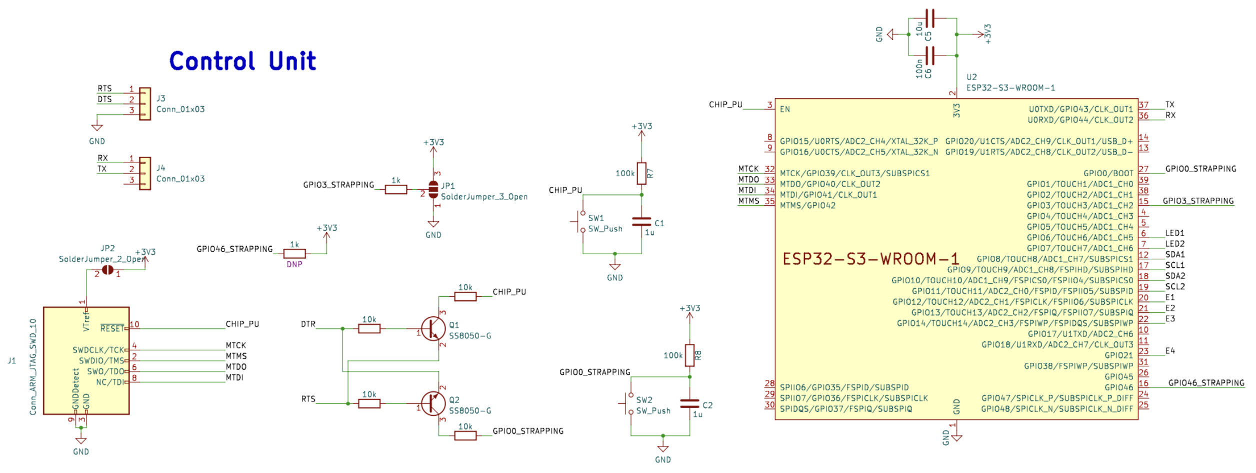


Figure 4: Schematic of the control subsystem

Our LED subsystem in figure 5 was intended to use an LED driver, so we designed a circuit for it which we have on our pcb. Due to time constraints however we did not have enough time to test it and ended up hotwiring the LEDs to bypass the LED driver.

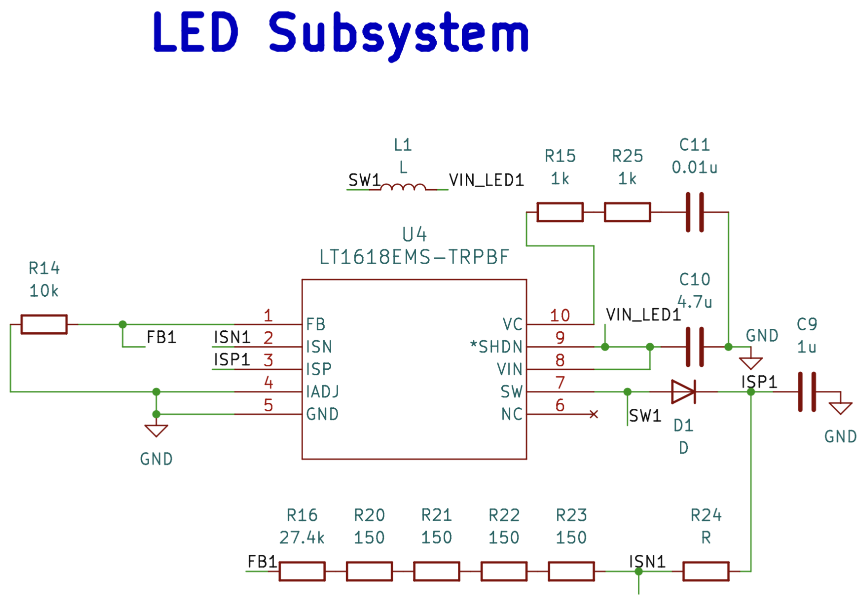


Figure 5: Schematic of the LED subsystem

The sensor subsystem in figure 6 consists of the ICM-20948 IMU which we use to detect the hand gestures. This schematic shows the suggested connections [5] for the IMU to communicate via I2C with the ESP32. Unfortunately, we were unable to get the IMU to work with the ESP32, so we had to use the breakout board which has a built-in voltage regulator and I2C communication.

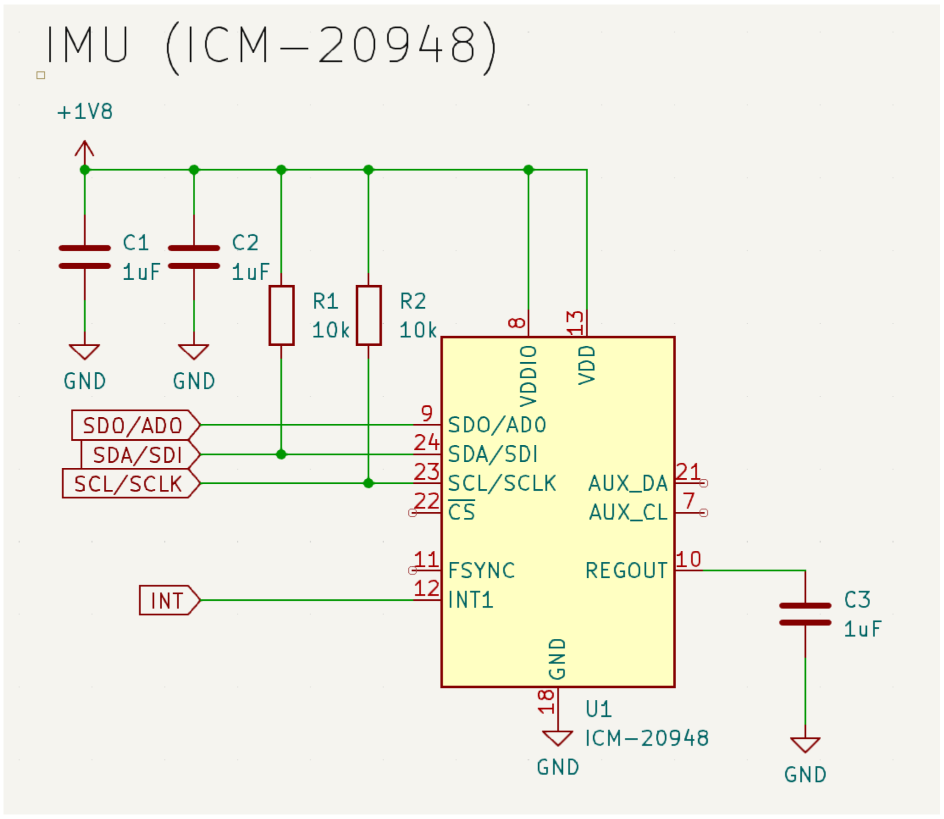


Figure 6: Schematic of the sensor subsystem

In order to power the system, we used a 3000mAh 4.8V NiMH battery from Panasonic [6]. As shown in table 1, our total current draw is 1864.2 mA so our estimated battery life of the system is $3000\text{ mAh}/1864.2\text{ mA} = 1.61$ hours.

Table 1: Current Draw of System Components

Description	**Value**
ESP-32 worst case current draw	355 mA
2x ICM-20948 IMU	4.6 mA each = 9.2 mA total
2x LED strip	750 mA each = 1500 mA total
Total current draw	1864.2 mA

Heat tolerance for battery

- Total Internal Impedance = $4\,\text{m}\Omega \times 4 = 16\,\text{m}\Omega$.
- $I = 2\,\text{A}$
- Time = 3600 seconds
- Mass = 0.228 kg for 4 cells
- Specific Heat Capacity: $900\,\text{J/kg}^{\circ}\text{C}$
- Total Heat Generated = $I^2 \times R \times \text{time} = 230.4\,\text{J}$
- Change in temp = $\frac{\text{Heat}}{\text{mass}\times\text{specific heat capacity}} = 1.123^{\circ}\text{C}$

Assuming an ambient temperature of 25°C, the temperature of the battery will be around 26.123°C after one hour of operation. This is well within the operating temperature of the battery, so we can conclude that the battery will not overheat during operation.

During testing we found that after powering the system for one hour in an ambient temperature of 22°C, the battery was 23.5°C. This confirms our calculations that the battery will not overheat during operation.

3 Verification

3.1 Control Subsystem

Our control subsystem is used to run computations on input data from the sensor subsystem and output it through the LED subsystem. Because of this our requirements reflect how it interacts with these subsystems.

The first requirement is that it must be able to communicate with LEDs and Sensors through the PCB. We were able to verify this by connecting them to the control unit and seeing that the LEDs turn on when given a signal from the control unit, and that it is able to read data from the IMUs.

Our second requirement is that it must be able to determine the correct turn signals from the IMUs. This was done in software and was easier to verify once the entire system was complete. We verified it by wearing the device and making each gesture 10 times, where we got 92.5% accuracy.

Our third requirement is that it must be able to turn on the LEDs based on the corresponding turn signal. We did this by connecting the LEDs to the Control Unit and seeing that they turn on when given a turn signal.

3.2 Power Subsystem

Our Power subsystem had three necessary requirements in order for it to work properly with the rest of the system. The requirements along with how we verified them are listed below.

The first requirement is that it must supply 3.3±0.3V to the IMUs and ESP32. We were able to verify this by using a multimeter to measure the voltage differential between the output of the linear voltage regulator and ground. We placed the red probe on the output pin of the linear voltage regulator and the black probe on the ground of the battery and read that the voltage differential is about 3.36V, meaning that anything connected between the two would receive about 3.36V.

The second requirement is that it must supply 4.8±0.3V to the LEDs. We were able to verify this by using a multimeter to measure the voltage differential across the LEDs. We placed the red probe on the emitter pin of the BJT and the black probe on the ground of the battery. When supplying power from the ESP32 through the base of the BJT, we were able to measure about 5.03V, and the LED between them would turn on. We were able to lower the voltage to 4.8V, but ended up using 5V from the battery because the LEDs are technically rated for 4.8V, not 5V.

The final requirement is that the temperature of the battery should stay below 50°C. We were able to verify this by leaving the entire system running for about half an hour and measuring the temperature of the battery with an infrared thermometer. We read that the temperature remained at 23.5°C which is way below our threshold of 50°C.

3.3 Sensor Subsystem

Our sensor subsystem consists of two 9 degree-of-freedom IMUs, meaning each one has an accelerometer, gyroscope, and magnetometer. For each IMU, we must verify that the sensor readings are accurate for each of the sensors that we use for gesture recognition.

The first requirement is that the accelerometer must be able to detect acceleration with an accuracy of ±1m/s^2. In order to verify this we connected each IMU to the ESP32 through the I2C protocol and measured the acceleration due to gravity for each of the three degrees of freedom (x, y, z directions). Then we can check the output on the serial monitor to make sure that the acceleration due to gravity is 9.8±1m/s^2 in the current direction parallel to gravity, and the acceleration readings in the other two directions are 0±1m/s^2. After conducting this verification, we found that the accelerometers were able to detect the acceleration in all three cartesian directions within 1m/s^2.

The second requirement is that the magnetometer must be able to detect true north accurately. In order to verify this, we first connect the IMU to the ESP32 through the I2C protocol. Then we measured true north with the compass app on our smartphones and compared it to the direction of the IMU when it read zero degrees on the serial monitor. We did multiple trials for each of the IMUs and found that the magnetometers always detected north correctly within ±10°. More importantly than being able to detect north, is the measured angles of the two magnetometers need to be accurate relative to each other. In other words, the difference between the magnetometer readings of both IMUs when facing the same direction should be negligible. We verified this by leveling both IMUs so that they faced the same direction and then printed out the difference in magnetometer readings to the serial monitor, and found that the difference was always less than 5°.

There is no verification for the gyroscope as we did not utilize this sensor for the gesture recognition algorithm.

3.4 LED Subsystem

The LEDs must be able to turn on in response to the other subsystems as well as be visible by a distance.

Our first requirement is that the LEDs must be able to turn on and off given a 4.8V input. We verified this by connecting our LEDs directly to our 4.8V battery and verifying that the LEDs turned on.

Our second requirement is that the LEDs must be visible from at least 250 ft away. We verified this by having one person walk 250 ft away with the LEDs while the other 2 filmed that the LEDs are visible.

4 Costs & Schedule

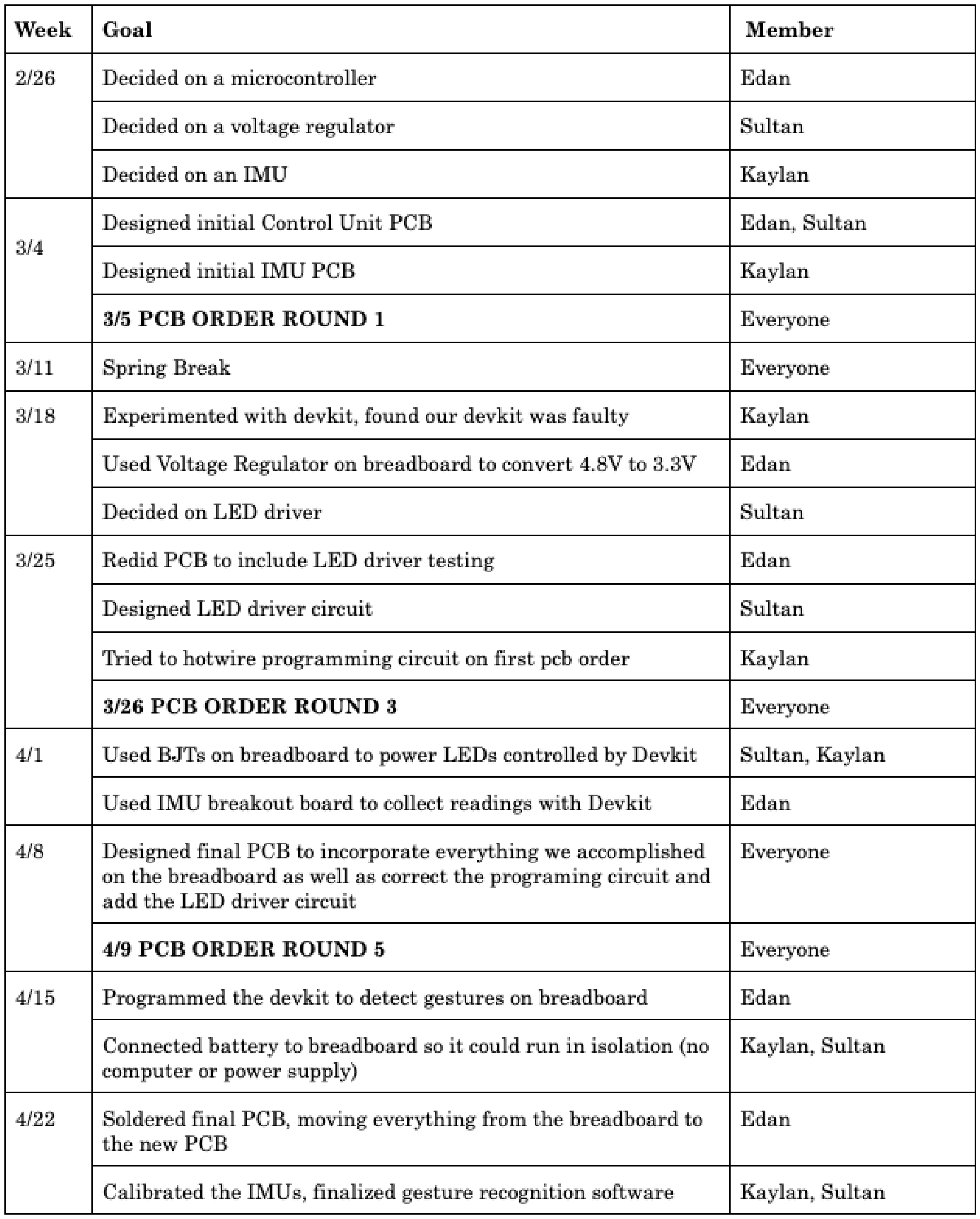
4.1 Costs

The total cost of parts will be approximately $\$234.96$ and the expected labor costs are calculated as $\$40$/hrs $* 2.5 * 60$hrs $= \$6,000$. This will be applied to all 3 team members so the total labor cost is $\$6,000 * 3 = \$18,000$. This comes out to a total cost of $\$18,234.96$.

Table 2: Bill of Materials

Description	**Manufacturer**	**Quantity**	**Price/unit**	**Total Price**	**Link**
velcro strips	VELCRO Brand	1	19.62	19.62	Link
ESP32	Espressif Systems	1	7.69	7.69	Link
LED strips	Aclorol	2	5.99	11.98	Link
Battery	Panasonic - BSG	1	40.50	40.50	Link
Voltage Regulator	STMicroElectronics	1	6.95	6.95	Link
IMU breakout board	Adafruit	3	18.50	55.50	Link
0.065 Ohm resistors	TFT Corp	4	0.55	2.20	Link
LED Driver	Analog Devices Inc	3	5.70	17.10	Link
1A Diode	onsemi	3	0.50	1.50	Link
Inductors	SAC Inc	3	0.82	2.46	Link
ICM-20948 IMU	TDK InvenSense	10	6.91	69.08	Link
Components Total	N/A	N/A	N/A	234.96	
Labor	N/A	3	40.00	18 000.00	
Total	N/A	N/A	N/A	18 234.96	

4.2 Schedule



Week	Goal	Member
2/26	Decided on a microcontroller	Edan
	Decided on a voltage regulator	Sultan
	Decided on an IMU	Kaylan
3/4	Designed initial Control Unit PCB	Edan, Sultan
	Designed initial IMU PCB	Kaylan
	3/5 PCB ORDER ROUND 1	Everyone
3/11	Spring Break	Everyone
3/18	Experimented with devkit, found our devkit was faulty	Kaylan
	Used Voltage Regulator on breadboard to convert 4.8V to 3.3V	Edan
	Decided on LED driver	Sultan
3/25	Redid PCB to include LED driver testing	Edan
	Designed LED driver circuit	Sultan
	Tried to hotwire programming circuit on first pcb order	Kaylan
	3/26 PCB ORDER ROUND 3	Everyone
4/1	Used BJTs on breadboard to power LEDs controlled by Devkit	Sultan, Kaylan
	Used IMU breakout board to collect readings with Devkit	Edan
4/8	Designed final PCB to incorporate everything we accomplished on the breadboard as well as correct the programing circuit and add the LED driver circuit	Everyone
	4/9 PCB ORDER ROUND 5	Everyone
4/15	Programmed the devkit to detect gestures on breadboard	Edan
	Connected battery to breadboard so it could run in isolation (no computer or power supply)	Kaylan, Sultan
4/22	Soldered final PCB, moving everything from the breadboard to the new PCB	Edan
	Calibrated the IMUs, finalized gesture recognition software	Kaylan, Sultan

Figure 7: Schedule of the project

5 Conclusion

5.1 Accomplishments

We were able to build a device that could successfully aid riders in signaling their intentions to others. The device met all three high level requirements. It differentiates between turn left, right, slow down, and hazard with an accuracy of over 90%. The jacket also maps the hand gestures accurately to the respective LEDs. Finally, the LEDs are clearly visible from 250 ft.

So, using this product, we could ask anyone to wear it and ride the bike regularly. The rider only needs to know the hand gestures, which are widely known already. A big bonus is that it's easy to use, and not complicated. Other models might require the rider to press buttons which could be distracting and unnatural.

5.2 Uncertainties

Some things did not go according to plan. For example, the LEDs could have been slightly brighter using an LED driver. However, we decided to omit it to increase the chances of having a working PCB since adding the LED driver would have been at the very last minute. We couldn't test it beforehand or use a breadboard because the LED driver only came as an SMD.

In addition, we tested the arm gesture while standing upright, and we managed to get around a 90% accuracy. However, when we went to the bike, we did not have enough time to calibrate it well before the demo. We used the magnetometer to determine which of the arm gestures are detected. It might have been easier in terms of calibration if we used a more automatic way of calibrating. For example using a button as we discussed below. In addition, using a Neural Network might have given us a better outcome in terms of accuracy, since we can use the data from all the sensors to determine the arm gestures.

5.3 Future Work

1. **Smaller PCB enclosure.**

   3D printing a PCB enclosure takes 13 hours, so we decided to make it a little bigger to ensure it fits perfectly. Thankfully, it fits but it's a little large. It might cause some disturbance to the rider, so making it smaller would be a good idea.

2. **Using a third IMU on the right arm**

   Due to time constraints, we were not able to connect the third IMU. The ESP32 has 2 I2C pins, and we could make a new one in software or maybe even use a multiplexer. Having an additional IMU will allow the rider to use the right arm to turn right.

3. **Calibration button**

Before usage, we had to calibrate the device manually by reading off the angles the user holds their arms at when making an arm gesture. This could be affected by personal preference or type of bike. This would be unnecessarily difficult for a user, but decrease the accuracy if they don't go through the calibration. Because of this, we would like to add a calibration button where the user makes a gesture and presses the button to calibrate it. The control unit would read the angle measurements of the users' arms and use those as the values for gesture recognition rather than our default ones.

5.4 Ethical Considerations

The biggest concern as it relates to ethics and safety for this project is with regard to the safety of the user and those on the road around the user. Under the IEEE code of ethics (Code I1), we are required to prioritize the safety of the public [8]. If the wearable isn't user friendly enough, or restricts any movements, this can lead to potentially catastrophic accidents. We can solve this by integrating the electronics out of the way of the user, such as in the inner pockets of the jacket (for the PCB and battery), and providing ample slack in the wires throughout. This will allow the user to move more naturally. Another concern might be the privacy of the user [8] because we will be collecting and processing data constantly during a ride/commute. We can limit the data collection to IMU data, so that nothing personally identifiable is collected, as well as deleting any data past a certain period of time.

5.5 Safety Considerations

We have to consider the brightness of the LEDs, and if they can be distracting to other drivers and pedestrians. Having bright LEDs can be beneficial for low light or adverse conditions, but can also be harmful if they dazzle other drivers, impairing their vision. There aren't any safety regulatory requirements for LEDs for bicycles relating to the brightness of the lights, so we make sure we are following the vehicle regulations for turn signals. [9] There are also consumer product safety standards that we need to follow for wearable technology, such as those related to electronics devices and battery safety. It is also important to note that wearing a battery is always dangerous. Because of this we will be following the guidelines on battery safety outlined by UIUC [10]. In addition, we verified in our tolerance analysis that the linear regulators will not get too hot to ensure the safety of the users.

References

[1] Centers for Disease Control and Prevention, "Bicycle Safety," 2024, [Online; accessed 21-February-2024]. [Online]. Available: https://www.cdc.gov/transportationsafety/bicycle/index.html

[2] VectorStock, "Front back and side views blank jacket Royalty Free Vector," 2024, [Online; accessed 21-February-2024]. [Online]. Available: https://www.vectorstock.com/royalty-free-vector/front-back-and-side-views-blank-jacket-vector-4239549

[3] Texas Instruments, *LM317 3-Terminal Adjustable Regulator*, 2023, accessed: Feb. 22, 2024. [Online]. Available: https://www.ti.com/lit/ds/slvs044y/slvs044y.pdf

[4] *ESP32 Datasheet*, Espressif Systems, 2016, accessed: Feb. 15, 2024. [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf

[5] TDK InvenSense, "DS-000189-ICM-20948 v1.3," https://invensense.tdk.com/wp-content/uploads/2016/06/DS-000189-ICM-20948-v1.3.pdf, 2016, accessed: [Online; accessed 22-March-2024].

[6] Panasonic, *ACG4000 Series Datasheet*, Panasonic Industrial Devices, 2023, accessed on: [insert today's date]. [Online]. Available: https://industrial.panasonic.com/cdbs/www-data/pdf2/ACG4000/ACG4000CE275.pdf

[7] S. Electronics, "Sparkfun 9dof imu icm-20948 breakout hookup guide," Online Tutorial, 2023, accessed on: [insert today's date]. [Online]. Available: https://learn.sparkfun.com/tutorials/sparkfun-9dof-imu-icm-20948-breakout-hookup-guide/all

[8] Institute of Electrical and Electronics Engineers (IEEE), "IEEE Code of Ethics," https://www.ieee.org/about/corporate/governance/p7-8.html, 2024, accessed: Feb. 15, 2024.

[9] Legal Information Institute, "49 cfr § 571.108 - standard no. 108; lamps, reflective devices, and associated equipment." https://www.law.cornell.edu/cfr/text/49/571.108, 2023, accessed: Feb. 19, 2024.

[10] Division of Research Safety, University of Illinois at Urbana-Champaign, "Battery Safety," https://drs.illinois.edu/Page/SafetyLibrary/BatterySafety, 2023, [Online; accessed 22-February-2024].

Appendix A

A.1 Subsystem Requirements and Verification Tables

A.1.1 Control Unit

Table 3: Control Unit Subsystem Requirements and Verification

Requirements	Verification
• Must be able to communicate with the sensors and LEDs through the PCB – Must be able to output 4.8±0.3V to the LEDs – The accelerometer must be able to detect acceleration within ±1m/s. – The magnetometer must be able to detect North. – The gyroscope must be able to detect angular rate within ±15 dps. • Must be able to determine the correct turn signal from the IMU • Must be able to turn on the LEDs based on the corresponding turn signal – Turn left: Must be able to flicker the left LED – Turn right: Must be able to flicker the right LED – Slow down: Must be able to turn on both LEDs – Hazard: Must be able to flicker both LEDs	• Connect the IMU to the ESP32 such that the ESP32 can print the IMU acceleration readings from the accelerometer, gauss reading from the magnetometer, and the angular rate readings from the gyroscope. Drop 3 times (once for each x, y, z axis), and verify that the acceleration is g±5m/s. Point the IMU in different directions and verify that the gauss peaks when pointing North. Spin the IMU using a motor and verify that the angular rate readings match the dps of the motor ±20 dps. • Using the IMU imitate the motion of each of the arm gestures and print out the detected turn signal using the software of the ESP32. Verify if the correct turn signal is displayed on the LED • Move the IMU at a constant velocity and stop quickly (simulate a crash) and verify that the hazard signal is activated through the ESP32 software.

A.1.2 Power Subsystem

Table 4: Power Subsystem Requirements and Verification

Requirements	Verification
• Must be able to supply 3.3±0.3V to the IMUs and the ESP32 • Must be able to supply 4.8±0.3V to the LEDs • The temperature of the battery should stay below 50 C during operation	• The battery will be connected to the voltage regulator and a multimeter will be used to verify that the voltage output by one of the voltage regulators is 3.3±0.3V • To use the multimeter, the positive lead will be connected to the output of the voltage regulator and the negative lead will be connected to the ground of the voltage regulator • Measure the temperature using a thermometer during operation and ensure it is below 50 C

A.1.3 LEDs

Table 5: LED Subsystem Requirements and Verification

Requirements	Verification
• Must be able to turn on and off given a 4.8V input. • Must be bright enough for drivers to see from 250ft.	• Connect the LEDs in series with a switch and 4.8V power source and verify that the LEDs turn on. • Walk 250 ft away from the LEDs while they are on and check that they are visible

A.1.4 Sensor Subsystem

Table 6: Sensor Subsystem Requirements and Verification

Requirements	**Verification**
• The accelerometer must be able to detect acceleration within ±1m/s. • The magnetometer must be able to detect North. • The gyroscope must be able to detect angular rate within ±15 dps.	• Connect the IMU to the ESP32 such that the ESP32 can print the IMU acceleration readings from the accelerometer. Drop 3 times (once for each x, y, z, axis), and verify that the acceleration is g±5m/s • Connect the IMU to the ESP32 such that the ESP32 can print the IMU gauss reading from the magnetometer. Point the IMU in different directions and verify that the gauss peaks when pointing North. • Connect the IMU to the ESP32 such that the ESP32 can print the IMU angular rate readings from the gyroscope. Spin the IMU using a motor and verify that the angular rate readings match the dps of the motor ±20 dps.

A.2 Micellaneous

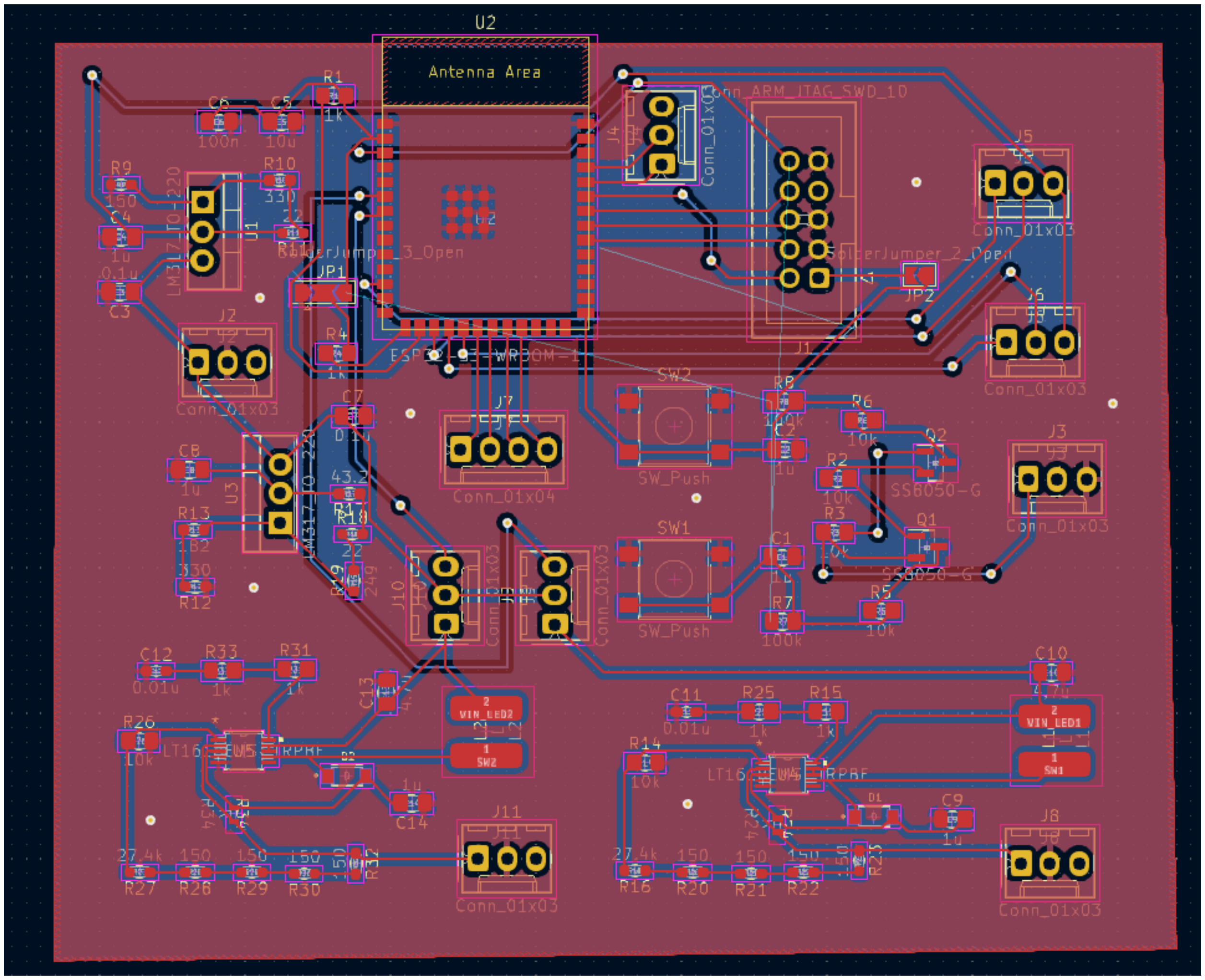


Figure 8: PCB layout of the final design

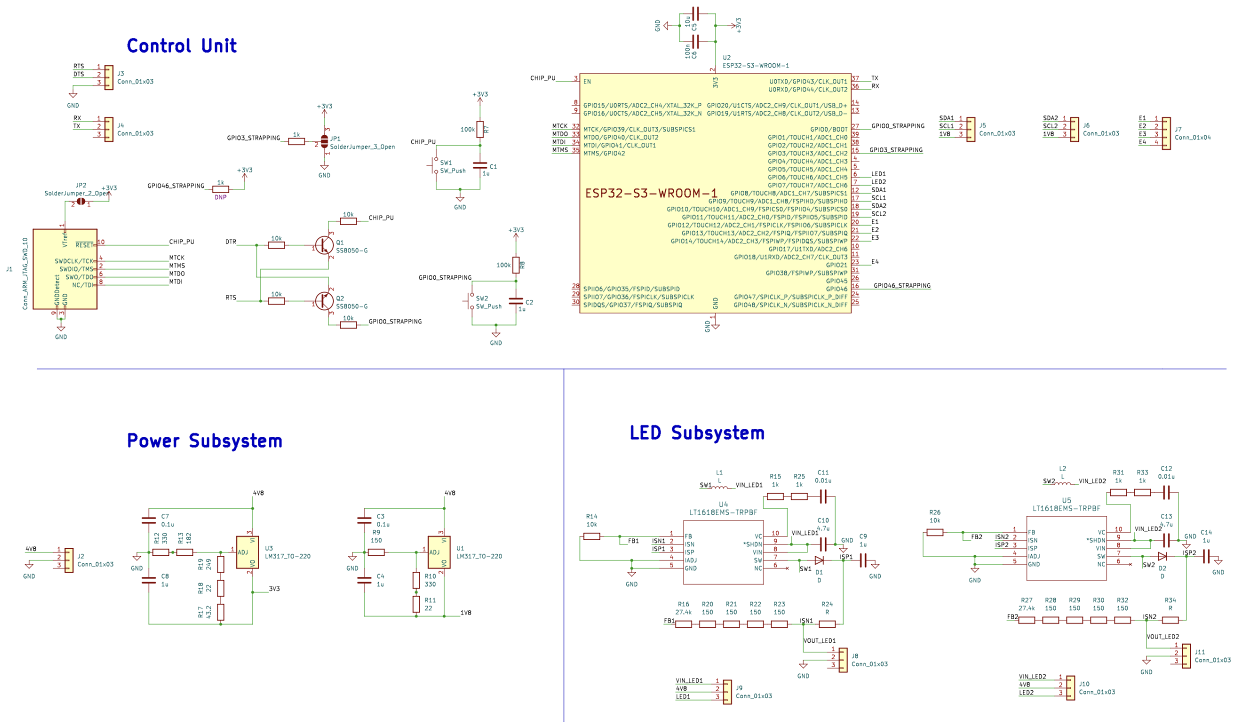


Figure 9: PCB schematic of the final design

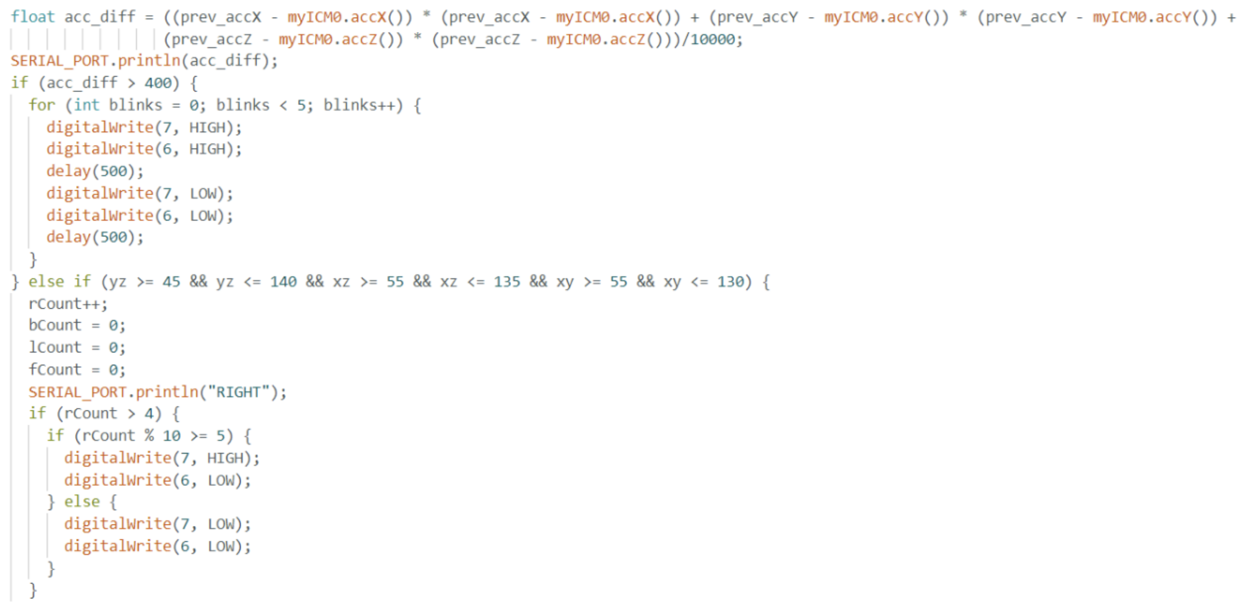
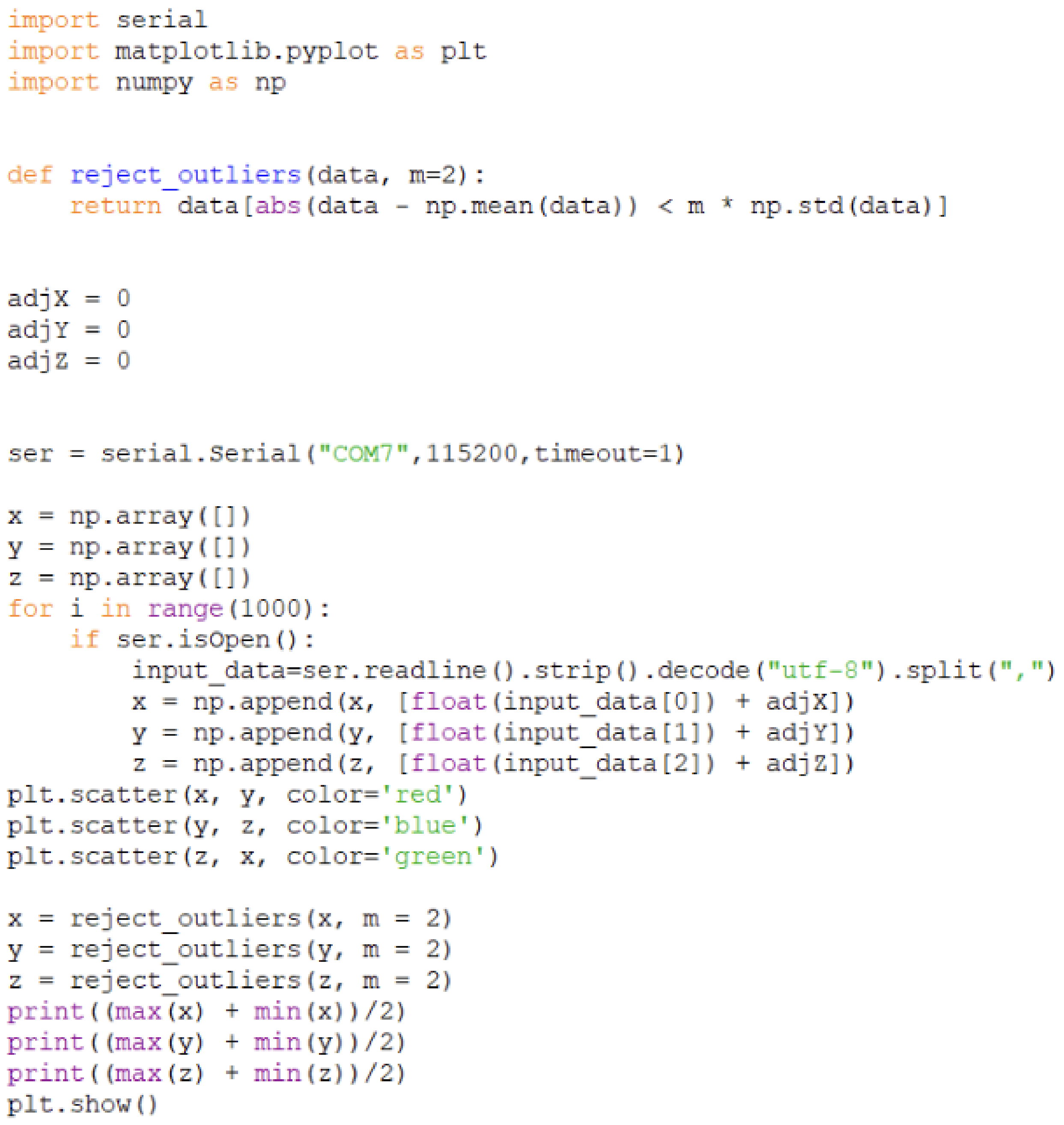

```
float acc_diff = ((prev_accX - myICM0.accX()) * (prev_accX - myICM0.accX()) + (prev_accY - myICM0.accY()) * (prev_accY - myICM0.accY()) +
                  (prev_accZ - myICM0.accZ()) * (prev_accZ - myICM0.accZ()))/10000;
SERIAL_PORT.println(acc_diff);
if (acc_diff > 400) {
  for (int blinks = 0; blinks < 5; blinks++) {
    digitalWrite(7, HIGH);
    digitalWrite(6, HIGH);
    delay(500);
    digitalWrite(7, LOW);
    digitalWrite(6, LOW);
    delay(500);
  }
} else if (yz >= 45 && yz <= 140 && xz >= 55 && xz <= 135 && xy >= 55 && xy <= 130) {
  rCount++;
  bCount = 0;
  lCount = 0;
  fCount = 0;
  SERIAL_PORT.println("RIGHT");
  if (rCount > 4) {
    if (rCount % 10 >= 5) {
      digitalWrite(7, HIGH);
      digitalWrite(6, LOW);
    } else {
      digitalWrite(7, LOW);
      digitalWrite(6, LOW);
    }
  }
```

Figure 10: Gesture recognition algorithm



```
import serial
import matplotlib.pyplot as plt
import numpy as np


def reject_outliers(data, m=2):
    return data[abs(data - np.mean(data)) < m * np.std(data)]


adjX = 0
adjY = 0
adjZ = 0


ser = serial.Serial("COM7",115200,timeout=1)

x = np.array([])
y = np.array([])
z = np.array([])
for i in range(1000):
    if ser.isOpen():
        input_data=ser.readline().strip().decode("utf-8").split(",")
        x = np.append(x, [float(input_data[0]) + adjX])
        y = np.append(y, [float(input_data[1]) + adjY])
        z = np.append(z, [float(input_data[2]) + adjZ])
plt.scatter(x, y, color='red')
plt.scatter(y, z, color='blue')
plt.scatter(z, x, color='green')

x = reject_outliers(x, m = 2)
y = reject_outliers(y, m = 2)
z = reject_outliers(z, m = 2)
print((max(x) + min(x))/2)
print((max(y) + min(y))/2)
print((max(z) + min(z))/2)
plt.show()
```

Figure 11: Calibration code for magnetometers