ENHANCED GOLF RANGEFINDER

By

Emma DiBiase

Jake Hindenburg

Peter Maestranzi

Final Report for ECE 445, Senior Design, Fall 2025

TA: Eric Tang

10 December 2025

Project No. 14

Abstract

This report details the process of the planning, building, and testing of the Enhanced Golf Rangefinder over the course of the Fall semester. Our device operates by measuring the distance to a target, like a normal industry rangefinder, while also measuring the environmental factors of wind speed, humidity, and temperature through various sensors. The effect of the environmental measurements on the flight of a golf ball is then calculated and used to provide an adjusted distance and summary of the data to the user through an OLED display. A golf club recommendation based on the measured data is also displayed on the OLED. Throughout the building and testing process, we found our device to be successful and were able to verify the requirements we set for the project.

Contents

1. Introduction

1.1 Objective

1.1.1 Problem

Golf is an extremely difficult game that requires a great deal of precision. There are a multitude of factors that can affect a single golf shot such as distance, weather conditions, and club choice. Jowett and Philips [1] show that even the most talented golfers will have shots and scores affected by adverse external conditions. While there are devices known as rangefinders that can measure the distance to a specific target on the course and help the golfer plan for their shot, these devices are not as precise as they can be. Due to the number of external factors that are present for a given golf shot, the adjusted distance that a golfer needs to plan for may not be the same as the distance provided on a normal rangefinder. Therefore, a more precise solution is necessary to help golfers plan for their shots better and overall improve their golfing abilities.

1.1.2 Solution

The solution to this problem is to implement a rangefinder that takes environmental factors into account when providing a distance measurement to the user. Our device measures not only distance but also wind speed, temperature, and humidity.

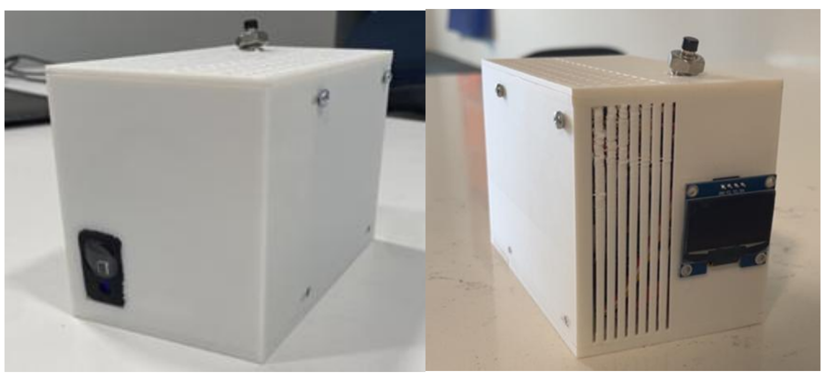

Figure 1: The Enhanced Rangefinder Front View (left) and Rear View (right)

Upon pressing the button, the enhanced rangefinder will measure the distance to the target, wind speed, temperature, and humidity. Immediately, the environmental data is sent to a microcontroller where the necessary calculations are performed to determine an adjusted distance based on the effects of the environmental measurements. The measurements, adjusted distance, and a suitable golf club recommendation based on the data are then displayed on the OLED display for the user to view. The user is now provided with a set of measurements that can help them better prepare for their upcoming shot.

1.2 High Level Requirements

Our three high level requirements are:

1. The rangefinder measures the correct distance from the user to the flag pin when the button is pressed. The distance should be constrained to the tolerance specified by our time-of-flight sensor's datasheet.
2. When the button is pressed, the environmental sensors collect the correct data and provide the necessary adjustments accordingly.
3. The user interface recommends a suitable club based on the measured distance. It is important to note that since we used a limited range time-of-flight sensor due to budget constraints, there was scaling that took place to meet this requirement. Our sensor measures distance up to 50 meters, while commercial rangefinders usually measure distance up to 200 meters and beyond. Due to this constraint, we scaled the measured distance by a factor of 4. For example, a measurement of 30 meters on our sensor will read as 120 meters on the display.

1.3 Block Diagram

Figure 2 depicts the high-level block diagram for our design. This includes 4 main subsystems: Power, Controller, Peripheral Devices, and Display.

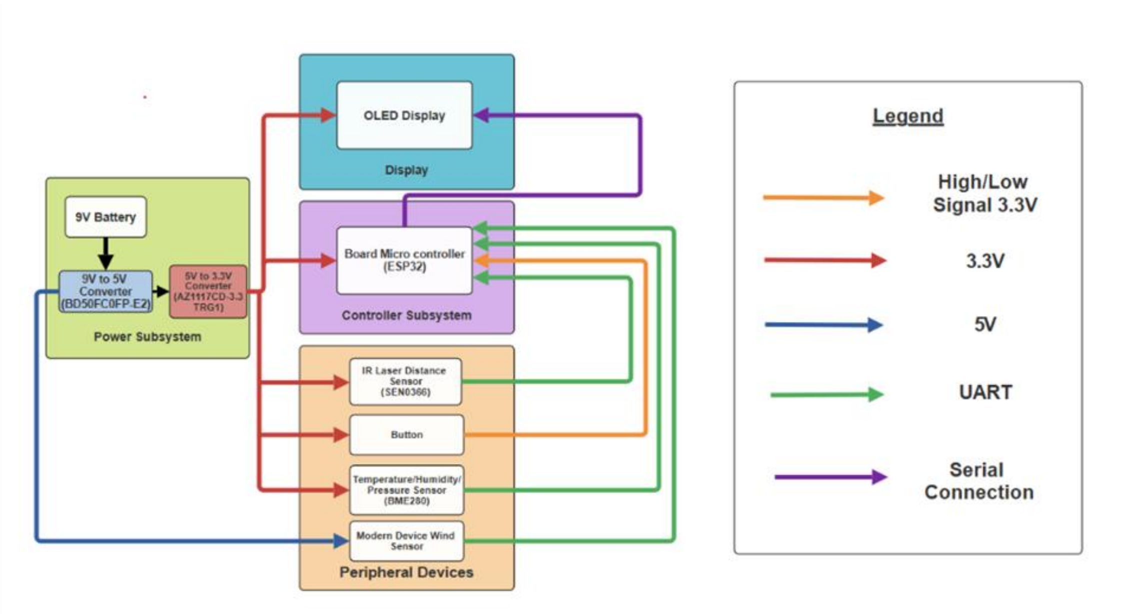

Figure 2: Block diagram of the Enhanced Golf Rangefinder's overall high-level design

1.4 Subsystem Overview

In this section, we will provide an overview of each of our device's subsystems. The results of testing and verifying these subsystems will be provided in a later section.

1.4.1 Power Subsystem

The power subsystem consists of a primary power source, a 9V DC Duracell battery, which is then converted to 5V via the LM2596S-5 buck converter. This 5V output is then integrated into the overall circuit design as it is used to power the separate Wind Speed sensor PCB we designed. Additionally, this 5V output is then stepped down to 3.3V through the LD1117S33 linear voltage regulator which is being utilized to power the remaining components of our circuit. This includes the BME280 temperature and humidity sensor, the SEN0366 time-of-flight sensor, the OLED Display, and the ESP32-S4-WROOM-1-N16 microcontroller.

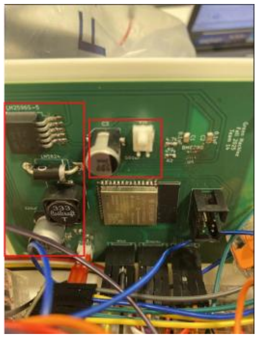

Figure 3: Power Subsystem Location on Main PCB

1.4.2 Controller Subsystem

The controller subsystem is based around the ESP32-S3-WROOM-1-N16 microcontroller. The ESP32-S3, with respect to the other subsystems, receives the required 3.3V from the Power Subsystem, takes sensor data from the Peripheral Devices Subsystem as the input to the controller, and outputs data onto the Display subsystem. For programming the device, we utilized the Arduino IDE. We also used the ESP32-Prog device to upload our program to the microcontroller accordingly.

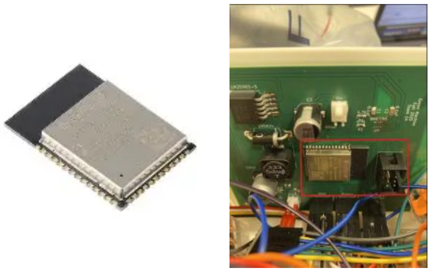

Figure 4: ESP32-S3-WROOM-1-N16 (left) and Controller Subsystem location

on main PCB (right)

1.4.3 Peripheral Devices Subsystem

The peripheral devices subsystem is composed of the SEN0366 time-of-flight sensor, BME280 temperature and humidity sensor, and the wind speed sensor, which was integrated into its own PCB and designed based on the Wind Sensor Rev. C from Modern Device [2]. Except for SEN0366, each sensor was integrated into either the main PCB or a separate PCB, meaning the only commercial product that we directly used in our design was the time-of-flight sensor.

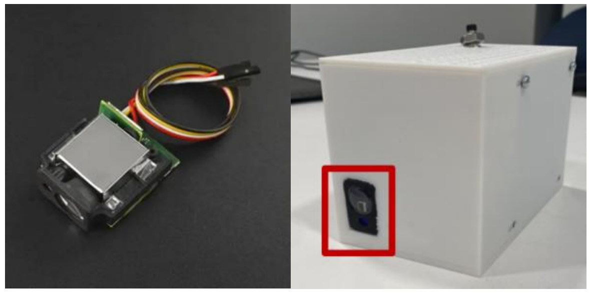

Figure 5: SEN0366 Sensor (left) and Position of Sensor in Enclosure (right)

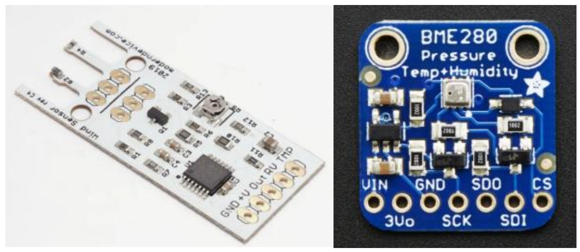

Figure 6: Wind Sensor Rev. C (left) and BME280 (right)

The role of the sensors in the peripheral devices subsystem is to collect data from the environment and then output this data to the microcontroller subsystem. The input to the subsystem is 3.3V and 5V from the power subsystem depending on the specific sensor. The BME280 and SEN0366 both receive 3.3V and the wind speed sensor receives 5V.

1.4.4 Display Subsystem

The display subsystem consists of simply just the 1.3" OLED Display monitor with a SH1106 driver chip. This system is receiving 3.3V from the power subsystem and is connected to the microcontroller via serial connection. The microcontroller communicates with the OLED display, telling it to output the peripheral devices' data that the system has measured onto the screen itself for the users to see. This process occurs when the button is pressed, triggering the system's program to run.

Figure 7: OLED Display Module

2 Design

2.1 Main PCB

The main PCB contains the bulk of our design and components used. This includes our Power Subsystem, Controller Subsystem, and part of the Peripheral Devices Subsystem. It was important to keep the final product as compact as possible, so we made the decision to design a separate PCB for our wind speed sensor rather than adding it to the main PCB. This allowed us to be more flexible with the space we had in our enclosure.

2.1.1 Main PCB Board layout and Schematic

Figure 8 shows the main PCB mounted within the enclosure as well as a 3D model.

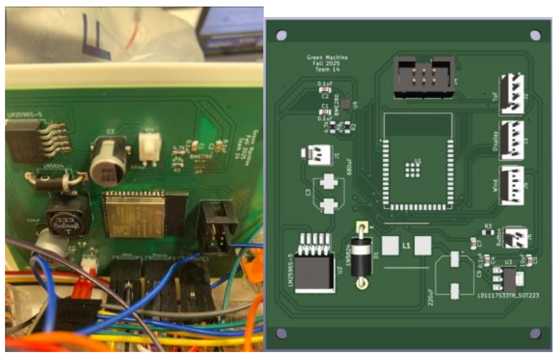

Figure 8: Main PCB mounted within enclosure (left) and 3D model of PCB (right)

The final dimensions of the main PCB are 89mm x 77mm. This fits under the limit of 100mm x 100mm for PCB sizing. For mounting, M3 mounting holes and crews were used to hold up the PCB and keep steady so that connections were not jostled easily during device usage. A PCB layout is depicted in Figure 9. Both PCBs were designed using the KiCad application.

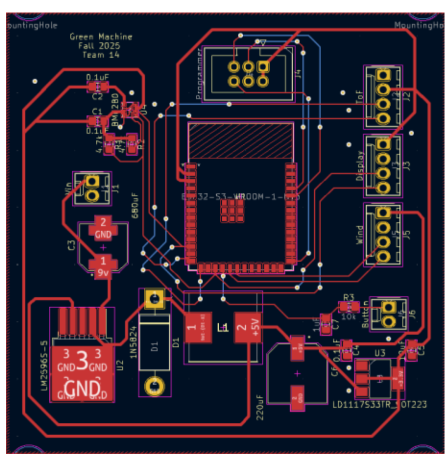

Figure 9: Main PCB layout and wiring

2.1.2 Power Subsystem Design

Since our power subsystem is the backbone of our overall design, we needed to ensure we met our proper output voltages while maintaining a high efficiency. Subsequently, we chose to go with the 9V DC battery as our power source since it was easily available and our design has lower power consumption. After going through the fine details of our components' datasheets, we found that the wind speed sensor drew about 20-40mA at an operating voltage of 5V [2]. This led us to conclude that we would need to step down our voltage from 9V to 5V. Since this is a slightly bigger voltage decrease, we opted to go with a buck converter because it utilizes the constant switching mechanism with MOSFETs. Thus, rather than burning off excess voltage as heat, the power is switching on/off which leads to a much higher efficiency and will prevent the 9V DC battery from quickly draining. The specific buck converter we chose was the LM2596S-5 which has an efficiency of 80% according to the datasheet, which was more than enough to meet our needs [8]. Similarly, we found that the remaining components of our design all maintained an operating voltage of 3.3V, so in this case we opted for a 5V to 3.3V linear voltage regulator (LD1117S33). This is because the voltage drop here is much less significant, so we are not dissipating so much heat and is therefore a safe and adequate option for the design. We found that when each component is operating, the load on the regulator is about 190mA, so an overall low-powered design at about 0.63W for the regulator and 0.2W for the buck converter.

Device	**Load**
OLED Display	Max 30mA (Screen On) [7]
Time of Flight Sensor (SEN0366)	Max 130mA (In Operation) [6]
BME280	0.0036mA [5]
ESP32-S3-WROOM-1 Microcontroller	Max 30mA (WIFI/Bluetooth Disabled) [11]
Total	~190mA -> 200mA Safety Margin

Table 1: Total Load Current on LD1117S33 Linear Voltage Regulator

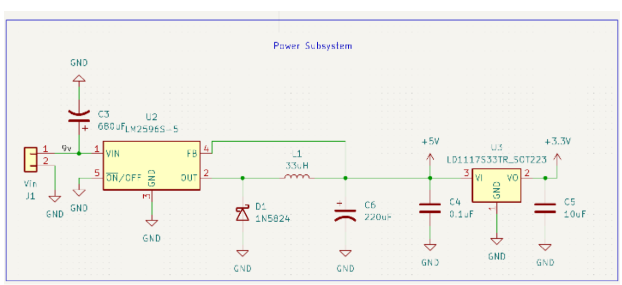

Figure 10: Power Subsystem KiCad Schematic

Following the LM2596-5 datasheet [8], we used a Cin value of 680 microfarads, an inductor value of 33 microhenries, a Cout value of 220 microfarads, and a Schottky diode (1N5824) as used in typical applications, and we placed these on our PCB. These components reduce inductor ripple, control switching spikes, and reduce noise within the subsystem.

2.1.3 Microcontroller

When deciding which microcontroller would best fit this design, we had multiple requirements to meet. These requirements included high processing power, low power consumption, sufficient number of

GPIOs, and easy to program using the Arduino IDE. We decided to use the ESP32-S3-WROOM-1-N16 microcontroller for our design, as it easily met these requirements and provided additional features for any desired future improvements to the device. For processing power, the ESP32-S3 uses a dual-core processor that can run up to 240MHz. This allows plenty of room for computations made in the program for signal processing, sensor interaction, calculations, and quick updates. This allows us to give users accurate data at the moment they activate the device for measurements. Even with this high processing power, the ESP32-S3 is efficient and has low power consumption, drawing a max 20-25mA of current. There are a lot of GPIOs included with the ESP32-S3 as well, including support for different functions like UARTs, I2C, and ADCs. Figure 11 includes the pinout of the microcontroller, showing the variety available. Finally, the ESP32-S3 is excellent for working in the Arduino IDE as there are many available libraries for sensors. This helped limit a lot of low-level debugging when setting up our program.

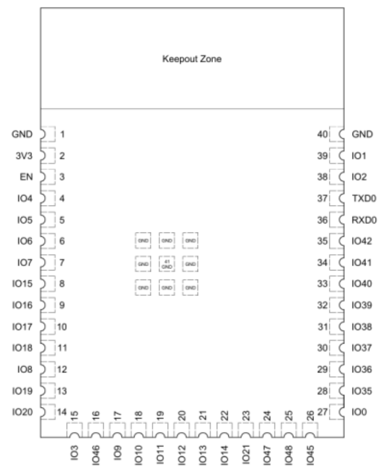

Figure 11: ESP32-S3 pin layout

2.1.4 Peripheral Devices

Apart from the wind speed sensor, peripheral devices required were a distance sensor and a temperature, humidity, and pressure sensor. Due to our budget, we wanted to utilize a distance sensor that could both measure as relatively far distance while staying within our budget range. This ultimately led to the choice of using the SEN0366, measuring 50 meters at max distance and staying within our budget range. This device was also a good choice due to its ease of programming with the Arduino IDE. For our Temperature, wind, and pressure sensor, we stuck with the BME280 as it is well known for being an accurate and reliable sensor for this relevant data. Because we want to give users the most accurate environmental measurements possible, we went with the best sensor to accomplish this.

2.2 Wind Sensor PCB

As mentioned earlier, the wind sensor was separated from the main PCB to be integrated into a separate PCB. This decision was made because there are many components on the schematic and we did not want to include these components in the main PCB because this would have introduced some spacing issues into our device.

2.2.1 Background Mathematical Information

The wind sensor was one of the most important components of our project, due to the volatility of wind and how difficult it is to accurately measure wind. Therefore, we conducted research on the effects of wind on the flight of a golf ball and based our Design Document tolerance analysis on the effects of wind. A general analysis of the math and physics of wind acting on the golf ball is as follows: The forces acting on the ball are gravity ($\boldsymbol{Fg}$), quadratic drag from air resistance ($\boldsymbol{F_D}$) and lift due to ball spin ($\boldsymbol{F_L}$). The equations for these forces are below. Note that vector quantities are bold.

$$\boldsymbol{F_g} = m\boldsymbol{g} = -mg\hat{\boldsymbol{y}}$$

$$\boldsymbol{F_D} = \frac{1}{2}\rho C_D A {v_{rel}}^2 \widehat{\boldsymbol{v_{rel}}}$$

$$\boldsymbol{F_L} = \frac{1}{2}\rho C_L A {v_{rel}}^2 \hat{\boldsymbol{n}}$$

$$m\frac{dv}{dt} = \boldsymbol{F_g} + \boldsymbol{F_D} + \boldsymbol{F_L}$$

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{v_x}$$

$$x_{wind} = \int_{\boldsymbol{0}}^{\boldsymbol{t_{impact}}} \boldsymbol{v_x dt}$$

$$\Delta x = x_{wind} - x_{measured}$$

Where:

$\boldsymbol{F_D}$: drag force [3]

$\boldsymbol{F_L}$: lift force [4]

m: mass of the golf ball (45.93 g)

$\boldsymbol{g}$: acceleration due to gravity (-9.81 m/s^2)

ρ: air density (1.2250 kg/m^3)

C_D: drag coefficient

A: cross-sectional area of the golf ball (.00143 m^2)

$v_{rel} = ||\boldsymbol{v_{ball}} - \boldsymbol{v_{wind}}||$

$\widehat{\boldsymbol{v_{rel}}}$: unit vector of v_{rel}

C_L: lift coefficient

x_{wind}: calculated “adjusted” distance based on wind speed and direction

$x_{measured}$: distance measured by the Time-of-Flight sensor

This math is part of the programming in the microcontroller that overall helps determine the distance adjustment from wind. The distance adjustment due to wind can be modeled as $\Delta R = R(wind) - R(no\ wind)$ where $R(no\ wind)$ is the distance measured by the time-of-flight sensor and $R(wind)$ is equal to x_{wind} above. Obviously, the drag coefficient, lift coefficient, and ball velocity are all values dependent on the golfer. The drag coefficient is dictated by a value known as the Renolds number Re. The Renolds number is defined as $Re = VD/\mu$ where V is linear speed, D is the diameter of the ball, and μ is the kinematic viscosity of air. Per Jenkins [3], the drag coefficient of an average golf ball sits between 0.24 and 0.7, with the lower bound correlating to higher ball speeds and the upper bound correlating to lower ball speeds. The lift coefficient is dictated by a spin parameter S. S is defined as $S = \omega r/V$, where is the angular velocity and r is the radius of the golf ball. Lift coefficient generally resides between 0.1 and 0.3. As for the velocity of a golf ball, it decreases as the club loft increases, i.e. a 9-iron will have less ball velocity than a 6-iron. Therefore, when implementing our distance adjustment in the microcontroller, we pass values for the coefficients and ball speed accordingly based on the initial distance measured.

2.2.2 Schematic

Below is the schematic of the wind PCB

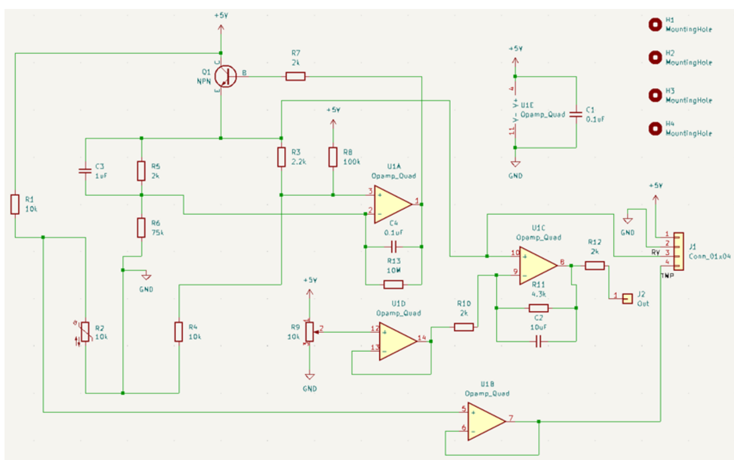

Figure 12: Wind Sensor PCB Schematic

2.2.3 Board Layout and Functionality

Below is the layout of our wind sensor PCB and how it appears within the enclosure.

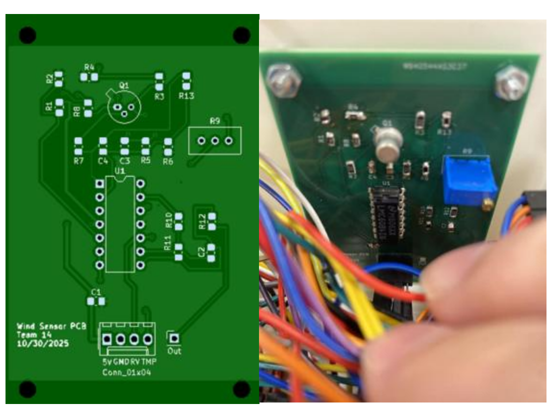

Figure 13: Wind Sensor Layout (left) and Wind Sensor Within Enclosure (right)

The 4 pin connector at the bottom of the board represents the connections that are being received from the main PCB (5V and GND) and sent to the main PCB (RV and TMP). The single pin connector "out" ended up being inconsequential to our design and was left floating. While the organization of the components on the PCB is clear, it is also important to describe the functionality of the sensor. The type of wind sensor that is implemented on the PCB is a "hot-wire" sensor. The functionality of our sensor is as follows: Once the PCB connected to power, the thermistor (R2) is heated to its operating temperature. After this, any airflow that is detected will cool the thermistor. The magnitude of the cooling of the thermistor is then translated into a wind speed measurement. This measurement is what is shown to the user on the OLED.

2.3 Enclosure

For the physical design of our device, we decided to 3D print a rectangular enclosure using PLA filament that replicates industry-standard golf rangefinders. This includes making the enclosure as compact and handheld as possible while still comfortably fitting our PCBs, wires, and sensors. This led us to create a 3D model on openSCAD with dimensions of 132mm (L) x 76mm (W) x 87mm (H) which accounted for the sizes of both of our PCBs while still fitting in your hand. Additionally, we implemented a removable lid which included ventilation holes along its entire length as well as ventilation grooves on the back face of the enclosure. This allowed for steady airflow to travel through the enclosure and hit the wind speed sensor so it can make accurate measurements, as well as provide a release for any heat that is dissipated in the power subsystem. The final design choice we made for the enclosure was including mounting holes for the PCBs, OLED, and button, as well as creating a cutout in the front face for the time-of-flight laser to emit.

Figure 14: Final 3D Enclosure Print

2.4 Programming

The programming for the device is based on the flowchart shown in Figure 15.

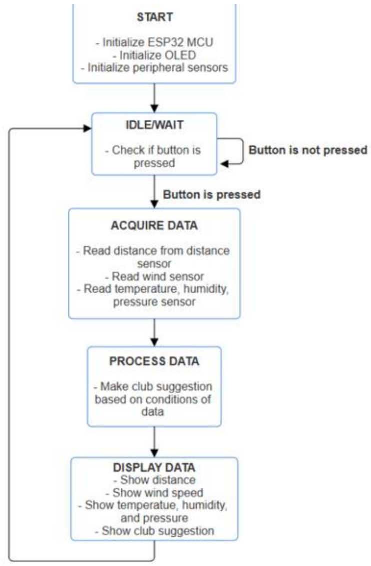

Figure 15: Program flowchart

Although the flowchart for this program is not extremely complex, there was a lot of necessary data filtering, compensation, and calculations that were required within our data processing step. The program begins by initializing the microcontroller, OLED, and peripheral devices. For communication, the OLED uses SPI, the SEN0366 distance sensor uses UART, the BME280 uses I2C, and the wind speed sensor uses basic ADC. These, compared to the original block diagram legend, differ as they ended up being much simpler communication options to implement for each device. The device then waits until the user presses the button to measure and process data. This raw data is filtered and compensated accordingly. Figure 16 displays an example of the rolling average filtering that our SEN0366 sensor utilizes to meet our distance accuracy requirements.


sketch_oct27a.ino

```
for (int i = 0; i < 10; i++) check += data[i];
check = ~check + 1;

if (data[10] == check) {
  if (data[3] == 'E' && data[4] == 'R' && data[5] == 'R') continue;

  float distance = (data[3] - '0') * 100 +
                   (data[4] - '0') * 10 +
                   (data[5] - '0') +
                   (data[7] - '0') * 0.1 +
                   (data[8] - '0') * 0.01 +
                   (data[9] - '0') * 0.001;

  if (!isnan(lastValid) && fabs(distance - lastValid) > MAX_DIFF) {
    heldReading = true;
    return lastValid;
  }

  distances[sampleIndex] = distance;
  sampleIndex = (sampleIndex + 1) % NUM_SAMPLES;
  if (sampleIndex == 0) bufferFilled = true;

  float avg = 0;
  int count = bufferFilled ? NUM_SAMPLES : sampleIndex;
  for (int i = 0; i < count; i++) avg += distances[i];
  avg /= count;

  lastValid = avg;
  heldReading = false;
  return avg;
}
```

Output

Figure 16: SEN0366 distance filtering and averaging in Arduino IDE

2.5 Design Adjustments

During the design process we found that we needed to make a few adjustments through testing and debugging. The first instance of this came as we created the power subsystem. In our original design, rather than using a 9V to 5V buck converter we planned to use a linear voltage regulator. However, we quickly found this to be not ideal since this was a large voltage drop and the regulator burns off excess voltage to maintain a steady output. Consequently, this was causing power loss and heavy amounts of heat dissipation which we wanted to avoid since we would be taking temperature measurements. This is because the excess heat in the enclosure could have skewed the data and output inaccurate readings. As a result, we switched this out for the LM2596S-5 which provides much higher efficiency and therefore less heat dissipation.

Another obstacle we ran into was the first version of our main PCB. Rather than using the dedicated Rx and Tx pins in our microcontroller for programming, we originally designed the PCB to use the standard GPIO Rx and Tx pins. This became an issue because the ESP32 was not recognizing our code since it was failing to be uploaded. As a result, none of our sensors could be calibrated so we re-ordered a new PCB with the new connections to the correct Rx and Tx pins.

3. Design Verification

All the requirements and verification tables can be found in Appendix A.

3.1 Power Verification

The power subsystem verification confirms that we are outputting our correct voltages within a suitable tolerance to ensure that each of our components are operating at full power. This includes the battery outputting 9V, the buck converter outputting 5V, and the regulator outputting 3.3V all within +/- 5%. We tested these requirements by using the lab multimeter and probing the outputs. The results of these tests can be seen below.

Figure 17: Output of Voltage Regulator (Left) Output of Buck Converter (Right)

Additionally, we found that we were maintaining an 80% efficiency in our buck converter which satisfied our needs [8]. We were also operating within 0 to 1A seeing as our wind speed sensor drew a max of 40mA and the rest of our design drew about 190mA. See Table 1 in section 2.1.2. Unfortunately, the only verification we did not meet was the efficiency of the linear voltage regulator [9]:

$$Pin = Vin * (Iload + Iquiescent) = 5V * (0.190 + 0.01) = 1W$$

$$Pout = Vout * (Iload) = 3.3V * 0.190 = 0.627W$$

$$Efficiency = \frac{Pout}{Pin} * 100 = 62.7\%$$

This is likely due to the functionality of a linear voltage regulator, in which it burns off excess voltage which lowers efficiency. Despite this being lower than ideal, we still found that it adequately met our needs in our design while maintaining a safe operating temperature well below its max rated 125 degrees Celsius since it was a small voltage drop that did not dissipate excessive heat. Additionally, it was a more inexpensive and simple option.

3.2 Controller Verification

The Controller subsystem verification ensures that the microcontroller is programmed properly, sensors gather data, and data is processed and output onto the OLED display. Because our device was fully functional, we can verify the Controller subsystem works as intended.

3.3 Peripheral Devices Verification

The verification of the peripheral devices subsystem consisted of comparing the measurements of our sensors against the measurements of commercial sensors and verifying that our measurements were within a reasonable tolerance. For our time-of-flight sensor, we wanted to verify that our scaled version was exactly 4X the distance of a corresponding tape-measured distance, and it was.

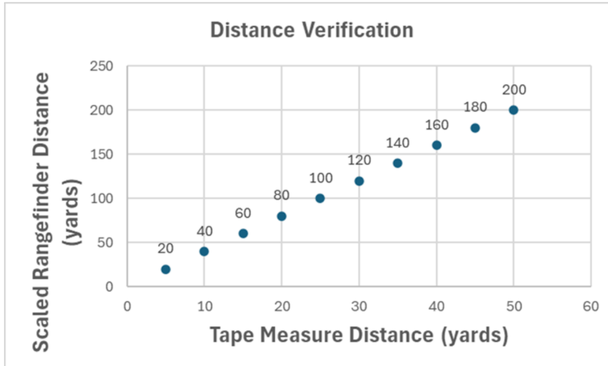

Figure 18: SEN0366 Verification

For the BME280, we wanted the humidity to be within 5% of the humidity percentage listed on the weather app and the temperature to be within 1 degree Celsius of the temperature listed on either the thermostat if indoors, or the weather app if outdoors. Both are verified by the data below.

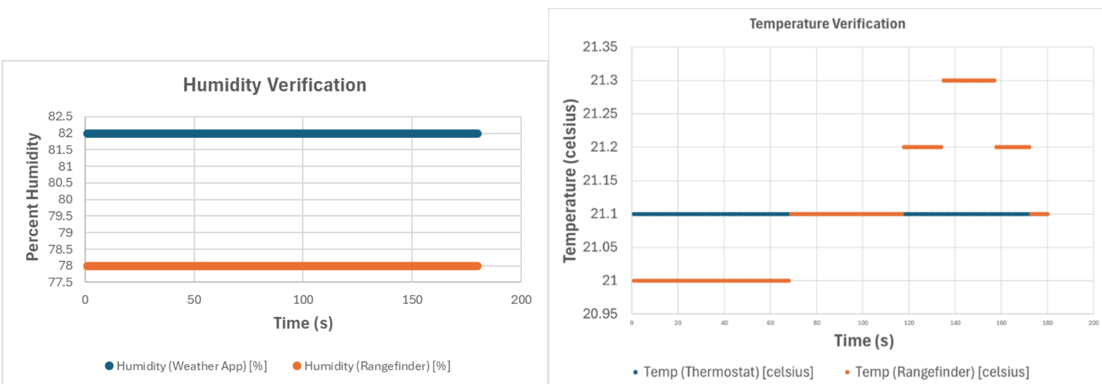

Figure 19: BME280 Verification of Humidity (left) and Temperature (right)

The only verification that we did not meet was the wind speed sensor. We wanted to verify our readings to within 5% of the readings of a commercial wind meter but were only able to get to as low as 14.7% when comparing the two devices on the same time interval.

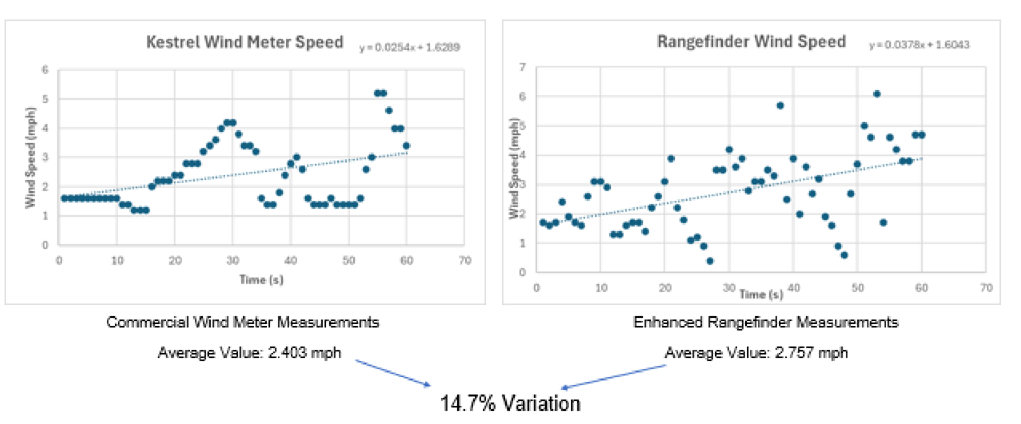

Figure 20: Verification of Wind Sensor

While the verification failed, the average wind speeds measured across the interval were still similar. We would attribute the error to the fact that our sensor contains a thermistor that is extremely sensitive, so it is prone to produce spikes in wind measurements and has an overall more scattered dataset. The wind meter that we measured it against does not have this design, so its data was much more compact. We are overall encouraged that the trendline and average values of the measurements were similar, even though the data did not fall within the 5% constraint that we had originally hoped.

3.4 Display Verification

The display subsystem verification simply consisted of successfully displaying our data onto the monitor for the user to see. We confirmed this by running the program and demonstrating that our measurements are visible. This included the temperature, humidity, wind speed, raw distance, adjusted distance, and club suggestion.

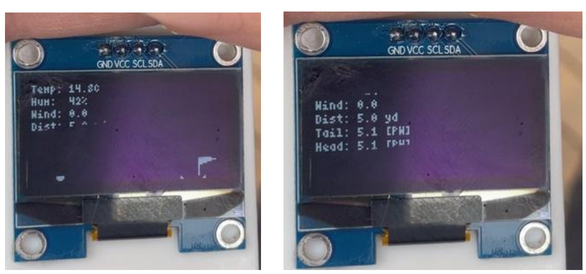

Figure 21: OLED Display Verification

4. Cost and Schedule

4.1 Parts

Below is a table of all the parts we utilized throughout the course of the project and the total cost of these parts. Note that the total cost of parts is greater than the $150 budget provided by the department for parts. We did use our own money to pay for some of the parts due to tariff rules and time constraints, so we did not request any additional budget from the department.

Part Name	Quantity	Manufacturer	Vendor	Cost per Unit	Total Cost
ESP32-S3-WROOM-1-N16 (Microcontroller)	4	Espressif Systems	ECE Shop	$6.56	$26.24
Modern Device Wind Sensor	1	Modern Device	Modern Device	$21.94	$21.94
BME280 3.3V Temperature Humidity Sensor Atmospheric Barometric Pressure Sensor Module	4	ATNSINC	Amazon	$12.99	$25.98
Pcs 1.3 Inch IIC I2C OLED Display Module	5	Hosyond	Amazon	$17.99	$17.99
SEN0366 IR Laser Distance Sensor	1	DFRobot	DigiKey	$80.00	$80.00
LD1117S33 (5 to 3.3) Linear Voltage Regulator	5	STMicroelectronics	DigiKey	$0.32	$1.60
LM2596S 5V Buck Converter	3	Texas Instruments	DigiKey	$6.99	$20.97
SMD Resistors and Capacitors	-	-	ECE Student Self Service and Senior Design Lab	-	-
LMC6684 Quad CMOS Operational Amplifier, Rail-to-Rail	1	Texas Instruments	ECE Student Self Service	-	-
2N3055G BJT NPN	1	onsemi	ECE Student Self Service	-	-
NTC0805J10K NTC Thermistor 10k	5	TE Connectivity Passive Product	DigiKey	$0.90	$3.97

0805					
Bourns 3296W-1 10k Potentiometer	5	Bourns	Digikey	$1.73	$8.65
EEE-FT1C681GV 680 µF Inductor	4	Mouser	DigiKey	$1.73	$6.92
Total Cost					$205.74

Table 2: Total Parts Cost

4.2 Labor

Below is a table of the cost of labor using a realistic estimate of an hourly rate that would be utilized in an industry project application.

Name	Hourly Rate	Hours	Total Cost
Peter Maestranzi	$125	300	$37,500
Jake Hindenburg	$125	300	$37,500
Emma DiBiase	$125	300	$37,500
Combined Cost			$112,500

Table 3: Labor Cost

4.3 Schedule

Below is the schedule and general work allocation that we utilized throughout the semester.

Week	Task	Responsibility
09/15/2025	**Finish Proposal**	All
	Soldering Assignment	All
	Team Contract	All
	Prepare for Proposal Review	All
09/22/2025	Proposal Review	All
	Order Parts	All
	Design PCB	Emma
	Update Schematics/Study Sensor Datasheets	Peter
09/29/2025	Finish PCB Design	All
	Test Sensors/Start to Program Microcontroller	Jake
	PCB Review	All
10/06/2025	**Breadboard Demo 1**	All
	Update Design Document	Peter
	Turn in Design Document	All
10/13/2025	Update PCB and Order Again if Necessary	Emma
10/20/2025	Component Testing	Jake
10/27/2025	**Breadboard Demo 2**	All

	Finalize Programming	Jake
11/03/2025	**Individual Progress Reports**	All
	3D Print and Assemble Parts in Enclosure	Emma
11/10/2025	Testing on Entire Device	All
	Finalize Documentation	Peter
11/17/2025	**Mock Demo**	All
	Final Assembly of Device	All
	Team Contract Assessment	All
11/24/2025	Fall Break	N/A
12/01/2025	**Final Demo**	All
	Mock Presentation	All
	Finalize Presentation	All
12/08/2025	**Final Presentation**	All
	Final Papers	All
	Lab Checkout	All

Table 4: Schedule

5. Conclusion

5.1 Accomplishments

Overall, our final design accomplished each of the high-level requirements that we had set from the beginning of the semester. Despite our success, the process of realizing this design had its roadblocks. This included waiting on PCB and parts orders, various design adjustments, as well as inexperience in specific areas of design. Through hard work, collaboration, and thourough documentation, we were ultimately able to accomplish designing a golf rangefinder device that measures an adjusted distance based on the raw distance and environmental conditions.

5.2 Uncertainties

The main uncertainties that occurred during our project primarily pertained to the wind speed sensor. We found during testing that our rangefinder had an average wind speed measurement of 2.757mph over a 60 second interval, while the commercial wind meter we used as comparison had an average value of 2.403mph during the same 60 second interval. See this data table in section 3.3. This led to a 14.7% difference in results which exceeded our 5% uncertainty requirement. As mentioned before, we deducted this issue being caused by the sensitivity of the thermistor since there are many spikes and inconsistencies in the graph. Unfortunately, we could not come up with a concrete solution due to timing constraints with the arrival of the PCB and the final demo; however, we managed to decrease the severity by adding smoothing filters into the program code.

As stated previously, there was also uncertainty in meeting the linear voltage regulator efficiency requirement where the calculations can be seen in section 3.1. This led to a difference of 27.6% between our true efficiency and ideal efficiency. This can be attributed to the functionality of the voltage regulator. However, despite this uncertainty we opted to keep the component since it remained a safe and inexpensive option as it was not dissipating excessive heat (did not skew temperature data) and properly powered our remaining components.

5.3 Ethical considerations

Throughout the semester we remained diligent in adhering to the IEEE Code of Ethics [10]:

[1] We will put the safety, health, and welfare of the first, act with honesty and integrity to avoid harm, provide clarity with risks, and minimize negative impacts on the environment by designing our system to prioritize safety, disclose known limitations, and document our testing

[2] We will work to deepen our own understanding of the technology while using it appropriately, remaining aware of the possible consequences

[3] We will be honest and realistic when describing the performance and expected outcomes of our project

[5] We will welcome any constructive feedback, acknowledge our mistakes, and give credit to others where it is due

[6] We will only take on risks for which we have proper training and fully state any limitations in our expertise

[9] We will avoid any actions that may cause harm to people, their property, reputations, or professional growth

[10] We will support our teammates and treat one another with respect as we grow in knowledge and encourage one another to uphold these ethical standards

We also ensured that we followed several ethical standards that applied to our project outside of the IEEE Code:

- We will ensure our device is made to be intuitive and usable for people of different physical abilities
- We will design our device for modularity and repairability rather than disposal
- Our device should only be used as an aid, not as a means of cheating or an unfair advantage
- We will ensure that the data displayed is accurate and easy to interpret so users do not make poor decisions based on misread information
- We will ensure that users understand that our device provides guidance only and does not guarantee performance outcomes or replace users' skill

5.4 Future work

When considering possible future work with extended time and resources dedicated to our project, there are many possibilities that we could consider. One such possibility is using a longer-range time-of-flight sensor. This would immediately propel our project to being a device that is industry-standard, because most commercial rangefinders can measure up to 200+ yards. Another possible addition would be a user interface in the form of a phone app with Bluetooth capabilities. This way, users can enter their personalized golf data, such as how far they hit each club. With a feature such as this, our rangefinder could be altered on an individual level where golfers of all calibers can benefit in their own way. Another possibility is a new wind speed sensor design. Given the resources available to us, we were pleased with how our wind sensor performed, but there were issues with spiking and overall inconsistent reads of wind speed. With a wider range of components available for selection, we could create a sensor that reads wind with the accuracy of a other commercial sensors. Finally, one last addition to our project could be to make the design more compact. Some rangefinders available commercially are small enough to fit in a person's pocket. Our design can be held with one hand but is bulky in the sense that it contains numerous PCBs and wires within the enclosure. With more advanced components, we would not need as much space, thus we could create a smaller enclosure.

References

[1] H. Jowett and I. D. Phillips, “The effect of weather conditions on scores at the United States Masters Golf Tournament,” International Journal of Biometeorology, vol. 67, no. 11, pp. 1897 1911, Sep. 2023.

[2] Modern Device, "Wind Sensor Rev. C," 2025. [Online]. Available: https://moderndevice.com/products/wind-sensor

[3] P. E. Jenkins, J. Arellano, M. Ross, and M. Snell, “Drag coefficients of golf balls,” World Journal of Mechanics, vol. 08, no. 06, pp. 236–241, 2018.

[4] R. Cross, Sports ball aerodynamics rod cross, Physics Department, University of Sydney, https://www.physics.sydney.edu.au/~cross/TRAJECTORIES/Sports%20Balls.pdf

[5] Bosch Sensortec, "BME280 Digital Humidity, Pressure and Temperature Sensor Datasheet," 2018. [Online]. Available: https://www.bosch-sensortec.com/products/environmental-sensors/humidity-sensors-bme280/

[6] DFRobot, "SEN0366 Infrared Laser Distance Sensor Communication Protocol and Datasheet," 2020. [Online]. Available: https://dfimg.dfrobot.com/nobody/wiki/068db268ba37a41067c1b17607932139.pdf

[7] Sitronix Semiconductor, "SH1106 OLED/PLED Controller Datasheet," Pololu Corporation, 2012. [Online]. Available: https://www.pololu.com/file/0J1813/SH1106.pdf

[8] Texas Instruments. *LM2596 SIMPLE SWITCHER® Power Converter 150-kHz 3-A Step-Down Voltage Regulator* (Rev. G). LM2596 Series Datasheet, SNVS124G, Nov. 1999 (revised Mar. 2023). [Online]. Available: https://www.ti.com/lit/ds/symlink/lm2596.pdf

[9] STMicroelectronics, *LD1117S33/TR Low-Dropout Positive Voltage Regulator (Fixed 3.3 V) *,DocID 2572 Rev. 38, Jan. 2025. [Online]. Available: https://www.st.com/resource/en/datasheet/ld1117.pdf](https://www.st.com/resource/en/datasheet/ld1117.pdf)

[10] IEEE *Code of Ethics*, IEEE, 2020. [Online] Available: http://www.ieee.org/about/corporate/governance/p7-8.html

[11] Espressif Systems, "ESP32-S3-WROOM-1 and ESP32-S3-WROOM-1U Datasheet," Oct. 28, 2021. [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32-s3-wroom-1_wroom-1u_datasheet_en.pdf

Appendix A Requirement and Verification Tables

Requirement	Verification	Verification status (Y or N)
• The battery must be able to supply a constant 9V DC supply	• Measure the voltage using an oscilloscope, verifying that the battery is supplying 9±5% V	Y
• The 9-5 converter must be able to successfully step down the DC voltage from 9±5% to 5±5% volts to be supplied to the wind sensor • Converter can operate within 0-1A output current • Efficiency should be higher than 80%	• Measure the voltage at the output using an oscilloscope, verifying it is 5±5% V • Connect a multimeter to the output of the converter in series and measure output current • Calculate efficiency with (Vout)(Iout)/(Vin)(Iin) equation	Y Y Y
• The 5-3.3 converter must be able to successfully step down the DC voltage from 5±5% to 3.3±5% V to be supplied to the following components: 1. Button 2. Weather Sensor 3. Time-of-Flight Sensor 4. Microcontroller • Converter can operate within 0-1A output current • Efficiency should be higher than 80%	• Measure the voltage at the output using an oscilloscope, verifying it is 3.3±5% V • Connect a multimeter to the output of the converter in series and measure output current • Calculate efficiency with (Vout)(Iout)/(Vin)(Iin) equation	Y Y N

Table 5: Power Subsystem Requirements and Verification

Requirement	Verification	Verification status (Y or N)
• The microcontroller successfully receives data from the peripheral sensors upon push button triggering. Data will be received from: 1. Weather Sensors 2. Time-of-Flight Sensor	- Ensure correct output is displayed by comparing output data from the serial monitor to the OLED display - Ensure data is received after push button is pressed	Y

Table 6: Controller Subsystem Requirements and Verification

Requirement	Verification	Verification

		status (Y or N)
• The SEN0366 sensor is safely powered and accurately measures the distance between the sensor and the target with ±1 mm accuracy as specified on the datasheet	• Verify data is being read from the sensor by observing on the OLED display • Validate distance measured by comparing a physical measurement using a tape measure between the rangefinder and the target. Verify that the distance reported on the rangefinder is within ±1 mm of the tape measure distance	Y
• The BME 280 temperature and humitidy sensor measures within ±1 degrees Celsius and ±5% when compared to a thermostat when indoors and the weather app when outdoors	• Verify data is being read from the sensor by observing on the OLED display • Ensure temperature measurement from the sensor is within ±1 degrees Celsius and pressure is within ±5% from the values reported on the weather app	Y
• The wind speed sensor measures a wind speed within ±5% of the wind speed reported by a commercial wind meter or sensor	• Ensure wind measurements are accurate within a ±5% error when compared to the Kestrel 1000 Wind Meter	N

Table 7: Peripheral Devices Subsystem Requirements and Verification

Requirement	Verification	Verification status (Y or N)
• The OLED successfully displays the correct data from the microcontroller conveniently to the user	• Ensure correct output is displayed by comparing output data from the serial monitor to the OLED display • Ensure that the data is organized in a way the user can easily find information	Y

Table 8: Display Subsystem Requirements and Verification