Autonomous Sailboat

Final Report

Austin Glass

Michael Sutanto

Devansh Damani

Team 24

TA: Koushik Udayachandran

Professor: Dr. Arne Fliflet

Abstract

An autonomous sailboat is one that is able to veer itself on an optimal course to get from a certain start to a desired destination by manipulating inputted telemetric data from sensors onboard the boat to control the sails and rudder. Sailing conditions for the sailboat can vary depending on external factors such as weather. In the case of non-ideal weather conditions, sailboats risk damage to not just the boat's frame but the electrical components too. The goal of this project was to create a sailboat capable of autonomous sailing, with additional capabilities of a manual remote-controlled commandeering mode whenever required. A third mode was also defined- the 'Return-To-Base' mode which, when on autonomous sailing mode, forces the boat to interrupt its course and return to a predefined 'base' coordinate**.**

Three high level requirements of the project were set to ensure a successful functionality of the boat: enabling the boat to maintain an optimal course autonomously, successful telemetric interpretation within the microprocessor, and the ability for users to switch between the three modes of sailing at-will via the remote controller. The remote controller aids in not just switching between modes, but serves as the main form of control when in manual control mode. The triple-mode functionality of the remote control helps amateur sailers understand the workings of the sails and rudder through observing directional changes based on changes to the sails and rudders' angles. Furthermore, it also helps in the case of extreme weather events or obstacles in its path hindering the boat's ability for safe autonomous travel. A user can invoke the manual control or Return-To-Base mode when necessary to ensure safety of the boat. Telemetric data is also outputted to a user's computer on the shore for them to assess the boat's safety and possible mode switches.

Table of Contents

1. Introduction

This boat's main purpose is to create an autonomously sailing boat which would compute the optimal path from a starting location to a desired destination. Through a remote controller (RC), users are able to switch between three possible modes: autonomous, manual and Return-To-Base (RTB). Autonomous mode is characterized by the autonomous sailing of the boat- enabling a hands-free user experience. Manual mode enables the user to use the RC remote's gimbals to control the rudder servo and tighten or loosen the sail to steer the boat in a desired direction. This enables users to not just understand and learn about the operation of a sailboat, but let the user take control of the boat whenever required (extreme wind, obstacles etc.). RTB mode is designed to be a failsafe measure to override the current operation of the boat regardless of its mode and have it automatically maneuver itself back to a 'home base'- the coordinates for which are set through the RC remote at the beginning of sailing.

The user only requires the RC controller and telemetry receiver to communicate with the boat- making usage simple. Telemetric data is sent to and from the boat via a transmitter on the boat and a receiver on a user's laptop which outputs telemetric data. Data is also outputted onto a user's laptop to aid in navigation and assessment of whether to switch modes.

1.1 Functional Overview

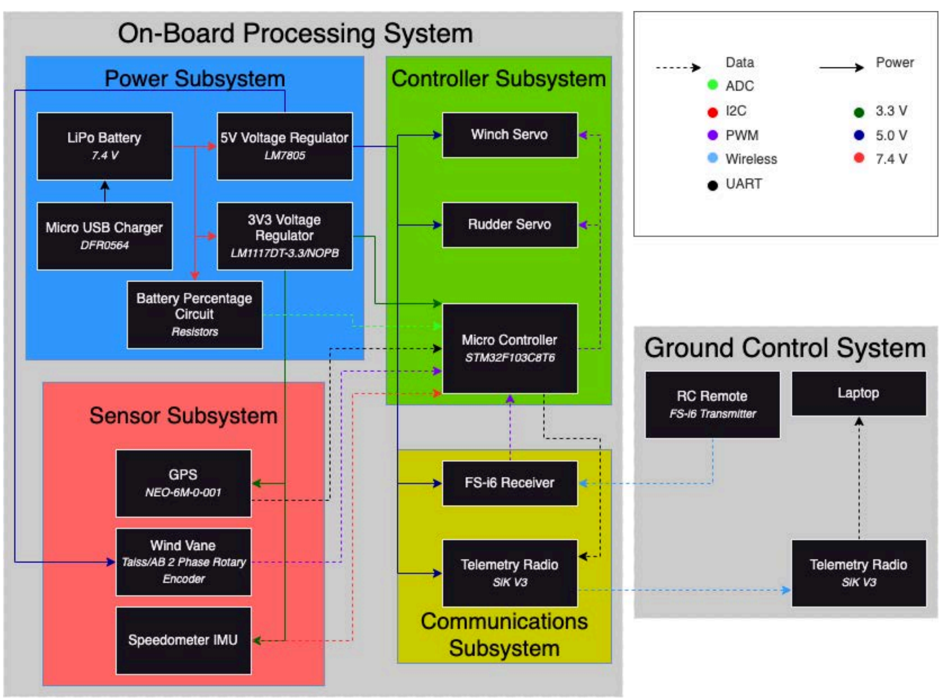


Figure 1: System Block Diagram

Power Subsystem: The power subsystem provides the circuit with the necessary voltage for it to operate safely and efficiently. The entire circuit is powered by a 7.4V lithium-ion battery and utilizes a 5V regulator and a 3.3V regulator to not over-power components. The 5V regulator provides 5V to the winch and rudder servos, the FS-i6 receiver, the telemetry radio and the wind vane. The 3.3V regulator provides 3.3V to the microcontroller, the GPS and the speedometer IMU. Furthermore, a micro-USB charger was implemented aiding in charging the battery whenever its voltage was too low.

Sensor Subsystem: The sensor subsystem acts as the eyes and ears of the boat. The GPS provides the boat's current latitude and longitude coordinates through a satellite connection. The wind vane tells the boat the direction of wind- which is then used to adjust the sails accordingly, and the speedometer IMU is used to measure the current compass heading of the boat in relation to the destination's compass heading and is used to steer the boat.

Controller Subsystem: The controller subsystem focuses on the microcontroller and how it affects the boat's steering. When in autonomous mode, data from all the sensors and the battery percentage calculator is fed into the microcontroller where it controls the winch and rudder servos. When in manual mode, the microcontroller takes in data from the RC remote via the FS-i6 transmitter and the FS-i6 receiver and adjusts the winch and rudder servos according to the movements of the gimbals on the controller.

Communications Subsystem: The communications subsystem consists of the FS-i6 receiver and the telemetry radio. The FS-i6 receiver takes in data from the RC remote and aids in switching between the three modes, and in controlling the winch and rudder servos when in manual mode. The telemetry radio aids in taking in data from the microprocessor and outputting it on a user's laptop to verify telemetric data.

Ground Control Subsystem: The ground control subsystem focuses on the operations of the user on shore. It sends the controls from the RC remote to the boat when in manual mode, and it outputs data onto the user's laptop from the boat. The data includes GPS coordinates, wind and winch angles, rudder angle, the current and desired headings, the roll and the battery percentage.

2. Design

2.1 Power System Design

2.1.1 Battery, Charger, and Switch

2.1.1.1 Battery

The battery has been increased from 1S2P Li-Po 3.7V battery to a 2S2P 7.4V Li-Ion battery [Appendix C figure C.3.1].

This step up in voltage allows for a more power-efficient design, allowing the circuit to step down voltage to 5V rather than stepping up to 5V, which requires a larger power draw.

As a battery discharges, the voltage it outputs drops slightly. The same is true in the opposite direction as the battery charges. The safe usable range of a 7.4V Li-Ion is 6.4V to 8.4V. Within this range, the battery is guaranteed to output enough power, and be ensured to not explode due to overcharging.

To optimize battery health, we define the lowest range to be 6.8V, and the highest range to be 8.0V. This is 60% of the battery's safe operating range, and ensures that under long term usage, the battery will reduce its life as little as possible.

2.1.1.2 Micro-USB Charger

Micro-USB Charger [Appendix C figure C.3.2] is capable of powering the circuit at approximately 20% per hour - or a rate of 0.25V per hour. It takes a 5V 1A connection from a Micro-USB port.

In our use case, we had a Micro-USB cable connected to a wall AC-DC adapter, which provides the 5V 1A DC connection required by the battery charging component.

2.1.1.3 Switch

A SPDT Switch [Appendix C figure C.3.3] was added in front of the battery, separating both the charger module and the circuit. This switch allows for the battery to become completely disconnected from the circuit, stopping it from discharging without the need of physically disconnecting the battery. When the PCB is physically installed in the boat’s inside, it makes battery removal tedious. It also allows the battery and charger to be disconnected from the overall circuit while being powered, which reduces any potential risk of unwanted current entering the circuit and damaging components.

The switch allows for 3 states in the circuit. When flipped down, the battery is now connected to the charging circuit. If there is power coming in through the Micro-USB connection on the charger module, the battery begins to charge. If there is no power coming through the Micro-USB connection, then the battery remains disconnected, and no power is drawn. When flipped up, the battery disconnects from the power circuit, and begins to power the board.

2.1.2 Voltage Regulation

The battery is only capable of outputting from 6.8V to 8.0V, and only outputs a constant voltage depending on its level of charge. All of our components use either 3.3V or 5V, which the battery is not capable of outputting. So, two types of voltage regulators are implemented to deliver power to the components of the device.

2.1.2.1 Voltage Regulator - 5V

Originally, the design of our board used an external 5V module from Makerfocus, which was capable of outputting 5V at a rate of 2A. This component was designed for stepping up voltage, which was not defined in its limited documentation. Instead, through our testing, we found it would output 5V when connected to a voltages less than 5V, and increase linearly when connected to voltages greater than 5V.

To resolve this problem, we found a similarly rated Breadboard 5V Regulator from Texas Instruments [Appendix C figure C.3.4], which is capable of outputting 1.5A. This component outputs a constant voltage no matter the input, as long as it is greater than 5V. It had the same pinout for the Battery, Ground, and Output pin as the one from Makerfocus, which enabled us to use the same pins on our PCB without changing the overall design of the PCB.

As defined in its documentation, the regulator is recommended to be connected to a 47nF capacitor, which we did not have room for on the PCB. It had excess pin room left over after being soldered into the board, which we were able to fit a Breadboard capacitor on some loose female to female connectors. This made sure the current output was safe from potential jumps in current.

Our Wind Vane Encoder, Servos, Telemetry Receiver, RC Controller Receiver, and Micro-USB test port (if used) all use the 5V output. As reported in the R/V section, these components all draw less than the regulator provides.

2.1.2.2 Voltage Regulator - 3V

The 3V regulator [Appendix C figure C.3.5] is capable of outputting 800mA to the circuit. It is used to power the Microprocessor, GPS, IMU, BOOT Pins, and Reset Button. It is also connected to an array of capacitors suggested by its documentation, which secures the components from potential jumps in current.

2.1.3 Battery Percentage Indicator

The battery percentage indicator is an array of 3 resistors [Appendix C figure C.3.6] in parallel that split the voltage into a safe readable range by the Microcontroller. The battery outputs 6.8 to 8.0V, and the microprocessor can only receive a maximum voltage of 3.3V. So, the voltage is split by 3, allowing it to receive a reading range of 2.27V to 2.67V. To reduce current draw, the resistors are 100k ohms, meaning 74 micro Amps at any point in time, minimally impacting the battery.

The port reads in values from 414 to 487, depending on the battery's voltage. It's multiplied by a constant (182.55) defined by the ADC port on the microcontroller, and then multiplied by 3 to be recovered from the split voltage to the full voltage.

2.2 Sensor Output Design

2.2.1 IMU Compass

The Adafruit BNO055 IMU [Appendix C figure C.4.2] is utilized as a compass to determine current heading Φ and heeling angle ϕ_{xyz}. To get these values the Adafruit_BNO055[28] and Adafruit_Sensor[29] libraries were used.
The heeling angle ϕ_{xyz} is approximated as the roll of the sailboat that is calculated like this:

$$\phi_{xyz} = atan2(a_x, a_y) * 57.2957$$

$$a_x = acceleration\ in\ x\ axis$$

$$a_y = acceleration\ in\ y\ axis$$

The current heading Φ is a direction offset from the magnetic north. This magnetic north is calculated using vector mathematics. These calculations can be seen below:

$$\text{origin} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

$$\text{m} = \text{Magnetic vector from sensor}$$

$$\text{a} = \text{Accelerometer vector from sensor}$$

$$E = \text{m} \times \text{a}$$

$$E = \frac{E}{\|E\|}$$

$$N = \text{a} \times E$$

$$N = \frac{N}{\|N\|}$$

$$\Phi = \text{atan2}(E \cdot \text{origin}, N \cdot \text{origin}) \times \frac{180}{\pi}$$

The C++ implementation of these calculations using the libraries stated above could be viewed in Appendix D Figure 1.

2.2.2 Wind Vane Encoder

The Wind Vane Encoder [Appendix C figure C.4.1] is a combination of a 3D printed wind vane and a 2-phase rotary encoder. This sensor is used to obtain the apparent wind angle θ_w , the. To get this value the RotaryEncoder library[30] is used.

The function getPosition of this library outputs a value δ that increases as the encoder is turned in the direction it was first turned until it reaches overflow. However, through testing, it was found that a full 360 degree rotation is a position value of 1190. Furthermore, we can obtain the direction of the first rotation ρ through the library function getDirection.

Both these values are used in the mapping function below to obtain the apparent wind angle θ_w :

$$\theta_w = \begin{cases} 360 - \mathrm{map}\left(\left(\delta \mod 1190\right), 0, 1190, 0, 360\right), & \text{if } \rho < 0 \\ \mathrm{map}\left(\left(\delta \mod 1190\right), 0, 1190, 0, 360\right), & \text{else} \end{cases}$$

In this equation:

- θ_w is the output angle.
- δ is the input value.
- ρ is a condition.
- $\mathrm{map}(x, a, b, c, d)$ is a function that maps the value x from the range $[a, b]$ to the range $[c, d]$.

In terms of implementation, a digital pin interrupt is assigned to each direction the encoder is turning, which triggers a callback function checkPosition(), which updates the current position value of the encoder object in the program.

The C++ implementation of the interrupts and the mapping using the libraries stated above could be viewed in Appendix D Figure 2.

2.3 Communication and Servo Control Design

2.3.1 RC Receiver Mapping

The RC transmitter that is used is the FlySky FS-i6 Remote [Appendix C figures C.5.1 & C.5.3]. In our implementation, we use 4 of 6 of the available channels for different applications. The mapping of the used channels can be seen in Table 1 below:

Channel	Remote	Mapping	Use
1	Right Gimbal ←, →	Winch Servo	←: Tighten Sail →: Loosen Sail
3	Left Gimbal ↑, ↓	Rudder Servo	↑: Rudder → Starboard ↓: Rudder → Port side
5	SwC	Sailing Mode	1: Manual 2: Autonomous 3: Return to Base
6	SwD	Set Base Position	1 → 2: Sets Base Position to Current GPS Location

Table 1: Remote Channel Mappings

The right gimbal does not maintain its position after movement, thus we decided that it is suitable for controlling the winch servo because the movement of the sails could be easily visually confirmed. In contrast, the left gimbal maintains its position, which is suitable for controlling the rudder as it allows the operator to have a decent estimate of the rudder's current position inside water.

The SwC and SwD switches were chosen due to them having an appropriate number of modes for the number of operations that are needed.

2.3.2 RC Servo Pulse Width Mapping

The gimbal channels of the receiver outputs in the form of PWM signals. The output signal ranges between a 2.3%-11.9% duty cycle for a 50Hz waveform which is around a 450-2050 µs Pulse Width time range.

The servo's [Appendix C figures C.6.1-4] PWM range does not don't have a direct mapping from this receiver. The servos have a 5%-10% duty cycle. For the winch servo which expects a waveform frequency of 40Hz the time range decided is 1205-1961 µs. For the rudder servo which expects a waveform frequency of 50Hz the time range decided is 1400-1611 µs.
The movement of these servos are limited by the physical constraints of the physical design, and thus the servo PWM ranges are calibrated to match the desired behavior of the servos. These PwM ranges and desired movement ranges could be seen in Table 2 below:

Channel	Use	Pulse Width Range Receiver (µs)	Pulse Width Range Servo (µs)	Movement Range (°)	True Servo Angle Range(°)
1	Winch Servo	950 - 2050	1205 - 1916	-45 - 45	37,165
3	Rudder Servo	1000 - 2000	1400 - 1611	0 - 90	65,110

Table 2: Receiver and Servo PWM Ranges

In the software implementation, we obtain the receiver pulse width by attaching a digital pin interrupt to each receiver channel. Each interrupt calls a callback function when there is a change in the signal. This allows us to get the pulse width by subtracting the interrupt time with the previous interrupt time which gives the pulse width in the time domain. This implementation could be viewed in Appendix D Figures 3 and 4.

2.3.3 Telemetry Information Relay

In this project, only a single direction communication for the radio is needed which is from the boat to the receiver. The radio [Appendix C figure C.5.2 & C.5.4] transfers data through serial communication; Thus the results could be viewed through a simple serial monitor from the laptop that has the telemetry receiver module connected via USB.

The data transferred includes the operation mode, GPS location of the boat, apparent wind angle, current winch and rudder angle, desired heading, current heading, heeling angle, and GPS location of the base. All of these are viewed through a serial terminal as seen below.

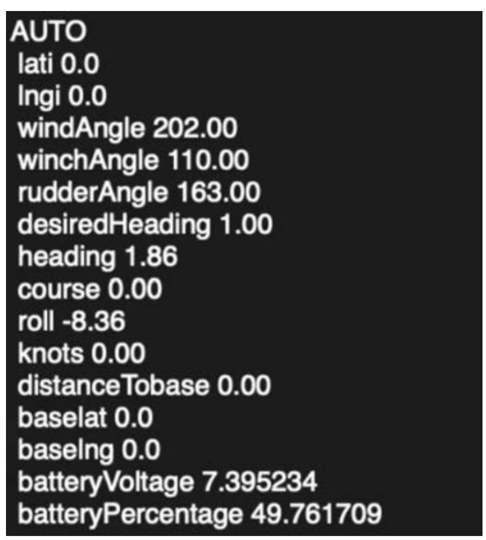


Figure 2: Telemetry Serial Terminal

The implementation of this print function could be seen in Appendix D Figure 5.

2.4 Control Algorithm Design

2.4.1 Sail Winch Control

The sail winch angle is controlled based on the apparent wind angle measured by the wind vane encoder [Appendix C figure C.4.3] and the measured heeling angle of the boat is. Depending on the heeling angle, there are two states the sail winch control program is in which are the "Safe State" and "Dangerous State". As detailed in Fig 3, when the absolute value of the heeling angle is > 15 degrees, the control is moved to the "Danger State", whereas when the heeling angle is < 15 degrees, the control moves to "Safe State". The Sail Winch Control programs are different for each state.

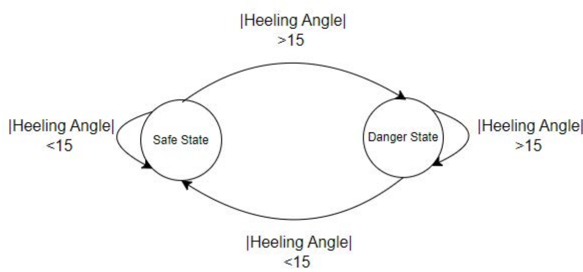


Fig 3: Sail Winch Control State Diagram

2.4.1a Safe State Control

When operating in the safe state the sail chord angle determined by the sail winch servo is set based on the value in the lookup table of optimum values in Table 3 corresponding to the Apparent Wind Angle obtained by the wind vane encoder.

Apparent Wind Angle	Point of Sail	Sail Angle	Winch Servo True Angle
$0 \leq \theta \leq 45 \| \| 315 \leq \theta \leq 360$	No-Go Zone	0°	65°
$45 \leq \theta \leq 75$	Close-Hauled	15°	65°
$75 \leq \theta \leq 105$	Beam Reach	-45°	69°
$105 \leq \theta \leq 135$	Broad Reach	-60°	74°
$135 \leq \theta \leq 225$	Running	±90°	110°
$225 \leq \theta \leq 255$	Broad-Reach	60°	69°
$255 \leq \theta \leq 285$	Beam Reach	45°	65°
$285 \leq \theta \leq 315$	Close-Hauled	15°	65°

Table 3: Optimal Sail Chord Angle for Different Wind Angles

The software implementation of this lookup table could be found in Appendix D Figure 6.

2.4.1b Danger State Control

When operating in the Danger State, we want to recover the absolute heeling angle to <15 degrees as fast as possible. This can be done by reducing the heeling force caused by the wind by "letting out sails" which means increasing the sail chord angle which decreases the angle of attack in regards to the wind. We leave this state when the boat is no longer in an absolute heeling angle of > 15 degrees . The software implementation of this control effort could be found in Appendix D Figure 7.

2.4.2 Rudder Control

The original design had the rudder controlled using a PI controller that takes a hyperparameter weighted sum of deviation angle and boat sideways drift speed as an input. Both of these values need to be counteracted by an appropriate rudder angle so the output of the PI controller is a rudder angle. The operation of this control system can be seen below in Fig 4.

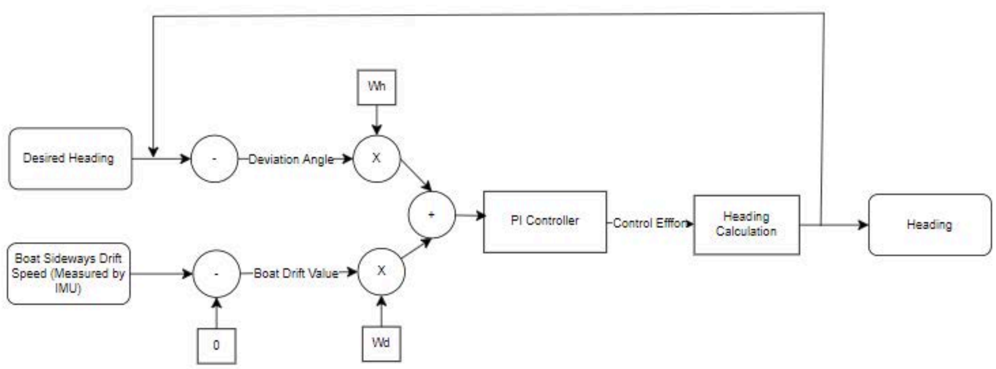


Fig 4: Planned Block Diagram for Rudder Control System

However due to time constraints, we were only able to get the rudder control working for the component with desired heading which can be seen in Figure 5 below:

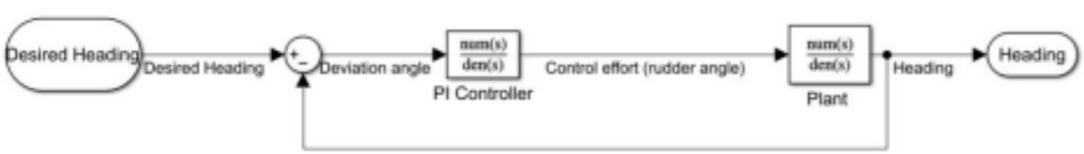


Fig 5: Final Block Diagram for Rudder Control System

The software implementation of this control algorithm and setting the parameters could be found in Appendix D Figure 7.

3. Requirements and Verification

3.1 Power Subsystem Verification

The total power draw, as defined by each component, is as follows:

Controller Subsystem (Total = 892mA)

- ❖ Microcontroller (STM32F103C8)[13] → 150 mA total current into VDD power lines
- ❖ Sail Winch Servo (Joysway 880545)[14]→ 0.15A no load * 0.2 + 0.4A working load *0.8 = 0.35A = 350mA
- ❖ Rudder Servo (Joysway 881504)[15]→ 0.16A no load * 0.2 + 0.45A working load *0.8 = 0.392A = 392mA

Sensor Subsystem (Total = 99mA)

- ❖ GPS(HiLetgo NEO6MV2)[10] → 6.7mA
- ❖ Wind Vane Encoder[19] → 80mA
- ❖ Speedometer IMU(Adafruit BNO055)[11]→12.3mA

Communication Subsystem (Total = 170mA)

- ❖ SiK V3 Telemetry Radio[12] → 100mA
- ❖ FS-i6 Receiver→70 mA[17] (Found from forums since datasheet can't be found)

Power Subsystem (Total = 5.125 mA)

- ❖ 5V Charge Discharge Integrated Module [21] → 0.1mA
- ❖ 3.3V Voltage Regulator [22] → 5mA
- ❖ Battery Indicator Circuit [1] → 7.4V/300kOhm = 24.7 microAmps

Total Current Draw = 1.006 Amperes

Note: The 5V and 3.3V regulators are capable of outputting 1500mA and 800mA, respectively. Accounting for the components connected to each regulator, 5V components require 910mA (400mA + 160mA + 100mA + 70 mA + 80mA), and 3.3V components require 209.11mA (12.3mA + 0.11mA + 40mA + 6.7mA + 150mA). At their maximum capacities, they would not be drawing any more power than the regulators are capable of providing.

Through our testing, we found that every component remained powered the entire time, with no components dropping. However, our rudder servo shorted during testing. Once testing began with a new rudder servo with a higher power draw, we found that at its peak usage, it would temporarily power off the telemetry.

So, under ideal conditions with the correct components we can safely say that the circuit is capable of providing enough power.

Figures 5, 7 - Correct Voltage Output 3V and 5V Regulators

We further tested each regulator, proving that they output 3.3V and 5V respectively. Upon checking numerous times over the course of the project, these outputs remained constant.

Figures 8, 9- Voltage of Battery Compared to Telemetry Output Reading

We also compared the true rating of the battery to the recorded output by the telemetry radio. The battery readings we estimated were consistently within 0.05V of the true reading, which is approximately 4% of the battery's real percentage. Through our testing we also confirmed that after 1 hour of charging, the battery increased by 0.25V, which would charge the battery completely within 5 hours. We did not perform any drainage tests, as we never made a significant dent in the battery through hours of normal normal testing to warrant it needing further testing.

3.2 Sensor Subsystem Verification

The verification of the different sensors in the sensor subsystem relies on the demonstration video [https://youtu.be/p3-AgPk7rfw]. Figure 10 below shows the path of how the boat was manually turned by our hand over time in the video. In this plot we can see that there is a big change in boat turn near the end of the chart, this is when the boat is being turned counterclockwise in the demo.

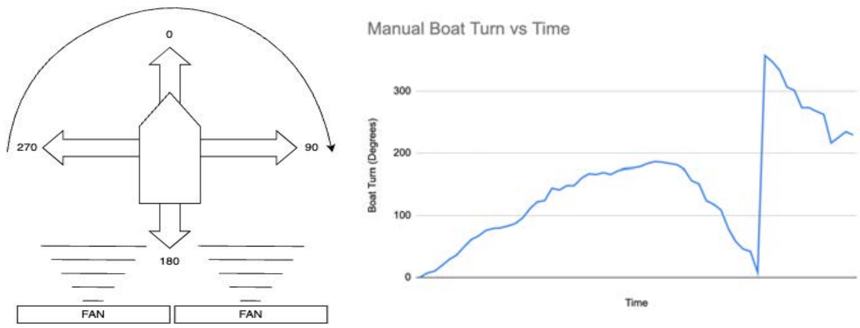


Fig 10: Boat Manual Turn Direction in Video

3.2.1 IMU Heading Verification

The IMU successfully captured the changes in heading throughout the demonstration [Reference Demo] according to the Requirement 1 at the Sensor Requirement Verification table in Appendix E, Table E.2.

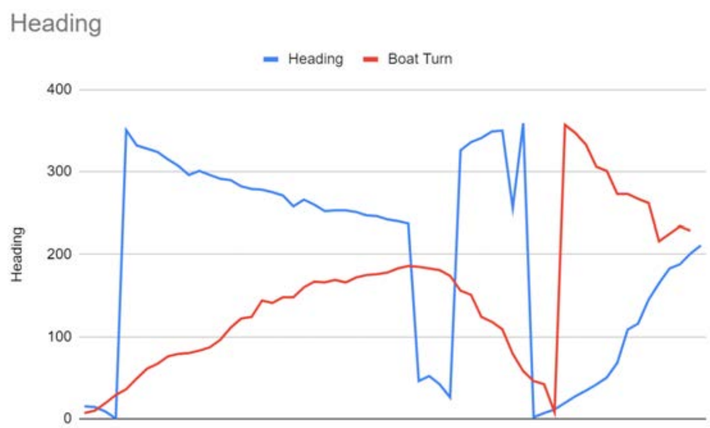


Figure 11: Heading vs Time Plot

The figure above shows a plot of the heading data over time, and the boat turn plot is added as comparison. The discrepancy between the angle from the Heading reading and the boat turn is because the heading is the deviation from the magnetic north, whereas the boat turn is based on the reference frame shown at Figure 11. However, the plot shows that the heading readings change at the relatively same rate as the boat turn, which shows that the IMU reading does show accurate data of the change in boat heading over time. There are exceptions such as around the middle of the plot where there is a dip in the IMU reading where the boat turn forms a peak, and this is due to the sudden turning motion which can be viewed in the demonstration video.

3.2.3 Wind Vane Verification

The Wind vane encoder successfully captured the correct apparent wind angle throughout the demonstration according to requirement 3 at the Sensor Requirement Verification table in Appendix E, Table 3.2.

Figure 12: Apparent Wind Angle vs Time Plot

The figure above shows a plot of the apparent wind angle data over time, and the boat turn plot is added as comparison. This plot shows very well that the wind angle captured by the wind vane matches the turning of the boat. Some points in the graph to look at are the start and the point where there is a jump in the boat turn graph. During these points, the wind angle is approximately 150 degrees. Referencing Table 3, this is the running angle range which indicates that the back side of the boat is aligned with the wind direction, which correctly describes the 0 degrees at the boat turn plot based on Fig 12.

3.2.3 GPS

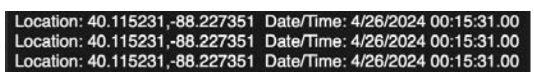

Location: 40.115231,-88.227351 Date/Time: 4/26/2024 00:15:31.00
Location: 40.115231,-88.227351 Date/Time: 4/26/2024 00:15:31.00
Location: 40.115231,-88.227351 Date/Time: 4/26/2024 00:15:31.00

Figure 13 - GPS connection established and GPS reading

We were able to test and verify the GPS Sensor separately and run a singular test, but were ultimately unable to integrate it into our circuit. Thus, we were not able to fulfill the verification requirement 2 from the Sensor Requirement Verification table in Appendix E, Table E.2. The problem that arose was that the GPS took very long to obtain a longitude and latitude reading, which caused the inability to present a demonstration with it. However, in our singular test we were able to establish a connection, and get some GPS readings out, as shown in figure 13.

3.3 Control Subsystem Verification

To talk about the control subsystem verification we will reference the Control System Requirement and Verification table in Appendix E, Table E.1.

3.1 Control System Successful Verification

By completing the autonomous mode demonstration video [Full: https://youtu.be/p3-AgPk7rfw, Rudder Only: https://youtu.be/VpdvSw39Gvc] and the manual mode demonstration video [https://youtu.be/6w3YkXUtZkk], we were able to verify that the requirements 1, 2, 3, 4, 5, and

9 from the requirement verification table were successfully verified. Requirements 1, 2, and 3 could be verified quantitatively by watching the videos.

Requirement 4 which talks about correctly adjusting the sail angle according to apparent wind angle can be verified qualitatively through the video and quantitatively through the Figure below which plots both apparent wind angle and the winch angle over time. The mapping of this plotted winch angle, which represents the true servo angle, to the actual optimal winch angle could be seen in the Table 3: Optimal Sail Chord Angle for Different Wind Angles above.

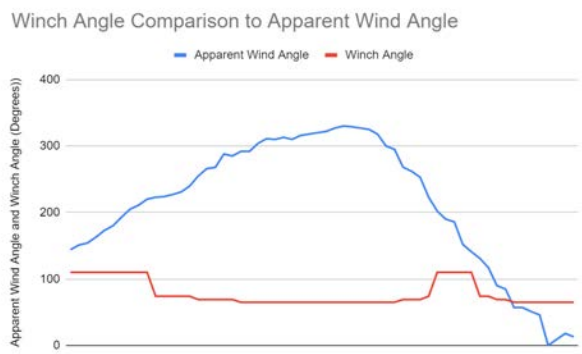


Figure 14: Winch Angle Comparison to Apparent Wind Angle

Requirement 5 which talks about performing rudder angle adjustments based on deviation of current compass angle and the destination compass heading is partially fulfilled. Based on the figure below which plots the deviation of the compass heading and the rudder angle changes, we see that the rudder angle puts in a control effort when the deviation is in a small range of the desired compass heading. However, when it passes a certain angle of deviation, it stays the same because it has done the maximum possible correction effort for the rudder motor. This makes sense as unlike the demonstration video where the boat is turned manually, the boat should not deviate more than 8 degrees from its desired course in either direction.

Figure 15: Rudder Angle Comparison to Deviation from Desired Heading

Finally, the requirement 9 which specifies having to respond to the RC remote in <25ms is fulfilled and can be verified by the almost instantaneous movement of the servos when controlled by the RC remote in manual mode in the manual mode demonstration video [reference video]

3.2 Control System Failed Verification

Due to not being able to get the GPS sensor working consistently and the lack of a water test, the control system requirements number 6,7,8,10 from the Control System Requirement and Verification table in Appendix E - Table E.1 were not fulfilled and could not be verified.

3.4 Communication Subsystem Verification

Two major requirements for the communications subsystem had been defined:

1. The data from the microcontroller has to be properly transmitted with latency of less than 60 ms through the telemetry radio.
2. Turning on the autonomous functionality should not be able to be turned on before 5s after the remote is turned on to ensure sensors calibrate.

To verify the correct functionality of the subsystem, we can verify if the FS-i6 and the telemetry radio are functioning and able to send data to/from the RC remote and have it successfully manipulate the boat in the desired way. This can be seen in the demonstration video [https://youtu.be/p3-AgPk7rfw], proving that we met these two requirements and successfully implemented the communication subsystem. The data is updated every second, but is sent in real time with the exact data on the board at the time of transfer. It can be seen that the current state is mirrored in each update of the data.

3.5 Ground Control Subsystem Verification

Two major requirements for the ground control subsystem had been defined:

1. The FS-I6 transmitter should transmit signals to the appropriate channel to move the appropriate servo in manual mode.
2. Laptop should be able to view data from telemetry radio.

The first requirement has been met, as can be seen in this video demonstrating the RC Controller's immediate response to input [https://youtu.be/6w3YkXUtZkk]. The second requirement can also be seen in our prior video demonstrating autonomous mode, where the data shown in the gray box is the real time output of the laptop [https://youtu.be/p3-AgPk7rfw]. The transmission of signals from the RC remote to move the appropriate servos while in manual mode is demonstrated in the videos, and you can see a clear output of telemetric data proving that we met these two requirements and successfully implemented the ground control subsystem.

4. Costs

4.1 Labor Cost

The average starting salary of a graduate from the University of Illinois Urbana-Champaign in the ECE program is $87,276.00 [2]. Since this salary is for a 52-week year, at 40 hours a week, this salary comes out to approximately $41.95/hour.

Under the assumption that 3 weekly hours of work should be dedicated to each credit hour a class is worth, each team member should be working 12 hours a week on the project. So, for 3 team members, that would come out to 36 hours per week worked. The TA also has weekly meetings with us, so for the hour meeting, and an estimated extra hour for checking emails periodically, we would be at 38. Professor Fliflet also dedicates some time of his week to our project, so our safest estimate would be 40 hours per week that would need to be compensated.

Each semester is 16 weeks, but we started work on the project on the 4th week, so there will be 12 weeks total that will be compensated. In total, $20,140.61 needs to be allocated towards the labor costs of this project.

4.2 Facility Cost

The cost of using a machine shop at the Grainger College of Engineering is approximately $60 an hour [3]. Assuming we spend 5 hours a week using these design facilities, for half the time of the project, that would be about 30 hours of the space needed. This comes out to $1,800 for use of the space.

4.3 Materials Cost

We already have many of these resources available to us, but most will need to be reordered. However, we will account for the unit costs, in today's value, of all the resources we will be needing in total. The exact breakdown and sources of components can be found in Appendix A. Referencing Appendix A, the total material cost for the entire project is $300.31.

4.4 Total Cost

The labor cost to complete this project is $20,140.61, the cost of the facilities is $1,800, and the material cost is $300.31. In total, this project would need to be funded $22,240.92.

To be safe, and account for any additional costs that may be incurred over the course of the project, whether that be damaged / lost parts, overtime, or any other circumstance, we would advise funding of $25,000 for the project in total.

5. Conclusion

5.1 Accomplishments

All of the high-level requirements we set out to accomplish were achieved. Our demonstration and subsequent video prove that the boat switches between all modes through the RC and a user can tighten/loosen the sails and steer the rudder from the RC itself. Our PCB ran as expected, and our component implementation worked as expected. Furthermore, we met most of our subsystems' RV Table requirements (with the notable exception of the GPS sensor). This is proved through our demonstration videos- both the autonomous and manual control videos highlight this.

5.2 Uncertainties

We had two major setbacks which hindered our final demonstration. Our primary setback was the GPS sensor which had multiple issues associated with it. The first obstacle was the shielding of the ECE Building prevented the GPS sensor from communicating with satellites- which hindered in-lab testing of the GPS along with other sensors and subsystems. Moreover, once we went out, the GPS took an excessive amount of time to calibrate- making testing nearly impossible. It was due to these reasons that the GPS sensor was not effectively integrated into our system. Possible solutions for this would be to conduct more research into a better sensor with a higher calibration rate and faster connectivity.

Our second major setback was our rudder servo shorting soon before the final demonstration, due to unknown causes. Our final product consisted of a temporary rudder to highlight the rudder functionality, however the temporary rudder servo did not fit on our boat- leading it to be shown externally. This is shown in our manual mode demonstration video, where you can clearly see a temporary, much larger rudder sitting on the frame of the boat. However, the rudder did perform the way in which we wanted. Solutions for this include reordering a new rudder servo that fits into the boat's frame, time permitting.

5.3 Boat Operation and Demo Safety

This project is a follow-up project to the Autonomous Sailboat senior design project done in 2022, furthermore it is definitely not the first Autonomous Sailboat project published throughout the internet, this brings the concern of originality and accreditation. An ethics policy that will be heavily taken into consideration is Section 7.6 of the IEEE Code of Ethics I.5 states, "to seek, accept, and offer honest criticism of technical work… and to properly credit the contributions of others" [4]. We will ensure to credit and cite any resources from previous projects and online and/or offline resources we use.

Furthermore, through our goal of making a seamless system of dual-mode control, we aim to adhere with the first IEEE Code of Ethics “to accept responsibility in making decisions consistent with the safety, health, and welfare of the public, and to disclose promptly factors that might endanger the public or the environment;". By enabling a dual-mode capability, the project aligns with this ethical guideline by ensuring that users can take control in situations that may require human judgment or intervention, thereby protecting the public and the environment from potential harm.

Finally, one of the mission and goal of this project, which is to create a user-friendly and more affordable autonomous sailboat, strongly addresses the fifth IEEE code of ethics, "to improve the understanding of technology; its appropriate application, and potential consequences".

We will follow the Lab Safety guidelines while working on the boat in the Senior Design Lab rooms while testing our circuits, sensors and soldering. As per OSHA guidelines for welding, cutting and soldering, we will ensure to work around a fire extinguisher or places with a fire-plan, wear protective clothing and gear such as goggles and gloves [5].

Our ground control system will allow users to monitor sensor data such as GPS coordinates. The ethical concerns about that would be that the user’s GPS coordinates would also be recorded as the “home base”, which would pose a risk to the controller’s privacy. In order to ensure that there are no privacy violations, we will protect and not monitor user data and uphold the IEE code I.1; “to hold paramount, the safety, health, and welfare of the public... and to protect the privacy of others”.

When testing in the lab, we are able to stay safe by operating the boat in a static environment, where no retrieval is necessary. We will maintain safety standards by working in a clean, uncluttered area, using a stable surface (the boat on its stand resting on a desk or floor). When using external tools, such as a box fan (for simulating wind), we will clear the area of any hazards that could be dislodged and damaged by the movement of air.

5.4 Future Work

Along with fixing the GPS and the rudder, we propose four major additions to the project to ensure proper functionality. Two of them are at the design-level while the other two are from a testing perspective. On the design-level, we believe that adding an LED as a battery level indicator on the boat would be a valuable addition. This would alert users that the battery level is low and enable them to act accordingly. While the battery percentage is shown on the telemetry data on the user’s laptop, there is a risk of the user forgetting about it or having it get drowned out within the other data, a LED enables direct and easy verification. Another design-level implementation would be the complete waterproofing of the electrical components of the boat. We had not gotten around to that due to us not being able to test the boat in the water.

Water testing the boat is our first proposed test-based addition. We believe that water testing is the ultimate way to verify whether the boat works or not, and while we tried to simulate the conditions on land, it is highly beneficial for a water test to highlight functionality. Another test-based addition would be to test the RTB functionality. We were not able to properly test the RTB function properly, due to not doing a water test.

References

[1] "Arduino Battery Voltage Indicator." Autodesk Instructables, https://www.instructables.com/Arduino-Battery-Voltage-Indicator/.
[2] “Electrical Engineering.” *Electrical Engineering*, The Grainger College of Engineering, grainger.illinois.edu/academics/undergraduate/majors-and-minors/electrical-engineering
[3] “Machine Shop” The Grainger College of Engineering, https://physics.illinois.edu/research/facilities/machine-shop
[4] “IEEE code of ethics,” IEEE, Jun-2020. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 8-Feb-2024].
[5] ‘OSHA 1910.252- General requirements,” OSHA, March 26th, 2012. [Online]. Available: https://www.osha.gov/laws-regs/regulations/standardnumber/1910/1910.252 [Accessed: 21-Feb-2024]
[6] Baker, Riley, et al. 2022, Autonomous Sailboat.
[7] "The Physics of Sailing: What You Should Know." MySail Team News, MySail, https://news.mysail.team/learn-to-sail/the-physics-of-sailing-what-you-should-know/. Accessed 21 Feb. 2024.
[8] "Tacking and Gybing." Wellesley College, Wellesley College, http://web.wellesley.edu/Athletics/PE-old/sailing/sailing_tacking.html. Accessed 21 Feb. 2024.
[9] "LSM303DLHC Data Sheet." Adafruit, https://cdn-shop.adafruit.com/datasheets/LSM303DLHC.PDF. Accessed 4 Mar. 2024.
[10] "NEO-6 Series Data Sheet." U-blox, https://content.u-blox.com/sites/default/files/products/documents/NEO-6_DataSheet_%28GPS.G6-HW-09005%29.pdf. Accessed 4 Mar. 2024.
[11] "BNO055 Intelligent 9-axis Absolute Orientation Sensor." Adafruit, https://cdn-shop.adafruit.com/datasheets/BST_BNO055_DS000_12.pdf. Accessed 4 Mar. 2024.
[12] "Sik Telemetry Radio V3." HolyBro, https://holybro.com/products/sik-telemetry-radio-v3?variant=41562952270013. Accessed 4 Mar. 2024.
[13] "STM32F103C8 Datasheet." STMicroelectronics, https://www.st.com/resource/en/datasheet/stm32f103c8.pdf. Accessed 4 Mar. 2024.
[14] "Sail Winch Servo Set DF65." Radio Sailing, https://radiosailing.net/products/sail-winch-servo-set-df65. Accessed 4 Mar. 2024.
[15] "New Digital Metal Gear Rudder Servo." Radio Sailing, https://radiosailing.net/products/new-digital-metal-gear-rudder-servo. Accessed 4 Mar. 2024.
[16] "Lithium Polymer Discharge Characteristics." Battery University, https://batteryuniversity.com/article/arquus_lithium-polymer. Accessed 4 Mar. 2024.
[17] "Flysky FS-i6 Receiver Current Draw." RCGroups Forum, https://www.rcgroups.com/forums/showthread.php?2963479-Flysky-FS-i6-Receiver-Current-Draw. Accessed 4 Mar. 2024.

[18] "5mm LED Datasheet." Components101, https://components101.com/sites/default/files/component_datasheet/5mm-LED-Datasheet.pdf. Accessed 4 Mar. 2024.
[19] "LED 5MM RED DOMED CLR 631NM." RS Components, https://docs.rs-online.com/32f2/0900766b81059a65.pdf. Accessed 4 Mar. 2024.
[20] "L37A52-2-1-3WA3/13692679" Digi-Key Electronics, https://www.digikey.com/en/products/detail/dantona-industries/L37A52-2-1-3WA3/13692679. Accessed 4 Mar. 2024.
[21] “5V Charge Discharge Integrated Module” Amazon, https://www.amazon.com/MakerFocus-Discharge-Integrated-Charging-Protection/dp/B07PZT3ZW2
[22] “LM1117DT-3.3/NOPB” Digi-Key Electronics, https://www.digikey.com/en/products/detail/texas-instruments/LM1117DT-3-3-NOPB/363583
[23] “Reading Battery Voltage with the STM32’s ADC” Vivonomicon, https://vivonomicon.com/2019/10/15/reading-battery-voltage-with-the-stm32s-adc/
[24] “Beaufort Wind Scale”, Weather.gov, https://www.weather.gov/mfl/beaufort
[25] “Autonomous Sailboat Team 24 Demo - ECE 445 SPRING 2024”, YouTube, https://youtu.be/p3-AgPk7rfw
[26] “Autonomous Sailboat Team 24 Demo - ECE 445 SPRING 2024 [RUDDER SERVO ONLY]”, YouTube, https://youtu.be/VpdvSw39Gvc
[27] “Autonomous Sailboat Team 24 Demo [RC CONTROLLER MANUAL MODE ONLY] - ECE 445 SPRING 2024”, YouTube, https://youtu.be/6w3YkXUtZkk
[28] Adafruit, "Adafruit_BNO055," GitHub, 2017. [Online]. Available: https://github.com/adafruit/Adafruit_BNO055. Accessed: May 1, 2024.
[29] Adafruit, "Adafruit_Sensor," GitHub, 2014. [Online]. Available: https://github.com/adafruit/Adafruit_Sensor. Accessed: May 1, 2024.
[30] Mathertel, "RotaryEncoder," GitHub, 2014. [Online]. Available: https://github.com/mathertel/RotaryEncoder. Accessed: May 1, 2024.

Appendix A - Material Cost

Part	Manufacturer	Part Number	Qty	Unit Cost	Total Cost	Supplier
3V3 Regulator	Texas Instruments	LM1117IMPX-3.3/NOPB-ND	1	$1.81	$1.81	DIGIKEY
7.4 V Li-Ion Battery	Dantona Industries Inc.	L74A52-4-10-2WX	1	$18.96	$18.96	DIGIKEY
Speedometer IMU	Adafruit	BNO055	1	$34.95	$34.95	DIGIKEY
Telemetry Radio	Holybro	SiK V3 17012	1	$58.99	$58.99	HOLYBRO
Battery Charger	DFRobot	DFR0564	1	$5.00	$5.00	DFROBOT
5V Regulator	Texas Instruments	LM7805	1	$1.10	$1.10	Supply Center
BJT Transistor	Micro Commercial Co	2N3904	4	$0.44	$1.76	Supply Center
ARM 6 Pin Connector	Amphenol CS	-	1	$0.78	$0.78	Supply Center
SPDT Switch	ECE Supply Shop	240156511	1	$1.30	$1.30	Supply Center
1 kΩ Resistor	YAGEO	RC1206FR-101KL	2	$0.10	$0.20	Supply Center
4.7 kΩ Resistor	YAGEO	RC1206FR-104K7L	5	$0.10	$0.50	Supply Center
10 kΩ Resistor	YAGEO	RC1206FR-0710KL	1	$0.10	$0.10	Supply Center
20 kΩ Resistor	YAGEO	RC1206FR-0720KL	2	$0.10	$0.20	Supply Center
100 kΩ Resistor	YAGEO	RC1206FR-07100KL	5	$0.10	$0.50	Supply Center
4.7 µF Capacitor	Samsung Electro Mechanics	CL10A475KQ8NNWC	1	$0.10	$0.10	Supply Center
10 µF Capacitor	Samsung Electro Mechanics	CL21B106KPQNFNE	2	$0.27	$0.54	Supply Center
100 nF Capacitor	KEMET	C0603C104K8PAC7867	5	$0.18	$0.90	Supply Center
2 Pin Male Header	Molex	22284028	3	$0.28	$0.84	Supply Center
3 Pin Male header	Molex	22284036	2	$0.39	$0.78	Supply Center
Microcontroller	STMicroelecrronics	STM32F103C8T6	1	$6.11	$6.11	ESS
Rudder Servo	Joysway Hobby	881504	1	$13.95	$13.95	Reusing
Sail Winch Servo	Joysway Hobby	880545	1	$31.97	$31.97	Reusing
GPS Module	Hiletgo	GY-NEO6MV2	1	$8.99	$8.99	Reusing
Wind Vane Encoder	US Digital	MA3-P10-125-B	1	$63.00	$63.00	Reusing
RC Controller	FlySky	FS-i6 6CH	1	$46.98	$46.98	Reusing
RC Receiver	FlySky	FS-iA6	1	$0.00	$0.00	Reusing

Cost of Materials from External Sources: $119.71
Cost of Materials from Supply Center: $15.71
Cost of Reused Materials: $164.89
Total Material Cost: $300.31

Appendix B - KiCad Models

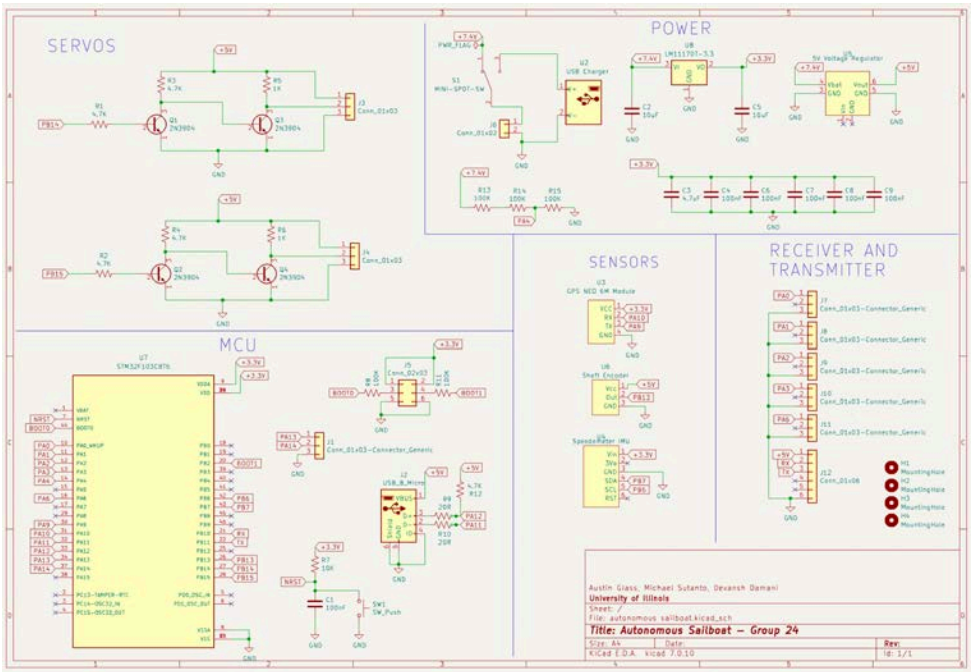
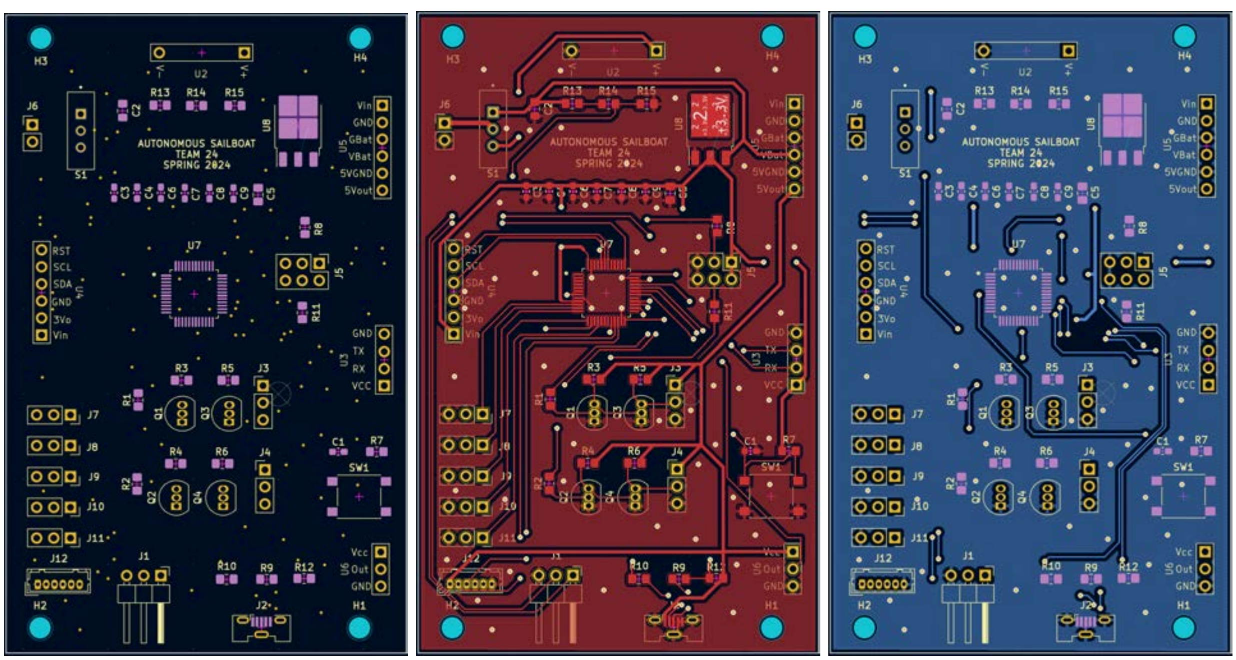


Figure B.1.1 - Whole Schematic



Figures B.1.2, B.1.3, B.1.4 - Full PCB W/O Copper Ground Layers and Connections, Top Copper Layer, and Bottom Copper Layer

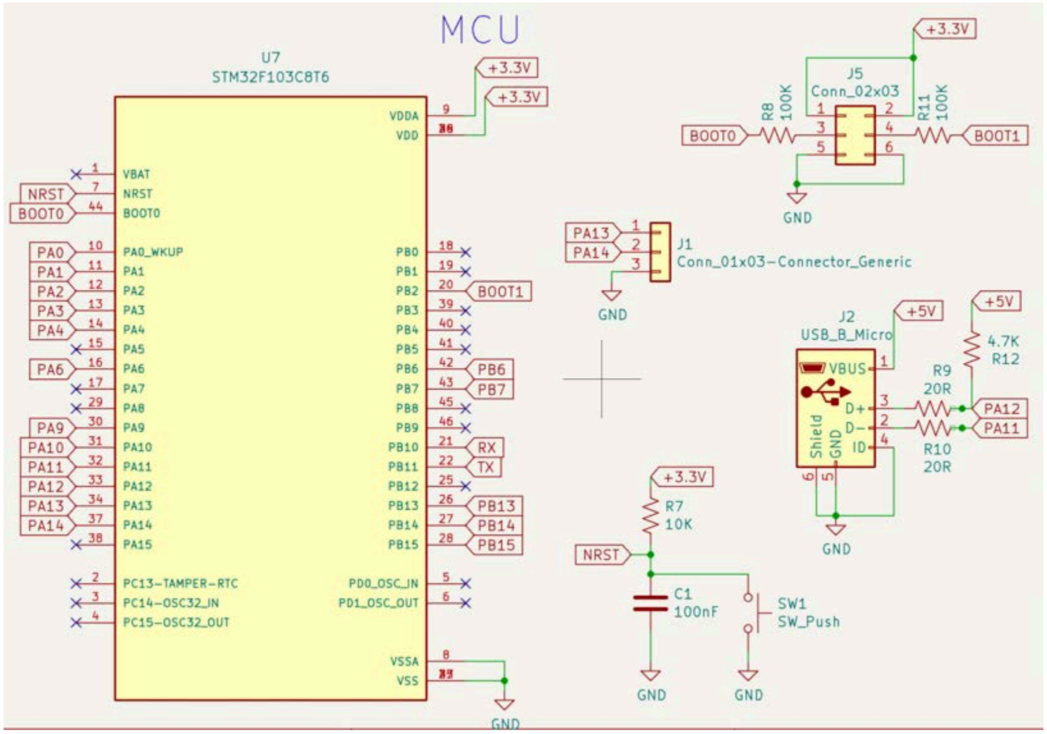


Figure B.2.1 - Schematic of Microcontroller (U7), Programming I/O (J1), Boot Pins (J5), Optional Micro-USB output port (J2)

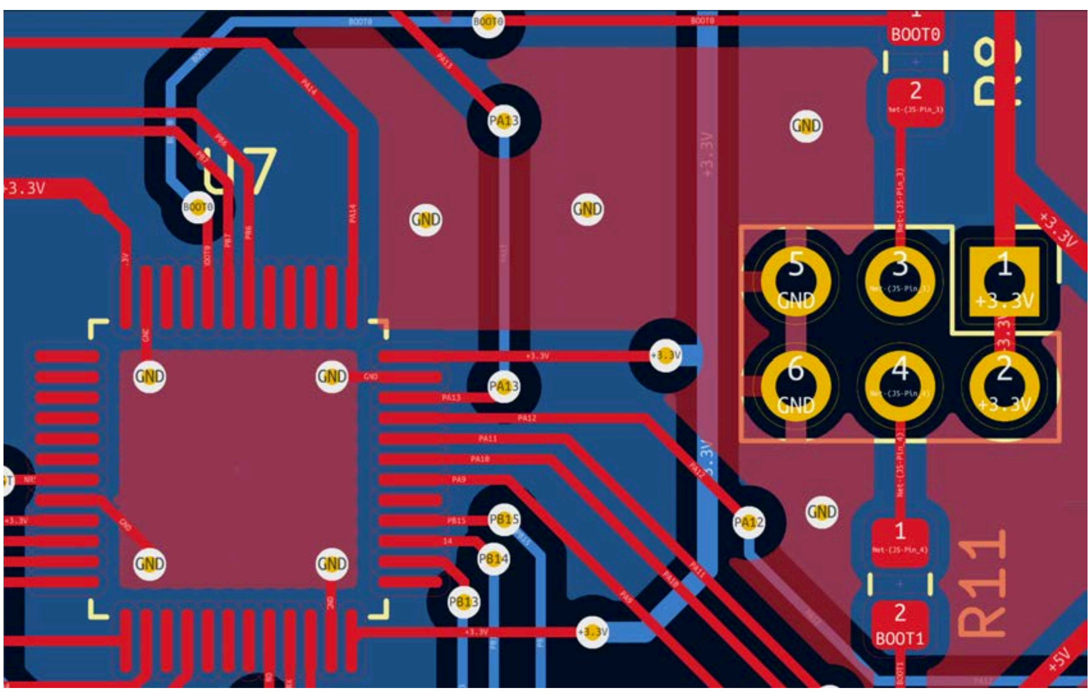


Figure B.2.2 - PCB design Microcontroller w/ Pin Assignments (U7), and BOOT 0/1 set pins (J5)

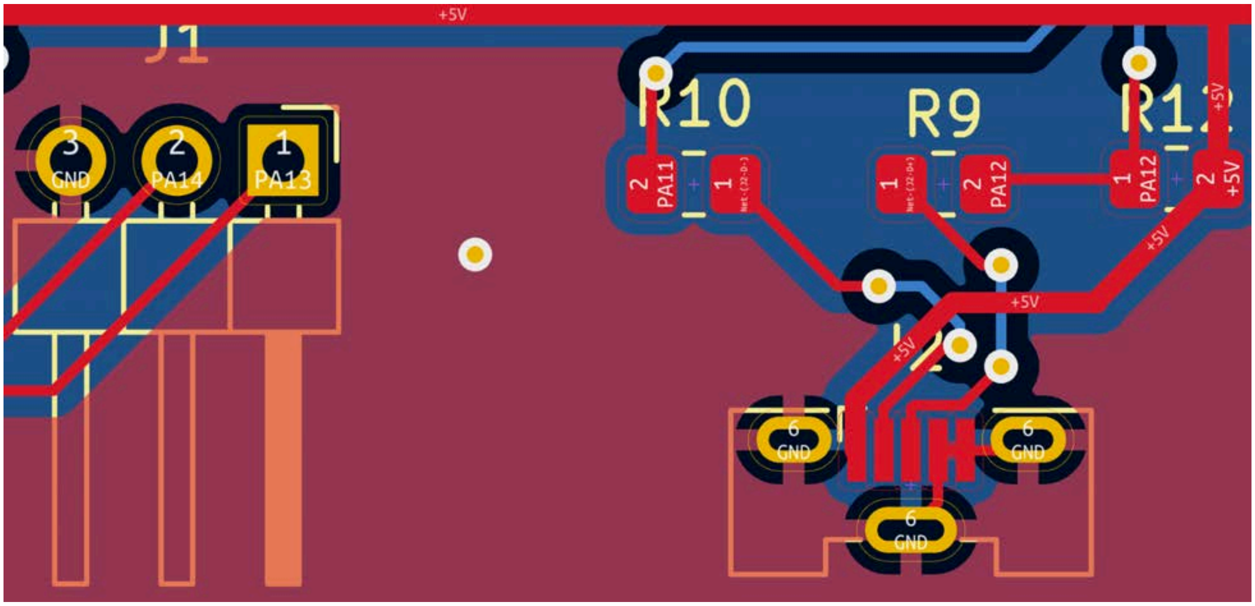


Figure B.2.3 - PCB design for Programming I/O (J1 - PA13 → SWDIO, PA14 → SWDCLK), and optional Micro-USB port (J2 - unused on final product)

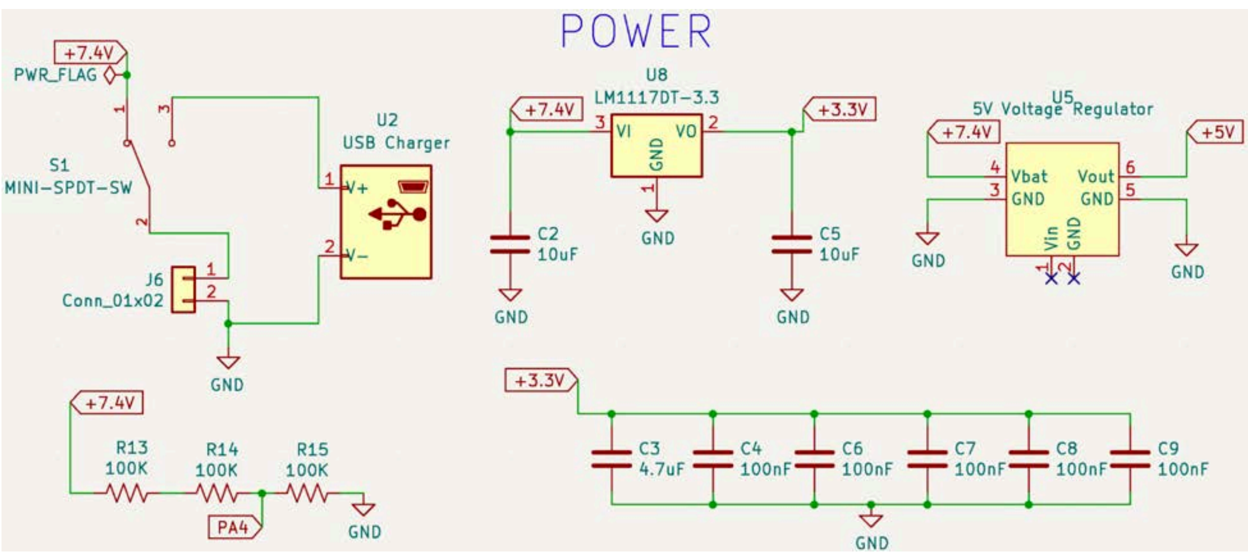


Figure B.3.1 - Schematic of Power Subsystem - Battery (J6), Charger (U2), Charger / Power Switch (S1), 3V Regulator (U8), 5V Regulator (U5 - repurposed for LM7805 5V Regulator), Battery Percentage Indicator, Circuit Protection Capacitors

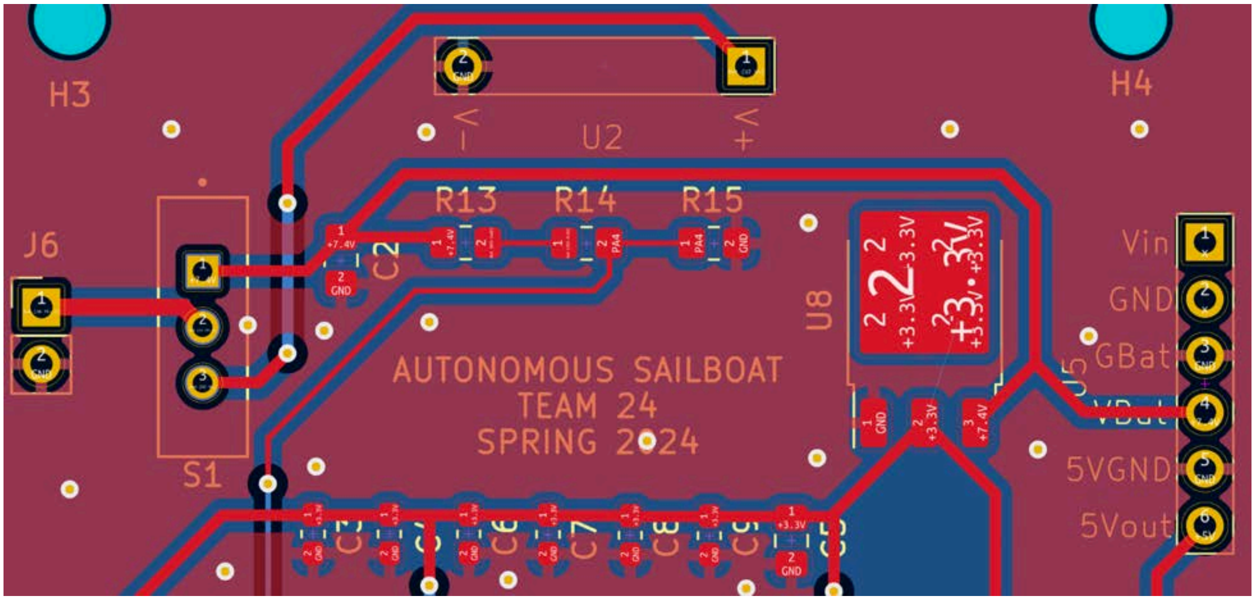
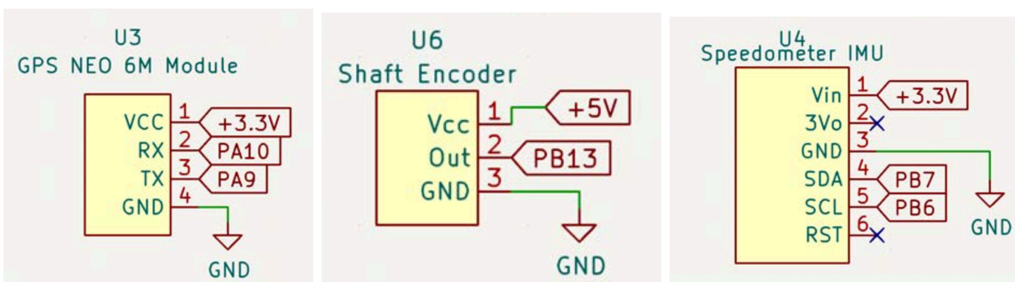
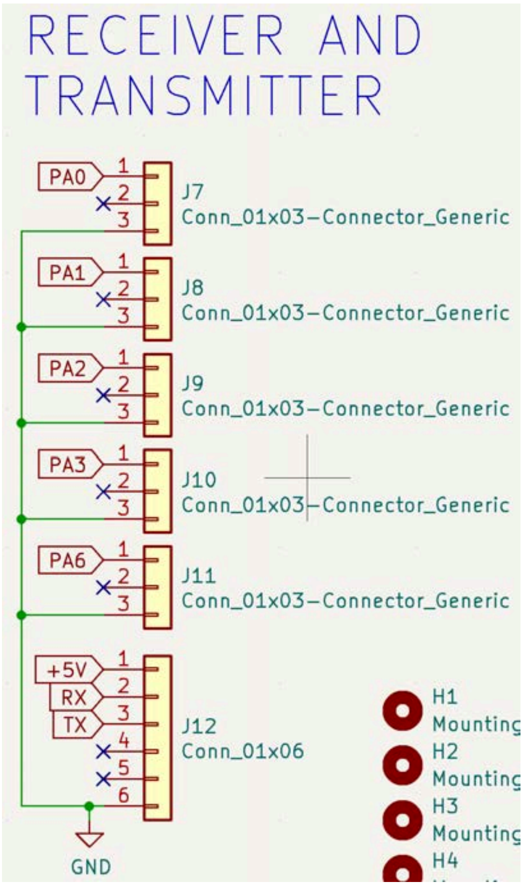


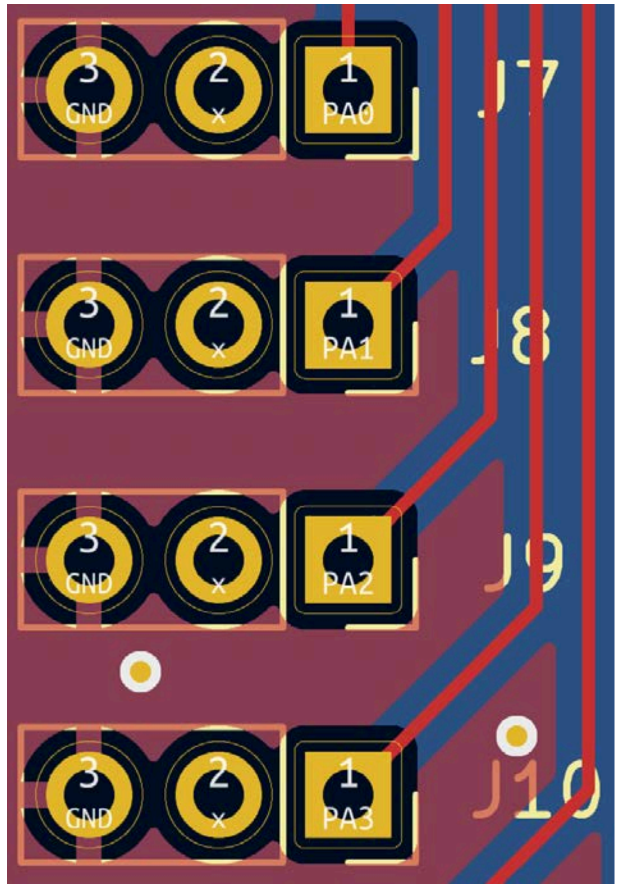
Figure B.3.2 - PCB design for Power Subsystem - Battery Port (J6), Switch (S1), Charger Port (U2), Battery Percentage Indicator Circuit, 3V Regulator (U8), and 5V Regulator pins (U5 - repurposed for LM7805 5V Regulator).



Figures B.4.1, B.4.2, B.4.3 - Schematics for Sensor Subsystem. GPS (U3), Shaft Encoder, Speedometer

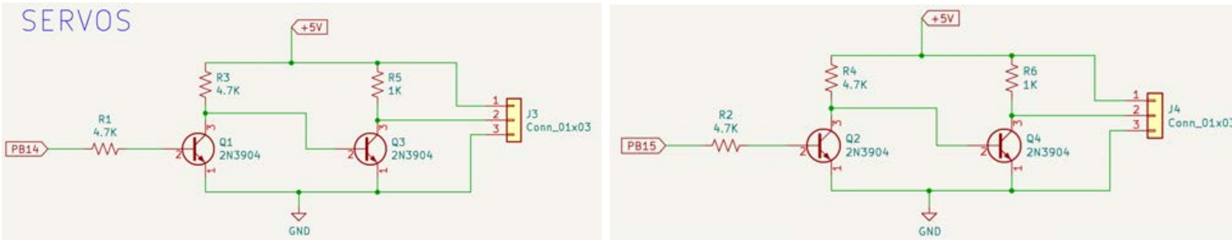
Figures B.4.4, B.4.6, B.4.7, B.4.8 - PCB Design for Sensor Subsystem - GPS (U3), Shaft Encoder (U6 & J11 - PA6 Pin repurposed for output pin 2 from shaft encoder), and IMU (U4)



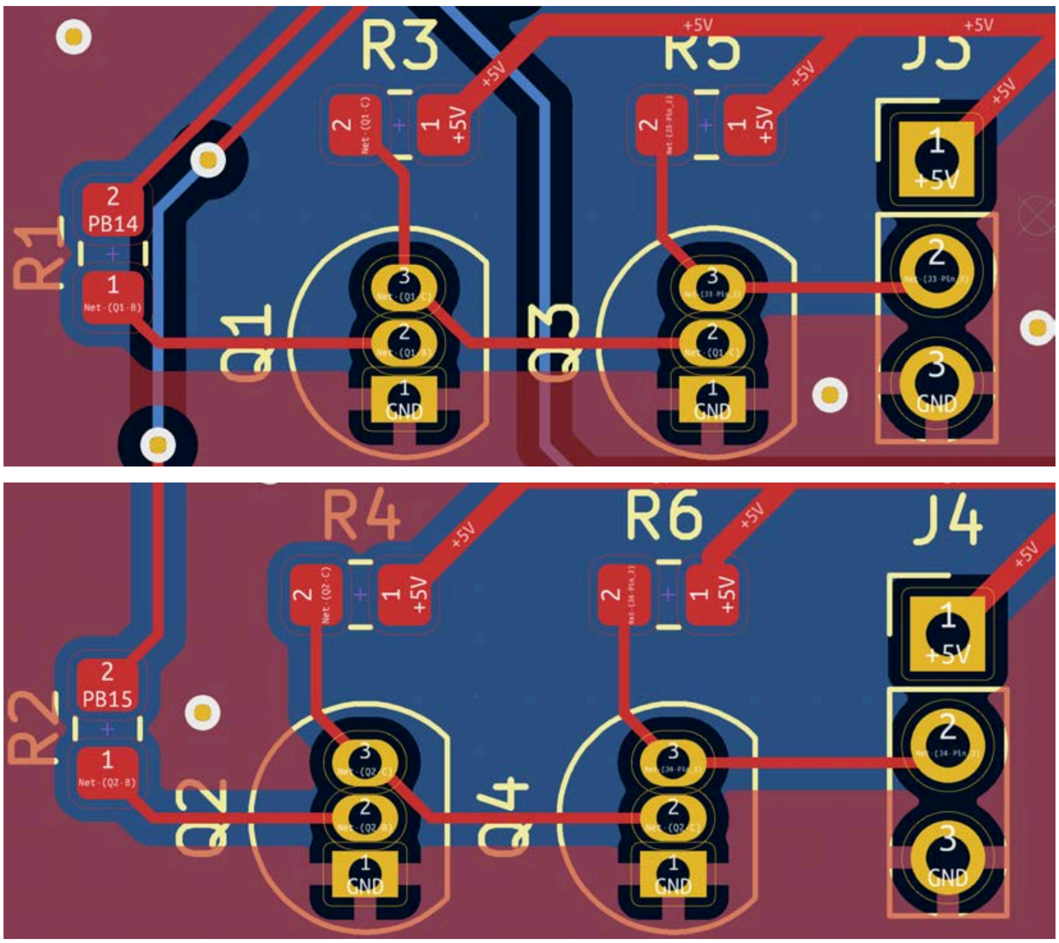


Figures B.5.1, B.5.2 - Schematic for Communication Subsystem - RC Receiver Channels (J7, J8, J9, J10, J11 - PA6 Pin repurposed for output pin 2 from shaft encoder), Telemetry Radio Receiver (J12). PCB Design for RC Receiver Channels (J7, J8, J9, J10)

Figure B.5.3 - PCB Design for Telemetry Radio Receiver (J12)

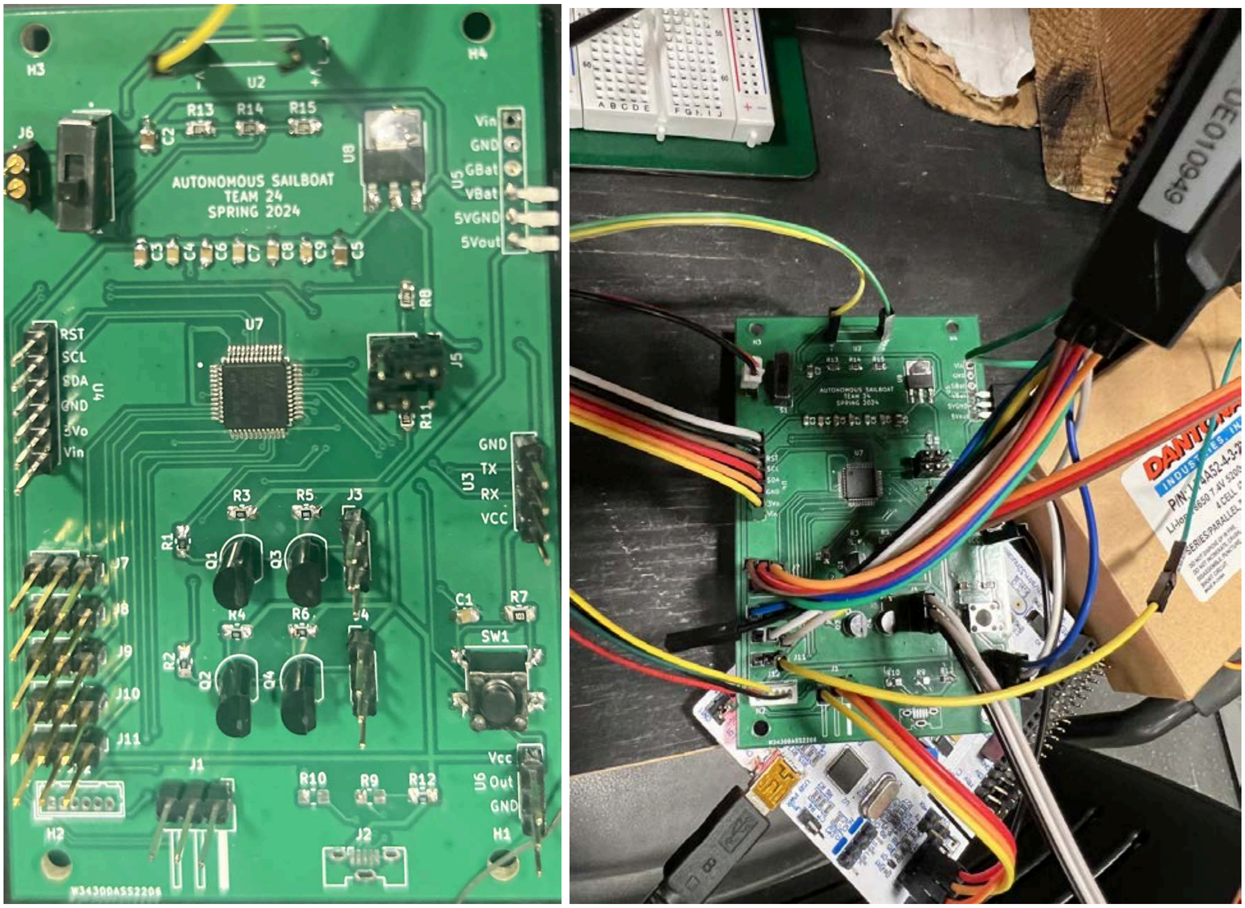


Figures B.6.1, B.6.2 - Schematic for Servo Controller Circuits (J3/PB14 - Winch, J4/PB15 - Rudder)

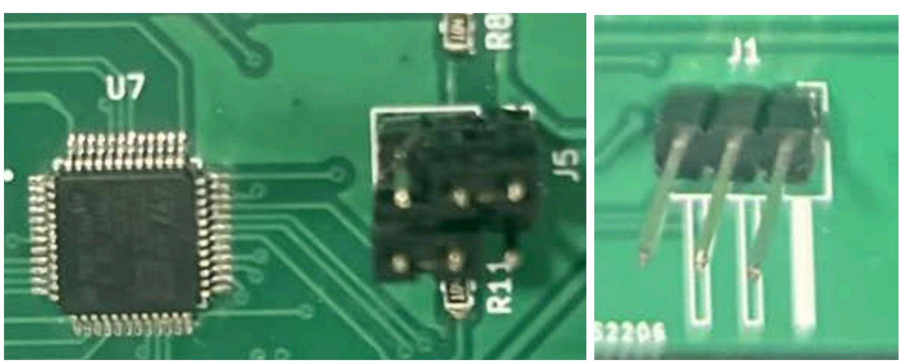


Figures B.6.3, B.6.4 - PCB Design for Servo Controller Circuits (J3/PB14 - Winch, J4/PB15 - Rudder)

Appendix C - Physical Components and PCB

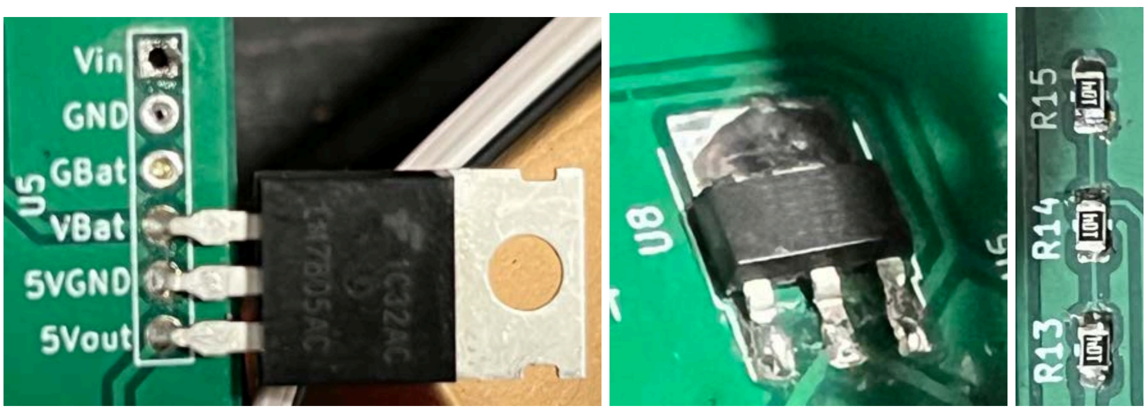


Figures C.1.1, C.1.2 - Full PCB with no connections, Full PCB With All Connections and ST-Link Dev Board

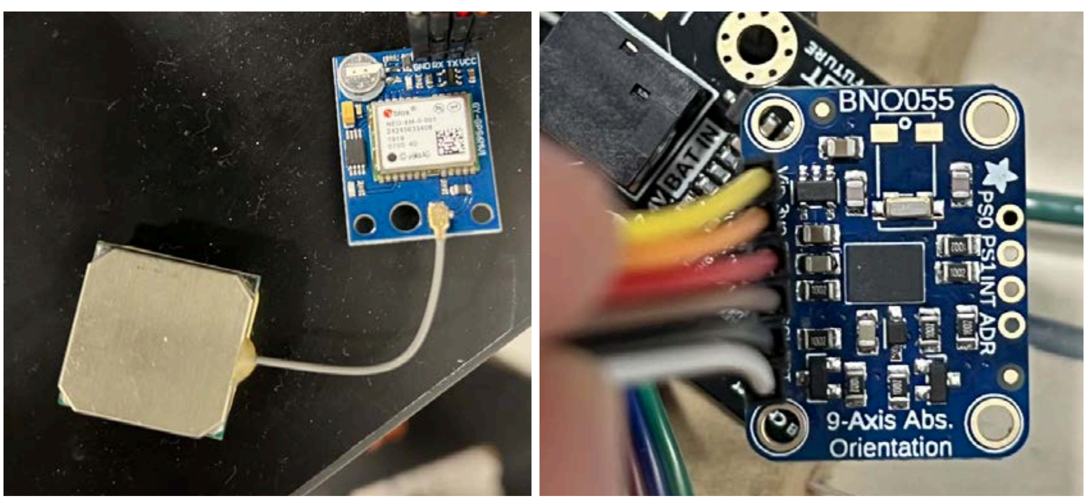


Figures C.2.1, C.2.2 - Microprocessor and BOOT 0/1 Pins, Programming I/O Ports

Figure C.3.1, C.3.2, C.3.3 - 7.4V Battery, 5V 1A Micro-USB Charger, Battery PCB Connection and PCB Power / Charging Toggle Switch



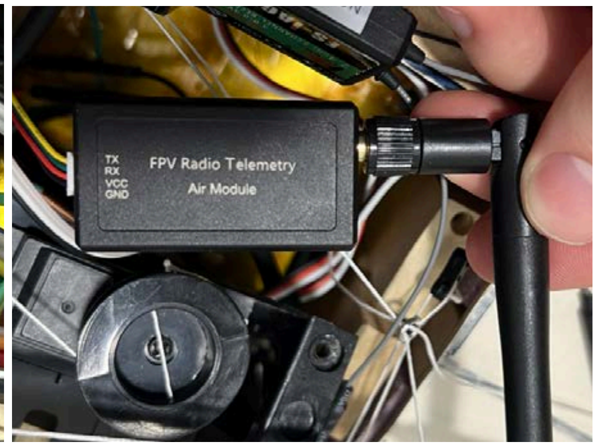
Figures C.3.4, C.3.5, C.3.6 - 5V Voltage Regulator, 3V Voltage Regulator, Battery Indicator Circuit



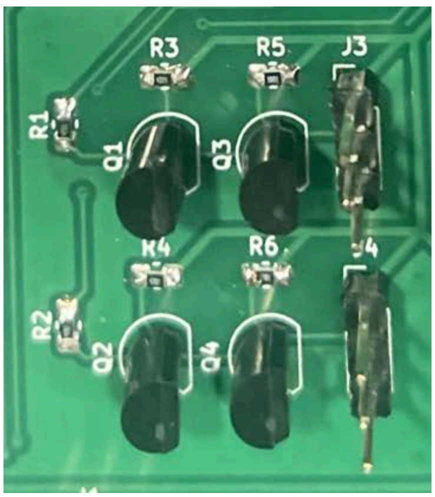
Figures C.4.1, C.4.2 - GPS, IMU

Figure C.4.3 - Shaft Encoder

Figures C.5.1, C.5.2 - FlySky Ground Control Remote, Telemetry Radio Computer Receiver



Figures C.5.3, C.5.4 - FlySky Receiver, Telemetry Receiver



Figures C.6.1, C.6.2 - Winch and Rudder Servo Power Ports, Winch Servo


Figures C.6.3, C.6.4 - Rudder, Temporary Rudder Servo

Appendix D - Code Screenshots

```
std::vector<float> getHeadingRoll(Adafruit_BNO055 &bno){
    imu::Vector<3> from(1,0,0);
    imu::Vector<3> mag = bno.getVector(Adafruit_BNO055::VECTOR_MAGNETOMETER);
    imu::Vector<3> acc = bno.getVector(Adafruit_BNO055::VECTOR_ACCELEROMETER);

    imu::Vector<3> E = mag.cross(acc);
    E.normalize();

    imu::Vector<3> N = acc.cross(E);
    N.normalize();

    float heading = atan2(E.dot(from), N.dot(from)) * 180 / PI;
    if (heading < 0) heading += 360;
    float roll  = atan2(acc[1],acc[2]) * 57.2957;
    std::vector<float> ret(heading, roll);
    return ret;
}
```

Figure D 1: IMU Heading and Roll Code

```
// Retrieving wind direction from rotary encoder "tick" position
int getWindDir(RotaryEncoder *encoder){
  int pos = abs((int)encoder->getPosition()) % 1190; //1190 full rotation
  int angle;
  if ((int)encoder->getDirection() <0) {
    angle = 360 - map(pos,0,1190,0,360);
    }
  else {angle = map(pos,0,1190,0,360);}
  return angle;
}
```

Figure D 2: Wind vane Mapping Function

```
void setRudder(){
  if (_mode == 0){
    float newAngle = map(val[1],range[1][0],range[1][1],servoRange[0][0],servoRange[0][1]);
    if (newAngle < (_rudderAngle - 7.7) || newAngle > (_rudderAngle + 7.7)  ){
      _rudderAngle = newAngle;
     }
  }
}
```

Figure D 3: Rudder Servo PWM Mapping Function

```
void setWinch(){
  /* Map PWM into range: -3,-2,-1,0,1,2,3 (multiplier for adjusting tightness) */
  newPhi = map(val[0],range[0][0],range[0][1],-3,4);
  float newAngle;
  if (newPhi > 0){newAngle = constrain(_winchAngle + ( (newPhi*newPhi) / 100),servoRange[1][0],servoRange[1][1]);}
  else {newAngle = constrain(_winchAngle -( (newPhi*newPhi) / 100),servoRange[1][0],servoRange[1][1]);}
  _winchAngle = newAngle;
}
```

Figure D 4: Winch Servo PWM Mapping Function

```
void printTelemetry(
    float _mode, unsigned long teleTime, unsigned long &prevTeleTime,
    double _lat, double _lng, float windAngle,
    float _winchAngle, float _rudderAngle, float desiredHeading,
    float _heading, float _course, float roll, float _knots,
    float distanceTo, double base_lat, double base_lng
){
    if (teleTime >= prevTeleTime + TELE_PERIOD) {

        if(_mode == 0)teleSerial.print("MANUAL ");
        else if(_mode == 1)teleSerial.print("AUTO ");
        else teleSerial.print("RETURN TO BASE");

        teleSerial.print(" lati ");
        printDouble(teleSerial, _lat,1000000);
        teleSerial.print(" lngi ");
        //teleSerial.print(_lng);
        printDouble(teleSerial, _lng ,1000000);
        teleSerial.print(" windAngle ");
        teleSerial.print(windAngle);
        teleSerial.print(" winchAngle ");
        teleSerial.print(_winchAngle);
        teleSerial.print(" rudderAngle ");
        teleSerial.print(_rudderAngle);
        teleSerial.print(" desiredHeading ");
        teleSerial.print(desiredHeading);
        teleSerial.print(" heading ");
        teleSerial.print(_heading);
        teleSerial.print(" course ");
        teleSerial.print(_course);
        teleSerial.print(" roll ");
        teleSerial.print(roll);
        teleSerial.print(" knots ");
        teleSerial.print(_knots);
        teleSerial.print(" distanceTobase ");
        teleSerial.print(distanceTo);
        teleSerial.print(" baselat ");
        printDouble(teleSerial, base_lat, 1000000);
        teleSerial.print(" baselng ");
        printDoublenewL(teleSerial, base_lng, 1000000);
        prevTeleTime = teleTime;
        teleSerial.print("\n\n");
    }
}
```

Figure D 5: Telemetry Serial Print Formatting Function

```
float getWinchAngle(float windAngle){
    if     (windAngle >=0 && windAngle <= 20  ) { return 72; }
    else if(windAngle >20 && windAngle <= 60  ) { return 72; }
    else if(windAngle >60 && windAngle <= 70  ) { return 72; }
    else if(windAngle >70 && windAngle <= 110 ) { return 75; }
    else if(windAngle >110 && windAngle <= 140) { return 80; }
    else if(windAngle >140 && windAngle <= 220) { return 110;}
    else if(windAngle >220 && windAngle <= 250) { return 80; }
    else if(windAngle >250 && windAngle <= 290) { return 75; }
    else if(windAngle >290 && windAngle <= 300) { return 72; }
    else if(windAngle >300 && windAngle <= 340) { return 72; }
    else                                        { return 72; }
}
```

Figure D 6: Wind Angle to Winch Servo Mapping

```
//Heeling Angle Adjustment
if ((abs(roll) < 160) && (abs(roll) > 115)){
if (heelingPhi >= (servoRange[1][1] - servoRange[1][0])){ heelingPhi = (servoRange[1][1] - servoRange[1][0]);}
else {heelingPhi = heelingPhi + 0.02;}}
else {
    if (heelingPhi <= 0.08) { heelingPhi = 0; }
    else { heelingPhi = heelingPhi - 0.1;}
}
```

Figure D 7: Danger State Control Code

```
double getRudderAngle(float deviation, float period){
    // PI CONTROLLER
    float integral  = PID_prevIntegral + (deviation * period);

    double newAngle = Kp*deviation + (Ki * integral);

    if (newAngle >= 45 || newAngle <= - 45 ){ integral  = PID_prevIntegral; }

    PID_prevIntegral = integral;

    return newAngle;

}
```

```
const float PID_period = 100; //in ms
float PID_prevIntegral;
const long Kp = 3.1;
const float Td = 0.1;
const float Ti = 3;
const float Ki = Kp / Ti;
const float Kd = Kp * Td;
```

Figure D 8: Rudder PI Control Function

Appendix E - RV Tables

Controller Subsystem- Table E.1

Requirement	Verification
The controller subsystem must not allow the rudder and sail servo to move in response to the RC remote when in autonomous mode.	1. Place the sailboat on the lazy Susan and turn off the box fan so that the wind vane encoder doesn't pick up any wind angle. 2. Turn on the sailboat, wait 5 seconds, and turn the autonomous switch to ON. 3. Move the two joysticks from manual mode and confirm that the servos are not moving.
The controller subsystem should send accurate sensor data through telemetry radio to the laptop in ground control subsystem.	1. Turn on the laptop with the data viewer. 2. Place the sailboat on the lazy Susan and set it to autonomous mode. 3. Turn on the box fan from any angle that is not in the No-Go zone. 4. The data viewer in the laptop should report values the AWA, compass heading, current GPS measurement, servo angles, speed estimation, and battery percentage estimate.

The controller subsystem should be able to send PWM signals to the rudder and sail servo and move it to its desired angle with a tolerance of maximum error of 2°.	1. Place the boat in the lazy Susan and add cardboards below the platform so the lazy Susan doesn't rotate. 2. Calibrate the sail servo and rudder servo to be aligned with the 0 degree start position (aligned with the hull centerline) 3. Run a custom program that sets the sail servo angle and rudder angle to a specific predetermined angle. 4. Use a protractor to measure the actual angles of the rudder and sail servo as they respond to the PWM signals. 5. Verify that the actual angles fall within a tolerance of ±2° of the desired angles.
The controller subsystem should correctly adjust the sail and rudder angle according to the apparent wind angle based to a maximum error of 2°.	1. Place the boat in the lazy Susan such that its heading is aligned with 0 degrees. 2. Place a box fan directly in front of the lazy Susan. 3. Calibrate the sail servo and rudder servo to be aligned with the 0 degree start position (aligned with the hull centerline). 4. Turn the boat to Autonomous mode on through the remote control. 5. For each desired apparent wind angle to be tested based on wind angle, rotate the sailboat such that its heading aligns with the desired angle. 6. Allow the boat's autonomous steering algorithm to adjust the rudder and sail servo angles. 7. Use a protractor to measure the actual angles of the rudder and sail servo as they respond to the PWM signals. 8. Repeat for each of the different experimental angles. 9. Verify that all the actual angles fall within tolerance of ±2° of the desired angles.

The controller subsystem should be able to adjust rudder angle based on the deviation of the current compass angle and the destination compass heading. This adjustment should put the boat back on an acceptable deviation range of $\leq 4°$ in less than 10s.	1. Place the boat in a small water bath (inflatable kid pool). 2. Set the boat to Autonomous mode and set an imaginary heading straight forward from the starting position of the boat. 3. Point a box fan to blow from any angle where the boat is not in the danger zone. 4. Allow the boat to sail autonomously. 5. View the data logged in the laptop and watch for the moments when the heading deviation is ≥ 4 degrees and measure how long it took to return to a heading less than that.
The controller subsystem should be able to perform rudder angle adjustments when the speedometer IMU detects a sideways drift speed of >2m/s. This adjustment should counteract the sideways drift and bring it back to an acceptable range of <2m/s.	1. Place the boat in a small water bath (inflatable kid pool). 2. Set the boat to Autonomous mode and set an imaginary heading straight forward from the starting position of the boat. 3. Point a box fan to blow from any angle where the boat is not in the danger zone. 4. Allow the boat to sail autonomously. 5. View the data logged on the laptop and watch for the moments when the speedometer measurement is > 2m/s and measure how long it took to return to a value less than that.
The controller subsystem must be able to reach its destined waypoint with an error range of 5m.	1. Place the boat in an outdoor water environment. 2. Mark a destination waypoint around 20m away from the boat but in hands reach from shore with an anchored float. 3. Set the boat to Autonomous mode and set a destination waypoint to the marked destination coordinates. 4. Allow the boat to sail autonomously until it reaches it completes its journey. 5. Record its ending location with another weighted float. 6. Measure the distance between the two float markers to derive its error. 7. Confirm error is below 5m.
The controller subsystem must prevent the return to base feature to be triggered if the base position is ≤ 5m	Turn on the sailboat and verify that the base position is set via Mission Planner. Move the sailboat ≤ 5 m in any direction and press the return to base button.

of the sailboat's current GPS location.	
The controller subsystem should move the servos in response to the RC wireless controller with latency of <25ms.	1. Place the sailboat on the lazy Susan. 2. Turn on the sailboat, wait 5 seconds, and make sure the autonomous mode is OFF. 3. Move a joystick from the RC controller. 4. Use the data logging capability to measure the difference between the timestamps of wireless controller input and servo movement and analyze the difference. 5. Confirm it is < 25ms.
The controller subsystem should be able to keep the heeling angle of the sailboat to <|15| degrees to prevent capsizing.	1. Place the boat in a small water bath (inflatable kid pool). 2. Set the boat to Autonomous mode and set an imaginary heading straight forward from the starting position of the boat. 3. Point a box fan to blow from any angle that is not in the No-Go zone [Table 2.4.1.1] 4. Allow the boat to sail autonomously. 5. Manually agitate the sailboat to have a heeling angle > 15 degrees. Confirm it is > 15 degrees by simultaneously viewing the data in the laptop. 6. Observe the response of increasing sail winch angle and see if it could recover its heeling angle to < 15 degrees.

Sensor Subsystem- Table E.2

Requirement	Verification
The IMU should be able to monitor the current heading of the boat to a tolerance of ±5°.	1. Place lazy Susan above a paper with different angles marked on it. 2. Place the boat on top of lazy Susan. 3. Align the boat to a known compass heading of 0°. 4. Turn on the IMU and record its reading. 5. Gradually rotate the boat in increments, noting the corresponding compass readings. 6. Compare the compass readings with the known headings, ensuring they fall within the tolerance of ±5°.

The GPS should be capable of reporting boat coordinates with an accuracy of ±2.5 meters.	1. Bring the boat into a room with 4 corners and dimensions ≥ 9 meters on each edge. 2. Start by bringing the boat to one corner of the room. 3. Measure the coordinates at that location. 4. Move to each of the other corners of the room and record the coordinates by the GPS at each of these corners 5. Calculate the accuracy of these GPS recordings compared to the supposed theoretical value based on the known dimensions of the room. The Windvane encoder data should allow the appropriate algorithm to measure the direction of wind within a tolerance of ±5°. Anything greater than this will cause errors in adjusting sail chord angle which may exacerbate the error between course sailed and compass heading.
The Windvane encoder data should allow the appropriate algorithm to measure the direction of wind within a tolerance of ±5°. Anything greater than this will cause errors in adjusting sail chord angle which may exacerbate the error between course sailed and compass heading.	1. Place the boat in the lazy Susan such that its heading is aligned at 0°. 2. Place a box fan directly in front of the lazy Susan. 3. Calibrate the wind vane encoder. 4. For each desired apparent wind angle to be tested based on Table 2.4.1.1, rotate the sailboat such that its heading aligns with the desired angle. 5. For each wind angle, check the laptop for the measured apparent wind angle by the wind vane encoder. 6. Verify that all the actual angles fall within the measurement tolerance of ±5° of the desired angles.
The IMU should be able to allow the algorithm to estimate the speed of the boat to an accuracy of 0.5m/s or 5% of true velocity.	1. Place the boat in the lazy Susan. 2. Place the lazy Susan in a stationary treadmill, where the hull centerline is perpendicular with the axis of the treadmill belt. 3. Turn the treadmill on and move it at a safe speed of 1.5m/s. 4. See the recording and estimated speed value calculated by the IMU. 5. Verify that is an accuracy of 0.5m/s or 5% of true velocity of 1.5 m/s.

Power Subsystem- Table E.3

Requirement	Verification
Must be able regulate battery voltage to power components throughout the discharge cycle of the battery and automatically cutting out power when battery voltage drops too low.	1. Connect the input of the voltage regulator to the battery supply. 2. Connect the output of the voltage regulator to a programmable load. 3. Set the voltage supply to the maximum battery voltage (e.g. 7.4V for a Li-Ion battery). 4. Measure the output voltage of the voltage regulator using a multimeter under both no-load and full-load conditions. 5. Ensure that the output voltage remains within the specified range (e.g., 3.3V ±5%) at all times. 6. Gradually lower the input voltage from the battery to simulate decreasing battery levels. 7. Verify that the output voltage of the voltage regulator remains stable until the battery voltage reaches a predefined threshold. 8. Once the battery voltage drops below the threshold, confirm that the voltage regulator cuts off power, resulting in a decrease in the output voltage to 0V or near 0V. 9. Confirm that the voltage regulator maintains power cutoff to prevent over-discharge of the battery, ensuring the protection of the battery and connected components.
The battery must be able to be recharged via the micro-USB connector within 6 hours.	1. Begin with the battery completely discharged. 2. Plug the micro-USB into the charging port. 3. Initiate a timer to track the charging duration. 4. Regularly check the battery's charging status and voltage using appropriate tools. 5. Note the time when the battery reaches full charge. 6. Calculate the elapsed time from the start of charging to full charge and compare it to the 6-hour target.
The battery should be able to last minimum of 3 hours from a full charge when boat is used for testing.	1. Begin with the battery completely charged. 2. Use the boat to test, verify, and debug other aspects of the requirements. 3. Ensure the boat is still on at the end of session. 4. If single session is > 3h, the verification is successful. Otherwise keep adding total times that the boat is not off yet and the verification succeeds when total time is > 3h.

The power subsystem should not short circuit under the conditions of water splashing and during boat use.	Part 1 1. Expose the power subsystem to simulated water splashing conditions. 2. Observe the system for any signs of electrical malfunction or short circuiting. 3. Repeat the test multiple times to ensure consistency. Part 2 4. Run a program that oscillates the rudder servos in a relatively high frequency to simulate the worst-case scenario of power consumption. 5. Monitor if there are any power malfunctions or short circuits.
The battery percentage read by the circuit should be accurate within 5% of the battery charge.	1. Connect a fully charged Battery supply to a programmable load (e.g. potentiometer). 2. Measure the output voltage of this with a multimeter under no load and full load conditions. 3. Plug the battery into the board and compare the calculated value using the voltage with the value read by the microcontroller.

Communications Subsystem- Table E.4

Requirement	**Verification**
The data from the microcontroller has to be properly transmitted with latency of less than 60ms through the telemetry radio.	1. Configure the microcontroller to send a series of data packets at regular intervals through the telemetry radio. 2. Set up the laptop to record the timestamps of incoming data packets as they are received from the telemetry radio. 3. Calculate the latency for each received data packet by subtracting the timestamp in the packet from the current time when the packet is received. 4. Verify that it is < 60ms

Turning on the autonomous functionality should not be able to be turned on before 5s after the remote is turned on. To ensure sensors calibrate.	1. Place the boat on lazy Susan and turn it on. 2. Place box fan on an angle that is not in the No-Go zone of the sailboat [Table 2.4.1.1]. 3. Turn on the RC remote and immediately try to turn on autonomous mode. 4. The boat should not move any of its servos indicating that autonomous mode was not turned on.

Ground Controls Subsystem- Table E.5

Requirement	**Verification**
The FS-I6 transmitter should transmit signals to the appropriate channel to move the appropriate servo in manual mode. Refer to table 2.4.4.1.	1. Place the boat in stationary lazy Susan. 2. Turn the boat on in Manual mode. 3. Move the right gimbal left and right and verify that it is the sail servo rotating. 4. Move the left gimbal up and down and verify that it is the rudder servo moving.
Laptop should be able to view data from telemetry radio.	1. Turn on the sailboat and run the monitoring program on the laptop. 2. See if the data changes as the different sensors are moved.