2023-2024 SailBot MQP

A Major Qualifying Project Report submitted to the faculty of
WORCESTER POLYTECHNIC INSTITUTE
in partial fulfillment of the requirements for the
degree of Bachelor of Science

Submitted by:

Matthew Gomes, RBE/CS
Erin Murphey, ME
Anthony Virone, RBE
Theodore Winter, RBE

April 25, 2024

Professor William Michalson, Advisor
Professor Kenneth Stafford, Advisor
Robotics Engineering Department

Abstract

Our project aims to create an autonomous robotic sailboat (SailBot) in order to compete in the International Robotic Sailing Regatta, building off of the results of previous years' SailBot MQP. We designed and manufactured a new hull to improve the fluid dynamics of the hull and to improve physical accessibility inside the boat. The new hull was manufactured out of carbon fiber and mechanical features were integrated from the previous year's SailBot including the sail, keel, and ballast mechanism. We fully replaced the boat's software systems, with a focus on communications reliability, failure recovery, and autonomous functionality. Additionally, we redesigned the boat's electrical system to ensure greater platform stability and longevity.

Acknowledgements

There are many people who helped and supported our project this year. Firstly, we would like to thank our advisors, Professor Michalson and Professor Stafford for their guidance and advice. We would also like to thank Oliver Peloquin for aiding in design and manufacturing throughout the year. We also received support from a lot of outside individuals and would like to thank them for lending their resources, workspaces, and time to assist our project. These individuals include Samuel Kierstead, and Mike, Becky and Grace Audette. We would also like to thank New England Boatworks (NEB), Bill Murphey, Taylor North, and Gaetano Virone for their generous donations of materials as well as previous SailBot MQP teams for their work and documentation of past research. Thank you all for your support.

Authorship

Section	**Primary Author(s)**
Abstract	Anthony Virone
Acknowledgements	Erin Murphey
Executive Summary	Matthew Gomes, Erin Murphey, Theo Winter
Project Purpose and Goals	Matthew Gomes
Previous System Architecture - Mechanical Systems	Erin Murphey, Anthony Virone
Previous System Architecture - Electrical Systems	Theodore Winter
Previous System Architecture - Software Systems	Matthew Gomes
Design Objectives	Matthew Gomes, Erin Murphey, Anthony Virone, Theodore Winter
System Design - Mechanical Design	Erin Murphey, Anthony Virone
System Design - Electrical Design	Theodore Winter
System Design - Software System	Matthew Gomes
Testing and Validation	Matthew Gomes, Erin Murphey
Future Work and Recommendations	Matthew Gomes, Erin Murphey, Theo Winter
Broader Impacts	Erin Murphey
Conclusion	Matthew Gomes, Erin Murphey
Appendix C: Electrical Schematics	Theodore Winter
Proofreading	Mathew Gomes, Erin Murphey, Anthony Virone, Theodore Winter

Executive Summary

The goal of WPI's SailBot team is to develop an autonomous sailboat designed to compete in the International Robotic Sailing Regatta. The purpose of this robot is to complete different tasks, some of which are remote controlled and others fully autonomous. This project is a continuation of the previous SailBot MQPs to improve upon the systems and functionality of robots. This year's team focused on improving various mechanical, electrical, and software systems.

This year's mechanical focus was on designing and manufacturing a new hull better fit for the purpose of the boat. This included improving the fluid dynamics of the hull by creating a more tapered stern to create less turbulence, improving accessibility into the hull to make maintenance and implementing new systems easier, and redesigning mounting systems to be more rigid so sensors cannot move. Simulations were done through SolidWorks to compare the fluid dynamics of the previous and new hull in order to show that drag was reduced. Also, an internal structure was designed in order to support the load of all of these components. Other mechanical tasks included integrating systems from the previous hull such as the keel, wingsail, movable ballast and rudders. Together, these systems were able to create a functional hull capable of carrying out the tasks required for competition.

The electrical system of the boat was completely overhauled, as the old one was unreliable, dysfunctional and prone to faults. The redesign consolidated the electronics onto a main PCB, to improve reliability and space efficiency. New functionality included a battery management system to protect the LiPo batteries, upgrading the boat to run on a Jetson Orin Nano, and the migration of trim tab communications to WiFi instead of Bluetooth. The trim tab controller was also redesigned, allowing for the battery to be charged with no electrical reconfiguration while also improving the reliability of communications with the main boat. Overall, the functionality and reliability of the electrical system has been massively improved from last year, with the boat suffering no critical electrical failures at any point during testing.

This year's team fully replaced the boat's software system. Key focuses include improvements to communications reliability, failure recovery, and autonomous functionality. The boat is now controlled through a cross-platform telemetry application, and the previous radio communication system has been replaced with a 4g modem. These systems allow the user to easily control the boat in any area that has cell coverage. The pathfinding systems have been completely redesigned, leveraging a combination of grid raycasting, tack-point calculation, a modified A* algorithm, and a PRM* algorithm, allowing the boat to select the most appropriate planning algorithm for any given path segment. Path following has similarly been redesigned, with two versions available for use depending on the user's preference. A look-ahead implementation was created to provide smooth path following, with some amount of corner-cutting accepted as a tradeoff. For paths where corner cutting is not acceptable, a vector field implementation was also developed, which can provide a more accurate path following in some scenarios.

In testing, the boat was able to sail autonomously point to point between different user defined waypoints and detect buoys within 7 meters of the bow of the boat. The autonomous sail control was particularly successful, and was shown to be able to maintain proper position (lift or drag) as the boat changed its heading through jibing and tacking maneuvers, even in varying wind conditions. With the many improvements made this year, we believe that we will be successful in this year’s SailBot competition.

Table of Contents

Table of Figures

1 | Project Purpose and Goals

The purpose of this project is to continue development on WPI SailBot, an unmanned surface vehicle (USV) intended to function as an autonomous sailboat. The primary goal for the development of this robot is to showcase autonomous sailing capabilities. Functionality should include point-to-point navigation, station-keeping, and path following guided by waypoints and buoys. Particularly, the goals we planned to achieve are:

- Autonomously navigate point-to-point in a wind of between 5 and 15 knots, between points marked either by GPS coordinates or by buoys, to within a radius of 10 meters of the target position
- Navigate a course defined by GPS locations for a period of 8 hours, allowing for intermittent RC control to avoid collisions, rounding each point within a radius of 10 meters
- Detect buoys and obstacles within 5 meters of its bow (90 degree field of view) so as to affect rounding and/or react in an appropriate manner to avoid collision
- Autonomously hold a given position to a tolerance of +/- 20 meters for a period of at least 5 minutes
- Autonomously sail to a given position at least 1 nautical mile away from its starting location, and return to within 10 meters of its starting position.

For several years, WPI SailBot has competed in the SailBot competition, an international robotic sailing regatta. As such, our team is targeting functionality that aligns with the challenges presented in the SailBot competition.

2 | Previous System Architecture

2.1 Mechanical systems

2.1.1 Free-Rotating Wingsail

Since 2018, WPI's SailBot has used a rigid wingsail instead of a traditional cloth sail. This provided several advantages, including improved control, decreased weight, and increased durability compared to a cloth sail. The wingsail is free-rotating in its mount on the hull, and its direction is controlled by a trim tab which extends backwards from the sail. More information on the function of this wingsail and trim tab can be found in Appendices B and C. The wingsail functions as an independent robot, controlled by the boat via bluetooth. It consists of a carbon-fiber mast connected to the large rigid lower wingsail, a smaller detachable upper wingsail which yields magnetically in high wind, a counterweight which attaches to the mast to balance the rotation of the sail, and the trim tab which is used to control the orientation of the sail. The electronics necessary to control the trim tab are contained in a control box located in the lower wingsail, which contained:

- An Arduino development board with WIFI and bluetooth capability
- A small LiPo battery
- A battery charging board
- A baseboard to connect the previous components

This control box drove a splash-proof servo mounted on the trim tab, which manipulates the trim tab to control the sail's angle of attack into the wind.

2.1.2 Sliding-Rail Ballast

The purpose of the movable ballast is to balance and optimize the heel angle of the boat. Last year's SailBot team implemented a sliding-rail ballast to replace an older mechanical arm ballast system. This system consists of a 2 stage linear sliding rail system with an aluminum bar to mount weights to, a motor to drive the rail system, and a potentiometer to provide rail position feedback. This system is driven through a Talon SRX DC motor controller.

2.1.3 Keel

The keel of the boat serves to balance the weight of the boat as it heels as well as counteract the leeward direction of the sail's lift in order to propel the boat forward. The current keel consists of a lead weight at the bottom to contribute to balance and stability, with a hydrofoil around the shaft to reduce drag and to provide lift.

2.1.4 Rudder System

SailBot has a dual rudder system, intended to ensure that the boat maintains steering capabilities as its heel angle increases. The rudders were angled to provide maximum turning force when the boat is sailing at its optimal heel angle of 20 degrees. The dual rudders are controlled through a linkage connected to a singular servo motor.

2.1.5 Hatches

The hatches were located on the deck of the boat and allowed access to the boat's interior. The old hull contained one, 7.25 inch diameter, circular hatch placed near the bow of the boat and one placed at the stern of the boat. These hatches were large enough to put the necessary hardware and electrical equipment in the interior of the boat. However, these hatches were difficult for an engineer to reach inside the boat and work on equipment in the interior.

2.2 Electrical Systems

2.2.1 Jetson Nano

The main controller for SailBot was a Jetson Nano. The Jetson Nano is a low power single-board computer designed with image processing and robotics applications in mind. SailBot's Jetson Nano ran all the ROS nodes responsible for controlling the robot.

2.2.2 Ethernet Bridge

SailBot communicated with the shore through a 900mHz ethernet bridge. The ethernet bridge was connected to the ethernet port on the Jetson Nano and powered through a Power Over Ethernet (POE) Injector. It is responsible for converting IEEE 802.3 Ethernet messages to 900mHz radio signals which are broadcast and received via the rear antenna. The shore station for SailBot contained a yagi antenna and a second ethernet bridge, which converts the 900mHz signal back into Ethernet messages. This could be connected to a computer and allowed for full remote access to SailBot's telemetry and control through SSH within the radio range of approximately 1 nautical mile.

2.2.3 Airmar Weatherstation

SailBot's main source of odometry is its Airmar 220WX weatherstation. The Airmar is a highly capable sensor package mounted on the rear of SailBot. This sensor package includes a GPS receiver, gyro, magnetometer and anemometer. The Airmar connects to the boat through a NMEA 2000 connection. NMEA 2000 is a version of the CAN Bus designed for marine applications. To convert the NMEA 2000 messages into a format readable by the Jetson Nano, SailBot employs a Maretron NMEA 2000 to USB converter to decode NMEA2000 messages, and transmit them to the Jetson via serial over USB.

2.2.4 RC Receiver

SailBot could be controlled with a FRSky Taranis X9D RC controller. To receive the RC signals, SailBot had a L9R RC receiver contained inside the hull. The L9R antenna allowed for the boat to be teleoperated within radio range. However, the signal received from the L9R was found to be noisy due to interference. To rectify this issue and prevent unintended commands from being processed, SailBot utilized a schmitt trigger to filter the signal. A schmitt trigger is a comparator circuit that provides hysteresis to the control input. By filtering the signal, the integrity of the serial communication from the L9R could be restored.

2.2.5 Ballast Motor Control

SailBot's linear ballast rail is powered by a Denso DC Automotive Window Motor. To accurately control the speed of the motor, SailBot employs a Talon SRX DC Speed Controller. The Talon SRX allows for precise control of the motors speed and direction through a PWM connection to the Jetson Nano.

2.2.6 Wingsail Controller

The wingsail has no wired connection to the boat. Instead, it has its own separate electrical system. The wingsail was powered by a 3.7V single cell LIPO battery. This was connected to a charger/booster to allow the LIPO to provide 5V power to the Arduino. The same circuit is also responsible for charging the battery. When the battery is being charged, a relay cuts off the supply of current to the Arduino to allow the battery to charge. The Arduino MKR1010 contained an integrated Bluetooth Low Energy (BLE) transceiver which allowed it to communicate with the boat. The Arduino received trim tab position commands from the Jetson and used the wind vane and trim tab servo to adjust the angle of the sail to the desired position.

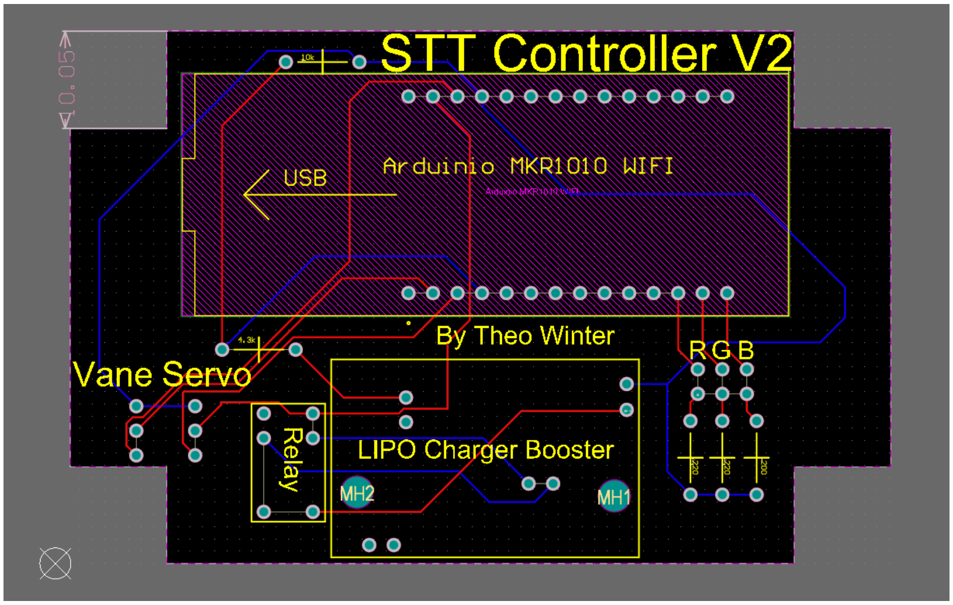


Figure 1: Original Wingsail PCB controller

2.3 Software Systems

2.3.1 ROS nodes

SailBot ran on ROS2 Dashing Diademata middleware software, which is a publisher-subscriber framework intended to facilitate rapid development of robotic systems. ROS introduces the concept of "nodes", which are independent processes which execute their tasks while publishing and subscribing to information channels for inter-process communication. The old WPI SailBot ran six nodes, described below.

1. The `airmar_reader` node, which reads information from the Airmar 220WX Weatherstation via the Maretron NMEA2000->NMEA0183 converter
2. The `serial_rc_receiver` node, which receives control signals from the FrSKY digital telemetry radio system
3. The `control_system` node, which integrates sensor information to make control decisions
4. The `pwm_controller` node, which receives commands from the `control_system` node and operated the ballast and rudder systems
5. `The trim_tab_comms` node, which commands the trim tab controller via a BLE (Bluetooth Low Energy) connection
6. The `debug_interface` node, which collects boat status information and sends it to on-shore telemetry via the onboard ethernet-radio bridge.

Figure 2 is a diagram of the flow of information in the ROS2 publisher/subscriber system. Input systems (nodes that interface with sensors) are to the left of the control_system.py block, and output systems (nodes that interact with the boats actuators) are to the right of the control_system.py block.

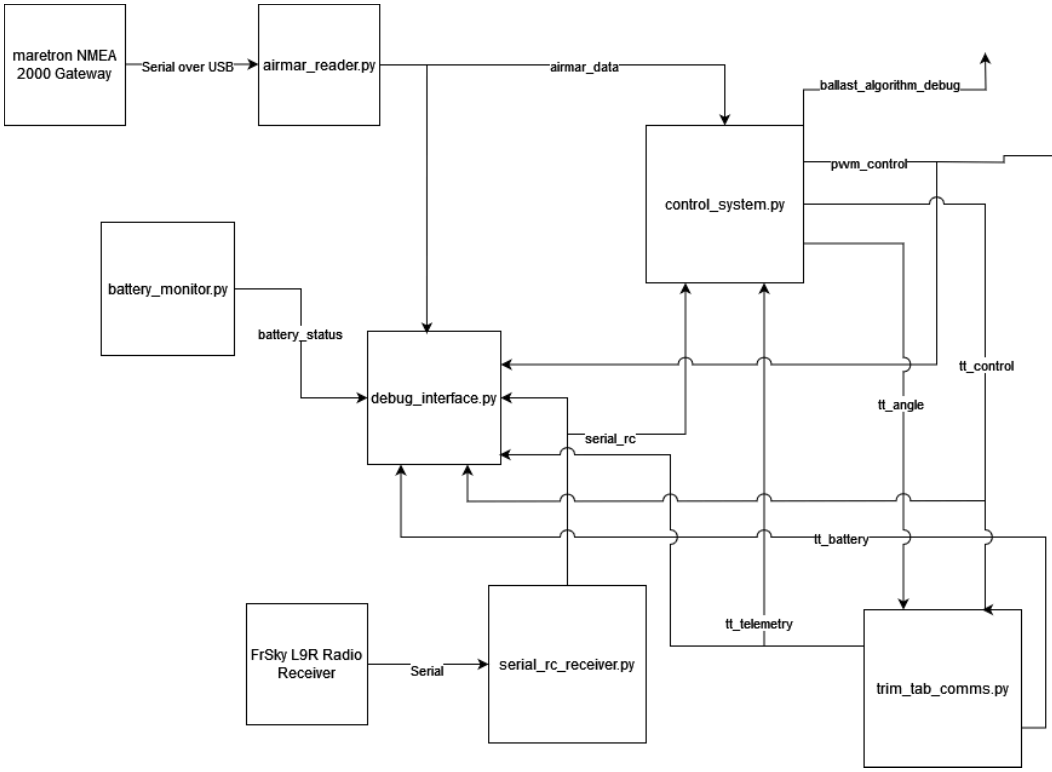


Figure 2. Block diagram of the flow of information in the ROS2 publisher/subscriber system

2.3.2 Sailing algorithms and autonomous control

The SailBot is intended to function as an autonomous sailboat. To achieve this goal, previous teams have implemented a high-level/low-level pathing solution which creates navigation maps from maps of the area the boat is sailing in, and finds navigable paths between desired points by combining positional data with environmental data such as wind speed and direction. Once a path is calculated, the low-level navigation system drives the boat along the path by adjusting the trim tab, rudders, and ballast to achieve an optimal (for the previous hull) heel angle of 20 degrees.

2.3.3 Telemetry dashboard

The previous telemetry system used two radio-to-ethernet bridges, one on the boat and one on shore, as well as a 900Mhz radio antenna to establish communication between the boat and a computer on shore. The telemetry information is then displayed in a web page running locally on the shore PC.

2.3.4 Computer vision

In the SailBot competition, colored buoys are used to mark courses which sailboats are intended to navigate autonomously. To achieve this, previous teams implemented a computer vision system using a YOLOv5 model to detect orange buoys. This system was never fully integrated into autonomous navigation, so the potential of this approach was never realized. Additionally the performance of this system was very poor, often running at 1 FPS, too low for our application.

3 | Design Objectives

To address the current issues with the project, we have identified four main technical focuses for our project work. These include:

- A hull redesign based around a more fit-for-purpose hull
- A re-implementation of the electrical systems in one or more PCBs
- A systems and software redesign focused around communications reliability and failure detection and recovery
- Integration of existing autonomous functionality, and implementation of new functionality to achieve our autonomous navigation goals

To work towards these technical goals, we identified a set of smaller design tasks which must be completed:

- Create a New Hull for the Boat
- Increase Interior Accessibility
- Improve Airmar Mount
- Transition Boat to PCB Mounted Electronics
- Add Battery Management System
- Migrate to a more modern software platform
- Improve platform stability and add failure recovery
- Validate existing autonomous modes and make improvements
- Replace radio telemetry with 4G
- Integrate computer vision into navigation system
- Improve debugging and visualization of boat processes
- Improve efficiency of boat software
- Create documentation

4 | System Designs

4.1 Mechanical Design

4.1.1 Hull Production

The previous hull design was built in 2017 and based off of the Imoca 60. This hull is a planing hull meant for ocean races. This means it is designed to gain enough speed to ride above the water for long downwind legs. This was an issue because the boat could never reach high enough speeds to plane which means it was displacing a proportionally larger amount of water than was intended. The previous hull also has very little accessibility to reach inside. Because of this the new design is better suited to how the robot currently functions.

The design is based on the International America's Cup Class (IACC) hull. Since this is a formula based class, each hull is slightly different. Because of this, the specific hull analyzed for comparison was New Zealand's NZL 32 *Black Magic* and the CR 914 which is a model boat designed after the IACC hulls*.* The difference between the IACC hull and CR 914 is that to scale, the CR 914 is deeper and wider (Figure 3).

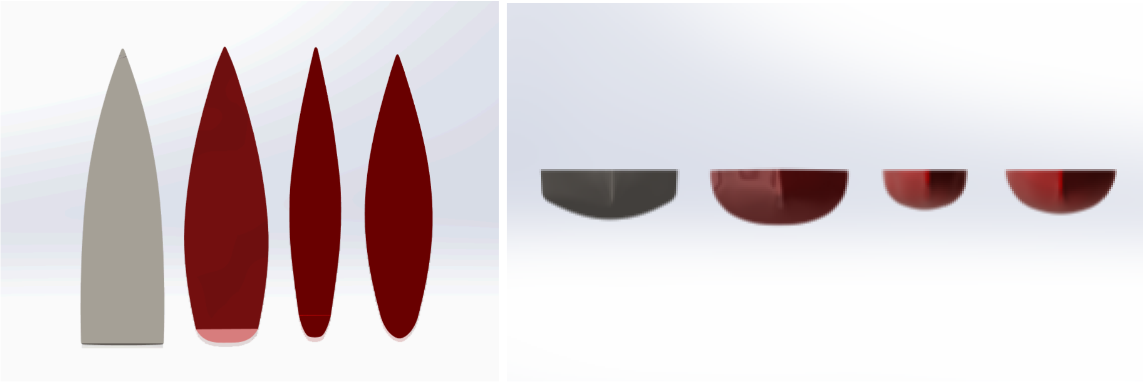


Figure 3. Hull models left to right: Previous hull, Current hull, IACC, and CR 914

The buoyancy of the design which dictates how well the boat floats was analyzed. It was predicted that the hull would displace 45 kg of water which is 0.045 m^3 of the volume. At scale, the IACC and CR 914 hulls are about half the volume of the previous hull which means they sit a lot lower in the water. In order to improve this, the design of the IACC was scaled to be much deeper and wider in order to have a higher waterline. A comparison of these various hulls are shown in Figure 3. These scaled dimensions for the new hull are advantageous in order to add structure, fit all necessary components inside, and increase stability.

The stability of the hull was also analyzed by analyzing the righting moment of the boat at various heel angles. The righting moment determines how susceptible the boat is to rolling. A large righting moment arm will return the boat to its regular upright position quickly. Small righting moment arms will do the same, but they restore the boats heel angle more slowly meaning that the boat will be more susceptible to rolling whether from wind or waves. Negative righting moments pull the boat in the opposite direction causing the boat to capsize. For stability, it is important to have a high righting moment so that the boat is difficult to roll from its upright position and will more quickly return to its upright position in adverse conditions.

The righting moment of the hull can be calculated as the distance between the center of mass and center of buoyancy at a given heel multiplied by the displacement of the hull (45 kg). The center of mass and center of buoyancy were simulated through SolidWorks and the distance between them were measured. This process was done for the moveable ballast centered and its greatest extreme. The purpose of the movable ballast is to control the righting moment as the boat heels. Typically this would be to increase the righting moment and keep the boat flat, but in this case the ideal heel will be 20∘ based on the lead distance between the mast and keel as in Section 4.1.2. Analyzing the righting moment with the ballast at these two points, provided the range of the righting moment as the ballast moved across the deck.

The righting moment was plotted against the angle of heel to create a stability curve which is shown in Figure 4. The high righting moment means that the boat is very stable while sailing and not affected as much by the variability of external factors like waves. It is important to note that the typical sailing range for this boat is between 0∘ to 40∘. This shows the most common righting moments affecting the hull at a given time and demonstrating that the ballast is able to contribute to increased stability of the hull. Another important aspect is how when the ballast is centered, the righting moment never becomes negative. This shows that when the ballast is centered, the boat will always return upright when it capsizes. Even when the ballast is at its least stable position, the boat will return upright when it capsizes unless it is overturned to 160∘.

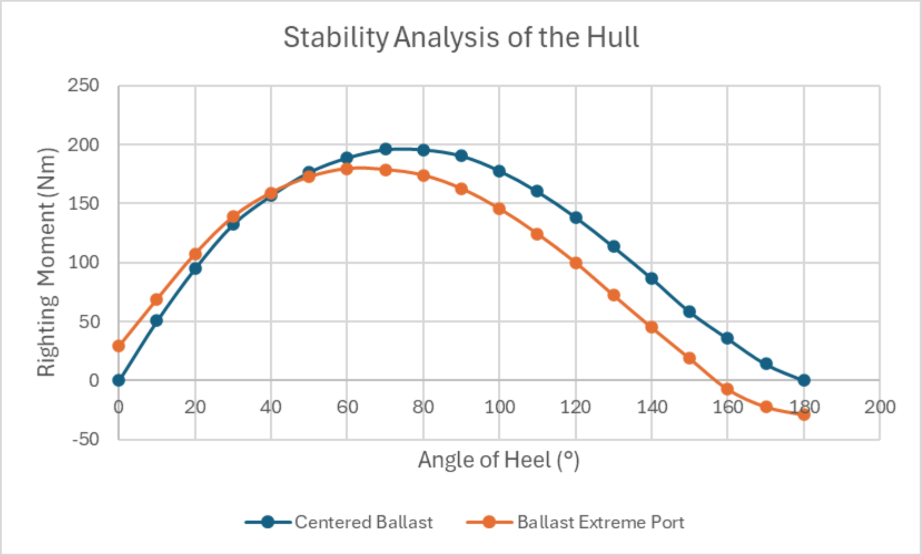


Figure 4. A stability curve of the hull with the ballast centered

One of the main changes from the old hull to the new hull is that the stern of the boat flows back as opposed to having a 90° angle. Because the old boat was heavier than expected, the hull sat deeper in the water and created a significant amount of drag at the stern. The new design rectifies this by tapering the sturn so the water is able to more easily flow off the stern of the boat. In order to see the difference between each hull, a flow simulation was done at 2 m/s (Figure 5). This shows how the new design eliminates turbulent flow around the stern area, substantially reducing drag as the boat moves through the water.

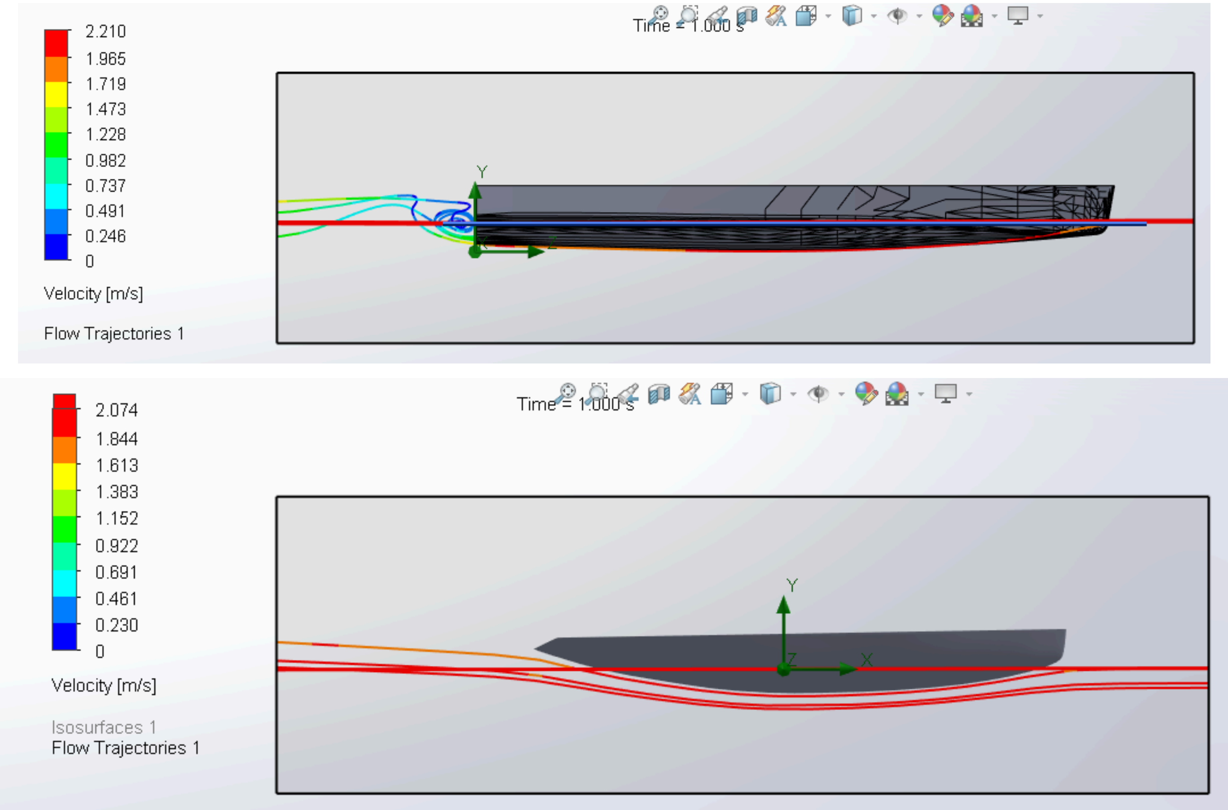


Figure 5. A Flow Simulation comparing the previous (top) and current (bottom) hull designs.

The new design is made from carbon fiber which is lighter than the old fiberglass material. It is also more rigid and durable. The hull was made in partnership with New England Boatworks. The carbon fiber was laid up in a male mold. This mold was made of MDF. Once the layup was complete, it was vacuumed bagged to hold all of the layers together and remove excess resin.

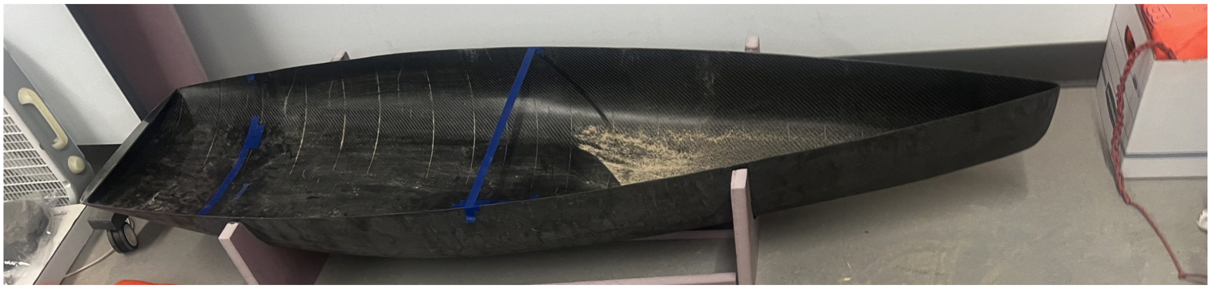


Figure 6. The shell of the hull as received from New England Boatworks

In order to complete the hull, the internal structures detailed in Section 4.1.2 such as the ribs and keel box were epoxied in. The deck was epoxied in and attached to the ribs and hull. Two layers of carbon fiber were laid up on top to reinforce the deck and seal the edge to the hull so there were no gaps.

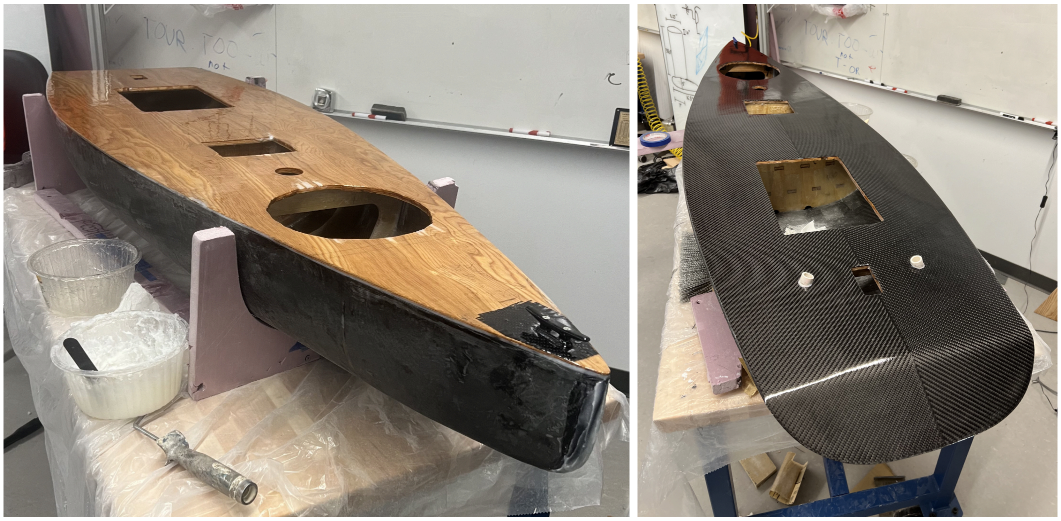


Figure 7. Before and after the carbon fiber layup

In order to complete the hull, the bottom was faired to remove any ridges leftover from the vacuum bag, and painted with epoxy to seal the exposed carbon fiber. It was then given finishing coats of primer and bottom paint. Finally, all systems were integrated into the hull. Modifications to these systems are detailed in the subsequent sections.

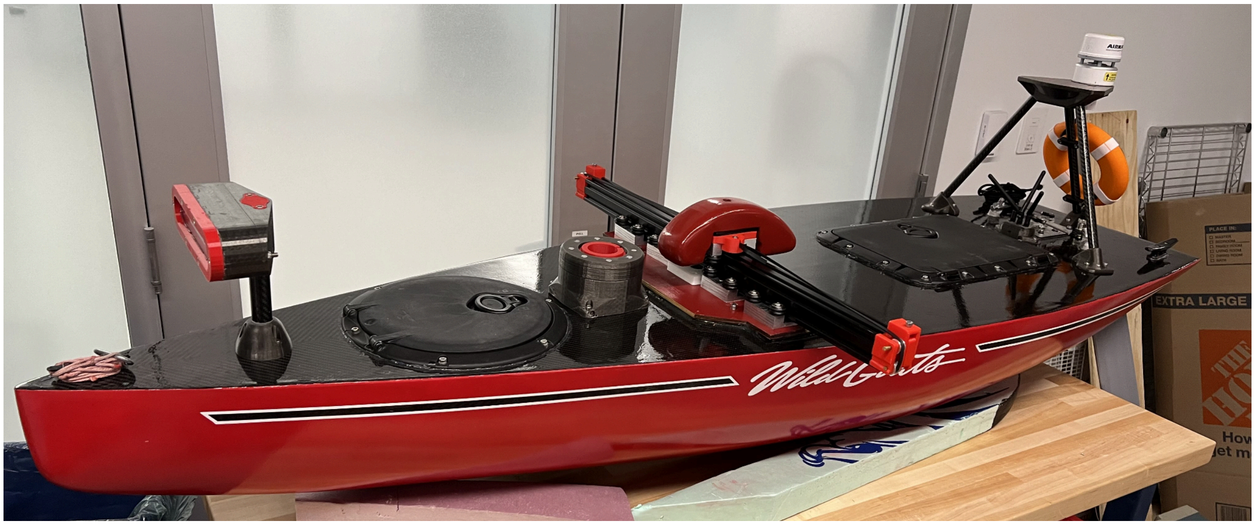


Figure 8. The final production of the hull

4.1.2 Internal Structures: Ribs, Keel Box, and Mast Step

In order to reinforce the hull, the deck is made of plywood with a carbon fiber layup going over the deck. This deck is supported by ribs that are CNC routed from plywood in order to conform to the curvature of the hull (Figure 8). The purpose of the ribs is to support the hull and the components on the deck.

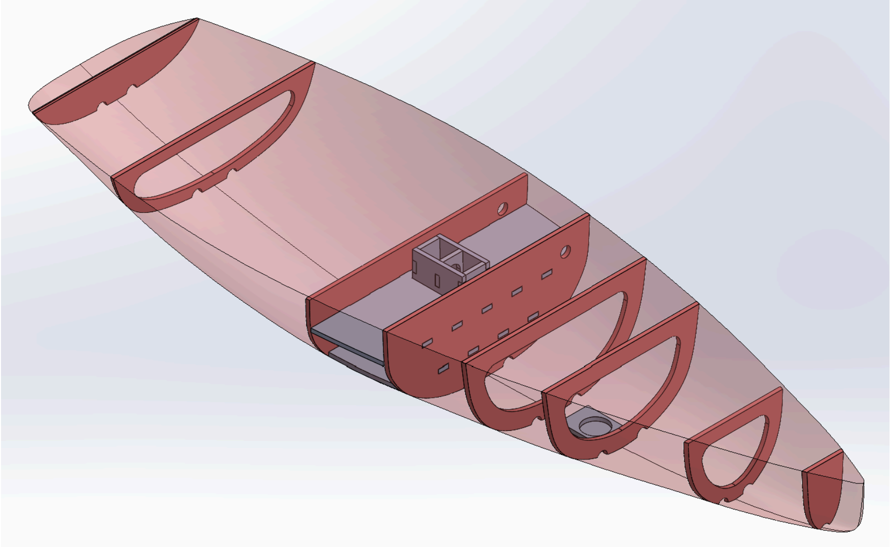


Figure 9. The internal structure of the boat featuring the ribs, keel box, mast step, and deck layout sketch

The internal structure of the hull has three sections: the bow, middle, and stern. The bow and stern have access hatches and house all of the electrical components. The middle section is generally sealed off from the others since it houses the keel box and mast step which hold the keel and mast in place respectively. This area is reinforced to withstand the stresses of mounting the keel and mast securely.

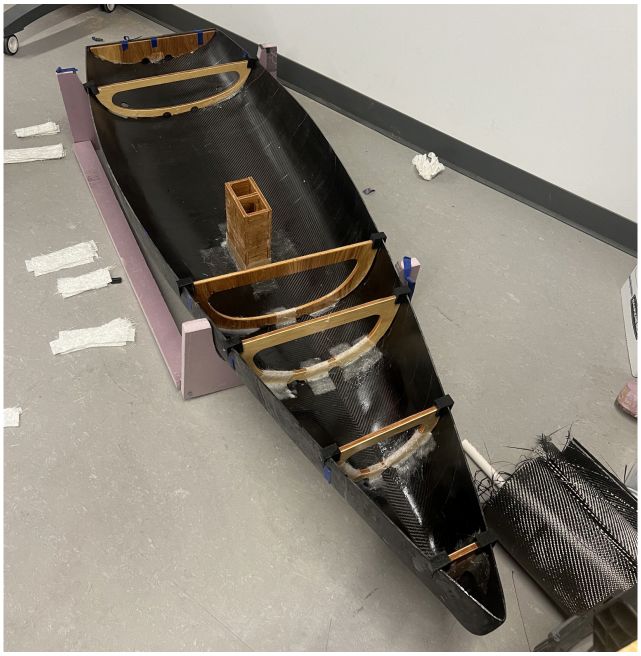


Figure 10. Rib installation into the hull

The keel of the boat has a mass of about 17 kg, therefore the keel box has to be able to support that amount of mass. Currently, the keel is attached at one point on the deck. In this new hull, there is a smaller main box that tightly goes around this point in order to support the weight of the keel (Figure 11). There is also an arch in the middle of the main box to help distribute the forces from the mounting point and allow for an option to add a second point of contact for the keel in the future. The box is held together by finger joints which are epoxied using West Systems Epoxy. There are two flat pieces as well that ensure the main box stays together and attaches to the main ribs with finger joints.There is one pipe that goes through this section in order to connect electronics from the bow and stern of the boat as well. This system was tested prior to securing it to the hull by assembling the keel box and adding 17 kg of weight at the same point the keel would be attached. As it was easily capable of withstanding this test, we determined that it would be able to hold the keel.

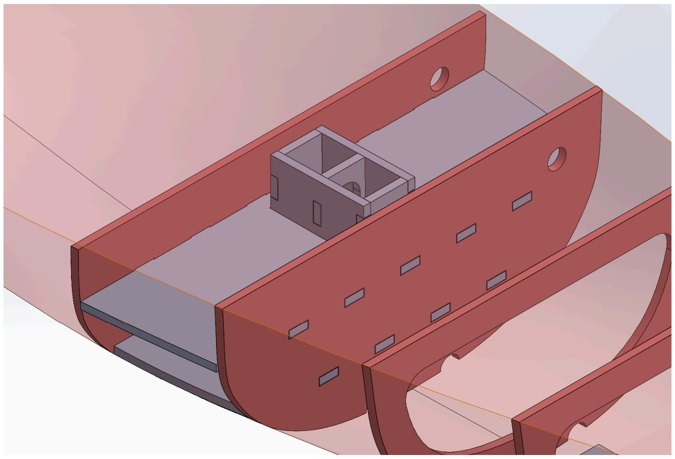


Figure 11. The keel box designed to be between 2 ribs

The mast step (Figure 13) of the boat was designed to withstand bending due to transverse forces on the mast caused by the wind. These forces generate lift, which is transferred through the mast to the hull in order to propel the boat forward. Analysis showed that at a top wind speed of 15 m/s, there would be a load of 81 N on the mast calculated as drag. Using static equilibrium equations, the mast step needs to withstand 738 N and the deck needs to withstand 847 N at a higher extreme wind speed. The mast step will be attached to the inside of the hull using epoxy with a PVC pipe sitting in the step going up and through the deck for a second point of contact and to distribute the force from the sail. The bottom of the mast pipe has a bearing that the mast attaches to and a collar around where the deck mount is. This allows the mast to be free rotating while also having the structural strength to hold the mast in place.

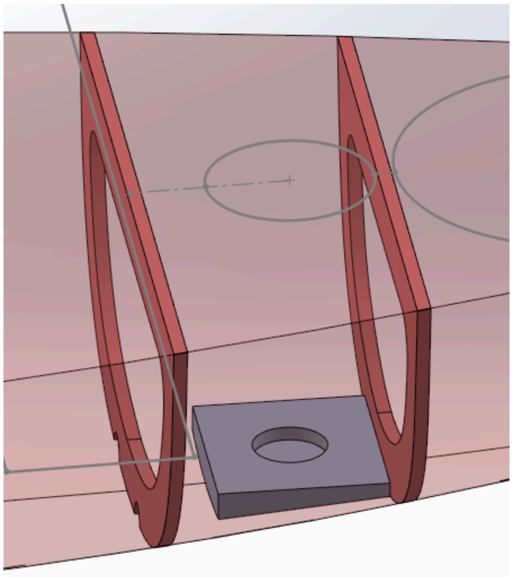


Figure 12. The mast step attached to the bottom of the hull

4.1.3 Keel and Mast Placement

Location of the mast step and keel were determined using equilibrium equations particularly around the yaw moment which controls balance port and starboard. This is important so that the boat is steerable and limits lee helm, uncontrolled turning away from the wind, and weather helm, uncontrolled turning towards the wind are minimized.

In order to limit lee and weather helm, yawing moment should equal 0. The forces acting on the system are the force of lift generated by the sail and the reaction forces from the keel (Figure 13). The lift can be separated into the y direction force moving the boat forward and x direction force perpendicular to that. The forces are generated from the center of efforts which is at the mast and 25% back from the front of the keel. Another important dimension is the total distance between the center of effort of the sail and keel. The yawing moment can be measured from the center of effort of the sail or keel.

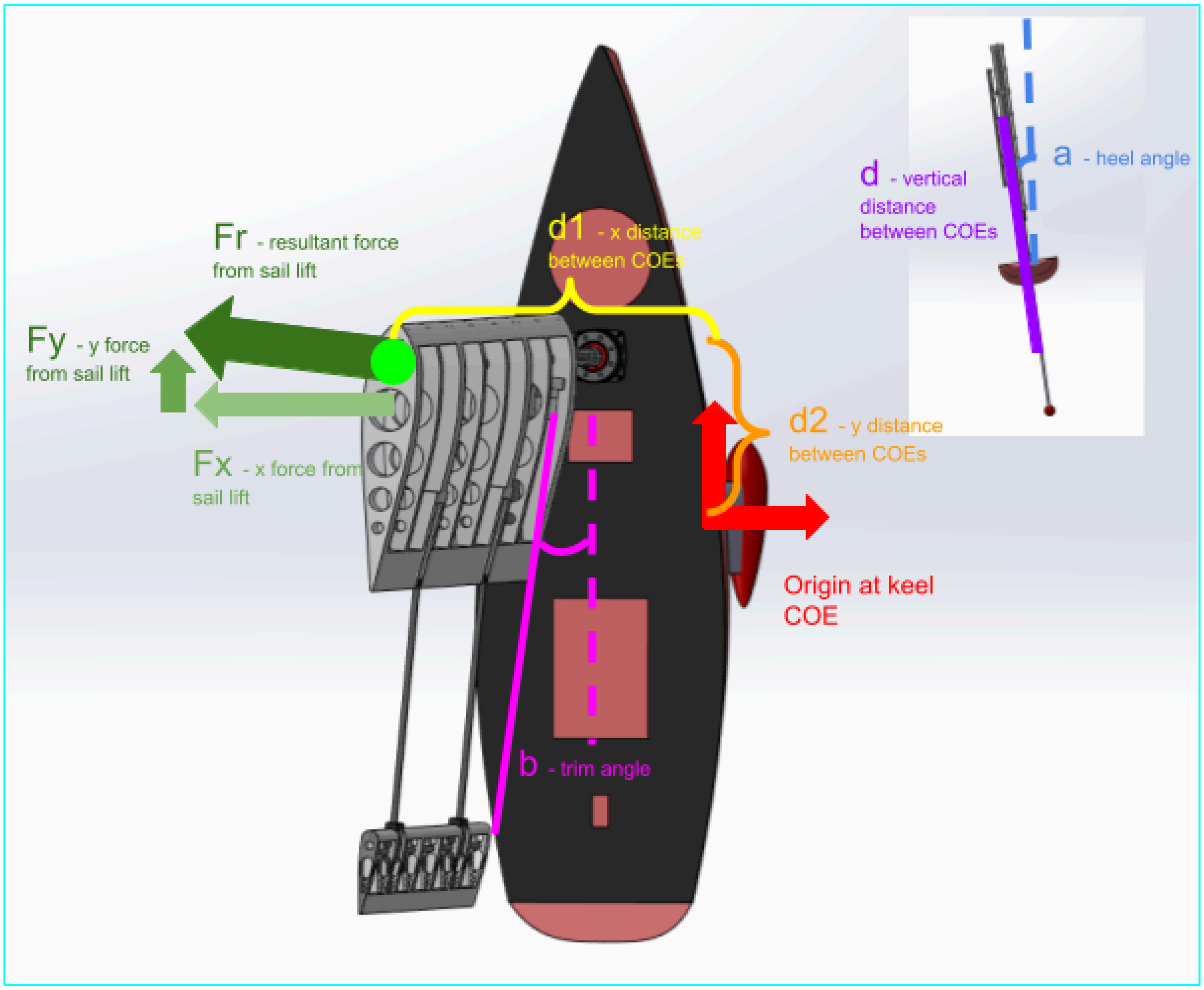

d - vertical distance between COEs
a - heel angle
Fr - resultant force from sail lift
d1 - x distance between COEs
Fy - y force from sail lift
Fx - x force from sail lift
d2 - y distance between COEs
Origin at keel COE
b - trim angle

Figure 13. A free body diagram of the forces on a sailboat

To determine the distance, the following formula was developed:

$$M = 0 = F_x * d_2 - F_y * d_1$$

$$d_2 = d_1 \frac{F_y}{F_x}$$

$$F_y = Fsin(b),\ F_x = Fcos(a)cos(b), d_1 = d * sin(a)$$

$$d_2 = d \frac{sin(a)sin(b)}{cos(a)cos(b)}$$

$$d_2 = d * tan(a)tan(b)$$

Using a heel angle (a) of 20° and trim angle (b) of 30°, the lead was calculated to be 0.48 m (18.9 in). The previous hull was generally stable and the lead on the previous hull was 0.22 m (8.7 in). The formula above was plotted based on the heel and trim angle (Figure 14). From this, it was shown that lead is independent and stability depends on the heel at a given trim. From this, the pitch (front to back) moment was analyzed to balance the floatation of the boat from the keel and a lead of 0.29 m (11.4 in) was chosen.

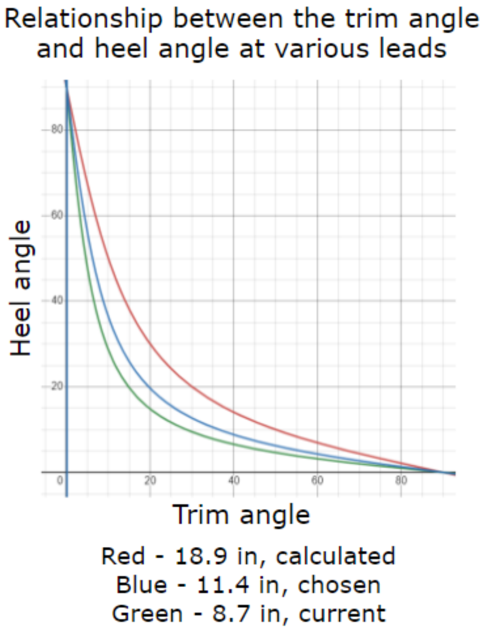


Figure 14. The relationship between the trim angle and heel angle at various leads.

4.1.4 Rudders

Similar to the previous design, the new hull is controlled using dual rudders. This was chosen because of the curvature of the stern and if the boat is on its side, one rudder may be out of the water or in a position that it will not have control. The rudders are located just below the waterline in order to have the most water flow over them and therefore the most control. To integrate the previous system into the new hull while preserving the rudders' full range of motion, the linkage length and post material was changed. In this design, the rudders are closer together, so the threaded rod linkage was modified to be parallel and reach the required range. To get the range needed, the rod length was modified to 2.75 in.

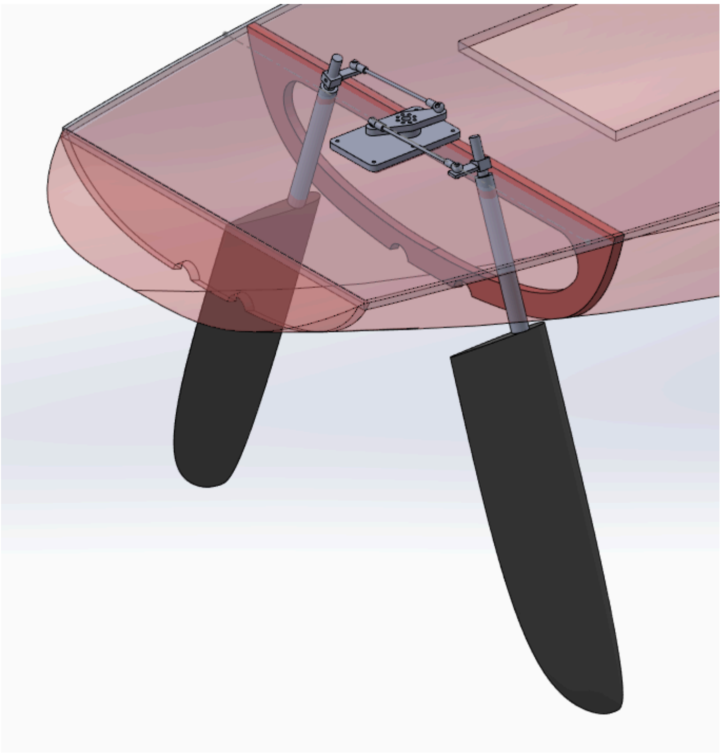


Figure 15. Design of the rudder system

The other improvement was made was the material of the rudder posts and bushings. The rudder posts hold the rudders in place and the bushings allow the rudders to turn easily in the post tubes. The previous design used metal tubes and bushings which would be more likely to corrode in water being in contact with dissimilar metal and were heavier. The materials used in the new design are PVC tubes and nylon bushings. These materials do not corrode in water and are lighter.

4.1.5 Increasing Interior Accessibility

The two current hatches provide little access to the interior of the boat. Although all components can fit through the hatch, it is difficult for an engineer to work, fix, and see in the interior of the boat. To fix this issue, we installed a larger circular hatch towards the bow and a larger rectangular hatch towards the stern of the boat. The circular bow hatch chosen is a Hobie Round Twist N Seal Hatch 8 inch Kit. This gives us an extra 0.75 inches to work in the interior towards the bow of the boat. Due to the shape of the hull, this was the largest water-tight hatch that could still be installed towards the bow of the boat. The rectangular stern hatch chosen is a Hobie Kayak Rectangular 11.25 x 15 inch Hatch Kit. Just as with the circular bow hatch, this is the largest water-tight hatch that can be installed towards the stern of the boat. This hatch significantly increases the port size, and therefore increases the available working space in the interior towards the stern of the boat. Since most of the electrical and mechanical systems that are located in the interior of the boat are located towards the boat's stern, the accessibility at this hatch is the most critical.

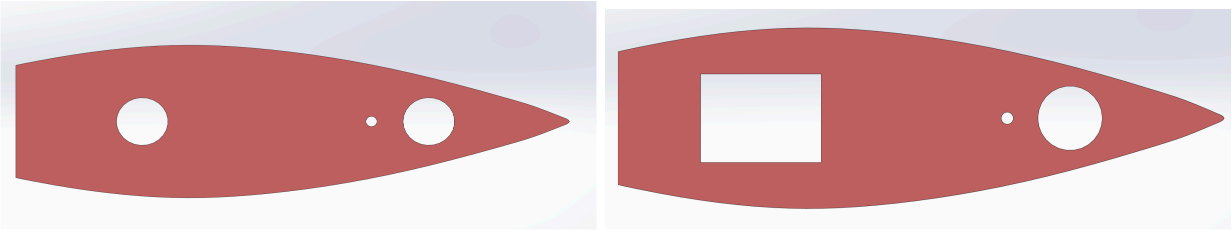


Figure 16. Comparison of previous and current hatch sizes respectfully

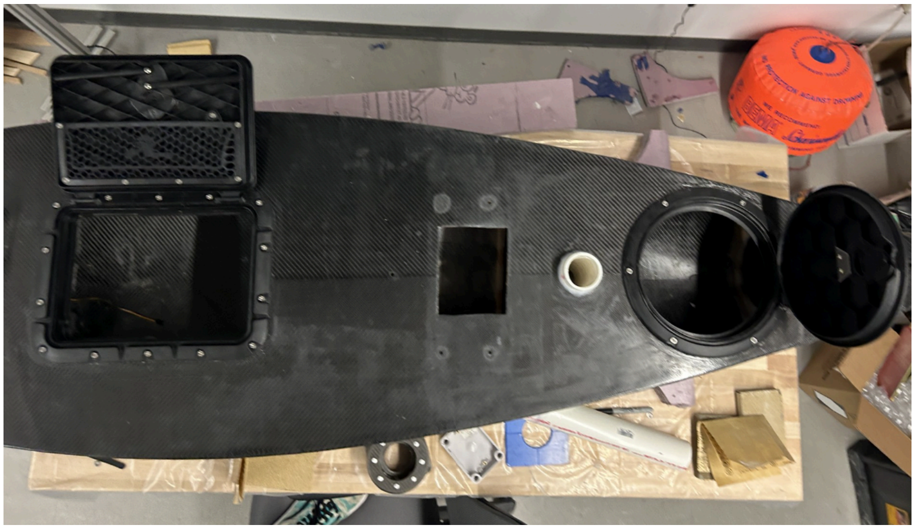


Figure 17. New Hatches Installed on the New Boat Deck

4.1.6 Redesigned Airmar Mount

The previous Airmar mount had a few problems that did not suit our new hull design and project goals. One issue is that it is not secure enough to keep the Airmar from moving. Any movement in the Airmar is detrimental to the controls of the boat as it is not able to properly determine where it is going. The new Airmar mount addresses this issue by having slots that screws fit snugly into, restricting the movement of the Airmar when tightened.

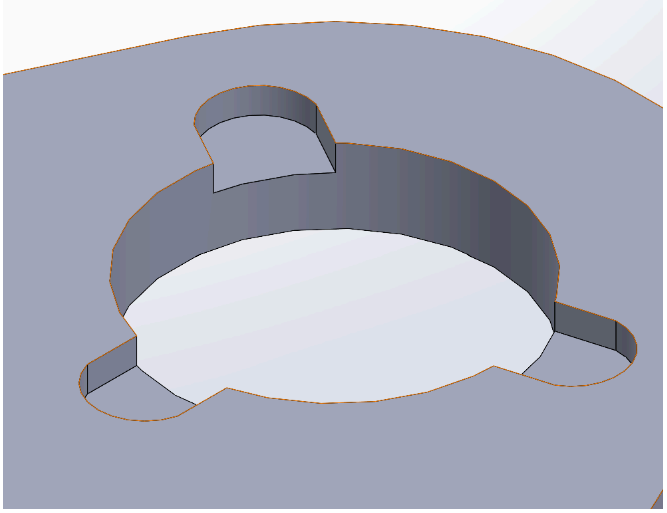


Figure 18. Airmar Retaining Slots

Another issue with the current Airmar mount is that, even though it fits the new deck layout, it does not fit in its optimal position: positioned directly forward.To fix this, the new Airmar mount has 60◦ brackets that attach to the deck, accompanied by 1 ft carbon fiber poles attaching the 60◦ brackets to the top bracket of the Airmar mount. The top bracket of the Airmar mount has holes for the Airmar and a radio antenna that was on the previous Airmar mount. Currently the radio antenna mount is unused as the boats telemetry no longer relies upon that system. It is fitted with a plug when not in use, but allows for the reincorporation of the antenna if desired in the future.

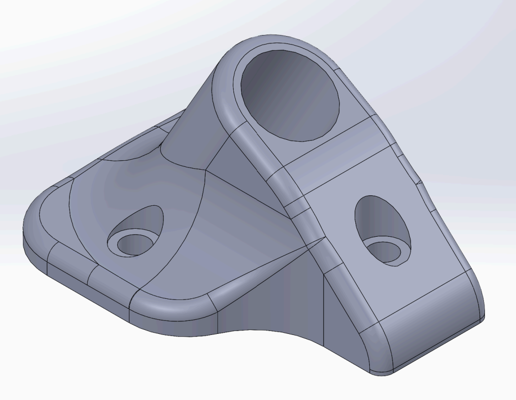


Figure 19. 60◦ Bracket

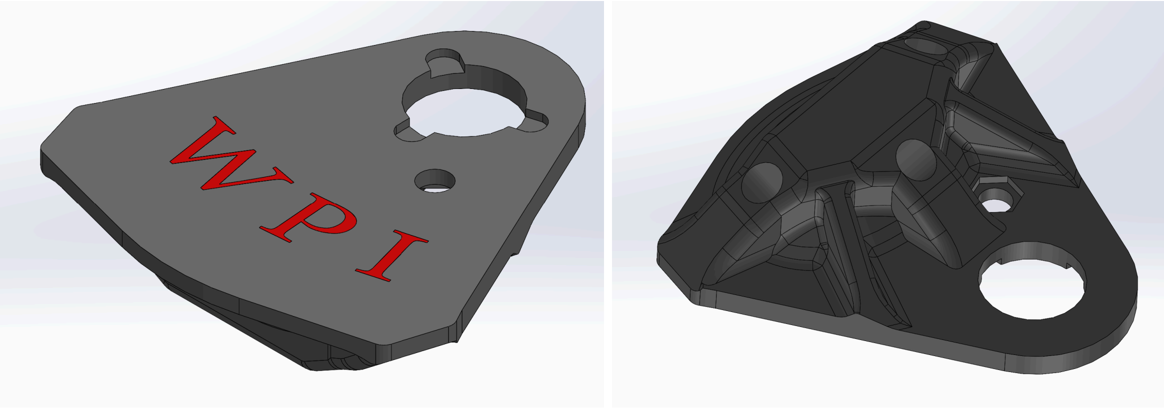


Figure 20. Airmar Mount Top Bracket

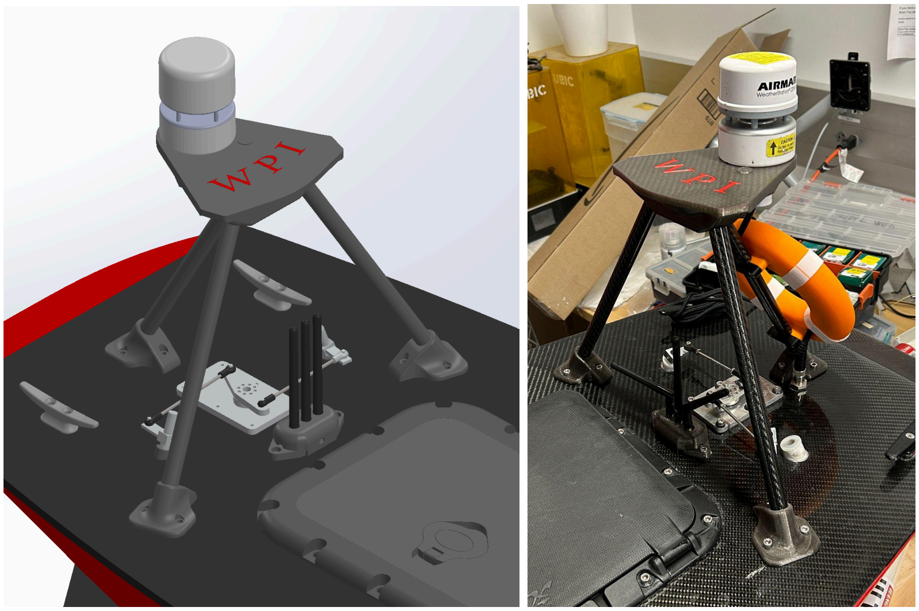


Figure 21. Airmar Mount on Deck

4.1.7 ZED Camera Casing and Mount

The ZED Camera is used for object detection, so that the boat can detect buoys in the water. This camera is not water resistant and in order to make sure it cannot get wet, a case was designed. This case is four 3D printed pieces with an acrylic window for the camera.These pieces contain a channel to allow for the USB cable from the ZED Camera to slide through and access the interior of the tube without exiting the mount. It is bolted together and is sealed with silicone in between each piece. The inside is coated with West Systems Epoxy to ensure no water can get inside.

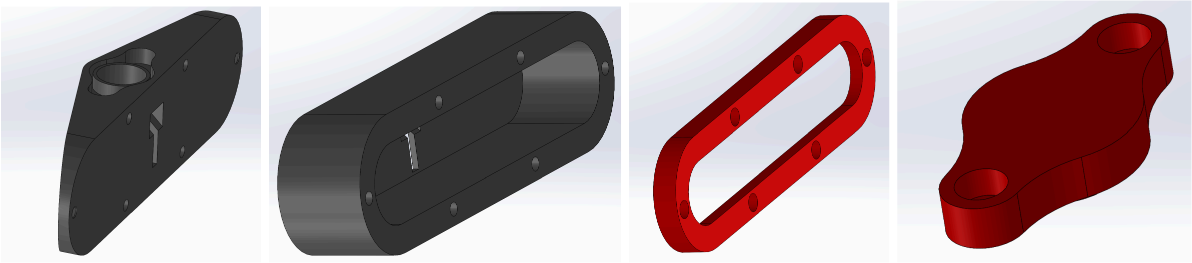


Figure 22. ZED Camera Casing

The casing is mounted on top of a carbon fiber tube and a 3D printed mounting piece. One important feature is that the camera mount on top of the tube features a gap for the ZED wire which goes down the pole and directly into the hull. This is to help improve the waterproofness of the connection between the ZED camera and the electronics.

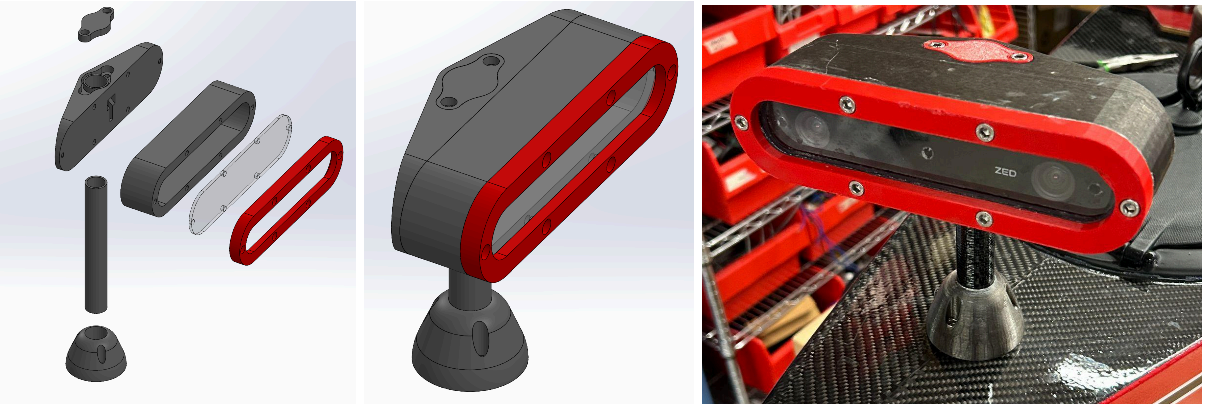


Figure 23. ZED Mount

4.1.8 Mast Bearing Mount

The Mast Bearing Mount holds the bearings that allow the mast to move freely and smoothly. It is composed of two 3D printed pieces, eight shafts, and 16 bearings. It also has a three piece 3D printed collar around the mast to ensure the mast can only spin freely and stop the metal bearings from damaging the mast. Previously the mount was secured to the deck with gravity alone. The new design features four bolting locations that securely attach the mount to the deck.

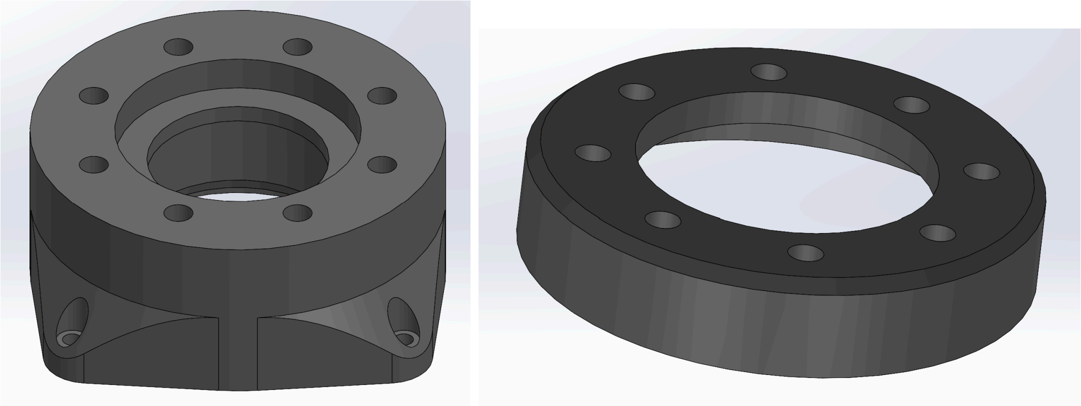


Figure 24. Mast Bearing Mount Components

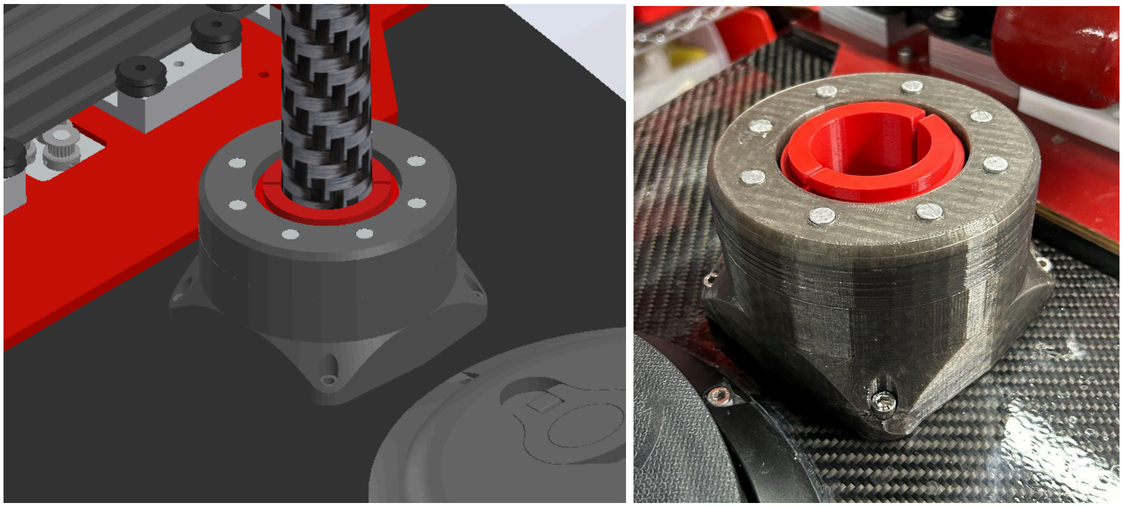


Figure 25. Mast Bearing Mount

4.1.9 WIFI and 4G Antenna Mount

The WIFI and 4G Antenna Mount holds one WIFI antenna and two 4G antennas. It creates a waterproof channel for the antenna wires to get into the interior of the boat. It is composed of three 3D printed pieces. The pieces are compression fit together with silicone sandwiched in between the pieces to waterproof the connection. Two bolts are used to secure the whole mount to the deck.

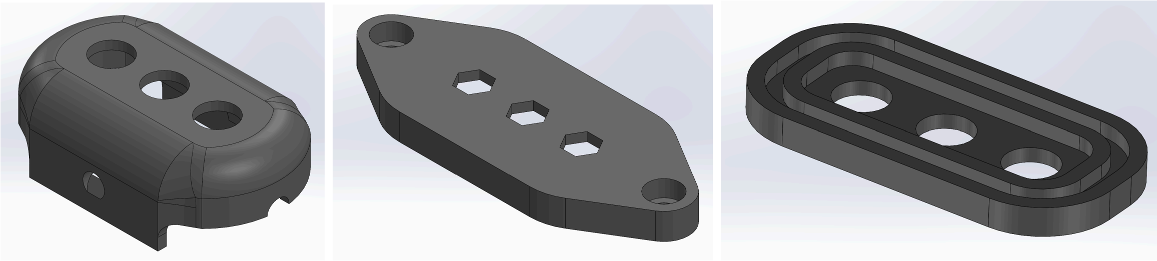


Figure 26. WIFI and 4G Antenna Mount Components

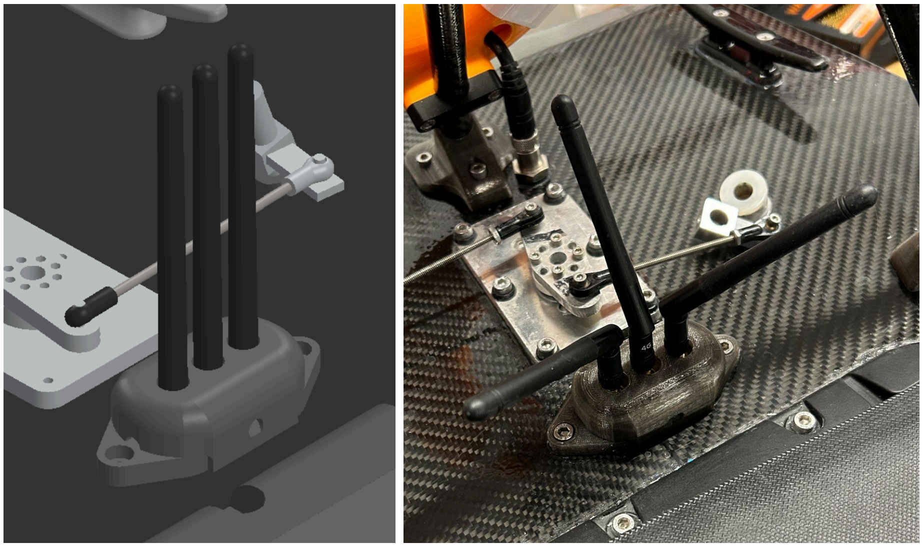


Figure 27. WIFI and 4G Antenna Mount

4.2 Electrical Design

4.2.1 Transition Boat to PCB Mounted Electronics

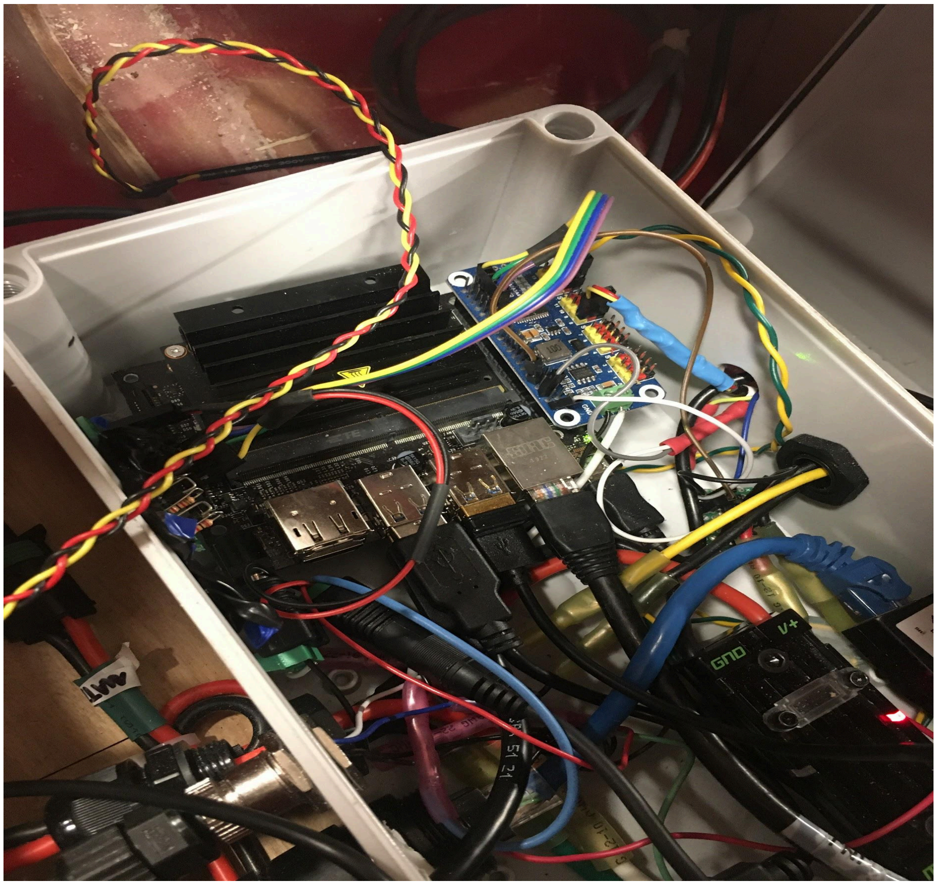


Figure 28. SailBot's old Electrical Control Box

The old boat's electrical system was disorganized and unreliable. Without any clear organization or overarching design, the electronics were assembled piecemeal by multiple past teams. This resulted in lacking organization throughout the boat and made accidental short circuits common and difficult to diagnose. Additionally, many of the connections were poorly made with no strain relief or other protections. This resulted in electrical connectivity faults when the boat is rocked or hit in a certain way. The new design rectifies this by utilizing a Printed Circuit Board (PCB) to house the boat's electrical components. A PCB eliminates the need to organize wires and dramatically reduces the likelihood of electrical faults.

Additionally, we have consolidated many of the peripheral electronics onto the PCB. Power distribution and overcurrent protection are now integrated into the PCB. This reduces weight and volume usage inside the boat, along with improving the watertight integrity of the electronic enclosure by reducing the amount of external connections.

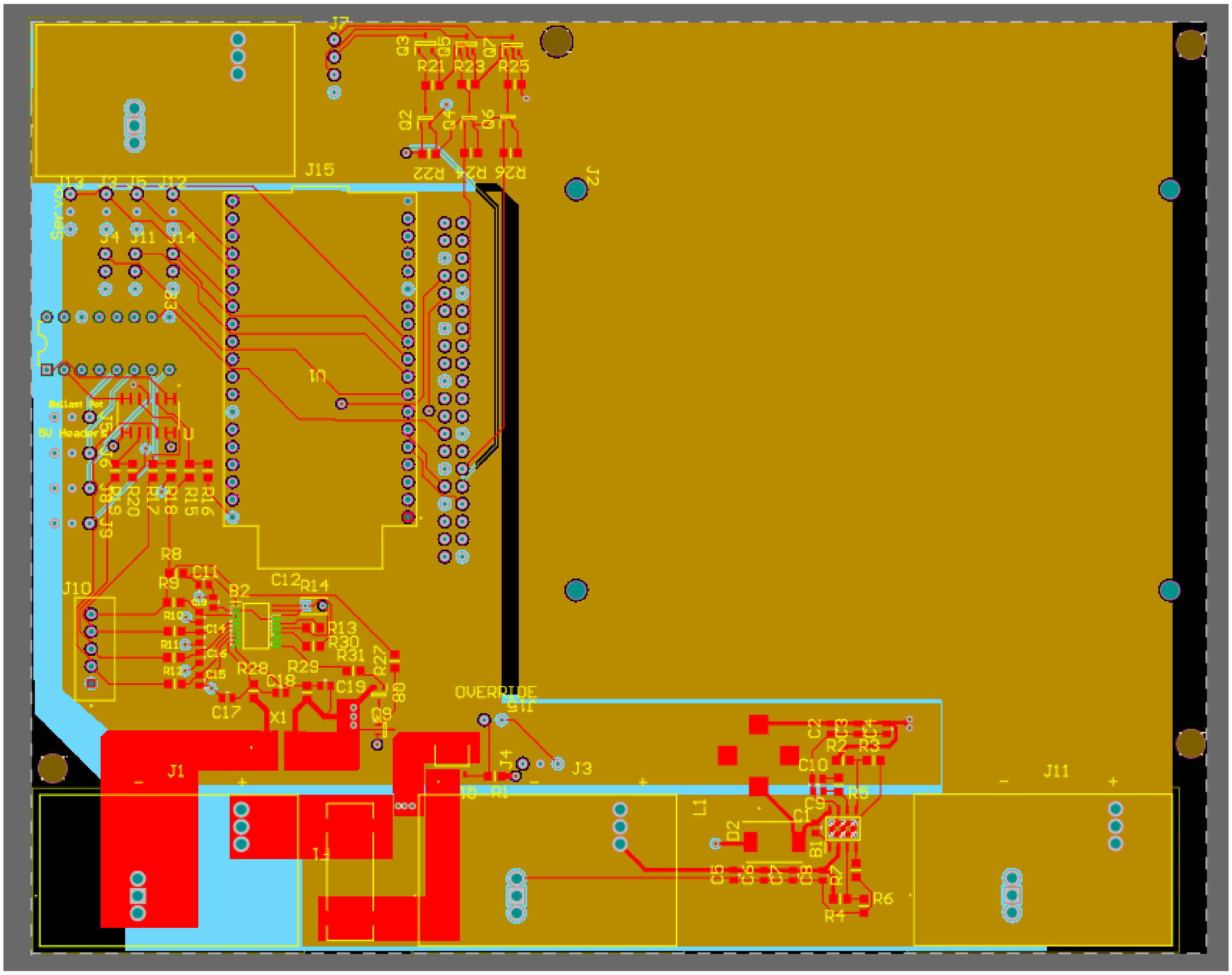


Figure 29. SailBot Main PCB with Power and Ground Planes Shown

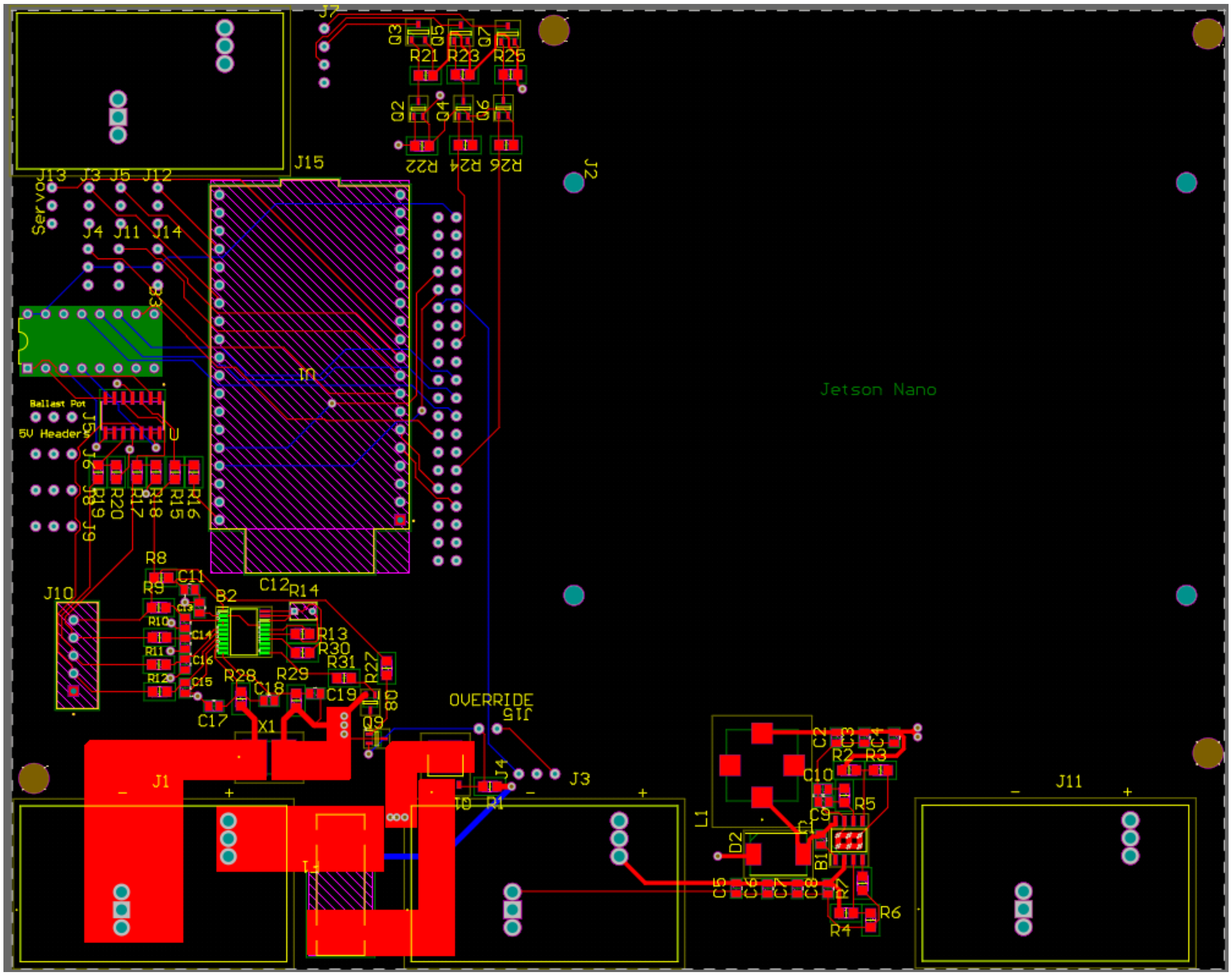


Figure 30. SailBot Main PCB with only Signal Layers shown

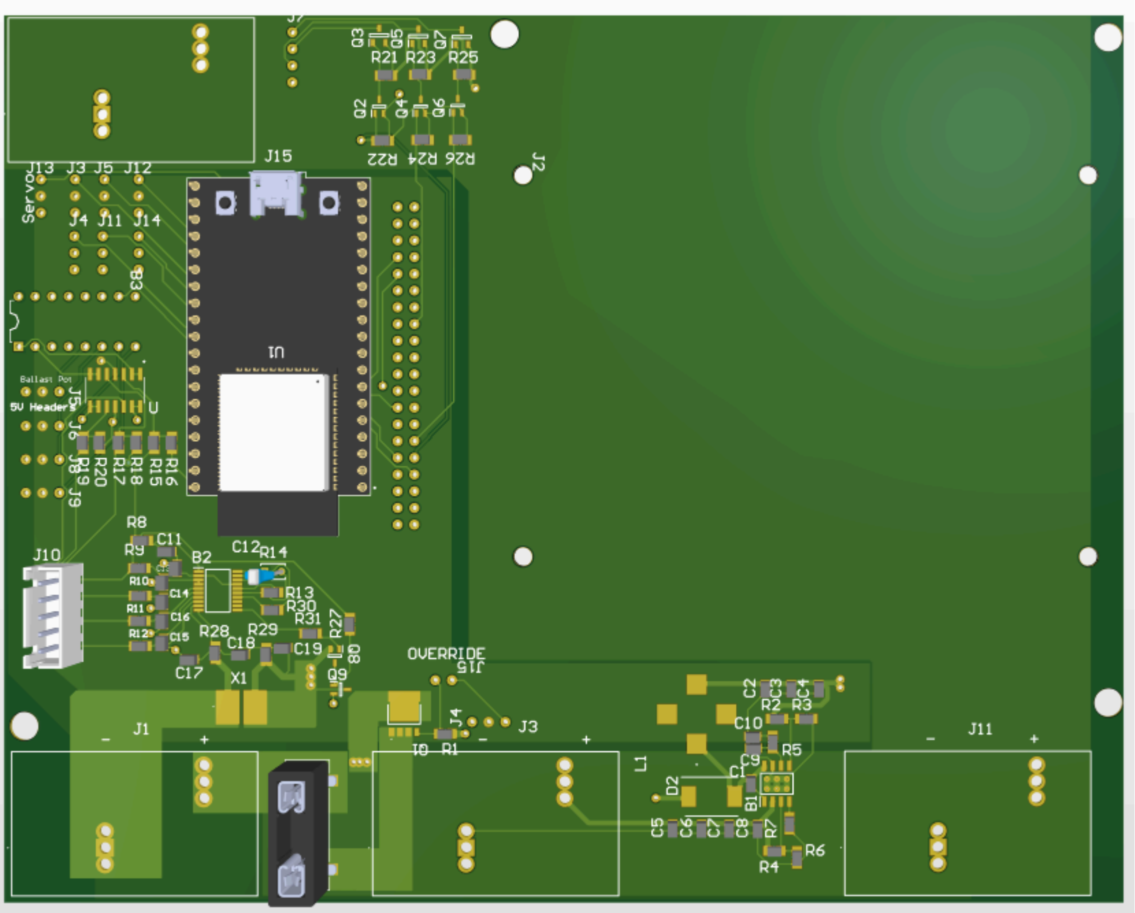


Figure 31. 3D Model of SailBot Main Control Board

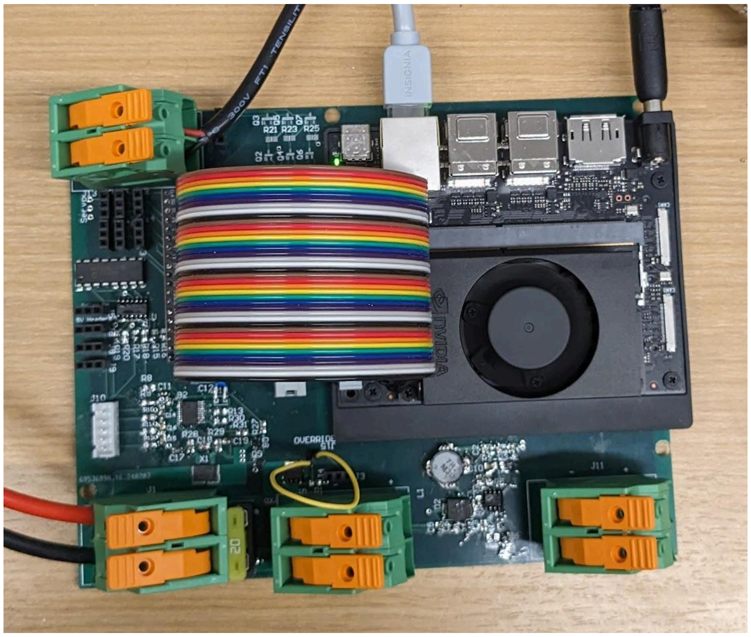


Figure 32. SailBot’s Assembled Main Control Board

The PCB for SailBot consists of 4 layers, a top signal layer, a power plane, a ground plane and a bottom signal layer. This layer stack configuration was chosen to reduce the effects of parasitic capacitance and crosstalk between the signal layers. Additionally the ground plane allows for effective heat dissipation through thermal vias from surface mounted components.

The PCB contains a battery management system, 5V DC-DC regulator, control circuitry for the stack light that was not implemented last year, and connectors for the Jetson Nano and ESP32.

Power for the board is regulated by the BQ77904 4-5 cell battery protection IC. By plugging the balance connector from the LIPO pack into the board we are able to measure the voltage across each individual cell. The BQ77904 monitors this voltage and will switch MOSFET Q1 off if the voltage strays outside the 3-4.2 volt range. Additionally through the use of a current sensing resistor, the BQ77904 monitors for overcurrent faults and breaks the circuit if a short circuit is detected. There is also a backup 20A blade fuse mounted to the board. It serves as an additional means of overcurrent protection when the boat is powered from a bench power supply for development purposes.

The battery level monitoring circuit has also been improved from previous years. The battery level checking circuit utilizes an MCP3008 8 Channel ADC and a MCP 604 operational amplifier. Through the use of 4 voltage dividers, the battery cell output voltages from the balance connector are reduced to a lower voltage to not damage the level checking circuitry. Because of the high impedance of this circuit, the MCP604 operational amplifier is used in a voltage follower configuration to buffer between the voltage dividers and the ADC to ensure an accurate reading.

The MCP3008 communicates with the Orin Nano through an SPI communication bus. The MCP 3008 monitors the voltage of each cell of the LIPO, and the ballast potentiometer's position. The 3 unused channels of the MCP3008 are broken out into standard 0.1in 5V 3 pin headers to allow for future expandability.

Unlike the Jetson Nano previously used, the Orin Nano requires a higher input voltage, operating between 9-19V instead of the 5V of the regular Nano. There is also circuitry that prevents the board from being powered through the GPIO pins, leaving the barrel jack connector as the only option. The Orin Nano is powered from the 15V battery power supply, from a connector on the board. The Orin's GPIO pins are connected via a ribbon cable to a 40 pin header on the main board.The Airmar weather station is able to accept between 9-40V, and is powered the same way as the Orin Nano. Data is transmitted through the Maretron converter back to the Orin Nano's USB port.

Other logic circuitry is powered by a TPS54560BDDA step down DC-DC converter. This device takes the power from the battery, which can vary between 12-16V, and regulates it to a steady 5 volts to power other logic circuitry onboard. The TPS54560BDDA can provide up to 5A at 5V, which is well above the max consumption of the board. There are also 3.3V devices as the GPIO pins on the Jetson and ESP32 are not 5V tolerant. For devices that attach directly to the GPIO pins, the internal 5V-3.3V regulators onboard the Orin Nano are used to power those devices.

4.2.2 Add ESP32 Coprocessor

Although the Jetson Orin Nano is a highly versatile platform, it is not adept at generating PWM signals. Additionally, the code used to communicate with the trim tab has poor compatibility with the ROS infrastructure on the boat. To rectify these issues we added an ESP32 microcontroller as a coprocessor. The ESP32 allows for the websocket code that communicates with the trim tab to be offloaded from the Orin Nano. The ESP32 generates PWM signals for the ballast motor's Talon SRX, and rudder servo motor. Many of the spare GPIO pins on the ESP32 are broken out into headers on the board to allow for future expandability. The ESP32 is capable of communicating with the Orin Nano via a serial connection.

4.2.3 Trim Tab Redesign

SailBot has a separate small electrical system built into its wingsail to allow for control of the sail's trim. This circuit has 3 main functions:

- Wirelessly communicate with the boat to receive desired trim updates
- Monitor the relative wind direction to determine the appropriate trim tab angle to achieve its goal
- Use a small servo motor to actuate the trim tab

The trim tab was redesigned last year, and although the redesign was a vast improvement in electrical reliability, not all of the design goals were met. First, the Bluetooth Low Energy (BLE) connection to the boat was often unreliable. Secondly, the desired functionality of being able to charge and discharge the internal battery along with programming the internal microcontroller with no electrical reconfiguration was not met. This issue was caused due to an interaction between the USB port on the Arduino MKR1010 Wifi and the relay responsible for switching the board into charging mode. If the board was directly switched from charging to discharging, the integrated LIPO would supply 5V to the Arduino's 5VIN pin. However, this pin was also connected to the USB port responsible for charging and programming the board. When 5V was applied to the 5VIN pin, the board would get a potential of 5V over its charging terminals. This would switch the board into charging mode and turn off the Arduino. With the Arduino depowered, the board would switch back into discharging mode, resulting in the Arduino power cycling many times per second.

Finally, the US digital MA3 encoder responsible for measuring the relative wind direction was wired improperly. The MA3 requires a voltage of between 4.5-5.5 volts to operate properly. However in the old configuration it was receiving only 3.3 volts.

To rectify these issues, the new trim tab has a variety of design improvements. To remedy the connection issues, the new trim tab uses a ESP32 instead of a Arduino MKR1010 Wifi. The ESP32 comes with a more powerful antenna, improving signal reception and transmission. Additionally the ESP32 now uses WIFI to communicate rather than BLE. The WIFI connection is much more stable than the BLE one, and grants the trim tab a tenfold increase in communications range compared to BLE. Although the WIFI connection does consume more power, (about 150 mA), given that the trim tab currently has a 2000mAh battery its lifespan is more than sufficient for our application.

To remedy the power cycling issue, a USB to Serial converter was used to segregate the power and data inputs to the microcontroller. The ESP32 is now programmed directly through the serial GPIO pins, and is no longer capable of powering anything through its USB port. Additionally the board now uses a USB-C connection which is more robust and capable of delivering more power compared to the old USB Micro B connector.

The MA3 digital encoder was powered directly from the battery so it is now running at its recommended level of 5V. However, the GPIO pins on the ESP32 are not 5V tolerant. To fix this issue, there is a level shifter utilized between the MA3 and the ESP32 to ensure the GPIO pins are not overvolted.

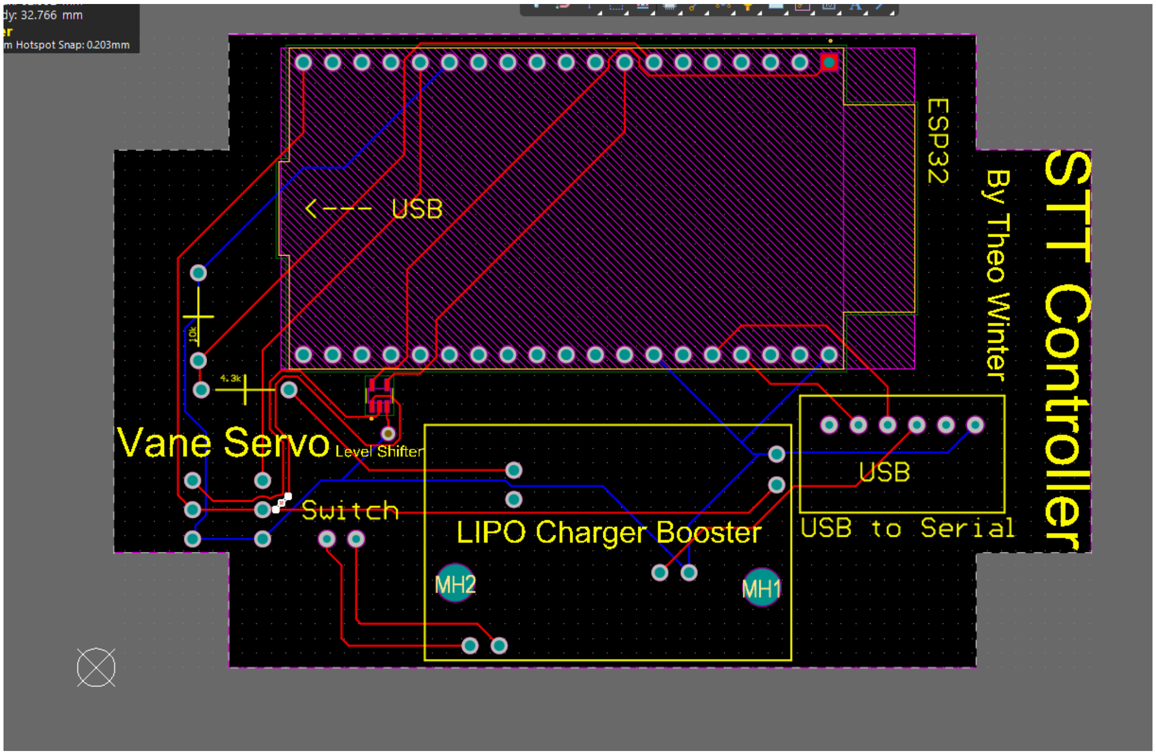


Figure 33. New SailBot Trim Tab Controller

4.2.4 Power Switch Assembly

SailBot was powered on via a magnetic switch and relay. Upon investigation, we found that the housing was melted and warped from thermal issues with the power switch. Additionally the switch was not functioning properly even without its thermal issues. The boat would turn on automatically when plugged into the battery, and although the switch could be used to turn it off, the boat could not be powered back on without disconnecting the battery first.

To rectify these issues, a new power switch board was designed. The new power board uses a latching hall effect sensor to detect the presence of a magnet above the deck. Instead of a relay, the new board uses a power MOSFET to control the flow of current. This was chosen for multiple reasons. Primarily, the MOSFET has no current draw to hold it open. An average 12v relay can have a coil current draw of around 100-150mA to stay active. Given the average power needed to run SailBot is around 500-600mA, this represents a significant additional power draw needed simply to turn the boat on. A MOSFET does not have this drawback. Secondly, since relays are devices controlled via electromagnetism, and since the switch is magnetically controlled, a sufficiently strong magnet could create unintended behavior in a relay when attempting to switch the boat on or off. Using a MOSFET means that the switching circuitry does not need to be separated from the magnetic sensing circuitry. To rectify the thermal issues, the board has been designed with thermal vias to conduct heat away from potentially problematic components, along with placing thermally active components on the far side of the board, where they are exposed to air inside the hull.

4.3 Software Systems

4.3.1 ROS nodes

SailBot now runs on ROS2 Humble Hawksbill middleware software, which is a publisher-subscriber framework intended to facilitate rapid development of robotic systems. ROS introduces the concept of “nodes” which are independent processes which execute their tasks while publishing and subscribing to information channels for inter-process communication. The WPI SailBot now runs ten nodes, described below.

1. The `airmar_reader` node, which reads information from the Airmar 220WX Weatherstation via the Maretron NMEA2000->NMEA0183 converter
2. `The esp32_comms` node, which manages communication with the ESP32-based trim tab control stack, and with the rudders and ballast PWM.
3. The `ballast_control` node, which runs the positional control algorithm for the sliding-rail ballast.
4. The `network_comms` node, which exposes a gRPC interface for telemetry clients to utilize
5. The `buoy_detection` node, which runs our CV algorithms on the output of the ZED2 camera
7. The `pathfinder_node` node, which runs an A-star algorithm on an abstracted representation of our state-space
8. The `path_follower` node, which interfaces with `pathfinder_node` computing sailable paths from sequences of waypoints, and selecting appropriate target positions for navigation
9. The `heading_controller` node, which receives targets from `path_follower` and generates rudder control signals which drive the boat towards its current target

10. The `state_manager` node, which communicates with neetwork_comms and manages the lifecycle and execution states of the other ROS nodes

Additionally, the path_follower and heading_controller nodes each have two variants, implementing different path following solutions. The first variant uses a “look-ahead” algorithm, where a target point is generated by path_follower some distance ahead of the boat’s current position on the path, and heading_controller points the boat at the target position. The second variant uses a vector field approach, where path_follower generates pairs of points (the current straight-line path segment being tracked), and heading_controller calculates the target heading on a vector field, and turns the boat to align with that heading. An example of vectors computed by such an algorithm for a given line segment is shown in Figure 34.

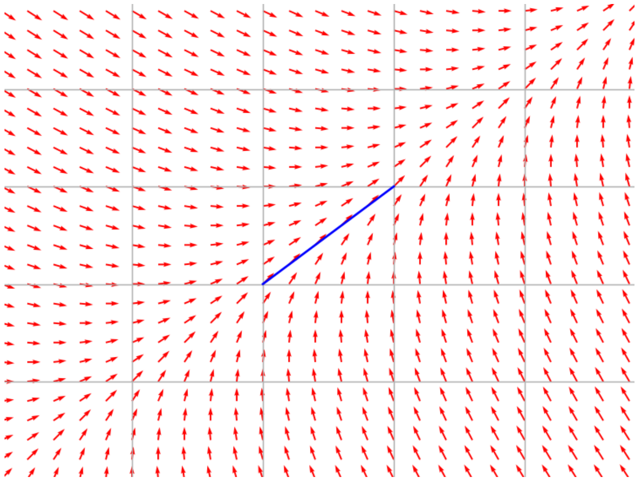


Figure 34. An example of the Vector Field Approach

Each of these approaches has some strengths and weaknesses- a look-ahead algorithm is inherently a target tracking algorithm, and thus has unavoidable inaccuracies inherent to such systems. Sharp corners and other tight turns are not possible to track accurately, as the target point (moving ahead of the boat) causes the boat to “cut” the corner. This could cause issues with buoy rounding at close distances, and even cause the path following to degenerate, leading to the boat becoming stuck facing upwind. Though these issues can be mitigated with larger buoy rounding distances and with modifications to the path generation algorithm, they were concerning enough to prompt the creation of a second path following system using vector fields.

The vector-field path following algorithm, in contrast, does not suffer from corner-cutting like the look-ahead function does. As long as the current path segment is computed properly, the robot will not deviate from that path unless acted upon by external forces. However, without some form of path segment blending, the robot will overshoot each corner of the path, as it cannot turn infinitely quickly to align itself with the vectors generated from the new path segment. Additionally, this system is potentially more vulnerable to being disturbed by small degenerate path segments. If the pathfinding system generates a strange path (for example, a path with a tiny loop) which can occur due to limited map resolution combined with sub-optimal buoy rounding calculations, the look-ahead algorithm will “smooth” such a segment out of the path, whereas the vector field approach may generate large, rapid changes in rudder angle. Regardless of these issues, this path following system is theoretically more optimal than the target tracking solution, and our team will evaluate the efficacy of both solutions before selecting one for further refinement and for use in competition.

4.3.2 Migrate to a more modern software platform

Until now, the project has been implemented on Ubuntu 18.04 LTS “Bionic Beaver” and ROS2 Dashing Diademata. Given that Ubuntu 18.04 reached end-of-life (EOL) in May of 2023, and ROS Dashing reached EOL in May of 2021, a platform upgrade was desired to maintain package compatibility. The project has been ported to Ubuntu 20.04 “Focal Fossa” and ROS2 Humble Hawksbill. These are LTS releases which will reach EOL in April 2025 and May 2027, respectively. Additionally, the boat’s main controller has been replaced with a Jetson Orin Nano, a modern replacement for the original Jetson Nano. Upon release of the StereoLabs ZED SDK for Nvidia Jetpack 6.0, the project was migrated to Ubuntu 22.04, which is ROS2 Humble Hawksbill’s recommended platform.

4.3.3 Improve platform stability and add failure recovery

In previous years, teams encountered issues with software stability, and a lack of failure recovery processes led to frequent manual retrievals of the boat. This was identified as an area where significant improvements could be made without much effort. To achieve this goal, the boat’s ROS nodes have been modified to make use of the concept of “lifecycle” state management, introduced to RCLPY (ROS2’s Python client library) as of ROS2 Humble Hawksbill. A central state management node manages the lifecycle of each node, transitioning them from their initial inactive state through to their active state. This system also allows us to control the lifecycle of individual nodes directly- if we wish to restart a single node, we can simply transition it to its inactive state, and then transition it back through its initialization state, and then to active.

4.3.4 Improve autonomous functionality

Previous teams put significant effort into autonomous functionality, however the potential of the most recent teams approaches were never realized, as much of the functionality was either never finalized or never successfully demonstrated or documented. As such, one of the major tasks of this year’s team is to redesign the boat’s autonomous systems.

To achieve the goal of fully autonomous sailing, a four-tier path planning system has been designed. First, global waypoints are defined, marking a sequence of points the user wishes SailBot to visit. To pathfind between any two points, the following three tiers are used as necessary: first, the SailBot raycasts on an occupancy grid representing the sailable regions of the body of water it is currently in. If the straight-line path is clear, and the path does not require the SailBot to sail upwind (as determined with a ~45° upwind “no-sail” zone), the SailBot plans a linear path between the two points. If the current wind direction prevents the SailBot from following such a linear path, two points C_1 and C_2 are calculated such that pathing from the first waypoint (point A) through C_x to the second waypoint (point C_1) sails the boat along the boundaries of the no-sail zone. Once calculated, the same ray casting process is used to check the validity of these paths. If either is determined to be valid, the boat plans a path through point C_x. The formula for calculating these tack points can be seen below:

$$d_{AB} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$
$$\theta_{AB} = atan2(y_2 - y_2, x_2 - x_1)$$
$$\theta_{NG1} = \theta_w - \pi - \theta_{NG}$$
$$\theta_{NG2} = \theta_w - \pi + \theta_{NG}$$
$$\alpha = \theta AB - (\theta_w - \pi - \theta_{NG})$$
$$\beta = \pi - 2 * \theta_{NG}$$
$$\gamma = \pi - \alpha - \beta$$
$$l_1 = \frac{d_{AB} * sin(\gamma)}{sin(\beta)}$$
$$C1_x = x_1 + l_1 * cos(\theta_{NG1})$$
$$C1_y = y_1 + l_1 * sin(\theta_{NG1})$$
$$l_2 = \sqrt{(x_2 - C1_x)^2 + (y_2 - C1_y)^2}$$
$$C2_x = x_1 + l_2 * cos(\theta_{NG2})$$
$$C2_y = y_1 + l_2 * sin(\theta_{NG2})$$

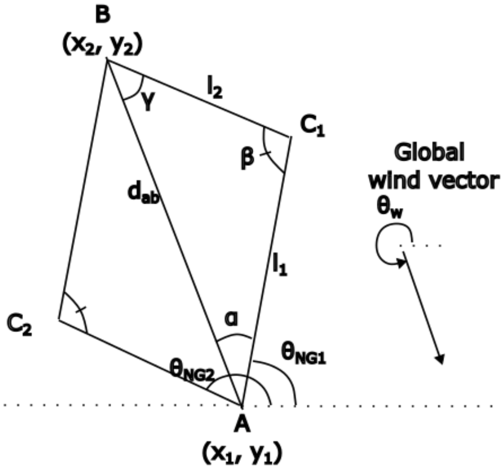


Figure 35. A diagram of the variables and setup for calculating tack points

In this formula, (x_1, y_1) and (x_2, y_2) represent the start and end points of the segment of interest, θ_w is the current wind direction in radians, and θ_{NG} is the upwind no-sail angle in radians.

If either of the first two tiers results in an invalid path due to projected collision with an obstacle, the pathfinding situation is assumed to be complex enough that a “real” pathfinding algorithm should be used, and the boat falls back to the third tier of path planning. Here, the boat rotates (through OpenCV image rotation) its representation of the state of its environment- the occupancy grid- such that the current true wind vector is axis-aligned with the grid. Then, a modified A* algorithm is run on the map. This algorithm uses a 7-neighbor approach (rather than the traditional 8-neighbor), where the missing neighbor represents movement directly upwind. Thus, the only available neighbors at any point during pathfinding will be ones outside of the upwind restriction. Because the boat cannot turn infinitely quickly, we also penalize paths when they turn to encourage the generation of truly sailable paths. Once a suitable path is found, the path is simplified using the same raycasting line-of-sight method used for the previous pathfinding tiers, and the boat accepts the simplified path as its plan. Figure 35 displays an example path calculated by this algorithm. First, the path is generated in the rotated coordinate space. Then, the path is simplified and re-projected back to the original map coordinate system. In these images, the wind direction is represented by the green arrow, and the green and blue dots represent the start and goal position, respectively. Black areas represent simulated obstacles.

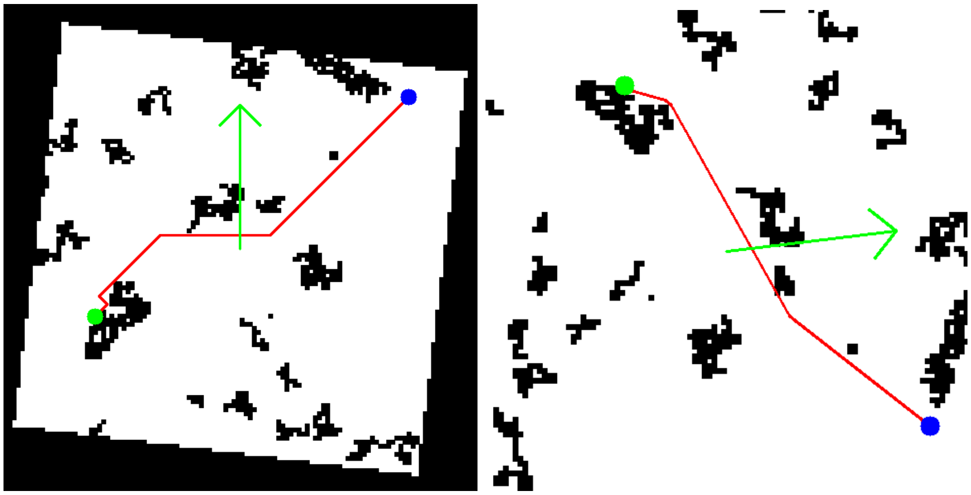


Figure 36. Rotated and reference maps for pathfinding in a wind-constrained environment.

The boat also has an alternate pathfinding algorithm using a probabilistic sampling approach known as PRM*. Instead of creating one node for each map cell, the algorithm instead samples points according to a probability function from the map, and creates a graph from these points by computing nearest neighbors for each point (We use the Computational Geometry Algorithms Library's KD tree for this operation). This allows us to sample points intelligently, sampling a high density of points around areas of interest (obstacles or buoys), while leaving sparse areas at a lower density. This saves both map setup time and search time, as far fewer redundant nodes will be explored. However, it does prevent the use of the map rotation developed for the A* algorithm, since the KD tree used for nearest neighbor operations cannot support such rotations. Instead, we must check the wind restriction angle for each neighbor connection we explore. Given that PRM* is not inherently limited to grid-aligned movement like the A* algorithm is, however, this downside does not degrade the quality of generated paths. Figure 36 shows an example of a node sample map greeted by PRM* is shown, alongside the path it generated.

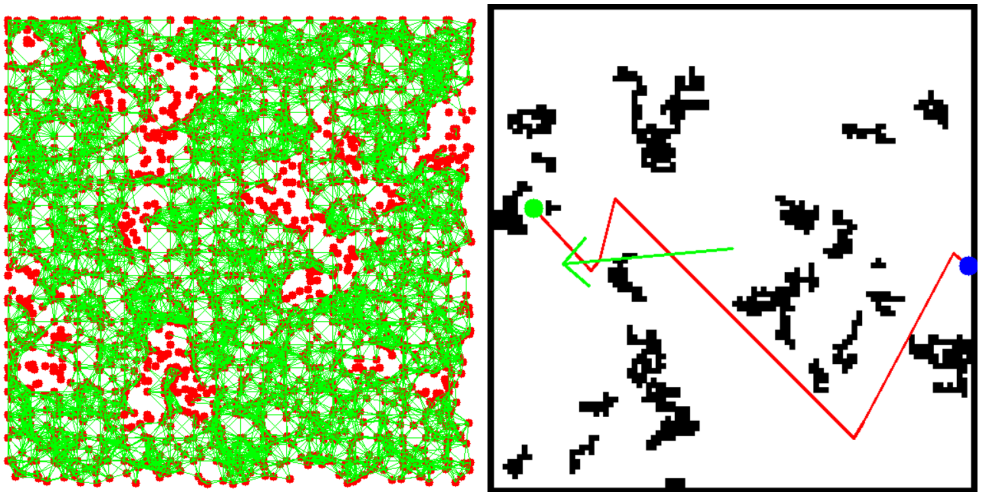


Figure 37. A node sample map generated by PRM* (left) and path generated (right)

Once a path is generated, the boat must generate proper control signals to drive itself along that path. The boat has three control systems- the rudders, the trim tab, and the sliding-rail ballast. The rudders, used to control the heading of the boat, are controlled using a fuzzy controller (implemented using scikit-fuzzy), operating on heading error and angular velocity. The rudder controller may be adjusted in the future to enable the use of more environment variables (for example, to account for control differences at various heel angles). The trim tab is controlled using the state-based controller developed by previous teams. The ballast control system currently uses a simple positional P controller, which is capable of reacting to boat heeling by shifting the ballast. This system averages roll values over long periods of time, and only moves the ballast in response periodically, as we are not interested in reacting to momentary shifts in heel angle, and the start current of our motor is very high. This system may additionally be improved by incorporating a mathematical model of the system, which would inform the controller about the likely impact of a given shift in any of the control surfaces.

4.3.5 Replace radio telemetry with 4G

Previous teams have experienced significant issues regarding radio telemetry reliability, reporting significant error rates and/or total loss of communication at ranges as short as 667 feet. Additionally, they reportedly experienced software stability issues when operating at the outer edges of this range, which significantly impeded their attempts at completing challenges in the SailBot competition in June of 2023. To rectify this issue, the 900MHz radio system has been replaced with a 4G LTE modem. To connect client devices with the boat, the peer-to-peer networking service Zerotier is used to assign a static IP address to the sailboat controller. This 4G modem and routing solution allows us remote access to the boat, unconstrained by line of sight and reliable even in non-ideal weather conditions. This approach has proven successful, and in our testing we were able to connect to the boat as long as a cellular connection is available.

4.3.6 Telemetry & RC system

The telemetry system for the boat operates on a server-client architecture. The SailBot exposes a gRPC interface which clients can utilize to access telemetry data or submit RC commands. This interface exposes functions to set ballast, rudder, and trim tab positions, to change the current autonomous state (full RC, auto-ballast, full auto), to update the list of global waypoints, to restart a given ROS node, and to retrieve telemetry data.

Additionally, we implemented a cross-platform client application using Flutter, capable of running on Windows, Linux, Android, and MacOS. This telemetry client exposes all of the functionality of the SailBot's gRPC interface to the user, contextualizing positional and rotational information on a map of the surrounding area. The client identifies server addresses by retrieving a list of servers (various iterations of the SailBot controller) from a Github repo, allowing for easy movement between testing platforms without rebuilding the client application.

4.3.7 Trim-tab communication rework

Previous teams have encountered significant difficulty in working with the Jetson-to-trim tab communication pipeline, resulting in connection failures and difficulty controlling the boat. After thoroughly investigating this issue, we have determined that there were three primary causes of this difficulty. Firstly, the communication occurred over Bluetooth Low-Energy (BLE), using a bluetooth USB dongle plugged into the Jetson Nano, and an Arduino MKR WIFI 1010 in the sail. This dongle had no external antenna, and the MKR WIFI has a very small antenna, leading to poor signal integrity. Secondly, the BLE client library being used on the Jetson was Bleak, a Python package. Bleak is a package that relies on Python's AsyncIO package for asynchronous communication with bluetooth devices. However, AsyncIO (as an asynchronous programming framework) relies on an event loop, which inherently conflicts with ROS2's event loop in the Python language. Essentially, either the AsyncIO event loop can be waiting for tasks, or the ROS2 RCLPY event loop can be waiting for tasks, but not both simultaneously. This cannot be resolved without splitting Bleak out into an entirely separate process, which would increase program complexity and opportunity for failure. Third, we encountered some issues with BLE communication which seemed to stem from incompatible implementations of the BLE standard in client devices and libraries, which lead to BLE topics being invisible to some devices, even when they were visible to others.

With all of these issues in mind, we decided to make three major changes to our communication implementation. Firstly, with the switch from the Jetson Nano to the Jetson Orin Nano, we obtained a much stronger network interface, which has proper antennas, resolving the signal strength concern. Secondly we switched from BLE to Websockets (over a dedicated AP), which resolved issues both with BLE implementation. Third, we transitioned away from having the Jetson communicate directly with the remote microcontroller, and instead added a second microcontroller (an ESP32) in the hull which the Jetson communicates with synchronously over UART serial, and swapped out the MKR WIFI in the sail for another ESP32. This restructuring allowed us to offload the asynchronous communication system onto the hull-based ESP32, side-stepping the conflicts between ROS2 and other asynchronous frameworks.

With these changes, we have completely resolved the trim tab communication issues, which allowed us to focus on higher-level concerns rather than on debugging communication protocols.

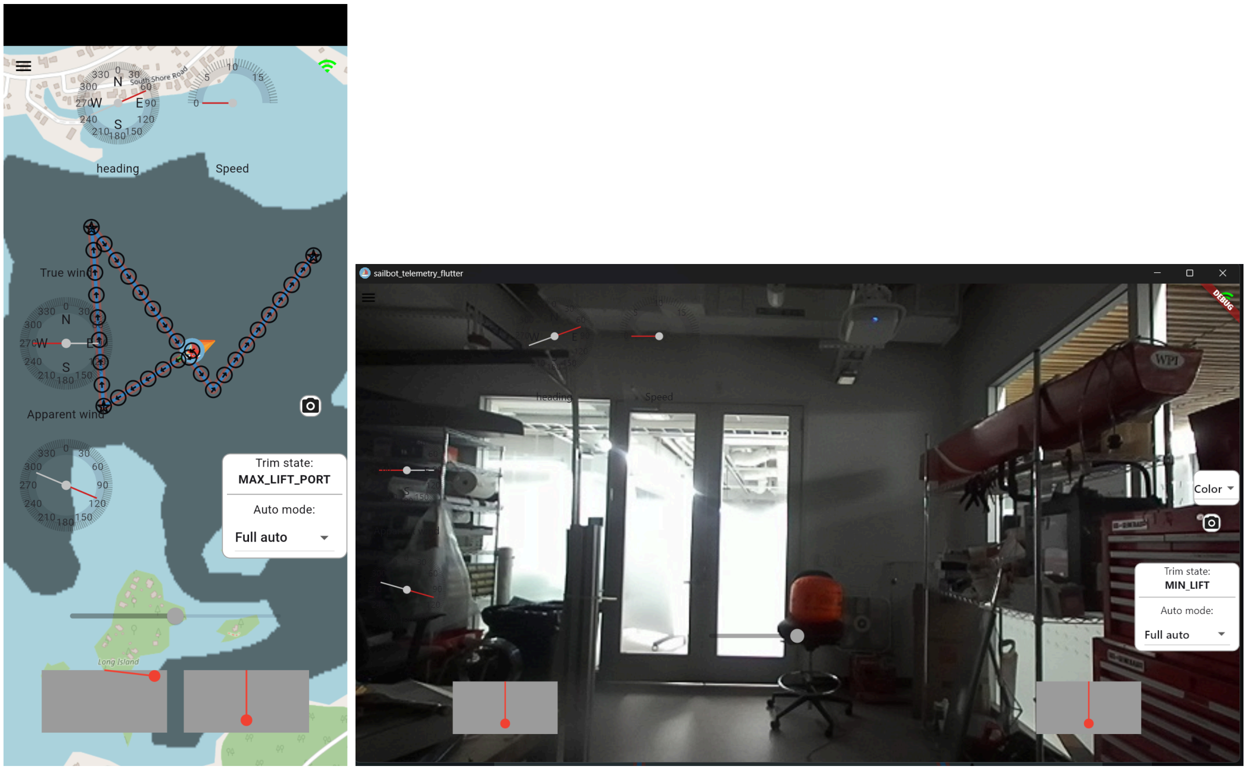
4.3.8 Integrate computer vision into navigation system

Previous teams have attempted to integrate computer vision and obstacle detection into their navigation algorithms, but attempts have thus far been largely unsuccessful. Previous work has focused on the use of machine learning (YOLO) object detection and classification networks, which has presented significant challenges regarding model training and GPU compute requirements. Our team believes that this approach, while valid, is likely overly complex, and that a simple edge-detection method (analogous to ones introduced in RBE 3001) combined with a point-cloud depth structure generated by the onboard ZED2 camera might be used to reach a "good enough" solution to this problem, given the limited scope of object detection problems presented in the SailBot competition. As such, we implemented an edge-based object detector using OpenCV, which detects orange objects in the image from the ZED2 camera that are approximately circular, and use the fact that we know the size of the buoys they represent to calculate their (x, y, z) world-space coordinates. Once such an object is detected, its position in camera-space is determined, and projected into lat, long coordinates based on the boat's current position. If the boat detects that a buoy is within a configurable range of a waypoint it is told to circle (often marked by a buoy on a sailing course), it will assume that the buoy it has detected is the one it is supposed to navigate around, and it will override that waypoint's position with the buoy's, for the purposes of navigation.

4.3.9 Improve debugging and visualization of boat processes

With our intent to focus on autonomous functionality, we foresee a need for remote visualization and debugging tools that allow us to investigate the state of the boat's processes while it is operating, without needing to be physically connected to it. As such, we implemented several visualization features into our Flutter telemetry client. Currently, this client is capable of displaying boat heading, speed, and position (contextualized with a map displayed in the application), as well as node status information, and past and (planned) future positions. It also displays rudder and trim tab status during autonomous operation, any paths the boat has planned, and information relating to control along said path (target point, current segment, etc), and displays markers at the lat, long coordinates of any buoys that are detected. Additionally, it provides the ability to display a camera feed from the ZED camera, as well as a video feed of those frames analyzed by our buoy detection algorithm with detected buoys circled in green, to allow us to both control our boat without having line-of-sight to it, and to visually debug computer vision problems without needing a display connected to the controller.

Figure 37 shows screenshots from the Flutter telemetry client, showing the primary control interface and the camera passthrough.



Trim state:
MAX_LIFT_PORT
Auto mode:
Full auto

Trim state:
MIN_LIFT
Auto mode:
Full auto

Figure 38. Flutter telemetry client showing control interface and camera feed from the ZED camera

4.3.10 Improve efficiency of boat software

Battery life is an integral part of the functionality of the boat. Improving battery life is critical to enabling longer continuous sailing sessions and less recharging downtime. To ensure our boat would meet the requirements of the SailBot competition, we monitored the boat's power draw (and that of the separate trim tab controller), and found that the power draw for each (with all processes running) would allow for over 12 hours of battery life, well above the 8-hour goal.

4.3.11 Create documentation

Every year, each new team spends a significant amount of time figuring out what the current state of the boat is, and how to use the software. This could be reduced significantly by maintaining basic documentation about boat functions and usage. Additionally, maintaining internal documentation will allow us to understand the current state of the robot without relying on the memory of the person who worked on any particular portion. Initial documentation includes information such as: how to set up and run the boat, the software structure, the purpose of each of the ROS2 nodes, and python documentation for the ROS2 project. Documentation also includes physical elements, such as object files for 3D-printed parts, design files for the hull and deck, schematic and layout files for the electrical system, and notes about the expected longevity of various parts and any need for maintenance.

5 | Testing and Validation

Testing took place in March and April at Lake Quinsigamond and Webster Lake. Many tests were performed to test the functionality of the boat and then improve upon before carrying out the large functionality goal tests.

5.1 Design Objectives

5.1.1 Create a New Hull for the Boat

The major mechanical goal for this project was to design and manufacture a new hull that better suits the function of the SailBot. This goal was very successful in which we designed, manufactured, and tested the new hull. In the water, the new hull floats around the predicted waterline. The bow is generally able to ride above the waves as intended, but due to the low freeboard, water often splashes over the top. This is not an issue though since the boat does not slow down a lot and the deck is well waterproofed, however, this can generally be resolved by adding more weight to the stern to increase the angle from the deck to the water. The boat was designed to have a slight angle between the deck in the water for this purpose.

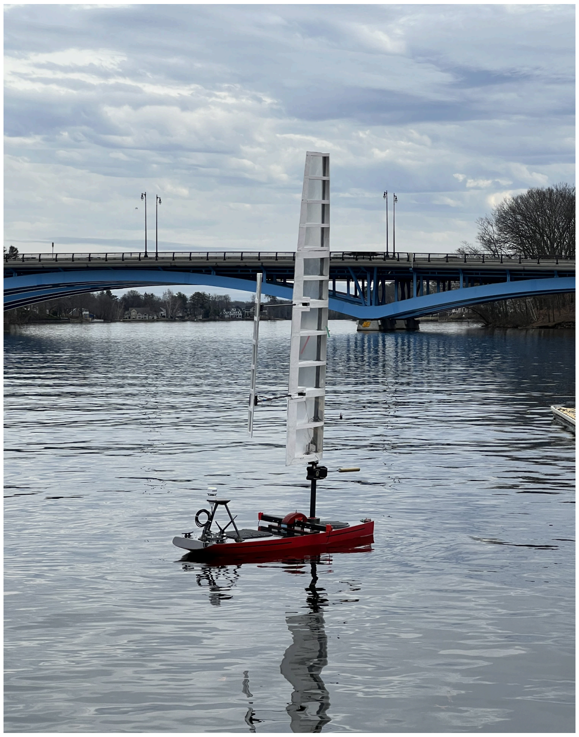


Figure 39. *Wild Goats* sailing in Lake Quinsigamond

5.1.2 Increase Interior Accessibility

Accessibility was an important goal throughout the design process. This goal was achieved by using the Hobie Round Twist N Seal Hatch 8 inch Kit and Hobie Kayak Rectangular 11.25 x 15 inch Hatch Kit. With the placement of these hatches, we are able to reach all parts of the interior unless physically blocked off such as the tip of the bow, very end of the stern, and keel box which is accessible from the bottom. This made maintenance and any changes very easy.

5.1.3 Improve Airmar Mount

The main goal of improving the Airmar mount was to stop the Airmar from moving. This design was successful because the 3D print was created to restrict rotation of the Airmar sitting in the mount. This will ensure that the Airmar will always read the correct position and cannot be put in any other direction.

5.1.4 Transition Boat to PCB Mounted Electronics

The boat's electronics were fully migrated to PCBs. This has significantly increased the stability and reliability of the system overall, as it is less prone to shorts, melting wires, and systems being disconnected from physical impact.

5.1.5 Add Battery Management System

A battery management system was integrated into the boat's main PCB, which allows the boat to read the battery's voltage, and also implements a voltage cut off to prevent battery damage.

5.1.6 Migrate to a more modern software platform

The boat's software was migrated to Ubuntu 20.04 and ROS humble. It was not able to be migrated to Ubuntu 22.04, however, so ROS must be built from source. When Nvidia releases JetPAck 6.0, and StereoLabs releases their ZED SDK for this new version, this migration will be possible.

5.1.7 Improve platform stability and add failure recovery

Platform stability and failure recovery is drastically improved compared to last year. The trim tab connection is extremely reliable, and in our testing has never dropped out or failed to connect. This is a significant improvement over the previous system, which was extremely unreliable, with constant drop-outs and connection failures.

5.1.8 Validate existing autonomous modes and make improvements

The existing autonomous modes were fully replaced with our new solution. The trim tab control code was overhauled to implement more correct movement and to respond to wind in a more controlled manner. Several true pathfinding algorithms were implemented, replacing the blind approach previously implemented. Additionally, several new control modes were implemented, allowing for more fine-tuned control over the actions of the boat.

5.1.9 Replace radio telemetry with 4G

The radio system was fully removed, and replaced with a reliable 4G connection on the Mint Mobile network.

5.1.10 Integrate computer vision into navigation system

The boat is now capable of detecting orange buoys from within 7 meters, and when given a waypoint that is close enough, it is able to snap the position of that waypoint to the position of the detected buoy.

5.1.11 Improve debugging and visualization of boat processes

The telemetry client we developed has been extremely helpful for debugging, visualization, and control of the boat. As shown in the previous design section covering the topic, it displays helpful information such as sail trim state, boat position, heading, and speed, true and apparent wind, ROS2 node statuses, waypoints, paths, buoys, and current navigable area.

5.1.12 Improve efficiency of boat software

Certain aspects of the boat’s code were restructured to avoid unnecessary movement and calculation. The ballast runs on a 10 second timer to avoid rapid adjustments. The trim tab and main controller also both average with values over several seconds to avoid attempting to react to small gusts.

5.1.13 Create documentation

Documentation was significantly improved compared to last year, though there is still significant room for improvement due to the rapid pace of development. Setup guides were created, both for boat assembly and software installation. Important functions were commented and explained in the code, and the github repositories now have “readme” documents explaining their purpose. Electrical schematics were created documenting the layout of the new PCBs, and the design files will be handed over to next year’s team to ensure they can modify them to suit their needs.

5.2 Functionality Goals

5.2.1 Point to Point Navigation

The first goal was to autonomously navigate point-to-point in a wind of between 5 and 15 knots, between points marked either by GPS coordinates or by buoys, to within a radius of 10 meters of the target position. Point to point navigation was tested by choosing a point on the UI to sail to and observing the movements of the boat. During this test the boat was able to create a path and approach the location chosen. Throughout this process, it recalculated its path every 10 seconds and updated it based on the boat's current location. When it reached the location, it sailed into the wind to hold the position.

Throughout testing this process, the boat seemed to have difficulty sailing on the direct path it created. This could be from the wind knocking the boat off course or current in the water. In order to account for this and make the boat more efficient, we decided to use a fuzzy controller to correct for the changing path so the boat stays on course. This allows the boat to fine tune its steering as it moves. Even with the fuzzy controller, it is still important to allow the boat to recalculate its position in case any anomalies such as a change in wind direction occurs.

5.2.2 Endurance Test

The next goal was to navigate a course defined by GPS locations for a period of 8 hours, allowing for intermittent RC control to avoid collisions, rounding each point within a radius of 10 meters. This test was meant to see how long the boat could last on the water going between different points. The factors analyzed over that time were battery life, monitoring mechanical systems for wear, and waterproofness. During this test, the boat was able to respond to the different way points, with intermittent returns to the dock to check the boat's systems.

The boat's battery life was only able to last 6 hours and 20 minutes in our latest test. This occurred primarily due to overuse of the movable ballast. The ballast motor is capable of drawing up to 10 amps, and without careful tuning, it can drain the boat's battery without improving its sailing characteristics. With a more careful application of this system, and/or a bigger battery, the system should be capable of reaching the 8 hour time target.

5.2.3 Buoy Detection

In order to ensure buoy detection, we wanted to ensure that buoys were able to be detected within 5 meters of the camera. To test the range of the camera, we placed an orange buoy in front of the camera. From then, we moved the buoy further away from the boat until the camera could no longer track the buoy.

From this test, it was determined that the buoy could be detected within 7 meters of the camera with minimal tuning. This provides a greater range that buoys can be detected off the bow. This is particularly helpful when the boat is not sailing directly towards the buoy because even at an angle it will still be able to detect the boat. Below, example images of the boat detecting the test buoy approximately seven meters in front of it are shown. These images were gathered through screenshots of our telemetry client, which does not have an image export feature.

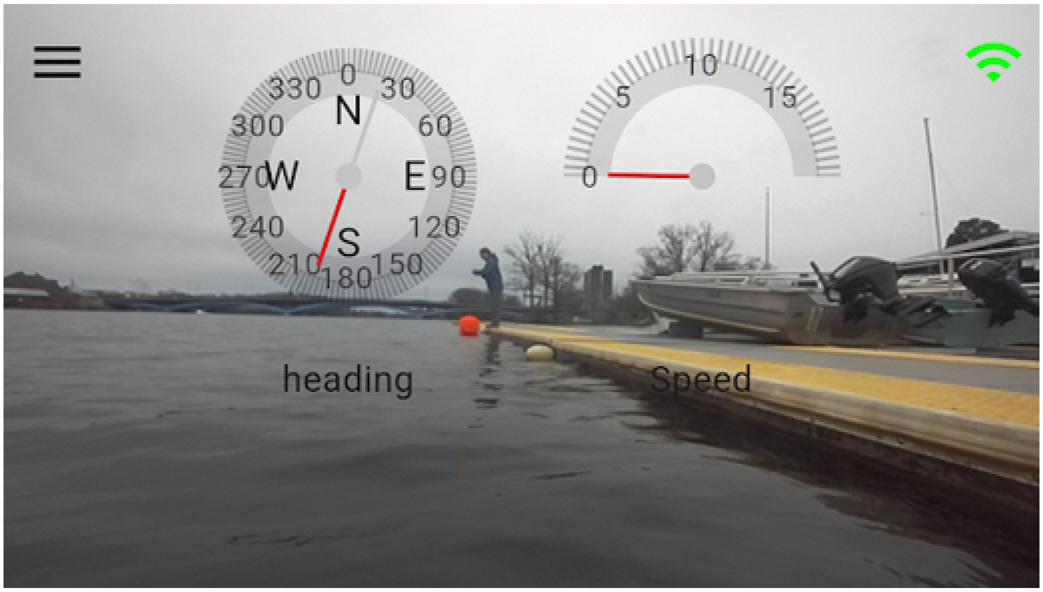


Figure 40. View through the camera during the buoy detection test

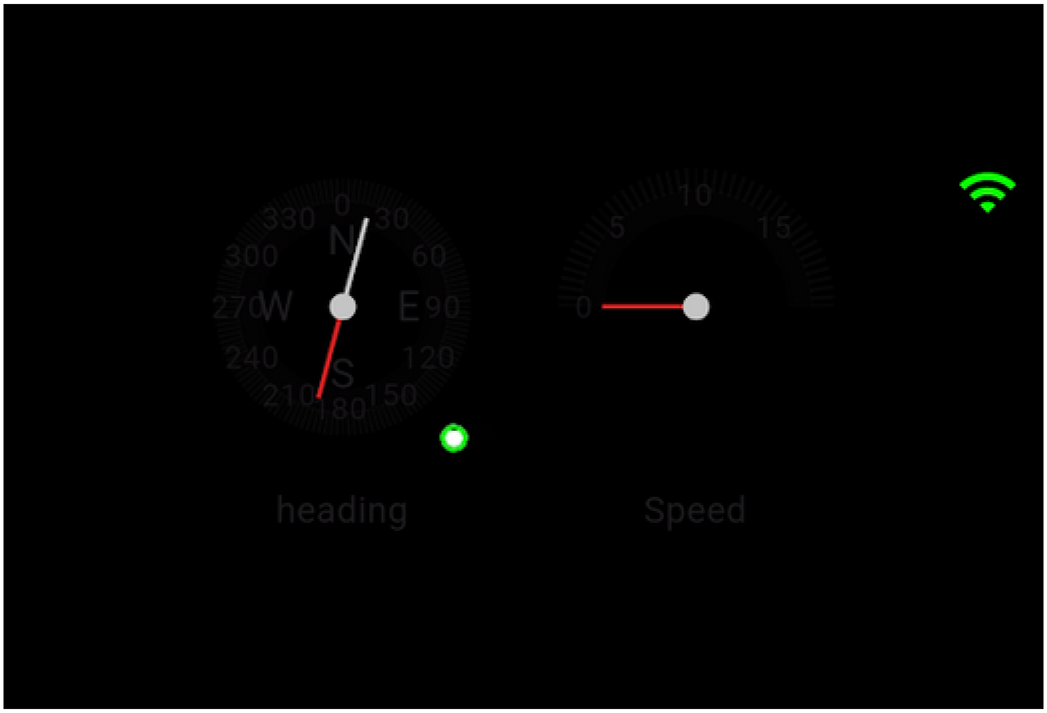


Figure 41. The buoy (in white circled in green) detected by the ZED Camera

5.2.4 Holding Position

The boat's GPS is accurate to within a handful of meters. Given the large tolerance on this test, simply setting a waypoint and having the boat target it endlessly is enough to fulfill this requirement. A more elegant solution would avoid over-actuation and simply drift until it was outside of a certain tolerance, but the brute-force solution described above will be functional enough.

5.2.5 Mile Sail

The final goal was to autonomously sail to a given position at least 1 nautical mile away from its starting location, and return to within 10 meters of its starting position. While we could complete this goal with automatic trim tab control, we were not able to test the autonomous navigation sufficiently to do it fully autonomously. Further development of this system will be completed before the SailBot competition.

6 | Future Work and Recommendations

6.1 Movable Ballast

The current movable ballast has a GT2 timing belt to move the ballast weight. This belt has a tendency to stretch with the given weight. This causes the belt to get loose and will unexpectedly shift to one end of the ballast offsetting the boat, getting the ballast stuck, and draining the battery as it tries to correct itself. In order to improve this, the movable ballast could be further improved in a few different ways.

One way could include replacing the belt with a material that is rated to move the weight of the ballast and will not stretch. Another option could include replacing the belt with a chain system, however, that would require extensive modifications of the system.

Another way to alleviate this issue is to do further analysis on the weight required to move the ballast to the ideal heel. The ballast created in previous years for this design is not the same and heavier than the one that was tested. With a new hull and the change in weight, further analysis could make this system more efficient and reduce the total weight of the boat.

6.2 Keel Mounting

Currently the keel is held on by one bolt, however due to previous issues, the bolt cannot go in all the way. This is concerning since there are no other ways the keel is attached to the boat.

Ongoing improvements this year include machining a new piece to be the mounting location. In order to fit this, the top of the keel mount will need to be removed so the new piece can sit in it properly. This would allow for a longer bolt to fit inside.

It is recommended in the future to completely replace the top mounting with more than one bolt so that the keel is not held in at one point. It may also be worthwhile to look into designing a new keel designed to balance this specific hull since this one was designed for the previous hull.

6.3 Wingsail

The electrical system of the trim tab is located in the sail. This can sometimes be difficult to access particularly on the water because it can only be reached by reaching up a hole in the foam of the bottom rung of the sail. One of the areas we improved with this is 3D printing a piece to fit around the sail and the trim tab hole to remove the load from the foam that was broken from overuse. Further improvements could include creating an access panel to make it easier to reach the trim tab box more regularly.

6.4 Electrical Systems

The robustness of SailBots electrical systems was dramatically improved, there are still areas for improvement. Currently SailBot uses two different kinds of connectors, XT90s and automotive wire connectors. The XT90s are not waterproof, and thus create the possibility of electrical faults if any stray water was able to get inside the hull. Therefore switching fully over to waterproof connectors is recommended.

Next, the Talon SRX could be switched over to CAN control. Although CAN is slightly more difficult to interface with, CAN control on the Talon SRX allows for its more advanced features to be used, such as current limiting and monitoring. This would protect the boat from stalling its motor out if the ballast malfunctions and improve the efficiency of the ballast motor.

Originally SailBot had plans for a LED stack light to be used to visualize information about the boat from shore. A provision was made for this on the SailBot PCB, however the light itself was not installed. Installing the stack light would allow for easy monitoring of the boat's status from a distance.

SailBot also originally had an E-Paper display that could show information about the boat. This E-Paper display was removed last year due to issues with the SPI communication. This was likely caused by either ringing in the SPI lines or parasitic capacitance between the data lines corrupting the signal when it was traveling over a long distance. SPI is not designed to communicate over distances longer than 10-20 cm. To reimplement the E-Paper display, the team recommends either adding filtering resistors and capacitors to reduce the ringing, getting an E-Paper display with a more appropriate communication protocol or installing a co-processor by the E-Paper Display to reduce the range SPI needs to be utilized over.

6.5 Software Systems

The software of the boat is an area which invites endless improvement. Many areas of the boat’s code can be improved or expanded upon. More advanced path planners could further optimize paths between waypoints, and dynamic map selection and transitions could greatly extend the boats effective range.

To significantly improve the boat’s control systems, a hydrodynamic model of the boat is required. This would allow simulation of the boat’s behavior (ideally pre-computed and stored in a lookup table) to inform the boat’s actions. Particularly, this lookup table could be used to correct for leeway angle caused by lateral forces on the sail, as well as for control changes caused by heel angle.

Buoy detection could also be improved- although seven meters may be sufficient for the tasks we intend to use it for, a greater range would be ideal. Additionally, the use of HSV image segmentation instead of a neural object detector such as YOLOv5 severely limits the ability of the system to deal with lighting changes. Ideally, the system would use a custom object detector trained on images of many buoys. This would require several thousand images of training data to become effective, however.

The error reporting and debugging of the system could be significantly improved, as well. Passing errors through the gRPC server to the telemetry client would aid in diagnosing errors without needing to launch the nodes in a visible terminal, which was a frequent point of friction we encountered during testing.

6.6 Transfer of knowledge

Since this project will be continued next year, it is important to pass on the all materials we have created. The CAD models of the boat will be shared through OneDrive and all written files will be shared through Google Drive. The complete electrical schematic for the boat is available in the appendix, and the layout files are also available in the drive. All code and documentation is publicly available through the WPI Sailbot Github. Additionally, all members for next year's team have been added to the WPI SailBot Discord to ensure they are able to contact the previous year’s teams to ask questions, transfer files, and get feedback.

7 | Broader Impacts

7.1 Engineering Ethics

Throughout this project, safety has been the highest priority. This includes safety in the systems designed as well as ensuring the testing environment was safe. Some things users need to be aware of are the boat's weight during assembly and disassembly and possible pinch points around all of the mechanisms.

Autonomous sailing allows for the boat to go places people cannot, ensuring the safety of anyone using the boat. Any risk to the user would be from assembly and disassembly of the boat, but otherwise the boat can be controlled remotely from any location ensuring human safety while in use.

7.2 Societal and Global Impact

Similar technology is being used today to develop surface autonomous vehicles for weather monitoring, research, and surveillance. Systems like ours can be applied to numerous applications by outfitting the boat with different sensors pertaining to the specified goal. Another application could be international shipping where wind powered vessels are being researched. Algorithms such as those we created can be used to control these types of systems and be optimized to provide more efficient shipping methods across the world.

7.3 Environmental Impact

One of the benefits of systems like this, is that as a sailboat, it is powered by the wind. While it does make the speed more variable due to the weather, it does not require nonrenewable resources as a source of propulsion and only electricity for sensors and moving the sail. This technology could be applied to larger vessels such as cargo ships which require fuel as a source of propulsion and therefore decrease the carbon footprint of international shipping. Future iterations of our technology could even include using solar panels to power the electronics system of the boat, making it fully independent which would allow for greater hours of operation.

One environmental concern would be the lead used as ballast. However, this lead is sealed with epoxy so it cannot make contact with outer environments.

7.4 Codes and Standards

The Convention on the International Regulations for Preventing Collisions at Sea, 1972 (COLREGS) determine the “Rules of the Road” on the water. This determines which types of vessels have right of way. For example, boats that are under power must stay clear of boats under sail, fishing, with low maneuverability, or not under command. Currently, there are no standards for autonomous boats and where they fall within these requirements.

Autonomous vehicles could be considered vessels that are not under command or have low maneuverability. Not under command is typically a status reserved for situations where the boat cannot be controlled. For example, something has broken and the ship is no longer maneuverable. According to COLREGS, vessels that are not under command or have low maneuverability must display 2 vertical red lights and 2 circular shapes. However, it is unclear if these regulations would pertain to our vessel.

With ever evolving technology, new regulations should be created to more clearly define these regulations. However, overall, it is the responsibility of all to avoid collisions on the water. For this, the SailBot does have a manual mode so the user can navigate it around any other obstacles on the water.

7.5 Economic Factors

This system is being produced for a competition about autonomous sailing and therefore will not be produced on a larger scale. There are companies that are working on producing various sailboats, but economic factors depend largely on the size and purpose of the vessel. Our specific design is quite costly due to the sensors required to provide all of the needed data. This includes the AIRMAR and Jetson Nano. This technology instead can be used as a proof of concept that can be used in many different industries to further develop autonomous systems as a whole.

8 | Conclusion

This year’s SailBot team was largely successful in our goal of creating an autonomous sailboat. Although we fell short of some of our functional goals due to time restrictions and the increasing complexity of the system, our most important functional goals were met, and our design and construction goals were fully achieved. Our team will continue development of the system until the SailBot competition in June of 2024.

9 | References

COLREG: Convention on the international regulations for preventing collisions at sea, 1972. (2003). . International Maritime Organization.

Del Vecchio, A., Sykes, M., Tesoriero, A., Kumar, D., Song, J., Thomas, J., Gruner-Mitchell, R., & Nurse, T. (2022). *SailGoat: Autonomous Sailing System.* : Worcester Polytechnic Institute.

Eusman, N., Zebrowski, L., Jackson, N., Scholler, C., Laks, M., Thammana, A., & Burri, C. (2021). *SailBot: Autonomous Sailing Robot.* : Worcester Polytechnic Institute.

Johnson, H. (2018). *SailBot 2017-2018.* Worcester: Worcester Polytechnic Institute.

Pfannenstiehl, O., Alden, S., Unger, J., Moore, D., & Pelletier, C. (2023). *SailBot 2022-23.* : Worcester Polytechnic Institute.

Randall, K., & Palmer, S. (2019). *WPI SailBot 2018-2019.* : Worcester Polytechnic Institute.

SailBot | International Robotic Sailing Regatta. (n.d.). Retrieved October 12, 2023, from https://www.SailBot.org/

Schifilliti, D., Regan, K., & Singer, D. (2017). *The Robotic Automated Wingsail.* : Worcester Polytechnic Institute.

Shanahan, L., Johnson, H., Norris, T., & Burklund, J. (2018). *SailBot 2017-2018.* Worcester: Worcester Polytechnic Institute.

Stafford, K. (2021). (tech.). ’21-’22 SailBot trimtab operation with sketch.

10 | Appendices

6.1 Appendix A: Sailing Basics (Direct excerpt from SailGoat: Autonomous Sailing System 2021-2022, Del Vecchio *et al.*, 2022)

2 Background

In this section we introduce the topics critical to understanding the SailBot project. SailBot is a robotic sailboat, and thus it is important to have a basic understanding of both sailing a traditional sailboat and a robotic one. Additionally, this section introduces the International Robotic Sailing Regatta (IRSR) rules and challenges. Finally, the states of the mechanical, electrical, and software systems of the vessel as left by the 2020-2021 SailBot team are discussed.

2.1 Sailing Basics

A general understanding of the terminologies and concepts of traditional sailing are crucial to the development and optimization of a robotic sailboat. The four areas of importance are: general terminology of sailboats, points of sail, tacking and jibing, and rights of way.

2.1.1 General Terminology

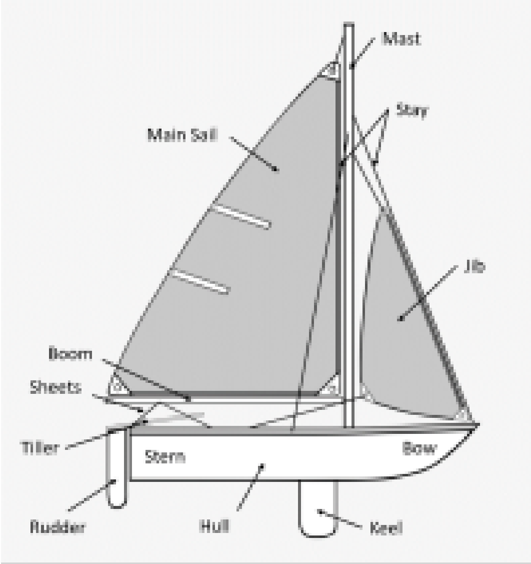
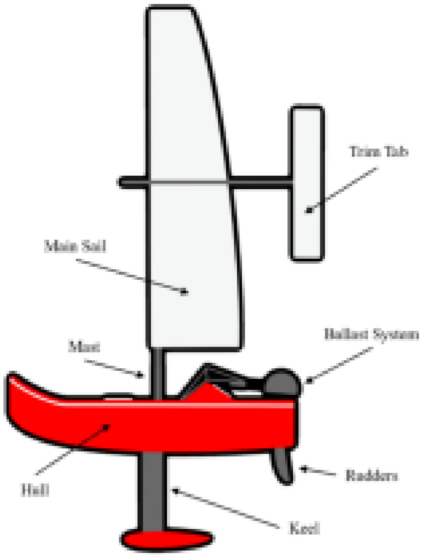


Figure 1: Anatomy of a typical sailboat

The terminology and function of each part/system in the boat are valuable to understanding how to develop and control a boat. A typical sloop rig sailboat is shown in Figure 1 with all the major components and systems labeled. Figure 2 illustrates the pre-existing components and systems on SailBot as left by the previous team. The primary structural body of a sailboat is the hull, to which all other systems are attached. Attached to the bottom of the hull is the keel, a fixed underwater wing used to prevent sideways drift and provide stability. Note the difference in keel shape from the traditional sailboat (Figure 1) to SailBot (Figure 2); SailBot's keel has a bulb at the end. The rudder is the sailboat's moveable underwater steering mechanism. In a traditional sailboat, the tiller, a wooden or metal beam, is used to turn the rudder, as seen in Figure 1. Typically, sailboats only have one rudder, however, SailBot has two rudders controlled by a single servo motor. SailBot's rudders are designed such that one will be vertical when the boat is heeled to the desired angle of 20 degrees.



2.1 Sailing Basics 3

Figure 2: Anatomy of SailBot 20-21

The next system is the sail assembly. The mast is a vertical pole used to support the sails. In traditional boats, the mast is held upright and supported using wires called shrouds (or stays). Attached to the mast is the sail, which is extended using the boom, a horizontal pole that attaches to the mast. The position of the sail is controlled by the mainsheet, see Figure 1. SailBot utilizes a rigid sail assembly instead of a traditional cloth sail. The rigid sail assembly consists of three main components: the mast, the mainsail, and a trim tab (Figure 2). Unlike the mast of a traditional sailboat, the mast for a rigid sail is freely rotating and does not have stays or additional supports. Additionally, the position of SailBot's rigid sail is controlled using a trim tab instead of a mainsheet.

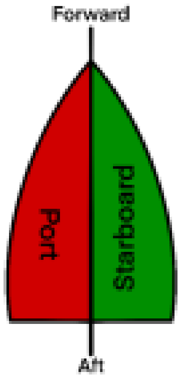


Figure 3: Diagram of port and starboard sides of a boat

The comprehension of the proper terminology of location and direction in a boat is critical in addition to understanding each part/system and its function within a sailboat, see Figure 3. The bow

of the boat is the front of the vessel, and the stern is the back. To explain relative positions in a boat, the terms forward and aft are used, instead of in front of or behind, respectively. Forward refers to anything near the bow within the reference frame of the boat, and aft refers to anything near the stern. While in open water, using the terms right and left can be arbitrary, and therefore are not detailed descriptors of position. Instead, the terms starboard and port are used to describe the position. Starboard refers to the right-hand side of the boat (when facing the bow), and port refers to the left-hand side.

2.1.2 Points of Sail

To successfully operate a sailboat, the position of the boat relative to the direction of the wind and the corresponding optimal sail trim must be understood. These relations are summarized by the six points of sail. The first point of sail is the no-go zone also referred to as being in irons or the no-sail zone (Figure 4). The no-go zone is too close to the wind to sail effectively because the sail cannot generate enough lift in the desired direction. To sail to a point that is directly into the wind, a method of zigzagging across the no-go zone must be utilized. Sailing towards the direction in which the wind is blowing is referred to as upwind sailing. Upwind sailing includes two points of sail: Close Hauled and Close Reach (Figure 4). While sailing on these two points of sail the sails are trimmed in close to generate maximum lift. The next point of sail is a Beam Reach, this refers to the position when sailing across the wind (Figure 4). In this position the sails are trimmed “half in half-out". The final two points of sail are used when sailing away from the wind, referred to as downwind sailing. Downwind sailing is in the direction in which the wind is blowing. The two points of sail included in downwind sailing are Broad Reach and Running (Figure 4). The sails are trimmed most or all the way out when on these points of sail.

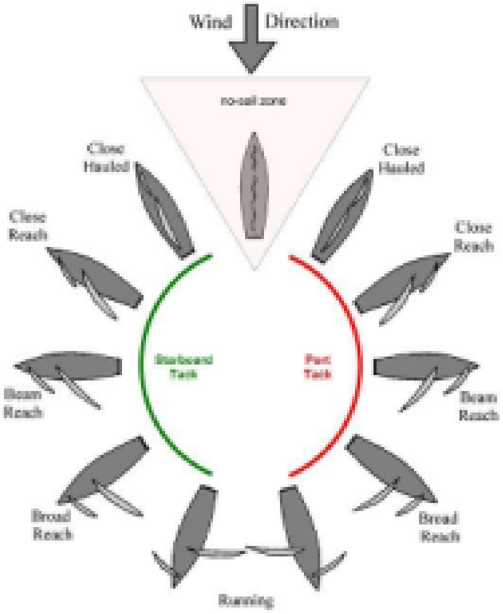


Figure 4: Points of sail
2.1 Sailing Basics 5

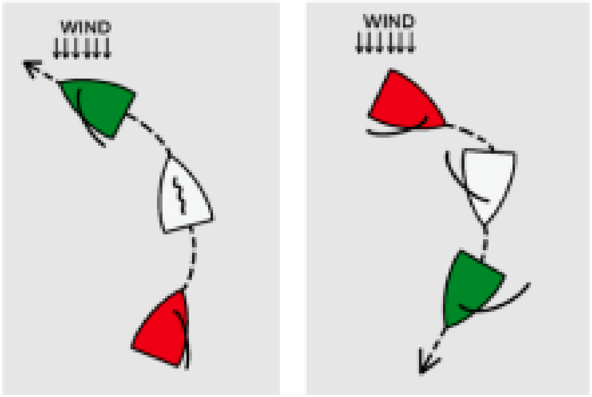
The points of sail are symmetrical across the wind, regardless of what side of the boat the wind is coming across. The side of the boat over which the wind is coming from is referred to as a tack. A port tack is when the wind is coming over the port (left) side of the boat, and a starboard tack is when the wind is coming over the starboard (right) side of the boat. The port tack is shown with a red semi-circle in Figure 4 and the starboard tack is shown with a green semi-circle in Figure 4.

The final concept to understand is windward versus leeward. Windward means upwind, or the direction from which the wind is blowing. A windward vessel refers to the vessel that is upwind of the other vessel, the leeward vessel. Leeward refers to downwind, or the direction opposite to the way that the wind is blowing from.

2.1.3 Tacking and Jibing

There are two basic turning maneuvers when operating sailboats: tacking and jibing. Understanding the difference between the two techniques requires an understanding of the wind direction and the points of sail stated above in section 2.1.2. Both tacking and jibing are used to change the tack of the boat, the side of the boat that the wind is coming over, which consequently switches the side of the boat the sail is on. Tacking refers to the maneuver in which the bow of the boat turns through the wind. Tacking is mainly used when sailing upwind while close-hauled or on a close reach. Jibing refers to the maneuver in which the stern of the boat passes through the wind. A jibe is used to maneuver the boat when sailing downwind, either on a broad reach or while on a run.

A tack is demonstrated in Figure 5a, in which the boat changes from a port tack (red) to a starboard tack (green) while sailing upwind. The boat is initially sailing on a port tack close-hauled (red), then the bow passes through the wind (white), and the boat continues sailing on a close hauled starboard tack (green). A jibe is demonstrated in Figure 5b, in which the boat changes from a port tack (red) to a starboard tack downwind (green). The boat is initially sailing on a port tack on a broad reach (red), then the stern of the boat passes through the wind (white), and the boat continues sailing on a broad reach starboard tack (green).



a) Tacking

b) Jibing

Figure 5: Tacking vs jibing
2.1 Sailing Basics 6

2.1.4 Rights of Way

While operating any vessel it is imperative to understand the boating right of way rules, comparable to "rules of the road" on land. Right-of-way rules are specifically designed maneuvering regulations for the purpose of avoiding collisions between vessels. The five main right of way rules are as follows (*Understanding Boating "Right of Way" Rules*, 2021):

1. Vessels under sail have the right of way over powerboats (with some exceptions).

2. When crossing, the boat on the right (approaching from starboard) has the right of way.

3. When meeting head-on, each vessel must alter course to starboard.

4. Any vessel overtaking another must keep clear of the stand-on vessel.

5. When approaching another vessel whose intentions aren't clear, take evasive actions early.

The five rules stated above must be understood by all vessels. In addition to the general boating right of way rules, there are specific right of way rules that sailboats must also adhere to. During the IRSR vessels will be exposed to situations in which two sailboats meet, in which the following rules will apply (*Understanding Boating "Right of Way" Rules*, 2021):

1. The boat on the starboard tack (wind coming over the starboard side) has the right of way.
2. When two vessels are on the same tack (wind is coming from the same side), the leeward (downwind) boat has the right of way over the windward (upwind) boat.
3. When on the same tack in a passing situation, the vessel being overtaken has the right of way always.

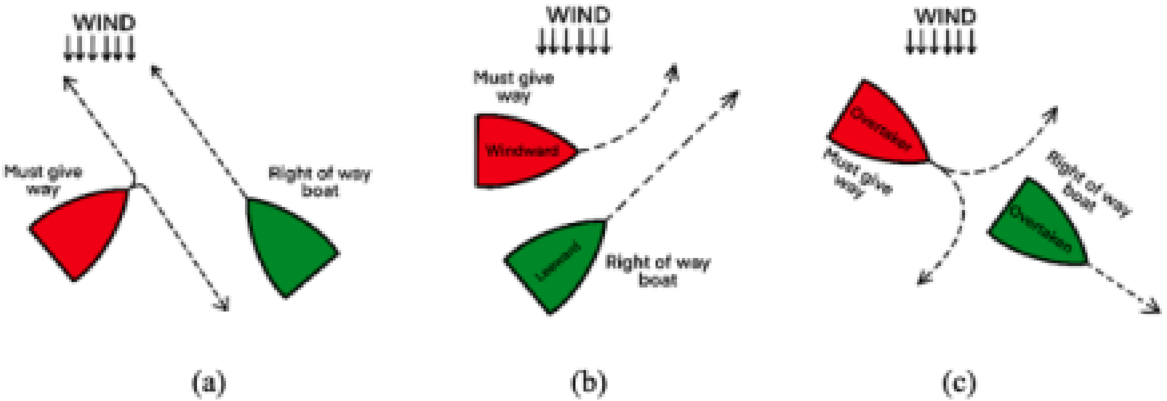


Figure 6: Examples of right of way scenarios

Figure 6a demonstrates sailing rights of way rule 1, in which the starboard tack boat (green) has right of way over the port tack boat (red) which must give way by heading off or tacking to avoid.

2.2 Sailing a Robotic Sailboat with a Rigid Wingsail 7

Figure 6b demonstrates sailing rights of way rule 2, in which two boats are on the same tack and the leeward boat (green) has the right of way over the windward boat (red). Therefore, the windward boat must head up or tack to avoid the leeward boat. Figure 6c demonstrates sailing rights of way rule 3, in which two boats are on the same tack and in a passing situation. The overtaking boat (red) must give way to the boat that is being overtaken (green) by heading up or down.

The comprehension of the eight rules stated above is crucial to maintain safe boating conditions for all vessels, especially during the IRSR.

2.2 Sailing a Robotic Sailboat with a Rigid Wingsail

This section presents the physics of sailing with a rigid wingsail. For most points of sail, our robotic sailboat is propelled through the water predominantly due to the lift generated by the rigid wingsail. The total aerodynamic force on a wingsail has two components – lift, which acts perpendicular to the wind direction, and drag, which acts in the same direction as the wind (Figure 7). Drag tends to push the sail downwind, which is sideways on most points of sail. However, when sailing on a run, drag is the predominant force and it pushes the sailboat forward since the direction of travel is directly downwind. It should be noted that SailBot's mast is designed to freely rotate, such that when the boat travels at an angle with the apparent wind, the wingsail will tend to rotate until the angle of attack (AoA), the angle between the apparent wind and the chord line, reaches zero degrees.

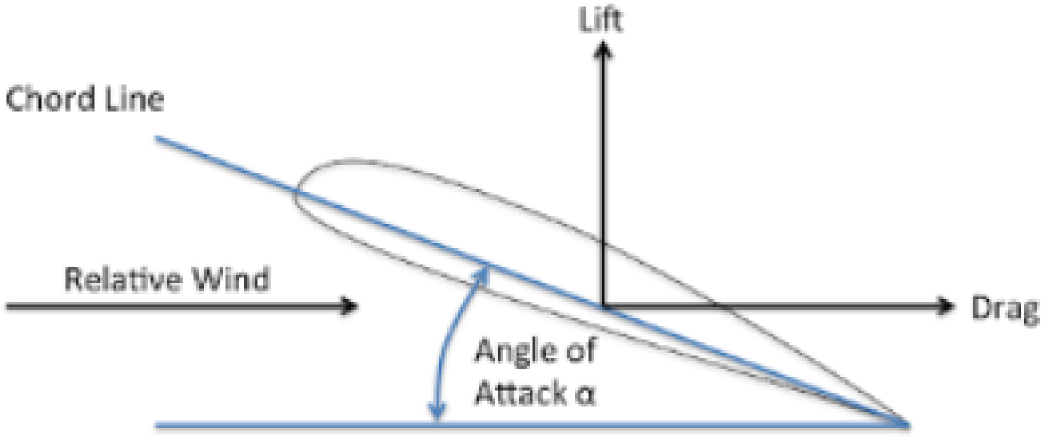


Figure 7: Diagram of angle of attack, lift vector, and drag vector on an airfoil

SailBot's wingsail is fitted with a trim tab, a smaller secondary symmetrical foil that is mounted midway up the mast at the end of two long booms that extends beyond the sail. This airfoil, like the airfoils in the wingsail, is also subject to lift and drag forces based on its angle of attack with the apparent wind. Its purpose is to help the sail maintain its desired AoA. The trim tab does this by creating a counteracting moment opposing the moment created by the wind on the wingsail when the wingsail develops any AoA. A static equilibrium between these moments balances out the wingsail, keeping it at the intended AoA.

6.2 Appendix B: Rigid Sailing and Trim Tab Information (Stafford, 2021)

Lift in a wing (or sail) is defined as the force vector perpendicular to the free stream fluid velocity (henceforth called "apparent wind direction or AWD"). A symmetric airfoil (or, indeed, hydrofoil) develops lift anytime there is an angle between the AWD and the line from leading edge to trailing edge of the wing (the "chord line"). This is called the "angle of attack or AoA". On a sailboat it should be clear that as long as the AWD has a non-zero angle with the intended direction of travel (this angle is called the "sailing angle or SA"), the lift vector can be broken into a component that would tend to drive the sailboat forward and an orthogonal component that would tend to drive the sailboat in the leeward/sideways direction. Practically speaking, the minimum SA on a well-designed SailBot where the forward force component is sufficient to make forward progress will be about 30-40 degrees.

1) When on a starboard tack, the trim tab will have an offset in a CW direction from its boom centerline (Port tack: CCW; i.e. the trim tab should always point to windward relative to its boom).

2) Providing the critical hull heel angle is not exceeded (~40 degrees) and the boat is commanded to move in a direction not downwind, the Teensy should always be controlling the trim tab to attain the max L/D AoA as measured by the wingsail's AoA sensor.

3) When going downwind, Teensy should set the trim tab for max excursion (port or starboard).

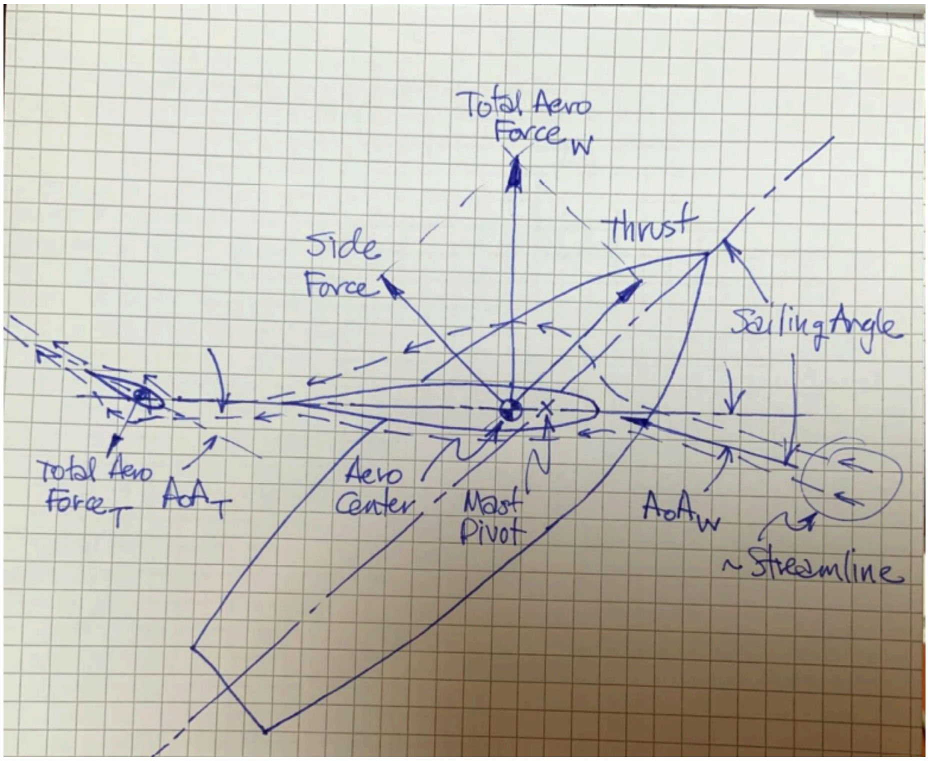
4) When heel angle exceeds preset max, Teensy needs to command trim tab so as to reduce the wingsail's AoA.

5) The Teensy needs the following inputs:
    a. Wingsail AoA from wingsail sensor (to adjust servo)
    b. Angle of heel from AirMar sensor (to enter feathering/reduced heel realm)
    c. Commands from hull:
        i. Zero lift (to stop and the failsafe condition)
        ii. Starboard Max L/D (to go upwind or reach with wind coming across starboard side of boat)
        iii. Port Max L/D (ditto for wind on port side)
        iv. Starboard Max Drag (to go straight downwind from starboard reach)
        v. Port Max Drag (ditto from port reach)

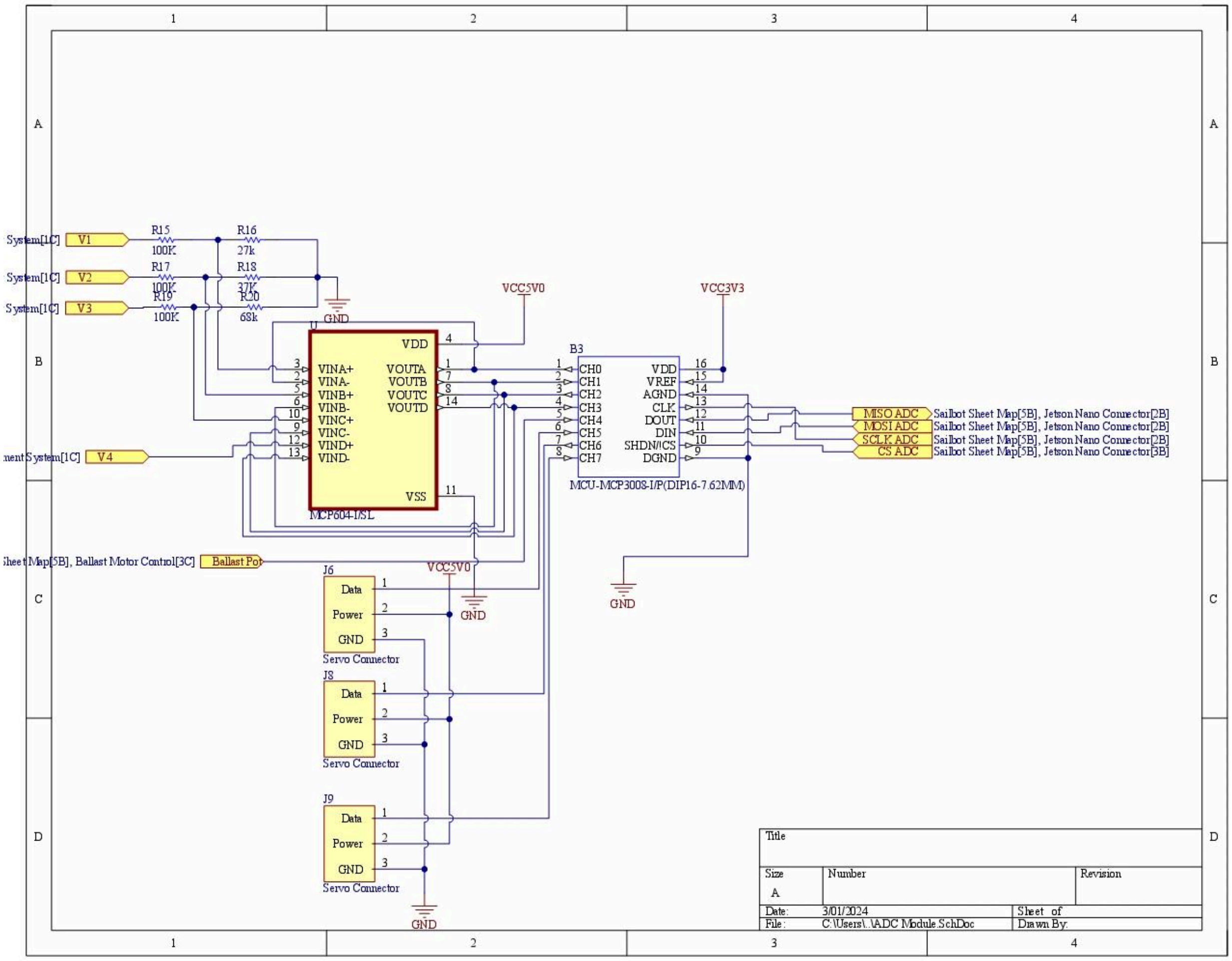
A few other considerations are worth noting.

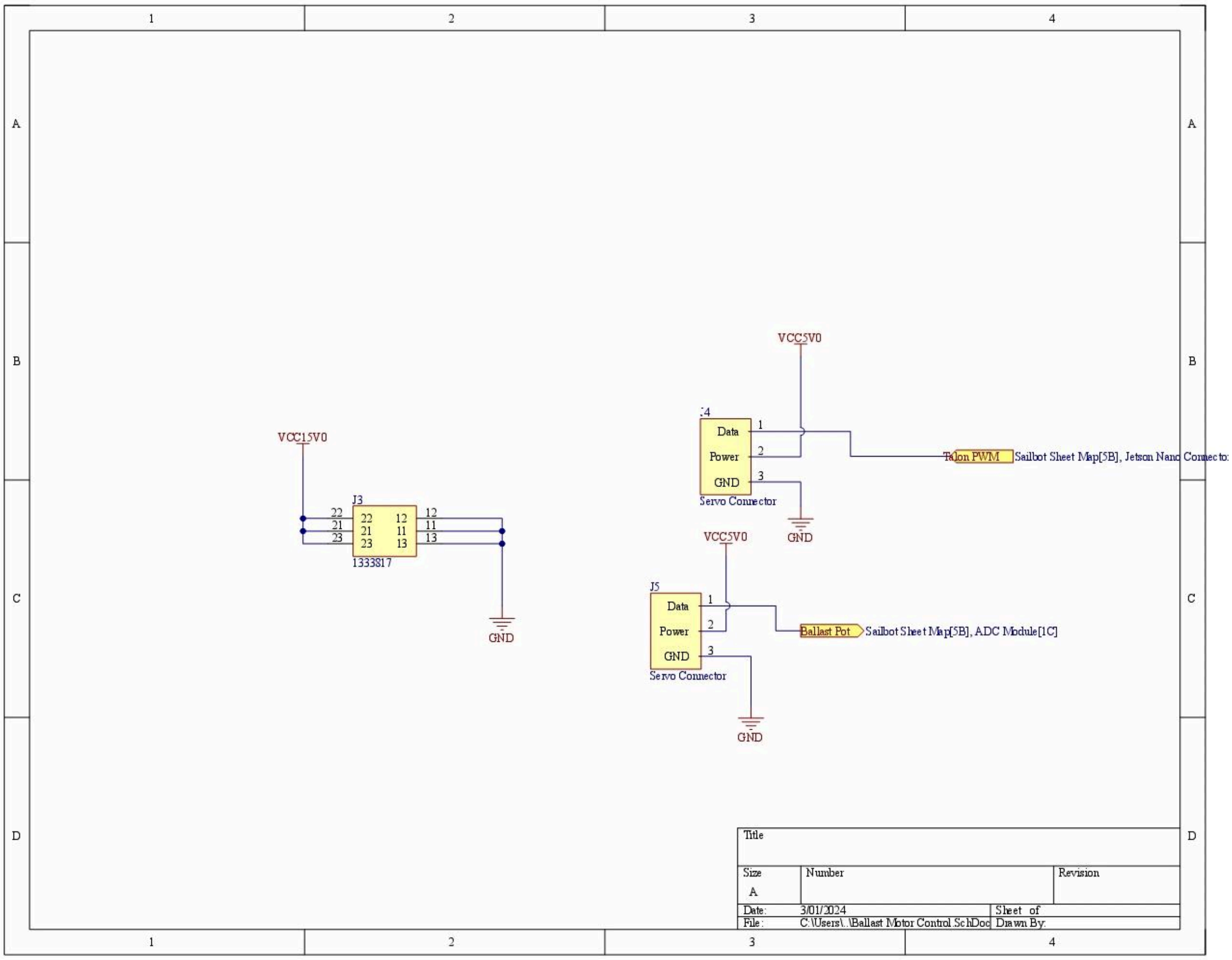
1) First is that the AWD (measured by the AirMar) is only equal to the true wind direction when SailBot is not moving. Once underway, the true wind direction and velocity are only available through calculation (by the AirMar) by doing the vector math with the actual hull track angle and speed.
2) The SA will normally be close but not exactly the angle between boat heading and AWD. To produce forces to counteract the leeward wingsail force component, the underwater "wingsail", i.e. the keel, needs a positive AoA (it is also a symmetrical foil). When going hard to windward, the keel's AoA can be up to 15 degrees (much larger if you get too slow or attempt to sail too close to the wind direction). This means that SailBot's track will vary from its heading by the keel's AoA.
3) The "tell-tales" on the wingsail are the best way to see if it is at the max L/D AoA. When set appropriately, both windward and leeward ribbons should be streaming aft with the leeward ones occasionally starting to lift off the wing surface. When the leeward ones are streaming downward or in the reverse direction it means the wingsail's AoA is too large and the wingsail is "stalled" (high drag/minimum lift).

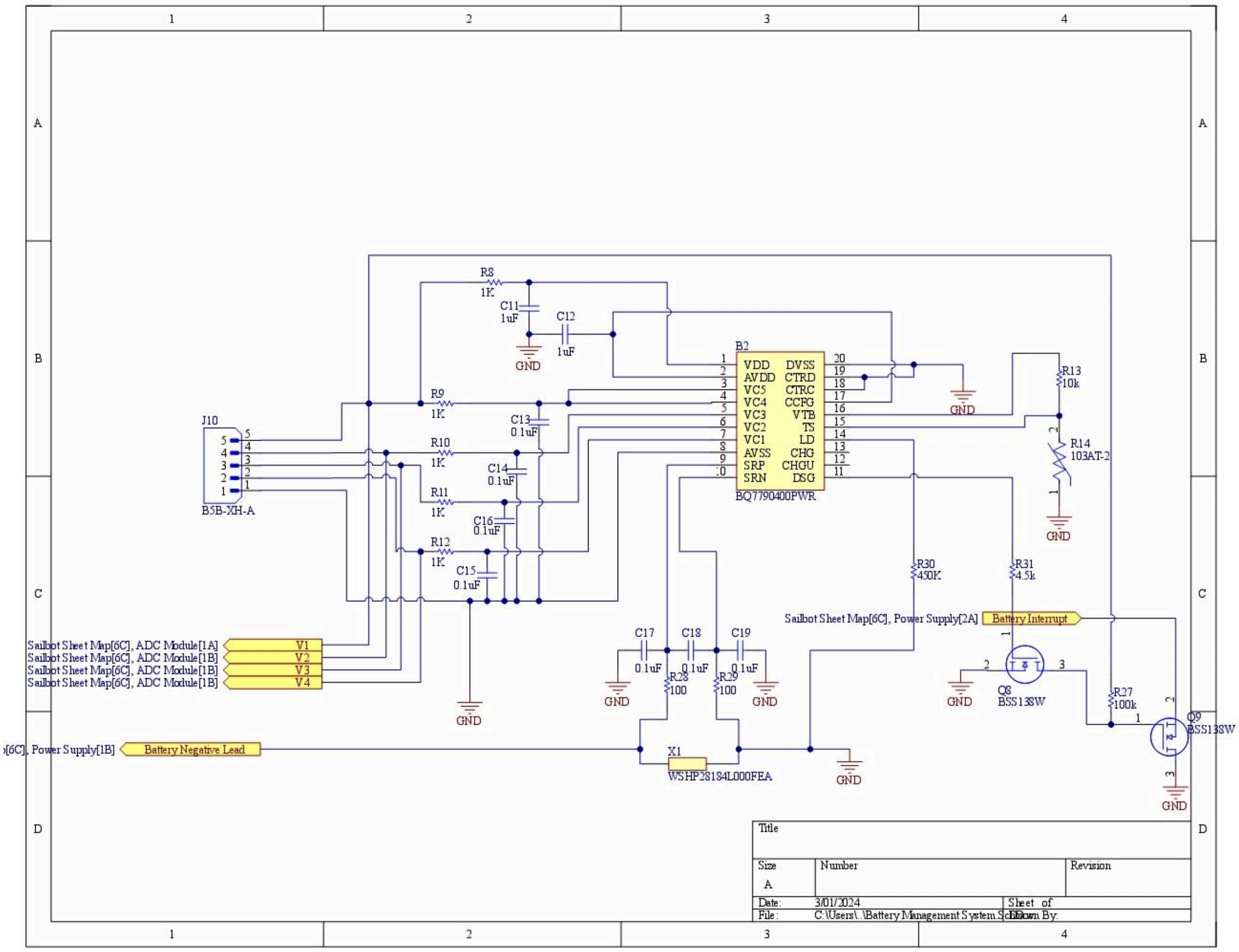
4) There are actually 5 separate symmetrical foils on SailBot, all subject to AoA, lift/drag, and stall considerations. These are of course the wingsail, the trim tab, the keel, and the rudders.

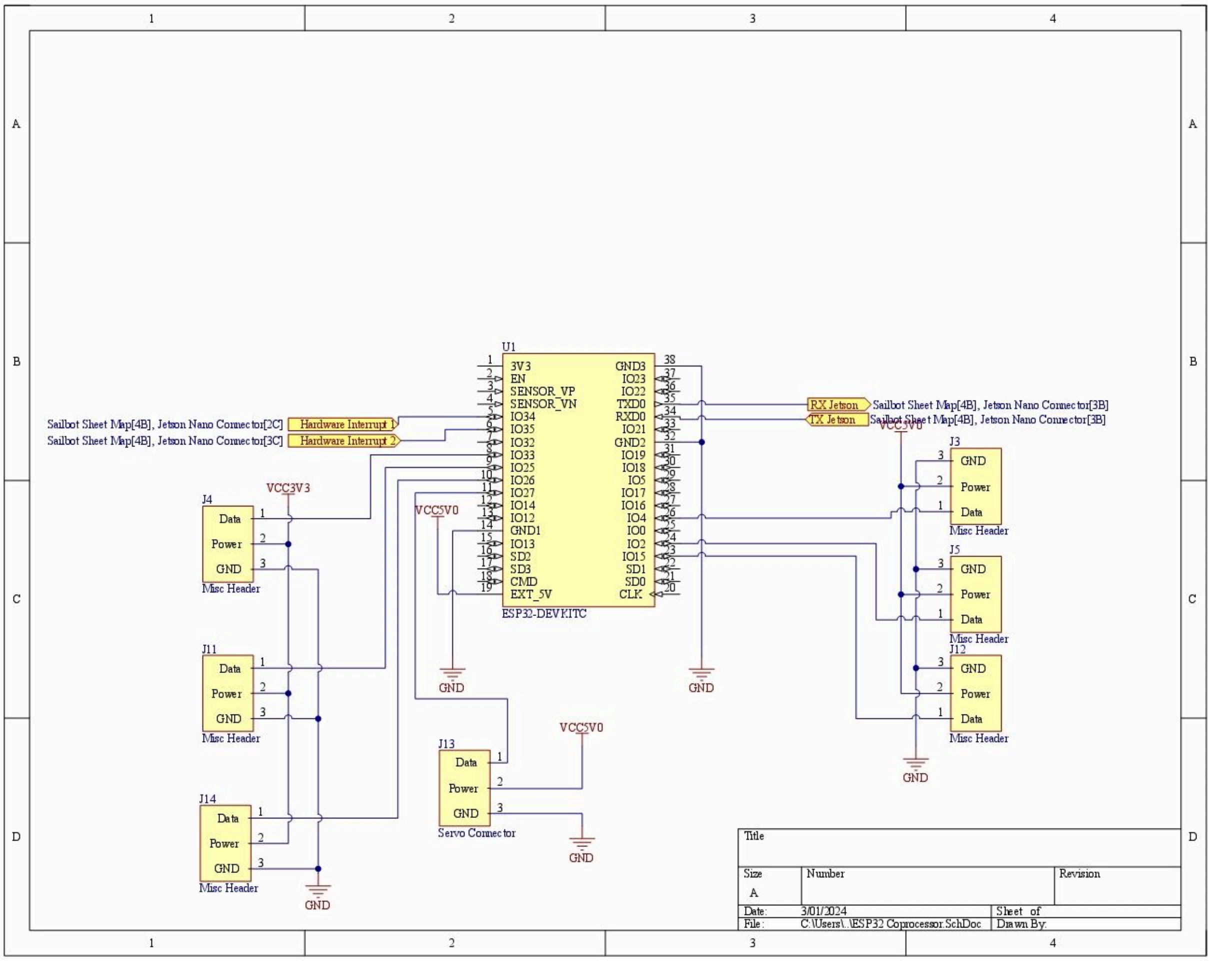


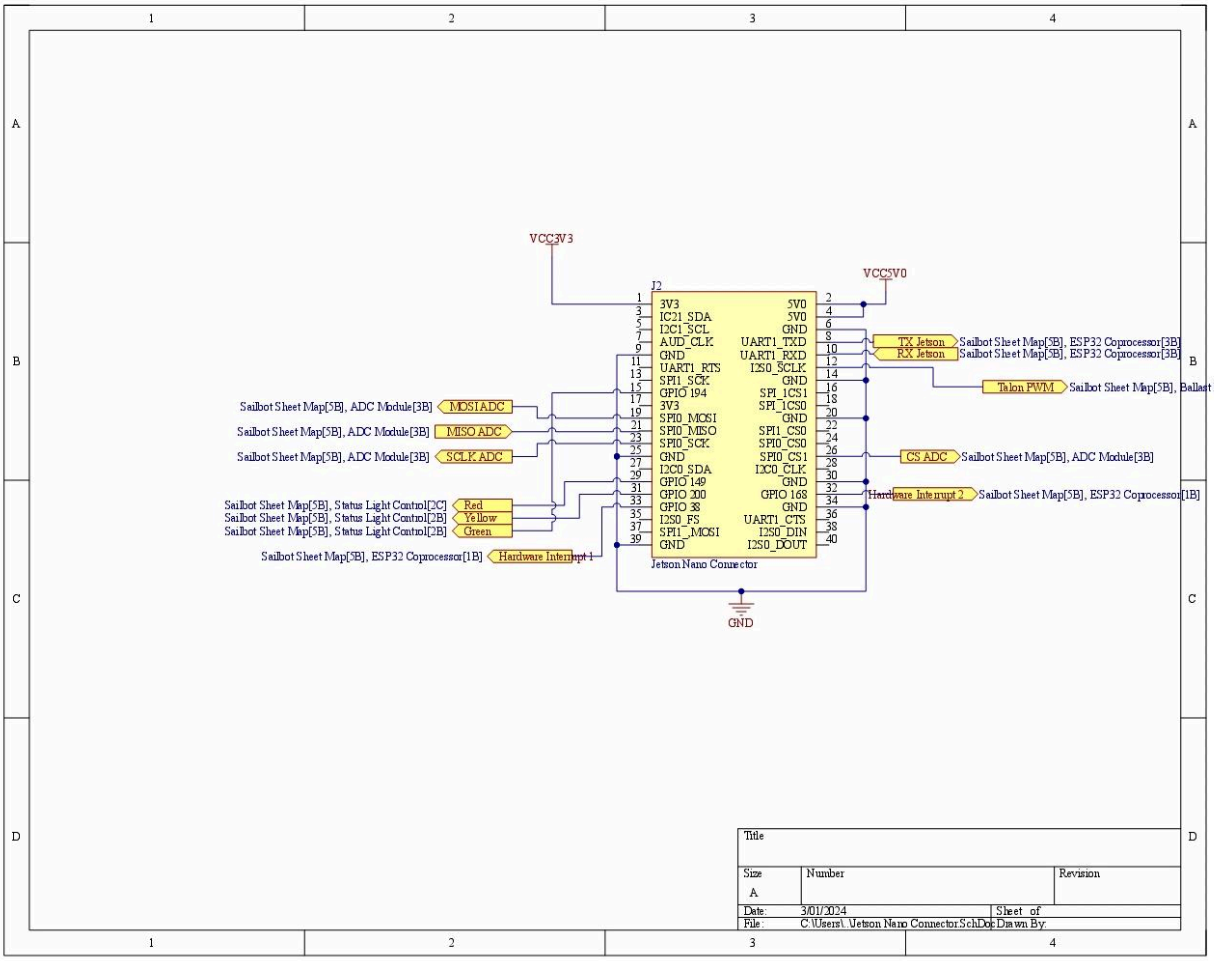
6.3 Appendix C: Electrical Schematics

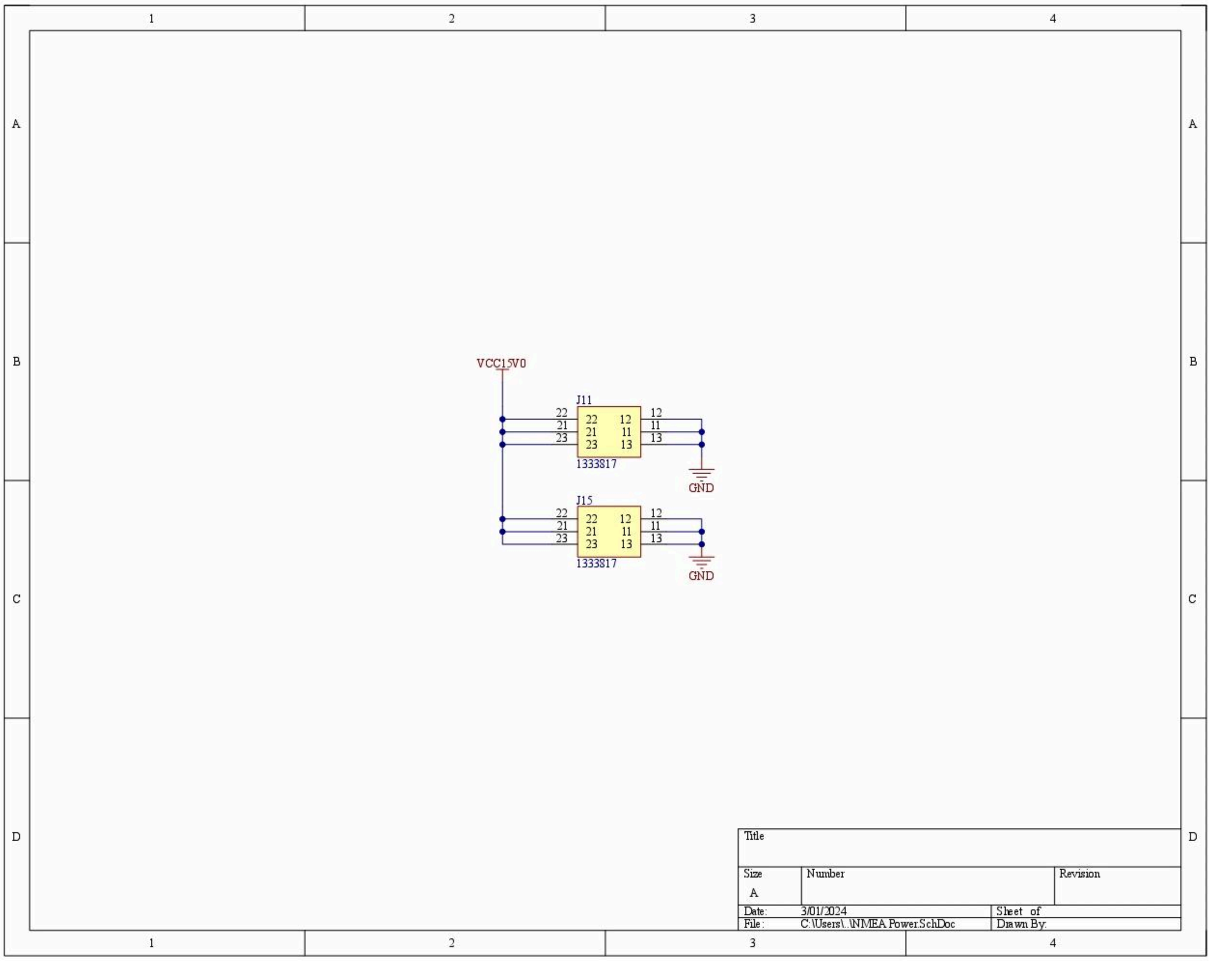


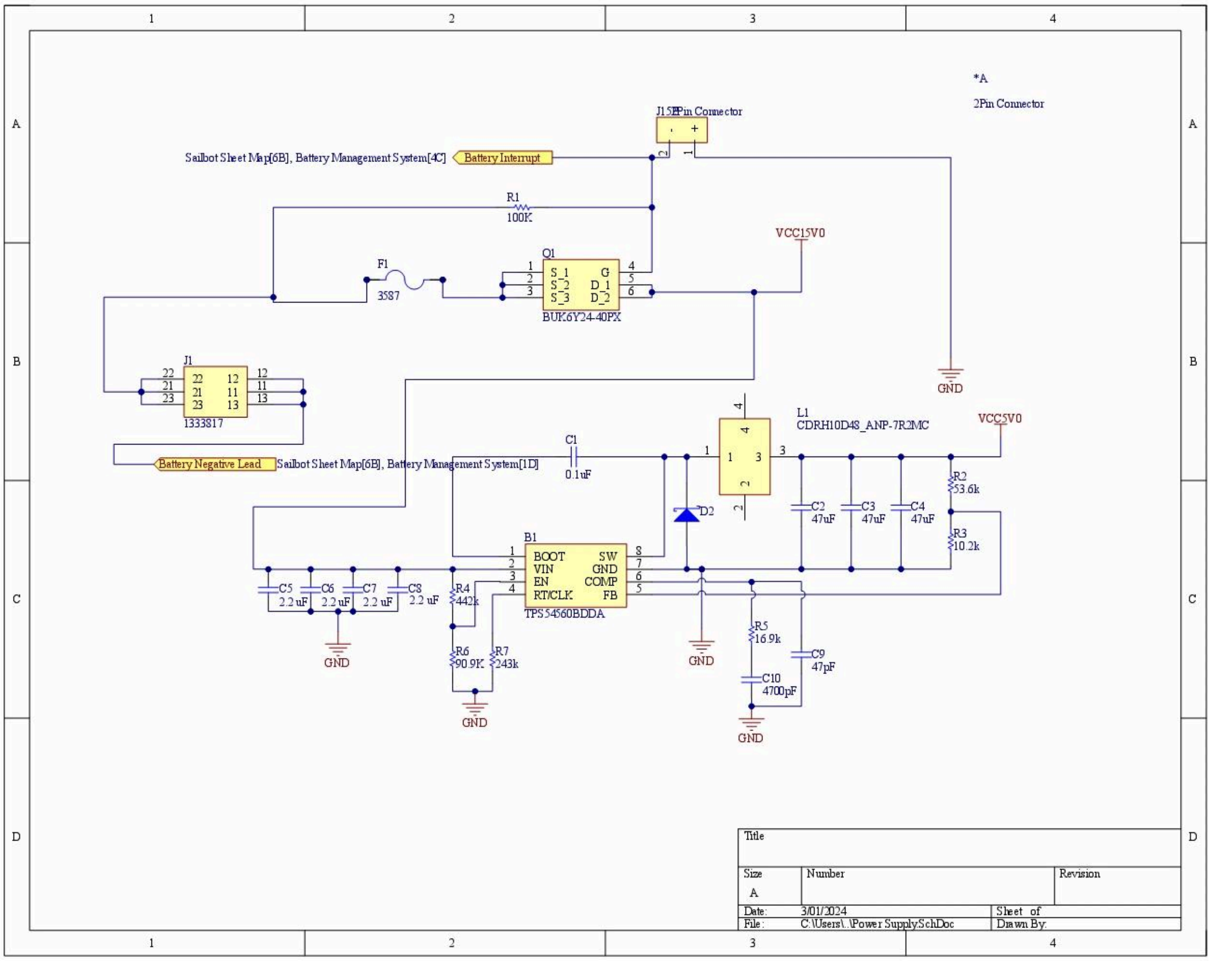

VCC15V0
J3
22
21
23
22
21
23
12
11
13
12
11
13
1333817
GND
VCC5V0
J4
Data
Power
GND
1
2
3
Servo Connector
GND
Talon PWM
Sailbot Sheet Map[5B], Jetson Nano Connecto:
VCC5V0
J5
Data
Power
GND
1
2
3
Servo Connector
Ballast Pot
Sailbot Sheet Map[5B], ADC Module[1C]
GND
Title
Size
A
Number
Revision
Date:
3/01/2024
Sheet of
File:
C:\Users\..\Ballast Motor Control.SchDoc
Drawn By:
1
2
3
4
A
B
C
D

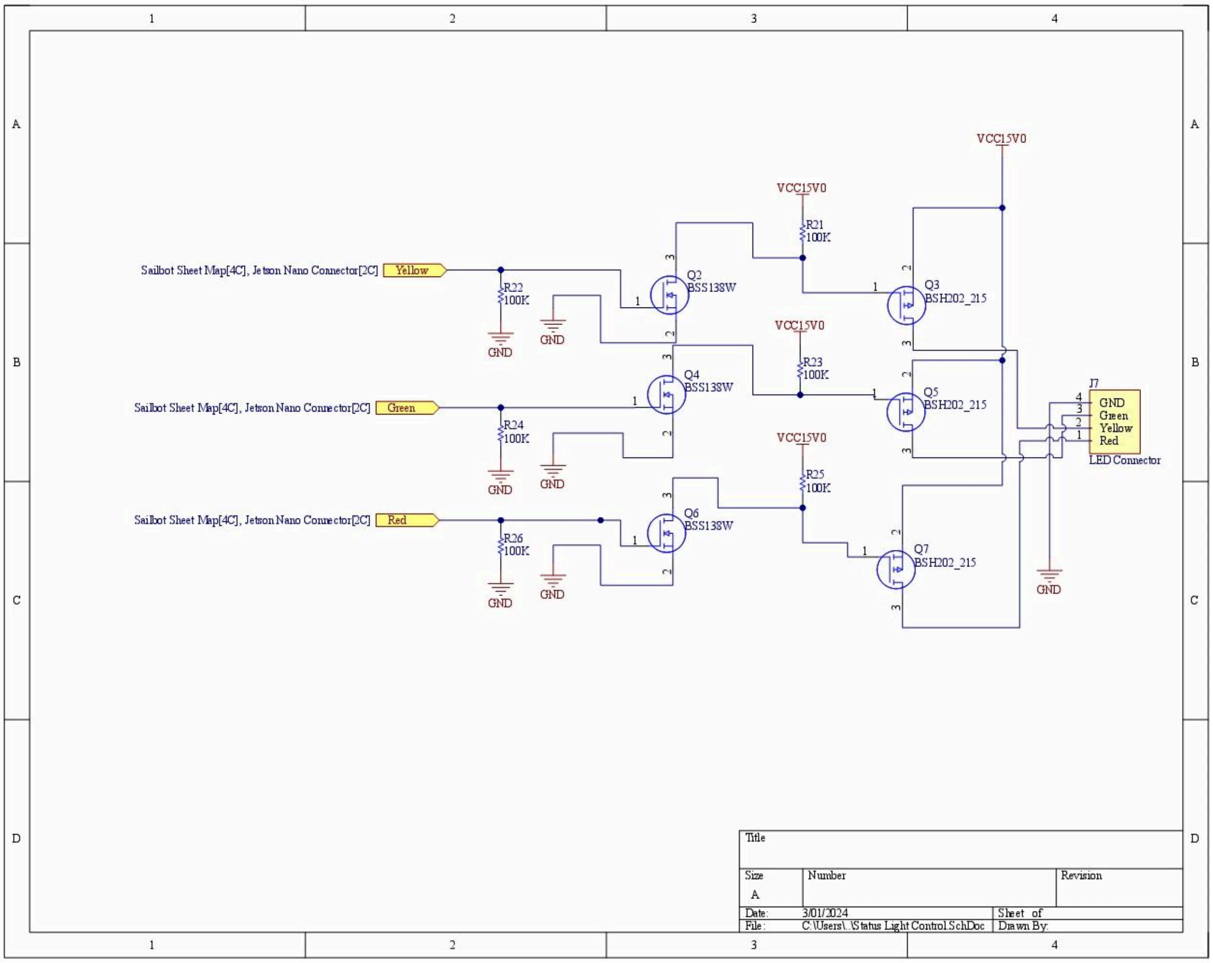

1
2
3
4
A
B
C
D
R8
1K
C11
1uF
C12
1uF
GND
B2
VDD
AVDD
VC5
VC4
VC3
VC2
VC1
AVSS
SRP
SRN
DVSS
CTRD
CTRC
CCFG
VTB
TS
LD
CHG
CHGU
DSG
BQ7790400PWR
R13
10k
R14
103AT-2
J10
B5B-XH-A
R9
1K
R10
1K
R11
1K
R12
1K
C13
0.1uF
C14
0.1uF
C16
0.1uF
C15
0.1uF
R30
450K
R31
4.5k
Sailbot Sheet Map[6C], Power Supply[2A]
Battery Interrupt
Sailbot Sheet Map[6C], ADC Module[1A]
V1
Sailbot Sheet Map[6C], ADC Module[1B]
V2
Sailbot Sheet Map[6C], ADC Module[1B]
V3
Sailbot Sheet Map[6C], ADC Module[1B]
V4
C17
0.1uF
C18
0.1uF
C19
0.1uF
R28
100
R29
100
Q8
BSS138W
R27
100k
Q9
BSS138W
[6C], Power Supply[1B]
Battery Negative Lead
X1
WSHP28184L000FEA
Title
Size
A
Number
Revision
Date:
3/01/2024
Sheet of
File:
C:\Users\..\Battery Management System.SchDoc
Drawn By:


1
2
3
4
A
B
C
D
U1
ESP32-DEVKITC
3V3
EN
SENSOR_VP
SENSOR_VN
IO34
IO35
IO32
IO33
IO25
IO26
IO27
IO14
IO12
GND1
IO13
SD2
SD3
CMD
EXT_5V
GND3
IO23
IO22
TXD0
RXD0
IO21
GND2
IO19
IO18
IO5
IO17
IO16
IO4
IO0
IO2
IO15
SD1
SD0
CLK
Sailbot Sheet Map[4B], Jetson Nano Connector[2C]
Hardware Interrupt 1
Sailbot Sheet Map[4B], Jetson Nano Connector[3C]
Hardware Interrupt 2
RX Jetson
Sailbot Sheet Map[4B], Jetson Nano Connector[3B]
TX Jetson
Sailbot Sheet Map[4B], Jetson Nano Connector[3B]
VCC3V3
VCC5V0
VCC5V0
VCC5V0
J4
Data
Power
GND
Misc Header
J11
Data
Power
GND
Misc Header
J14
Data
Power
GND
Misc Header
J13
Data
Power
GND
Servo Connector
J3
GND
Power
Data
Misc Header
J5
GND
Power
Data
Misc Header
J12
GND
Power
Data
Misc Header
GND
GND
GND
GND
GND
Title
Size
A
Number
Revision
Date:
3/01/2024
Sheet of
File:
C:\Users\..\ESP32 Coprocessor.SchDoc
Drawn By:


VCC3V3
VCC5V0
J2
1 3V3
3 IC21_SDA
5 I2C1_SCL
7 AUD_CLK
9 GND
11 UART1_RTS
13 SPI1_SCK
15 GPIO 194
17 3V3
19 SPI0_MOSI
21 SPI0_MISO
23 SPI0_SCK
25 GND
27 I2C0_SDA
29 GPIO 149
31 GPIO 200
33 GPIO 38
35 I2S0_FS
37 SPI1_,MOSI
39 GND
2 5V0
4 5V0
6 GND
8 UART1_TXD
10 UART1_RXD
12 I2S0_SCLK
14 GND
16 SPI_1CS1
18 SPI_1CS0
20 GND
22 SPI1_CS0
24 SPI0_CS0
26 SPI0_CS1
28 I2C0_CLK
30 GND
32 GPIO 168
34 GND
36 UART1_CTS
38 I2S0_DIN
40 I2S0_DOUT
Jetson Nano Connector
GND
Sailbot Sheet Map[5B], ADC Module[3B] MOSI ADC
Sailbot Sheet Map[5B], ADC Module[3B] MISO ADC
Sailbot Sheet Map[5B], ADC Module[3B] SCLK ADC
Sailbot Sheet Map[5B], Status Light Control[2C] Red
Sailbot Sheet Map[5B], Status Light Control[2B] Yellow
Sailbot Sheet Map[5B], Status Light Control[2B] Green
Sailbot Sheet Map[5B], ESP32 Coprocessor[1B] Hardware Interrupt 1
TX Jetson Sailbot Sheet Map[5B], ESP32 Coprocessor[3B]
RX Jetson Sailbot Sheet Map[5B], ESP32 Coprocessor[3B]
Talon PWM Sailbot Sheet Map[5B], Ballast Mo
CS ADC Sailbot Sheet Map[5B], ADC Module[3B]
Hardware Interrupt 2 Sailbot Sheet Map[5B], ESP32 Coprocessor[1B]
1
2
3
4
A
B
C
D
Title
Size
A
Number
Revision
Date: 3/01/2024
Sheet of
File: C:\Users\..\Jetson Nano Connector.SchDoc
Drawn By:


1
2
3
4
A
B
C
D
VCC15V0
J11
22
21
23
12
11
13
1333817
GND
J15
22
21
23
12
11
13
1333817
GND
Title
Size
A
Number
Revision
Date:
3/01/2024
Sheet of
File:
C:\Users\..\NMEA Power.SchDoc
Drawn By:


*A
2Pin Connector
J15: 2Pin Connector
Sailbot Sheet Map[6B], Battery Management System[4C]
Battery Interrupt
R1
100K
VCC15V0
F1
3587
Q1
S_1
S_2
S_3
G
D_1
D_2
BUK6Y24-40PX
GND
J1
1333817
Battery Negative Lead
Sailbot Sheet Map[6B], Battery Management System[1D]
L1
CDRH10D48_ANP-7R2MC
VCC5V0
C1
0.1uF
D2
R2
53.6k
C2
47uF
C3
47uF
C4
47uF
R3
10.2k
B1
BOOT
VIN
EN
RT/CLK
SW
GND
COMP
FB
TPS54560BDDA
C5
2.2 uF
C6
2.2 uF
C7
2.2 uF
C8
2.2 uF
R4
442k
R6
90.9K
R7
243k
R5
16.9k
C9
47pF
C10
4700pF
GND
Title
Size
A
Number
Revision
Date: 3/01/2024
Sheet of
File: C:\Users\..\Power Supply.SchDoc
Drawn By:


Sailbot Sheet Map[4C], Jetson Nano Connector[2C]
Yellow
Sailbot Sheet Map[4C], Jetson Nano Connector[2C]
Green
Sailbot Sheet Map[4C], Jetson Nano Connector[2C]
Red
R22
100K
R24
100K
R26
100K
R21
100K
R23
100K
R25
100K
VCC15V0
VCC15V0
VCC15V0
VCC15V0
Q2
BSS138W
Q4
BSS138W
Q6
BSS138W
Q3
BSH202_215
Q5
BSH202_215
Q7
BSH202_215
GND
J7
GND
Green
Yellow
Red
LED Connector
Title
Size
A
Number
Revision
Date:
3/01/2024
Sheet of
File:
C:\Users\..\Status Light Control.SchDoc
Drawn By: