

April 2019

## WPI Sailbot 2018-2019

Kellen B. Randall  
*Worcester Polytechnic Institute*

Sierra Taylor Palmer  
*Worcester Polytechnic Institute*

Follow this and additional works at: <https://digitalcommons.wpi.edu/mqp-all>

---

### Repository Citation

Randall, K. B., & Palmer, S. T. (2019). *WPI Sailbot 2018-2019*. Retrieved from <https://digitalcommons.wpi.edu/mqp-all/7024>

This Unrestricted is brought to you for free and open access by the Major Qualifying Projects at Digital WPI. It has been accepted for inclusion in Major Qualifying Projects (All Years) by an authorized administrator of Digital WPI. For more information, please contact [digitalwpi@wpi.edu](mailto:digitalwpi@wpi.edu).



Worcester Polytechnic Institute  
Robotics Engineering Program

Sailbot 2018-2019 Final Report

---

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 by:

Sierra Palmer, RBE  
Kellen Randall, ME  
Sydney Fisher, RBE

---

Project Advisors:  
Professor Kenneth Stafford  
Professor William Michalson

Date: 25 April, 2019

# **1 Abstract**

The goal of this Major Qualifying Project, or MQP, is to create an autonomous sailboat, known as "The Wide Awake", that builds upon lessons learned from previous WPI Sailbot projects. The team used the International Robotics Sailing Regatta (IRSR) rules to guide the creation of the boat for the 2019 competition. These guidelines included, but were not limited to, the following challenges: precision navigation, fleet race, and endurance. The final products of this MQP were a more mechanically and technically reliable boat, a better navigation system, and a user-friendly guide on how to run and manage "The Wide Awake".

# Contents

<b>1</b>	<b>Abstract</b>	<b>i</b>
<b>2</b>	<b>Introduction</b>	<b>1</b>
<b>3</b>	<b>Background</b>	<b>2</b>
3.1	What is a sailboat? . . . . .	2
3.2	The physics of sailing . . . . .	2
3.3	Right of Way Rules . . . . .	3
3.4	International Robotic Sailing Regatta (IRSR) . . . . .	4
3.5	Data about Lake Quinsigamond . . . . .	6
3.6	Previous WPI Sailbots . . . . .	7
3.6.1	Mechanical Systems . . . . .	7
3.6.2	Electrical Systems . . . . .	9
3.7	Software System . . . . .	11
<b>4</b>	<b>Purpose</b>	<b>12</b>
<b>5</b>	<b>Methodology</b>	<b>13</b>
5.1	System Overview . . . . .	13
5.2	Mechanical Systems . . . . .	13
5.2.1	Keel Design . . . . .	13
5.2.2	Rigid Sail Design . . . . .	16
5.3	Electrical Systems . . . . .	19
5.3.1	Arduino MEGA with Ethernet and USB Shield . . . . .	21
5.3.2	Teensy 3.5 with ESP8266WiFi Chip . . . . .	23
5.3.3	CTRE HERO Board . . . . .	27
5.3.4	Maretron . . . . .	29
5.4	Software System . . . . .	30
5.4.1	Course Logic . . . . .	30
5.4.2	Reusable Code . . . . .	33
<b>6</b>	<b>Preliminary Calculations</b>	<b>38</b>
<b>7</b>	<b>Future Recommendations</b>	<b>39</b>
7.1	Mechanical Recommendations . . . . .	39
7.2	Hardware Recommendations . . . . .	39
7.3	Software Recommendations . . . . .	40
<b>8</b>	<b>Appendices</b>	<b>41</b>
8.1	References . . . . .	41
8.2	Roll and Pitch Calculations . . . . .	43
8.3	User Manual . . . . .	44
8.3.1	Mechanical Setup . . . . .	44



8.3.2	Hardware and Software Setup . . . . .	44
8.4	Code Description . . . . .	45
8.4.1	Arduino Code . . . . .	45
8.4.2	HERO Code . . . . .	45
8.4.3	Teensy Code . . . . .	45
8.5	Testing Kit List . . . . .	46

## List of Figures

1	Parts of a Sailboat [1]	2
2	Points of Sail [2]	3
3	Right of Way Rules [3]	4
4	Weather Data for June 2016 [6]	6
5	Weather Data for June 2017 [6]	7
6	Weather Data for June 2018 [6]	7
7	NACA 0012 Airfoil used for keel shape	14
8	NACA 63018 Airfoil used for keel bulb	14
9	Keel hull insert design	15
10	Keel hull insert FEA	16
11	Deflection due to carbon fiber	17
12	Mast insert for rigid sail	18
13	Dyneema cable trusses	18
14	Stiffened section of rigid sail mast	19
15	Hardware Schematic	20
16	Arduino with shields setup	21
17	12V to 5V 2A converters for Arduino setup and rudder servo	22
18	Teensy with ESP8266	23
19	Cl/Cd v Alpha for Rigid Sail	26
20	HERO Board	27
21	Pinout of Gadgeteer cable from Talon User Manual	28
22	Maretron unit	29
23	Code Process for Endurance/Long Distance Challenge	31
24	Code Process for Precision Navigation Challenge	32
25	Code Process for Station Keeping	32
26	Rounding a buoy diagram	33
27	Lane cost layout	35
28	Wind heuristic	36
29	Calculations of roll and pitch	43
30	Continuation of calculations	43
31	Kit of parts list	46

## List of Tables

1	Trim tab LED Sequences . . . . .	24
2	AT commands used in code . . . . .	25
3	Trim tab angles . . . . .	26
4	NMEA-0183 Sentence Character Codes . . . . .	30
5	Pros and Cons of Lane Sailing Ideas . . . . .	34

## 2 Introduction

The 2018-2019 Sailbot Major Qualifying Project (MQP) is a continuation of two previous MQPs. The basic description of this project is to create an autonomous sailboat that consists of design and control components dictated by the rules of the International Robotic Sailing Regatta (IRSR). The goal for this year's MQP team is to improve upon previous systems to develop a boat that is easier to control and is more user friendly through the creation of user guides. The tasks that the boat will need to accomplish are as follows:

- A navigation test to maneuver around a triangular course marked by buoys
- A fleet race that uses manual control to maneuver around a triangular course
- An endurance test that has the boat sail around a rectangular course for seven hours
- A station keeping test that requires the boat stay within a 40x40 meter box for five minutes before exiting
- A payload test that moves cargo 200 meters, either autonomously or manually
- A collision avoidance test with a manned boat
- A search test that has the boat find an object within a 100 meter radius

Systems that the team improved upon included the keel, rudders, and wing sail. The previous keel moved the center of gravity from a position slightly aft of the center of the hull to seven inches further back. This caused the back of the boat to sit deeper in the water, while the bow was almost completely in the air. The shorter length of the previous rudders made turning difficult at times. Lastly, the wing sail had issues with stiffness and communication. The thin carbon fiber mast had too much play, and would cause flexure of up to 20 degrees at some points; this caused major issues at the point where the mast interacted with the boat. Another issue with the wingsail was its lack of communication with the rest of the boat. With those two major problems, the wingsail had to be replaced with a cloth sail for the 2018 boat. With all of these ideas in mind, the team created a more controllable boat that is easier to build upon in the future.

## 3 Background

### 3.1 What is a sailboat?

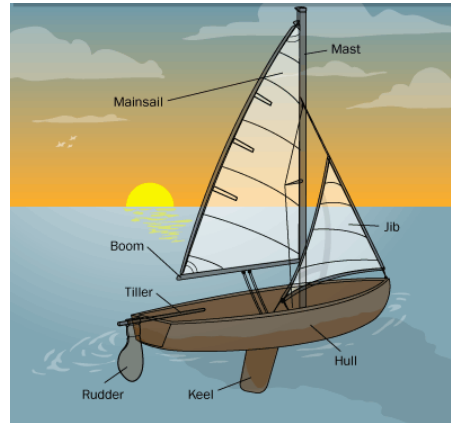


Figure 1: Parts of a Sailboat [1]

A sailboat is a boat that derives its locomotion from the wind. By catching the wind in its sail, a sailboat is able to harness the effort from the wind and translate it to a driving force. Depending on the amount of main sheet let out using a winch, the sail angle of the boat changes, providing optimal sailing in different conditions.

The sail also creates a heeling moment about the hull, which means that the boat will lean under the influence of the wind on the sails, while simultaneously creating forward lift. If left with no system to right this applied moment, a sailboat would capsize immediately. However, sailboats are fitted with keels, which deliver a moment opposite to that of the sail. The hull provides buoyancy to this system, and guides water around the boat to assist in steering. The boat's rudders are also used for steering of the sailboat, where pushing the tiller towards starboard and port turns the boat towards port and starboard, respectively.

### 3.2 The physics of sailing

Piloting a sailboat is a fairly simple process. The sailor hoists the sail upwards and the wind propels the boat based upon the position of the sail. However, taking a closer look uncovers much more of the physics required for this activity. Like airplane wings, sails function using Bernoulli's principle. This principle states that increased speed results in decreased pressure, therefore keeping the net energy the same. The sail is almost always curved to generate lift. The wind moving around the downwind side of the sail is forced to follow a longer route, while the wind moving around the upwind side of the sail is allowed to move faster, creating lift. The coupled forces of lift from the sail and the keel propel the boat forward, allowing a sailor to traverse the water.

In order for a sailboat to move, the sailor must know the six points of sail. The six points of sail

are based upon the wind direction in relation to how far the boat's sail is pulled in or let out. The first point of sail is the ninety-degree area around the direction of the wind. This is called the “no-go zone”, which is also referred to as sailing in irons (Labeled 1 in Figure 2). This zone is too close to the wind to sail effectively because the sail's lift force does not have a significant forward vector component to sail. It is possible, however, to sail by zigzagging across the zone, allowing one to head upwind. Upwind sailing is sailing towards the direction from which the wind is blowing. It includes two points of sail: Close-Hauled and Close Reaching (Labeled 2 and 3 in Figure 2).

Sailing across the wind is called Beam Reaching (Labeled 4 in Figure 2). Downwind sailing refers to sailing in the direction the wind is blowing. It includes both Broad Reaching and Running (Labeled 5 and 6 in Figure 2). These same points of sail can be assigned whether the wind is blowing over the right or left side of the boat as shown in the diagram below.

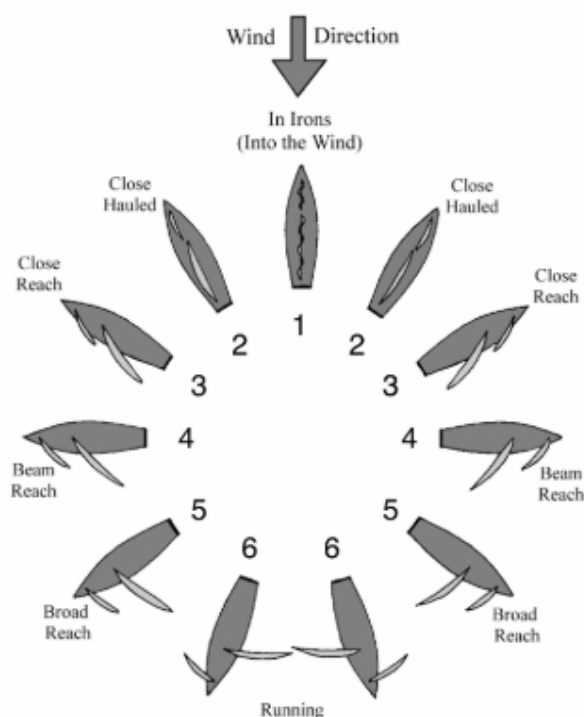


Figure 2: Points of Sail [2]

### 3.3 Right of Way Rules

Another concept all sailors need to understand while sailing are right of way situations, which happen when two boats occupy the same water space. A sailboat will sometimes have right of way over a motorboat depending on relative maneuverability and speed. A sailboat always has right of way with a power boat or canoe/kayak, unless it is overtaking such or the power craft is constrained in some

manner. Determining right of way between two sailboats is based on a the following few rules. First, right of way depends on where the wind is coming from and which direction the sailboat is facing. If the boats are on different tacks (sails on different sides of each boat), then the sailboat on the starboard tack (wind coming from the starboard side, with sails out on the port, or left, side) is the vessel with the right of way. The boat on port tack (wind coming from the starboard side) must change its course. If the two boats are on the same tack, the leeward (downwind) boat has the right of way, and the windward (vessel traveling upwind) boat must give way. Some examples of right of way situations can be seen in Figure 3, where the P stands for port tack, the S stands for starboard tack, and the wind is heading straight down the figure.

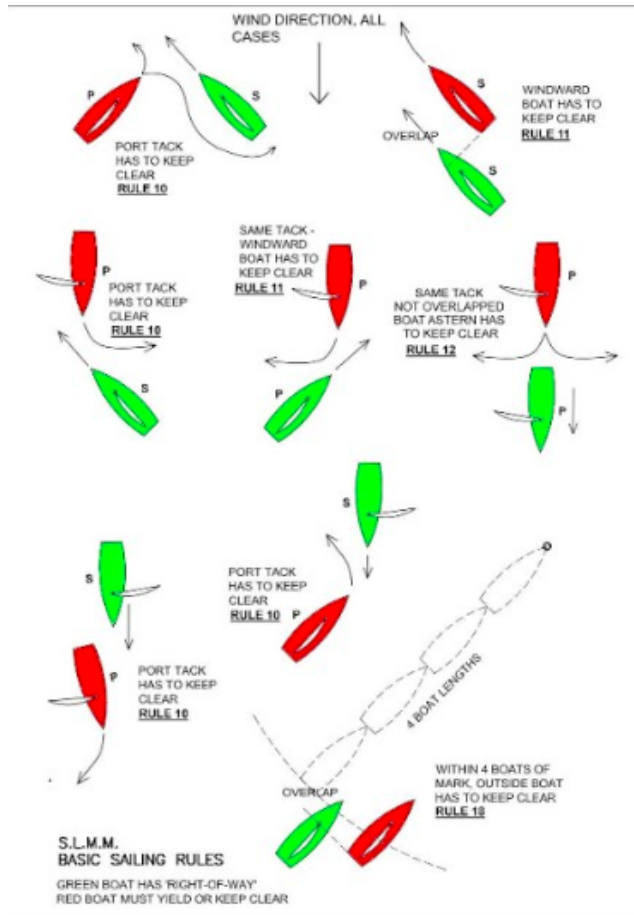


Figure 3: Right of Way Rules [3]

### 3.4 International Robotic Sailing Regatta (IRSR)

The International Robotic Sailing Regatta, also known as IRSR, is a robotic sailing competition where teams from all over the world build autonomous sailboats and come to compete in multiple challenges. There are multiple divisions and multiple challenges within each division. The Sailbot division boats

are no larger than two meters long and they compete in many categories listed above in the introduction. After once again winning last year's competition, the IRSR will be hosted by Worcester Polytechnic Institute in June of 2019.

The rules for the IRSR as of June 2014 are as follows [4]:

1. Overall length including hull, all spars and foils oriented in their fore and aft directions and at their maximum extensions if applicable, shall not exceed 2 meters measured parallel to the waterline. Sensors and their mountings are not included in the overall length measurement. Floating waterline must be clearly marked for use during measurement.
2. Beam shall not exceed 3 meters overall width at zero heel angle.
3. Number of hulls, depth, mast height, number of masts, sail area, and number of sails are unrestricted subject to the following event limitations:
  - (a) The draft in normal sailing condition shall not exceed 1.5 meters.
  - (b) Total overall height from the lowest underwater point to the highest point on the largest rig shall not exceed 5 meters. Sensors and their mountings are not included in the height measurement.
4. Teams shall provide a suitable stand to support the boat in a vertical position while fully assembled for the purposes of judging.
5. Boats shall not have any direct human contact during on-the-water events except as permitted by the event Notice of Competition and Sailing Instructions
6. External control shall be limited as specified in the event Notice of Competition and Sailing Instructions, however a means for full remote radio control is required at all times during competition to allow avoidance of collisions with other boats
7. Data transfer from the boat to shore is unlimited, but shall be on an approved frequency.
8. Radio frequencies used by each boat shall comply with host country regulations and are subject to approval before competition.
9. In each event, boats shall compete by sailing only (no alternate sources of propulsion).
10. Construction materials are not restricted provided they cannot cause environmental damage during operation. In particular, lead ballast must be completely sealed.
11. Power for onboard control systems may be provided by any source other than biological, and must be fully contained within the vessel.
12. The configuration of the boat shall not change during the course of any event. (i.e. components cannot be jettisoned or added during the race). Bilge pumps are permitted as long as they do not provide additional propulsion to the vessel.



13. Any parts may be replaced or repaired between events as necessary.
14. Sail configuration changes (hoisting/dousing) are allowed during a race provided such a change is initiated and executed only by the onboard systems.
15. In case of uncertainty about interpretation of these rules, please contact the event organizers for clarification.

### 3.5 Data about Lake Quinsigamond

In order to develop a better understanding of the sailing conditions that the boat would be encountering, the team looked at the weather data for the first week of June for 2016, 2017, and 2018, which coincides with the week of IRSR. With this data, it was found that there were, on average, thirteen days of fog and thirteen days of precipitation. During that first week, it was also found that the average wind speed was about fifteen miles per hour, which is important in regards to sail design. Below are graphs containing temperature, precipitation, and wind information during the first week of June 2016, 2017, and 2018. 2017 looked to be a little hotter than 2016 and 2018, with the temperatures reaching close to 90 degrees by the end of the week. However, in 2016 and 2018, the temperatures stayed around 80. It also showed that both years had at least one day of rainfall, with a wind speed range between 12 mph and 25 mph on June 9, 2016 [5].

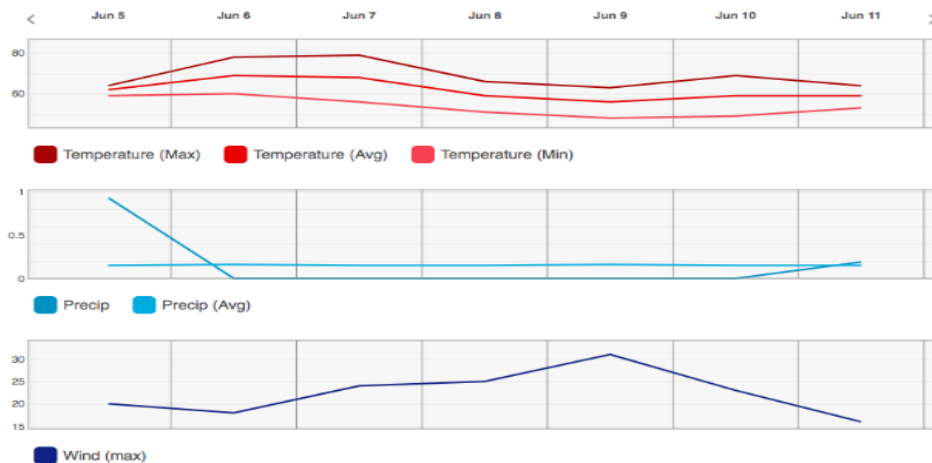


Figure 4: Weather Data for June 2016 [6]

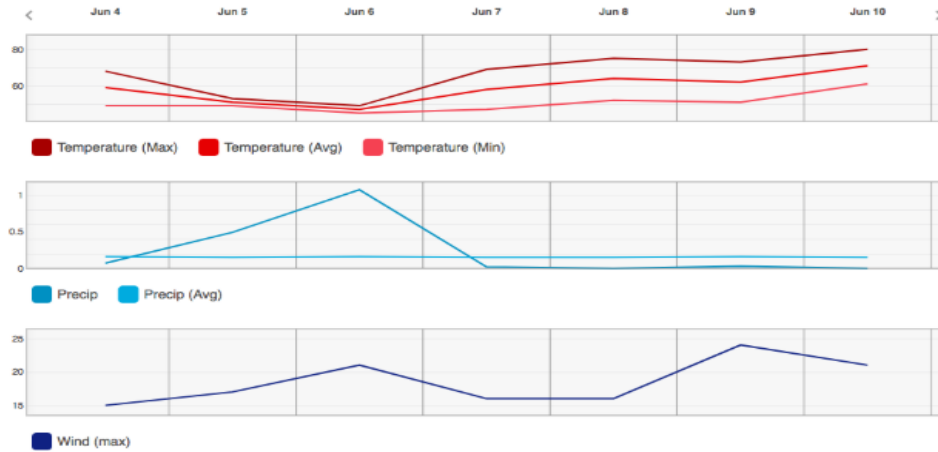


Figure 5: Weather Data for June 2017 [6]

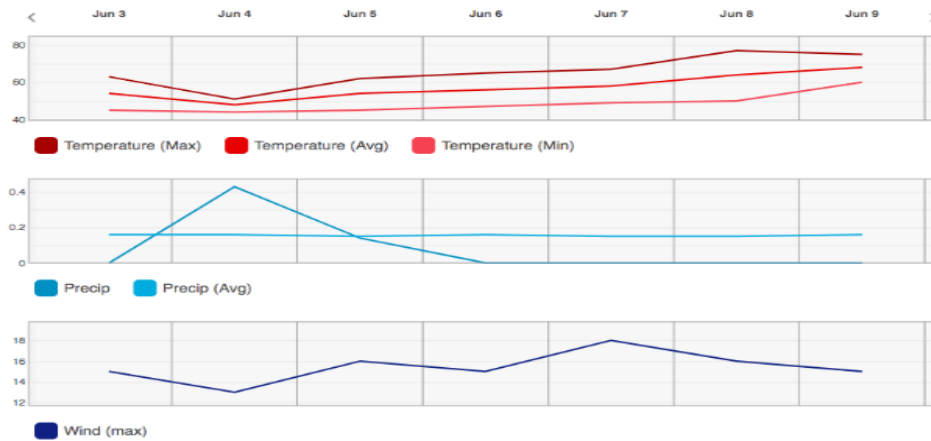


Figure 6: Weather Data for June 2018 [6]

## 3.6 Previous WPI Sailbots

The two previous MQP sailbots are from the 2016-2017 and 2017-2018 academic years. Each had its own pros and cons that allowed the current team to learn more about the operations of the boat and what improvements would need to be made over the 2018-2019 academic year.

### 3.6.1 Mechanical Systems

The mechanical systems of the boat are necessary for its motion, and are comprised of several sub-systems. The two previous competition boats consisted of a sail, winch system, keel, rudders, and a hull they built themselves. During the 2017-2018 year, a movable ballast was also added. The surfaces

of the boat generate different forces and interactions to provide an effective sailing vessel. By utilizing the 2017-2018 team's hull and movable ballast, the problem of integrating more effective and dynamic righting systems became the focus of this year's MQP on the mechanical side. This was done through improvements of the keel, rudders, and rigid wing.

## Hull

The 2016-2017 sailbot's mechanical systems focused on manufacturing a hull utilizing a form donated by the U.S. Naval Academy. The fabrication process included creating an internal structure made of plywood, fiberglassing over the structure, then creating a plywood and fiberglass deck to complete the system. This process was also used for the 2017-2018 sailbot. However, the team decided to design their own hull form with inspiration from racing yachts by creating a wider stern to make the boat easier to balance while rolling. Designing the hull from scratch also allowed for less water intrusion and more control into how the manufacturing could affect the mechanical and control system development.

## Sails

There are two types of sails that have been utilized over the years: rigid and cloth. Even though both types of sails had been created, the final boats in previous years only used a cloth sail. This was due to the incompatibility of the rigid sails and the boat designs. The first rigid sail was designed as an MQP back in 2017, but was unable to be used on the boat. For the 2017-2018 boat, another was designed in order to better fit the requirements for the rigid wing, which were as follows:

1. Must effectively sail under normal wind conditions under all points of sail
2. Must be able to depower itself completely
3. Must be wirelessly connected to the boat's controls
4. Should not be taller than 3.5 m
5. Must be easily disassembled for transport.

The design fit almost all of these criteria, but the mast was too flimsy, creating give in the z direction of almost 20 degrees in certain conditions. With this much flexibility, the sail was not useful for proper and effective sailing.

There were also many connection issues trying to operate the sail wirelessly through the boat's controls. These problems included dropped messages between the Teensy and the BeagleBone Black, and lack of serial communication between the ESP8266 Wifi Chip and the Teensy. This is something that this year's MQP team has worked on to improve.

## Winch System

A winch was designed for the 2017-2018 sailbot that would allow for better tensioning so the mainsheet would not get tangled while slack. This system was implemented after the rigid wing issues were discovered, as mentioned previously.

## Keel

The keel, as designed, was structurally deficient in providing balanced moments. This was in respect to the moments generated by the sail and those generated by the force of components on the boat. Some other issues that were discovered had to do with the structural integrity of the keel itself when a moment was exerted along the bottom of the hull. The keel had a tendency to create a larger than expected drag.

## Movable Ballast

One final system that was new to the 2017-2018 boat was the moveable ballast, which was made to help balance the boat's roll when heeling. This is the secondary system that provides righting moments to the boat by swinging a lead mass over the side of the boat to simulate a person hiking out. It consisted of a lead weight on a carbon fiber rod that was connected to custom gearbox designed by the previous team. This was then connected to a Mitsubishi window motor, which was run by a TalonSR motor controller, and then had its output inputted into a SCAMP. The SCAMP was a board that was originally created two years ago that would take inputs and convert them into NMEA2000 messages. Currently, the system generates a significant amount of variability in control. The gears include improper meshing and the shafts maintain a significant amount of play when moving.

## Rudders

Another system that was changed were the rudders, as the 2016-2017 boat had only one, while the 2017-2018 boat had two that operated through one servo.

### 3.6.2 Electrical Systems

The electrical system of the Sailbot has essentially stayed the same for the last two years (from 2016-2018). The main communication system of the boat is the NMEA2000 bus, which provided both power and communication to devices connected on the bus. The power that was provided was a 12 volt battery connected to a fuse box. This then split off into four different components, all meeting the waterproofing standards with its waterproof connectors. The components included the winch, the movable ballast, the BeagleBone Black with ethernet, and the motor controllers for the rudder. The Airmar sensor, used for GPS and wind direction, already used the NMEA2000 standard, which is another reason the NMEA2000 was chosen as the CAN bus standard. This led to the creation of the SCAMP two years ago, which would take CAN communication coming from the motor controllers, and then convert them to NMEA2000 messages so all communications were in the same standard when reaching the BeagleBone Black.

## Central Processing

Both systems used a Beaglebone Black as the central processor, but grew in the control structure and number of sensors over the years. They both utilized an Airmar 220WX WeatherStation that measured the GPS location of the boat, wind measurements, and contained a 9-axis IMU. This allowed

the boat to determine its location, wind speed and direction, current heading, and current heel angle in order to operate autonomously.

## Communication

The BeagleBone Black and the NVIDIA TX2 are connected to the onboard Wifi router and the long-distance radio through Ethernet. This allowed the systems to communicate both software updates and boat telemetry information. Wifi could be used along with Ethernet, but can only be used when the boat is in range. Wifi allows for short range communication for data intensive tasks, such as deploying the code.

## Microcontrollers and Motor Controllers

A custom unit was developed that was simpler to program using the Arduino IDE, an Integrated Development Environment (IDE) that is familiar to many robotics engineering students at WPI. They did so by using an ATmega328p microcontroller that ran off an 8 MHz clock, a MCP2515 CAN controller that used SPI to communicate between the microcontroller and the NMEA bus, and a VN12SP30 motor driver IC. For the 2017-2018 Sailbot, PWM motor controllers drove the winch motor, while the custom PWM controller controlled the rudder servo over the CAN bus. As mentioned previously, they utilized a CAN bus to control the ballast and winch, and a driver IC to reduce the complexity and footprint of the PCB. This worked to cut down on the amount of time that had to be spent debugging. All of the CAN busses were then connected to the NMEA2000 bus that allowed for communication throughout the boat.

A few more design aspects went into the 2017-2018 boat to fix the microcontrollers, such as choosing a higher clock speed to allow for more complex algorithms to be used while also allowing the boat to be more deterministic in certain scenarios. They went with a new microcontroller that had better built-in CAN peripherals that could allow for direct communication between the bus and the controller without requiring external hardware to check voltages.

## On-boards Displays and Cameras

One requirement for the team was keeping track of all systems on the boat externally by displaying their status. In order to achieve this, they selected an ePaper display with no backlight. It does not create a power drain while also making it easy to read on a sunny day, which was important for when the team was out on the water.

A camera connected to a Raspberry Pi Zero ran OpenCV for hue and saturation segmentation to help with buoy detection. Once detected, the buoy's location would be sent over the wifi connection to the Beaglebone Black, where it would generate the algorithm to create the correct heading to move to the desired location. The custom boards for the winch were then notified to reel in or take out the mainsheet to move the sail in the correct direction. All of these actions were communicated over an Inter-Process Communication (IPC) mechanism, which allowed for code to run simultaneously on the boat and in simulation.

### 3.7 Software System

In terms of software, not much is known about the previous boats due to lack of documentation. The most that was available was the use of OpenCV and a camera to do buoy tracking. This was done by creating waypoints around the buoys and using the camera detection to make sure the boat was still on the correct course. However, this was mostly achieved through putting in the waypoints manually and using the camera to correct itself rather than doing navigation planning.

The processes on the boat are run by the software on the BeagleBone Black. This system ran by using a publisher-subscriber architecture, which runs by sharing messages between individual processes, known as nodes. Each node provides a different function to the system. The three main nodes were the Primary Control Node, the State Estimator Node, and the CAN Node. The Primary Control Node controlled the path planning and overall decision making of the robot. The State Estimator Node provided the wind, heading, and position data, which was used to produce motor control outputs. The CAN Node communicated these outputs to the motor controllers on the CAN bus. There are other nodes in the system, but they were used mostly for debugging.

#### Visual Buoy Identification

Last year's Sailbot used a special colorspace transform that was used to pre-filter noise. While the filtering found the buoy in a frame, a particle filter estimates the location of the buoy. This system worked out fairly well, but the boat would often detect a buoy and continue in its general direction instead of turning to face the buoy.

## 4 Purpose

An autonomous sailboat has many real life applications, including long term data collection and continuous monitoring. An autonomous sailboat does not need a crew and utilizes the wind to generate movement, so long sailing times can be achieved due to low power consumption. With no need to refuel with food and gasoline, the only limiting factor is battery life. Instruments can be included on the open hull and deck to monitor and test the surrounding environment as there is no need for crew accommodations. Due to the Sailbot's use of wind, it can also move to provide coverage to a large coastline or body of water. The applications extend to scientific, military, and rescue capabilities.

The purpose of this MQP was to create an autonomous sailboat that built upon lessons learned from the previous Sailbot projects, as well as find ways to improve the longevity of this project by creating more documentation and informational videos for sailbot operation. In the following chapters, the team will describe the mechanical, electrical, and software design choices and their use. While the team is utilizing the same hull as last year's project, the paper will be going into detail about the changes that occurred in the boat's keel, rudders, and sail. The change in the hardware architecture will be described, as well as some of the software system flows.

## 5 Methodology

### 5.1 System Overview

On an autonomous sailboat, there are several parts that help to control and propel the sailboat. These systems can be expressed through three different categories. The mechanical components which govern the performance of the boat and physically interact with environment, the electrical components which interact with the mechanical components and cause for the communication throughout the boat and the software components which send and process the signals required for guiding the boat and controlling its movement.

### 5.2 Mechanical Systems

For the mechanical systems, all equations to model the physics of the boat and its coinciding systems, such as the keel and the sail, were integrated together to better inform design choices. Once all of the forces were calculated, the systems themselves were developed. The three major components that needed improvement were the keel, rigid sail, and the rudders. To improve the keel, force diagrams were developed to determine the best angle and mounting location. With a mounting angle and position determined, the airfoil and fixturing methods could be selected, which assisted other forces acting upon the boat. Stiffening components were added to the wingsail to help cut down the wing's flexibility. This included using new materials for the wing and creating sturdier mounting point to affix the wing to the hull. Lastly, the rudders were redesigned for autonomous control for better trimming capabilities. Evaluating the implications and integration of better-tuned control surfaces serves to create a sailboat that could sail more effectively.

#### 5.2.1 Keel Design

The keel is the most important component to the stability of the boat; without a keel, a sailboat would turn over with any wind and not sail straight. Keels extend downward from the hull and act as a cantilever for the forces of the sail acting to roll the boat over. In order to achieve this, keels are made of heavy ballast, usually lead. By adding extra weight to the boat, one has to consider the benefits and costs with that decision. Weight is favorably eliminated on boats to decrease the amount of effort needed to propel the boat. Because the keel needs to resist the forces of the sail and be as light as possible, the farther the center of mass is placed from the waterline of the boat, the lower the mass of the keel will be. This requires anchoring a heavy chunk of lead far from the boat with as light of a connection as possible. To accomplish this task, this year's team worked to create a keel that is balanced between the front and the back of the boat, and able to provide an acceptable righting moment to the hull while maintaining a lightweight structure.

#### Keel Mass

The mass of the keel is derived from the equations for calculating the righting moment of a sail of certain force and the moment supplied by the weight of the sail. The equation for the righting moment



is represented by the function:

$$Mr = -Rh[R\sin(\lambda)\cos(\lambda)(1+\cos(\theta))+Fs(\cos(\lambda)2-\sin(\lambda)2\cos(\Theta))] + RbFb\sin(\Theta)$$

Where  $Mr$  is the righting moment,  $Rh$  is the height distance between the center of mass of the keel and the center of effort of the keel,  $R$  is the hydrodynamic drag force,  $\lambda$  is the angle between the boat and point of sail,  $\theta$  is the hydrodynamic drag angle,  $Fs$  is the force of the sail,  $\Theta$  is the heel angle,  $Rb$  is the height distance between the center of buoyancy of the boat and the center of gravity of the keel, and  $Fb$  is the buoyancy force. Each of these values is calculated from other primary values seen in the appendix.

The mass of the keel is directly related to the righting moment and the torque supplied by the mass of the sail. Solving these equations out led the team to find a required torque of 1000 in.-lbs. Using this torque, lengths and masses were selected to work with. Once the length was chosen to be 40 inches, the necessary ballast was calculated to be 25 lbs.

## Keel Shape

To allow the keel to move through the water, selection of appropriate airfoils must take place. Using a symmetrical airfoil is important to a sailboat as the lift that is generated by a keel must occur on both sides of the boat. This is unlike an airplane, which mostly needs to generate vertical lift. The airfoil shape of NACA 0012 was chosen for its symmetrical shape and its efficiency at the speeds the boat needs to operate at in the water. This is shown below in Figure 7. Given that the angle of attack of the keel relative to the point of sail of the boat is between 3 and 5 degrees normally, the NACA profile needs to have a low coefficient of drag at these angles.

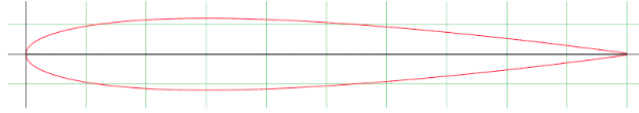


Figure 7: NACA 0012 Airfoil used for keel shape

To accommodate the lead mass that was fixtured to the bottom of the keel, a shape that is both hydrodynamically efficient and has a large volume needed to be selected. Given that the NACA 0012 profile has such a small thickness, a larger airfoil with similar drag characteristics at the expected angle of attack needed to be chosen. The keel bulb shape was eventually decided to be a revolved NACA 63018 profile affixed to the bottom of the keel made of lead because of its high density and small area.

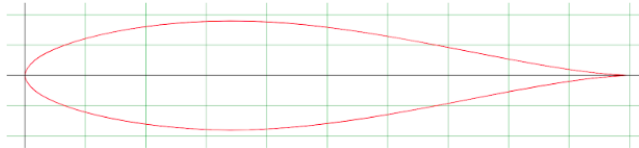


Figure 8: NACA 63018 Airfoil used for keel bulb

## Keel Stiffness

The main issue with suspending a heavy mass cantilevered from a boat is the matter of reducing the weight of the suspension method. In both the keel and the rigid wingsail, this proved to be an issue. Naturally, the issue of reducing stress on cantilever masses manifests itself in many ways. People use objects that are cantilevered every day, from bridges to buildings, consumer products to sports products. Most items are designed to use as little material as possible and provide the strongest stiffness. The keel is essentially a lead mass isolated away from the hull. To support this mass, a lightweight, yet stiff, structure must transfer the loads effectively into the hull. One solution that came to mind was the use of I-beams. I-beams utilize the moments of inertia around a neutral axis to resist deflection and are lighter than a full beam of similar properties. The team experimented with lamination of a birch wood board and carbon fiber sides. These composites acted as an I-beam because the inside sandwich piece of wood acts as a skinnier piece of carbon fiber in the deflection formula. This idea proved difficult to manufacture due to problems with adhering the carbon fiber to the wood. An alternative solution came from the world of sports; golf and hockey players put excessive amounts of force into the equipment that they use. Golf players swing clubs that are designed to bend, but take large loads. Hockey sticks similarly take heavy strokes and pressure from the players themselves. The keel needed to be strong and flex with the rotation of the boat without shearing at the interaction point with the hull. By using products like the hockey stick and golf club that are made to resist extreme forces, the keel bulb could be cantilevered from the bottom of the boat. As previously mentioned, the keel takes around 1000 in-lbs of torque and the components must be designed to be able to take these loads.

## Integration

The keel must be removed from the boat for transportation and to reduce the stresses on the hull when out of the water. This necessitated a mount in the boat as well an insert that attached to the keel to assist with mounting inside the boat. For the mount in the boat, the idea was to rout a slot in the bottom of the hull using a form to guide the router. This slot is oversized to allow for a new glassed, internal structure to fit inside of the hull to add additional strength to the fixture. To allow for the keel to be anchored in the new slot, a hull insert was designed and tested to observe the effects of stress on the part. A mounting screw was inserted through the deck of the hull and attached to the hull insert to secure the hull insert into the boat. To reduce the stress between the hockey stick and the hull insert, rounded fillets spread the stress out over a larger distance.



Figure 9: Keel hull insert design

This hull insert takes an incredible moment on its features. To help manage this moment, the hull has a little bit of flexure to it to allow the hull insert to spread the force out through some of its features. To resist the twisting forces commonly experienced by suspending a heavy mass on the end of a rod, the hull insert extends down the length of the hull allowing for that twisting moments to be applied to the hull and not about the main structure that is supporting the keel. Using Solidworks FEA, a simulation was created to map the forces on the main joint.

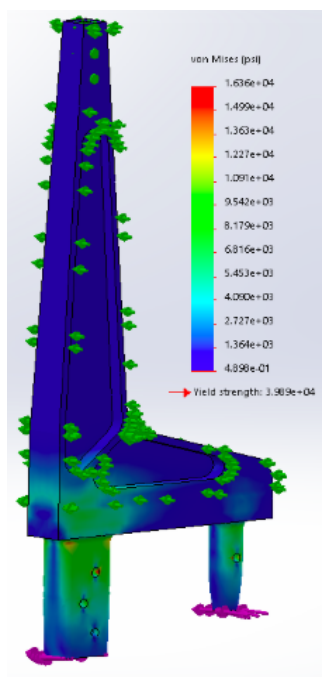


Figure 10: Keel hull insert FEA

Shown in the above figure, the forces experienced by the hull insert are roughly 30% of the yield strength of aluminum. This allows for the keel to heel at the angle it needs to keep the boat sailing well.

### 5.2.2 Rigid Sail Design

A rigid wing sail acts as a plane wing that sticks out of the top of the sailboat. In addition to providing driving force to the boat, a rigid wing sail contains its controls, controlling a trim tab that provides lift to the boat in most sailing conditions. The original rigid wing sail was designed by a 2017 MQP team of aerospace students. This wingsail was too stiff and heavy, as it was intended to be used on a different boat and oversized for the need. The weight made the sail close to impossible to use and the wingsail was too bulky to be effective. In 2018, a group of students in the WPI Robotics Club created a lighter-weight sail and reduced the overall size. In doing so, the Robotics Club introduced new techniques into the sail for reduction of weight, but overlooked some of the mechanical dependability that the sail needs. By creating a lightweight and stiffer sail, the problems associated with flexibility

and weight will be solved. In addition to the problems with stiffness, the sail does not protect itself from capsizing, thus destroying the electronics components inside for controlling the sail angle.

## Stiffening

The process of stiffening the sail came from researching lightweight structures such as radio towers and the methods that those structures use to keep from failing. The primary design originated out of the idea of using ropes with little elasticity secured and tightened to the top and bottom of the sail. The first idea was to use an x pattern of criss-crossing ropes from a point closest to the mast to the outside of the spars, where at both connections the ropes pass through carbon fiber ducts. This idea failed for many reasons. The stiffness of the ropes is dependent on the stiffness of the anchoring at the top and bottom of the sail on the edges of the spars, which puts a great amount of torque on the spar. Additionally, this design allows the spar to act as a pulley at the point where it passes through the duct closest to the mast. This pulls the duct outward to the edge of the spar instead of focusing the forces around the mast. To remedy these design flaws, a reduction of the number of ropes is used, and anchoring the ropes at both ends of the mast closest to the center allows them to pass only through the outside ducts on the spars. The rope then passes over the mast and through the next spar outside edge on the opposite side of the mast. This system allows for the rope on both sides to always be in tension. Additionally, when the mast bends, the tensioned side will always be in greater tension than the compression side. This system works for the part of the sail that is covered in a skin to reduce the stress. At the connection with the boat a separate system is utilized. To remedy this, an insert was designed to fit inside the shaft of the sail and reduce the bend on the carbon fiber mast. This part was machined out of aluminum to the max shear of the carbon fiber and will distribute the load of boat on the mast. This part's curve was developed through the equation for displacement of a cantilever beam with an applied moment on one end.

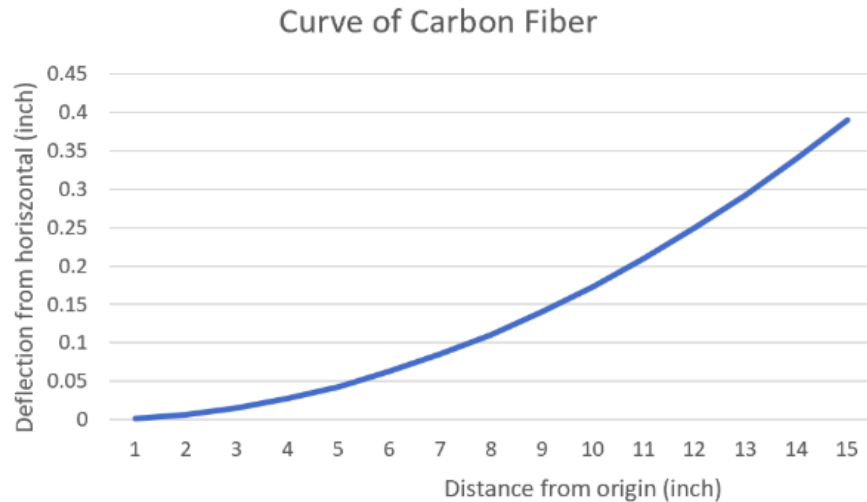


Figure 11: Deflection due to carbon fiber

When the mast bends, it contacts the aluminum profile and transfers some of its load on to the part,

reducing the overall stress. It is similar to adding an internal chamfer to a part.

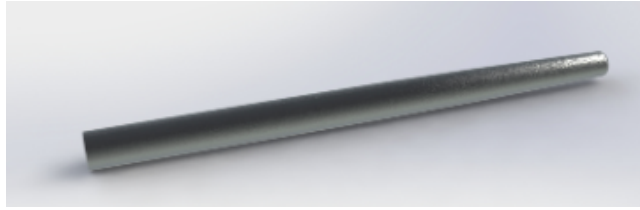


Figure 12: Mast insert for rigid sail

When testing this internal mast insert, a problem was discovered. The upper section of the sail was rigid as it was tied together with the dyneema cable system and the lower section of the sail also was rigid, but the connection between the two of these elements was a weak point. To remedy this problem a stiffer section of the mast was created where the dyneema cables could be anchored to. This stiffer section of the mast consists of a fiberglass tube with acetyl caps filled with expanding polyurethane foam around the outside of the previous mast. This lower section couples with the mast receptacle in the hull and provides a strong connection for the forces of the sail to be transferred onto the boat. In creating this coupled joint with the hull receptacle an increase in stiffness was experienced by 700%. Additionally with the integration of this stiffer section of the sail, the dyneema cables were now accessible outside of the boat and could be tightened in the field.

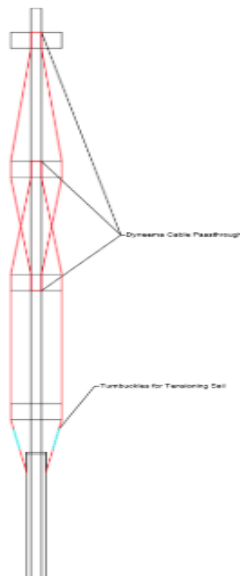


Figure 13: Dyneema cable trusses

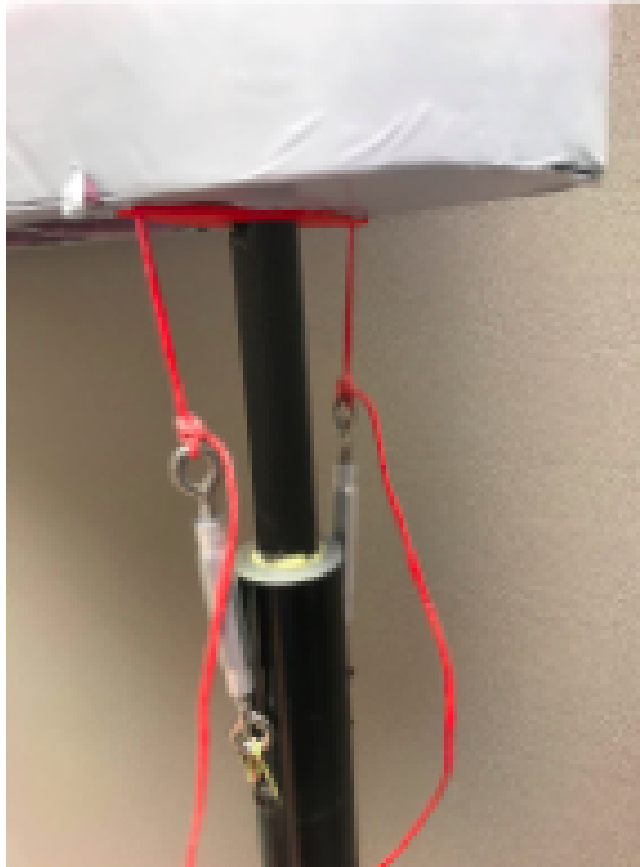


Figure 14: Stiffened section of rigid sail mast

## Waterproofing

The previous sail did not provide any waterproofing for the electronics contained within. This year, a lightweight foam structure with an included waterproof container makes up the electronics housing on the rigid sail. To turn on and off the trim tab, a hall-effect magnetic switch was used and a USB plug to the rear of the enclosure was used for recharging the internal battery. To manufacture the new system, layers of polyurethane foam were hot wire cut to different profiles and then glued together. 3D printed inserts were added to the locations of high stress such as the connection between the trim tab and the sail.

### 5.3 Electrical Systems

For the electrical and software improvements, the first goal was to get the boat on the water to test how well the boat operates under remote control via the ethernet bridges. However, the team encountered many problems when trying to test, such as connection issues during RC, poor planning for lake availability, or lack of batteries. Another issue that was mentioned in documentation from the 2017-2018 team was that the compass heading would sometimes be inconsistent, differing by almost 20

degrees at times. However, the Airmar220WX Owner's Guide and Installation Instructions [7] noted that the Airmar must be mounted at least three feet from electronic devices to avoid any interference with the magnetic compass, and free from any obstructions that might interfere with wind flow. This is mentioned in the recommendations section for the 2019-2020 team.

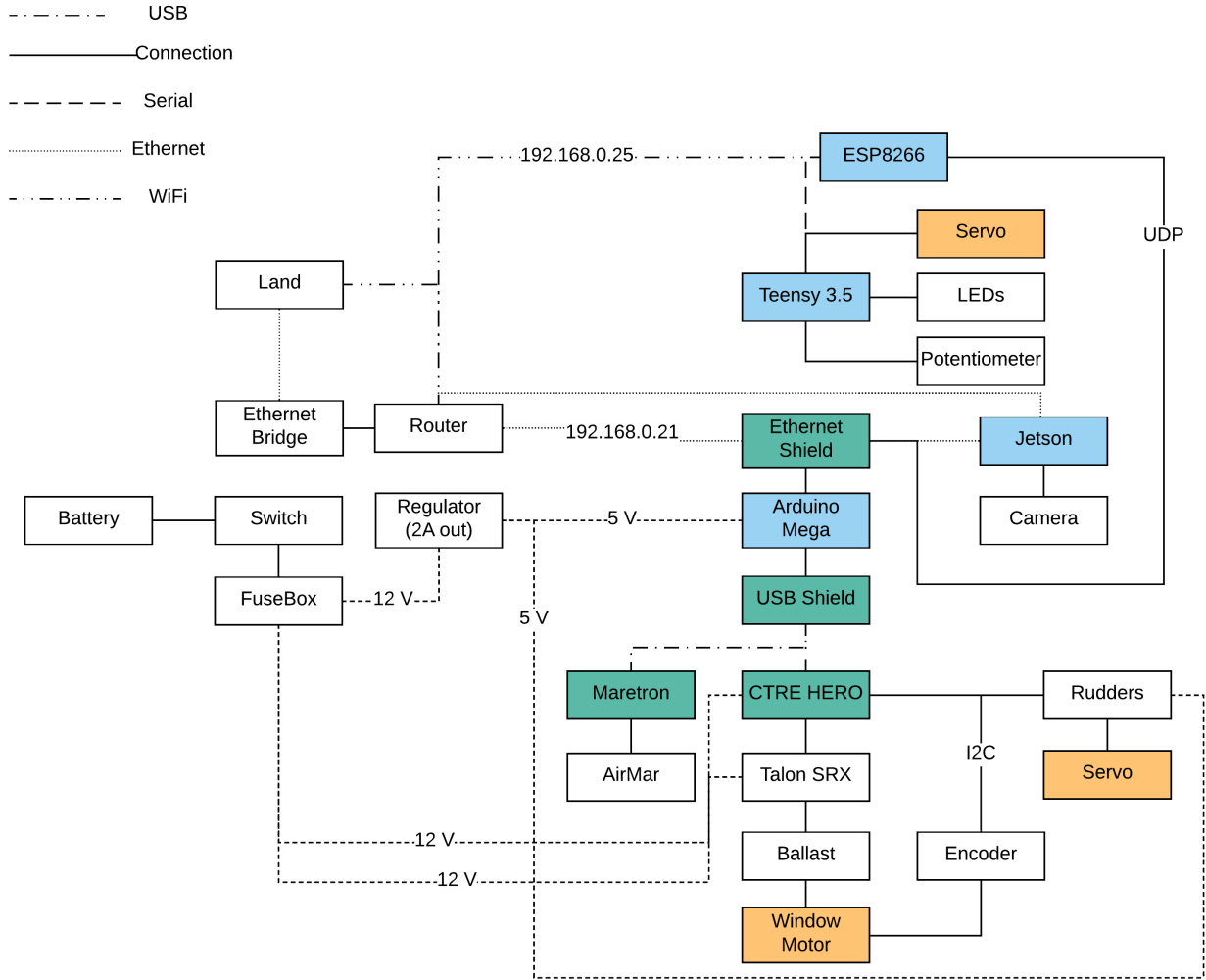


Figure 15: Hardware Schematic

The team ran into some problems with the winch, as it only worked intermittently in past years. One problem was the winch would only run in the clockwise direction, and never in the counterclockwise direction. The team tried to figure out a solution to this problem, but they have tried re-soldering the board, which did not fix the problem. After testing both the motor and the encoder, the team determined that it was the motor controller that was the issue. After testing the motor controller by running the winch, it was determined that it was a signal problem between the motor controller and

the NMEA bus. This was determined because, while the ballast motor controller lit up either a red or green light depending on the direction the ballast was being run in, the winch motor controller only ever lit up a red light for the clockwise direction. It was suggested by last year's team to power cycle the boat, but after doing so, the winch still did not work. After all of these issues, the team decided to redesign the entire system, even replacing the BeagleBone Black with an ArduinoMega connected to an Ethernet and USB shield for ethernet communication and communication with other systems.

### 5.3.1 Arduino MEGA with Ethernet and USB Shield

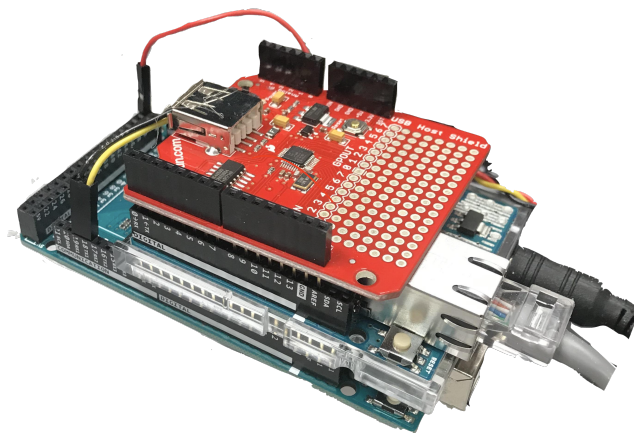


Figure 16: Arduino with shields setup

After multiple issues regarding old code and the BeagleBone Black, it was decided that a full hardware redesign was required. This would allow for the project to continue moving forward instead of being stalled like it had been for the previous months. Thus began the process of trying to decide between different boards that would be easier to interface with. The team considered three boards: an Arduino MEGA with multiple shields, a Teensy 3.5 or 3.6, and a Nucleo-F746ZG. There were pros and cons associated with each, but it finally came down to what could be implemented the quickest with the time constraint of about 4 weeks at hand.

With the time as a key factor, the Arduino MEGA with multiple shields was chosen. This was due to the team's previous knowledge of working with the Arduino IDE, and the public resources available. The issue with using the Teensy 3.5 or 3.6 was that there were not as many external resources, and the team was worried about how much the Teensy would actually be able to communicate with the rest of the boat. The reasoning behind not selecting the Nucleo was having to figure out how to integrate everything with it in a quick amount of time.

Once the MEGA was selected, the shields would then have to be selected for integration with the other parts of the system. One of these shields had to be an ethernet shield so that the Arduino could host a webserver and then connect to the router via an ethernet cord. The second shield that was



selected was a CAN bus shield. The purpose of the CAN bus shield was so that the TalonSRX motor controller could be used to control the movable ballast. However, upon further testing, it was realized the the CAN bus shield would not allow for the TalonSRX to initialize, as it could only operate with a RoboRIO or a CTRE HERO board.

After reading through the TalonSRX's user manual [8], it was thought that all would have to happen was initialize the TalonSRX's ID, and then it would be able to be used by any CAN device. The team borrowed a RoboRIO from another group on campus and then proceeded to set the ID of the TalonSRX to "0". With this completed, the team then brought the Talon back to the lab to work on developing the handshake between the CAN shield and motor controller itself. Using a logic analyzer, the CAN message sequence was found, and was programmed to send the message from the Arduino, through the CAN bus shield, and to the Talon. However, the handshake didn't work. This left the team with the only option being to purchase a CTRE HERO off the shelf, removing the need to keep using the CAN bus shield in the system.

The now-second shield that would have to be used was the USB shield. This was originally thought to run a few things: the HERO, the Maretron, and the controller for the system. A multi-USB port was purchased to run all of these things. However, after some testing, it was quickly discovered that this would not work for the communication for all three systems. This was later solved with more HERO testing, which will be later explained in section 5.3.3.

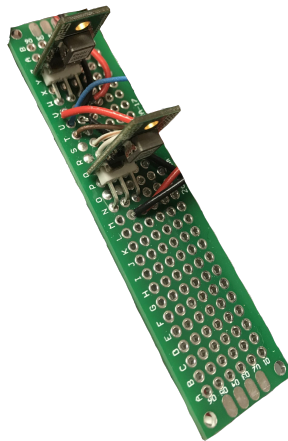


Figure 17: 12V to 5V 2A converters for Arduino setup and rudder servo

The final hardware portion of the Arduino that had to be fixed came about when starting to work on the code. It was realized that the Arduino would only fully function when plugged into a laptop to be programmed. Once it was unplugged, the Arduino had problems keeping the web server running and keeping up any communication with the rigid wing's Teensy. It was found that, once both of the shields were plugged in and the rudder was being powered by the Arduino, it pulled too much current for full function. To alleviate this issue, two 12V to 5V 2A converters were inserted: one to operate

the Arduino, and the other to operate the power and ground for the rudder. Signal for the rudder is controlled by the HERO board.

### 5.3.2 Teensy 3.5 with ESP8266WiFi Chip

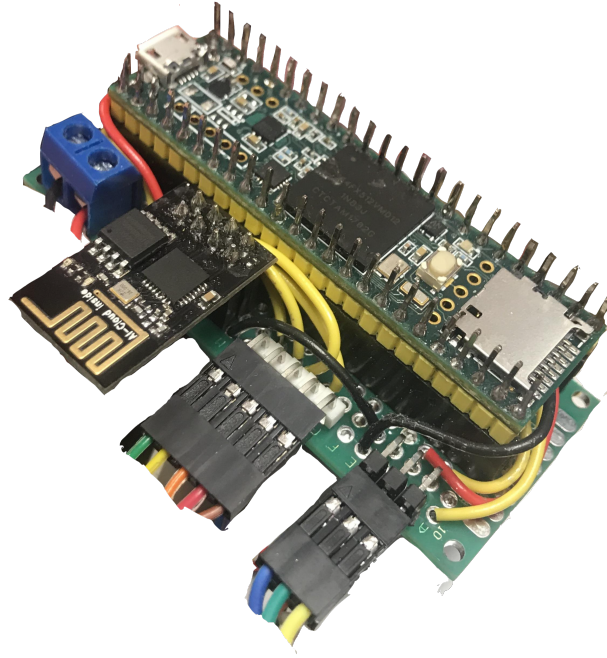


Figure 18: Teensy with ESP8266

The Teensy 3.5 works as the main controller for the rigid wing. Most of this system stayed the same from the previous year, with just some minor changes. A lot of the changes consisted of soldering onto new prototyping boards for cleaner connections and easier visuals while programming. The biggest change, however, had to do with the programming of the components. This consisted of programming different LED sequences, communication between the ESP and the Teensy, and then communication between the Teensy and the Arduino.

For the LED sequences, all that had to be done was initialize the LED pins, and then program to be high at depending upon the different states. The sequences can be seen in the table below.

LED sequence	Meaning
Only Red	Maximum Lift
Only Yellow	Maximum Drag
Red and Yellow	Zero Lift
Red, Yellow, White, White	Lost Communication

Table 1: Trim tab LED Sequences

To develop lift, the rigid sail needs to be angled at either maximum lift, maximum drag, or no lift. To find the proper angles for the sail to be located in relation to the apparent wind, the airfoil of the wing is analyzed. To find the angles of attack that generate the most lift and drag a graph of  $Cl/Cd$  vs Alpha is observed.  $Cl$  stands for the coefficient of lift,  $Cd$  stands for coefficient of drag and Alpha is the angle of attack. A higher value of  $Cl/Cd$  signifies a greater overall lift and a lower value signifies a greater overall drag. Shown in the graph below are the values for the airfoil shape of the wingsail.

Once the new Serial port was created, multiple commands could be written in order to start communicating between the Teensy and the Arduino. This could be done using the “ESP8266 AT Instruction Set” [10], and testing using sample code online. This allowed to make sure the communication was functional. This simple loop checked for which serial port was available, and then would wait for a user to type in an input into the serial monitor’s text window. Using this, the sequence of AT commands could be selected in order to go through the process of sending UDP messages. It did this by connecting to the boat’s Wifi, starting the UDP connection, sending a message using that connection, then closing the connection when finished. This can be better demonstrated in the table below, taken from the AT Instruction Set.

AT Command	Response	What it does
AT	OK	Test AT Startup
AT+CWMODE=3	OK	Queries the current WiFi mode, and sets it to SoftAP and Station Mode
AT+CWJAP="sailbot",... "Passphrase123"	WIFI DISCONNECT WIFI CONNECTED WIFI GOT IP OK	Connects to the AP with the SSID of "sailbot" with the password "Passphrase123"
AT+CIPMUX=1	OK	Enables 1 connection
AT+CIPSTART=1,"UDP",... "192.168.0.21",80,... 1112,0	OK 1, CONNECT	Opens a UDP connection on the one opened connection at the IP address of 192.168.0.21. This is open on port 80.
AT+CIPSEND=1,6	OK	Sends a message of 6 bytes on the connection
AT+CIPCLOSE=1	OK	Closes the 1 connection

Table 2: AT commands used in code

With the AT commands initialized, the Teensy then waits to receive the message from the Arduino containing the number of the button that was pressed, which correlates to different rigid wing states. The response will be seen as "+IPD,1,1:X", where "X" is the value of the button press. For the maximum drag and maximum lift states, the trim tab rotates to be in the same direction as the wind sensor. This is calculated by checking the encoder values being returned by the wind sensor. If the sensor is returning a value less than 730, then the wind is blowing from the port side. Otherwise, the wind is blowing from the starboard side.

To determine the angle that the trim tab needs to move to be in either maximum lift or maximum drag, the team used the Joukovsky 0015 airfoil (the shape of the rigid sail) and a Reynolds number of 24, 141 calculated from the chord width of the sail, the expected wind, and the density of the air [11]. This produced the following values:

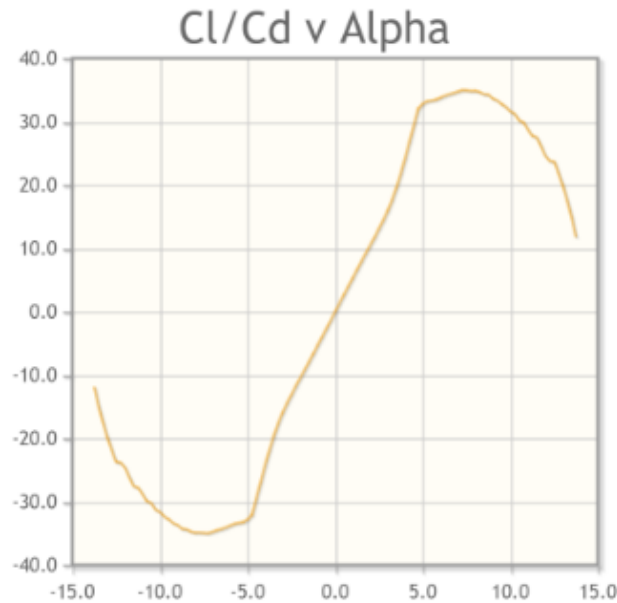


Figure 19: Cl/Cd v Alpha for Rigid Sail

At an angle of 58.5 degrees, the wingsail generates the greatest amount of overall lift. This is the condition of maximum lift. At an angle greater than 15 degrees, the wingsail generates the least amount of overall lift and the greatest amount of drag. This condition is the maximum drag position. Lastly, at an angle 0 degrees, the rigid sail develops no lift and reduced drag.

The next thing to focus on was developing the communication between the ESP8266 wifi chip and the Teensy 3.5. The previous team looked to do this by creating TCP communication, which allows for streams of data to be exchanged between the board and the chip. One issue with this was that they set the Serial port incorrectly based upon where they had the ESP chip connected to the Teensy. They defined it as Serial4, which is supposed to be connected to pins 31 and 32 for RX and TX. However, the RX and TX were connected to pins 9 and 10, which are associated with Serial2. Instead of dealing with all of this, it was just defined as “ESPSerial” in the code, and associated with pins 9 and 10 [9]. This was done using the SoftwareSerial.h library in Arduino.

Lift type	Required angle (degrees)	Servo value rotating left	Servo value rotating right
Maximum Lift	58.5	87	113
Maximum Drag	15	72	128

Table 3: Trim tab angles

The servo values were found using trigonometry. The required angle had to be the angle between the

"0" position and the new position of the trim tab. With that, the team then had to find out what the range of servo values gave them the correct positions. This range was 55 to 150, which was found by just sending the values to the servo and seeing where the trim tab ended up. From there, using the tangent of the angle, the length of the shortest side between the "0" position and the aft-most point of the trim tab could be found. Using this length, the corresponding servo value could be found that would put the aft-most point of the trim tab in the correct position.

### 5.3.3 CTRE HERO Board

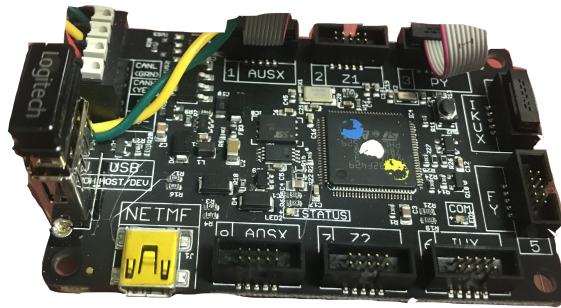


Figure 20: HERO Board

After purchasing the HERO board, integration with the full boat started to go quickly. By following the directions in the HERO User Manual [12], setup went fairly smoothly. The biggest things to note were knowing which ports were associated with which functions, and that you had to use the USB-A to A to initialize the board before being able to start programming using the MiniUSB cable. The HERO is programmed using C# with VisualStudio17 currently being the recommended IDE.

The first step in the process of getting the HERO to function with the components was to connect one set of the CAN cables (yellow and green) from Talon to the CAN ports on the board, which are color-coded. The other set of CAN cables had to be terminated with a 120 ohm resistor. For this, the team just ended up putting two 10 ohm resistors and one 100 ohm resistor in series between the two wires. Once the CAN cables were connected and the HERO was plugged in using the MiniUSB, the board was powered, thus powering the TalonSRX. When this was done correctly, the LEDs on the Talon would alternate flashing orange, notifying the team that the CAN bus was connected, but the robot was not initialized. Knowing this was the case, the next step was to start connecting the board in a way so that it was powered by the boat, not the computer programming it. A power and a ground port were located next to the CAN ports on the board, so the inserted leads were taken from those ports to a connector that would lead to the fuse box.

After powering the board and initializing the Talon, the next step was to test the Talon's functionality. The HERO's user manual comes with a lot of sample code snippets to help get started, so this process was quick. All that was required was to put that code sample into VisualStudio, plug in a Logitech controller, and then turn the whole system on. With this in place, the movable ballast could

move along its full range by pushing the left joystick on the controller forward or backward, which was associated with joystick axis 1.

Once the Talon was fully functional, the next step of the process was to determine how to communicate with the rudder. It was already being powered via one of the 2A converters (see section 5.3.1), so the last portion was to determine what signals to send it. After some thought and reading through the user manual, the best way to do this was to treat the rudder's servo as a PWM controller. To correctly plug this in, a Gadgeteer cable from CTRE was required. Page 10 of the HERO User Manual gives all of the pinouts depending on the required connection. The team decided to use Port 3, then connected Pin 9 to the signal pin of the servo. At first, the team tried to just connect using the female end of the Gadgeteer cable and connecting to the servo pin. However, when the whole system was moved to the boat before testing, this was starting to cause shorting issues.

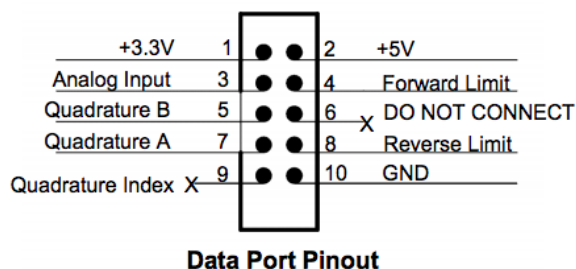


Figure 21: Pinout of Gadgeteer cable from Talon User Manual

To alleviate this, the team instead cut the female port off and just soldered connection from the desired wire that would go to the rudder. After the sample code was implemented, the rudder could be controlled by moving the right joystick left and right, which was associated with joystick axis 2.

After having full control of both the rudders and the movable ballast, it was determined that the next step would be to figure out the communication between the Arduino and the HERO. This would help to create the communication throughout the entire boat, from HERO to Arduino to Teensy. For this communication, the best option was to communicate through UART. To do this, Gadgeteer pins 4 and 5 on Port 1 had to be connected to pins 18 and 19 respectively on the Arduino. This coincided with Serial1 [13] on the MEGA, thus making the process a little easier when reading in data. The only data that was originally being sent was one way from the HERO to the Arduino containing the number associated with the button press of the Logitech controller. For autonomous implementation, information would also have to be sent from the Arduino to the HERO to give control commands to the ballast and the servo.

The last major factor in creating full functionality of the robot was to start receiving feedback from the movable ballast. The previous team had already developed a breakout board that utilized an AS5048A magnetic rotary encoder to determine the position of the ballast depending on its rotation. To use this, I2C communication had to be implemented between the HERO and the board. When

using I2C communication with the HERO, all of the ports create the same connection, so it did not matter which port the team selected to use. In this case, Port 4 was used, and pin 9 was connected to SCL, pin 8 to SDA, pin 1 to 3.3 V, and pin 10 to ground. After seeing how the I2C board was previously installed onto the board, it was determined that slave address 0x40 was the correct address for communication based upon the pads that were selected on the breakout board. The best resource that was used in determining how to write I2C for the HERO was found in a CTRE github page regarding using a sonar sensor [14].

#### 5.3.4 Maretron



Figure 22: Maretron unit

Using the Maretron on the boat was a big step for the team this year. While using old code earlier in the year, it was thought that maybe something had happened to the Airmar sensor, as all of its log files were returning straight zeroes. However, after downloading the Maretron software using the Maretron's User Manual, the team was able to hook up the system to a NMEA2000 bus with the Airmar, plug the Maretron into the computer with the software, and test the system outside. This began returning full values for everything that would be needed for the project, such as the GPS coordinates, wind direction, wind speed, roll, pitch, and yaw.

Once it was found that the Airmar was indeed working, the next step was to figure out how to read NMEA-0183 sentences. Using information found at [15], a simple parser was created that would look for the starting few characters, "\$-", and then save data according to the three character code that came after the start. The three character codes are listed accordingly:



Three character code	Information
GGA	GPS fixed data
GLL	Geographic position in lat/long
GSA	GPS and active satellites
GSV	Satellites in view
HDT	True heading
RMC	Recommended minimum navigation info

Table 4: NMEA-0183 Sentence Character Codes

These codes could be read from a serial stream that would go through the a USB-B cable from the Maretron to the USB shield on the Arduino, providing full information, such as GPS coordinates, roll, pitch, yaw, current heading, and wind direction.

## 5.4 Software System

By improving the remote control, the team can better the autonomous controls by mimicking its RC operation. For buoy detection, there will be on-water testing in different weather scenarios to check how well it picks up the color and size of buoys. After that is improved, this can then be applied in order to create a better navigation planning system for the boat to complete three of the courses. This year’s Sailbot team has learned from previous team’s software designs. They originally decided on a continuing the system developed by the 2017 Sailbot team. However, after wrestling with the code for months, it was decided to start from scratch after the electronics redesign.

### 5.4.1 Course Logic

When looking to develop the navigation strategies for the boat, the team first looked at the different challenges presented by the 2018 competition, as it was assumed that the challenges will be similar for the 2019 competition. The main challenges that the team are focusing on to base the navigation strategies on are the fleet race, endurance test, precision navigation, station keeping, and search.

#### Fleet Race

For the fleet race, teams are challenged to sail via remote control with the other boats from the starting line, around a buoy, and back. The biggest portion of this competition is having the remote control programming working well to make the boat easier to sail. Throughout the process of designing, however, the team has encountered many setbacks in improving upon this code. First, there was very little documentation to direct the team. This led to lots of research on the Basel website to determine how to correctly set up the run environment and install the 16.04 Ubuntu virtual machine.

After setting up the virtual machine, the team found that the joystick controller was not working. One of the problems is there are two modes to a Logitech Gamepad Controller: X and D. Mode X is for

XInput and D is for direct input. XInput means there is no customization and should work natively with every game. Direct input is used for customized controls, thus being the desired mode for the team; however, by switching between the two modes, they created two different filters on the virtual machine. This caused the joystick to be “confused” in the virtual machine and not run at all. Once they deleted the XInput filter, the joystick then was able to run in joystick testers, such as Jstest (a Ubuntu tester) and HTML5 Gamepad Tester (a web browser). Because both software developers were using a virtual machine on a MacOS, there are two joysticks automatically added and seen by the machine. This caused problems because the code reads from the joystick input file “/dev/input/js0”, but the joystick is actually sending inputs to “/dev/input/js2”. Once js0 was changed to js2 in the code, the RC control code started reading inputs from the joystick.

After the redesign, which is discussed in section 4.1.2, to start RC, the user only has to push the “start” button on the controller to turn on the controller, and use the left joystick to control the ballast and the right joystick to control the rudders.

## Endurance Test

The endurance test requires the team to create a program that will allow the boat to sail about a rectangular, 1 nautical mile long course for up to seven hours. After discussing as a group about how to approach this task, it was decided that the program would be codependent on navigation code as well as camera tracking of buoys. This year, the team developed an idea as to how to correct this with buoy detection strategies. As for the endurance test, the robot would need to sail along a path towards a buoy, and sail about the starboard side of the buoy before continuing on. The biggest issue regarding this was determining how best to round buoys. The process for this is shown in the following flow chart:

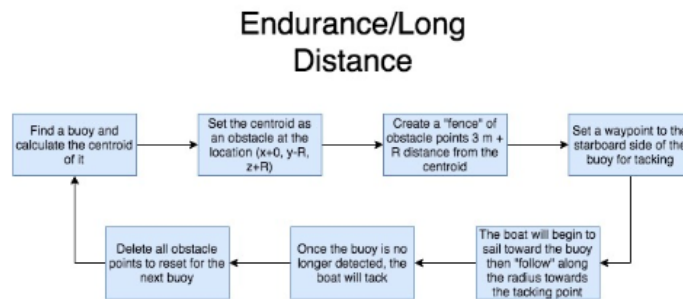


Figure 23: Code Process for Endurance/Long Distance Challenge

## Precision Navigation

For the precision navigation challenge, robots are tasked with navigating a triangular course and then sailing between two buoys, which act as the finish line. The plan was to have most of this code be reused from other challenges, including the buoy navigation and sailing along a path challenges. The main difference for this challenge was that a buoy count needed to be implemented so that the

robot knew when to round the base of the triangle. Once it had done so, it looked to find a gate to sail through to finish the race. This changed the logic slightly because the robot was to detect the starboard-most buoy, and find the midpoint between the two and try to sail between them. The process for this challenge is shown in the following flow chart:

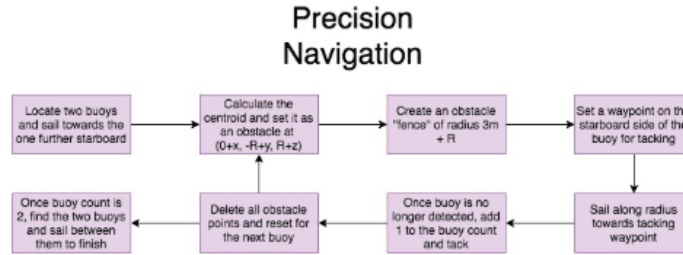


Figure 24: Code Process for Precision Navigation Challenge

## Station Keeping

The point of this challenge is to have the robot enter a 40 meter squared area, stay within it for five minutes, and then exit within thirty seconds of the end of the five minutes. With that in mind, the original plan of attack was to have the robot use the same gate navigation from precision navigation, go into the box, set up an “obstacle gate” about the edges of the box, sail within it for five minutes, and then use gate navigation when it hits five minutes on the onboard clock to exit. After discussion, the team realized this might not work. First, the boat can be remote controlled to enter the box for the first time. A second thing is that it would be easier for the boat to try and stay in one spot rather than having it sailing in circles. To fix the second issue, it was thought that it might be better to have the boat attempt to sail into irons to stop sailing once it entered the box. However, this also would not work due to the fact that it is extremely difficult for the sail to get out of irons to exit. For the final idea, the robot would enter the square and send the rigid wing a signal to create zero lift, making it unable to sail for the five minutes, then move the sail into a max lift position once the end of the five minutes hits. The process for this challenge is as follows:

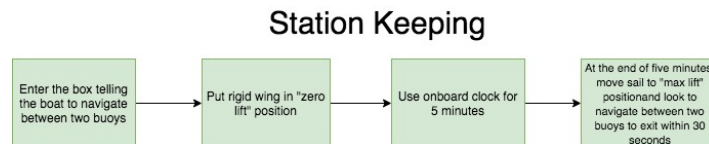


Figure 25: Code Process for Station Keeping

## Search

The goal of the search challenge is to have the robot find a randomly placed buoy within 100 meters of the start point during a five minute time window, tap the buoy, and then signal back to land that the buoy has been found. The original idea with this was to use the A\* search algorithm to look at

all the nodes, then returning the best path once the buoy has been found. One major issue with this was having to change the path limitations to account for wind direction and path width. The team will try to utilize the lane detection code to work on this for competition, but as for now, it is outside of the scope of the MQP itself.

#### 5.4.2 Reusable Code

One of the other major goals that the team was focusing on was creating code that can be reusable across the different activities. For this case, it was decided that the two major reusable functions should be the buoy rounding and sailing along a path.

#### Buoy Rounding

The idea behind the buoy rounding was that almost every challenge utilizes a buoy in some way, which led to different types of strategies. The original idea for this was to focus solely on using the camera to measure where the boat was in reference to the buoy. However, after discussing with advisors, it was found that the camera would only be able to detect the buoy within 10 meters. Due to this, a new plan of setting a waypoint in front of the buoy was decided upon. It was thought that this would allow the boat to get within ten meters of the buoy and then sail about it, but that was not completely correct. One of the major flaws in that idea was that the team was treating the boat as though it were a land vehicle or a motorized boat. Even if the boat was able to make it straight to the waypoint in front of the buoy, it would have a really hard time approaching the waypoint required to sail about the buoy due to having to follow either a straight line or a curved path. After this, it was determined it might be better to set the waypoint to the starboard side of the buoy so that the boat could navigate along a further distance rather than try to make a short turn to get around the buoy. This can be shown below:

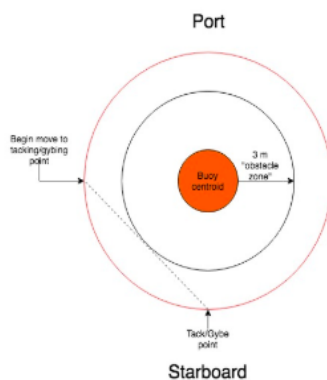


Figure 26: Rounding a buoy diagram

#### Sailing along a path

Sailing along a path is the most important piece because it is what allows the robot to sail from point A to B. This is one of the parts that has taken some of the most thinking due to how much

information the robot needs to receive before it can operate. The first idea was to focus on avoiding objects and shore detection. To solve this problem, solutions were researched based on autonomous movement along rivers, open water, and on non-sailboats. There were three main solutions that were determined, which are outlined below:

	Pros	Cons
360 Degree Camera Mount	<ul style="list-style-type: none"> <li>- 360 degree view of what's around the robot</li> <li>- Just reuse the camera detection code to determine the shoreline</li> <li>- Based on similarities, not absolutes</li> </ul>	<ul style="list-style-type: none"> <li>- The sail could block camera views depending on location</li> <li>- Boat shape isn't constant even with wing sail</li> <li>- Might have to use two different cameras</li> </ul>
IMU/GPS/digital compass Integration	<ul style="list-style-type: none"> <li>- Already use some of the sensors currently on the boat</li> <li>- Calculation consist mostly of linear algebra</li> </ul>	<ul style="list-style-type: none"> <li>- Have to manually set waypoints</li> <li>- Might be too much onboard processing with what is already being done</li> </ul>
GPS "lanes"	<ul style="list-style-type: none"> <li>- Based upon GPS location at the beginning of each turn.</li> <li>- Can change width of lane by determining how many degrees wide the lane should be</li> </ul>	<ul style="list-style-type: none"> <li>- If the boat doesn't reset when it rounds a buoy, the "lanes" will be wrong</li> <li>- Not much information on this</li> </ul>

Table 5: Pros and Cons of Lane Sailing Ideas

After going through this pros-and-cons table, it seemed the best option would be to create a 360 degree camera mount that can be mounted at the top of the mast. The camera setup was to be a CCD camera mounted in the bottom of a pneumatic tube. At the top would be a convex lens that would reflect the camera's direction to get a 360 view. However, for the purpose of this project, it was slightly outside of the scope, as the environment would be mostly controlled, and it could be too heavy on the top of the mast. To alleviate this issue, the 360 camera idea was dropped. The new idea would be to check where the latitude and longitude coordinates of the shore or docks and then create an obstacle along a parallel latitude and longitude that would not allow for the boat to sail across.

Once this was determined, there were a series of heuristics that had to be decided upon in order to start the process of developing the code. These include: wind direction from the Airmar, a lane to sail through based on latitude and longitude, and the distance to waypoint. Using these, the team will begin the process of integrating the code with last year's code in order to be able to start boat simulations.

## Lane Detection

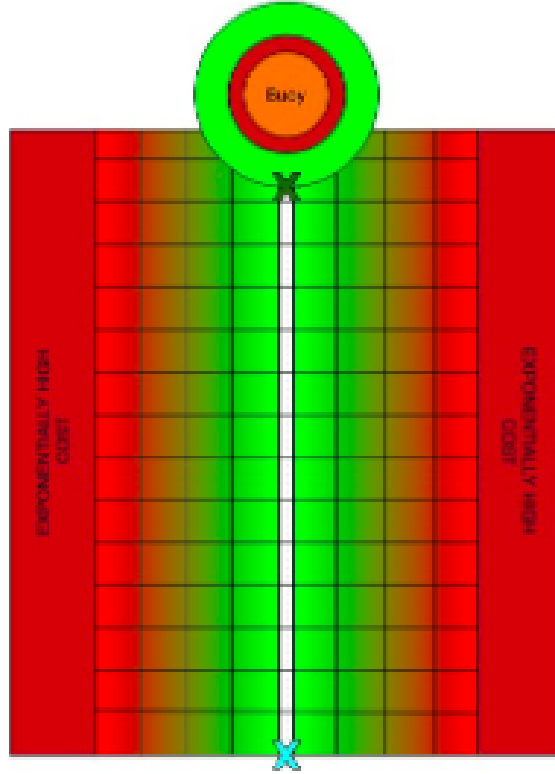


Figure 27: Lane cost layout

When looking to create reusable code as mentioned in Section 4, one of the major things that the team focused on was the idea of “lane” detection. With this idea, the team would be able to create “lanes” between buoys on each section of a course. The robot would then be able to navigate freely using a combined heuristic to reach the goal destination. The goal destination would be placed at the maximum distance at which the camera can locate a buoy. This would allow for the camera detection to take over to do the buoy rounding, as mentioned in section 4. The starting position is set when the boat has finished a tack or gybe or starting a challenge. From there, the biggest challenge is to determine the lane width. To determine the width, the boat would take in its GPS coordinates and then convert them to their local Cartesian coordinates, measuring everything in meters. Using the NMEA-0183 standard, the latitude and longitude coordinates are returned in degree and decimal minutes (DDM), which is equivalent to XX (Degree) XX.XXX’. In order to correctly convert this into the Cartesian coordinates in meters, the DDM coordinates need to be converted into Decimal Degrees (DD), which is equivalent to XXXX.XXX. This can then be converted using the knowledge that each degree latitude is equivalent to 111000 meters per degree, and each degree longitude is equivalent to the cosine of the latitude in DD multiplied by the length of degrees in miles at the equator, which is 60 nautical miles. The calculations used to do this can be seen below:

$\text{latitude.d} = \text{Latitude minutes} / 60$   
 $\text{latitude.DD} = \text{Latitude degrees} + \text{latitude.d}$   
 $\text{Latitude in meters} = \text{latitude.DD} / 111000$   
 $\text{Longitude in meters} = (\cos(\text{latitude.DD})(60)) / 111000$

Due to the length of the boat being two meters, it was decided that a safe distance lane width would be about twelve meters, with six meters on either side of the starting position. The next step would be to set up a grid consisting of 1x1 meter squares that the boat can perform A\* through. Anything outside of the twelve meter lane would be considered to have a highly exponential cost, and the cost increases radially from the straight line path as well. This is indicated in Figure XX, as the greener the square is, the lower the cost.

The other issue that arises with using the lane method alone is the fact that the boat is not self-propelled. With this being the case, another set of heuristics had to be taken into account to correctly plan the course for the boat. A paper written by C. Petres, M.-A. Romero-Ramirez, and F. Plumet also researched this topic [16], revealing the figure below.

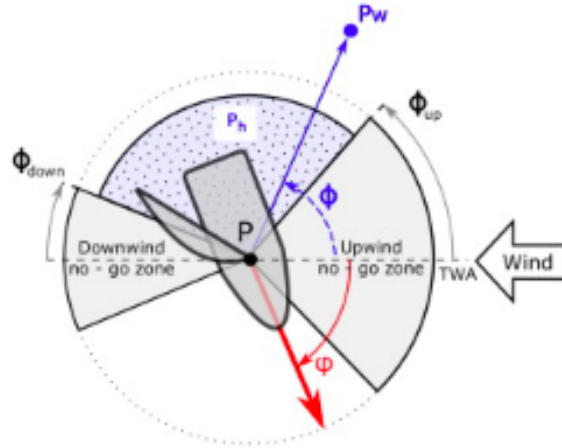


Figure 28: Wind heuristic

This accounts for the different points of sail for the boat, therefore making sailing upwind a high cost move, as well as sailing along the downwind line. Another position it takes into account is the current heading of the boat. Although a possible position might be for the boat to turn to the position behind itself and sail, that will cost higher, as that maneuver will take more time. Therefore, the least costly maneuver will be if the boat can continue upon its current heading.

Overall, the combination of these two methods will produce the following heuristic to solve for  $h(n)$  for lane navigation planning using the equation  $f(n) = g(n) + h(n)$ , where  $f(n)$  is the estimated total cost

through  $n$  to the goal,  $g(n)$  is the cost so far to reach the node, and  $h(n)$  is the estimated cost from  $n$  to the goal. To solve for  $h(n)$ , the following equations are used, partially using heuristics developed by processes in [16]:

- For wind direction:

- If upwind:

$$Pu = \begin{cases} dist(Pw, P), & \text{if } 0 < |\phi| < \phi_{upwind}. \\ 0, & \text{elsewhere.} \end{cases} \quad (1)$$

- If downwind:

$$Pd = \begin{cases} dist(Pw, P), & \text{if } 0 < |\phi - \pi| < \phi_{downwind}. \\ 0, & \text{elsewhere.} \end{cases} \quad (2)$$

- For lane width with “ $x$ ” being  $x$  value of the start location, and having 6 meters on either side:

$$Multiplier = \begin{cases} 1, & \text{if } x - 1.5 < x < x + 1.5. \\ 10, & \text{if } x - 3 < x < x - 1.5 \text{ or } x + 1.5 < x < x + 3. \\ 50, & \text{if } x - 4.5 < x < x - 3 \text{ or } x + 3 < x < x + 4.5. \\ 100, & \text{if } x - 6 < x < x - 4.5 \text{ or } x + 4.5 < x < x + 6. \\ 1000, & \text{if } x < x - 6 \text{ or } x > x + 6. \end{cases} \quad (3)$$

- Where  $Pl$  is the value of “ $x$ ” multiplied by the multiplier

-  $Pdist$  is the straight line distance between the current location and the goal

Therefore,  $h(n) = Pu + Pd + Pl + Pdist$



## 6 Preliminary Calculations

In order to understand the dynamics of the boat, modeling must be utilized. This modeling will inform design choices such as centers of effort and centers of mass for different components of the boat. To better balance the boat in multiple scenarios, a MathCad script was developed to analyze the moments in the two axis of roll and pitch.

By utilizing the dynamic ability to change sail angle and pitch arbitrarily, the team was able to see how different parts of the boat responded to the changes in environment. This also aided in developing a design that took a multi-axis approach to solving for conditional requirements. These calculations can be found in Appendix 8.2.

## 7 Future Recommendations

### 7.1 Mechanical Recommendations

The mechanical systems in the boat always benefit from being lighter, as the boat decreases in weight, it increases in speed. Some of the areas of the boat where weight can be reduced are in the electronics enclosures, the balancing of the rigid sail and the sensor mast that holds the Airmar.

An additional mechanical improvement to boat is waterproofness. The cable glands that protect electronics and their enclosures are not designed for water submersion, hopefully this never happens in the boat. Additionally, the connections through the hull of the boat are not as waterproof as they can be, redesigning the deck mounted joints would help to improve this risk of failure on the boat.

The moveable ballast is driven by a powerful motor but sometimes over rotates and locks itself in a damaging position where it can not turn back the other way under its own power. A system that integrates a clutch that limits the movement of the arm back and forth would help to remedy this problem.

### 7.2 Hardware Recommendations

#### Battery Change

The current battery used in the boat is a lead acid battery. These batteries are inexpensive for the capacities they can achieve but compromise in weight and maximum current draw. Currently the peak draw from the boat while motors are running exceeds this current rating, dramatically reducing the lifetime of the battery. This also limits the maximum sailing time that the boat is capable of.

Better solutions can be found using lithium-polymer or lithium ion battery technologies. Both of these battery types support a much higher energy density for both their size and weight. This would allow the boat to use a much smaller and lighter battery with greater sailing capabilities. These batteries can also provide much more current than lead acid batteries without damaging themselves.

This improved current capability would mean that the boat would have more power available during operation and that batteries would survive many more recharge cycles.

#### Airmar Location

A second hardware recommendation would be to try and move the Airmar to a different position on the boat. As of right now, the Airmar is positioned at the back of the boat, which is about half a meter above the electronics housed in the hull. After reading through the user manual, the team thought it might be better to move the Airmar to a position that is at least a meter from the back of the boat. This should help with any inaccuracies within data readings. This was meant to be achieved by the team this year, but ended up being put on the back burner after the hardware redesign.

### **7.3 Software Recommendations**

One major recommendation for the software portion of the project would be to try and move into working with a more robust programming language. While programming using the Arduino IDE benefitted the team in the short time line allotted, using a language such as C or C++ might help with some other issues. This could definitely come into play as further development is pursued on the current system.

## 8 Appendices

### 8.1 References

- [1] Conger, Cristen. “How Sailboats Work.” HowStuffWorks, HowStuffWorks, 11 Mar. 2008, [adventure.howstuffworks.com/outdoor-activities/water-sports/sailboat1.htm](http://adventure.howstuffworks.com/outdoor-activities/water-sports/sailboat1.htm).
- [2] Heard, John. “Club Sailing Rules - Springfield Lakes Maritime Modellers.” Google Sites, 2017, [sites.google.com/site/springfieldlakesmmodellers/comps-rules](http://sites.google.com/site/springfieldlakesmmodellers/comps-rules)
- [3] International Robotic Sailing Competition. 2018 URL: <http://sailbot.org>
- [4] “my2fish: a Blog about Sunfish Sailing.” my2fish a Blog about Sunfish Sailing, 2008, [my2fish.wordpress.com/tag/points-of-sail-diagram/](http://my2fish.wordpress.com/tag/points-of-sail-diagram/).
- [5] “Lake Quinsigamond, Worcester Climate History.” Lake Quinsigamond, Worcester, Climate Profile, Weather2, 2017, [www.myweather2.com/Fishing/United-States-Of-America/Massachusetts/Lake-Quinsigamond-Worcester/climate-profile.aspx?month=6](http://www.myweather2.com/Fishing/United-States-Of-America/Massachusetts/Lake-Quinsigamond-Worcester/climate-profile.aspx?month=6).
- [6] Worcester, MA History. (n.d.). Retrieved from <https://www.wunderground.com/history/weekly/us/ma/worcester/KORH/date/2018-6-4>
- [7] Airmar Technology Corporation. (2019). Ultrasonic: Smart Sensor WeatherStation Instrument Owner’s Guide & Installation Instruction [PDF File]. Retrieved from <http://www.airmartechonology.com/uploads/InstallGuide/17-461-01.pdf>.
- [8] Cross the Road Electronics. (2017). Talon SRX - User’s Guide [PDF File]. Retrieved from <https://www.ctr-electronics.com/Talon%20SRX%20User’s%20Guide.pdf>.
- [9] “Using the Hardware Serial Ports”. PJRC. Retrieved from [https://www.pjrc.com/teensy/td\\_uart.html](https://www.pjrc.com/teensy/td_uart.html).
- [10] Espressif Systems (2019). ESP8266 AT Instruction Set [PDF File]. Retrieved from [https://www.espressif.com/sites/default/files/documentation/4a-esp8266\\_at\\_instruction\\_set\\_en.pdf](https://www.espressif.com/sites/default/files/documentation/4a-esp8266_at_instruction_set_en.pdf).
- [11] “Joukovsky f=0% t=15% (joukowsk0015-jf)”. Airfoil Tools. Retrieved from <http://airfoiltools.com/airfoil/details?airfoil=joukowsk0015-jf>
- [12] Cross the Road Electronics. (2018). HERO User’s Guide [PDF File]. Retrieved from <https://www.ctr-electronics.com/downloads/pdf/HERO%20User’s%20Guide.pdf>.
- [13] Serial. (n.d.). Retrieved from

<https://www.arduino.cc/reference/en/language/functions/communication/serial/>.

[14] CrossTheRoadElec. (2017, March 6). CrossTheRoadElec/deprecated-HERO-Examples. Retrieved from <https://github.com/CrossTheRoadElec/deprecated-HERO-Examples/tree/master/HeroSonarModuleExample/HeroSonarModuleExample>.

[15] (n.d.). Retrieved from <http://freenmea.net/docs>.

[16] Pêtrès, C., Romero-Ramirez, M., & Plumet, F. (2012). A potential field approach for reactive navigation of autonomous sailboats. *Robotics and Autonomous Systems*, 60(12), 1520-1527. doi:10.1016/j.robot.2012.08.004

## 8.2 Roll and Pitch Calculations

Boat		Constants	
$\beta = 0\text{deg}$	Pitch	$\rho_a = 1.225 \frac{\text{kg}}{\text{m}^3}$	Density Air
$\Theta = 20\text{deg}$	Roll	$\rho_w = 1000 \frac{\text{kg}}{\text{m}^3}$	Density Water
$\beta = 30\text{deg}$	Angle POS and Apparent Wind	$Cl_s = 1$	Coeff Lift Sail
$\lambda = 20\text{deg}$	Angle Boat Axis and POS	$Cl_k = 1$	Coeff Lift Keel
$\delta_m = 10\text{deg}$	Sail Angle	$Cd_s = 0.003$	Coeff Drag Sail
$V_a = 3.5 \frac{\text{m}}{\text{s}}$	Apparent Wind Velocity	$Cd_k = 0.0065$	Coeff Drag Keel
Geometries (measured from COB)			
	Hull	Keel	Sail
Mass	$m_h = 16.19\text{kg}$	$m_k = 11\text{kg}$	$m_s = 3.095\text{kg}$
COM l	$lm_h = -0.12065\text{m}$	$lm_k = 0.115\text{m}$	$lm_s = 0.1996\text{m}$
COM h	$hm_h = 0.001\text{m}$	$hm_k = -1.1\text{m}$	$hm_s = 1.57\text{m}$
Area	$A_h = 1\text{m}^2$	$A_k = 28\text{m}^2$	$A_s = 1.73\text{m}^2$
COE l	$le_h = 0\text{m}$	$le_k = 0.26\text{m}$	$le_s = 0.1996\text{m}$
COE h	$he_h = 0\text{m}$	$he_k = -0.5\text{m}$	$he_s = 1.57\text{m}$
Ballast			
			$m_b = 9.23\text{kg}$
			$lm_b = -0.121\text{m}$
			$hm_b = 0.01\text{m}$
Statics			
	Hull	Keel	Sail
Force	$F_{sh} = m_h \cdot 9.8 \frac{\text{m}}{\text{s}^2}$	$F_{sk} = m_k \cdot 9.8 \frac{\text{m}}{\text{s}^2}$	$F_{ss} = m_s \cdot 9.8 \frac{\text{m}}{\text{s}^2}$
Moment Roll	$M_{rh} = F_{sh} \cdot hm_h \cdot \sin(\Theta) = 0.654\text{J}$	$M_{rk} = F_{sk} \cdot hm_k \cdot \sin(\Theta) = -40.537\text{J}$	$M_{rs} = F_{ss} \cdot hm_s \cdot \sin(\Theta) = 16.363\text{J}$
Moment Pitch	$M_{ph} = F_{sh} \cdot (lm_h) = -19.143\text{J}$	$M_{pk} = F_{sk} \cdot (lm_k) = 12.397\text{J}$	$M_{ps} = F_{ss} \cdot (lm_s) = 6.06\text{J}$

Figure 29: Calculations of roll and pitch

Aerodynamic Lift Force	$L_a = 0.5 \cdot \rho_a \cdot V_a^2 \cdot A_s \cdot Cl_s = 12.98\text{N}$
Aerodynamic Drag Force	$D = 0.5 \cdot \rho_a \cdot V_a^2 \cdot A_s \cdot Cd_s = 0.039\text{N}$
Aerodynamic Drag angle	$\theta_a = \tan\left(\frac{D}{L}\right) = 0.336$
Total Aerodynamic Force	$F_t = \sqrt{L^2 + D^2} = 12.98\text{N}$
Total Hydrodynamic Force	$R_t = -F_t = -12.98\text{N}$
Hydrodynamic Drag Force	$D_w = 0.5 \cdot \rho_w \cdot V_a^2 \cdot A_k \cdot Cd_k = 10.351\text{N}$
Hydrodynamic Side Force	$F_s = \sqrt{(R_t)^2 - (D_w)^2} = 7.832\text{N}$
Hydrodynamic Drag angle	$\theta_h = \tan\left(\frac{F_s}{R}\right) = 54.092\text{deg}$
Driving Force	$F_r = F_t \cdot \sin(\theta_h) = 10.514\text{N}$
Heeling Force	$F_h = F_r \cdot \cos(\theta_h) = 7.613\text{N}$
Angle of Attack Keel	$\alpha_k = \lambda = 20\text{deg}$
Angle of Attack Sail	$\alpha_s = \beta - \lambda - \delta_m = 26.509\text{deg}$
x btw COB and COM	$Rb = \sqrt{(lm_h - le_h)^2 + (hm_h - he_h)^2} = 0.121\text{m}$
x btw COM and COE sail	$Ra = \sqrt{(lm_h - le_s)^2 + (hm_h - he_s)^2} = 1.601\text{m}$
x btw COM and COE keel	$Rh = \sqrt{(lm_h - le_k)^2 + (hm_h - he_k)^2} = 0.629\text{m}$
Bouancy Force	$F_b = 9.8 \frac{\text{m}}{\text{s}^2} \cdot (m_h + m_k + m_s + m_b) + (L - R \cdot \sin(\lambda) - F_s \cdot \cos(\lambda)) = 389.357\text{N}$
Heeling Torque	$M_h = Ra \cdot (L \cdot \cos(\beta - \lambda) + D \cdot \sin(\beta - \lambda) \cdot \cos(\Theta)) = 20.481\text{J}$
Righting Torque	$M_r = -Rh \cdot [R \cdot \sin(\lambda) \cdot \cos(\lambda) \cdot (1 + \cos(\theta_h)) + F_s \cdot (\cos(\lambda)^2 - \sin(\lambda)^2 \cdot \cos(\Theta))] + Rb \cdot F_b \cdot \sin(\Theta) = 8.936\text{J}$
Total Heeling Torque	$\Sigma M_h = M_h + M_{rh} + M_{rs} = 36.837\text{J}$
Total Righting Torque	$\Sigma M_r = M_r + M_{rk} = -31.62\text{J}$
Torque on Boat	$\Sigma M = \Sigma M_h + \Sigma M_r = 5.217\text{J}$
Total Pitch Torque	$\Sigma M_p = M_d + M_{ph} + M_{pk} + M_{ps} + M_{pb} = -3.377\text{J}$

Figure 30: Continuation of calculations

## 8.3 User Manual

For the user manual, the team created a YouTube channel for videos to help with this process. Navigate to <https://www.youtube.com/channel/UCTJcecIIHhpFGg6Isr1fq9w> to find the videos.

### 8.3.1 Mechanical Setup

The mechanical setup videos are as follows:

1. Rudder installation
2. Rigid wing installation
3. Movable ballast installation
4. Charging the rigid wing
5. Keel installation

To prepare for any water testing, follow the "Kit Walk-through" video.

### 8.3.2 Hardware and Software Setup

An electronic schematic can be found in section 5.3.

1. Once all the wires are connected to each other, make sure to plug in the batteries labeled with a B or D, and then turn on the boat with the gray magnet piece. (Reference YouTube "Turn on boat/rigid sail"). If the boat does not turn on right away, the first thing to check is the fuse. If the fuse is blown, check to see which LED has illuminated next to a fuse. Then follow the leads from that fuse. Check for any shorts, which are commonly found at the section where the HERO board is. If the fuse is fine, the batteries might be dead. These batteries go through charge rather quickly (a future recommendation is to change them), so make sure you are always charging the batteries.
2. The trim tab electronics enclosure can also be turned on with the magnet. Make sure to the keep the magnet piece away from the electronic enclosure once it is turned on/off, because the magnet could switch the state of the enclosure if held too close.
3. To control the boat, we used a wireless LogitechF710 controller. For the controller, the team decided to make the left joystick the axis to control the movable ballast by moving it left and right. Joystick axis 2 was chosen to control the rudders. The rudders could be controlled by moving the right joystick left and right.
4. All of this can be referenced in "Running RC" on the WPI Sailbot channel

## 8.4 Code Description

All code can be accessed on the WPI Sailbot GitHub page, located at <https://github.com/wpisailbot/sailbot1819NEW>. Make sure you are in the sailbot1819NEW code, as that was what was written for this year's MQP. Other repositories can be used for reference.

### 8.4.1 Arduino Code

The main code for the Arduino was written using the Arduino IDE. The program to run on the Arduino is named "WebServerTestArduino". The point of this code is to set up the web server that can be run through the router housed within the same container. The code also allows for serial communication between itself and the HERO, and UDP messaging with the Teensy.

### 8.4.2 HERO Code

The HERO code was written in C# using VisualStudio 2017. The program to run on here is "HERO Serial Example". This runs all of the controls for the components of the main hull and lets them be controlled by the LogitechF710 wireless controller.

### 8.4.3 Teensy Code

The Teensy code was written using the Arduino IDE with the Teensy plugin, which can be found at [https://www.pjrc.com/teensy/td\\_download.html](https://www.pjrc.com/teensy/td_download.html). Once that is fully installed, then code can be uploaded. Use "rigidWingStateMachine" for this board. It allows for serial communication between the ESP8266 Wifi chip and for UDP messaging with the Arduino. Any further questions regarding any of this code should be able to be found in the comments.



## 8.5 Testing Kit List

This list contains all that is required for water testing. You can find a walk-through on the WPI Sailbot YouTube channel.

# PROPERTY OF WPI SAILBOT

## CONTENTS INCLUDE:

- 2 DIAPERS	- 2 SPONGES (ONE IN BOAT)
- 15 SMALL ZIPTIES	- 2 GADGETEER CABLES (FOR HERO)
- 1 USB-B CABLE	- 1 MINI USB CABLE
- 1 MICRO USB CABLE	- 1 LOGITECH WIRELESS GAMEPAD F710
- 1 ALLEN WRENCH SET	- 1 ALLEN WRENCH SET
- THUMB SCREWS (NEED AT LEAST 3)	- MISC. OTHER SCREWS
- 1 ANEMOMETER	- ELECTRICAL TAPE
- DOUBLE SIDED TAPE	- VARIOUS ROPE OF VARIOUS SIZES
- 1 PHILLIPS HEAD SCREWDRIVER	- 1 TUBE OF E600
- 1 PAIR OF NEEDLE NOSE PLIERS	- 1 SHARPIE
- 3 EXTRA O-RINGS	- CARBON FIBER TUBE FOR RIGID WING CONNECTION
- LARGE WASHER	- MAGNET FOR INITIALIZATION
- 12 AWG FLEXILE SILICONE WIRE (BLACK AND RED)	- EXTRA BATTERY CHARGER
- SANDPAPER	

### ALSO REMEMBER TO BRING:

- LAPTOP FOR CODE, 2 PFD'S (LIFE VESTS), RUDDERS, WOODEN DOWEL FOR TOP PART OF RIGID SAIL, TWO CHARGED BATTERIES

Figure 31: Kit of parts list