# Socially Distanced Robotics
# RoboBoat 2021 - Georgia Tech

Tyler Campbell, Justin Chau, Yash Chitale, Luke Dorrian, Sean Fish, Daniel Foreman,
Eric Fu, Ali Kazmi, Chenxuan Li, Jeffrey Pattison, Hannah Payne, Eric Phan,
Coline Ramee, Rahul Rameshbabu, Tristan Sarton Du Jonchay, Saahas Yechuri

***Abstract*—For many disciplines and industries, remote collaboration has become a necessity during the COVID-19 pandemic. Likewise, the Marine Robotics Group at Georgia Tech has adapted to the current situation through the use of online tools. This paper details developments in our vehicles, electronics, and design process made in preparation for the 2021 RoboBoat competition and other activities. While our access to campus facilities was limited, a new autonomous surface vehicle (ASV) was constructed without the use of advanced manufacturing methods, and a new hydrophone system prototype was built. Our Unmanned Aerial Vehicle (UAV) system received upgrades to its navigational capabilities. A new method of live collaboration for software development was created, allowing for members to develop without needing to install the full development environment.**

## I. COMPETITION STRATEGY

Prior to the start of lockdowns, the team had been preparing for the RoboBoat 2020 competition with the development and construction of a newly designed autonomous surface vehicle, hereafter referred to as ASV2020 [1]. This new vessel was designed with new pontoons featuring improved hydrodynamic qualities and durable dual layered fiberglass. The foam structures of the pontoons had been manufactured prior to the pandemic, but the fiberglass process had not been completed. Unfortunately, due to limitations imposed by the lockdown and concern for personal health and safety of our team, ASV2020 is still awaiting final assembly and its maiden voyage.

In the case RoboBoat 2021 had been held in person this year, the team would have prioritized completing construction in late spring and early summer, as vaccines became available. The new boat would have utilized the existing Adept software stack which has been tested in past competitions.

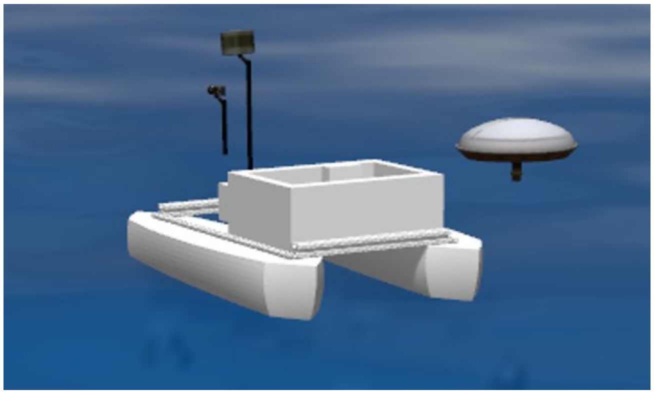

Figure 1: Simulation of ASV2020.

While our in-person activities were hindered by the circumstances, we took advantage of this opportunity to focus on other aspects of design and collaboration.

### A. Design

With the distance separating our members, we focused on parallel development of new hardware systems. Each project involved the development of a prototype, with each built by an independent member. This way, development of a system could progress, with the hope that future iterative improvements could be made as a team when we are able to interact in person.

*1) Autonomous Surface Vehicle:* This year, a new autonomous surface vehicle was developed separately from the existing ASV2020 project. This new ASV, known as the *Measure Once Cut Twice* (MOCT), was assembled with the purpose of testing new ideas on a smaller and simpler testbed. This vessel is not intended to serve as a competition vessel, but rather a supporting vehicle to facilitate iterative development and provide a teaching platform.

*2) Hydrophone:* Significant progress was made in the design of a hydrophone system developed in-house. Research into necessary hardware and filtering has informed our component selection, and a prototype is in development. This hydrophone system will enable our team to be able to localize pingers in the RoboBoat competition environment, as well as in other competitions. The knowledge gained in building a sensor system from the ground up will be helpful as we seek to build other in-house modules as well.

*3) Unmanned Aerial Vehicle:* The UAV underwent considerable development during the past year. Autonomous navigation was implemented, and the addition of further capabilities such as transportation of objects is planned. The creation of a companion system will further increase the potential of our system in the RoboBoat competition, specifically for tasks relating to transporting objects, and will also have potential uses in the upcoming RobotX competition.

### *B. Collaboration*

This year, we have faced the challenge of onboarding new members without face-to-face interaction and without access to the machines or dedicated computer hardware. Another challenge we have had during this period of remote work is lost knowledge, as many of our integral team members have graduated. To solve these issues, we have been utilizing collaboration software.

Our collaboration method leverages another component of our strategy, which is the use of simulation. Through the use of the Virtual RobotX (VRX) simulation environment [2], we have been able to test ideas without the need to go to a physical testing location. Further, a model of ASV2020 has been created through the adaptation of the VRX simulation to enable testing specific to the vehicle.

## II. DESIGN CREATIVITY

This year, many of our innovations were made on individual components, as our work on hardware was focused on developing independent prototypes. In addition to this, changes at an organizational level have provided a new way to collaborate on software development.

### *A. Autonomous Surface Vehicle*

Figure 2: Front view of catamaran, with thrusters.

The *Measure Once Cut Twice* was constructed off campus, without access to advanced manufacturing methods such as 3D printing, CNCs, or laser cutters. It was designed to minimize cost, size, and complexity in construction, and to facilitate easy testing and transportation, as the vessel is able to easily fit into the passenger seat of a car. Building MOCT also provided a test of new construction methods and materials. With a rather small displacement of around 15lb, it is designed to only carry a light computer and a few small sensors.

The hull of the new ASV is a catamaran design, providing stability like our previous ASVs [3], but differs greatly in its assembly. It is one piece, constructed from a used Tri-fold Foam Display Board (28” x 40”) which was scored and folded into the shape of a boat. This hull was then given a coating of rubber using Flex Seal Liquid, while the structure was held in place using pins and packing tape.

The boat is equipped with two thrusters constructed from converted bilge pumps, as is common in home-made ROVs [4]. These thrusters are mounted on a construction of Owens Corning pink foam and foamboard, with the thrusters located close to the center of mass as to facilitate turns while minimizing motion using differential thrust control. This differs from the holonomic thruster configuration [1] to be implemented on ASV2020, but still provides a simple, if somewhat noisy, control layout.

While design and assembly took place over several semesters, the actual time spent constructing the boat was about a day with just one person. All materials excluding the electronics can be sourced from local stores such as Walmart and Home Depot. Since the liquid rubber can cure in 24 hours, it is possible for the hull to be completed within two days, in case a vehicle needs to be rapidly constructed.

### *B. Hydrophone*

This year, there was significant progress made in the design of an in-house hydrophone system. Cumulatively, the hydrophone system incorporated multiple engineering disciplines: Mechanical design to create a watertight external housing, Electrical to design and fabricate a Printed Circuit Board (PCB) with the required signal preprocessing and connectivity, and Computer Science to perfect the signal processing algorithms.

*1) Mechanical Design:* Hydrophone circuitry is notoriously sensitive to environmental noise, and in past attempts, directly connecting them to voltage sources shared by even something like an Arduino board sending Pulse-Width modulated signals has caused severe measurement issues. Additionally, the hydrophone team wished to have a portable solution that could be used not only on this boat, but on other vehicles such as our RoboSub and WAM-V. For this reason, the best solution was deemed to be an external, watertight, housing for the hydrophone circuitry. Due to the pandemic, work on this enclosure has yet to begin.

*2) Electrical Design:* After researching commercial and other RoboBoat teams’ hydrophone solutions [5], we identified multiple requirements for the hydrophone electronics. A steady voltage source, as well as electronic decoupling from other onboard electronics was critical to ensure the hydrophones’ circuitry did not experience transient electronic ripple, which would make readings unreliable. Capacitor-based filters and a variety of voltage-regulator designs were identified as possible solutions. Hydrophones, for this application, ideally would have a uniform gain distribution. The RESON TC 4013, which had been procured in prior years, satisfied this requirement. Transduced hydrophone signals would be highly transient and of low-voltage, making them susceptible to external interference and loss during the signal transmission and Analog to Digital (ADC) sampling processes. To combat this, an op-amp solution was formulated that would amplify the hydrohone signal, and re-bias the signal to be centered in the ADC’s measurement range. Additionally, with the relatively high bandwidth of the hydrophones and amplifier, spurious noise was expected. As the design was already actively amplified, an active bandpass filter was conceived to perform pre-filtering on the hydrophone data, passing through signals between 30-45KHz, the expected frequency of the acoustic beacon. Although this general design was conceived in the 2020 competition year, in the 2021 competition year, many of the numerous unknowns were fleshed out in the design, although the exact circuitry has not been decided upon at this time. For sampling and data-processing, the STM32 NUCLEO-H745ZI-Q board was chosen for its 16-bit, 3-channel SAR ADC that could handle external voltage references and sample at 3.6 Megasamples-per-second (Msps). The STM32 board is well-supported and documented, and its built-in ADC simplified the circuitry design for the hydrophone board. External references would allow the board to accept the same voltage used by the amplification and pre-processing circuitry, and eliminate biases that would otherwise be accrued if the two electronic systems were independently generating voltage references. The Megasampling rate was critical to ensure proper data collection. Reviewing sampling solutions from other RoboBoat teams, hydrophones were typically sampled at rates between 500Ksps to 42Msps. In prior attempts to sample hydrophone data, 110Ksps was used to sample a signal at 45KHz, and although it was above the Nyquist frequency for the data, performed very poorly in capturing the signal. To prevent this from happening again, a sampling rate in the Msps was very desirable.

*3) Algorithm Design:* In 2020, the hydrophone algorithm attempted to directly measure the time of arrival of a sinusoidal hydrophone signal, and use 3 times of arrival to perform 2D multilateration. However, this design had multiple problems. Due to multipathing and signal corruption, the sinsoidal signal emitted by an acoustic beacon was not actually sinusoidal when it was received by the hydrophones. Secondly, there is frequently ambiguity in when the signal was received due to a noise-floor in the measurements. The multilateration formulas are highly sensitive to ambiguity in arrival time, especially when the acoustic source is located a far distance from the hydrophones, compounding the problem. To combat these problems, a differenced version of the multilateration algorithms was used, transforming them from a parabolic to hyperbolic form. This technique, which appears to be the standard approach to problems such as this, addresses both of the above problems when used in conjunction with correlations between hydrophone time-series. By repeatedly calculating element-wise correlations between recorded hydrophone time-series, each repetition shifting the points to be correlated by some time delta, the best estimate for time of arrival difference could be found by selecting the time delta corresponding to the maximum correlation. This selection process allowed time differences to be accurately found for non-sinusoidal signals, and for noise-floor considerations to have less of an impact on the calculated solution.

### C. Unmanned Aerial Vehicle

One task for this year's RoboBoat competition was to deliver up to four objects to the delivery dock. To complete this task, the payload could either be delivered using the ASV or a UAV. In order to achieve this goal, the Georgia Tech Marine Robotics Group decided to design and build a UAV to deliver the payload. The UAV would launch from the ASV and then fly autonomously to the delivery dock location to drop the payload.

*1) Vehicle Design:* When designing a UAV, the first design consideration is to decide what the UAV needs to accomplish. First Person View (FPV) racing UAVs have a much different design than UAVs used for delivery. The FPV UAVs have smaller frames with faster motors and are designed for agility and speed, while the delivery UAVs utilize larger frames and slower motors to trade agility and speed for stability and the ability to lift heavier payloads. With this in mind, a 450 mm quadrotor frame was chosen. On this larger frame size, 12 inch propellers can fit to provide plenty of lift while also having enough space for all the required electronics and payload. After deciding the frame, the next step was to design the power train of the UAV, consisting of the battery, ESCs, and motors. The motors need to be selected based on the lift they can produce to lift the weight of the UAV, but at this point the weight of the UAV is unknown so the thrust required is unknown. To circumvent this conundrum, an estimation of the final weight of the UAV was created to start selecting which motors to use. The estimated weight of the UAV was constructed by combining the weight of the frame, approximate weight of four ESCs, approximate weight of four motors, the flight controller, flight computer, GPS, battery, payload, and landing gear. The weights of the motors and ESCs were approximated due to not having a specific motor chosen. The final UAV weight estimate was a conservative 7 lbs. With this initial estimate, the RCTimer HP2814

810 Kv motors were chosen because they produce over 2kg of thrust per motor with a 4 cell LiPo battery and 12 inch propellers. A lower Kv rating was chosen for the motors since larger propellers were used, and the rotor/stator size were selected due to the high torque to easily turn the larger propellers. For UAVs, the thrust to weight ratio should be over 2 to ensure good controllability of the UAV, and this initial thrust to weight ratio is approximately 2.5.

With a 4 cell LiPo battery, the motors selected can pull up to 23 amps per motor. The ESCs and battery need to be able to handle this strain. In order to handle the current required for each motor, four 30 amp ESCs were selected. The battery also needed to be able to handle the current output of four motors potentially pulling 23 amps each, plus the onboard electronics required. The battery selected was a 5000mAh 4 cell LiPo battery with 35C rating, which allows the battery to discharge almost 170 amps. This battery is more than enough to handle the demand of the electronics and motors.

With the power train designed, the flight controller needed to be selected. A Pixhawk with Ardupilot was selected due prior experience working with the equipment. Connected to the flight controller was a GPS that had accuracy within 3 meters. The combination of GPS with the Pixhawk flight controller allowed for autonomous GPS waypoint following. The Pixhawk flight controller allowed for lower level control and stability, but a flight computer was required to do higher level computation like image classification and mission execution. A Raspberry Pi 4 with 2GB of RAM was selected as the companion computer since it was easy to integrate with the Pixhawk.

The final configuration of the UAV consisted of the 450 mm frame with 4 motors turning 4 12 inch propellers powered by a 4 cell LiPo battery and driven by a Pixhawk flight controller connected to a Rasbperry Pi companion computer. A downward facing camera was added to be able to find the delivery dock. The final weight without payload was found to be 5 lbs, which is below the initial estimate of 7 lbs. With this final weight, the thrust to weight ratio is 3.52, which is sufficient for the UAV since it will need to lift a payload. With the weight and power train, the estimated flight time was 12 minutes, which is more than would be required in the competition to deliver the payload to the delivery dock.

Figure 3: UAV during test flight.

*2) Mission Planning and Mission Execution:* While a mission plan was created, the actual development and implementation of the required programming for mission execution was not completed, but below is an outline of how it would have been done. After examining the course, the best time for the UAV to launch would be when the ASV was at the acoustic docking station. This point in the course is next to the delivery dock so the UAV would not have to fly far. The ASV can SSH into the Raspberry Pi to run a Python script to begin the UAV mission. DroneKit is a Python package that allows the Raspberry Pi to communicate with the Pixhawk flight controller to send commands. The ASV would then command the Raspberry Pi to launch this mission script. The mission script would contain other functions including Takeoff, Scan, Land, Deliver, and Return to Home. The Takeoff function would run first to get the UAV to launch to a given altitude.

After launching, the UAV would then begin flying in the general direction of the delivery dock in the Scan segment of the mission. Using a Raspberry Pi camera on the UAV with an image classifier on the Raspberry Pi, the pattern of the delivery dock could be recognized and used as a target for the UAV and a trigger to begin the descent and drop off. Again, the DroneKit package would use the image from the Raspberry Pi camera with OpenCV [6] to update the UAV on where to go to get the dock directly below the UAV, and the Raspberry Pi would then send the commands to the Pixhawk. After the delivery dock was located and centered under the UAV, the UAV would then begin the Land and Deliver portion of the mission.

Unfortunately, the UAV was not developed enough to have a release mechanism for the payload just yet. However, the plan was to add a servo motor powered from the Pixhawk to carry a net that could contain the payload. Upon landing on the delivery dock, the servo would turn to release the net from the drone, completing the delivery. Upon completing the delivery, the UAV would enter into the Return to Home function, where the UAV returns to the predetermined Home on shore instead of trying to return to the ASV. The Return to Home segment is enabled by the Pixhawk with the ability to fly to a GPS waypoint and land, thus completing the mission.

### *D. Software*

We are still utilizing the Adept software stack that has been in development for the past few years, and have been working with this stack in simulation. However, this pandemic has provided us the opportunity to start porting our system from ROS1 to ROS2, as the gap in competitions gives us more time to solve various issues that may arise. One area of development was improved navigation through the use of a Kalman Filter.

### *E. Kalman Filter*

In years past, GT RoboBoat has attempted to improve its navigation system with a Kalman filter to blend IMU and GPS measurements. This was motivated by sporadic GPS measurements during competitions, which paralyzed the GPS-exclusive navigation system, as well as a general desire to improve state estimation for the boat. In the spring of 2020, a Kalman filter was implemented, but failed to converge on a solution, and was ultimately discarded. In 2021, a second attempt was made to perform Kalman filtering. To prototype this filter, the filter was first applied in simulation in the VRX WAM-V Gazebo simulator. An Extended Kalman Filter (EKF) was chosen, as it would perform better than a standard Kalman Filter under nonlinearities in the estimated system. The filter estimated x-y planar position and yaw, as well as their first and second derivatives. Updates to thruster commands and IMU and GPS measurements were used to perform correction updates to the estimated state. GPS latitude, longitude and height were converted into a delta x and y from the boat's starting point. IMU data was handled raw. Thruster commands, ranging between -1 and 1, were converted into expected thrust forces and moments using knowledge of the simulated model's geometry and corresponding calculated mass properties. For additional realism, the model had the optional ability to account for viscous damping constants in each of the 3 principal directions. The filter could predict future propagated states under its model, as well as generate its internal Jacobian via analytical derivation or numerical differencing.

The 2021 Kalman filter showed some promise of potentially replacing the current navigation algorithm, a GPS-only exponential filter, as it was shown to be able to blend GPS, IMU, and thruster updates into its estimation. This represented a step forward, as earlier attempts had rapidly diverged due to instabilities inherent to their state transition matrices. While the 2021 filter still would diverge, especially when estimating viscous drag, it had the ability to perform state estimation while maneuvering the simulated WAMV moving at full speed.

Currently, the 2021 Kalman filter needs improvements before it can be a true replacement for the existing navigation solution. Its instability is a critical issue, and is worsened by a missing knowledge of thrust forces, observed imprecisions during matrix inversions and a numerically-sensitive formulation of drag force. Additionally, the rate information from the IMU appears to warrant further investigation, as it appears that the observed velocities have the right sign, but are far too small compared to a ground truth, indicating a possible timing issue within the filter. When compared against the exponential GPS filter for positioning, the Kalman filter was found to have more error, possibly owing to the rate information not being properly passed and interfering with positioning estimation. Further investigation and tuning will be needed to improve the accuracy of the RoboBoat's navigation system.

## III. EXPERIMENTAL RESULTS

The main software stack was tested in the VRX simulation. Through observation of the performance of the vehicle in VRX, the team determined potential areas of improvement. Further, parameters of the dynamic model of the ASV2020 simulation were adjusted by trial and error to better simulate realistic performance.

As our team was unable to meet in person, physical tests have largely been limited to independent systems.

Figure 4: View of catamaran, in pool.

### A. Autonomous Surface Vehicle

A pool test for the MOCT vehicle gave valuable insight into the performance of the new thrusters. The new thrusters use brushed DC motors unlike our usual Blue Robotics thrusters, so the method of powering and controlling these motors was new and involved the use of a dual H-bridge motor driver. In the test, the thrusters were only able to sustain propulsion for several seconds at a time. It is suspected at this time that the current PWM frequency is too high, causing the motor driver to overheat. A future test will confirm if this was the issue.

### B. Hydrophone

To test the efficacy of a differenced version of the multilateration algorithms, a prototype algorithm was created in MATLAB and applied to data collected four years ago using 2 hydrophones and an acoustic beacon. As 3 hydrophones are needed to fully constrain and identify a unique planar solution, the multilateration formulas were modified to assume parallel incident rays being received by both hydrophones. Correlation was used to identify the time difference between hydrophone measurements, and the result was used to identify the heading of the acoustic beacon relative to the perpendicular bisector of the two hydrophones. The results appear promising, as they match up nearly identically with the heading that was identified by the original test-operators.

Figure 5: Testing hydrophone circuitry.

In addition to this, tests of the actual hydrophone were made using an oscilloscope, and successfully demonstrated that the presence of the pinger could be detected using our sensors.

### *C. Unmanned Aerial Vehicle*

The UAV system was tested several times. While there were some initial issues with controller settings, these were subsequently resolved. The UAV succesfully demonstrated a flight, maintaining a defined altitude , navigating to a specified GPS coordinate, and then landing.

### *D. Collaboration*

This past academic year marks the first time we started leveraging live collaboration in our development process. Through the use of programs such as BlueJeans video conferencing software and the LiveShare extension for Visual Studio Code, we were able to develop interactively. This provided advantages such as rapid iteration in software prototyping and reducing the need to resolve merge conflicts. This method was first utilized in the Virtual Ocean Robotics Challenge [2], where we created a new software stack from scratch in a week that performed reasonably well.

This spring, we began using this live collaboration process for onboarding, to get around the difficulties of installing ROS and Linux on new members' machines. This method proved to be quite effective in providing a crash course on the software stack and framework we use. However, there were limitations to this technique that we still need to overcome. One issue was that work could only be done when the host computer was active, thus making all work simultaneous, which is good for tutorials but not for entire projects. Due to this, it was still required for members to install a local Linux and ROS environment on their own computers once the tutorial portion of the semester was over, nullifying the original advantage in the long run. This technique is great for collaborative coding sprints, but is not sustainable for long term projects. As new software such as Windows Subsystem for Linux 2 are updated, we hope to further reduce the difficulties of installing robotics development environments.

## IV. CONCLUSION

Though this past year has been unexpected and challenging, we are excited to apply the new skills and knowledge we have gained. We have been able to construct component systems that will be integral to our future efforts, and have been able to improve our collaborative capabilities as an organization. With a return to full in-person efforts this upcoming fall semester, the team will be able to start off strong with the developments made and lessons learned in preparation for this year's competition.

## V. ACKNOWLEDGEMENTS

This team's activities are made possible by the generous contributions from Georgia Tech's Aerospace Systems Design Lab, Student Government Association, and School of Aerospace Engineering Student Advisory Council. Our group is also supported by our sponsors: Fischer Connectors, Greenzie, ConnectTech Inc., and Pattent. Our team thanks them for their support! Further, we would like to acknowledge the former and current members of the Marine Robotics Group whose work forms the basis of all of our activities. To those who have graduated and are graduating during this time, we will miss you!

## VI. REFERENCES

## REFERENCES

[1] A. Kazmi, A. Sinha, A. Velte, C. Ramee, D. Foreman, D. DiCarlo, E. Jahoda, E. Fu, E. Phan, J. Pattison, J. Adrande, J. Chau, L. Dorrian, M. Schlaff, M. C. Martin, M. Gilmartin, P. Meyer, P. Eddu, R. Rameshbabu, R. Kuramshin, R. Vepa, S. Fish, T. Daino, T. Campbell, and V. Chow, "Roboboat 2020: Technical design review,"

[2] B. Bingham, C. Aguero, M. McCarrin, J. Klamo, J. Malia, K. Allen, T. Lum, M. Rawson, and R. Waqar, "Toward maritime robotic simulation in gazebo," in *Proceedings of MTS/IEEE OCEANS Conference*, (Seattle, WA), October 2019.

[3] V. Ruia, J. Chen, H. Chen, A. Spalding, A. Joshi, L. Arsene, and S. Seifert, "Hall and boats: Roboboat 2017 - georgia tech,"

[4] S. Thone, "Converting a bilge pump to run a propeller."

[5] R. Summers, "Hydrophone development."

[6] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

## VII. APPENDIX A - COMPONENT SPECIFICATIONS

| **Component** | **Vendor** | **Model/Type** | **Specs** | **Cost** | **Status** |
|---|---|---|---|---|---|
| MOCT Hull | In-house | N/A | 40” length, ∼15 lb max displacement | $20 | Testing |
| MOCT Thruster | Sanuke | Bilge Pump - Heavily modified | 12V 1100gph | $20 | Testing |
| MOCT Motor driver | TIMESETL | L298N Driver | Double H-bridge, 5V-35V, 2A max | $3 | Testing |
| MOCT Teleoperation | Spektrum | AR610 Receiver | 6ch 2.4GHz DSM2/DSMX | $40 | Testing |
| MOCT Computer | Raspberry Pi | Model 3B | 1GB RAM | $49 | Not Integrated |
| MOCT Battery | Zippy | Compact LiPo | 11.1V 3s 1300mAh | $12 | Testing |
| Hydrophones | Teledyne Marine | Reson TC 4013 | http://www.teledynemarine.com/reson-tc4013 | Unknown | Testing |
| Hydrophone processor | STMicroelectronics | STM32 NUCLEO-H745ZI-Q | https://www.st.com/en/evaluation-tools/nucleo-h745zi-q.html | $29 | Development |
| UAV Platform | Uncertain | 450mm | 450mm frame | ∼$20 | Testing |
| UAV Battery | Tattu | LiPo | 14.8V 4s 5200mAh | $61 | Testing |
| UAV Motor | Rctimer | HP2814 Brushless Motor | 710KV | $20 | Testing |
| UAV Receiver | Turnigy | iA6C PPM/SBUS Receiver | 8ch 2.4GHz | $11 | Testing |
| UAV Propeller | QWinOut | Carbon Fiber Propellers | CW/CCW 1245 | $31 | Testing |
| UAV Flight Controller | Pixhawk | 2.4.8 | https://docs.px4.io/master/en/flight_controller/pixhawk.html | $65 | Testing |
| UAV Companion Computer | Raspberry Pi | Model 4B | 2GB RAM | $35 | Testing |
| Algorithms | In-house | | | | |
| Vision | OpenCV and In-house | | | | |
| Acoustics | In-house | | | | |
| Localization and mapping | In-house | | | | |
| Autonomy | In-house | | | | |
| Team Size (number of people) | 16 | | | | |
| Expertise ratio (hardware vs. software) | 1:3 | | | | |
| Testing time: simulation | 10+ hours | | | | |
| Testing time: in-water | 1 hour | | | | |
| Inter-vehicle communication | WIP | | | | |
| Programming Language(s) | Python, MATLAB, C++ | | | | |