Crystal Clear: RoboBoat 2024: Technical Design Report

Erin Beazley, Shawn Coutinho, Sean Fish, Santiago Fiz, Zachary Greenberg, Yiming Guo
Ryan Otsuka, Manuel Roglan, Mitchell Turton, Aaron Wu, Jorge Ortiz Solano
Georgia Tech Marine Robotics Group
Atlanta, Georgia

***Abstract*—In preparation for the RoboBoat 2024 competition: Ducks Overboard, the Georgia Tech Marine Robotics Group built upon the *Crystal Clear* Autonomous Surface Vehicle platform, which debuted at RoboBoat 2023 [1] [2]. This year, the primary design goal was to enhance existing systems, improving their robustness and reliability with a special focus on the autonomy systems. *Crystal Clear* features a new arm subsystem tasked with retrieving game objects from designated field areas and a revamped racquetball shooter. Major improvements were made to the ASV's perception and path planning due to newly acquired hardware and additions to the software stack. Extensive work was put into shrinking and organizing the vehicle's electrical system. Advancements in PCB and power system design were implemented to ensure a safer system overall.**

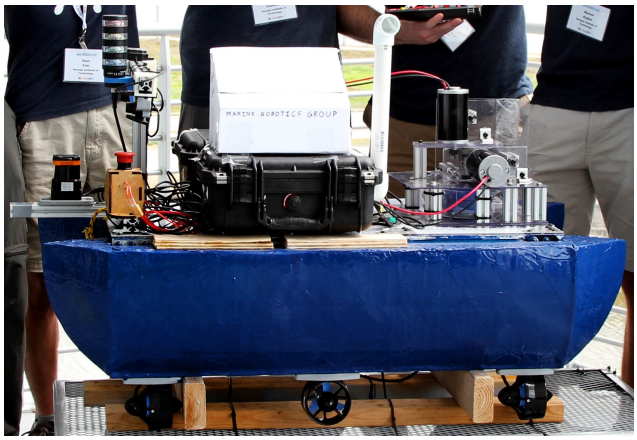

Fig. 1. *Crystal Clear* out of water.

I. Technical Strategy

GT Marine Robotics Group's (GT MRG) approach for RoboBoat 2024 was to focus on building off of its successful platform, *Crystal Clear*, from the prior year. This meant staying true to the Autonomous Surface Vehicle (ASV)'s initial, core design principles while improving the consistency of and increasing the capabilities of the ASV. To accomplish this, the team divided the competition up into various categories of tasks, each of which would require different subsystems and therefore different levels of work from its three main subteams: mechanical, software, and electrical. The major task categories identified by the team are as follows: navigation (encompassing Tasks 1, 2, 5, and 8), docking (covering Tasks 3 and 4), and object manipulation (involving Tasks 6 and 7). This year, to build on the team's past success, the team has decided to attempt every task, however, tasks considered navigation and docking take a far higher priority.

II. Competition Strategy

A. Navigation Tasks

For navigation tasks, we coordinate our perception, navigation, and controller software subsystems to complete the task. Generally, the perception subsystem will identify objects of interest, such as buoys, and send the information to a main control node. The main control node is a finite state machine which uses perception information as well as task goals to determine where the ASV should navigate to. The main control node will send a goal waypoint to the navigation subsystem, which will plan a path to the goal. Upon reaching the goal, it informs the main control node. The controller subsystem follows the path created by the navigation subsystem to propel the ASV along the path.

The navigation tasks generally involve navigating between gates or channels formed by buoys. To perform these tasks, the perception subsystem identifies buoys in the field of view of the boat. The main control node uses this information to find a waypoint in the center of the next gate. Then, the navigation and controller subsystems guide the ASV to the waypoint and inform the main control node it has reached the goal. This process is repeated for as many gates as the task requires.

Navigating between tasks is one of the hardest challenges on the course. Our strategy traditionally has been to estimate waypoints where each task starts and then perform the task using perception once at the starting waypoint. This year, largely due to acquiring a VLP-16 LIDAR with a 100 meter range, we are working to develop algorithms that will guide the ASV to the next task without needing pre-set waypoints.

The subsystems required for completing navigation tasks are split into packages and are designed to allow work on different packages to proceed in parallel. This methodology increases the number of team members which can work on the software for these tasks effectively.

B. Docking Tasks

The Docking Tasks category, which includes Docking and Duck Wash, is accomplished by the ASV by first identifying the dock via the LIDAR. Once the rough contour of the dock has been identified, the ASV then plots a course to come within viewing distance of the dock to identify the various placards that could be placed at each bay. Once the cameras are within view of the dock and the ASV has successfully

reached its designated waypoint, the ASV then begins strafing utilizing its holonomic movement to ensure each bay is viewed by the cameras for sufficient identification time. The images are then processed onboard the Nvidia Jetson Orin, classifying the image viewed and designating the bay based on the shape and color it viewed. The ASV will only view as many bays as necessary to find its targets (namely, the color/shape the team was assigned before the run and the duck). Once the required bays are identified, the path planner will set a waypoint target in the designated bay, having the ASV enter the bay and remain close to the dock utilizing station keeping for a few seconds to demonstrate its parking. Upon completion of its parking, the path planner will now set course for the bay with the duck while simultaneously rotating the boat such that the back of the boat faces the dock. This allows the water shooter to effectively spray the placard without risking electronics getting wet from the water spray.

C. Object Manipulation Tasks

The Object Manipulation task category encompasses Collection Octagon and Delivery octagon and marks the area that saw the most change concerning game design between RoboBoat 2023 and RoboBoat 2024. The team identified that a portion of the Delivery Octagon task could be accomplished by the ball shooter present on *Crystal Clear* from the last competition season, assuming preload racquetballs were available. This would be accomplished following a similar logical pipeline as last year's Feed the Fish task, namely locating the target area utilizing the ASV's LIDAR and cameras, positioning the ASV next to the task utilizing the ASV's path planning, localization, and station-keeping capabilities, and spinning up the single flywheel to deliver the racquetballs to the target area. As for the Collection Octagon, the ASV identifies the task and the relative position of racquetballs to collect, maneuvers to a suitable position calculated based on the arm's trajectory and the ASV's position in relation to the task, and actuates its arm to scoop the racquetballs to be fed directly into the aforementioned shooter. Given the simplistic actuation method of the arm, its trajectory should remain quite similar in theory, and, as such, the software can plan out exactly what distance the ASV needs to be away from the task such that the arm can successfully pick up its target. The team is then able to multipurpose its sequencer mechanism from the ball shooter to store the collected racquetballs until the ASV reaches the Delivery Octagon.

D. Managing Complexity

While we enjoy developing more complex systems that allow the ASV to perform more difficult tasks, we have also striven to maintain reliability. To do this, our software subsystems have a variety of different algorithms that can be used to accomplish goals such as identifying buoys or navigating to goals. Some algorithms are more complicated than others and can be harder to debug if (and when) they fail. In order to ensure no one algorithm serves as a point of failure for accomplishing a task, we have created many parameter files that allow us to easily tune existing algorithms or swap algorithms out for others. At competition, if we find an algorithm is failing, such as a path planner, we can easily swap it out during a run and work on debugging later.

Furthermore, the electrical system was designed with reliability in mind. In this case we sacrificed some modularity in order for more secure connections in the form of wire wraps, tighter cable routing, and specific connectors, such as JST. This is all done to ensure that the ASV is able to keep running despite problems it might encounter in the software and so that problems can be more quickly isolated to either software or electrical.

III. Design Strategy

A. Mechanical Design

Crystal Clear was designed to have several improvements over previous vessels fielded by GT MRG, especially in terms of maneuverability, hull efficiency, and consistency. The mechanical design priorities of this year were to reinforce what was manufactured last year, revamp subsystems with subpar performance, and design a new subsystem to effectively accomplish a new task.

1) Hull and PowerTrain Design: The team decided to attempt to design a variety of new hull designs during the earlier part of the season to explore whether or not it was worth it to manufacture a brand-new hull. This new hull would be hollow, allowing for easier storage and transportation, improving modularity, and reducing the overall height of the ASV. However, it was decided to stick with the original dual epoxy and fiberglass coated foam pontoons of *Crystal Clear* due to their stability, inability to leak, and relative simplicity. The powertrain was also decided to be kept the same, though not before extensive tests were run to evaluate the performance of various powertrain configurations, namely tank drive (4 front-back thrusters), X-Drive (4 diagonal motors, offset °45 to front-back), 4 thruster H-Drive (2 front-back thrusters at back, 2 left-right thrusters at front), and 6 thruster H-Drive (4 front-back thrusters, 2 left-right thrusters in the middle). The results of these trials showed a clear advantage to utilizing a 6 thruster H-Drive which is the original configuration of *Crystal Clear*.

2) Ball Shooter: The ASV's ball shooter has gone through three different major iterations so far and each has significantly improved on at least one aspect of the previous. The first version was built using wood sides and was a fully vertical shooter, utilizing top spin to gain extra distance and accuracy on the shot. However, this design was deemed to make *Crystal Clear* far too top-heavy as the ASV was seen taking on waves during water tests. As a result, the design was scrapped for a horizontal sequencer and vertical shooter with backspin which helped drastically lower the center of gravity of the ASV. However, this design was manufactured purely by hand and therefore suffered from a lack of precision and consistency in terms of racquetball compression throughout the sequencer and flywheel, hurting performance on the water. Therefore, the newest design looks to learn from all of these lessons and

features a properly manufactured single flywheel featuring new brushless motors at a 1:1 gear ratio for both the sequencer and flywheel. Additionally, due to the addition of an arm, the carrying capacity of the new ball shooter has been upgraded to six racquetballs in the sequencer at any point in time with any additional picked up stored in the arm.

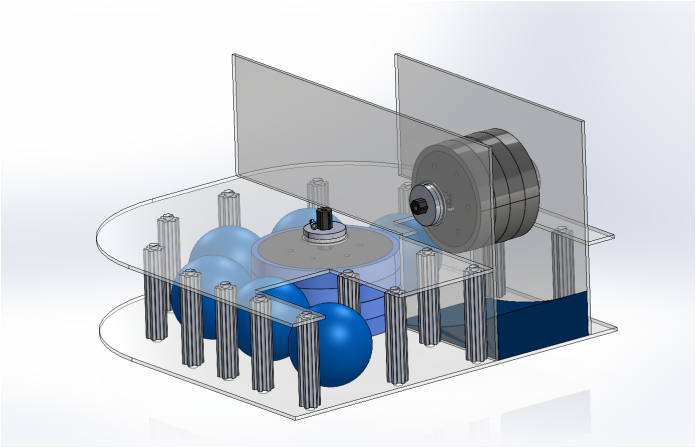

Fig. 2. Ball launcher with sequencer and single flywheel, no motors render.

3) Arm: *Crystal Clear*'s arm was designed with very specific design constraints given its precarious positioning and necessary detachability. These constraints were decided such that if the arm got caught on the task, the arm would sooner detach than pull the ASV underwater. Additionally, the team recognized the need for the arm to intake off the side of the ASV, therefore introducing concerns of weight imbalance which would force the software to compensate. Additionally, it was decided that the arm must have a non-powered intake due to extra weight and the team not wanting to work with fully waterproof motors. Finally, if the team were to invest the time to develop an arm, the racquetballs the arm retrieves must be able to feed smoothly into the sequencer for the ball shooter. Given all of these requirements, nearly all of the common designs for powered arms were disqualified such as 2-bars, 4-bars, etc. Additionally, fishing net systems were also not an acceptable choice due to the difficulty associated with feeding the game objects collected to another system. The final design that had merit was an arm that resembles the joints of a finger with an additional 4th stage that flips out on deployment. With such a system, *Crystal Clear* can maintain a relatively low center of gravity as the winch motor actuating the system can be placed towards the center of the vehicle with most other components at the edge being lightweight plastics (i.e. polycarbonate and acrylic). This design is accompanied by a scoop that allows *Crystal Clear* to intake racquetballs and rubber ducks while allowing water to spill out through lightening holes. Additionally, when the arm intakes racquetballs, the ASV can feed them into the sequencer as the racquetball falls through sequential digits, directly into the sequencer.

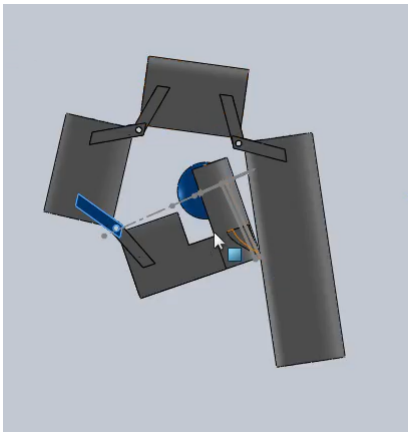

Fig. 3. Arm featuring racquetball for scale, no motor render.

B. Software Design

The software for *Crystal Clear* is built using ROS 2 (Robot Operating System). ROS is an open-source collection of libraries that greatly aid in the development and organization of software for robots. During the past year, we mainly focused on improving the software from last year and creating new debugging tools. Our software stack, called Virtuoso, can be roughly divided into a few major subsystems: a localization subsystem responsible for determining the robot's pose in the world, a perception subsystem responsible for identifying and locating objects of interest, a navigation subsystem responsible for finding paths around obstacles to goals, and a controller subsystem responsible for translating plans into motor outputs.

1) Localization: To localize the robot, our localization subsystem uses GPS and IMU sensors. The GPS position, GPS velocity, IMU rate gyro angular velocity, and IMU magnetometer heading are fused through an Extended Kalman Filter implementation provided by the open-source Robot Localization package.

2) Perception: To perceive objects of interest in the environment, our perception subsystem uses camera and LIDAR sensors. The Velodyne LIDAR mounted near the top of the robot provides the ASV with a point cloud representation of obstacles nearby. The point cloud is useful for the navigation subsystem to plan paths around obstacles. Also, can be useful for locating objects of interest with algorithms like Euclidean Clustering being used to group points of the same object into one bounding box. However, in order to distinguish between objects by color, such as buoys, camera data is needed.

We currently have two different methods for identifying buoys (or other objects) with camera data. Traditionally, we have used conventional computer vision algorithms with OpenCV such as color filters and contour identifiers to distinguish objects of interest (see Fig. 4). These algorithms are currently still in use, and we have dedicated time in the past year developing a web GUI for tuning our color filter (see Fig. 5). However, this year we also explored using deep learning models like YOLO for buoy and general object detection. So far, we have found the most success using the YOLOv8 small

model for buoy detection (see Fig. 6). By competition, we hope to combine the two approaches to create a more robust buoy detection algorithm. Regardless of the detection algorithm, we map the buoys identified with camera data to objects detected by the LIDAR to determine the position of the buoys relative to the ASV.

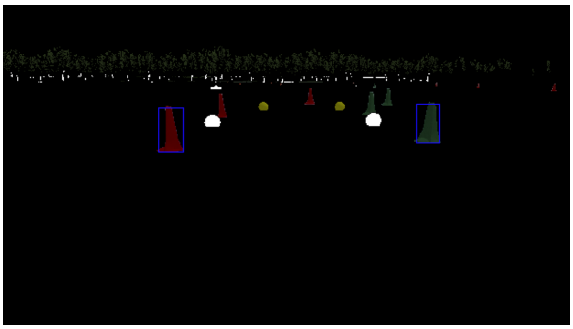

Fig. 4. Simulated color filters and contour identifiers.

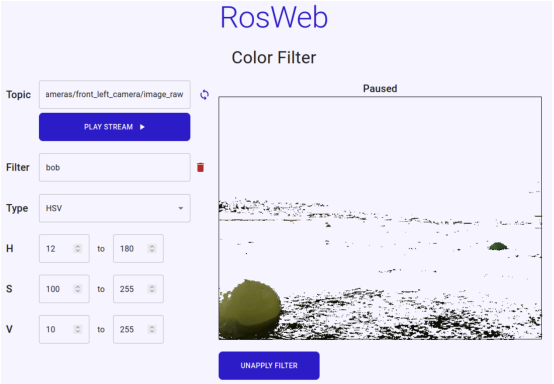

Fig. 5. GUI interface for color filter.

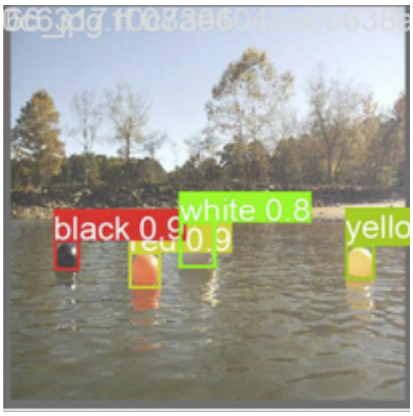

Fig. 6. Example of buoy detection on a captured image during test.

3) Navigation: A major component of our navigation subsystem involves creating a map of the environment. We create a costmap by first filtering the raw LIDAR data into voxels and then populating a costmap with high-cost areas representing areas with obstacles detected. Additionally, we add an inflation layer around obstacles so that the ASV does not plan paths too close to any obstacles. The resolution and inflation layer size of the costmap can be changed easily as well. To create a path to a goal, we have currently implemented two planners, RRT and A*. Fig. 7 shows the RRT path planner in action searching for a path to the goal.

Fig. 7. RRT path planner branching out while looking for a suitable path.

4) Controller: The controller's objective is to follow the path planned by the navigation server. The lowest layer is the motor command fuser which takes in target force goals in each axis and a target torque goal and translates these to direct motor commands using differential thrust. In the event of a change in motor configuration, this is the only layer that requires changing. A PD (proportional differential) controller is used to generate target torques to achieve the commanded heading. For distances under 2 m from the goal position, a simple PID (Proportional Integral Differential) controller based on position and velocity is used to generate target force commands. For these distances, the vehicle will maintain its goal heading and utilize holonomic translation to maintain its position. For larger distances, a PD controller is used to target a certain velocity and the vehicle's target heading is set so that it is facing in the direction of the target velocity. This velocity is generated as a sum of a vector in the direction parallel to the path and a vector perpendicular to and in the direction of the path. The further away from the path, the larger the vector perpendicular to the path is weighted. The greater the error between the vehicle's heading and target heading, the smaller the overall velocity to allow the vehicle control authority to reorient itself.

5) Firmware: The firmware links the autonomy to the physical motors on the ASV. Firmware running on our Teensy handles motor commands for moving the ASV, both from the autonomy stack and over R/C. Firmware running on our Arduino Due handles motor commands for other auxiliary mechanisms, such as arms or water shooters, on the ASV. The code is written in C++ and uses the micro-Ros library to interact with the autonomy stack.

C. Electrical Design

The boat electronics system is divided into four separate subsystems, the power system, the sensors, the controllers, and the processing and communications subsystems. Fig. 8 Fig. 9 describe the electrical subsystems.

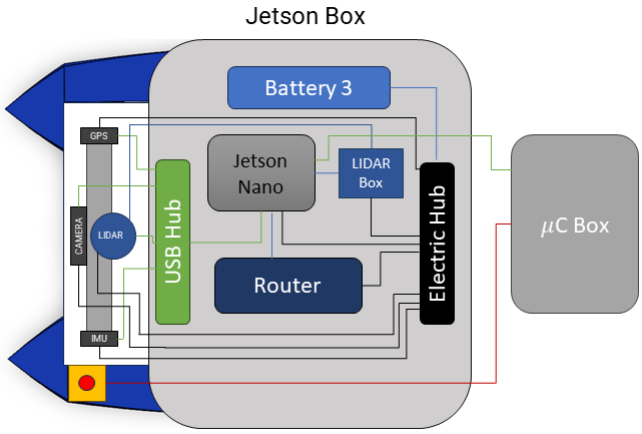

Fig. 8. Jetson Nano diagram.

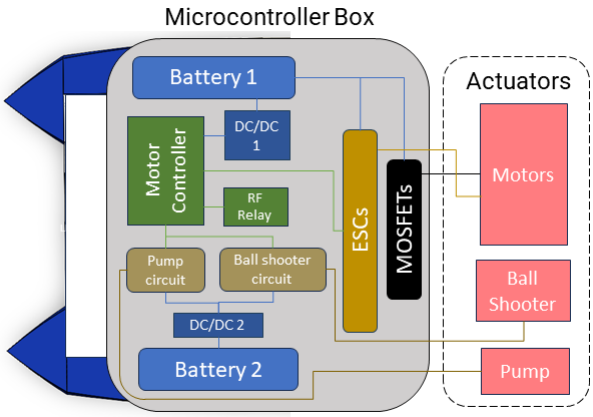

Fig. 9. Microcontroller diagram.

With many insights from the 2023 competition, the primary electrical design goals for the ASV this year revolved around improved organization of the power system, resolution of previous electrical bugs, and upgraded PCBs. Additional supplementary PCBs were added to further reduce the use of jumper wires in the electrical system. A large focus was put on strain relief, as intermittent electrical connection proved to be one of the main problems faced during field tests.

1) Improved Electrical Wiring Organization: Last year's electrical system had little organization in terms of wire routing and wire labeling. The first thing done this year was organizing the electrical devices and labeling all wires with their usage and the components to which they were connected to. This proved vital in the structuring of future improvements to the boat and generally improved quality-of-life during field tests and lab tests.

2) Resolution of Electrical Bugs: Problems were experienced when running all the electrical subsystems at once. When this edge case was tested, the ASV's main computer would lose connection to the microcontroller board. It was discovered that the problem was due to the lack of separation of controller and power ground cables. Because of the large difference in currents of these two systems (controller currents being at maximum a few milliamps and power currents remaining around tens of amps), a combined ground cable provided the opportunity for unintended behavior in the system. To solve this, a system involving optocouplers to provide physical separation of the power and control grounds was implemented. These devices allow signals to be transmitted between two circuits without a physical connection. Beyond this, the previously mentioned improved organization fixed many of the electrical bugs due to wires losing connection during testing and intermittent connection due to weak cable connectors.

3) Improved PCB Design and Additional PCB Design: One of the benefits of the organization of the devices was the possibility to redistribute and improve the hardware and technology used.

Last year's electrical board was fabricated on an in-house mill and lacked much of the refinement desired. This year major advancements were made to have the micro-controller board more resistant to modifications, as shown in Fig. 10 a. Additional headers were routed to each pin on the main microcontroller to allow for on-the-fly modifications if necessary. A highly reliable connection was made possible by the use of JST pins rather than standard header pins.

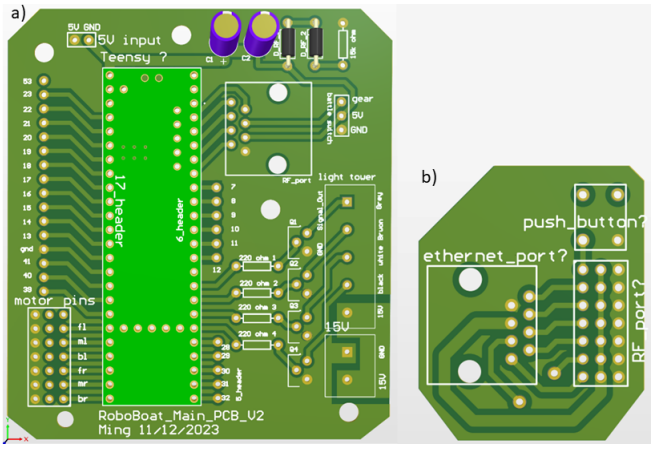

Fig. 10. 3D view of PCB BOards. a) main controller board. b) Auxiliary RF PCB board.

An additional RF PCB was created to allow for the implementation of a single ethernet cable to connect the RF receiver. This makes it possible to be placed high up without needing to run many individual cables alongside it. The PCB maps the needed pins from the receiver to an ethernet cable that is

then connected to the main motor controller board, as shown in Fig. 10 b.

IV. Testing Goals

A. Simulation

The primary goal of simulation testing was to test the perception subsystem, the navigation subsystem, the controller logic, and the coordination of subsystems during tasks. We did not focus on replicating the vehicle dynamics in simulation as the effort would not be worth the possible marginal improvements to our controller. With these goals in mind, we determined to use the Virtual RobotX simulation built with Gazebo Garden. The simulation provides us with an ASV that has configurable sensors and motors as well as common objects like buoys and docks. Additionally, Gazebo Sim integrates nicely with ROS, which our software uses. We customized the ASV to match *Crystal Clear* as closely as possible and made sure any basic changes between simulation and the physical robot, such as camera distortions or sensor heights, were easily configurable in our software. The sensors used in simulation were a 3-D LIDAR, two cameras, a GPS, and an IMU.

By using VRX's pre-built tasks or dragging buoys to create our own tasks, we were able to test the aforementioned systems without needing a field test. In Gazebo, the software was able to complete all purely navigation tasks, such as the Navigation Channel, Follow the Path, Docking, and the Speed Challenge.

B. Dry Testing

Dry tests took place a few days before lake tests. There were three main goals to dry tests: to ensure the electrical system was functioning properly, to ensure the software system and relevant hardware was working, and to ensure that the software was able to communicate with the electrical system.

Testing the electrical system involved making sure R/C communication was working as well as physical and remote e-stops.

Testing the software system involved first making sure that the LIDAR, camera, GPS, and IMU sensors were all able to send data to the computer on-board. We then checked that the software system was able to start up and launch relevant subsystems without error. Furthermore, we made sure that remote accessing the computer over the WiFi router was working.

Finally, testing communication between the software and hardware involved establishing a micro-ROS connection between the firmware and the computer. To make sure the firmware and software were connected, we would send motor commands from the software and make sure the motors on the vehicle moved correctly.

C. Field Testing

Field tests were largely done to test full system integration in competition-like environments. Tests were done at either Lake Lanier or Sweetwater Creek. Firstly, the PID controller for the ASV was tuned at lake tests as we avoided modeling the ASV's hydrodynamics in simulation. Secondly, basic operations like navigating to a GPS waypoint and station keeping were tested. Thirdly, large amounts of camera data was collected while on the water to tune and create new perception algorithms at later times. Finally, individual tasks were also attempted. We were able to replicate the Navigation Channel, the Speed Challenge, and a simpler Follow the Path task on the water.

Fig. 11. A gate as seen from the ASV's camera.

Fig. 12. A buoy channel with the ASV at Lake Lanier.

V. Acknowledgements

A. Sponsors

The Marine Robotics Group thanks all of our sponsoring organizations that have contributed to the team, whether financially, through hardware, or any other form of support demonstrated. Our corporate sponsors and supporters include Greenzie, Altium, Firefly Automatix, Ansys, and Blue Trail Engineering. Our academic sponsors include the Georgia Tech Student Government Association, the Georgia Tech Student Organization Finance Office, the Georgia Tech Student Foundation, the Aerospace Systems and Design Laboratory, and the School of Aerospace Engineering Student Advisory Council.

B. Mentors

The team thanks Carl Johnson for taking over as the primary advisor on the project, Tanya Ard-Smith for her help with competition logistics, Professor Dmitri Mavris for providing for the team at large, and Dr. Michael Steffens for his guidance early on in the project.

REFERENCES

[1] Tyler Campbell et al. “Socially Distanced Robotics RoboBoat 2021-Georgia Tech”. In: (2021).

[2] Shawn Coutinho, Manuel Roglan, Aaron Wu, et al. “Crystal Clear: RoboBoat 2023: Technical Design Report”. In: (2023).

APPENDIX A
COMPONENT LIST

Component	Vendor	Model/Type	Specs	Custom/Purchased	Cost	Year of Purchase
ASV Hull Form / Platform	In-House	Dual Pontoon	4'x3' footprint	Custom	$300	2020-2023
Waterproof Connectors	Huayi-Fada Technologies LTD.	Various	IP68	Purchased	Unknown	2015
Propulsion	Blue Robotics	T200 Thruster	T200 Website	Purchased	$200	2016
Power System	Multistar	4S 12C LiPo	4S1P, 14.8V, 10Ah	Purchased	$120	2023
Motor Controls	Teensy	Teensy 4.1	Teensy 4.1 Specs	Purchase	$31.50	2023
CPU Orin Nano Specs	Nvidia Purchased	Orin Nano $500	 2023	limit=9	locale=en-us##	
Teleoperation	OrangeRx Spektrum	R615X DX6e transmitter	3.7-9.6 V; 2.4 GHz	Purchased	$17	2016
Compass	ArduSimple	ArduSimple RTK	Compass Specs	Donated	$619	2020
Inertial Measurement Unit (IMU)	LORD MicroStrain	3DM-GX3-25	IMU Specs	Purchased	$2640	2017
Camera	Playstation	Eye	640 x 480 60 fps; USB	Purchased	$43.99	2023
Stereo Camera	Luxonis	Oak-D	Oak-D Specs		Purchased	$249
LIDAR	Velodyne	VLP-16	VLP-16 User Manual	Donated	$4000	2023

APPENDIX B
CAD OF ASV

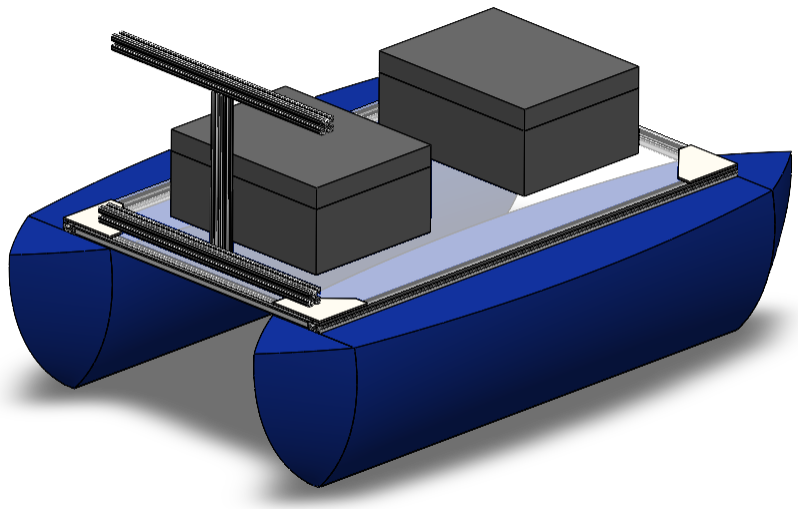

Fig. 13. 3D view of boat

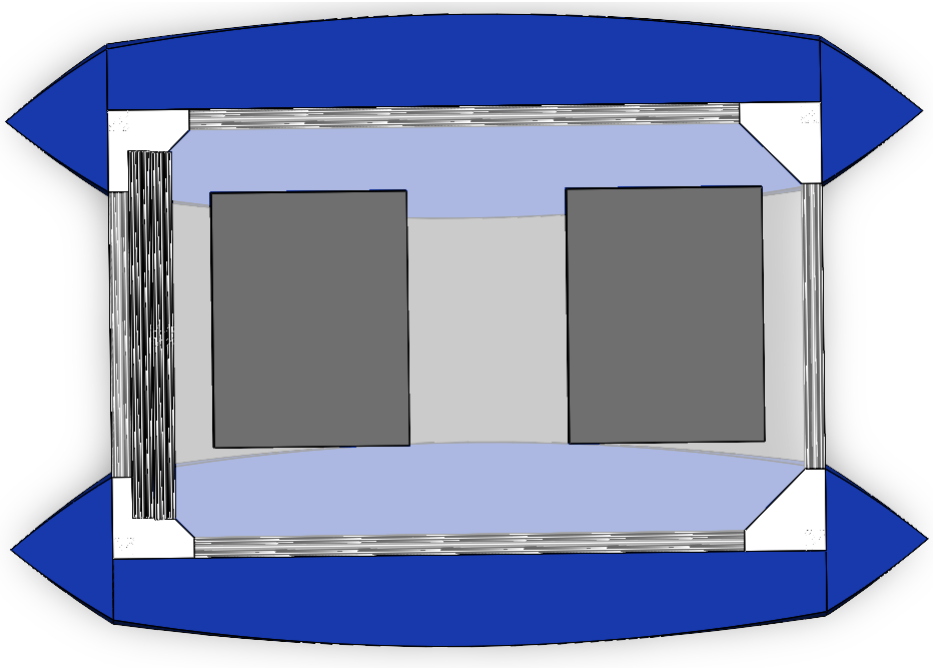

Fig. 14. Top view of boat

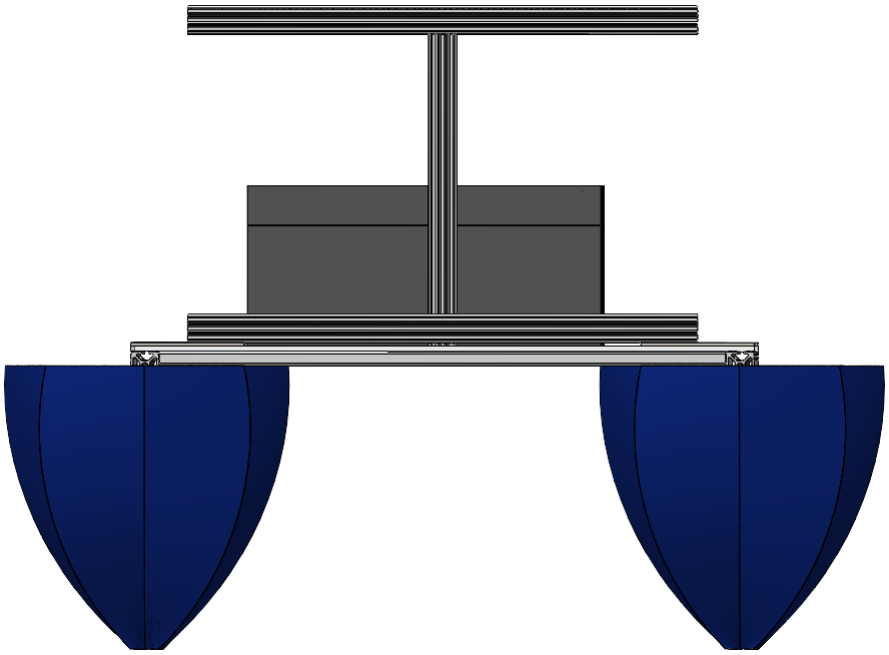

Fig. 15. Front view of boat

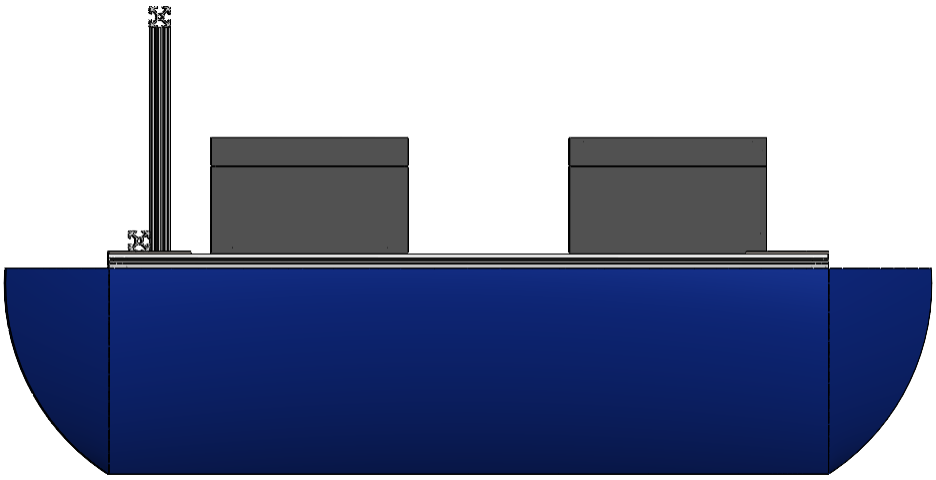

Fig. 16. Side view of boat