Roboboat 2019: Technical Design Report

Vinayak Ruia, Robert Kuramshin, Daniel Foreman, Sean Fish, Eric Fu, Patrick Meyer
Georgia Tech Marine Robotics Group, Georgia Institute of Technology
Atlanta, Georgia, United States

***Abstract*—This technical design report details the design and development of the autonomous surface vehicle (ASV) to be entered in the 2019 Roboboat competition. The primary change in the system from last year is the switch to Robot Operating System (ROS). The transition to a ROS based system will result in shorter learning curves for new team members and an opportunity to refine our software development practices. The overall competition strategy involves focusing our software development efforts on the key skills shared between multiple tasks. Additionally, hardware tweaks were made to ensure reliability of the platform and longevity of the system for future team members based on observations made and feedback received during earlier competitions. The most notable hardware change involves the design decision to a lower capacity Li-Po battery system that dramatically reduces the weight of the vehicle,potentially increasing weight points awarded in the competition. As this is a large rebuilding year for the team, we are excited to participate with our majorly revamped software stack and refined hardware platform.**

I. Introduction

The Georgia Tech Marine Robotics Group has been involved in the maritime Robonation challenges for many years, but the team membership, the vehicles, and the software stack is constantly changing and evolving.

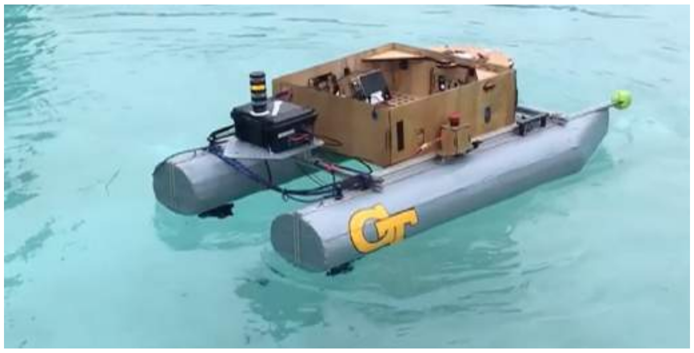

Fig. 1. The Roboboat 2019 platform, significantly lighter than the 2017 / 2018 platform

While the Roboboat hardware platform, shown in Figure 1, has stayed relatively stable since the Roboboat 2017 competition, the Roboboat 2019 entry will include major software changes. Before the RobotX 2018 competition, the team decided to embark on the journey to replace a previously developed custom software application for simulation and control. The team decided that a switch to a Robot Operating System (ROS) based software architecture will allow for better collaboration between teams both internal and external to Georgia Tech. As the team's composition is shifting towards more undergraduate participation, the switch to ROS can be seen as a short term sacrifice in order to strengthen the long-term team's understanding of the development of software for autonomous maritime systems. The readily available documentation, and open-source nature of the ROS platform will ease the learning curve for newer members [1].

The remainder of this document is as follows: the software development strategy and design priorities are further outlined in Section II: Competition Strategy. Further, in Section III: Design Creativity, certain system and subsystem level decisions will be detailed. Finally, Section IV: Experimental Results will describe the simulation and in-water testing that has been done so far. It will also describe the further experimentation that will be needed in order to gain confidence in the new Roboboat 2019 system before the competition this June.

II. Competition Strategy

A. Development Strategy

As mentioned before, the Georgia Tech Marine Robotics Group has opted to deprecate an older Georgia Tech developed software stack, ARCS [2], for simulation and control of the Roboboat platform in favor of using ROS based packages. In January 2018, the Marine Robotics Group performed a field test with an incoming batch of new team members where the Robosub team would develop their autonomy on a ROS based stack and the Roboboat team would develop their autonomy in ARCS. It was observed that the development learning curve for ARCS based development was significantly steeper than the ROS development due to the available documentation and the pre-existing ROS community. By August 2018, it was decided that ROS based development would become the new standard for the teams participation in RobotX, Robosub, and Roboboat due to the difference in learning curve.

The team did not completely abandon the old ARCS software, but instead wanted to develop the ROS based system with the lessons learned during the development of ARCS in mind. ARCS did a few things well. It had an easy to use GUI/mission manager, the ability to switch between vehicles without disruption, and the ability to switch between modules with ease.

For example, the software was developed so that it was very easy to switch an A* based path planner with a Djikstra's based path planner for different missions. The ARCs GUI also made it very easy for a user to input rough estimates of course layouts, such as in Figure 2, and simulate different course configurations. The team believes that with intentional development to take advantages of the ROS subscriber/publisher

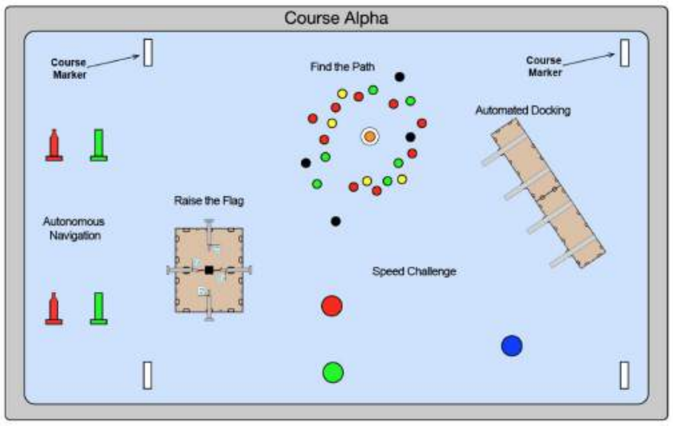

Fig. 2. From a competition perspective, it is important to have a GUI that will allow for users to input course estimates, such as Course Alpha depicted above, could be readily inputted by an end user.

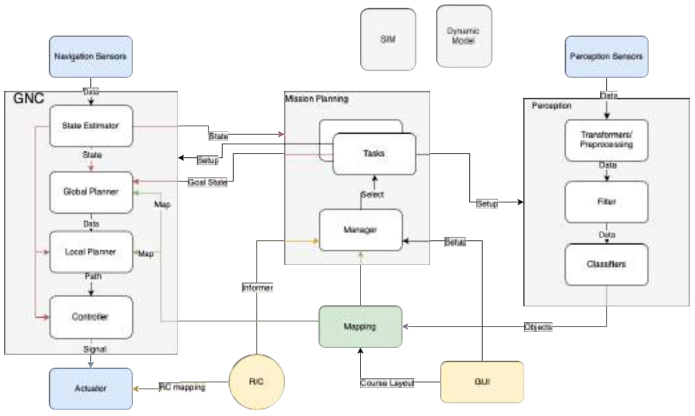

Fig. 3. The full diagram of how the Marine Robotics Group ROS architecture is structured, with the arrows representing data flow, and the boxes representing software modules

node philosophy we can learn and grow from the previous development done in ARCS.

The full system architecture conceived is shown in Figure 3. While a detailed explanation for each component described is outside the scope of this paper, the architecture can be broadly split into Guidance, Navigation, and Control (GNC); Mission Planning; and Perception. The development of this system architecture built upon the team's experience developing ARCS.

Although we aim for excellence in all facets of our software stack, full development and mastery of every module shown in Figure 3 is not necessarily needed to complete the tasks, nor is it practical from a project management perspective. As we rebuild our software capabilities, it is important to note which aspects of the autonomous system are crucial to overall performance, and which ones are superfluous to the primary objectives.

Figure 4 depicts the core functions the vehicle must have to perform well on all in-water tasks in the competition. It is arranged so that left to right it also represents the hierarchy of needs for the competition. It is most important that the left

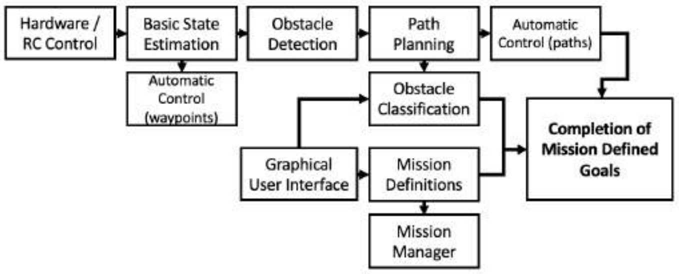

Fig. 4. The 'hierarchy of needs' for the Roboboat 2019 competition, with the origin of the arrow representing the **prerequisite** core skill

most box (having all the boat hardware, thrusters, and sensors) work before the team spends valuable time and resources on other software capabilities. Similarly, it is important that the automatic control to waypoints is developed before automatic control to paths are developed.

Figure 4 also represents our time line during this rebuilding of the software stack. A design decision was made to develop the minimum viable product - the simplest version of the platform that meets the set forth requirements. For example, the current vehicle 'state estimator' is simply a low pass filter on both IMU and GPS data, projected on to a local coordinate grid. Due to the large quantity of software development needed to be ready for the competition, the team took a 'breadth but not depth' approach to developing the key pieces of software shared by all the tasks. With functional navigation, controllability, and obstacle detection/classification, the team is leaving the development of the specific task descriptions for the weeks to come. The minimum viable product approach was an intentional project management made given time and resource constraints the team faces.

B. In-water Competition Strategy

Last year, during the 2018 competition, The team had missed first place by a margin of 700 points. Therefore, that value became the goal for improvement in points made using simple hardware changes. First off, the team has shed the 50 lb, 110 Ah Li-Po battery that was used on the 2017 and 2018 Roboboat platforms in favor of two 10 Ah, 2 lb Li-Pos, yielding roughly 250 points for the team. The point contributions due to dry weight and due to 'thrust to weight ratio' are estimated below for both last year and this year, where w represents the weight of the vehicle in pounds, and t represents the thrust of the vehicle, in pounds.

$$\begin{aligned}
2018: w &\simeq 115 \text{ lbs} \\
t &\simeq 14 \text{ lbs} \\
\text{If } w &> 110: \\
-250 - 5 * (w - 110) &= -275 \text{ points} \\
100 * (t/w) &= 12.7 \text{ points}
\end{aligned}$$

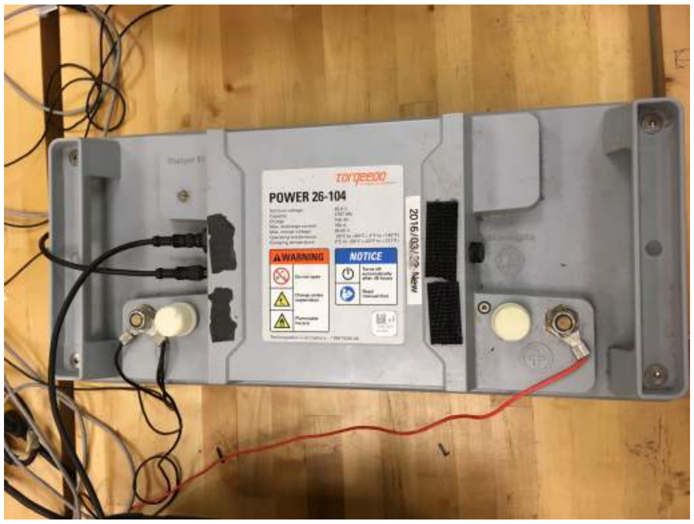

Fig. 5. The Torqueedo Power 26-104, the over capacity 50 lb battery used in the Roboboat 2017 and 2018 competitions, however, yielded around 5 hours of drive time at max thruster power draw

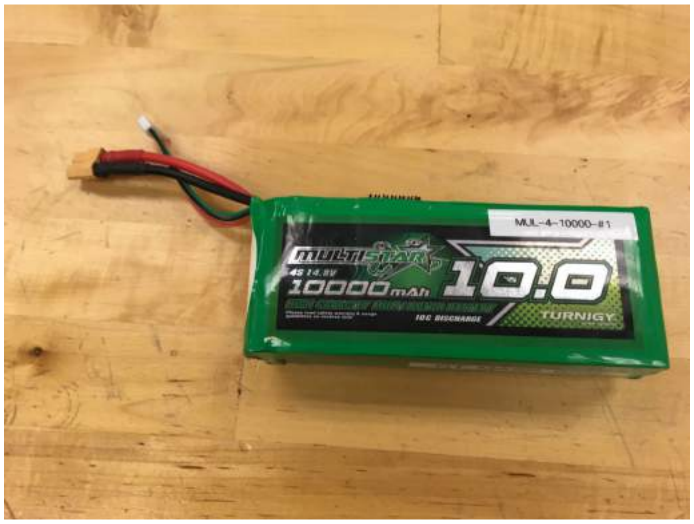

Fig. 6. The new 10 Ah, 20C LiPo that will gain the team many points due to difference in weight alone, however, will only yield around 1/2 hour of drive time at maximum thruster power draw

$$\begin{aligned}
2019: w &\simeq 60 \text{ lbs} \\
t &\simeq 14 \text{ lbs} \\
\text{If } w &\leq 70: \\
80 + (70 - w) &= 90 \text{ points} \\
100 * (t/w) &= 23 \text{ points}
\end{aligned}$$

Originally, the 2017 and 2018 overcapacity Li-Po (Figure 5) was chosen to reduce the logistical complexity of charging the battery during the week of competition. In an eye opening moment during the 2018 competition, the team did not initially qualify for semi-finals even after better perceived autonomous performance than some rival teams. The team later realized that this was due to the points awarded for vehicle weight. The team believes that the switch to lighter RC aircraft style Li-Pos, depicted in Figure 6, is justified. This change yields the team 250 points, leaving just 500 points out of the 700 point deficit that the team aimed to make up.

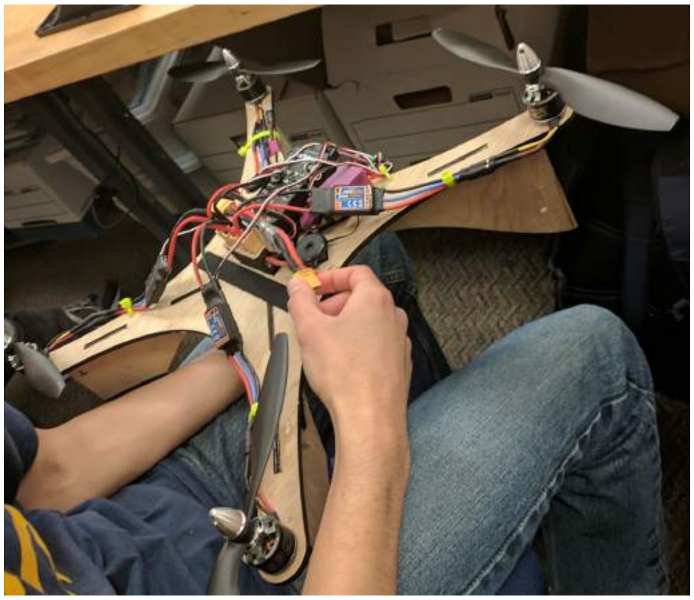

Fig. 7. A version of the UAV that will be brought to the competition, based on a design that was originally designed to be laser-cut and assembled during a high school summer camp hosted by Georgia Tech

Another hardware decision the team made was inclusion of the UAV as a minimum on the vehicle. Ever since it's introduction, the UAV has been an afterthought for the Marine Robotics Group as it is only used in one competition task. This year it was determined that the points gained solely for an autonomous launch of the AUV, an estimated 250 points [1], justifies the development of the drone system to, at a minimum, launch autonomously through ROS. Of course, the team would like to legitimately try to conquer the computer vision and communication challenges introduced by full completion of the Raise the Flag Challenge, but again, the minimum viable product first approach applies in this situation as well. Based on our progress with the UAV development this year, we are fairly confident that atleast the UAV will be launched autonomously during some of our competition runs, and an early version is depicted in Figure 7.

In terms of the remainder of the on-water tasks, the team strategy remains the same as it has been in years past. The team is confident that the key skills developed such as object classification and automatic control are sufficient. The development challenge that remains before the competition involves porting over task descriptions for the on-water tasks into our ROS based architecture. Tasks such as automated docking using hydrophones are perceived as extremely difficult in proportion to the points awarded for successful capability, and will be avoided in lieu of tasks that can performed reliably

[1]This estimate does not include the points lost due to the increased weight of the system due to the UAV, as the UAV weight has not been fully determined yet

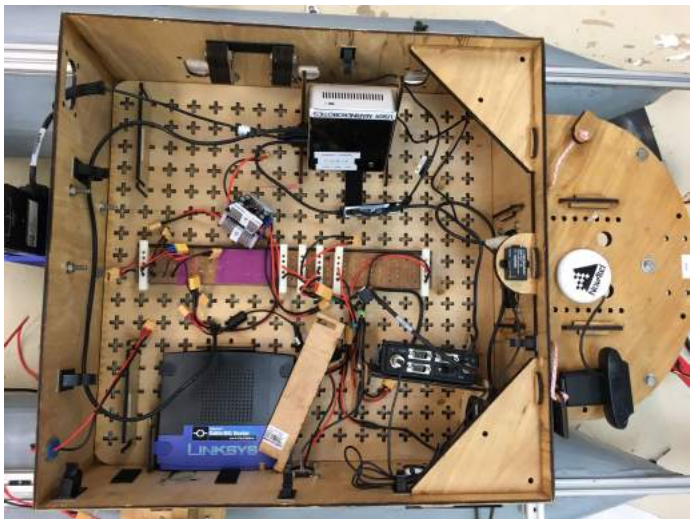
Fig. 8. A Pegboard design electronics casing allows for flexible repositioning of every component, reducing week-of-competition stress

using the core skills shared by all tasks. Tasks such as the Mandatory Navigation Gate and Speed Challenge share many characteristics. Tasks such as Automated Docking, and Raise the Flag share a lot in common as well (in that the ASV interfaces with non-buoy objects, and contact with an object must be detected). The last task, the Find the Path challenge is seen as the most difficult challenge, and requires mastery of automatic control, perception, and path planning.

III. Design Creativity

A. Modular Hardware

One of the most creative aspects at the system level of the Roboboat platform since 2017 remains the modular design of the vehicle. This became an important design goal for the platform as the team has realized that a lot of time is wasted during the week of competition making hardware changes. For example, the pegboard, allows for a wide range of sensor positioning within the electronics casing as shown in Figure 8. The electronics casing also contains a central power rail which offers a simple robust plug-in solution to powering all onboard electronics. Even the pontoons of the catamaran platform have 80/20 aluminum extrusions embedded within the pontoon, both on the top and the bottom so that mounts and frames can easily be replaced and modified.

The team benefited from this design aspect when the motor mounts on the bottom of the pontoon broke during the 2018 competition, and the team was able to fashion an alternate mount, Figure 9, using only parts from the local hardware store, instead of having to re-epoxy a permanently attached mount as some designs may require a team to do.

B. Perception: An example of minimum viable product

The design of the perception system is selected as a creative approach to the perception challenge of both recognizing in-water objects, and then classifying them. In order to avoid

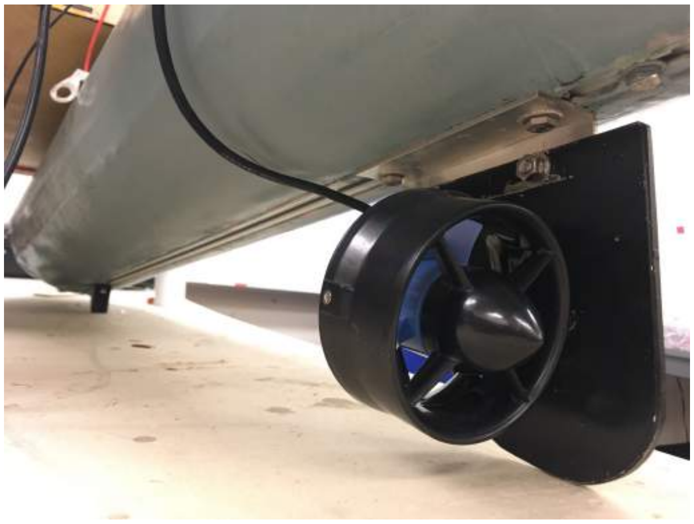
Fig. 9. The mount that was fashioned in the manner of an afternoon during the week of competition in 2018. The team greatly appreciated the 80/20 extrusions when replacing the previously broken mount

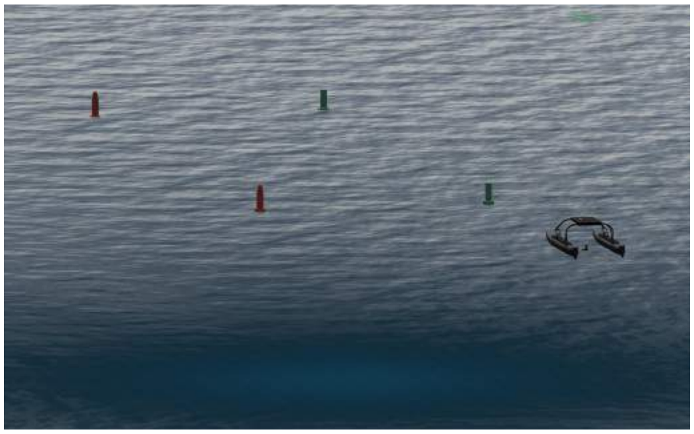
Fig. 10. The Gazebo/VRX simulation [3] environment for the mandatory navigation gate challenge. While a WAM-V is simulated instead of the Roboboat, the simulation environment is sufficient to model sensors and ASV dynamics

unnecessary complexities of techniques such as camera-based computer vision, or machine learning based LiDAR object classification, a heuristic approach was developed to achieve the set out competition goals. It is important to note that the heuristic that will likely not scale to applications beyond ASVs in the Roboboat competition, but this approach can later be supplemented with machine vision techniques that will use when the ASV encounters more complex environments.

Take for example, the simulation environment shown in Figure 10, where a simulated 3D LiDaR, much like the on-board Velodyne VLP-16, generates a point cloud. Since, in real life, only in-water obstacles will produce significant and/or dense 3D LiDAR returns, a DBSCAN based approach is perfectly sufficient for recognizing course obstacles without misclassifying unrelated terrain features. This is shown in Figure 11. The parameters for appropriate size of a cluster,

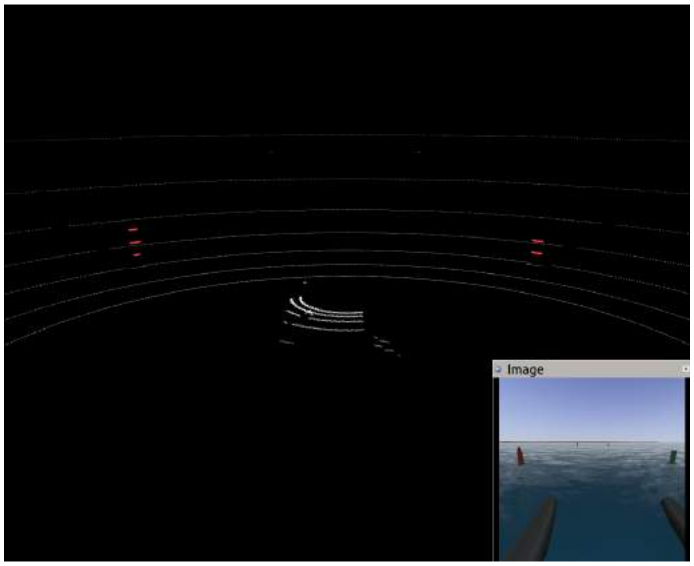

Fig. 11. The LiDAR point returns for the environment shown in white, where the DBSCAN output (points that belong to a cluster), shown in red

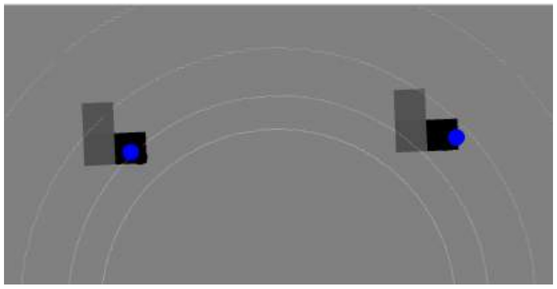

Fig. 12. A top down view of the occupancy grid generated that will be used for path planning, with darker colors representing a higher probability an object occupies that space. This is updated based on the amount of times an object has been in that grid space, and this probability either grows or decays with time. The blue dots represent the 'can buoy' objects output after the points clustering occurs

and the approximate density of a cluster are inputs to the DBSCAN clustering algorithm [4]. Once these clusters are perceived, they are used to populate an occupancy grid, shown in Figure 12. Additionally, the clusters are checked to see if the height to width ratio matches that of either a dock, can buoy, or circular buoy. Finally, once the cluster is associated with an object type, its position will be matched by the vehicle's a priori knowledge of the rough course layout and the vehicles state estimate to determine what competition task the object belongs to.

As described, the perception stack is a good example of something that was done in the simplest possible manner for quick implementation in the competition, however, this structure leaves a lot of room to include more complicated techniques (such as webcam input to perceive color) later on to supplement this base-level technique. The bare-bones approach taken in the perception stack extends throughout the redesigned ROS system, as a project management decision to balance project scope and project resources.

IV. Experimental Results

In-water testing time is a luxury in Atlanta, a land-locked city. The nearest appropriate testing site, Sweetwater Creek Lake Park, is almost a 40 minute drive away. Therefore, the team was only able to make two trips to test in the water. In-water testing time is enjoyable and gives the team a chance to showcase our vehicle to the public but in-water testing also requires a lot of logistical planning and valuable weekend time. While experimentation opportunities such as simulation using the VRX pacakge for Gazebo and lab bench testing were taken advantage of whenever possible, there are no perfect substitutes for in-water lake testing. To date, the in-water testing time involved validating the full hardware stack (vehicle, thrusters, and sensor packages), as well as tuning the automatic controller to way point control. These are validations that must be performed in the water, as they are difficult to simulate (the dynamic model of the vehicle has not been captured in the ROS architecture, and would not have been sufficient to tune the controller). Unfortunately, these two in-water sessions give the 2019 team of only 8 hours of valuable in-water testing hours to date. More real in-water testing time is planned prior to the 2019 competition, where the team aims to further the tuning of the controller, and completion of the mandatory Navigation Gate task before beginning travel to Daytona Beach.

V. Conclusion

When the team first made the decision to revamp the software stack, it seemed like an extraordinary amount of work. It was a major undertaking to make this transition in a manner of a few months, but by using the minimum viable product approach the team was able to assemble a system with our limited resources. Now we strongly believe that a ROS based platform and a fundamentals-first approach will prove to be appropriate for the 2019 competition and for future iterations of the team. We are extremely excited to put our hard work to the ultimate test this June during the competition.

Acknowledgment

The Georgia Tech Marine Robotics Group thanks the Aerospace Systems and Design Laboratory (ASDL) for supporting the teams work. We also thank The Hive, a student run maker space at Georgia Tech, for providing guidance and manufacturing support for some components used on the vehicle. We are also appreciative of the funding received by the Georgia Tech Student Government Association. Lastly, we would like to acknowledge all current and past team members of the ADePT Lab and Marine Robotics Group. Everyone's hard work and dedication is greatly appreciated.

References

[1] ROS.org — About ROS", Ros.org, 2019. [Online]. Available: http://www.ros.org/about-ros/. [Accessed: 27- May- 2019].
[2] S. Seifert, P. Meyer, C. Ramee, E. Evans, W. Roberts, K. Griendling, and D. Mavris, Arcs: A unified enironment for autonomous robot control and simulation, in OCEANS 2017 MTS/IEEE Anchorage, IEEE, 2017.
[3] C. Aguero and B. Bingham, VRX https://bitbucket.org/osrf/vrx, 2018.

[4] ”How DBSCAN works and why should we use it?”, Towards Data Science, 2019. [Online]. Available: https://towardsdatascience.com/how-dbscan-works-and-why-should-i-use-it-443b4a191c80. [Accessed: 27- May- 2019].

Appendix A

Component	Vendor	Model/Type	Specs	Cost (if new)
ASV Hull form/platform	Self Developed	N/A	3100 in^3 in volume per , pontoon, ~4.5 feet long	~$120
Waterproof connectors	Huayi-Fada Technologies LTD.	Varies	IP68	Varies
Propulsion	Blue Robotics	T200	https://www.bluerobotics.com/store/thrusters/t200-thruster	$169
Power system	Multistar	4S 12C LiPo	4S1P, 14.8V, 10Ah	$60
Motor controls	Maytech Innovation	MTDU30A	30A	$99
CPU	Intel	NUC5i7	https://ark.intel.com/products/87570/Intel-NUC-Kit-NUC5i7	$540
Teleoperation	Persistent Systems	Wave Relay MPU5	https://www.persistentsystems.com/mpu5-specs/	Unknown
Compass	Novatel	FlexPAK 6	https://www.novatel.com/products/gnss-receivers/enclosures/	Unknown
Inertial Measurement Unit (IMU)	Microstrain	3DM-GX4-25	https://www.microstrain.com/inertial/3dm-gx4-25	$2,640
Doppler Velocity Logger (DVL)	N/A	N/A	N/A	N/A
Camera(s)	Velodyne LIDAR	VLP-16	https://velodynelidar.com/vlp-16.html	$8,800
Hydrophones	Teledyne Marine	Reson TC 4013	https://tinyurl.com/TeledyneReson	Unknown
Aerial vehicle platform	Internally Developed	N/A	N/A	<$20
Motor and propellers	Mutltistar	Multi-Rotor Motor	V-Spec 1104-3600KV	$13
Power system	Rhino	2S 20C Lipo Pack w/XT60	2S 20C, 7.4V	$7
Motor controls	HobbyKing	Brushless ESC	30A UBEC	$11
CPU	Raspberry Pi	3, Model B	4 x ARM Cortex-A53, 1GB DDR3	$35
Camera(s)	To be determined	To be determined	To be determined	To be determined
Autopilot	PixHawk	Version 4	https://docs.px4.io/en/flight_controller/pixhawk4.html	$180
Algorithms	Internally Developed			
Vision	Internally Developed			
Acoustics	Internally Developed			
Localization and mapping	Internally Developed			
Autonomy	Internally Developed			
Team Size (number of people)	10			
Expertise ratio (hardware vs. software)	1:2			
Testing time: simulation	12+ hours			
Testing time: in-water	8+ hours			
Inter-vehicle communication				
Programming Language(s)	Python, ROS, Arduino/C++			