The Design of Team Inspiration's 2021 AUVs

Ashiria Goel (team lead), Rishi Veerepalli (deputy), Eesh Vij (deputy), Maxime Ellerbach, Isabelle Gunawan, Aditya Mavalankar, Shruti Natala, Ashika Palacharla, Pratham Saxena, Raina Shapur, Pahel Srivastava, Colin Szeto, Mabel Szeto, Noah Tang, and Claire Zhao

Abstract*—Team Inspiration focuses on perpetually learning and improving. Our third-year team of 15 middle and high schoolers designed our 2 Autonomous Underwater Vehicles (AUVs), Onyx and Græy (modified versions of our previous AUVs), for the 2021 RoboSub competition. In designing Onyx and constructing Græy, we learned to use industry standard practices for design, data analysis, and design of printed circuit boards (PCBs). Through the continuation of COVID-19, our team learned effective virtual collaboration, distribution of tasks, and parallel development. Our strategy of full environment simulation allowed learning of Unity and integration and testing of the whole system. Recognizing the importance of having a realistic simulator for RoboSub, our primary goal was to open this project to other teams around the world for use and collaboration on further development, in preparation for 2022 and future competitions.

I. Competition Strategy

Last year, we aimed to perform in the top third and exceeded our expectations by ranking 1st overall. The systems engineering process, doctrine of K.I.S.S. (keep it simple, silly), and the Agile/sprint process were integral in our success last year, so we continued using these processes to drive our team. To expand our knowledge and optimize our AUV, we designed custom hardware, software, and electrical components while buying commercial off the shelf parts to supplement as needed. Besides updating and building our physical AUVs, we heavily focused on simulation; we worked to recreate the Transducer Evaluation Center (TRANSDEC) competition environment in Unity to be able to integrate and test our systems in an environment similar to the competition. Furthermore, we wanted to create a collaborative environment that RoboSub teams from around the world could utilize and build upon. Simultaneously, we continued to develop our two AUVs to verify the simulator and transfer our efforts to real life.

In our efforts to create a solid base for 2022 future competitions, we continued a strategy to compete with two AUVs because of the advantages provided by the rules. For example, using two AUVs nearly doubles our time underwater, as the run only ends when both AUVs have surfaced. Also, only the highest points earned at each attempted task will be counted. In addition, we can gain points through the intersub communication task. We deemed that the point benefits of running two AUVs far outweigh the cons presented of having weight penalties. We focused on planning inter-sub communication, which we started learning about last year, as the main component of our mission planner because we wanted to optimize our scoring capabilities between the two AUVs. We aimed to create a set of AUVs that, when combined, could complete all the competition tasks. Accordingly, we designed both AUVs with essential sensors such as cameras, while we installed specialized components like hydrophones on only one AUV.

As per our plan for the next in-person competition, both AUVs will pass through the gate and spin to earn the style points. Additionally, both AUVs will be equipped with downward-facing cameras used to follow the orange path. Græy will hit the buoys, drop the team markers, and complete the bin tasks. Meanwhile, Onyx will launch torpedoes, surface in the octagon, and utilize the hydrophones to locate the random pinger that marks the missions. Through inter-sub communication, both AUVs will approach missions looking for the same image set (G-men or Bootleggers). To accomplish the missions, the AUVs rely heavily on our program's ability to identify and classify objects, as well as navigating to the missions. Therefore, we also focused on improving the AUV's software system to fulfill these goals. The

programs on the AUV's main computer serve a variety of purposes, including localization, perception, mission planning/execution, and sensor data fusion. Computer Vision (CV) is used to identify the missions and differentiate whether the tasks correspond to the G-man or Bootlegger. We enhanced our vision capabilities by focusing on integration and increased volume of testing through simulation to refine its accuracy, creating a realistic testing tool for the RoboSub community. To enhance navigational accuracy, we added a Micro Electro-Mechanical Systems Altitude and Heading Reference System (MEMS AHRS) sensor and a 3D imaging sonar (loaned to us by our sponsors). During mission execution, sensors are compared against each other to filter out inconsistent data. Additionally, to gain a deeper understanding of our robot's navigational system, we decided to design our own flight controller; the development process was aided by the simulator we developed. We use Robot Operating System (ROS) to enable interprocess communication between the programs for a modular software system.

We updated Onyx and Græy's electronics and enclosure holders, staying true to our simple and modular design. The design allows for alterations by increasing the cylinder diameter and length to accommodate for the increased amount of electrical components and ports for added equipment.

II. Design Creativity

A. Unity Simulation

As a team, we realized that a realistic simulator is paramount to any team's success, including our own, and we developed a vision of sharing our project with other teams. A realistic simulation can provide a proof of concept and design, while illustrating a real demonstration of a system in a virtual environment. With the redesign of our control system, we decided to move away from Gazebo, a 3D simulator we used in 2020, and started developing a new platform in Unity, a cross-platform game engine that allowed more flexibility and low-level control, as well as the realism we wanted.

We chose Unity for its multi-functional tools and simplified process to develop photorealism. The Unity platform contains an adjustable lighting system, high resolution imaging, and high definition assets, materials, and textures. Since we were unable to access the competition pool throughout the season, Unity best allowed us to test our system in the competition setting, including the game missions and elements. A simulation environment with accurate physics gives the software team room to prototype in parallel and evaluate the accuracy of their programs.

We have opened this platform to the RoboSub community to allow teams from around the world to test their systems without physical dependencies, while improving the realism of the simulation environment. Additionally, we are working to redesign our flight control system from the ground up, including designing the electrical fragment and rewriting the software, without any previously written software. Our goal is to create the RoboSub version of the VRX system utilized in the RobotX competition [1]. This will allow teams to test software before the hardware is complete, using our default design as a baseline for testing in the simulation.

Our development of the TRANSDEC environment in Unity was a documented, collaborative process. At the beginning of the season, team members followed tutorials to learn to use Unity. Afterwards, we had team members create and add 1:1 3D models of the competition environment to our cloud-based Unity Teams repository. Our assets include the competition mission models and sensors on our AUVs, programmed to output functional data. Simultaneously, team members also worked on developing the greater virtual environment. We discovered a TRANSDEC Pool environment by Wrocław University of Science and Technology [2] and modified its scenes to fit our requirements. The final assets of the game elements were transferred to the TRANSDEC Unity project and set in their respective locations in the pool, according to the official guidelines.

The team was introduced to the SDSandbox, a Self Driving Car sandbox that uses a Unity 3D game engine to simulate car physics in the 3D world. We received access to the DonkeyCar simulation, where SDSandbox is utilized, and tested its primary functions before using some of the communication scripts for our own project.

As the project grew, we moved to GitHub to collaborate more seamlessly. Additionally, the GitHub repository is publicly available, which allows teams from around the world to engage and collaborate on the development of our simulation tool.

Our simulator works with one server and two clients. The clients represent a flight controller and a computer processor that would be found on a physical AUV. The processor client is simulated through a ROS node that receives data from all sensors. The flight controller client processes data and sends thruster commands that translate to AUV movement.

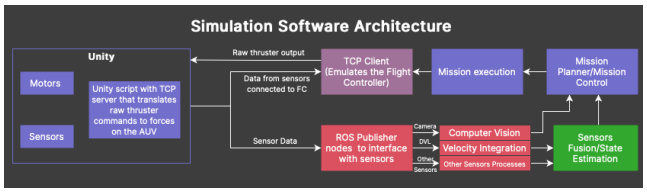

Fig. 1. *The simulation architecture, along with the control code, on our AUV.*

We used functions and Game Objects in Unity to send and receive data for the sensors that we chose to simulate, including the camera, Inertial Measurement Unit (IMU), Doppler Velocity Log (DVL), and sonar. For example, to simulate the camera stream, a camera object inside the Unity environment is exported and accessible through an external interface. Similarly, the sonar script, which is linked to the 3D model, functions by using a raycast which points from the sonar out into space. Then, when this raycast hits another object, it returns the distance of that object and therefore provides the distance from the AUV to another object. We then broadcast the sensor data using the User Datagram Protocol (UDP) client. We also use a Transmission Control Protocol (TCP) server to handle the controls of the AUV, through the upward, forward, pitch, roll, and yaw forces.

We have compiled a comprehensive guide present on our website and simulation GitHub [3], for teams to learn how to use Unity as a whole and on our specific project. This includes instructions to add a custom AUV, assets, and sensors into our environment. As we expand on this guide, we hope to include information about all areas of RoboSub, creating documentation similar to Game Manual 0 [4] used in FIRST Tech Challenge. We hope that other RoboSub teams will find this useful and join us to improve this simulation platform.

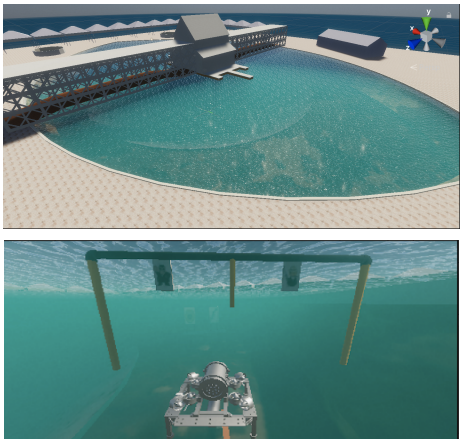

Fig. 2. *The current version of our RoboSub simulation includes a model of the underwater missions and our AUV.*

B. Mechanical

To foster unique designs that met requirements we established at the beginning of the season, our team conducted Preliminary Design Reviews (PDR) and used systems engineering. These unique requirements were all centered around the factor of modularity, maintainability, simplicity, durability, and scalability—essential parts of assembly and our team's core philosophy as a whole. Below are some specific details about Onyx that fulfilled our requirements throughout the development process.

Much of Onyx is based on Græy, and much of Græy is based on Orange (our 2019-2020 AUV), so we continued to use 80/20 extrusions for the base construction of the frame. Utilizing 80/20 extrusions provides us with better modularity with respect to the construction of our AUVs. While 80/20 extrusions may not be as compact as alternate options, like CNC waterjet

parts, its quick assembly and cost efficiency allowed us to deliver within a tight schedule. Furthermore, this system's modularity allowed us to rapidly make crucially needed modifications. For example, we fixed an unequally distributed buoyancy on Græy by shifting the positions of its two enclosures. With the 80/20 build system, we can use our manufacturing method of 3D printing to create accurately and properly toleranced parts that interface well.

We understood our manufacturing abilities and designed our enclosure holders accordingly in multiple pieces to fit on our compact 3D printers. We also decided to manufacture our thruster mounts from laser cut delrin, due to its lightweight and durable nature. This would be more effective than our FDM manufacturing method.

This season, we wanted our electronics enclosure to be maintainable, simple, durable, and scalable. We were able to fulfill these requirements by following the systems engineering process. Specifically, when we were deciding how to arrange the electronics, we used a decision matrix, picking our final design based on how maintainable and modular the design was. We picked a design in which there are different plates stacked on top of each other, with each able to slide out to access the electronics.

C. Electrical

Compared to our first year AUV, Orange, Græy far superseded it in its capabilities; Græy had greater mobility, more processing power, and more installed sensors. Still, there was still much more to be desired. In theory, Græy was a high-performing AUV, but while manufacturing Græy's electronics, we found several major oversights.

First and foremost was the amount of space the wiring took within the enclosure. From power to signal cables, wires were strewn about the enclosure forcing the removal of the Hydrophone system in order to keep other core navigational components.

Along with the issue of space, maintenance became nearly impossible due to the one-way method in which some wires were connected. In concept, the idea of Græy was to have four slide-in plates which could be individually removed. However, with the several wires traveling in-between plates, this system was no longer feasible and thus the removal of the entire enclosure was required for maintenance.

Onyx, our latest AUV, addresses these issues by incorporating a much larger 8 inch diameter by 24 inch length enclosure. This extra space allows us to place several components in more accessible locations, and also helps create dedicated wire paths within plates to allow for easier connections.

Since we are still limited by the 8 inch diameter of the enclosure, we decided to have only three component layers instead of four for ease of access. This is possible due to the increased enclosure length which can now house all the electronics without the need of 4 layer stacking. This will also increase the ability of heat dissipation within the enclosure. We further managed the heat through various heatsinks and fans which will route hot air toward the aluminum end caps. The heat conductivity characteristics of the aluminum will thus remove the directed heat from the system into the water surrounding the enclosure. We also plan to have a copper or heat-absorbing plate running through the enclosure, connected to the end plates, to dissipate heat in a larger volume. Further, we isolated the noise from our more sensitive electronics through our electronics placement and through shielding.

For interplate routing, we designed a PCB backplane which the plates can "plug" into like a server-slot system. The wires of components on each plate plug into a custom PCB contained within the plate. The PCB appropriately routes the wires according to the signal type, power or otherwise, and when the plate is slid in, plugs into a backplane in the front of the enclosure. The other plate functions similarly and the backplane makes the necessary connections between the plates. This allows the removal of individual plates without the previous requirement of disconnecting wires.

Using a custom PCB, similar to the one produced with the help of UCSD for Græy last year, our power distribution and conversion functions will be localized entirely within one compact board. The board will also contain the capabilities for a hot-swappable battery. This will be done using a low-power Lithium-Ion battery which will keep the AUV running for up to 15 minutes without motor usage. When the primary battery is plugged back into the AUV, it will subsequently charge the smaller Li-On while also providing for its other primary functions.

On Græy, we used the compact Nvidia Jetson Nano to power our AUVs. This season, we've decided to use a donated NVIDIA Jetson Xavier, on Onyx. The Xavier allows us to process data from multiple cameras and all the sensors on the AUV. It can also properly handle the more power-intensive ROS software architecture we've implemented. The benefits of using the Xavier far outweigh any potential cons, like its larger size or more power required. Additionally, the larger AUV enclosure provides us the ability to utilize the Xavier.

We continued to use a metal–oxide semiconductor field-effect transistor switch that uses an external magnetic switch to smoothly kill the power of the entire AUV.

D. Computer Vision (CV)

Last season, we experimented with using Machine Learning, Vuforia, and conventional Machine Learning (OpenCV). We decided that OpenCV would be the most effective method to implement the primary position of CV this year. As opposed to Machine Learning, OpenCV takes significantly less processing power, which also makes it easier to rapidly run simulations. Additionally, OpenCV allows for greater customizability compared to Machine Learning. Thus, we applied OpenCV to the gate, buoys, bins, and octagon missions with Vuforia for specific image recognition. However, we understand that using Machine Learning can be more accurate and foster more learning for our team. Therefore, we are concurrently developing machine learning algorithms for image recognition tasks.

Our OpenCV code utilizes the following algorithms in order: color isolation, binary thresholding, erosion and dilation, area thresholding, Contrast Limited Adaptive Histogram Equalization (CLAHE), and contour approximation.

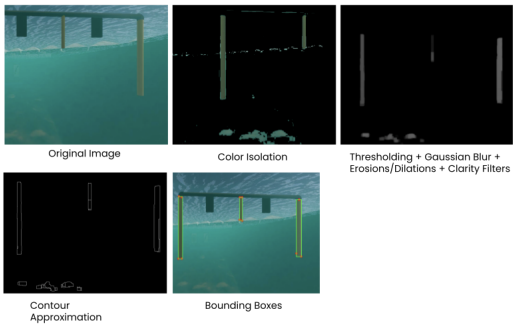

Fig. 3. *The process of performing the outlined filters on a photo of the gate's leg underwater.*

Color isolation was observed as the best method to initially threshold the mission colors that tend to contrast with the surrounding water. Color isolation removed >90% of the image. Binary isolation was layered on top so that the target color would be white and everything else would be black. When we started creating computer vision programs, we solely did color isolation, followed by binary isolation, and then contour approximation. However, we quickly learned that this would not work as these programs resulted in hundreds of bounding boxes instead of the one or two that we needed. We discovered that underwater, small specks of the necessary color were dispersed throughout the image. To combat this, we utilized erosion and dilation to remove these small specks. CLAHE was used to emphasize the remaining block of color. This served as a failsafe for the erosion/dilation as it crispened the remaining block of color. These algorithms were utilized to make the image clearer and improve recognition accuracy.

We used bounding box coordinates to locate tasks, and we usually only change the Hue Saturation Value (HSV) parameter between missions. We used HSV values as they work best in irregular lighting as the value parameter

(lighting variance) is a unit of measurement in HSV [5]. We used failsafes to achieve maximum reliability, including a size threshold of 10 pixels^2, ensuring that we only target recognized tasks. In isolated testing, using both real and simulated footage, we've found these algorithms to have an accuracy of ≥85% (by comparing computer vision program outputs to the real image coordinates).

After utilizing OpenCV, we implemented image recognition through Vuforia—an image recognition library used in industry applications—to identify targets in each mission. The Vuforia code was developed in 2020 based on our experience using it in prior ground-robot competitions. To easily integrate the Vuforia system with our AUVs, we ran the code on a Raspberry Pi running Android. The *(x,y,z)* distances between the camera and the image were sent to the navigation program running on the Jetson Nano via Android Debug Bridge (ADB). Vuforia was precise in determining the coordinates, so we used it for image recognition and short-distance navigation relative to the specific images. For example, once we navigate to the buoys' relative location, we use Vuforia to differentiate between the gun and badge on the buoys.

Furthermore, we also realized that the ability to identify an object's position and depth could enable our AUVs to navigate more accurately. From previous seasons, we saw that positioning relative to an object to perform the required task requires knowledge of how far the object is from the AUV. As a result, with a focus on simulation this season, we implemented both simulated and real-life versions of stereoscopic computer vision.

In the Unity simulation, we implemented a camera game object that simultaneously produced rightEye and leftEye views. Using these camera views, we developed depth maps to highlight any disparities between the two images and produce a single image (the disparity and depth map) that would be readable by the computer vision program. Using the OAK-D stereo camera from OpenCV AI, we were able to recreate the multi-step process from the simulation in real life.

E. Software and Navigation

1) Navigation Software Architecture. Our software architecture remains largely unchanged from the previous year. The biggest change was that the thruster command program has been replaced with a dedicated mission planning and execution program. The mission planner is covered in more depth later. Since navigating in an aqueous environment requires many processes to run simultaneously, we've continued to utilize ROS's publisher-subscriber model to exchange data between different nodes.

When deciding which inter-process communication framework to use, we chose ROS over TCP/UDP network sockets. We made this decision as ROS makes it much easier to receive and publish data from multiple different sources without needing to worry about the presence of active clients at any given moment. This means that ROS is more manageable as we scale up our software with more sensors and processes.

We have a group of programs that are specific to how their corresponding sensors process the data; computer vision programs parse camera feeds, signal processing occurs on the hydrophones' data, and so forth. The data is then passed to a sensor fusion program and a voting program. The AUV's location/state is passed to the mission planning program. We would like to expand our mission planning capabilities beyond a simple linear set of tasks. Our goal is to have the AUV respond to failures or failed attempts, and figure out the best path based on past events. The mission planning program then enters a mission execution loop for the next mission, and sends the corresponding movement commands to the flight controller.

The flight controller, which we are designing and programming from scratch, takes a desired motion as its input and produces thruster powers as its output. Its functionality can handle both six degrees of freedom movements and also concurrent actions like holding a certain depth. When determining thruster output, the flight

controller compensates for drift caused by currents, buoyancy, and other factors that exert forces upon the AUV.

2) Navigation Algorithms

a) Localization, Perception, and Waypoint Navigation. For localizing the AUVs, we used cameras, hydrophones, IMU, DVL, and Sonar.

The DVL sensor is crucial for our AUVs to operate at target velocities and position themselves. We created algorithms that optimized the information collected by the DVL by devising a proportional integral derivative (PID) controller that summed the error between target and current velocity with the derivative of each error value itself. Implemented continuously, this PID controller outputs a deceleration curve as the motors reach the target velocity and the derivative of error is calculated to be an increasingly negative number. Each value outputted by this controller is treated as a factor that is multiplied with motor power values to generate new values. This controller smoothly accelerates and decelerates our AUV to a target velocity.

Additionally, we created an algorithm for utilizing the DVL feedback to increase the accuracy of our DVL-calculated AUV pose. Because the DVL measures velocity and time, it also measures displacement. By setting an anchor (or origin) point in the TRANSDEC pool, we are able to collect the AUV's displacement over time, and add this to the anchor point to compute the AUV's current position relative to the set origin. Furthermore, task locations in the pool are also measured in terms of displacement relative to the origin. With these two vectors computed, our algorithm is able to compute the path and distance needed for the AUV to reach various tasks throughout the TRANSDEC pool. This means that by knowing the displacement of the AUV and target destination relative to a set point, our algorithm can help the AUV travel to this target destination.

b) Voting and Sensor Fusion. We combined multiple sources of sensor data to yield more accurate results. If the program detects an inconsistency between sensor data in the main execution loop, basic voting and comparisons between sensors are performed to determine which sensor has a higher probability of being correct. This is useful in applications such as short range perception. For example, the change in position of an item in the frame of a computer vision program can be used to verify an IMU reporting AUV drift. We also use a ROS library, *robot_localization*, to perform sensor fusion in order to produce accurate location information.

c) Mission Planning and Inter-Sub Communication. We realized that if we want to optimize our scoring capabilities, both AUVs need to be able to prioritize the missions and not get stuck in a mission loop if a portion fails. Therefore, we developed a mission planner with two key parts: the decision maker and the mission scheduler. If the missions are completed successfully, the AUV continues to follow the ideal path we outlined in the mission scheduler. The decision maker works like a trade study. It plans the next steps the AUVs will take through weighing different variables: the current status of the missions, time it takes to complete each of the remaining missions, the point value of the remaining missions, the probability of successfully completing the remaining missions, and the time remaining in the run. The status of the missions comes from the mission log on the AUV and from the intersub communication which provides the approximate location. This is important as it gives the point value that our team has earned at that point in time. The time estimates are a combination of the average time it takes for each individual mission, data which comes from the multiple in-water and simulator tests, and a calculation of the time it takes to navigate to a mission based on the current position of the AUVs relative to the mission locations in the pool. The probability of each mission being successfully completed is also derived from the previous mission runs. All values taken from simulation and in-water tests are updated before we run the AUVs in the water. Just like a trade study, the weights of the criteria are adjusted based on how important the criteria are. For example, the weights will change

according to the time left in the run, because as time runs out, there's a better likelihood of winning points from a less time-intensive task than one that requires more time to complete. The planner includes consideration of redoing a mission if the mission was unsuccessful via time out or not hitting the object. Missions where the AUVs drop or shoot objects are not re-attemptable as the robot cannot pick up and reload those payloads. After making its decision and logging it, the AUV updates the mission scheduler. The mission scheduler has each mission's program contained as a block and will point to the mission block that the decision maker decided on. Then the AUV will send a signal with the updated schedule to the other AUV. This way, both AUVs will be operating on the same mission schedule, and conflicts are avoided.

d) Mission Execution. Each mission is associated with a set of potential states that the robot can be in (determined by sensor inputs), and a set of potential actions the robot can take to get from one state to another. The appropriate transitory action between states is determined by the nature of the mission. For example, in the case of the buoy mission, the states could be that two buoys are within the immediate sight/proximity of the sub, one buoy is within the sight/proximity of the sub, or that none are, along with where the sub is relative to the buoy—these states are obtained from the readings from CV, the sonar, and the localization processes. The sub can perform certain actions based on the sensor data it reads to go from one state to the next. To perform actions, commands are sent to the flight controller, which translates the commands into thruster outputs to move the sub in a stable fashion.

G. Hydrophones

This season, the hydrophone sub-system went through several iterations to better triangulate the pingers within the pool. Building off our design on Græy, we strove to make the system as independent as possible. We offloaded as much filtering as possible to custom designed hardware, freeing up resources within the microprocessor. Each hydrophone has its own daughter-board containing noise-isolated circuitry. This includes the pre-amp, a custom amp, a custom variable frequency filter, adjustable between 10khz to 40khz, a digital potentiometer for variable gain, and a custom active high-pass filter to ensure the signal is as clean as possible.

Each daughter-board plugs into its respective slot on the motherboard allowing for power and signal transfer which is routed accordingly on the motherboard. Since the signal has already been selected and then dynamically filtered by the daughter boards, the microprocessor, a Teensy 4.1, no longer has to waste resources on computational heavy processes such as Fast Fourier Transforms (FFT) just to verify or select a pinger. Instead, the signal is promptly analyzed and used in custom algorithms, utilizing the time of arrival and signal phase to triangulate a heading for the pinger. The Teensy can also dynamically and autonomously change the gain using the digital potentiometer over Serial Peripheral Interface (SPI) in order to equalize and contain the signal within an analyzable state.

The Teensy transmits this heading to the main processor in degrees relative to the hydrophones' positions. The Xavier can also send commands to Teensy over the same serial protocol such as selecting which frequency to locate.

Both the motherboard and daughter-boards are the result of industry best practices that are adapted to work for our team. For instance, we followed proper signal routing requirements and created wire traces with acid traps in mind. The designs follow industry standard PCB design guidelines and were verified in design reviews before being manufactured.

III. Experimental Results

We believe that failing fast and adapting quickly is indispensable to effectively learning and implementing reliable mechanisms and software. This led us to do many rapid prototypes and experiments in the beginning of the season to quickly determine what does and doesn't work. With a tight schedule, we worked on our design

and test process in-tandem with each other. We spent time planning PDRs to foster unique designs and implemented those that were in line with the competition requirements.

One of the most important parts of our design process to ensure that any component on our AUV's software or hardware works is using detailed test plans. This not only ensures that the tester knows what to do, but also allows us to reference our learnings in the future.

We divided system testing into four key areas: test benches, simulation, CAD, and in-pool testing.

A. Test Bench Tests

Although we focused on developing a simulation platform this season, we valued tests on physical sensors to verify our simulated results. Thus, we set up a full test bench including most of our sensors, a Jetson Nano and eight continuous rotation servo motors in place of thrusters. This allowed us to run programs above water without damaging the thrusters, but still being able to see the movement.

1) IMU Test. In previous years, we used the IMU from our flight controller, the Pixhawk, for navigation. We experienced issues with inaccuracy and drifting of the Pixhawk's readings. This year, we were introduced to two MEMS AHRS devices; we compared them with the Pixhawk's IMU. To do this, we used our full test bench and let the three devices collect data into a file with timestamps while they were static. Then, we used *Plotly*, a Python plotting software, to produce the graph in Figure 4 and verify the drift of the PixHawk and the consistency of the readings from the new AHRS devices.

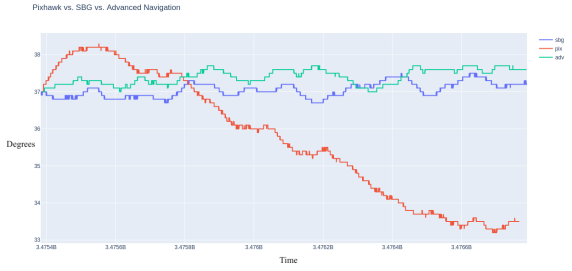

Fig. 4. *Graphed data of the PixHawk IMU (red) and AHRS heading values (blue and green). PixHawk values seem to drift significantly. Values from the AHRS devices had an better accuracy of ~0.8º*

2) DVL Test. To test our algorithm to parse the DVL's output, we fed it a series of data values representative of a typical AUV pathway, like starting in a stagnant position and increasing velocity to reach what is optimal in certain portions of the competition course. Given these initial example parameters, we observed the feedback loop and saw that the factor at which motor powers were calculated to change at were greater when the overall error between current velocity was greater. This factor for motor power change decreased when the AUV's velocity reached the target velocity, and inversely increased when current velocity was further from target velocity. In other words, our tests demonstrated that the PID controller generated a functioning acceleration or deceleration curve.

B. Simulated Tests

1) Computer Vision. Before integrating our computer vision onto the AUV, we simulated our tests in Unity. First, we used screen recordings from the simulator of the specific mission our program was for. This allowed us to test with simulated conditions such as lighting or water color before the simulation was developed enough to send live video feed. Next, once the computer vision program was working on recorded data from the simulators, it was integrated into the simulator AUV and tested with the virtual camera output from Unity. Following this test, further changes were made until the computer vision code was accurate. The computer vision code was then integrated onto the AUV, where it went through further tests to adjust to the real world.

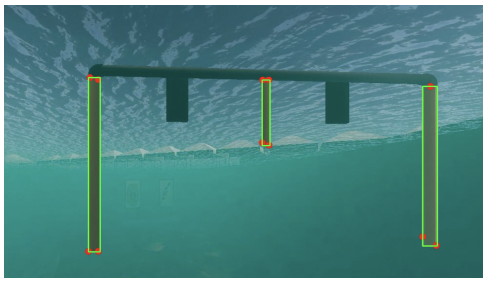

Fig. 5. *OpenCV code identifying the gate on live video feed from the Unity simulation.*

2) Stereo Vision. Using OpenCV documentation, we developed a program that compared leftEye and rightEye camera views from the Unity simulation to generate an accurate disparity/depth map. The program required adjustments to a myriad of variables, like the number of disparities, the range of visible depths, and the size of pixels near distinct objects; proper tuning of these variables yields a better depth map. We utilized a GUI to more efficiently adjust the variables. After the fine-tuning was completed, the disparity/depth map produced better, smooth results and allowed for a more idealistic implementation of stereoscopic vision to accurately estimate objects' position in the simulation.

C. CAD Tests

Our team makes sure to CAD and review each component of our AUV with mentors before beginning the physical construction. This allows us to do all of our iterations in the virtual environment, saving time and resources during the assembly process. Designing in CAD also allows us to test the durability of each part through stress analysis to understand potential points of failure.

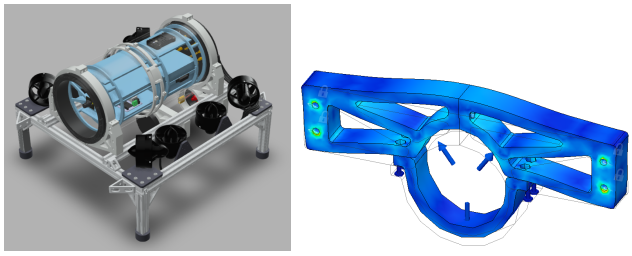

Fig. 6. *A CAD model of our latest AUV, Onyx and a 10 Newton stress test on our 3-inch enclosure*

D. In-Pool Tests

1) Thruster Test. This season, we performed various in-pool tests to verify the functionality of our physical AUV. One such test was to ensure the thrusters function and spin in the proper direction. This is because when we first put Græy in the water, all the motors were spinning in random directions. We identified that this issue could be because of a physical mounting or software configuration issue. We tested both hardware and software configurations for each individual thruster, and documented every change. After analyzing our results, we found the direction each motor propels the AUV depends on the physical propeller direction. Therefore, we re-mounted and verified every reversed thruster.

Regardless of the nature of each test, test plans help us identify the goal or measure the success of each component. We have had many failures with testing components and we have documented each of them on our website. If a test fails, we identify the root cause of failure and work on fixing and preventing it in the future. For example, at the beginning of the season, our epoxy-sealed penetrators had cracks that caused water to enter our electronics enclosure. Realizing that this problem would eventually occur again, even if we re-potted the penetrators, we researched a new type of connector that would seal the connector from both sides of the enclosure end cap. This would greatly reduce the chances of leaks in the penetrators, and be more permanent than repotting.

Acknowledgements

Team Inspiration would like to thank the following sponsors: Northrop Grumman, Qualcomm, Biosero, Kenautics, Water Linked, Blue Trail Engineering, Medtronics, ePlastics, Nvidia, MathWorks, Sempra Energy, Solidworks, tinyVision.ai, Brain Corp, Manna's Martial Arts, Metal Master, Jet Brains, Forecast 3D, and Roboctopi. We received engineering support from Water Linked [6], Aquarian [7], and Blue Trail Engineering [8]. We are grateful to our mentors: Jack Silberman, Alex Szeto, Amit Goel, Pat McLaughlin, Venkat Rangan, Teresa To, Kris Chopper, Kenzo Tomitaka, Rory Miller, Ralph Morales, Kevin Bowen, Andrew Raharjo, Alan Kenny, John Roscoe-Hudson, Debra Kimberling, and Cris O'Bryon. We appreciate the RoboSub organizers for putting the competition together to challenge us. We thank Porpoise Robotics for the partnership and opportunity to design low-cost underwater remotely-operated vehicles and their desktop development systems for STEM education. Lastly, we are indebted to our parents for their continuous support.

References

[1] (2021)VRX (Version 1.2) [source code] https://github.com/osrf/vrx
[2] PWr Diving Crew (2020) TransdecEnvironment (Version 1.0) [source code], https://github.com/pwrdc/TransdecEnvironment
[3] P. Saxena, “RoboSub-Simulation Documentation,” unpublished. https://docs.google.com/document/d/1kxkTuTfGHhB9nrnwl3W8o6WbOPRRYIR3o-PVBe9p4dI/edit?usp=sharing (accessed Jun. 27, 2021)
[4] https://gm0.org/en/latest/
[5] R. Guha, “RGB v/s HSV for Computer Vision,” Imaginea Labs, Dec. 4, 2019, https://labs.imaginea.com/rbg-vs-hsv-for-computer-vision/ (accessed Apr. 20, 2021).
[6] Water Linked, https://waterlinked.com (accessed May 7, 2021).
[7] Aquarian Audio & Scientific, https://www.aquarianaudio.com (accessed Apr. 15, 2021).
[8] Blue Trail Engineering, https://www.bluetrailengineering.com/ (accessed Apr. 23, 2021).

Appendix A: Component Specifications

Component	Vendor	Model/Type	Specs	Cost (if new)	Status
Buoyancy Control	Blue Robotics	LAST-A-FOAM® R3318 2. Stainless Steel Ballast Weight 3. 2” series enclosure, 6” series enclosure	1.9in x 3.4in x 4.9in 2. (200 g, 7 oz) 3. Refer to Waterproof housing specs https://bluerobotics.com/store/watertight-enclosures/buoyancy/float-r3318-r1/	Legacy	Installed
Frame	80/20 Inc.	1010 aluminum extrusion	Weight: 25 oz Size: 22 in x 7.9 in	$234.73	Installed
Waterproof Housing	Blue Robotics	Acrylic tube, 8 in. series	Size: 8” diameter	$343	Installed
Waterproof Connectors	Blue Trail Engineering	10 cobalt series dummy plug 10 cobalt series locking sleeve 4 cobalt series cable termination kit	https://www.bluetrailengineering.com/product-page/cobalt-series-dummy-plug https://www.bluetrailengineering.com/product-page/removable-cobalt-locking-sleeve https://www.bluetrailengineering.com/product-page/cable-termination-kit	$201.75	Installed
Thrusters	Blue Robotics	T200 thrusters	Full Throttle FWD/REV Thrust @ Maximum (20 V) https://bluerobotics.com/store/thrusters/t100-t200-thrusters/t200-thruster-r2-rp/	Legacy	Installed
Motor Control	Blue Robotics	Basic ESC	30A brushless ESC https://bluerobotics.com/store/thrusters/speed-controllers/besc30-r3/	Legacy	Installed

High Level Control	Blue Robotics	Fathom-X and Fathom-X Tether Interface (FXTI)	Communication: USB 2.0, Ethernet 10/100 https://bluerobotics.com/store/comm-control-power/tether-interface/fathom-x-r1/	Legacy	Installed
Actuators	Blue Robotics	Newton Subsea Gripper	Grip Force: 28 N Jaw Opening: 2.75 in https://bluerobotics.com/store/rov/bluerov2-accessories/newton-gripper-asm-r2-rp/	Legacy	Installed
Propellers	Blue Robotics	T200 propellers	Max thrust: 49.82 N	Included with thrusters	Installed
Battery	Blue Robotics	Lithium-Ion Battery	4s 14.8V 18Ah	Legacy	Installed
Converter	Blue Robotics	i^2C level converter	Operating Voltage 3.3v - 5v Max Current @ 3.3v Vout 150 mA Output Connector 4 pin 0.1” header Input Connector 4 pin 0.1” header + JST-GH + DF13	$15.00	Installed
Regulator	Blue Robotics	5V, 6A power supply	5V, 6A	Legacy	Installed
CPU	Nvidia Nvidia	Nvidia Jetson Nano Nvidia Jetson AGX Xavier	1.4 GHZ clock speed 4 GB RAM https://developer.nvidia.com/embedded/jetson-agx-xavier-developer-kit	$99.00 on Græy On Onyx Donated by sponsor	Installed Purchased
Internal Comm Network	ROS	ROS 1/Kinetic	N/A	Free/Open Source	Installed
External Comm Interface	N750 Wireless Dual Band Gigabit Router	TL-WDR4300	Simultaneous 2.4GHz 300Mbps and 5GHz 450Mbps connections for 750Mbps of total available bandwidth	Legacy	Installed

Compass	Pixhawk	ST Micro LSM303D 3-axis 14-bit accelerometer / magnetometer	3-DOF Magnetometer (on the Pixhawk)	$139.00 on Græy	Installed
Inertial Measurement Unit (IMU)	Pixhawk	Invensense® MPU 6000 3-axis accelerometer/ gyroscope	32-bit ARM Cortex M4 core with FPU 168 MHz/256 KB RAM/2 MB Flash 32-bit failsafe co-processor	Included with Pixhawk on Græy (compass)	Installed
	Dampener	XTORI Pixhawk dampener	Materials: plastic and rubber Weight: 17 g	$7.99	Installed
AHRS	SBG Systems	Ellipse A AHRS	https://www.sbg-systems.com/products/ellipse-series/#documentation	on Onyx *Loaned by sponsor*	Installed on test bench
	Advanced Navigation	SPATIAL	https://www.advancednavigation.com/products/spatial	on Onyx *Loaned by sponsor*	Installed on test bench
	Advanced Navigation	ORIENTUS	https://www.advancednavigation.com/products/orientus	on Onyx *Loaned by sponsor*	Installed on test bench
Depth Sensor	Blue Robotics	30 Bar, High-Resolution 300 m Depth/Pressure Sensor	Operating depth: 300 m Supply voltage: 2.5-5.5 volts	Legacy on Græy $72.00 on Onyx	Installed
Doppler Velocity Log (DVL)	Waterlinked	A50	Transducer frequency : 1 MHz Transducer setup : 4-beam convex Janus array Transducer beam angle : 22.5 degrees Ping rate : 4-26 Hz (adaptive to altitude) Sensor assist: Integrated	$7,000.00	Installed physically, Selected for simulation

			AHRS/IMU (Yost Labs TSS-NANO) Min altitude: 5 cm Max altitude: 50 meters (> 35 m is dependent on seabed conditions, salinity levels etc.) Max velocity: 2.6 m/s Velocity resolution: 0.01 mm/s Long term accuracy: ±1.01 % or ±0.1 % (Performance version) https://waterlinked.com/product/dvl-a50/		
Vision	Blue Robotics	Low Light HD USB Camera	Pixel count: 2MP 1080P Onboard H.264 compression chip 32x32mm	Legacy on Græy $198.00 (2x) on Onyx	Installed
Sonar	Blue Robotics Teledyne Marine	Ping Sonar Altimeter and Echosounder BlueView M900-2250-130-Mk2	- 30m distance @ 15cm resolution - 300m depth rating http://www.teledynemarine.com/M900-2250-130-Mk2?ProductLineID=1	Legacy on Græy $251.10 on Onyx On Onyx Loaned by sponsor	Installed Loaned
Acoustics	 Aquarian Audio & Scientific	Teesny 4.0 Aquarian PA4 preamp Custom NE5532 Op-Amp AS-1 Hydrophone	Sample Rate: 200kHz, 12 bit Gain: 50db or 0.005% THD @ 1kHz Gain: 21db or 0.002% THD @ 1kHz Linear range: 1Hz to 100kHz ±2dB Horizontal Directivity (20kHz): ±0.2dB	$314.77 Legacy	Installed on test bench Installed on test bench

			Horizontal Directivity (100kHz): ±1dB Vertical Directivity (20kHz): ±1dB Vertical Directivity (100kHz): +6dB -11dB		
Manipulator	Blue Robotics	Newton Subsea Gripper	Grip force (at tip): 97 N Grip force (at middle): 124 N Jaw Opening: 62 mm Time to Open/Close: 1.2 secs https://bluerobotics.com/store/rov/bluerov2-accessories/newton-gripper-asm-r2-rp/	$439.00	Purchased
Algorithms: Computer Vision, OpenCV	Custom	OpenCV	Color isolation, binary thresholding, contour approximation, erosion and dilation, area thresholding, and Contrast Limited Adaptive Histogram Equalization (CLAHE)	Free/Open Source	Installed
Algorithms: vision	PTC Inc.	Vuforia/Vuforia License	Vuforia Engine version 8.6	Free	Installed
Algorithms: acoustics	In-house	Custom	Fast Fourier Transform (FFT)	Free	Installed
Algorithms: localization and mapping	In-house	Custom	DVL, Hydrophones, CV	Free	Being developed by the team
Algorithms: autonomy	In-house In-house	Custom Custom	Voting algorithm Mission planner	Free Free	Being developed by the team
Open source software	Open-Source (n/a)	Open Computer Vision, Robot Operating	Computer Vision, Inter-process communication, programming, computer	Free	Installed

		System, Python, C++, Linux	operating system		
Team size (number of people)	n/a	15 Students	Middle and high schoolers	Priceless	Working
Expertise ratio (hardware vs. software)	6:12	Hardware subteams: mechanical, electrical, payload Software subteams: software architecture, navigation, computer vision, payload	Colin, Noah, Ashika, Pahel, Eesh, Mabel Aditya, Rishi, Ashiria, Pahel, Shruti, Ashika, Eesh, Pratham, Maxime, Isabelle, Claire, Raina	Priceless	Working
Testing time: simulation	80 hours of testing	Software subteams: software architecture, navigation, computer vision	Aditya, Rishi, Ashiria, Pahel, Shruti, Ashika, Eesh, Claire, Pratham, Maxime, Isabelle	N/A	Working
Testing time: in-water	15 hours of testing	Hydrophones, OpenCV, Neural Networks, Vuforia	Eesh, Shruti, Pahel, Ashiria	N/A	Working
Inter-vehicle communication	Waterlinked	Modem M64	- Two-way communication - 64 bits/second net data link, both ways - Latency: 1.5 – 2.5 sec - Range: 200 m https://waterlinked.com/product/modem-m64/	$3,000 (2 modems)	Installed physically, tested in test bench
Programming Language 1	C++	C++ 17.0	N/A	Free/Open Source	Installed

Programming Language 2	Python	Python 3	N/A	Free/Open Source	Installed

Appendix B: Outreach Activities

Our team's motto, "To learn, to share, to innovate, to inspire," highlights our goal to share our enthusiasm for STEM and inspire others to participate in STEM. Most of us on Team Inspiration had early exposure to robotics and are working to share our experiences and opportunities with others. We especially focus on those who are less fortunate and/or lack access to resources and mentorship. We've created sustainable robotics programs in 4 countries, taught robotics camps/classes, and reached out to adolescents at science museums alongside all of the major San Diego STEM fairs. In the past year alone, Team Inspiration has participated in over 1,400 hours of community outreach and service. To spread our knowledge of RoboSub, we have brought our AUV to many public and private events, sharing our RoboSub journey with thousands of people.

We also continue to host monthly webinars through our SWENext club, WE STEM, to connect the public and local robotics teams with professors and industry professionals in STEM, focusing on female industry professionals, including Christy Predaina, a Senior Director at Northrop Grumman, and Pamela Cosman, a Professor of Electrical and Computer Engineering at the University of California, San Diego.

We wanted to highlight interesting projects and engineering concepts through our speaker series so that we can engage the public in STEM and encourage others to learn more. Specifically, our speakers have been experts in systems engineering and underwater and surface vehicles; we wanted to provide expertise that would apply to STEM organizations/teams and to promote more interest in marine technology.

One of the aforementioned speakers was Damon McMillan from Blue Trail Engineering. He presented SeaCharger, his autonomous surface vehicle that traveled a quarter of the way around the globe powered solely on solar energy.

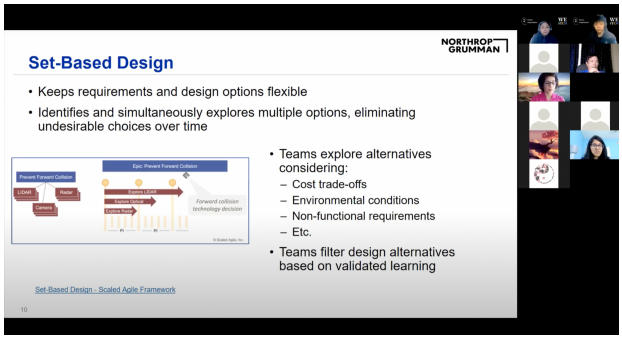

Fig. 7. Gabriela Coe, a Consulting Engineer and Technical Fellow at Northrop Grumman, presenting about Agile Systems Engineering during our speaker series.

Because of our success in the 2019 and 2020 RoboSub competitions, we received multiple opportunities to share our RoboSub journey with others. For example, we have promoted our team's story on local news stations like ABC 10 and CBS 8. We also got the opportunity to present at several professional engineering conferences, such as the San Diego County Engineering Council, Marine Technology Society, and the International Council on System Engineering. Recordings of the broadcasts and submitted journals for these public appearances can be found in the Media section of our team's website.

Fig. 8. The flyer to publicize one of our webinars, which we presented at the Marine Technology Society and others.

During these presentations, we discussed our experiences competing in Robosub competitions, emphasizing how we utilized systems engineering and extensive testing to achieve high rankings in the competition. We will also be holding a Jeston AI Labs livestream with Nvidia later this year, as well as speaking at the Global OCEANs 2021 conference.

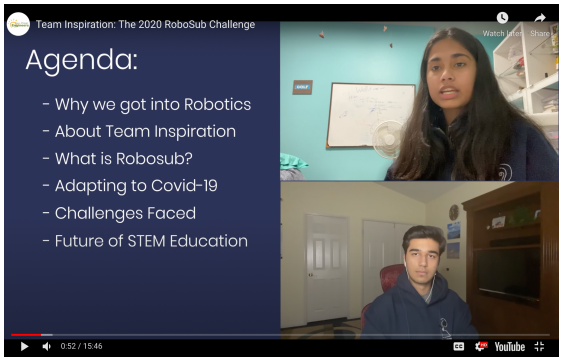

Fig. 9. On the SDCEC website, there is a video of our journey during Robosub.

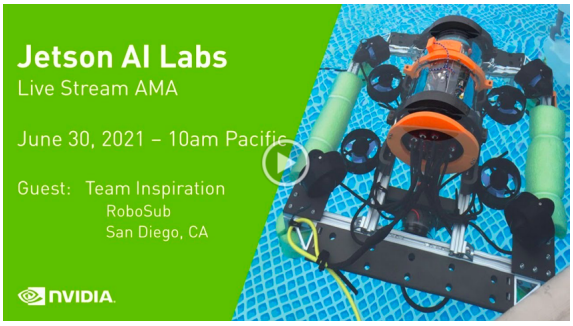

Fig. 10. Flyer for a planned Nvidia livestream on June 30, 2021.

As a result of our participation in numerous robotics competitions, including FIRST Tech Challenge, we have developed a number of connections with professionals from a myriad of industries. One of these professionals is Ranch Santa Fe's Coach David Warner. He donated the SeaPerch materials that he used to teach his middle school students to us. Using these resources, we hosted an event that highlighted underwater robotics, featuring the AUVs we designed for the RoboSub competition and the SeaPerch materials, exposing the students to a type of robotics they have not tried before. We will be using the SeaPerch materials to continue passing on knowledge of underwater robotics.

We strive to apply our experience in Robosub to benefit the local and global STEM education community. Thus, we have been collaborating with Porpoise Robotics, a team of former submarine engineers, to design low cost, underwater remotely-operated vehicles (ROV) to make STEM—especially underwater robotics and mechatronics—more accessible. The ROV is designed to be an affordable robot that can be used in schools. Our team members have helped develop the ROV robot by testing the custom propellers, writing the base algorithms, and creating a web client to control the ROV. We have been working with Porpoise Robotics to create a 2-week curriculum for the AUV that schools can use to teach students Python, underwater mechanics, and electrical integration.

The Porpoise Robotics underwater ROV comprises of a Raspberry Pi computer, LiPo Battery, servo driver, four Electronic Speed Controllers (ESCs) and Thrusters, camera and external lights, all connected via a 30-meter ethernet tether to a computer and joystick.

All in all, the system we've worked on has the potential to dramatically increase access to underwater robotics within the scope of education.

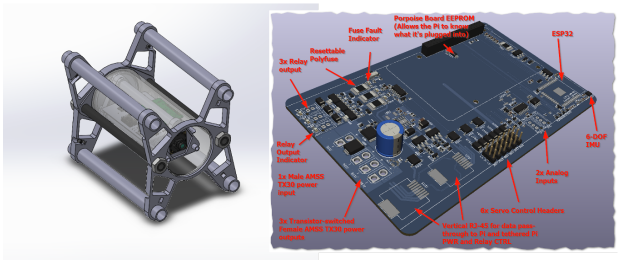

Fig. X. Prototype ROVs developed in collaboration with Porpoise Robotics.

Overall, we've hosted various events, presented at conferences, led workshops, and developed platforms that help us accomplish our mission of "To learn, to share, to innovate, and to inspire." And we have worked to ensure our efforts are self-sustaining and expand beyond our time in robotics.