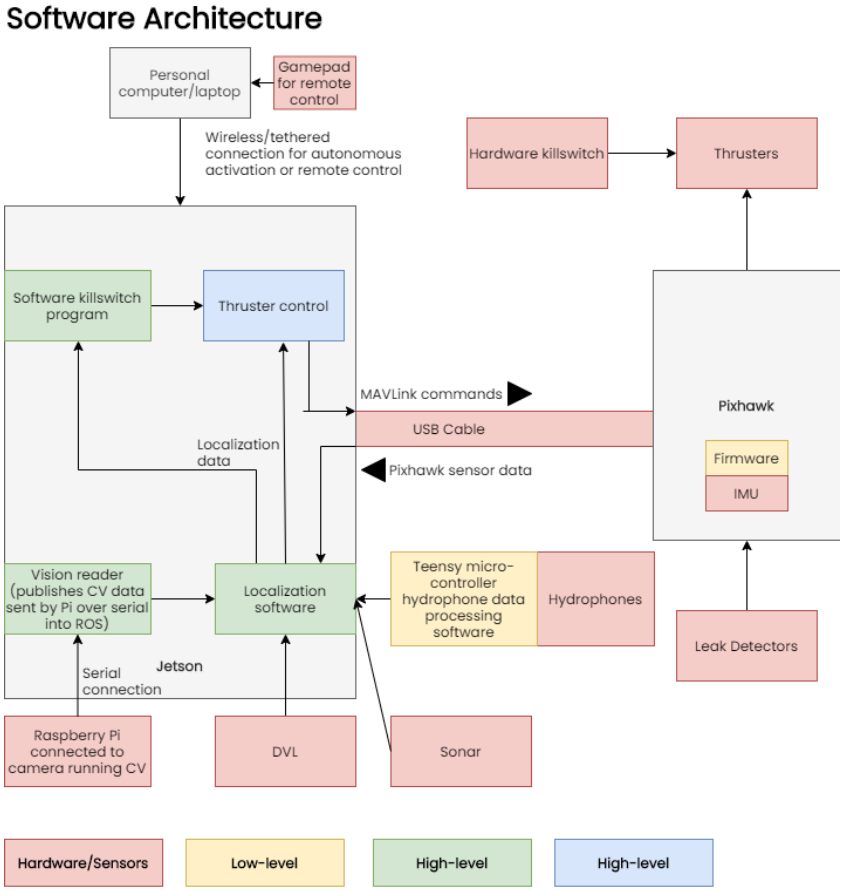
Software

The software team had the unique challenge this year of utilizing a simulator to prove and test programs without access to the assembled sub.

Architecture

General



The software system consists of four different parts. A group of programs

Flight Controller

No image	Contrary to last season, we are moving in the direction of directly controlling our thrusters and developing a custom flight controller for Onyx. In simulated tests, we are mimicking the behavior of the thrusters by having each simulated thruster propel a force as controlled by a software client that represents the flight controller. Basic translation is handled by the flight controller, meaning that if we wish the AUV to move in a certain direction, we simply need to send a command to the flight controller with the specified direction and power. The details concerning which motors should be activated to translate in the specified direction (forward/backward, laterally, yaw, vertically, etc) are handled by the flight controller. The flight controller will compensate for drift caused by currents and other factors, and will have features like holding the sub's depth while underwater to compensate for its buoyancy. We also take advantage of the flight controller's capability to maintain a depth while translating forwards, backwards, and laterally.

Mission Planning and Execution

No image	This season, we've begun working on developing a mission planner between the two AUVs to be able to prioritize tasks and maximize our ability to gain points while underwater.
	The mission planner has two key parts: a decision maker and mission scheduler. The decision maker works like a trade study, weighting different variables to decide which task to pursue. The different variables are the current status of the missions, time it takes to complete each of

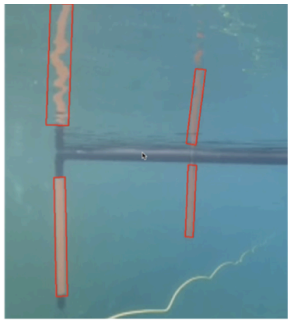
	the remaining missions, the point value of the remaining missions, the probability of successfully completing the remaining missions, and the time remaining in the run.
	Græy and Onyx are designed to communicate with one another. Inter-Sub communication and mission planning go hand-in-hand. This way, we avoid overlap while strategically planning to gain the most points.

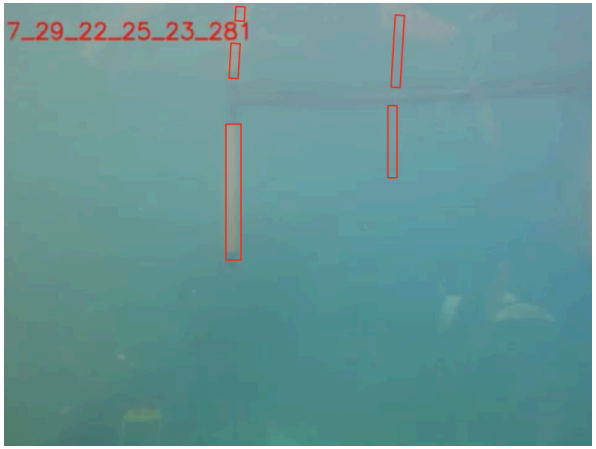
Voting

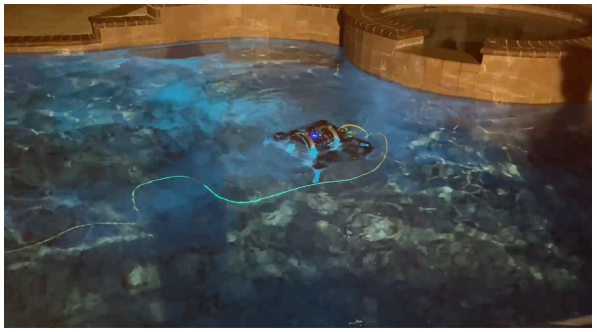
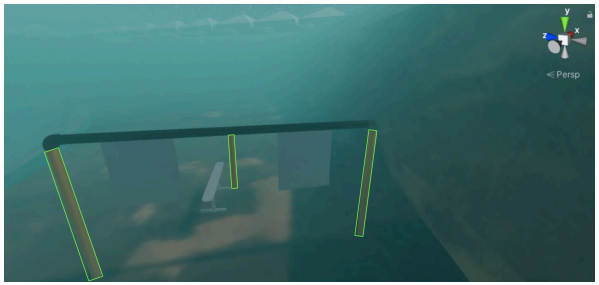
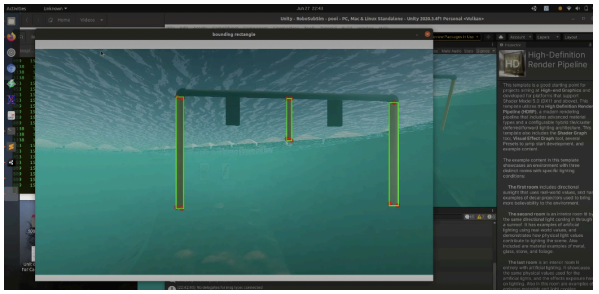
See last year's page (Software > Voting) https://team11128.wixsite.com/main/graey

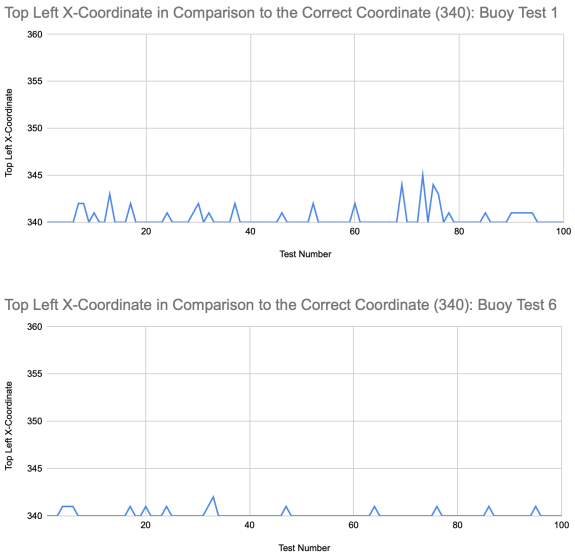
Computer Vision

CV Evolution

 	In past seasons, the computer vision subteam focused on learning and curating programs that would work for multiple missions. We did this by experimenting with 3 different forms of computer vision: OpenCV, Vuforia, and machine learning. Through the development of these programs, we established a set of basic programs that could be repurposed for different missions in Robosub. We created programs to detect the gate, buoy, octagon, and bins and planned to expand to other missions. At this stage, we had not implemented our computer vision programs

	successfully onto the physical robot. This would turn into a goal that we work to accomplish this season.
	Furthermore, we noticed that we did not have a well established testing strategy. When testing our programs, we typically only tested our program on a few pre-recorded videos recorded from the team pool and from TRANSDEC during our 2019 season. In order to increase the accuracy of our computer vision programs, we needed to test on more footage in order to ensure our program works in any circumstances. We also needed to improve our test plan so we could collect quantitative data that would help us find our program's accuracy and see when and what needed to be improved. We also needed to increase the specificity of our testing goals so that other programmers would be able to understand our test data and clearly understand what we changed and why. We could improve this by writing improved test plans in the beginning of our test and collecting test data that describes what we are testing and what changes we made beforehand. This would make the test results more clear to viewers.

	During this season, we implemented these changes in our testing strategy. Since in previous years, we tested our programs on old in pool footage, we already had relatively accurate programs for those videos, but we soon discovered that this would be ineffective, because we were only testing in a limited set of conditions.
	We tested these programs on pre recorded simulator footage this season to see if these programs would maintain their accuracy in a new environment. The subteam working on the simulator sent us screen recordings of the asset we wished to test on. CV members tested a corresponding program.
	Once tested on pre recorded footage, we tested in the simulator. This involved taking the video stream from the virtual camera and applying the filters directly to them. Our test results were significantly different, dropping from around 85% accuracy to below 50%. This was likely caused by changes in the lighting conditions/fog settings of the underwater environment, so we adjusted the HSV values accordingly. We were able to move the robot around and test the accuracy of the CV by looking at the bounding boxes formed. Originally, problems were faced with the CV program detecting parts of the floor as the same color as the gate, and this problem was fixed by cropping out the bottom at the beginning of the program. We also used the same code to test the path, and it was able to draw a bounding box around the path with ~90% accuracy. Simulating the robot also gave us a baseline

	for integrating CV with our actual sub, since integration is done in the same way.
	The data collected from these tests were quantified and displayed on a graph to visualize the accuracy of the test results. In the example to the top left, the graph is comparing the top left x-coordinate received from a CV test on buoy footage. The closer the line is to the 340, the higher the accuracy of the program was. The more tests we did, the closer the line got to 340, indicating to others that the program increased in accuracy. In the graph on the bottom left, the line is close to 340.
	This season, we were able to integrate our CV program with the sub as a whole within the simulation. This allowed us to test in more versatile conditions, and gave us more data points to improve our program and introduce/change the order of our filters. Because we were able to focus on OpenCV, we were able to carry out the process of making programs for each mission more effectively.

	In the future, we plan on integrating our programs with the sub much earlier so we can test more accurately earlier on. We will continue to test in both real-life conditions (through videos and on the physical sub), as well as the simulation. We also plan on reintroducing Machine Learning because it will allow for the greatest amount of learning, and after further development, could yield more accurate results.