CTHULHU: The Design and Implementation of the Duke Robotics Club's 2021 RoboSub Competition Entry

Eric Jiang, Nikhil Chakraborty, Muthukurisil Arivoli, Shaan Gondalia, Kara Lindstrom, Reed Chen, Brandon Bae, Tyler Feldman, Vincent Wang, Kevin Yang, Andy Zhou, Drew Council, Maverick Chung, Christopher Cameron, Alan Bi, Sarah Glomski, Ricky Weerts, Samuel Rabinowitz, Trevor Fowler, Eric Chang

***Abstract* —Introducing Cthulhu v2, the 2021 RoboSub entry of Duke Robotics Club. The pandemic has underscored the importance of flexibility and adaptability, and Cthulhu v2 was built around these core tenets. Our team of 25+ student engineers has built an AUV which not only can reliably perform "fundamental tasks" (e.g. movement, object identification, gripping, etc.), but can also effectively execute high-level "complex tasks" (e.g. drop marker in bin) by concurrently running these fundamental task building blocks. Our priority was to design simple, modular systems with minimal interdependence, and this focus on modularity has allowed our many subteams to quickly iterate in a virtual environment. Notable strategic additions include: brand-new pneumatic actuators, revamped computer vision, a feature-rich custom simulation, and streamlined CI/CD pipelines. Duke Robotics Club is also actively involved in community outreach, closely mentoring teens at Families Moving Forward (a transitional housing center in Durham, NC) through biweekly virtual engineering workshops.**

I. Competition and Design Strategy

Based on our experience from prior years, we had a good idea of what tasks would appear in competition. We would have to pass through the gate, followed by some combination of bumping buoys, dropping markers, shooting torpedoes, and picking up objects. We would also need to be able to follow the task markers – meaning our AUV would need to see the marking lines and register the acoustic pingers.

We chose to keep general robot design flexible so we could easily debug components on the fly and quickly adapt to the specifics of this year's competition. We did prioritize some tasks over others to maximize our point total while maintaining reliability, and we will delve in-depth into this strategy in the subsections below.

A. General Strategy: Flexibility and Modularity

Flexibility starts with the mechanical design. We overhauled the mechanical stack, designing a brand-new frame as well as pneumatic-based actuators (for launching torpedoes, dropping the markers, etc.). Ultimately we chose a flat, rectangular design with multiple capsules to keep the electronics neat and the robot well balanced, as well as to allow for easier access to the batteries (placed in an individual capsule) which needed to be swapped often. One of our key strategies for earning points was to ensure that Cthulhu v2 had six degrees of freedom. We achieved this by designing our frame to be wide and flat, such that the thrusters could be spaced far apart and generate more torque. This wide range of maneuverability affords our software team great flexibility in task planning.

Previous iterations of our robot featured a single cylindrical capsule to house our electronics. We found that this cylindrical design was neither space efficient nor flexible, hindering our ability to quickly access and debug components as well as incorporate new hardware (this was a large problem for us in previous competition

years). Our new rectangular design remedies these limitations. The multi-capsule design allows for improved organization, ease of access, and improved balance. We added legs to allow Cthulhu v2 to either stand right-side-up or upside-down, affording us easy access to adjust components when pool testing or at competition. Other key strategic outcomes include ease of manufacturing and adjustable mounting.

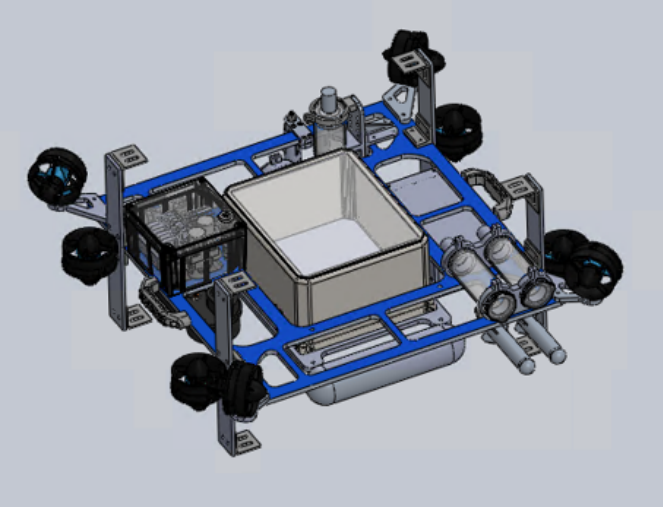

Fig. 1. Cthulhu v2's Redesigned Frame

Among our biggest takeaways from our previous experience at 2019 RoboSub was that we needed a more flexible software system that could be easily adjusted and added to on the fly. In the past two years, we have completely overhauled our software stack with flexibility as the guiding principle. When writing code, our primary focus is simplicity; rather than hastily developing sloppy, complicated packages, we focus on carefully creating basic, fully-tested modules. Using this approach, we can easily layer modular building blocks to accomplish complex tasks.

We have built a lean, organized codebase based on Robot Operating System (ROS) where the code for each software system (e.g. computer vision, controls) lives in its own package. All code is highly documented, with README's for each package and docstrings for all functions. This emphasis on organization has proved crucial for quickly onboarding new members, as well as facilitating a streamlined development process.

Rather than take a top-down approach to writing code for competition tasks, the software team chose to utilize a bottom-up programming methodology. In our system, all tasks inherit from an abstract "Task" class, which gives us a standard interface to program, execute, and coordinate individual tasks. Our primary software strategy was to develop basic "fundamental tasks", which can then be combined to form complex high-level tasks. The fundamental tasks can be considered basic building blocks which can be combined/executed in any order. This allows us to mix and match fundamental tasks to accomplish a variety of competition tasks without having to specifically overengineer for any particular one.

The glue that holds our strategy together is computer vision (CV). Our CV team has developed a deep-learning-based object detection/classification library, Detecto. Detecto is built on top of a Faster R-CNN neural network, and it is not only capable of identifying the location, but also classifying the type of nearly any object in an image. Detecto requires a relatively minimal of training data to produce reliable models, which gives our team the flexibility to add new objects of interest with ease. Detecto is a key part of our competition strategy, ensuring that Cthulhu v2 is never hindered by "not knowing" what objects are in its field of view.

B. Gate Task

We made it a top priority to be able to consistently identify and move through the gate, since this is the standard qualification task. Our mechanical design and thruster placement is already well-suited to handle the movement required for going through the gate. CV can accurately identify the boundary of the gate and the tick in the middle. It also creates a depth map telling us approximately how far the robot is from the gate using stereo vision.

With all the above inputs and our bottom-up task planning framework, writing the code to perform the gate task simply involves arranging our modular task-planning building blocks with the right logic. These building blocks include include, for example, moving to a desired local position or rotating – together, the blocks produce complex movements (see *Novel Aspects*). Overall,

we were able to accomplish the gate task with no added complexity.

Our CV can effectively identify the "role" our robot has chosen. This is another source of points going forward.

C. Style Tasks

Our frame and thrusters allow for 6 degrees of freedom, meaning that theoretically, Cthulhu v2 can accomplish any style task we desire. Programming style tasks is made simple due to our modular task planning codebase. We feel that we can effectively gain points by performing a variety of maneuvers, including barrel rolls and spins.

D. Buoy Task

Thanks to our deep-learning-based computer vision algorithms, this once challenging task in previous years has suddenly became fairly simple. Our CV is robust enough to correctly identify the location of the buoy and classify the corresponding image, even from a distance and in a variety of lighting conditions. Task planning is once again easy with our bottom-up approach. We were able to integrate this task into our autonomous runs with minimal added complexity.

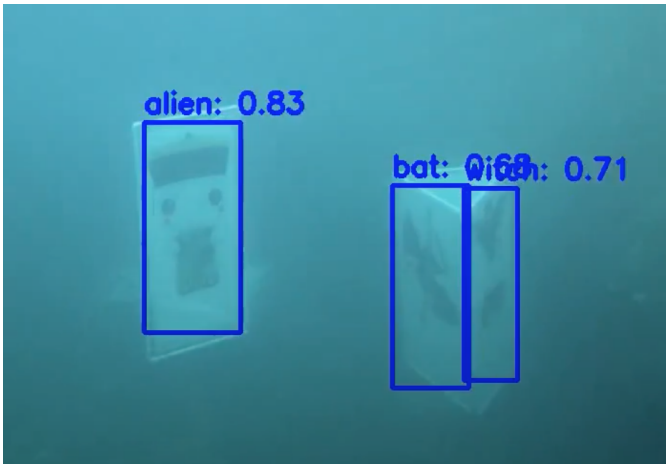

Fig. 2. Buoy Object Detection via Detecto CV

E. Bin Task

A big win this year are our redesigned mechanical actuators. Our torpedo launcher is now pneumatically powered, and can launch a torpedo with a single puff of air. For the Bin Task, our "markers" are simply torpedos, which are fired by downward-pointing torpedo launchers. By reusing the same launcher mechanism for both the Bin Task and Torpedo Task, the mechanical team was able to focus their attention on perfecting one specific actuator, rather than splitting efforts between two disparate mechanisms.

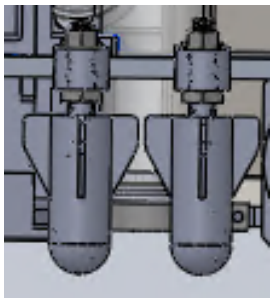

Fig. 3. Downward-facing Torpedo Launchers

CV can once again tell us the positions of the bins, as well as provide a distance estimate with stereo vision. Based on pool testing, we believe we will not have to move the cover of the bin before we apply CV and drop our marker, although this is something we would have to investigate on-site during the competition. Our CV can differentiate between the bootlegger and the G-man bins.

Although this is a more difficult task overall, Cthulhu v2 is prepared with all the necessary components to execute it. Thus, we have made it a priority to accomplish this task.

F. Torpedo Task

We feel that the Torpedo Task is similar in difficulty to the Bin Task. The CV requirements are identical – we need to find and differentiate the Bootlegger and G-man targets. When designing the torpedo launcher, our main strategic priorities were to:

1) Impart sufficient lateral force to mitigate the effects of drag.
2) Ensure the torpedo path would not deviate significantly from the target.

We have tested our torpedo actuators and are confident in their power and accuracy, so we are going for this task.

G. Octagon Task

Given the current capabilities of Cthulhu v2, the Octagon Task is the most challenging. Cthulhu

v2's claw consists of four moving jaws (similarly to an arcade claw). This revision, compared to our former two-dimensional design with one moving jaw, enables Cthulhu v2 to grab bottles at more angles, hence decreasing the strain on our programmers and increasing our chance of successfully grabbing bottles regardless of orientation. Unfortunately, testing has shown that the claw is still not reliable enough; while it can grasp most objects, sometimes the claw cannot maintain its grip.

In addition, the task planning and CV capabilities required to execute this task seem out of reach right now. We have to spot every bottle with CV, manipulate the claw to pick it up, make sure the claw is in the correct position, grasp the bottle, and move and drop the bottle in the correct location. Given the complexity of the undertaking, we decided it was more prudent to focus on other tasks. Accomplishing the Octagon Task is a great goal for next year's competition, but was not feasible this year.

H. Task Markers

This year, we prioritized the ability to detect task markers, namely the path segments and the pingers. Our CV model can effectively detect path segments, which is a huge strategic advantage to direct Cthulhu v2 toward the next task. We use two sets of hydrophones to triangulate the position of the pingers. Our acoustics package is designed to combine the data collected by each hydrophone set to yield a good estimate (see *Novel Design Aspects*), and we validated its performance in a virtual environment. Overall, we are confident Cthulhu v2 can reliably see/sense the task markers, which ensures our robot can navigate toward point-generating tasks.

II. Design Creativity

A. Mechanical Subsystem Novel Aspects

Cthulhu v2 features a brand-new suite of pneumatic-based actuators. In previous years, the mechanical team heavily relied upon servo-based mechanisms, which were often difficult to debug and tricky to integrate with the rest of the robot stack. This year we chose pneumatic-based mechanisms because they have faster response and cycle times, reduce the complexity of our actuators, and increase reliability during competition. Furthermore, pneumatics don't short or react dangerously with water, and thus they are much safer than comparable motor-based systems.

1) Claw: Cthulhu v2's claw is significantly more versatile than its predecessor. Our original claw only had two jaws, while our new design has four. The quad-jaw design allows our claw to obtain a secure grip on a wide variety of objects. Furthermore, the jaw is actuated by a piston rather than a servo. Similarly to the torpedo launchers, this shift away from servos resulted in a decrease in the overall system complexity while improving reliability.

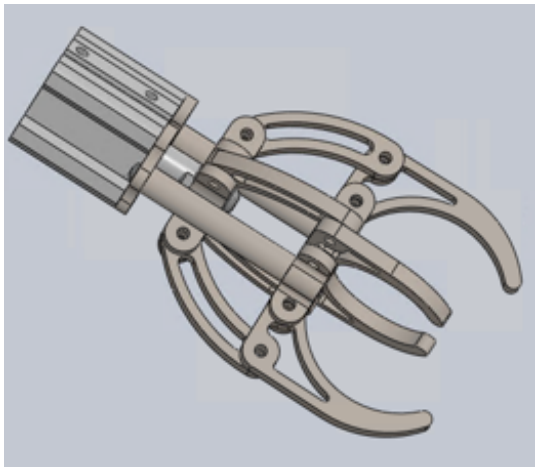

Fig. 4. Piston-actuated Claw

2) Torpedo Launchers: In previous robot designs, we utilized spring-loaded torpedos actuated by servomotors. These designs were unnecessarily complex and unwieldy. In order to generate sufficient power, we had to utilize springs with large spring constants, and the resultant stiffness made it very difficult to cock the torpedos. Furthermore, the springs would often separate from the launcher body during the firing process. Pneumatic actuators posed a safer, less complicated solution.

Via a single puff of air via a single-direction solenoid, our new launchers can now consistently launch 3D-printed torpedo's. The lack of moving parts makes the new launchers much less prone to failure, while also requiring less maintenance (we no longer have to test the electrical connections or debug servo Arduino code).

Due to the effectiveness of the core launcher design, we chose to use a set of downward-facing

torpedo launchers for the Marker Dropper task.

B. Electrical Subsystem Novel Aspects

The electrical team is the unsung hero of Duke Robotics Club; their work allows other subsystems to abstract away the low-level implementation details and simply trust that the electrical components on board simply will work when needed. Due to the pandemic, the electrical team had very limited access to the physical robot. However, the team continued the theme of abstracting away complex systems by taking a renewed focus on low-level software. Significant strides were made to streamline low-level driver code. Novel new developments include a comprehensive CI/CD infrastructure to facilitate work for the software team, as well as a revamped acoustics platform.

1) CI/CD: Due to the largely virtual nature of the club's work this year, it was especially critical that our teams had a robust Continuous Integration/Continuous Development (CI/CD) system to enable our software teams to rapidly develop, iterate, test, and deploy. The electrical team developed a comprehensive CI/CD platform that alleviated the bulk of frustrations experienced during previous years (e.g. inconsistent development environments, out-of-date dependencies, buggy code, and repetitive manual tasks).

The first key addition to our CI/CD stack is Docker, the industry-standard containerized platform that effectively enables each Robotics member to mirror the Linux operating system aboard the Robot on his/her own local computer. All dependency versioning and package installation is handled under the hood, thus removing a large barrier of entry for new Robotics members. Continuing the theme of automation and abstraction, a build script was created to abstract away the time-consuming and complicated process of starting up our multiple ROS workspaces.

Our CI/CD infrastructure also places a significant emphasis on code quality and correctness. We have a build & test pipeline which runs on a nightly basis, ensuring that all code builds and passes basic unit tests, integration tests, and linting (code-style) tests. This allows us to quickly diagnose any errors or bugs missed during development. More importantly, it ensures that our codebase is almost always ready to be deployed into production.

2) Acoustics: Cthulhu v2 uses two separate arrays of four omnidirectional hydrophones to locate acoustic pingers. We found through testing that we needed two sets of hydrophones to obtain an accurate distance estimate. Our hydrophone data starts out as analog data which is then fed through a Saleae logic analyzer to attain digital data for our algorithm. The location algorithm involves two major steps.

First, we use one array to obtain an initial guess for the octant in which the pinger is located relative to Cthulhu v2. The four hydrophones are generously spaced 0.3 meters apart in a right triangular pyramid configuration. Using a Butterworth bandpass filter [5] on the desired frequency, we obtain the time differences between the pings in the x, y, and z axes as they reach each hydrophone. After performing a cross-correlation to obtain the time differences, we can derive our "guess octant".

Next, we use our guess and the other hydrophone array to precisely locate the pinger. The second array consists of four miniature hydrophones spaced just millimeters apart in a square. Using synchronized readings from the hydrophones, a Short Time Fourier Transform is used to obtain a magnitude and phase list. After several window selections, a final small window is determined based on the stability of the phase difference between each hydrophone pair. The phase differences and the guess from the first step are inputted into our Time Difference of Arrival algorithm to locate the exact horizontal and vertical bearings for the pinger.

We were able to test our algorithm virtually with a data generation program, and we also developed a wrapper class for the acoustics algorithm. This integrates acoustics nicely into the rest of our codebase, abstracting away many of the complexities of the acoustics interface. Task planning merely must specify a few parameters such as the target frequency, and the acoustics algorithm will run under the hood and output the estimated pinger location.

C. Software Subsystem Novel Aspects

1) Computer Vision: The past two years have been a breakthrough period for the computer vision team; we developed version 1 of Detecto [1], an accessible, performant object-detection deep-learning framework. Detecto uses a machine learning model called Faster R-CNN [4] – an industry-standard architecture for segmenting and classifying images. We actively maintain Detecto as an open-source library, and it has since received more than 30,000 downloads on Github. Continuing this momentum, the team's primary focus this year was stereo vision, which allows us to identify the 3D locations of target objects. Detecto generates bounding box predictions for target objects, and with stereo vision, we now can estimate their depths as well. This information is crucial for the task planning team, which not only depends on position but also depth data to ascertain whether Cthulhu v2 is within an acceptable range of an target object (e.g. a pipe that needs to be gripped by a claw).

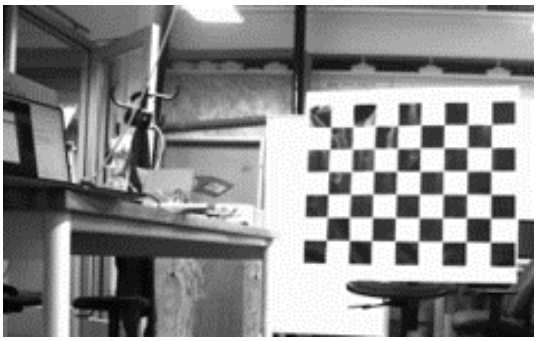

Fig. 5. Original Image Taken With Right Camera

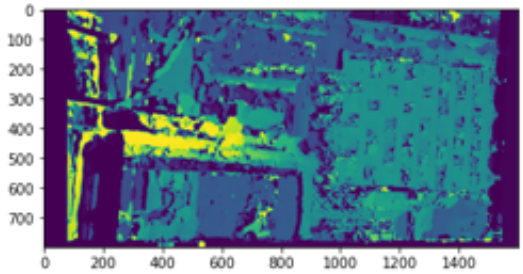

Fig. 6. Calculated Disparity Map of Original Image

Stereo vision is possible due to Cthulhu v2's two frontward facing cameras. We first collected calibration images, which involved simultaneously capturing an image from each camera of a checkerboard held at varying angles and depths. We then performed stereo camera calibration, which provided the relevant matrices to rectify image pairs. We then used stereo semi-global block matching (SGBM) with OpenCV to create a disparity map and convert it to a depth map for each image pair. Finally, to obtain depth measurements for target objects, we created a binary mask for each bounding box prediction, applied this to the depth map, filtered outliers, and calculated the median depth value for the object.

2) Controls: The controls team's primary goal is to provide a simple software interface to control the movement of the robot. Ideally, the task planning team should be able to issue a simple command for Cthulhu v2 to move to a specific location, and the controls algorithms will handle the low-level implementation details and hardware interactions under the hood.

This year, we revamped our controls algorithm to utilize "cascade control" [3] via nested PID loops (one for position, and one for velocity) rather than a single position PID loop. The outer position loop guides in the correct general direction, and the additional velocity loop corrects any local disturbances.

The new algorithm works as follows: First, the robot's instantaneous state information is fed into the top-level position PID loop (to ensure we are headed in the correct direction). The output of the position PID is then fed into the velocity PID loop (to ensure that locally, we are not getting pushed off course). Cthulhu v2 then gets its thruster power adjustment. The inner velocity loop runs at a much higher frequency than the outer position loop, so it quickly corrects for local disturbances. Linear algebra is used to calculate the thruster power allocations, based on a matrix of outputs from the PID loops and a static matrix which models the torque exerted by each thruster.

A significant advantage of using an additional velocity PID loop is the ability to issue desired velocity setpoints. Using a simple command, we can bypass the position PID loop and tell Cthulhu to maintain a specific heading indefinitely. This is extremely useful for style tasks such as the barrel

roll (where we just apply velocity control along the yaw direction). Cthluhu's new nested PID loops afford us much more granular control over the setpoint convergence behavior of our robot, and they also make the robot significantly more resilient to currents and disturbances in the water.

3) Task planning: We briefly discussed our new abstract, bottom-up task planning architecture in *Competition Strategy*. Now, we consider the high level task of moving through a gate, shown in Figure 8. Movement tasks (highlighted in blue), such as "move to a desired local position" or "rotate," are fundamental tasks. Decision tasks (shown in diamonds), such as "determine if the robot is horizontally centered with respect to a target object," are another type of fundamental task. Based on the output of the decision task, the robot state will transition to another task accordingly. Naturally, these fundamental movement tasks can be combined, reused, and executed concurrently to accomplish a variety of high level tasks (e.g. move through the gate, touch buoys, etc.). By treating each high-level competition task simply as a combination of many basic tasks, the software team can easily adjust and fine-tune specific tasks without breaking others. Having a toolbox of modular fundamental tasks also means that we are well-prepared to tackle any new competition tasks that may be announced in the future.

4) Simulation: Given that much of our work was done virtually this year, our simulation platform was a crucial alternative to in-person pool testing. We revamped last year's simulation code, which largely centers on realistically simulating the underwater environment. For every frame: gravity, linear drag, and rotational drag are applied on the center of gravity; buoyant force is applied on the center of buoyancy; and thruster forces are applied on the position of each respective thruster. We made several notable improvements to this core platform. First, we added a simulated pressure sensor, allowing for more accurate control in the z-axis when testing. Second, we implemented a pseudo-computer vision system, which provides simulated CV data for the virtual underwater environment. This works by projecting objects within the simulation onto a 2D plane parallel to the camera and calculating their bounding boxes.

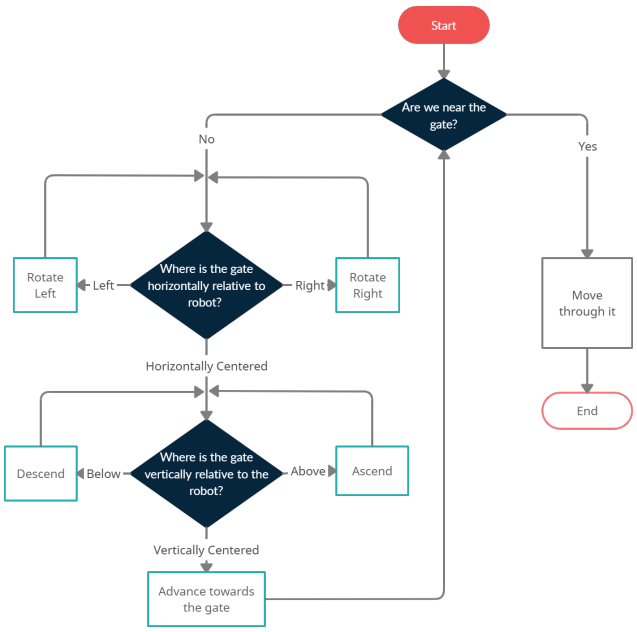

Fig. 7. Flowchart of Gate Task

This "mock" CV was crucial in allowing our task planning team to test their algorithms. Third, we modified the communication pipeline of the simulation, moving much of the buoyancy and drag calculations from the simulation engine's native Lua code to object-oriented Python scripts on the host machine. This resulted in better performance as well as clearer, more accessible code.

III. Experimental Results

A. Mechanical

Despite COVID-19 restrictions, all mechanical designs underwent thorough testing. During the latter half the spring semester, the mechanical team was able to test the new frame along with all the actuators in a large sink in our lab. The first round of testing for our pistons and pneumatics simply involved hooking them up to an Arduino, checking for airflow, and identifying any leaks. Eventually, we were able to test all actuators in the sink. We found that our pneumatically-powered mechanisms actuated quickly and reliably. We also tested our claw; while our new linear piston design was a marked improvement upon our previous servo-based design, our claw is not yet accurate enough to consistently pick up and drag objects. This is clear area of improvement for next

year.

In addition, we tested the new rectangular frame and its ability to achieve 6 degrees of freedom. Similarly to the actuators, frame testing occurred in the large sink in our lab, and Cthulhu v2 was able to move in all 6 degrees of freedom. We found that our 8 thrusters were dangerously exposed, so we designed and 3D-printed thruster guards to prevent damage during competition. We first utilized CAD FEA (Finite Element Analysis) to validate our models. Then, we used mallets to test the structural integrity of the fabricated designs. The thruster guards proved to be sufficiently strong.

The mechanical team also validated the new rectangular frame design and thruster placement in our simulation. We created an URDF model of our new frame design, and upon simulation, we found that our roll, pitch, and yaw movement all performed exceptionally well. Not only were we able to move freely in all 6 degrees of freedom, but we were also able to use significantly less power (as compared to Cthulhu v1) for roll movements due to the increased torque provided by the widely-spaced thrusters.

B. Electrical

The electrical team successfully integrated a number of new sensors in the past year. First, we fully integrated low-level camera code into our ROS pipeline. Previously, our camera driver code was housed entirely separately from ROS, which made it challenging to pass data, as well as limiting flexibility to use new cameras. We implemented the ROS interface noted in the ROS image pipeline [2], allowing us to have full control of the ROS ecosystem. As proof of this flexibility, we were able to implement image rectification using built-in ROS packages without needing to account for the low-level details. Furthermore, we validated the flexibility of our new camera code by swapping our old cameras with brand new cameras from a different manufacturer. The process was seamless and required minimal modification to the ROS driver.

We also integrated a pressure sensor which yields a pressure reading that can be converted into a highly accurate depth (z-value). This depth value is significantly more reliable than the depth value we derive from the our Doppler Velocity Logger (in the past, we relied on a localization package to roughly estimate depth based on z-velocity readings from the DVL). The electrical changes were minimally complex; we split the existing I2C connection to the robot's multiplexer in two, allowing us to combine the pressure sensor with a level shifter. We validated that the pressure sensor was successfully integrated during an in-person pool test. We placed the robot at various depths and compared the calculated depth reading with the actual ruler-based measurement. The standard deviation between the values was less than 5%.

C. Software

The software team attributes much of its success during the pandemic to its robust custom simulation platform. The virtual simulated environment simulates a multitude of physical factors such as buoyancy and water currents, allowing us to convincingly model conditions similar to those at San Diego's TRANSDEC.

The controls team heavily relied upon the simulation to refine and tune our new nested PID loop algorithm covered in *Design Creativity*. We developed an automated testing package to tune our PID constants via the Ziegler Nichols method [3], which performed reasonably well. A series of automated scripts test varying combinations of PID constants for x/y/z, and roll/pitch/yaw. We tested the output PID constants in simulation, and we validated that the robot was able to quickly maneuver to any desired local position or velocity while minimizing the amplitude of oscillations around the setpoint.

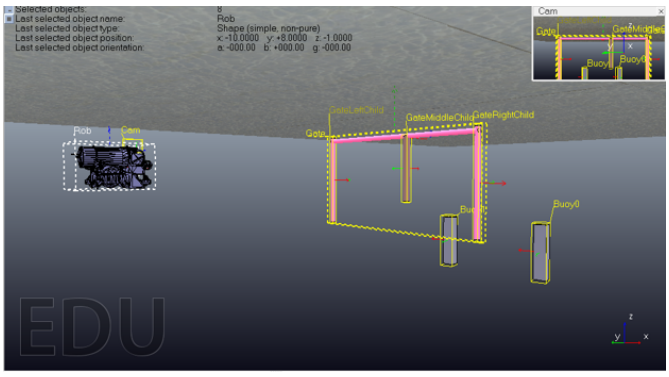

Fig. 8. Simulation Environment With Mock Computer Vision

The task planning team likewise performed the bulk of testing in simulation. We designed a number of virtual models, such as a gate, buoys, and pipes, and we laid out these objects in the simulated environment to mirror a competition setting. Using our "mock computer vision" shown in Figure 8, task planning was able to treat the virtual environment as if it were its fully-fledged real life counterpart. Within simulation, task planning demonstrated that Cthulhu v2 could successfully complete the gate task, style tasks, and buoy task.

Duke Robotics had limited opportunities to conduct testing in-person at a pool. During these pool-tests, our main priority was to validate that all core functionality and basic movement worked flawlessly, rather than focusing on trying to overengineer a specific competition task. We validated the water-tightness of our frame, and we confirmed that all critical electrical components were functional. Furthermore, we showed that the robot could hold depth, move to any desired position, maintain a constant desired velocity, and even do a barell roll. All in all, our pool tests helped us conclude that the core robot systems were healthy and reliable, giving each subteam confidence to continue layering on complexities.

D. Lessons Learned

The past year has been replete with new experiences, challenges, and lessons learned. Prepandemic, robotics members would spend countless hours each week designing, programming, soldering, and tinkering with the robot, all while working together under the same roof. The shift to a virtual-first work environment and the inability to perform in-person work on the robot was unprecedented. However, Duke Robotics adapted well to the new paradigm, in large part because we actively practiced the following key principles:

- **Be flexible.** The pandemic has taught us that anything can happen, and thus we have prioritized flexibility in nearly all aspects of our work. We are flexible with club meeting times to accommodate people in different time zones. Our codebase is organized into flexible modules, enabling us to quickly iterate, test, and deploy. Our new robot capsules are wide and flat to provide flexibility for any modifications or hardware additions. All of our work is driven by flexibility in mind, such that we are prepared to quickly adapt to any unforeseen obstacles.
- **Set clear short-term goals.** Our long-term vision is to win Robosub, and we must take gradual, incremental steps to achieve this vision. By setting short-term goals each week, our club was able to stay focused and productive amid the chaos of the pandemic. Explicitly stating and delegating short term goals allowed our members to take ownership and stay on track toward our long term goal.
- **Communicate, communicate, communicate!** This year, more than ever, it was important that all members of the club stay in constant communication. Subteam leaders frequently met via Zoom outside of regular meeting times in order to plan and coordinate. All members stayed in constant touch over Slack, giving weekly updates to ensure everyone was on the same page about completed tasks and new undertakings. This consistent communication allowed members to stay informed not only about their own subteam's progress, but also the progress of the club as a whole.

ACKNOWLEDGMENTS

Duke Robotics Club is housed within Duke University's Pratt School of Engineering. We gratefully acknowledge the Duke faculty and administrative staff who have helped and continue to help make the club a success. We are indebted to Pratt Director of Undergraduate Student Affairs Jennifer Ganley, our advisor Professor Michael Zavlanos, and the Engineering Alumni Council. Furthermore, we owe our continued success to our long-time sponsors: The Lord Foundation, Duke Student Government, General Motors, and SolidWorks. Finally, we thank RoboNation, whose commitment to RoboSub empowers student engineers like ourselves to explore our passion for robotics to the fullest extent.

REFERENCES

[1] A. Bi, "Welcome to Detecto's documentation!". ReadTheDocs.io. https://detecto.readthedocs.io/en/latest/ (accessed May 8, 2021).

[2] A. Blasdel, "Image Pipeline," ros.org. http://wiki.ros.org/image_pipeline (accessed May 25, 2021).

[3] O. Arrieta, R. Vilanova, and P. Balaguer, "Procedure for Cascade Control Systems Design: Choice of Suitable PID Tunings," International Journal of Computers Communications & Control, vol. 3, no. 3, p. 235, 2008.

[4] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137–1149, 2017.

[5] Z. Li, "Design and Analysis of Improved Butterworth Low Pass Filter," 2007 8th International Conference on Electronic Measurement and Instruments, 2007.

Appendix A: Component Specifications

Component	Vendor	Model/Type & Specs	Cost (if new)
Buoyancy Control		Ridgid Foam Blocks	
Frame	8021 Aluminum, custom		
Waterproof Housing	Polycarbonate, custom		
Waterproof Connectors	Subconn + Seacon	Wet-mate Connectors	
Thrusters	Blue Robotics	T200-Thruster-R1-RP	
Motor Control	Blue Robotics	Basic ESC	
High Level Control	Arduino	Nano w/ PWM Multiplexer	
Actuators	Hitec	D646WP Waterproof Servos	
Propellers	Blue Robotics	Included w/ T200 Thrusters	
Battery	Turnigy	HC 5S 12C 16000mAh Lipo	
Converter	Kohree	DC/DC 36V/12V	
Regulator	N/A		
CPU	NVIDIA	Jetson TX2	
Internal Comm Network	TP-Link	5 Port Gigabit PoE Switch	
External Comm Interface	NETGEAR	Nighthawk R7000	
Programming Language 1	Python	2.7	
Programming Language 2	C++		
Compass	Built into IMU + DVL		
IMU	VectorNav	VN-100	
Doppler Velocity Log (DVL)	Teledyne	Workhorse Navigator 1200	
Vision	Edmund Optics RMA Electronics	Allied Vision Mako G-234C Tamron M112EM08	
Acoustics	Teledyne	TC4013 omnidirectional	
Manipulator		3D-Printed ABS Plastic	
Algorithms: vision		Machine Learning: Resnet 50 Architecture based on Faster RCNN	
Algorithms: acoustics		Butterworth Filter Cross Correlation Time Difference of Arrival Short Time Fourier Transform	
Algorithms: localization and mapping		Extended Kalman Filter SLAM w/stereo cameras	
Algorithms: autonomy		In-house task planner	
Open source software	Docker + ROS	ROS Melodic	
Team size (number of people)		28	
HW/SW expertise ratio		45/55 (13 HW, 15 SW)	
Testing time: simulation		180 hours	
Testing time: in-water		10 hours	

Appendix B: Community Outreach

Mentorship: We at Duke Robotics Club are deeply passionate about sharing our love for engineering with the local community. In recent years, we have worked extensively with students ranging from elementary school to high school to foster interest in engineering.

During the pandemic, we doubled down on our outreach efforts. In the past year, we began working with Families Moving Forward (FMF), a transitional housing program in Durham, NC which serves underprivileged families with children. Duke Robotics members work closely with the teens at FMF, who range in age from 10-16, on a biweekly basis. Each workshop is held over Zoom and focuses on a particular engineering skill. While the sessions are delivered virtually, we develop the content such that the teens gain hands-on experience by working with software tools to program/design projects. Some topics we have covered include CAD modeling, Website Design (HTML/CSS), and basic machine learning. We have also delivered workshops that span multiple weeks, including an Arduino/circuitry project which involved assembling and coding mini robot cars. All in all, this experience has been incredibly rewarding for both Club members and the students, who have been able to explore exciting engineering skills.

In 2018-2019, members of the Duke Robotics Club mentored a brand new team comprised of local Durham high school students, guiding them in creating a robot for the FIRST Tech Challenge, and also helping to bring them to the finals of the regional competition for the FIRST Robotics Competition. The members joined this FRC Team 6426 Robo Gladiators multiple times per week to help them strategize, build, and program their robot. Advancing to the finals of the regional competition as a rookie team, they outperformed many veteran teams. Looking forward to next season, the Robo Gladiators hope to extend their successes and become national champions, and the Duke Robotics Club looks forward to once again helping them succeed.

In 2019-2020, we continued the mentorship program with the same team. Despite the season getting canceled early due to the pandemic, it was still an amazing experience for all involved, and the team did get to attend their first competition.

Within Duke: Duke Robotics Club prides itself in cultivating a passion for engineering in all undergraduate and graduate students. We have a no-experience-required policy to join the club; members of any experience level are welcome to join and are encouraged to pick up skills on the fly. In the fall of 2020, we completely overhauled our onboarding program to complement the virtual-first environment. Team leads developed a comprehensive virtual curriculum (featuring a multitude of video walkthroughs, wiki's, and mini-projects) to aid in new members' learning process. This curriculum was well-received, and was critical in allowing us to introduce nearly 30 prospective new members to our robot, many of whom chose to stay with the club in the past year.