Technical Overview of the Continued Development of Autonomous Underwater Vehicle LANTURN

Andrew Ye (ME, Team Captain), Dervin Lopez (ME), Mark Raspopov (CS), Andrew Gill (ME), Alynne Wong(ME), Cosme Lavalle (EE), Roger Roldan (CS), Nehemias Orellano (EE)

1. ABSTRACT

RoboSubLA continues to utilize lessons learned from previous years' competitions to improve our autonomous underwater vehicle; LANTURN. Following the troubles with getting LANTURN dive-ready for the 2023-24 competition season, achieving vehicle readiness was the primary focus for this years' development process. With system modularity and ease of iterative development as priority goals, the mechanical and electrical subteams of RoboSubLA worked to create new upgrades on LANTURN to a usable state. Meanwhile, our software subteams worked to develop a future-proofed architecture for our autonomous vehicles, leveraging new technologies and open source tools to provide a clear path for management and upgrade in the future. New software components were integrated into our stack, including a localization system for discerning vehicle state, a mapping system for storing environment state, and a watchdog (referred to as the monitor) to handle unexpected system conditions and command appropriate responses to prioritize vehicle safety.

2. COMPETITION GOALS

Our competition strategy for this year focuses on leveraging the strengths of each system to accomplish different tasks in parallel, in the most efficient manner possible. LANTURN has been designed with capability in mind, being able to complete all competition tasks solo if needed.

Specifically, LANTURN is designed with the intention to complete Collecting Data, Navigate the Channel, Drop a BRUVs, Tagging, and Ocean Clean Up.

3. DESIGN STRATEGY

A. MECHANICAL

I. FRAME

Many vehicle design features have carried over from the 2024 competition for the configuration of our 2025 vehicle upgrades.

LANTURN's improvements for 2025 include a new ball dropper, robotic arm, and torpedo, to enable the completion of the tasks required. These additions were easy to integrate thanks to the modular mounting system on LANTURN's underside, made of sheet metal aluminum for its light weight and strength. LANTURN was created with the technical design strategy that prioritized modularity and adaptability, as we have learned the ability to meet new challenges is one of the most important for a robotic system.

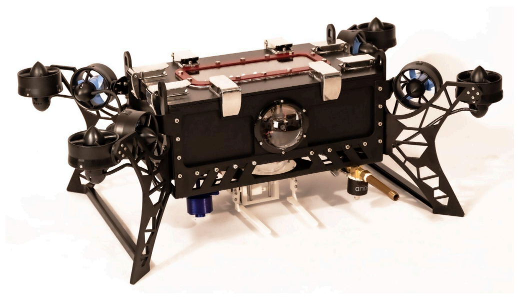

Figure 1. *LANTURN Photo*

The hull of LANTURN is made of 6061-T6 aluminum to help with cooling of electronics. It features several material cutouts throughout that have the purpose of weight reduction. Pressure vessel FEA reports were conducted to examine how the cutouts would affect the integrity of the hull. Most of the machining of the hull and lid were done on a 3-axis CNC and a horizontal mill. The watertight enclosure of the hull was designed using face type O-rings and gland cavities. A vacuum test was conducted to check for leaks in the watertight enclosure.

The design of the electronics carriage revolved around accessibility and functionality. The design criteria were that it needed to have enough space for all the internal electronics including a large battery, to be removable from the hull, user friendly for diagnosing electronics, and the battery must be rapidly exchangeable. The structure needed to be strong enough to support the heavy 3.89 lb battery. Thus, a combination of carbon fiber rods, laser cut acrylic, and 3D printed parts were used for the assembly.

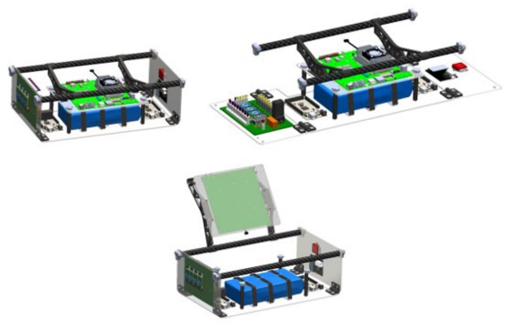

Figure 2*. The electronics carriage features 3 modes to meet the design criteria. Mode I, is for transport. Mode II, is for troubleshooting electronics. Mode III, is for quickly swapping the battery.*

II. ACTUATORS

A new set of actuated systems was designed for LANTURN in order to facilitate better integration with the systems for this years' tasks. The ball dropper, claw, and torpedo launcher have all been redesigned to be more useful, and in the case of the torpedo launcher, eliminate the need for single-use CO2 cartridges which required reloading prior to every run.

The new designs are all 3D printed and utilize waterproofed servo motors for actuation, simplifying the control system and eliminating high powered solenoids from the system entirely.

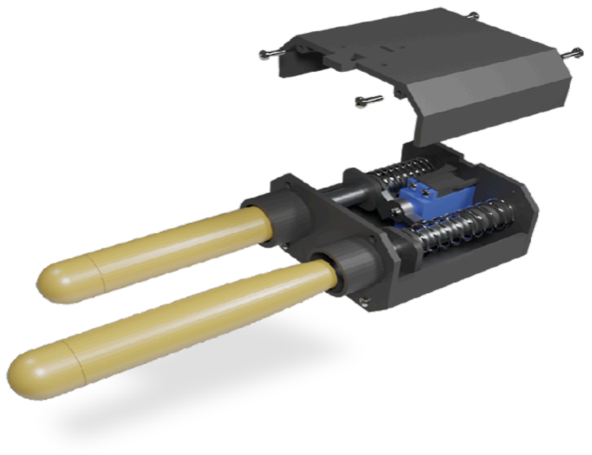

Figure 3*. Redesigned mechanical torpedo launcher*

In the case of the ball dropper, the markers have a tear-drop design to reduce the drag force, and a fishing weight can be inserted inside of them since the markers have threads to open and close them. The markers are released from their housing using a lever, which is activated by a waterproofed servo.

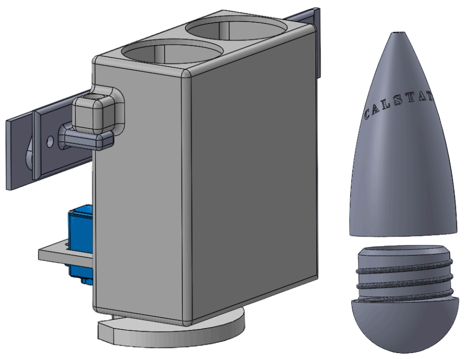

Figure 4*. Ball dropper and marker CAD design*

The torpedo launcher will utilize springs to generate the force necessary to release the torpedoes. Similarly to the ball dropper, it will also use a lever to release a torpedo using a waterproofed servo.

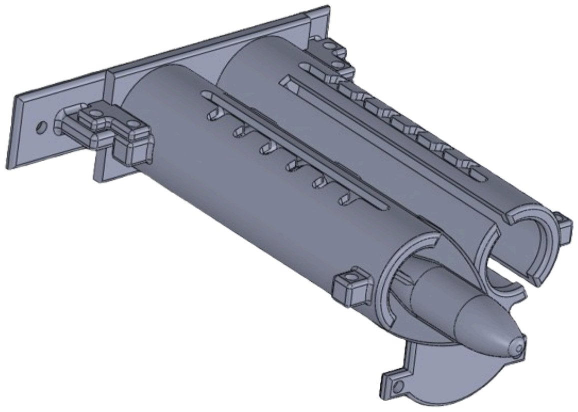

***Figure 5**. Torpedo launcher's CAD assembly*

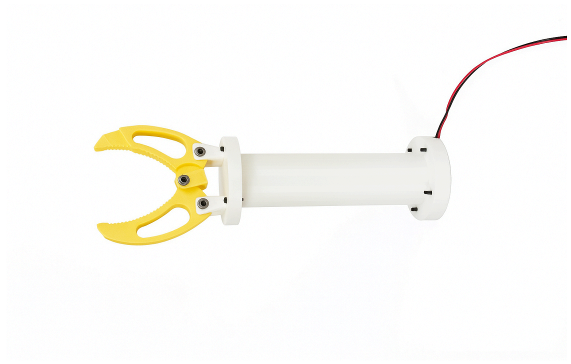

***Figure 6.** Redesigned Claw*

This year we designed and 3D printed a lightweight robotic arm with a functional gripper claw, using a DC motor for movement and a linear actuator to control the opening and closing of the claw. To ensure waterproofing, we 3D printed a custom sealing component using flexible TPU filament, preventing water from entering the casing, and also using o-rings. The entire system was engineered to be both cost-efficient and practical for use in environments where durability and affordability are key.

B. ELECTRICAL

I. POWER

The electrical system on both robots consists of a thruster signal routing board, power distribution board, microcontroller, and sensor suite which are powered by a 14.8V 20Ah LiPo battery.

The sensor suite includes:

- VectorNav VN-100 IMU
- Blue Robotics Bar30 Barometer
- Aquarian AS-1 Hydrophones
- Blue Robotics Ping Sonar
- Blue Robotics USB Camera
- Teledyne Wayfinder DVL

Continuing with the previous year, our bot sports an electrical loadout with the inclusion of a Jetson Orin Nano rather than a Jetson TX2, allowing us to access modern tools such as ROS2 natively on the Ubuntu 22.04 operating system. Power for all auxiliary components, such as the controls board, thruster board, and interface boards for sensors and actuators is provided by similar regulator electronics to the 2024 system, as we still have power margins which can be used.

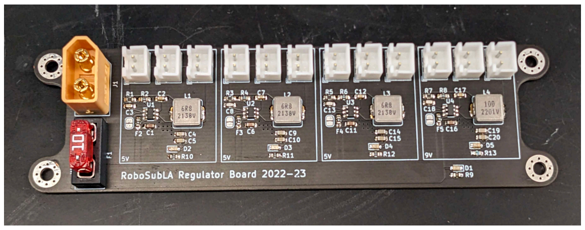

***Figure** 7. Custom Power Regulator Board*

In order to solve the overheating problem of our robots from previous years, LANTURN was mechanically designed to have the frame as a heat sink.

C. SOFTWARE

I. CONTROLS

From previous years, in order to streamline current and future developments, a custom controls carrier board was developed which includes a Teensy 4.0 module with a CORTEX-M7 32-bit ARM microcontroller onboard running at 600 MHz. This allows us a significant headroom for software development, as even somewhat inefficient implementations will run at far greater speeds than needed. The controls carrier also includes locking board to board connectors for multiple communication buses, including I2C, SPI, and UART for communication with sensors and other subsystems.

In order to simplify the development of both vehicles' control software, a significant portion of time this year was dedicated to developing subsystems that allow identical software to be deployed irrespective of the differences between vehicles. To accomplish this, many components of the software are abstracted by one or multiple layers, and configuration parameters per frame are stored in each vehicle's control board EEPROM.

The most obvious component of this vehicle androgyny is the thruster solver, a clever implementation of matrix multiplication that allows a six-component control vector (comprised of 3 axes each force and torque) to be converted into an output vector with components corresponding to individual thruster force requirements, dynamically allocated based on vehicle thruster positioning. Each thruster's individual force is then converted to an output signal compensated for battery voltage and forward/reverse thrust differences, and sent to the PWM generator onboard each vehicle's thruster board.

II. AUTONOMY

Building on the success of implementing behavior trees [1] in our software stack from previous years, we have completely retooled our development strategy for the vehicles' shared autonomous system. Our autonomous system is structured now to break tasks into smaller tasks, as small as possible, and let the behavior trees framework combine them for complicated behaviors. This departure from more "monolithic" task programming means we can easily and readily modify the parameters, or even the structure of any given tasks' tree, avoiding costly compilation time when testing many iterations of the software. This also makes it much easier to test individual components of the autonomous software, and allowed us to perform unit testing on each individual action and condition in our autonomous system to validate functionality before final task descriptions were available.

Our focus continues on accurate simulations of our systems to enable testing while we are unable to utilize pool facilities for the physical robots. While dedicated projects such as Project DAVE [2] are no longer updated enough to utilize with our backend system, modern tools such as Ignition Gazebo are more than capable of picking up the slack. To test the high-level functionality of our autonomous system, we first developed a highly simplified simulation of the vehicle dynamics, responding exactly to the demands of our velocity & position controllers, and having a perfect understanding of the current vehicle and environment state. Further development is ongoing into a more realistically based simulation, with appropriate sensor and state noise, implementation of buoyancy and hydrodynamics for accurate dynamics simulation, and the inclusion of the computer vision subsystem to process simulated camera images from the vehicles' multiple views. We also plan to test the actual vehicle hardware utilizing HITL testing, connecting a dedicated simulation box to the onboard computers of both systems and feeding simulated state estimates to the onboard controllers, pulling vehicle commands out to perform the simulated dives we need to validate functionality.

III. COMPUTER VISION

The implementation of computer vision for LANTURN relied on the You Only Look Once (YOLO) Version 7 Convolutional Neural Network (CNN) in addition to the Python library OpenCV. YOLOv7 was decided with considerations to the Nvidia Jetson TX2 device within the submarines. YOLOv7 is the most recent official release which is substantial since it derives its architecture from its preceding instances, such as YOLOv4 which we used last year, and optimizes for the prioritization outperforming "... DETR, Deformable DETR, DINO-5scale-R50, ViT-Adapter-B and many other object detectors in speed and accuracy" [3]. In determining which CNN to use, we revisited YOLOv4 with the same annotated dataset of 1000 images. In the training process, we learned that the mean average precision of the object detection model trained with v4 was marginally better than v7 (96.423 vs. 96.318), but much slower in training and inference. Additionally, Version 7 is defined in Pytorch which is lightweight and efficient for the computational constraints. For OpenCV, we leveraged various algorithms for applying augmentations to the images in our dataset including, but not limited to Gaussian blur, cropping, shifting, adding static and grayscale. Electing to keep augmentations more cosmetic and not changing orientation was intentional to enable us to create orientation (90°, 180°, 270°) classes for each object.

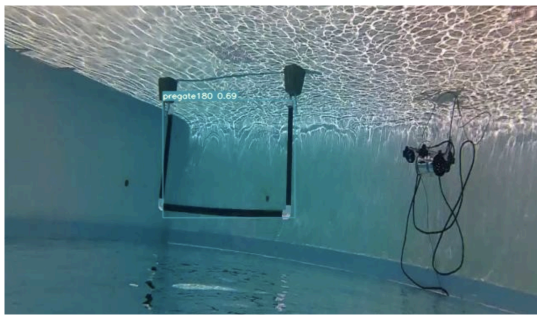

Figure 9. *Example of Additional Class to Determine Object Orientation*

Doing so enables us to pass positioning information to determine navigation and adjustments of the submarine as needed, which we lacked last year. Moreover, we used OpenCV to extract and save frames from LANTURN's cameras. In preparation and training purposes, we reconstruct the provided obstacle objects, take underwater videos to best simulate competition conditions, extract at least 1000 frames before augmentations, annotate and then train to build our custom models.

4. ACKNOWLEDGEMENTS

Our team would like to thank our faculty advisors, Dr. Salvador Rojas, Dr. Mike Thorburn, Dr. Mark Tufenkjian, and Richard Cross for their invaluable inputs to our project during review sessions and team meetings. We would also like to thank Cal State LA and the ECST department for their continuous support of our RoboSub team. Our team is extremely grateful to our sponsors who supported this project through hardware donations and generous discounts. These include the Office of Naval Research, VectorNav, Blue Trail Engineering, Blue Robotics, MathWorks, and Dassault Systemes. Thank you for making the development of our 2025 RoboSub vehicles possible!

5. REFERENCES

[1] M. Colledanchise and P. Ögren, Behavior Trees in Robotics and AI: An Introduction. Boca Raton: CRC Press, 2020.

[2] Project DAVE, https://field-robotics-lab.github.io/dave.doc/ (accessed Jun. 16, 2023).

[3] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," arXiv.org, https://arxiv.org/abs/2207.02696 (accessed Jun. 16, 2023).