“ANDY” - Autonomously Navigated Discovery Yacht at RoboSub 2022

T. Giles, E. Ruiz, C. Alford, A. Morris, M. Huntoon-DeRoche, J. Leonard, L. Sadrin,
N. Azzopardi, S. Cooke, C. Rhodes, K. Fontanilla
Oregon Institute of Technology, 3201 Campus Dr, Klamath Falls, OR 97601

I. Abstract

The goals of the AUVSIR Club at Oregon Institute of Technology (Oregon Tech) encompasses; obtaining learning objectives and experiencing hands-on knowledge and tasks while strengthening theoretical aspects in highly collaborative formats to excel at the 2022 RoboSub competition. Our purpose this year is to design and manufacture an effective autonomous robotic submarine capable of navigating environments by referencing spatially memorized landmarks, fine-tuning our sensor input and passing levels of AI learning through these avenues, and accomplishing as many tasks as possible in the 2022 RoboSub competition.

II. Competition Strategy

2.1 Mission Control

The software side of the project had vast improvements due to the new contributions and consistent passion for it. Before ANDY can engage any sort of processing, the “Mission Control” will designate the list of commands it must do, however simple or complex. This way, specified tasks and goals will be remembered before submersion, and subsequently accomplished during the mission runtime. Mission Control encompasses any programs that need to be done on the onboard computer before the vehicle would descend into the water.

Most of the Mission Control takes place in the planning phases and pre-determined instructions of our “mission files”. These files have encoded text that our application reads, and can interpret into basic missions and tasks pertaining to competition and navigation. Most of these specific instructions are simple commands such as moving forward and rotating right, but more of the complex ones rely on vision processing and other external sensors during mission execution, such as:

‘Log Start Point’, ‘Return to Start’, ‘Move to Target’, or ‘Fire Torpedo at Target’.

Fig. 1: Mission-Based GUI(Deprecated).

Once the sub is turned on underwater, the Mission Execution would start when the vehicle is turned on externally and take the text file from Mission Control. Based on the instructions from the said file, the code would extract information from our vision recognition system, which calculates any important variables from the camera via OpenCV, such as the distance and vector between the RoboSub and its possible targets. Based on the results of the information, the code will tell the Arduino to make the motors run in intuitively calculated sequences to reach the assigned objective(s). This process is looped until all assigned tasks are complete.

2.2 Survivability

Our entire ROV design process has had a theme of sustainability and preservation tactics, so as to accomplish its tasks and mission even if it was in an unfavorable position or had a loss of certain systems. When systems fail, the software is able to

continue on and do limited data parsing and sensing in order to plan out how to do tasks at a "lower" level of operation if intuitively deemed necessary. If our sonar system were to fail, we could continue using our IMUs, cameras, and other sensors to finish what tasks we could.

III. Engineering Design

3.1 Chassis

"ANDY" has been worked on for almost three years, being methodically improved every year, as we've yet to find anything we can't improve for future effectiveness. The main chassis and hardpoints consist of five major components:

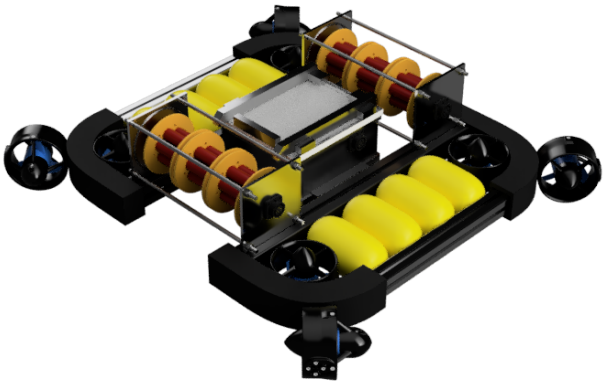


Fig. 2: Fusion 360 Render of ANDY.

Electronics Housing:

Found at the heart of the chassis is a central aluminum waterproof containment system designed to house and protect key electronic components. The body and lid are machined from aluminum stock, then supplemented with epoxy, polycarbonate, and rubber gaskets to create a functional, easily accessible housing.

Now machined from billet aluminum, it serves as a mounting platform for electrical components such as voltage regulators, companion computers, switches, probes, internal measurement units, sensors, and our main power management board system.

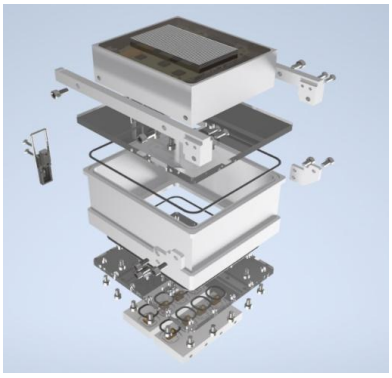
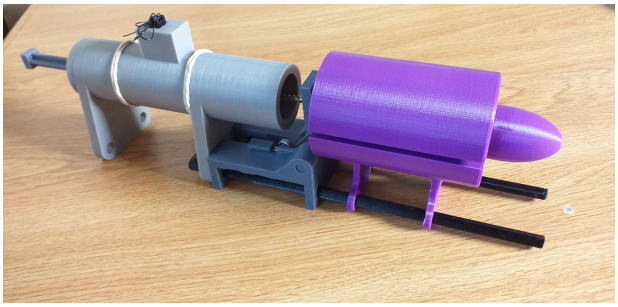


Fig 3. Electronics housing.

Previous designs required welding to construct the housing. Creating the housing from a single billet has allowed us to reduce weight while increasing pressure and depth capabilities. The Electronics housing consists of three main components. The lid, the potted ESC's, and the Polycarbonate sheets. The lid construction is dual purpose component machined from billet aluminum bolted to a polycarbonate base, potted Electronic Speed Controllers for thruster control, however, this year the ESC's are smaller, lighter weight, and using polycarbonate sheets allows for a visual inspection of the gasket surface.

Torpedo:

Fig 3: 3D printed assembled torpedo.



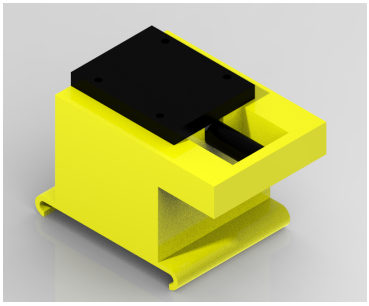
The self-propelled torpedo simply contains a CO2 canister in a more streamlined and physical fashion. Instead of having a sensor like last year, we have a simple CO2 "thruster" that is simply activated by a servo releasing an arm attached to a pin and hammer. The pin attached to the plunger system launches the torpedo by piercing the CO2 canister at precisely the right spot.

Marker dropper:

The marker dropper uses a mechanical system to drop two cylindrical markers held by a pin rod. The dropper system uses a high torque waterproof servo with a connector rod connecting to the pin. The servo then rotates pulling the connector rod and pin back releasing the two markers.
Using this method, the marker is dropped by using the cameras on board to signal when the submarine is in position. The use of this system allows the usage of two lightweight markers as well as a simplistic system with fewer moving parts.

The system besides the servo is entirely 3D printed out of PETG with a 0.2 mm layer thickness. The print has a 40% infill; this amount of infill allows for a robust design as well as limiting the empty space that could have water entering, making the sub heavier. The high torque servo is housed in a cavity of the print with a top lid so as to not allow for interference between the outside environment and the connecting rod.

Fig 4: Assembled view of marker dropper.



Robotic Arms:

This system is similar to that of a bottle recycler. It has two rollers, one driven roller, and a free-spinning roller. The goal of the system is for the bottle to be quickly pulled into the machine while the belts pull in the bottles. The entrance into the intake is adjustable as well as having an adjustable tension system to allow for bottles to come in at different angles, keeping a firmer grip on the bottles. The bottles will be collected into a housing bin formed with the hexagonal flooring of the system as well as the top lid. Once stored the bottles can be ejected out of the system by reversing the rotation of the motors allowing the release of the bottles. The bottle intake/arm system will be mounted underneath the center of the frame to allow a greater amount of free space for the modularity of the design.

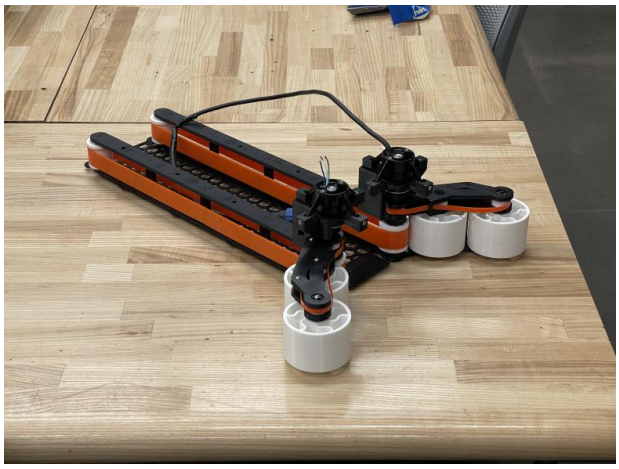


Fig 5: Arm assembly.

3.2 Embedded System Design

The purpose of this system is to control an autonomous robotic submarine that will be submitted in the RoboSub competition, but also to create a navigation library for students at the Oregon Institute of Technology for development on future subs and projects. For the autonomous submarine, it must be able to perform various advanced tasks requiring streamlined communication between hardware points and software objects without human interference. Thanks to the consistently increasing participation of CSET team members, we have been able to continue developing our evolving embedded computer and software system. The hardware package consists of a Raspberry Pi 4 to communicate with the sonar sensors, our stereo camera system, as well as the Arduino Mega and it's included hardpoints, such as our WT61P IMUs, T200 thrusters, water probes, etc. The software stems from the RPi on Python but grows out to communicate with C libraries for the sake of efficiency, as well as the Arduino through serial procedures ensuring rapid but balanced updating of current sensor data. Additionally, we have built a virtual simulation with 3D models in the past to make environments and scenarios for the AI camera/navigation system, and we do use it to

further develop task completion strategies when we find it necessary to use simulated data over real test data

Hardware:

The main electronics housing contains a primary Raspberry Pi 4B+ for main control and computation USB connected to a PhidgetSpatial 9dof IMU for independent, no-interference vector and positioning data, with an Arduino Mega for controlling 8x T200 BlueRobotics thrusters, arm servos, as well as extra sensor hookups if necessary. Every peripheral sensor or board is integrated into the system so that it could be removed and the rest of the system could still run at varying strengths. This was implemented in order to help us in certain situations, such as if we did not have the board at the time, or wanted to debug consolidated parts of the navigation system. This is a hardware-software symbiotic relationship system that can be tested and developed quicker and easier on other future robotic platforms.

Although the embedded system is not custom integrated into one another through a PCB or all-in-one board, the communication between our multiple boards/sensoors is very steady, with no problems having arisen from the directly connected serial communication lines. The Raspberry Pi is able to efficiently decode the data sent out and in of itself, whether it be to and from the Arduino Mega, or from the Phidget22 IMU over USB. When parsing data from a gyro connected to the Arduino Mega, there is no noticeable data loss when passing it onto the Raspberry Pi.

Software:

The basic navigation system is built around allowing our RoboSub to travel to any orientation matrix or position using coordinates and waypoints. We do not use many external libraries, as we want all the navigation code to be our own, with only some imported Python libraries for certain sensors or peripheral communication(BlueRobotics-Pinger, PySerial, OpenCV-Python). The code is written in such a way to ease the testing and development for our limited software team.

Our sensor capabilities have increased in recent years, as we now have more experience working with variations of sensors and figuring out not only what sensors work best for our situation, but what sensors we work best with. We currently use two internal IMUs to determine our rotation in space, but we plan on incorporating more if possible to not only increase the quality of orientation data, but to also ensure further mission sustainability.

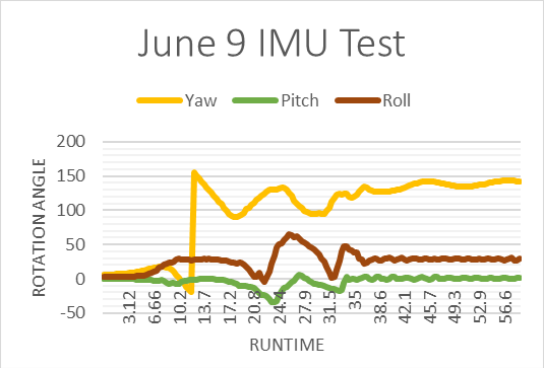


Figure 6: IMU Yaw, Pitch, Roll data received while doing a submerged Sonar+WT61P Test, PID orientating to Yaw=150.

The RoboSub is completely autonomous within a state-like system, where it reads commands generated from GUI that enables any person, programmer or not, to generate a list of commands or "mission" for our RoboSub to process and turn into logical navigation data. The sub starts every mission through either a start button on the top of the sub, or manually running a certain script in the console of the Raspberry Pi host computer.

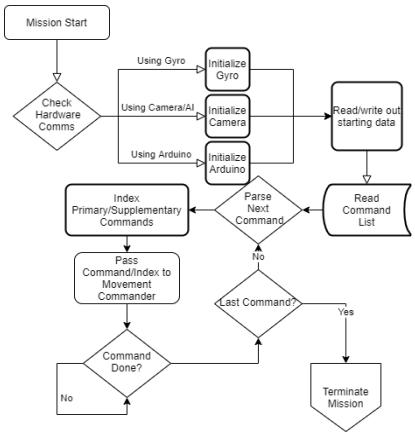


Figure 7: Simple Software/Mission Control Flowchart.

IV. Experimental Results

Simulation:

We were able to test certain parts of the software through tests in simulated environments, but we also have a simple framework for generating a coordinate map of the RoboSub's physical environment in real time for actual in-water testing and missions. The rendered environment has proven useful for vision testing, but the intuitive coordinate mapping is slowly being trained off of more testing and simulation data in order for it to be genuinely useful for our application.

Artificial Intelligence:

In the competition, our plan involved knowing the location and object type in the camera's view. For this, we had to choose an object detection architecture and train it to detect the objects in the competition.

Our process involves using transfer learning on top of the "CenterNet HourGlass104" architecture, trained on the COCO dataset. The everyday Objects In Context (COCO) dataset is a set of high-quality datasets for computer vision. After using the COCO dataset to train a good base for the model, we gathered training images for buoys and other objects in the competition by annotating video footage and using unity autogenerated examples.

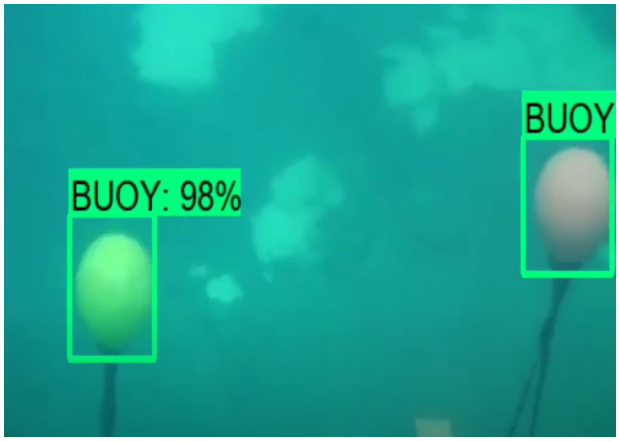


Figure 8: AI Vision System determining buoy confidence and location in relation to ANDY.

V. Acknowledgements

Thank NASA Oregon Consortium for sponsoring the project partially by NASA Oregon Space Grant (80NSSC20M0035). We would also like to thank OIT for partially sponsoring the project through RBC, NASA, and Oregon Space Grant Consortium Cooperative Agreement 80NSSC20M0035 for their support and encouragement.

Special thanks to Oregon Tech MMET Sr. Project Team, as well as the Oregon Tech AUVSIR Club members.

Our advisor Don Lee has been an excellent source of information and support throughout this process.

Student Members:

- Theodor Giles: Management, Embedded Lead
- Eduardo Ruiz: Amphibious, Waterproofing
- Chance Alford: Mounts, Waterproofing
- Austin Morris: Amphibious, Strength
- Lauren Sadrin: Treasurer, Amphibious Design
- JayCe Leonard: AI Lead, Software
- Mauricio Huntoon DeRoche: Design Lead
- Noah Azzopardi: Arms Design, Manufacture
- Scott Cooke: Arms Design, Prototyping
- Kai Fontanilla: Torpedo Design,
- Izzy Shepard: Torpedo
- Nick Costley: Electrical

Special Thanks:

- Dongbin "Don" Lee, Advisor, mentor
- Kapil Gangwar, Future faculty advisor
- Tim Pasang, MMET Department chair, mentor
- Ken Fincher, Foundation Vice President
- Dr. Nagi Naganathan, Oregon Tech President
- Tom and Dee Thompson, Foundation donors