Senior Design Final Report

Members:
Ricardo Medina
Albert Chew
Gabriela Cortes-Mejia
Heriberto Gonzalez
Horacio Mondragon
Brandon Pham
Sana Shaikh
Wilson Weng
Kevin Williams.

Faculty Advisor:
Richard Cross

Liaison:
Mark Tufenkjian

Introduction:

The Robosub project is targeted towards creating an Autonomous Underwater Vehicle (AUV) capable of completing complex tasks to compete in the yearly Robosub Competition sponsored by Robonation and the Office of Naval Research. This competition requires teams to implement similar technologies to the growing self-driving car industry such as integrating sensors to observe the environment around the AUV, using these sensors to identify objects, and implementing a sophisticated Artificial Intelligence capable of executing the various movements required to complete each task.

Projects this complicated always come with problems. As you may know we are currently unable to meet in person, as such the availability of sensors and hardware for code testing purposes is unavailable to us. Testing out our controls system, the software responsible for movement and stabilization like that of a drone, is an important feature that we cannot follow through with on built hardware. Similarly, the AI system of the AUV cannot be fully tested as it needs live data from sensors to trigger its various capabilities.

This year, with the competition held as an online only format we took the opportunity to lay the groundwork for making this project an ongoing project. We focused on documentation and onboarding materials in order to ensure the following team can get up to speed in a quick manner. We also dedicated time to creating workflows, centralizing data, and creating reusable code frameworks and compartmentalizing the submodules. The benefits of creating this framework are the ability to change pieces of code without it affecting the overall execution of each module.

Related Works and Technologies:

As a yearly competition, there are code repositories available from teams that choose to share their code. In general terms depend on running Linux and the Robotic Operating System (ROS) to organize and run the various scripts and programs required to run an AUV. ROS is a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot behavior across a wide variety of robotic platforms. It is the industry standard for projects of this magnitude. The focus was using this framework for its ability to create independent packages that can be used alongside each other to build complex systems, as well as its built in libraries for communications between independently running scripts on multiple platforms

Arduinos (small microcontrollers) are used to control various actuated systems, as well as controlling thrusters. ROS is used to integrate this independent microcontroller to receive signals from another independent system.

Sensors on the AUV have available libraries that we use to pull data. These libraries are general purpose and must be modified to filter and send the data up to the higher level systems. These were used due to their out of the box performance and ease of use.

System Architecture:

As this is a complicated piece of software the team organized into subteams all with dedicated software goal as follows:

Controls:

The control subsystem is in charge of controlling the eight thrusters of the AUV. It works in combination with the IMU sensor that provides us with the current location of the AUV along with Navigation's SMACH node that helps navigate the robotic submarine. The SMACH node informs the AUV what path to take and which direction to move in. This is carried out through an algorithm that combines the Proportional, Integral, and Derivative, or the PID, calculation to move the AUV from its current state to its desired state. The PID controller implemented in Arduino is responsible for performing tasks like roll, pitch, and yaw movements to successfully maneuver the AUV vertically and horizontally. Pitch tilts the AUV forwards and backwards, roll moves it to left and right, while yaw rotates the sub left and right. All of these movements can be performed by the AUV both simultaneously or individually on their own with the help of our PID algorithm along with a block of code containing mathematical calculations that combine the PID output values.

The thrusters on the AUV move according to the four final PWM signals that the code generates by combining the output values from the PID controllers. The PWM signals are the generated speed for our thrusters, telling them which way to move or spin. For movements like pitch and roll, the PWM signals generated by the PID algorithm are sent to four of the eight thrusters responsible for moving the AUV vertically, see *figure 1.1* below. Similarly, for movements like yaw, the AUV moves horizontally using the other four of the eight thrusters, see *figure 1.2*

below.

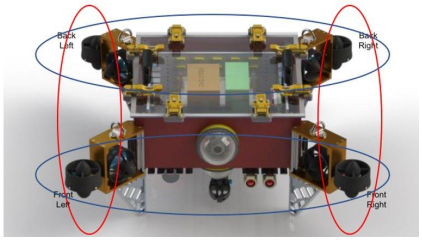


Figure 1.1: Red circles represent the thrusters that will move for roll, while blue circles represent pitch.

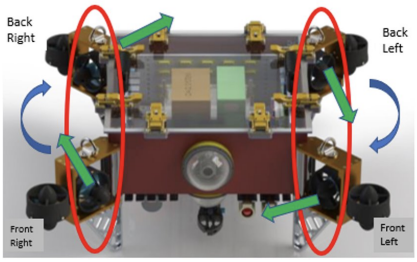
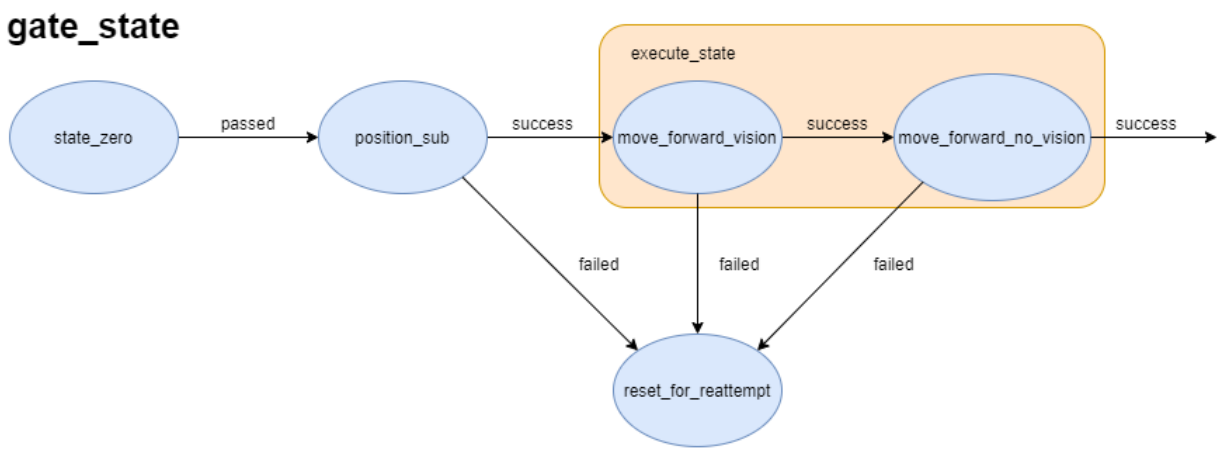


Figure 1.2: Red circles represent the thruster that will move for yaw; green and blue arrows show the direction of movement.

Mission Planning:

Mission Planning(MP) uses SMACH to create state machines that allow the AUV to perform tasks given to it. The state machines work alongside other subsystems such as Computer Vision(CV) to achieve the various tasks outlined by the competition. SMACH states are subroutines that are developed to perform a specialized task. This will usually lead to an execution state that triggers a transition to a new state.

To use the AUV as an example, the gate task starts with the task of finding and recognizing the gate with CV data. The AUV then transitions to the tasks of moving toward the gate and going past it with thruster instructions provided by Controls. Once the AUV has completed these tasks, it will transition to the next task given to it. These tasks also contain outcomes that transition the AUV to a state designed to reset the subs' positioning if something goes wrong. All of the states are given to the AUV in this order of events.



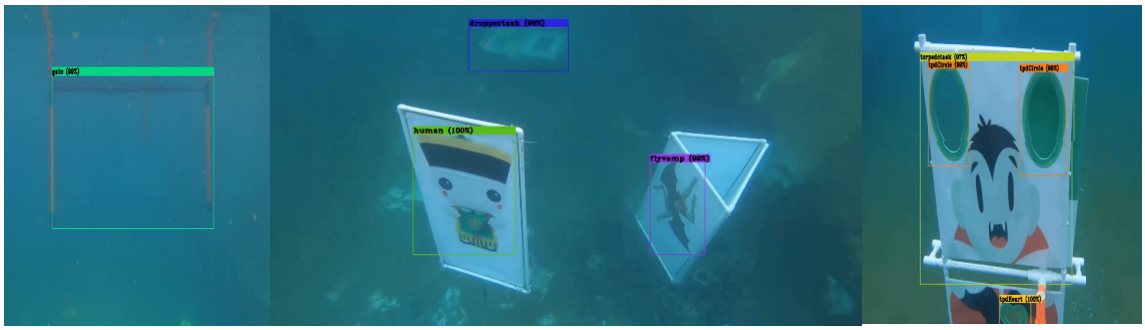
Navigation:

Our navigation system entails a few select devices: Ping Sonar, IMU, Hydrophone, DVL, and a Barometer. The Ping Sonar will be utilized to detect the distance of the largest object relative to the face of the sub. The IMU mode is primarily used for rotational movements such as yaw, pitch, and roll as well as translation movements. The hydrophone will record and detect where pingers in the water are located which will tell the sub where specific tasks are located, and the Barometer will read the depth of the sub in the water. The vehicle will move primarily with the use of the IMU until it locates a sonar-observable object and needs to adjust accordingly.

Computer Vision:

Robosub will need object detection software to be able to detect, classify, and localize objects underwater. To accomplish this, the team utilized Deep Learning algorithms, more specifically Convolutional Neural Networks (ConvNet), which is a type of deep neural network inspired by the human brain visual cortex, which is responsible for processing visual information. The algorithm that was used is called YOLOv4 (You Only Look Once) and can detect objects and their locations/sizes simultaneously. YOLO is one of the Convolutional Neural Network (CNN) approaches with the best speeds of object detection compared with other algorithms. It does this by considering object detection as a regression problem to assign class probabilities to spatially separated bounding boxes. Our network is trained using Darknet, an open source neural network framework, and given a dataset of images which have been labeled to distinguish the objects we want to detect. The algorithm was trained on over 4 thousand images and was trained to detect 12

unique objects. The training was done over a 50 hours period and had gone through 24 thousand iterations and had processed over a million images. After the training of our Convolutional Neural Network we then tested it on a video of the competition as you can see below the results are pretty good with majority of the objects being detected with a confidence of over 80%, with the expectation of one or two objects the network performs well and we now know what must be done to improve upon it for next time.



Results:

Controls:

Our goal was to implement an Arduino to create movement on AUV using the eight thrusters. We improved and developed previous horizontal movement control code to test on the AUV but due to the current circumstances we had limited to no access to the submarine. Therefore, we built a drone prototype as a substitute for the AUV, however, we were unable to advance because we encountered power supply issues with two of the four motors on the drone. Hence, we designed a graphical user interface, or a GUI, to visualize the PWM speeds for all thrusters. We also created diagrams and code flowcharts to visualize the movement process of the AUV, which can be found in the Robosub Github repository.

Navigation:

The goal this year is to finalize code for reading sensor data from the various onboard sensors. On the AUV the onboard sensors include:

- IMU (Inertial Measurement Unit): Used to detect the relative angles of the sub as well as basic translational movement
- Sonar: Detects if there is an object in front of the sensor within 20 ft
- DVL (Doppler Velocity Log):Tracks translational movement using an array of sonars
- Barometer: Detects pressure of fluid to determine depth

These were all built as independent packages with the intention of them being reused. As stand alone packages the process of integrating the package to a new software system for a future competition is greatly simplified.

Mission Planning:

As a result of the inability to accurately test state machines with hardware, more focus was placed in creating code skeletons along with descriptions and flowcharts that generally show what is expected in the performance of the AUV in a competition setting. Currently the state machines are designed only to transition between each other, and stop when the last state machine is reached. All materials and descriptions are posted in the Github repository for the AUV.

Computer Vision:

Our goal was to implement object detections within videos using dataset from past competitions. We were successful in producing a video that was able to detect objects using YOLO as our CNN Model. Our results were able to confidently create guidelines for future developers to follow. We tried implementing many online methods to be able to recognize objects on images. We were able to successfully recognize images but were not able to work with video implementations. For future implementations, our goal is to be able to test algorithms using live video feed from cameras implemented on the submarine. Another task for future implementations is to be able to communicate with other subsystems from the results we produce.

Conclusions:

The circumstances of this year made a project of this magnitude a difficult process. We constantly had to reevaluate how we would move forward week on week. We focused on the future of this project, with code snippets and designs to be completed by the following team. We believe that this foundation will allow this software to be completed and functioning for the 2022 Robosub Competition.