ELECOMP CAPSTONE
VIRTUAL SUMMIT
FRIDAY, MAY 7, 2021

THE UNIVERSITY OF RHODE ISLAND

THINK BIG WE DO℠

Applying ML Research for Undersea Object Detection

Team Members: James Morris (ELE), Anthony Neves (CPE), Andrew Gomes (ELE)

Technical Directors: Thomas Santos, Dana Brown | Consulting Technical Directors: Noah Daniels, Al Gaines (‘88), Najib Ishaq (‘20), and Jeremy Peacock (‘20)

Project Motivation

A difficulty of utilizing Unmanned Underwater Vehicles (UUV) is accounting for physical obstacles in the path of that UUV that may require additional navigational commands. Rite-Solutions is collaborating with Dr. Noah Daniels and his graduate research team to develop a machine learning model that can identify objects on the seabed. This technology could be applied across several commercial industries, as well as the Navy. The technological advantages of the machine learning model include avoiding potentially hazardous objects for the UUV, searching for and identifying objects, and eventually aiding in autonomous UUV operations.

The software package being designed to satisfy the above criteria requires translation of existing Python/Tensorflow code into a language with a higher focus on safety, Rust, to implement the software onto standalone devices properly. To test the software package prototype, a MATLAB program will be created to simulate an active side-scan sonar, where results can be validated under different conditions of the sea.

Key Accomplishments

- **Sonar Emission:** The MATLAB simulation of a side-scan sonar requires both an emission of energy that can be applied to an object, as well as sensors that receive feedback from energy bouncing off of an object. The emission of this energy from a sonar sensor has been simulated in a very simplistic form, which is the foundation of the High-Level Design Flowchart for our MATLAB program.

- **MATLAB Examples:** Several MATLAB resources are influential to the Sonar emission, detection, and object detection codes that we are creating. While creating a simulation of a side-scan sonar in MATLAB, we are also editing and applying the “Underwater Target Detection with an Active Sonar System” mathworks program to our own coding. By utilizing this example, the group can evaluate which methods of sensor and signal representation best fit our goals as we progress.

- **Objects as MATLAB Targets:** Replacing the targets of Sonar example codes with solid objects instead of designated targets strength objects is a key step in the development of the MATLAB code. The desired output is the feedback from the sonar simulation when energy is reflected off of the object, therefore targeting that object with the sonar emission is critical.

- **Rust Translation:**
  The existing Machine Learning model for Object Detection is written in Python and, to make the project more safety-oriented/operable on a standalone machine, the code must be translated into rust. There are many parts to the CLAM code created by Dr. Daniels and his research team, and parts of that code are completely translated into rust. The Graph portion of the code has been implemented, with key functions throughout the code translated into Rust as well. This was an entirely new language, and a large foundation has been created to move forward with the complete translation of the program into rust **(Fig 4)**.

- **Cylinder Target**
  The MATLAB program produced data from a side-scan sonar simulation, projecting energy onto a cylinder underwater. The data is in the form of Integrated Pulse Data, with the measurements being voltage, time, distance from Sonar source, and angle with the receiver **(Fig. 2)**. The data is simulated under ideal conditions, with a flat seabed and a speed of sound underwater that represents mild ocean conditions with no obstacles. This data is then intended to be tested with Machine Learning, to reproduce the 3D-object, only using the Integrated Pulse Data.

- **MATLAB Object Viewer:**
  The MATLAB simulation includes a script to view the object created for data simulation. The objects are created using target strength patterns, which rely on azimuth and elevation angles, and well as length, width, and a stacking method. A 2D-Object is repetitively stacked based on the target strength formula, and a 3D-object is created as a result. The MATLAB Object Viewer allows the user to view the slices that are created that make up the object as a whole **(Fig. 1)**. This provides insight into how the objects are created, as well as a good guideline for moving forward to create more objects to be tested. The object viewer also serves as a test for the object’s shape before testing with machine learning, to make sure the desired object is actually represented **(Fig. 3)**.

- **Target Strength Sphere:** Attempts to use several different MATLAB functions were used to produce different target strength objects, including a target strength sphere. Similar methods were used as were done with the target strength cylinder, including tracking the output data of the target object and initiating different inputs for the MATLAB script to run. At the time of this report, the methods of the cylinder have not worked for the sphere. The misuse of the necessary functions and parameters can be a leading possibility to why such issues arise.

Anticipated Best Outcome

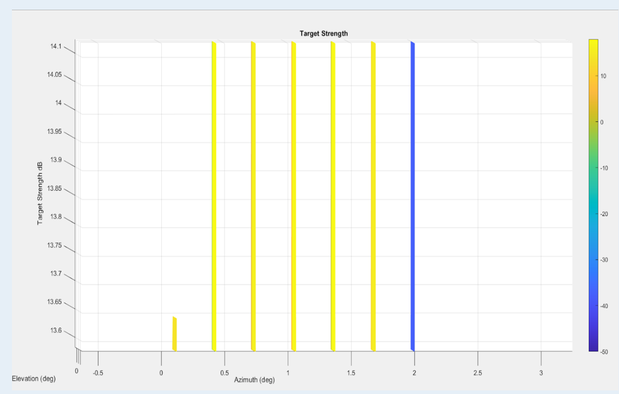
The primary objective of the project is to design a software package that can quickly and accurately detect objects underwater. In order to accurately test this software package, the simulated sensor data from MATLAB must be created and verified to be working properly as well. As a result of the translation of the existing Python code into Rust, the effectiveness of the new programming language Rust can be evaluated as well for future products with similar purposes made by Rite-Solutions. The best possible outcome would be realized with an effective Rust and MATLAB software package that can simulate side-scan sonar data and detect underwater objects from that data accurately.

Project Outcome

The Anticipated Beat Outcome was not achieved, whereas of right now, the final deliverable product is a compilation of Rust and MATLAB software to be used moving forward with Undersea Object Detection. The results from the MATLAB code from the Sonar Simulation are for the cylindrical targets with different sizes, angles, and locations. While this code is being sent to Dr. Daniels’ team, the main body of the Graph implementation is thoroughly documented. This documentation is to ensure clarity and understanding in both how to properly utilize its individual components as well as the overall architecture of the program.

Figures

Fig. 1: The MATLAB Object Viewer is seen here for a Box Target. The individual slices are seen as rectangles, the x-axis representing the azimuth angle of the target, and the z-axis representing the Target Strength in decibels, the color bar on the right serves as a visual aid for seeing the target strength.



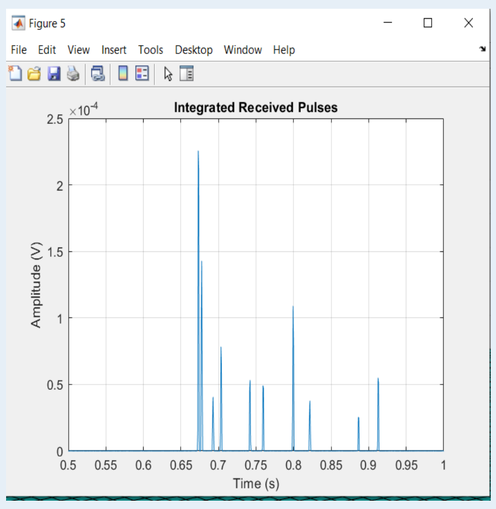
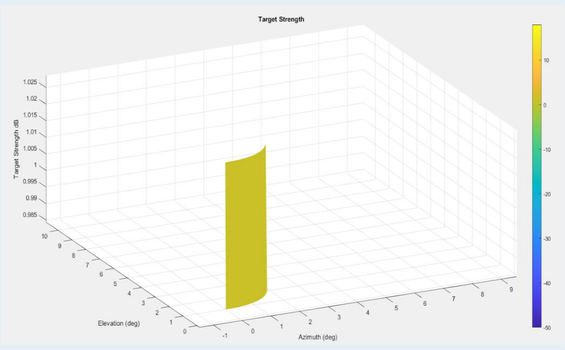


Fig. 2: The Integrated Pulse Data of a cylindrical target is seen here, with a total of 100,00 samples. There is a data point taken for every meter that the sound wave from the Sonar moves, about 1,520 samples per second. The spikes in data represent energy bouncing off of a higher target strength, with an amplitude of 0 representing no returned signal.

Fig. 3: The stacking of slices can be seen here with the partial Cylinder recreated from the cylinder target. When viewed from the angle of the source, the target appears cylindrical, although the figure is rotated here to show all axes and that the target generated does not need to be completely solid, it must only appear solid to the sonar.



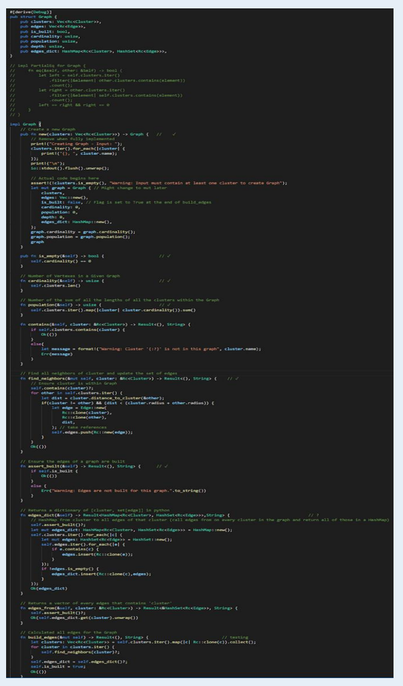


Fig. 4: The overall coded implementation of the Graph program. This program takes in a mass of clusters (clustered data points) and creates edges between them. This essentially makes a wireframe mash for the future CHAODA implementation to take as input to differentiate different objects.

ELECOMP Capstone Program Director: Professor Harish R D Sunak

email: sunak@ele.uri.edu telephone: 401-874-5859 ELECOMP Website: uri.edu/elecomp-capstone