

TEC-V

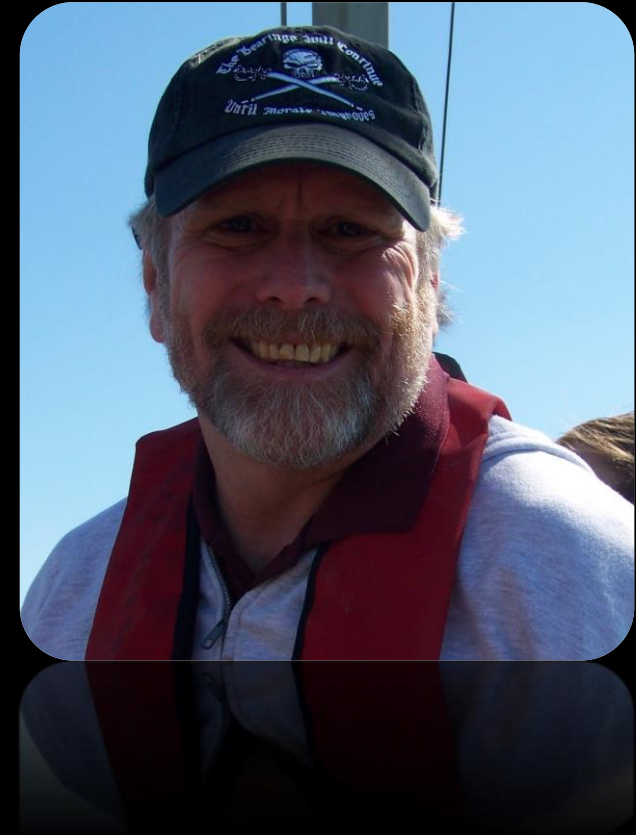
MILESTONE 2

By: Michael Dowling & Zealand Brennan



CLIENT

- DR. Wood
 - **Professor** | Ocean Engineering and Marine Sciences
 - **Program Chair for Ocean Engineering**



MILESTONE 1 OVERVIEW

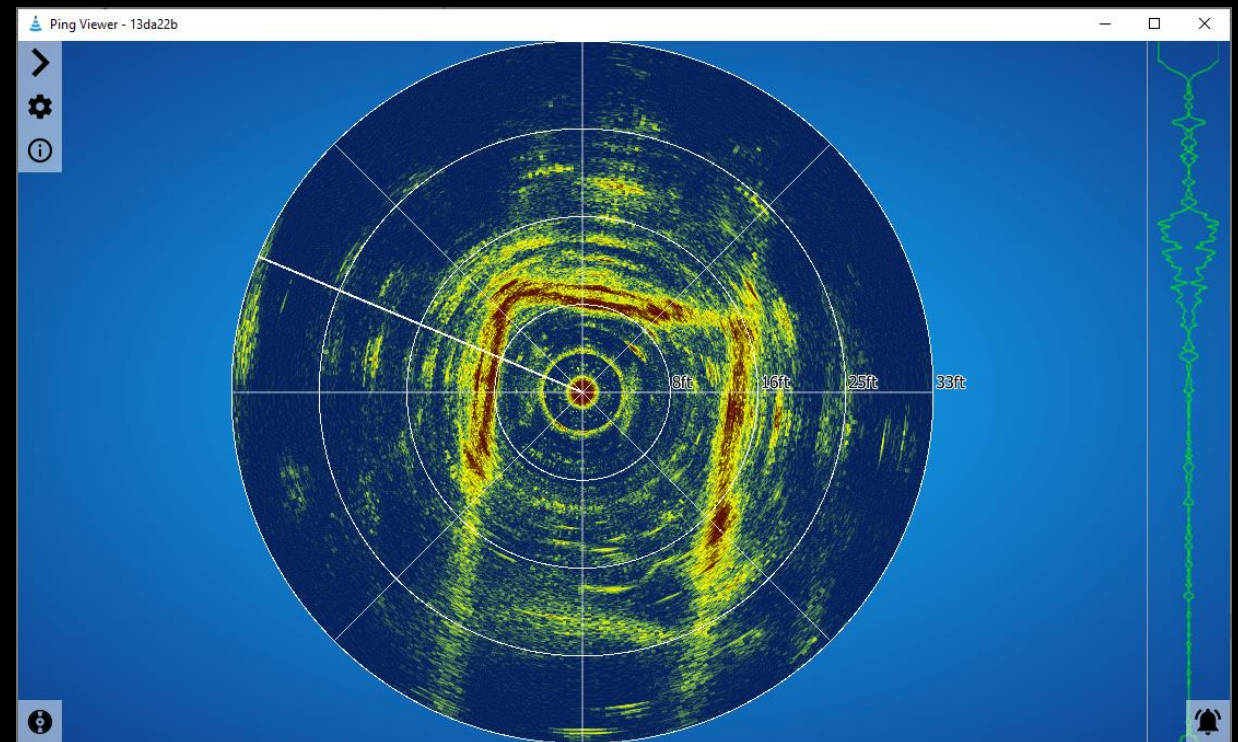
- ❖ Sonar Data Retrieval
- ❖ Information Saving
- ❖ Testing
- ❖ Data Interpretation
- ❖ Point Cloud Plotting



SONAR DEVICE

Ping 360

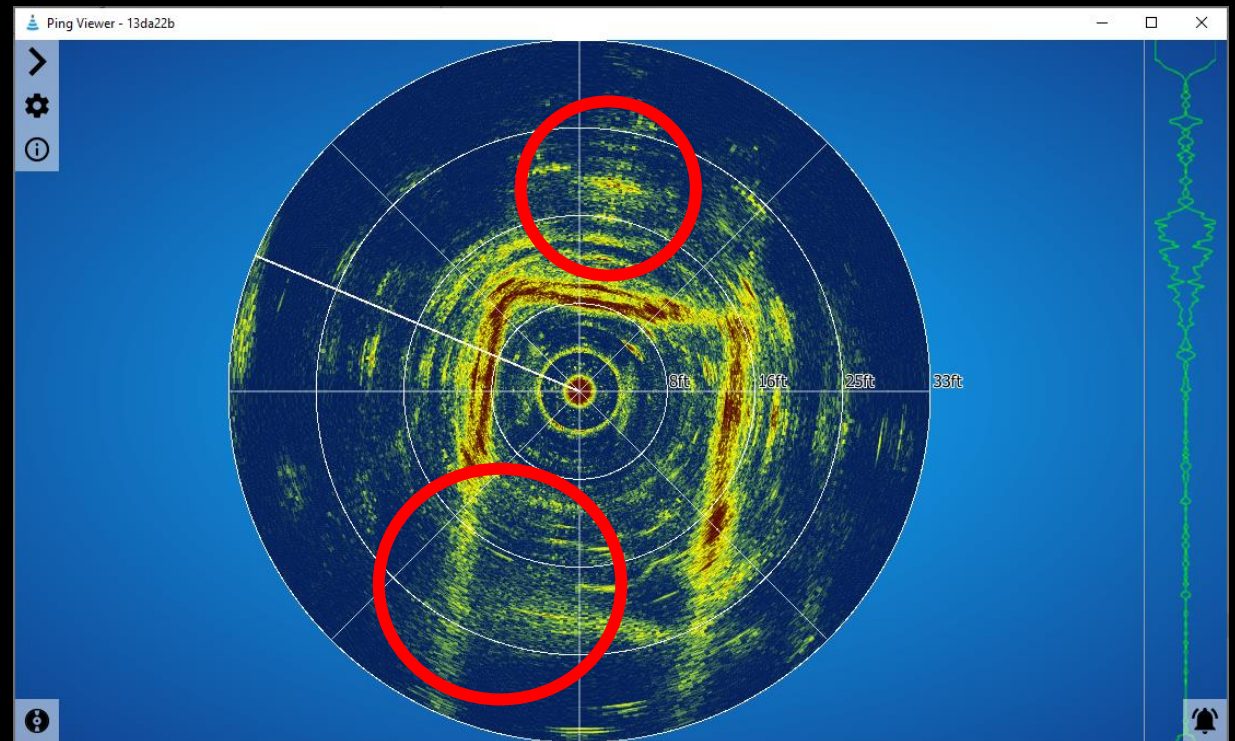
- Data is read by degrees and intensity values.
- Main view of what data you may come across



DETAILS

Main Problems

- Data can be missed by the sonar
- Reflections can cause higher intensity values than actual readings
- .8 meters from center is not viable



DATA RETRIEVED

Message

- Loop that asks for the intensity values at x degree to be sent back
- Intensity values 0-255
- Range 1200 in array

EXAMPLE: Intensity array

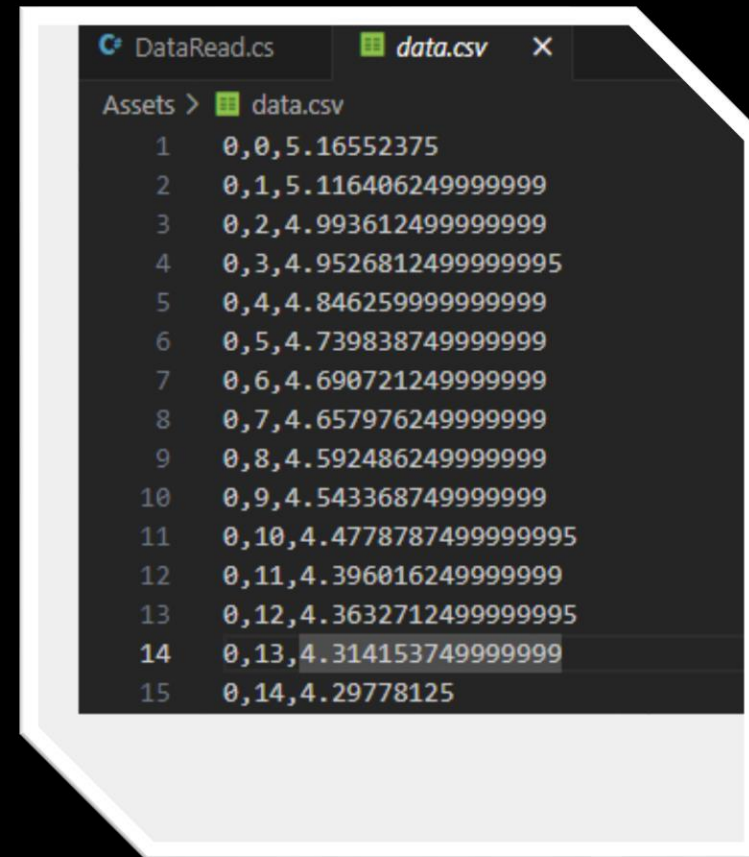
[0, 4, 134, 55, 20, 100, 160, 255, 240.....]

```
78
79 for currentAngle in range(400):
80     # Read a single iteration of intensity data
81     ping_data = ping360.transmitAngle(currentAngle)
82
83     # Extracting intensity as integer values
84     intensity_data = [(struct.unpack('!H', int(data).to_bytes(2, byteorder='big'))[0], i)
85                       | for i, data in enumerate(ping_data.msg_data)]
86
87     # Store the raw data for the current angle
88     raw_data[currentAngle] = intensity_data
```


DATA SAVING

Data.csv

- Three categories
 - Depth (in progress)
 - Angle
 - Most likely distance to object



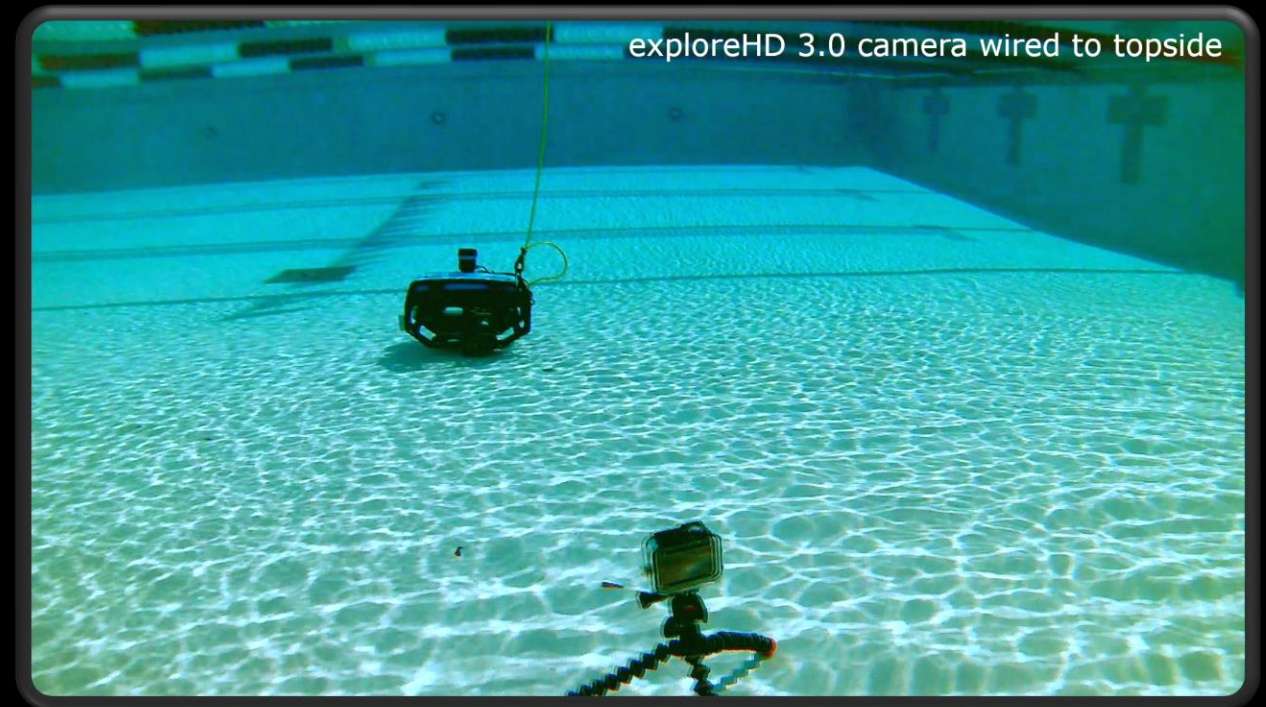
The screenshot shows a code editor with two tabs: 'DataRead.cs' and 'data.csv'. The 'data.csv' tab is active, displaying a list of 15 rows of data. Each row contains three comma-separated values: a number (1-15), a three-part coordinate (e.g., 0,0,5.16552375), and a long decimal number (e.g., 0.16552375). The 14th row is highlighted.

Line	Coordinate	Value
1	0,0,5.16552375	0.16552375
2	0,1,5.116406249999999	0.116406249999999
3	0,2,4.993612499999999	0.093612499999999
4	0,3,4.9526812499999995	0.0526812499999995
5	0,4,4.846259999999999	0.046259999999999
6	0,5,4.739838749999999	0.039838749999999
7	0,6,4.690721249999999	0.060721249999999
8	0,7,4.657976249999999	0.057976249999999
9	0,8,4.592486249999999	0.042486249999999
10	0,9,4.543368749999999	0.043368749999999
11	0,10,4.4778787499999995	0.0778787499999995
12	0,11,4.396016249999999	0.096016249999999
13	0,12,4.3632712499999995	0.0632712499999995
14	0,13,4.314153749999999	0.014153749999999
15	0,14,4.29778125	0.0778125

TESTING

10-21-23

- Clemente Pool 10 a.m. to 1 p.m.
- Goal:
 - Test sonar data retrieval
 - Collect Data for Cloud Plotting
 - Have a real-world test to see accuracy



POOL TEST

Transcription

- Idea 1: Using Java
- Original Formula was incorrect

```
// Normalize the angle
double angleDegreesNormalized = angleDegrees % 360;
double angleRadians = Math.toRadians(angleDegreesNormalized);

double x = distance * Math.cos(angleRadians);
double y = distance * Math.sin(angleRadians);
```

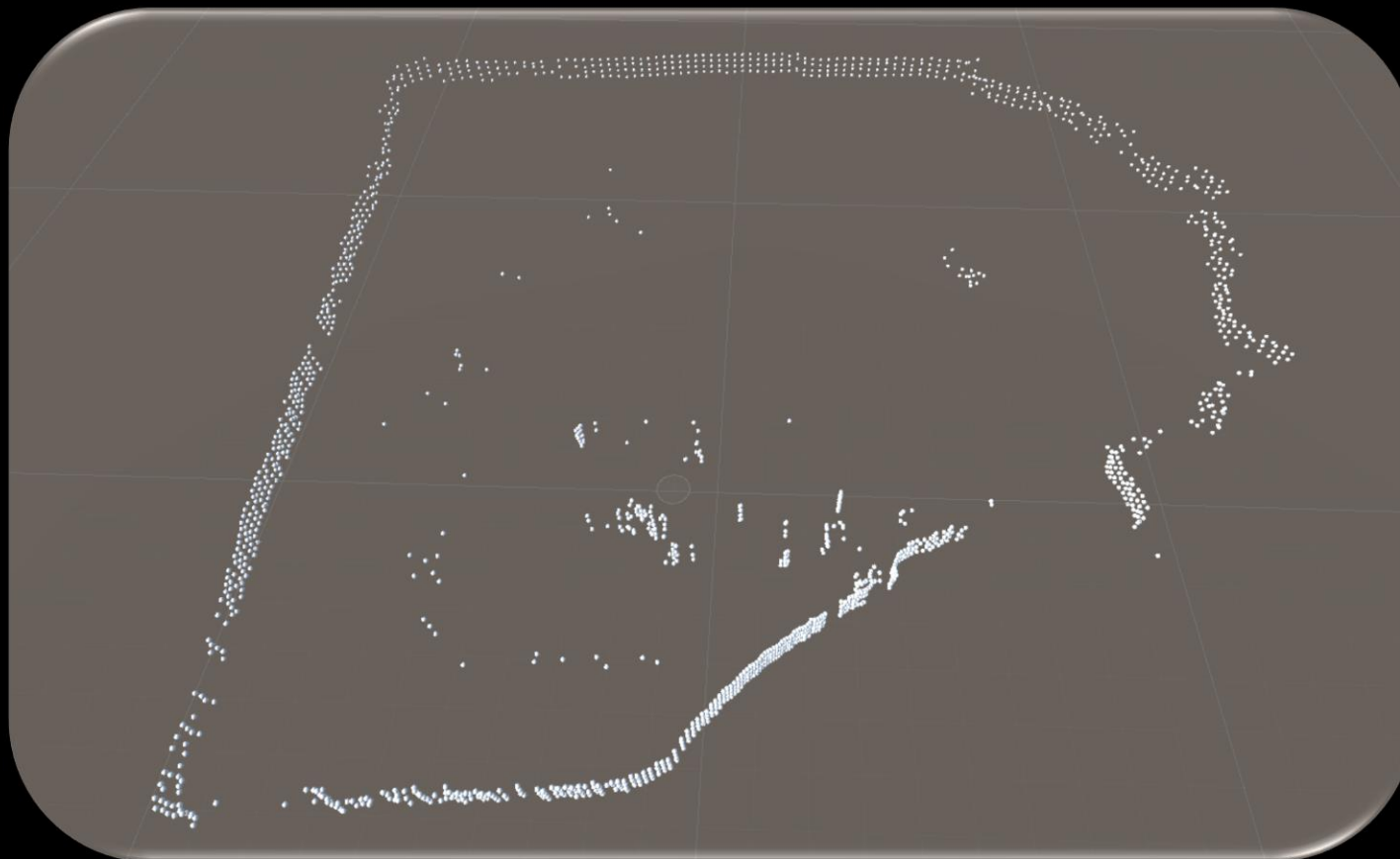
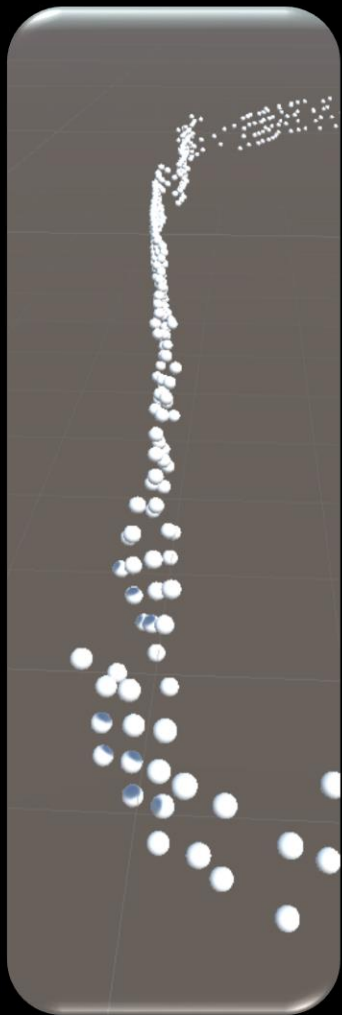


UNITY

- Idea 2: Unity
 - Secondary formula corrected
 - Better data manipulation

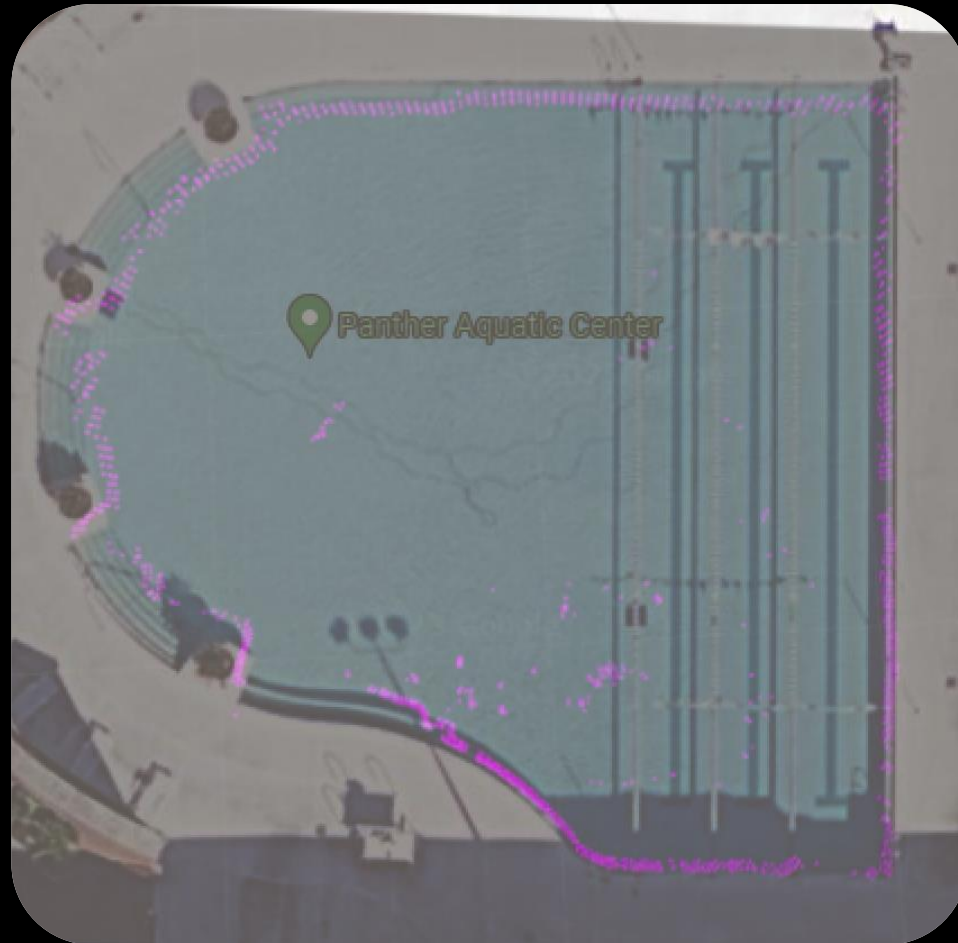
```
while (!endoffile)
{
    string Data_String = stReader.ReadLine();
    if (Data_String == null)
    {
        endoffile = true;
        break;
    }
    var datavalues= Data_String.Split(',');
    Debug.Log(datavalues[0].ToString() + ',' + datavalues[1].ToString() + ',' + datavalues[2].ToString());
    Instantiate(sphere, new Vector3(Mathf.Cos(float.Parse(datavalues[1]))*
        (Mathf.PI/200))*scale*-1* float.Parse(datavalues[2]),
        float.Parse(datavalues[0])*scale/5, Mathf.Sin(float.Parse(datavalues[1]))
        * (Mathf.PI / 200)) * scale *float.Parse(datavalues[2])),
        new Quaternion(1, 1, 1,1));
}
```

UNITY



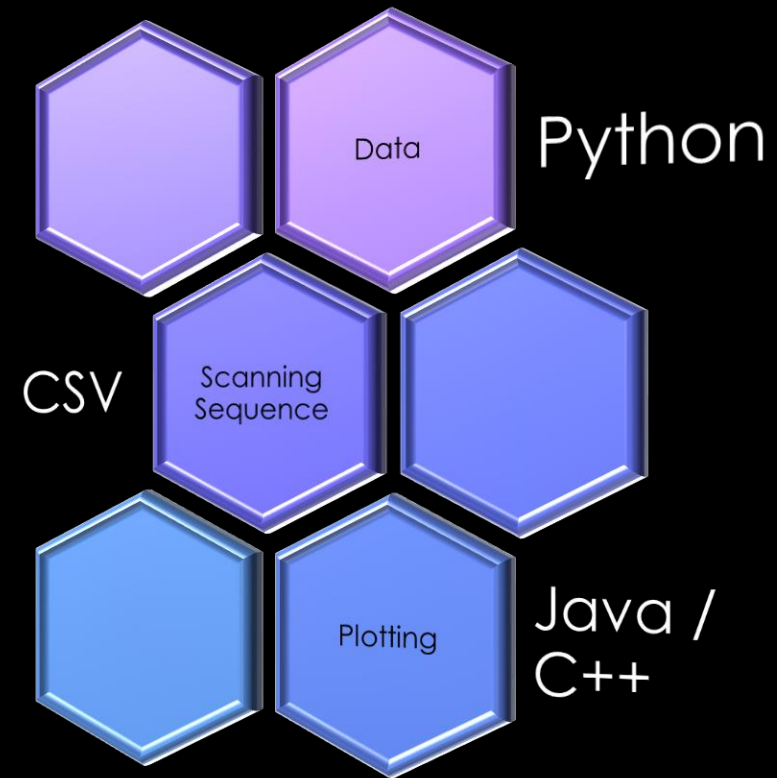
TRANSPOSE

- Data shows accuracy along flat edges
- Slight difficulty along the shallow end
- Shadows from where the sonar was unable to see



TOOLS

- Data: Python
 - Git Hub package that allows for simple commands
- Plotting: Unity / C++
 - Allows for better data manipulation in 3D environment



MILESTONE 3:

Task	Michael	Zealand
False Data	Create an algorithm to remove false data points or fill in the shadows within the data.	
Depth Finder	Identify the protocols to find and retrieve this data, may need to be done through Arduino.	
Compass and Telemetry	Identify the protocols to find, retrieve, and save the information.	
Cloud Plot Application		Work on creating an environment that will transpose the data and allow for Autonomous testing

MILESTONE 3: TASKS

1

☐ Improve False Data:

- ☐ Create a sorting algorithm to remove false data

2

☐ Telemetry Data:

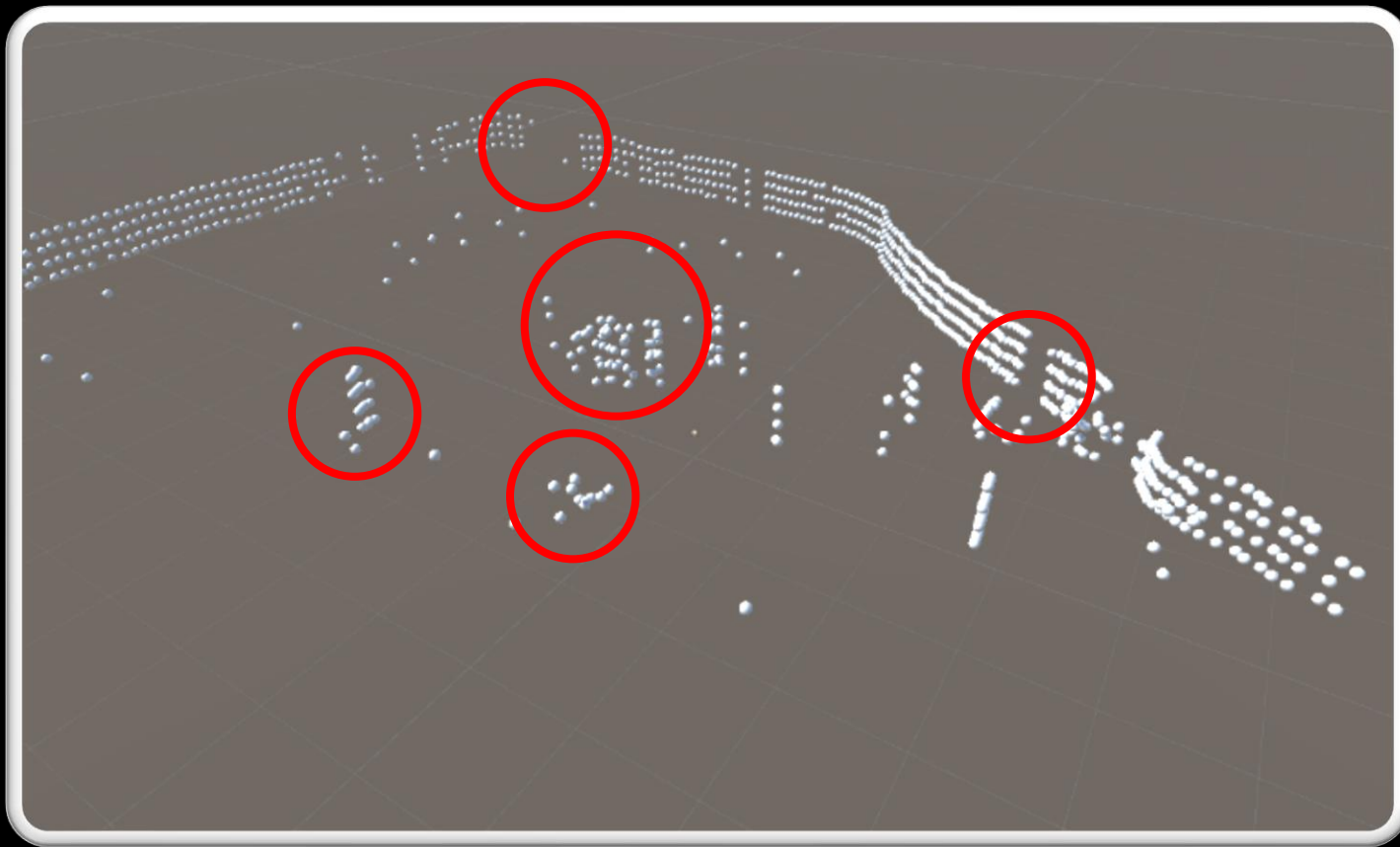
- ☐ Gain access to accurate depth and positioning instruments

3

☐ Cloud Plotting / Testing

- ☐ Use Gazebo to plot and test AI pathing

FALSE DATA





Live Demo

WEBPAGE LINK

TEC-V

https://bluecodehydra.github.io/FIT_Project-TEC_V/data.html

QUESTIONS?

