*i*Journals: International Journal of Software & Hardware Research in Engineering (IJSHRE)
ISSN-2347-4890
Volume 10 Issue 1 January 2022

Bathymetric Scanner for Underwater Terrain Survey using Embedded System and TF Mini LiDAR Sensor

Aadi Verma[1]
Student at Jumeirah English Speaking School, Dubai
Completed college level engineering course at Johns Hopkins University
Email: aadismart@hotmail.com

ShekharJain[2]
Chief-Mentor and Co-Founder of On My Own Technology Private Limited Mumbai,
Email:reetu.jain@onmyowntechnology.com

Abstract: ***Deep-sea terrains are hard to map, some because they are remote others because of danger and some, are just too hard. The current way of mapping sea terrains is done by a multi-beam echo-sounder which is attached to a submarine. This is an inefficient technique for various reasons, to start with it is just too expensive, risky and time-consuming. Deep-sea terrain mapping is important because it helps us understand the Earth's environment and it can better help to understand how to respond to natural disasters. The multibeam works by emitting several single narrow beams. The transducer is mounted in the keel of the 'RV Simon Stevin'(A boat used to perform coastal oceanographic research) and produces a range of sound waves. With this technology, the bed of the sea is examined with a line of continuous beams that are perpendicular to the travelling vessel. To do this the device has to be in the water on the keel of the boat making it an expensive endeavor. To combat this issue we have developed a remote lidar scanner that can be used to map out deep sea terrains using a TF mini LiDAR. The scanner builds out a 3D printable PLA Material device that works as an x-y scanner and gives terrain surface data by scanning the terrain with coordinates and graphing out those coordinates.***

Our device can easily be deployed at a much cheaper and quicker rate. Our product will be able to revolutionize this space and increase the advancement in this field at an exponential rate. The device that we have developed can be remotely controlled meaning that there is minimal risk and maximum efficiency. The device can be mounted on any form of transportation and builds images from above the water or terrain. Through building a 3d model, the user would be able to rotate the model and virtually explore it giving users a better understanding of the terrain they are exploring and also minimizing the risk.

Keywords: Lidar, Bathymetric, Remote terrain, Laser scanning, Deep sea, Assistance, Scanner.

A. Introduction:

Through our research, we found out the importance of the exploration of deep-sea terrain. Another thing that was figured out during the research is that the current solutions are not a viable option due to the capital and time required to use them. Therefore, we decided to start exploring a solution for this issue. We were able to figure out that the most effective way to solve this problem is by using lidar technology.

By using bathymetric LiDAR technology, we can save time and money while effectively building a 3d model of the terrain. The device can be mounted on a drone above the water, and we will be able to build a 3d model of the underwater terrain using the data collected by the device. This will also increase the number of terrains that will be mapped because the cheaper it is to map the terrain the more people will be doing it. Since more people will be mapping underwater terrains,it will be easier for scientists to study the terrains. After doing some research we were able to find different solutions to the problem like the multibeam transducer which once again

was too expensive for it to be used effectively because it must be mounted on the keel of a boat and constantly be moved around. We can overcome this issue by having a range of 10m and being able to detect terrain through substances; our product can also be remotely controlled and piloted meaning that it is extremely safe for the user. As seen in fig A there is an underwater scanner that uses lidar but it has to be on a boat and in such a big case[1].

[FIG:A] Mapping underwater terrain with bathymetric LiDAR Technology

We aim to overcome this and make the most effective scanner with the best practices possible which includes minimizing size, cost and risk. Since it is hard to steer a ship around to map a terrain, especially in Dubai we have decided to make a device that would be as portable as possible so we can optimize its effect and maximize its outcome. By making it a portable device we can maximize its usability and give it limitless applications.

B. Literature Survey:

[1]One of the current solutions available is a Bathymetric Lidar scanner but this scanner unlike ours requires a huge team to operate and is extremely large and expensive. This scanner was made with the purpose of shallow water scanning which means it might not be as effective as necessary. The company that is currently developing this system is called Leica Geosystems and the product is priced extremely high and requires experts to operate the product with a geographical and electrical background.

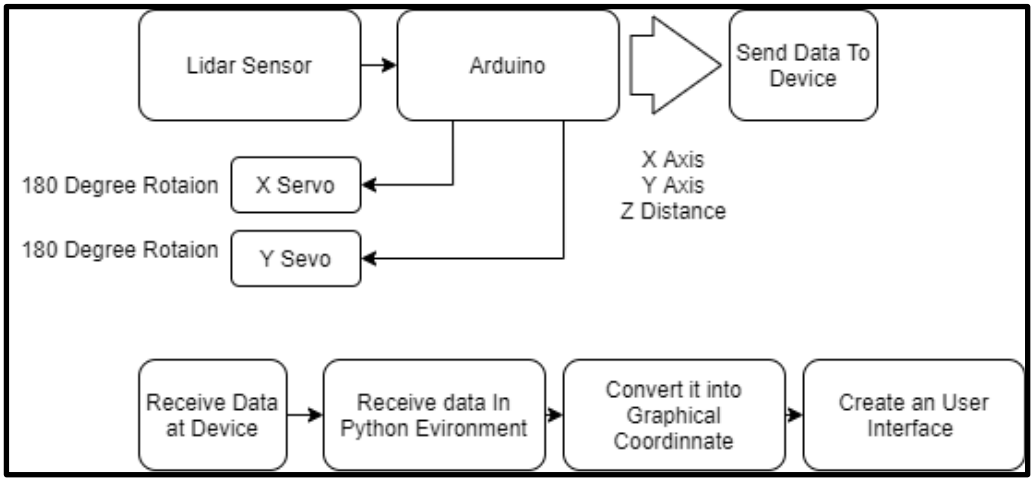
[2]Another current solution to this problem is a High-Resolution Underwater Mapping Using Side-Scan Sonar This solution uses an echo to create its images but this too has to be submerged in water and would still take time to deploy because it is still an idea and not yet a working solution.

[3]Teledyne Blueview's 3D Multibeam Scanning Sonar creates high resolution, laser-like, imagery of underwater areas, structures and objects of interest. This product is more compact than the other solutions discussed but is still an expensive solution with a lot of additional costs and does require you to know how to use the system and operate the sonar scanner.

C. Working Principle:

This project is based on the fact that deep-sea underwater research is difficult and currently is not accessible and affordable. So to combat this issue I have created a portable, mobile terrain building device. This device can form underwater 3d models of terrain. This is done by a lidar sensor attached to the device which can rotate 360degrees with the help of 2 servos which it is attached to. The lidar sensor then sends all the coordinates of each point that it scanned to the laptop through a receiver which builds a 3d model of the terrain with the help of a python program. This process starts with the Lidar sensor sending each point on a terrain the Arduino which then sends the data to a device, the lidar sensor can rotate by using 2 different servo motors that rotate the x and y-axis. The data is then received by a device that sends it to a python environment, the python program then converts this data into a graphical coordinate by graphing each point that the lidar has scanned and this then

builds a 3d model of the environment. The data is being scanned in the x,y and z-axis with the z-axis being the distance and the x and y being the horizontal and vertical position respectfully.



[FIG: B] System overview diagram

The data is sent from a transmitter to a receiver connected to the laptop, the address that these are communicating on is 4*01. Once the data is received it is decoded and then shaped. This happens by plotting the data that is then run on the script which builds a 3d model using that data using the x,y and z-axis. 3 programs are written for the lidar to send the data to the Arduino which is connected to receive that as well as that there is a code for the 3d model to be built based on the code. The code is a 3d processing code that receives the data and uses a 3d graph to plot every single point to build a full image.

D. Technology:

The project has 2 major parts:

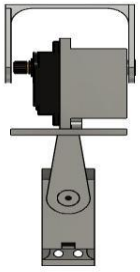
1. 3D Lidar scanner device and
2. Ground station unit.

Both the modules are connected wirelessly using the NRF24l01 Module which provides a range of 100m line of sight works at low frequencies.

1. 3D LIDAR Scanner Device

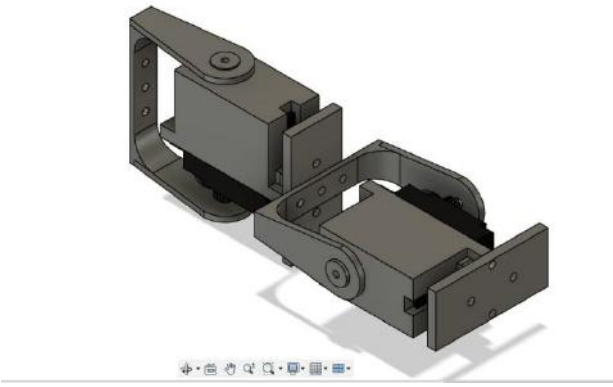
1.1 CAD Design.

This model was designed and simulated using Fusion 360 Autodesk's platform and went through various designs and modifications. The model was originally built for MG556 servo motors but then because of usability the smaller servo 9G was better and it was then adapted for the smaller servo motors.



[IMG: A] Front View

The 3d models connect the 2 different servo motors which work on the 180-degree angle to allow the lidar sensor 360-degree rotation which is key in allowing the lidar sensor to build a 3d model. The model is designed in such a way that it can balance the weight and centre of gravity of the servos(14 grams) and lidar(10 grams). This allows the model when built to not fall apart.

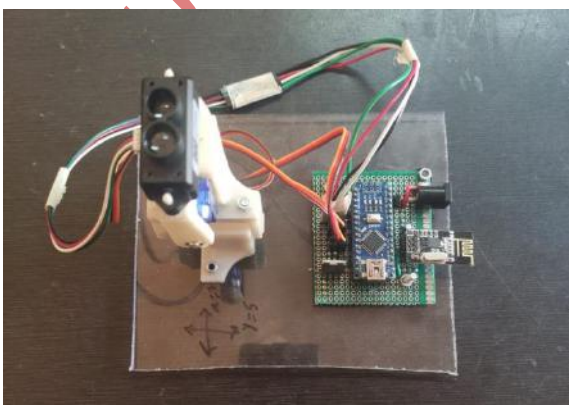


[IMG: B] Isometric view

The model is printed using ABS plastic material using an ender 3 v2 type of 3D printer which gives us good accuracy and precision with the parts.

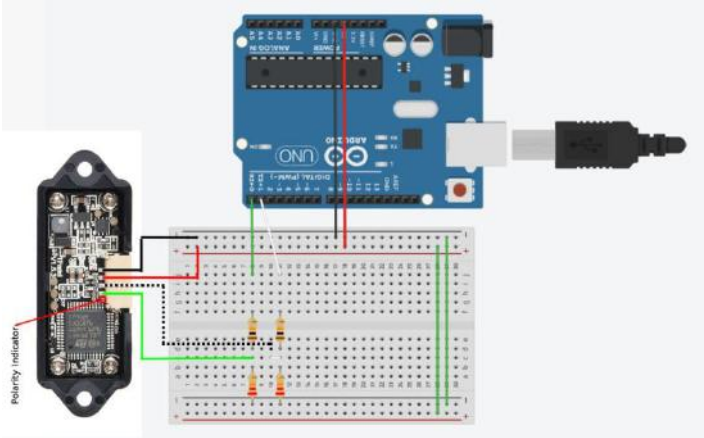
1.2 Electronic Circuit.

The circuit went through various design changes but we ended up using an Arduino nano so we can minimize the weight of the board and since we don't require a lot of components this was a perfect fit. We were also able to connect transmitters and receivers to the circuit so we can remotely control the device and receive data.



[IMG:C] Complete Ground Station

Electronics of the ground station built using Arduino nano and nrf module to send data to the ground module and receive instruction to perform specific operations. We have interfaced nrf modules using MISO, MOSI, SS, CLK, CE, and E type of interface which utilises 13,12,11,10,9 and 8 pins of Arduino respectively. The communication uses a handshake protocol to transfer data from one device to another.



[FIG: C] Interfacing of TF mini Lidar Sensor

As shown in fig C the interfacing of the lidar sensor uses low power TTL (i.e Transistor-Transistor Logic) which requires 3.3v to drive the logic levels so we have used two resistors-bridge to achieve the power level in the circuit. With the help of such configuration, the circuit runs on a low power supply and makes it easy to use on remote vehicles.

1.3 3D Lidar Scanner Device Programming.

As shown in fig C the interfacing of the lidar sensor uses low power TTL (i.e Transistor-Transistor Logic) which requires 3.3v to drive the logic levels so we have used two resistors-bridge to achieve the power level in the circuit. With the help of such a configuration, the circuit runs on a low power supply and makes it easy to use on remote vehicles.


```
void getDist(){
  delay(50);
  while(tfSerial.available() >= 9)
  {
  if(tfSerial.read() == HEADER)
    {
      if(tfSerial.read() == HEADER)
      {
        a = tfSerial.read();
        b = tfSerial.read();
        c = tfSerial.read();
        d = tfSerial.read();
        e = tfSerial.read();
        f = tfSerial.read();
    check = (a + b + c + d + e + f + HEADER + HEADER);
    if(tfSerial.read() == (check & 0xff)){
      dist = (a + (b * 256));
      strength = (c + (d * 256));
        }
      }
    }
  }
}
```

[IMG:D] Code for receive bathymetric distance data from Lidar Module

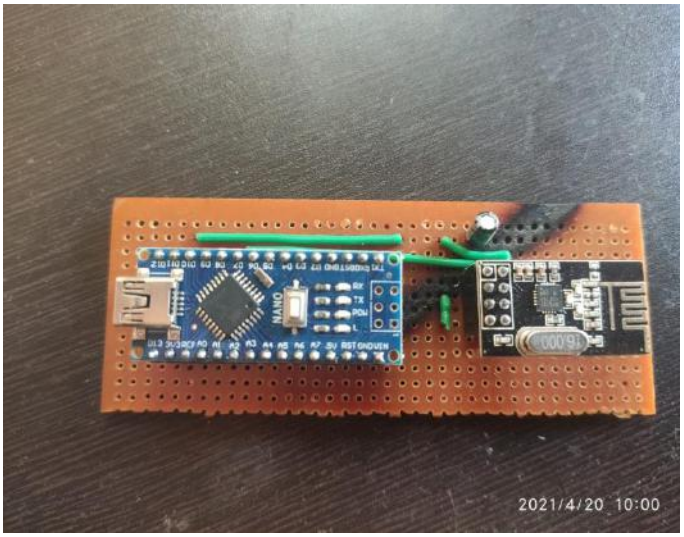
```
#include <SoftwareSerial.h>
SoftwareSerial tfSerial(2, 3);//RX, TX
#include <Servo.h>
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>

int myData[6]; //array


const int HEADER  =  0x59;
int TF01_pix;
int dist = 2200;
int strength,lastDist;
int a, b, c, d, e, f, check, i;

Servo servoRotate;
Servo servoScan;
int servoRotatePosition = 0;
int servoScanPosition = 0;
int servoRotateDirection = 1;
int servoScanDirection = 1;
int sensorEnablePin = 4;
int distance;

RF24 radio(9, 10); // CE, CSN

const byte address[6] = "00001";

void setup()
{
  tfSerial.begin(115200);
  Serial.begin(115200);
  radio.begin();
  radio.setPALevel(RF24_PA_MIN);
  servoRotate.attach(5);  //Servo making the pan movement
  servoScan.attach(7);    //Servo making the tilt movement
//  Serial.begin(9600);
  servoRotate.write(170);
  servoScan.write(180);
  delay(1000);

  //Keep pooling the serial buffer until any command is received f
  radio.openReadingPipe(0, address);
  radio.startListening();
  while (!radio.available()) {}
  //Moving to initial position
  servoRotate.write(servoRotatePosition);
  servoScan.write(servoScanPosition);
  delay(1000);
}

void loop()
{
  //Limit to only one scan, for more the program (or the Arduino)
  if (servoScanPosition <= 180)
  {
    servoRotate.write(servoRotatePosition);
    servoScan.write(servoScanPosition);
    getDist();
    if (dist > 2200){
      distance = 0;
    }
    else{
    distance = dist;
    }
//    distance = random(0,180);  // Shift high byte [6] 8 to the l
    // Print Distance
    printOutput(servoRotatePosition, servoScanPosition, distance);
    // Pooling the sensor from inside nested loops was giving bugs
    // The following code will make both servos move using only th
    //Making the pan servo move both ways
    if (servoRotateDirection == 1)
      servoRotatePosition++;
    else
      servoRotatePosition--;
    //Changing the direction in the edges
    //Moving the tilt servo only when the pan reaches each side
    if (servoRotatePosition == 180 || servoRotatePosition == 0)
    {
      servoRotateDirection = -servoRotateDirection;
      servoScanPosition = (servoScanPosition + 1)%182;
```

[IMG:E] Main Code for scanner device

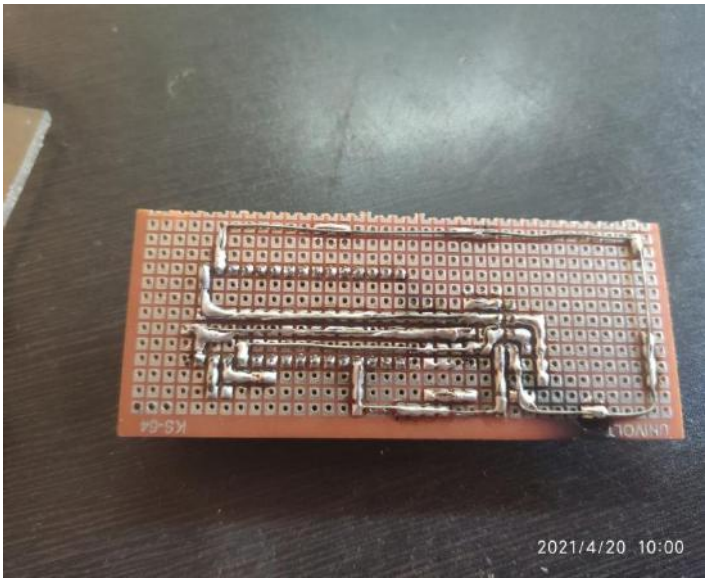
2. Ground Station Device

2.1 Electronic Circuit.

The ground station device is configured in a simple way it has mainly 2 components interfaced i.e Arduino nano and the nrf24l01 module, the nrf module receives data from the scanner device and it gives to the laptop using USB 2.1 interface also known as UART universal asynchronous receiver and transmitter.



[IMG D] Ground Module



[IMG E] Soldering/Bottom of Ground Module

Above images D and E Shows the configuration and setup of the ground module. The test pilot design is made up using a zero size Veroboard.

2.2 Arduino Programming.

The Arduino is programmed to receive data as coordinates from the device and this happens live as the device is scanning. Once the computer starts receiving data it then sends this to a python program which graphs all the coordinates to build a 3D model.

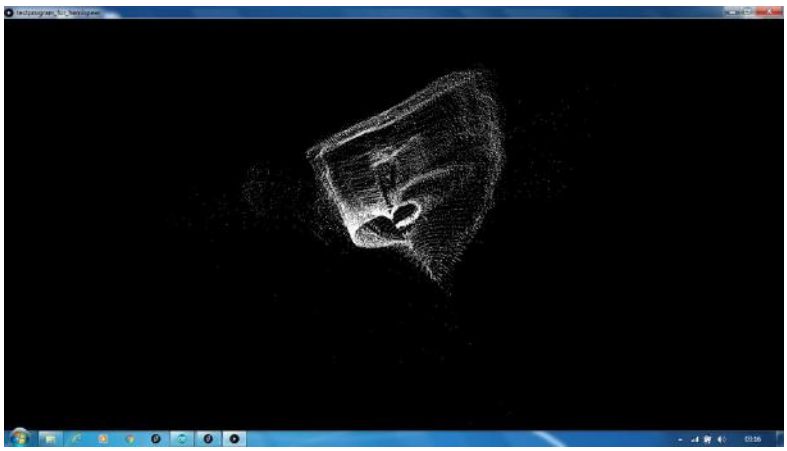
```
// transreceiver
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>

RF24 radio(9, 10); // CE, CSN
const byte address[6] = "00001";

void setup() {
    Serial.begin(115200);
    radio.begin();
    radio.setPALevel(RF24_PA_MIN);
//  radio.openReadingPipe(0, address);
//  radio.startListening();
}
void loop()
{
  if (Serial.available() > 0){
  char data = Serial.read();
  radio.openWritingPipe(address);
  radio.stopListening();
  char text = data;
  radio.write(&text, sizeof(text));
  Serial.print("Send data= ");
  Serial.println(text);
  }
  radio.openReadingPipe(0, address);
  radio.startListening();
  if (radio.available() > 0){
      int text[6];
      radio.read(&text, sizeof(text));
      if (text[0] < 10)
        Serial.print("00");
      else if (text[0] < 100)
        Serial.print("0");
      Serial.print(text[0]);
      Serial.print(",");
  //  Serial.print(" P ");
      if (text[2] < 10)
        Serial.print("00");
      else if (text[2] < 100)
        Serial.print("0");
      Serial.print(text[2]);
      Serial.print(",");
  //  Serial.print(" V ");
      if (text[4] < 10)
        Serial.print("000");
      else if (text[4] < 100)
        Serial.print("00");
      else if (text[4] < 1000)
        Serial.print("0");
      Serial.println(text[4]);
```

[IMG:F] Main Code for ground module

```
import peasy.*;
import processing.serial.*;

Serial serial;
int serialPortNumber = 0;
PeasyCam pCamera;

float radius = 2200.0;
float rho = radius;
float factor = 0.2;
float x, y, z;

ArrayList<PVector> vectors;
PVector[] sphereVertexPoints;

void setup() {
  size(1080, 720, P3D);
  pCamera = new PeasyCam(this, 8000);
  vectors = new ArrayList<PVector>();
  String[] serialPorts = Serial.list();
  String serialPort = serialPorts[serialPortNumber];
  println("Using serial port \"" + serialPort + "\"");
  println("To use a different serial port, change serialPortNumber:");
  printArray(serialPorts);
  serial = new Serial(this, serialPort, 115200);
}


void draw() {
  background(0);
  noFill();
  stroke(255);
  beginShape(QUAD_STRIP);
        String input = serial.readStringUntil(10);
        if (input != null)
          {
            String[] components = split(input, ',');
            if (components.length >= 3)
                  {
                    vectors.add(new PVector(float(components[0]), float(components[1]), float(components[2])));
                    println("X=",components[0]);
                    println("Y=",components[1]);
                    println("Z=",components[2]);
                  }
          }
        if (keyPressed && (key == 's')){
            serial.write('s');
        }
        int size = vectors.size();
        for (int index = 0; index < size; index++) {
        PVector v = vectors.get(index);

        float theta = map(v.y, 0, 180, 0, PI) ;
        float phi = map(v.x, 0, 180, 0, PI);
        float rho = map(v.z, 0, 2200, 0, 2200);

        x = rho * sin(phi) * cos(theta);
        z = rho * sin(phi) * sin(theta);
        y = rho* cos(phi);
```

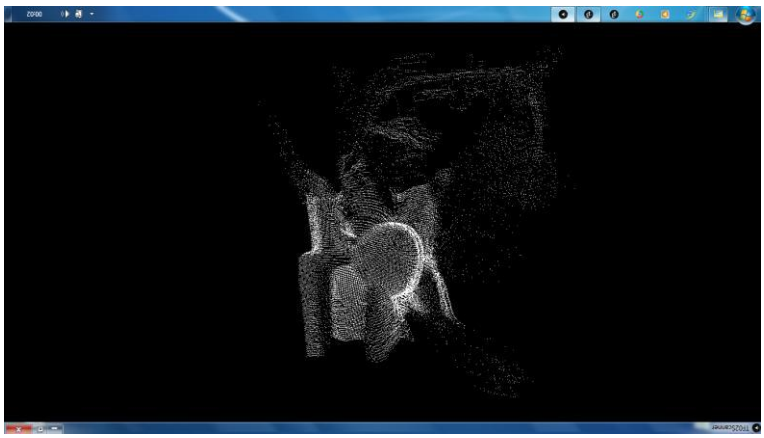
[IMG:G] P3D ground module plotter code

E. Results:



[IMG:H] Result one

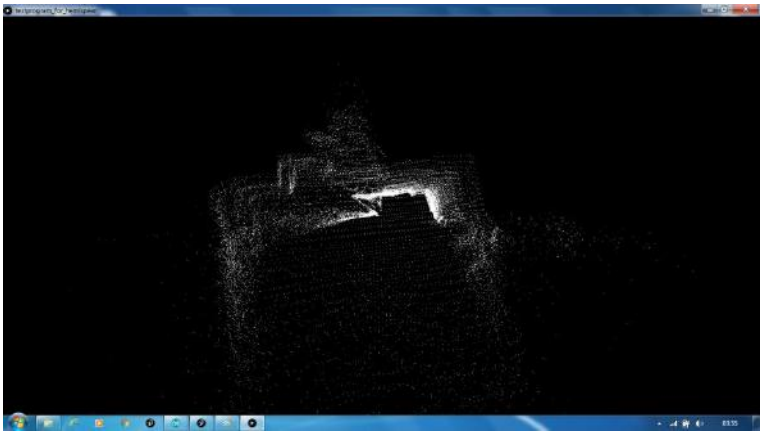
The use of the TF-Mini Lidar sensor along with servo motors provides 2mm precision point accuracy to plot the graph of the virtual window using the P3D processing draw function. This method provides the ability to view plotted 3D graphs using any viewing angle (i.e 360 degrees).



[IMG:I] Result 2

The data is transmitted almost instantly but it takes the device 10 minutes to build a full model because it picks up on the coordinate of each point and the plots that coordinate on a graph to build the model. The first test we did for the result is by placing the device in a room and then allowing it to scan. We did this because we wanted to test the system and make sure that it can pick up on all the points in a simple environment.

This first result allowed us to make sure that the model that is built is up to the standard and all the parts are calibrated properly. Our second test was then run indoors in a dark environment. This allowed us to make sure that all our bases were covered and that we tested the sensor in different environments. The test in the dark allowed us to realize that the test done in regular light had a bit of saturation to it and that the sensor is affected by the amount of light in the environment greatly affecting the sensor's reading. To test this we did our third test outdoors in broad daylight. As we expected the reading was a bit more saturated but our sensor was able to still make a good model with all elements required being visible.



[IMG:J] Result Three

After the initial test's we did our water test over a tub of water. We firstly put elements in the water to recreate the environment of the sea. We did this by first adding sand and then various blocks into the water. After doing this we placed our device over the water and allowed it to scan. We were able to see what was under the water quite clearly, we then decided to raise the sensor and the higher we got the more elements were in the model but there were more unnecessary elements that were in the background we realized that our model works best closer to the surface of the water.

F. Conclusion:

This device aims to provide a cost-effective and simple way for scientists and researchers to research water bodies. We hope to make it an accessible device that will help with marine exploration and help save underwater wildlife. Our device uses Lidar technology to help solve this problem. We use 2 servos that rotate on two different axes to allow for 360-degree rotation and an Arduino board with a transmitter and receiver to remotely send data that allows the device to build and send 3d models to live while scanning.

The device takes in each coordinate and stores that in an Arduino program which is then read by a python file that plots each of those points on a graph with an x, y, and z-axis. Since this plotting process is almost instant, researchers can monitor the device live and keep an eye on any developments in the environment and if the scanning is happening properly. Based on tests we were able to identify that it would be best to run tests during times with low light interference, so researchers can get the most detailed model possible with little to no saturation. Our solution is both extremely efficient and cost-effective because it is extremely portable and can be used by someone with a minimal skill set. The mobility of our device allows researchers to research remote locations without physically being there. This decreases the risk involved and increases the areas where our device can be used.

G. Acknowledgement:

I have been supported to complete this project by various subject matter experts whom I would like to acknowledge. Many thanks to OMOTEC coaches and mentors, who provided feedback and direction to arrive at the final research paper. And finally, thanks to my parents who supported me during this long process, always offering support and love.

H. Author Information:

Aadi Varma [1]
Student at Jumeirah English Speaking School, Completed Jhon Hopkins Innovation and Engineering Undergraduate program. Email:vvermaaadi@outlook.com

Shekhar jain[4]
Chief Executive Officer and Co-founder of On My Own Technology Private Limited Mumbai, shekhar.jain@onmyowntechnology.com

I. References:

I. Leica Geosystems. "Mapping Underwater Terrain With Bathymetric LiDAR." Leica Geosystems,leica-geosystems.com/case-studies/natural-resources/mapping-underwater-terrain-with-bathymetric-lidar, Accessed 27 Oct 2021.
II. "P3D." Processing, processing.org/tutorials/p3d/#3d-transformations. Accessed 29 Oct. 2021.
III. "Arduino Based NRF24 Transmitter-Receiver Setup." Arduino Project Hub, create.arduino.cc/projecthub/Arnov_Sharma_makes/arduino-based-nrf24-transmitter-receiver-setup-d73279. Accessed 6 Nov. 2021.
IV. "NRF24L01 Interfacing with Arduino | Wireless Communication." Arduino Project Hub, create.arduino.cc/projecthub/muhammad-aqib/nrf24l01-interfacing-with-arduino-wireless-communication-0c13d4. Accessed 10 Nov. 2021.
V. "Arduino Lidar Scanning and Java Rendering." Arduino Project Hub, create.arduino.cc/projecthub/TravisLedo/arduino-lidar-scanning-java-rendering-6b2124. Accessed 16 Nov. 2021.