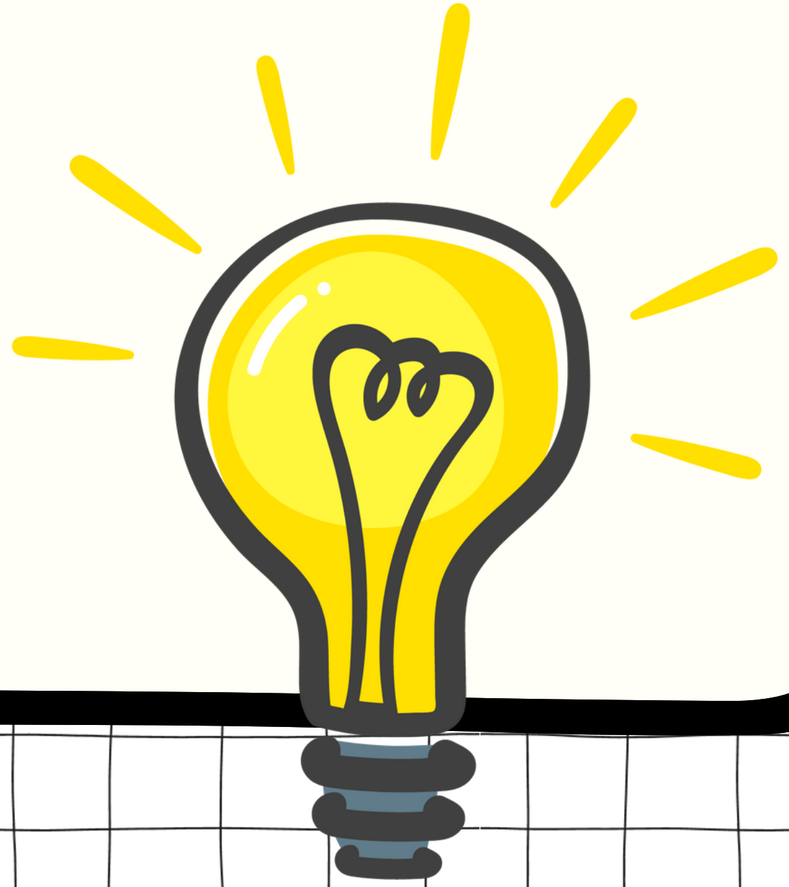
Obstacle Avoidance

ESW project Mid-Evals (TEAM 5)



What all have we done?

- Hardware Interfacing (Hardware API)
- Calibrating sensors
- Robot Assembly (Hardware)
- Publishing data to thingspeak
- Algorithm Coding

Algorithm

- We are using a simplified potential field algorithm for navigating our path. Based on the obstacles and target, net force is calculated which is broken down into components. Y-direction is prioritised and if there is some obstacle present then go in X-direction

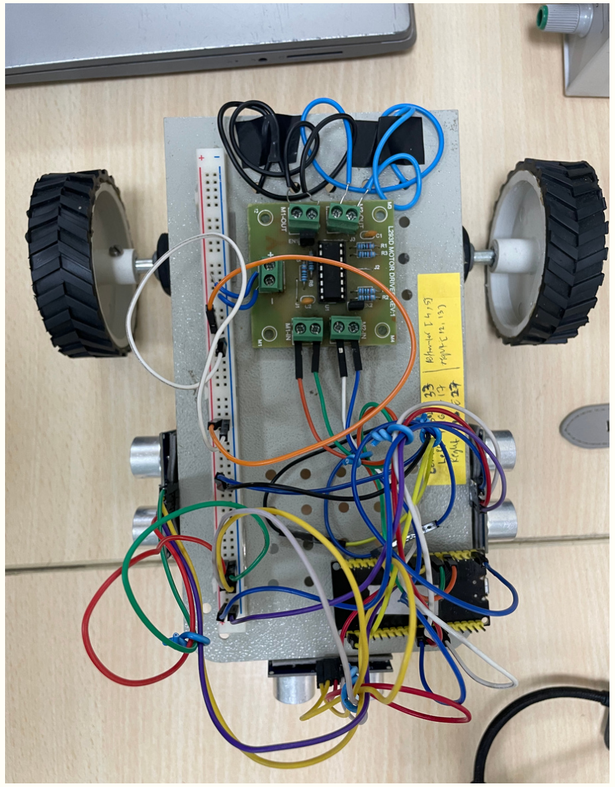
```
void Robo::Algorithm()
{
  while(!(curX == end_x && curY == end_y))
  {

    // take reading and set obstacles
    setObstacleCells();
    Serial.println("obstacles set");

    pair<double, double> netF = GetNetForce();

    // decide next cell (giving priority to neighbours along Y direction)
    pair<int, int> nextCell;
    if(netF.second < 0 &&  0 <= (curY - 1) < grid_cols && grid.Array[curX][curY - 1] == 0)
      nextCell = std::make_pair(curX, curY - 1);
    else if(netF.second > 0 &&  0 <= (curY + 1) < grid_cols && grid.Array[curX][curY + 1] == 0)
      nextCell = std::make_pair(curX, curY + 1);
    else if(netF.first < 0 &&  0 <= (curX - 1) < grid_rows && grid.Array[curX - 1][curY] == 0)
      nextCell = std::make_pair(curX - 1, curY);
    else if(netF.first > 0 &&  0 <= (curX + 1) < grid_rows && grid.Array[curX + 1][curY] == 0)
      nextCell = std::make_pair(curX + 1, curY);
    else // nowhere to move
      Confused();

    Move(nextCell);
  }
}
```

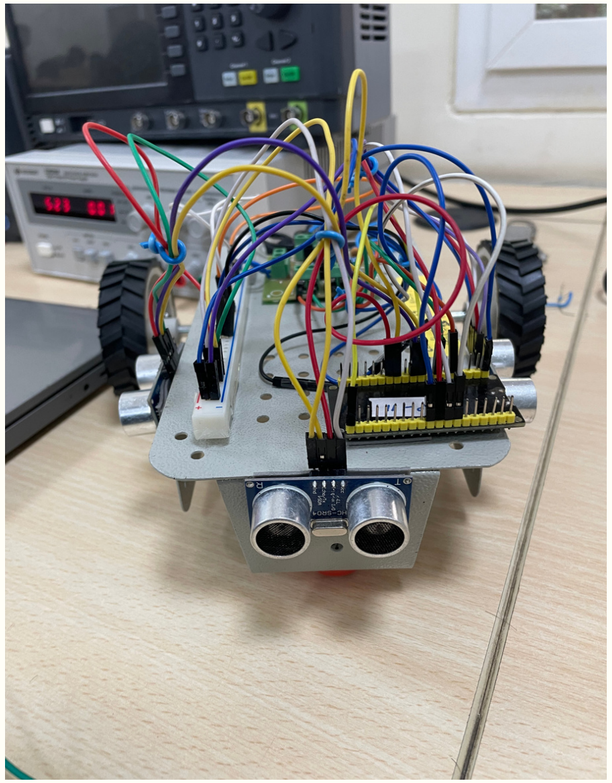
Robot Assembly

01 Placement of all the sensors.

02 Sensors and ESP circuitry.

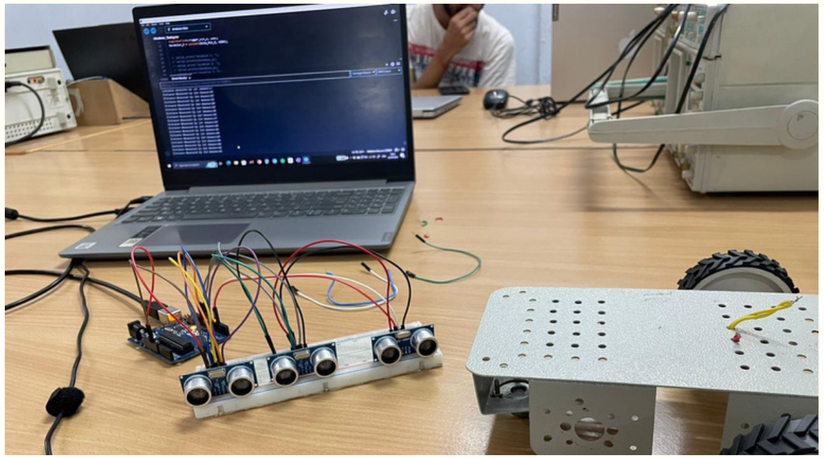
03 Actuators Integration

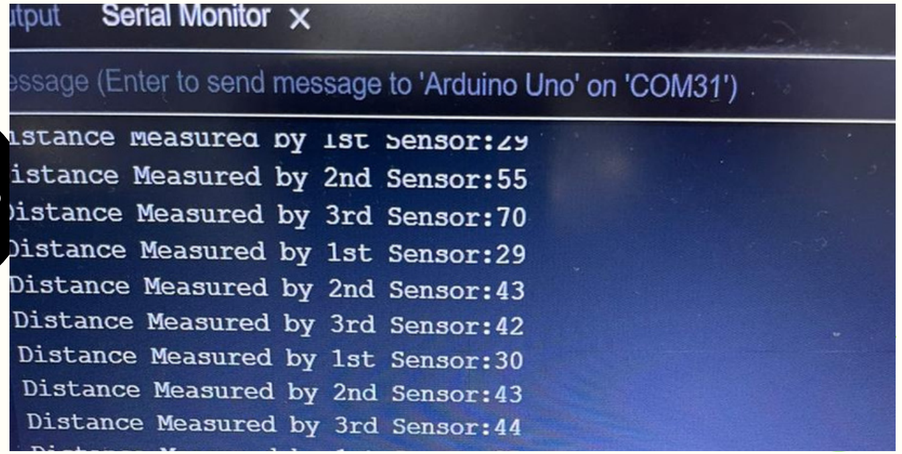




Sensors Calibration

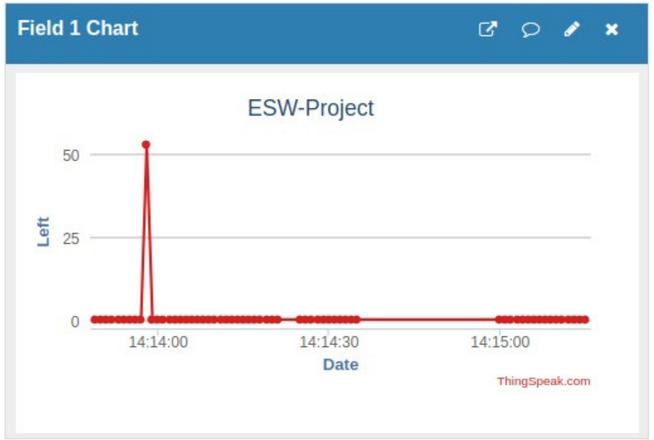
Testing the sensors output on previously known input so as to check for some errors and then adjusting the received data accordingly.

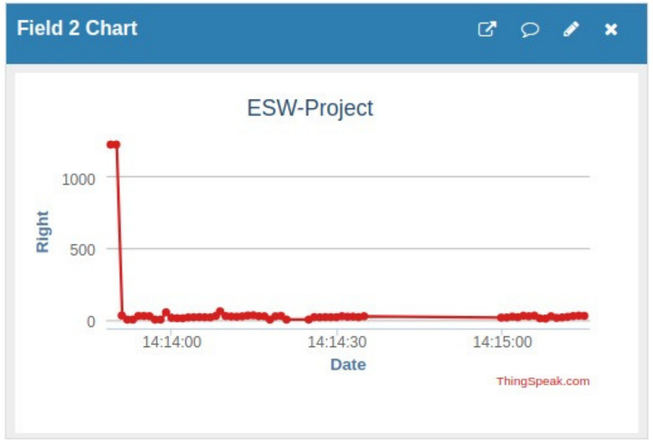


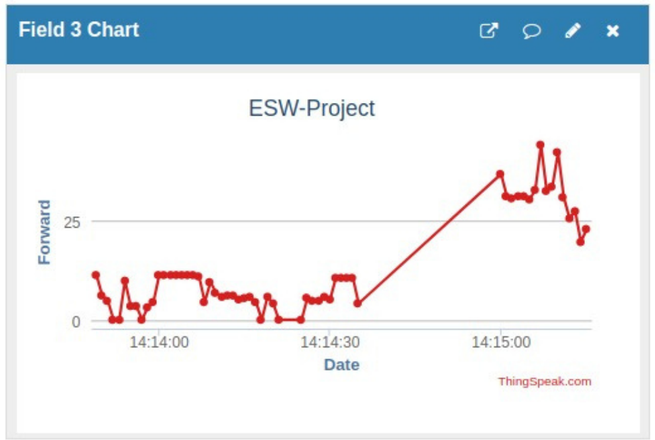


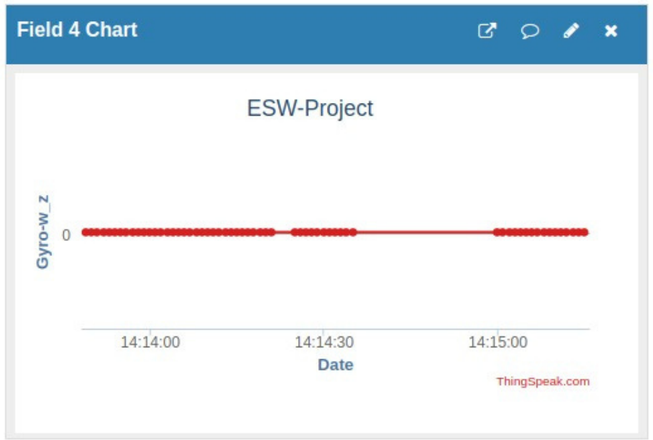
Thingspeak

We are publishing Ultrasonic sensor readings left, right and front, angular velocity along z axis and linear acceleration along y axis.



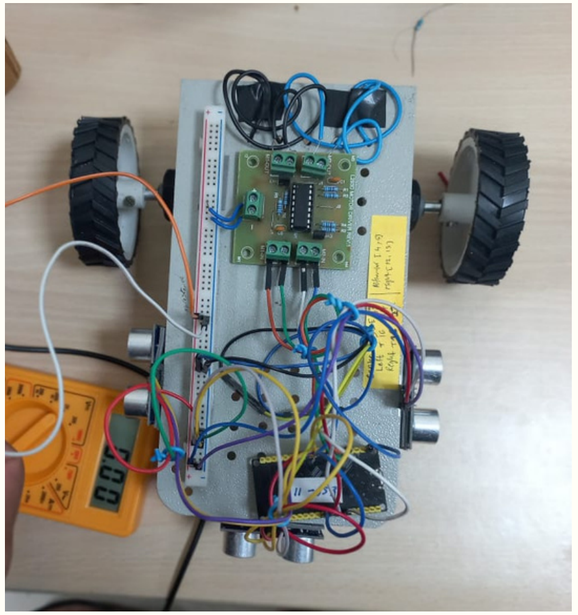






Hardware Interfacing

Connected all the sensors and actuators to our ESP and coded the hardware api to read data from the sensors and direct the actuators based on algorithm logic



```
#include "headers.h"

Ultrasonic::Ultrasonic()
{
}

void Ultrasonic::Setup(uint TRIG_PIN, uint ECHO_PIN, int ERROR) {
  speed_of_sound = 35000; // cmps
  trig_pin = TRIG_PIN;
  echo_pin = ECHO_PIN;
  error = ERROR;

  pinMode(trig_pin, OUTPUT);
  pinMode(echo_pin, INPUT);
  distance = 0;
  time = 0;
}

float Ultrasonic::take_reading() { // waits for 10 microseconds
  digitalWrite(trig_pin, LOW);
  delayMicroseconds(2);
  digitalWrite(trig_pin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trig_pin, LOW);

  time = pulseIn(echo_pin, HIGH);
  distance = ((float) time / 1000000) * speed_of_sound; // in cms
  distance = distance - error;
  return distance;
}
```

Thank you

Team Members:

Vinit : 2022111001

Abhinav : 2022101089

Yash : 2022111009

Samyak : 2022101121