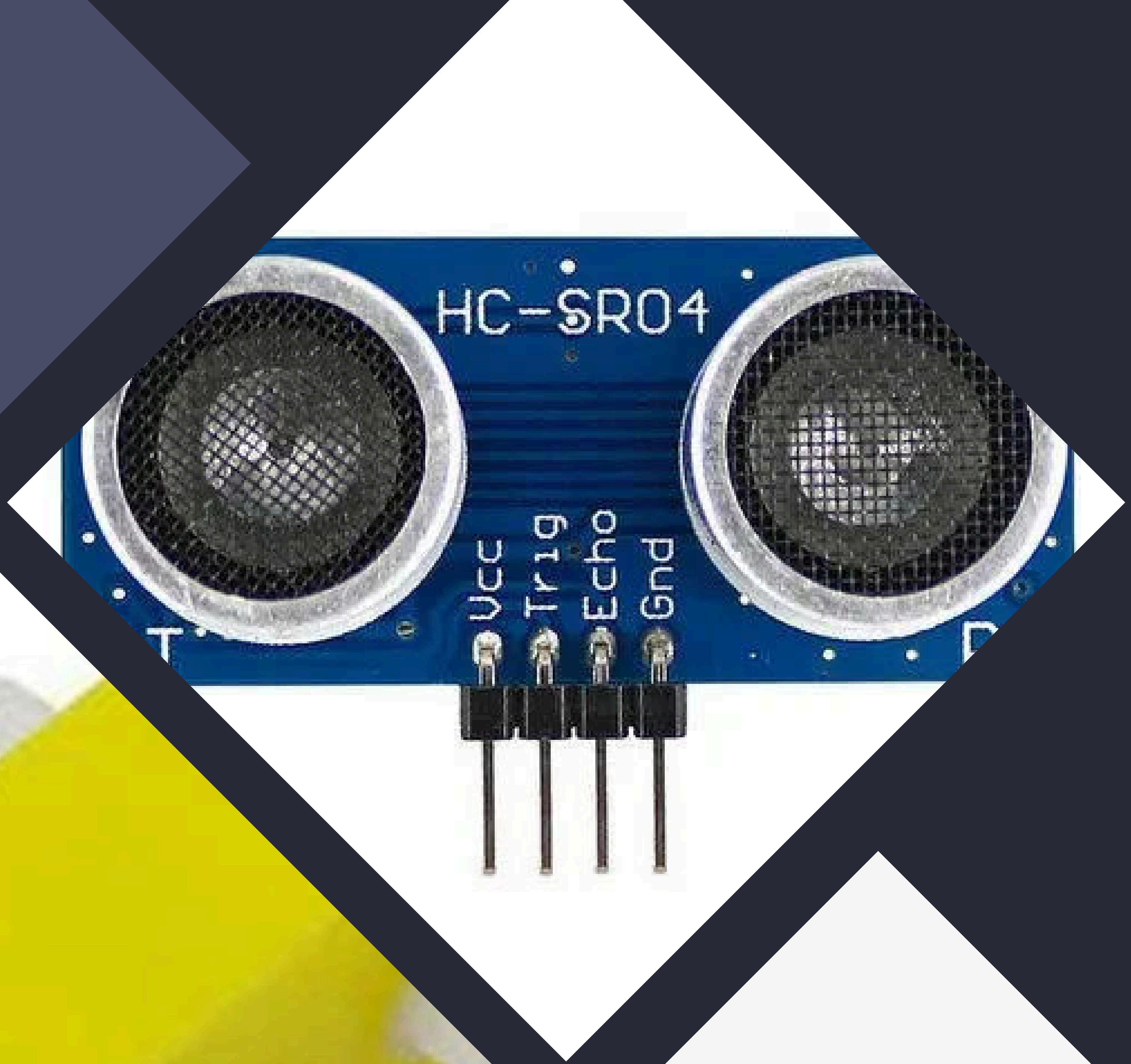
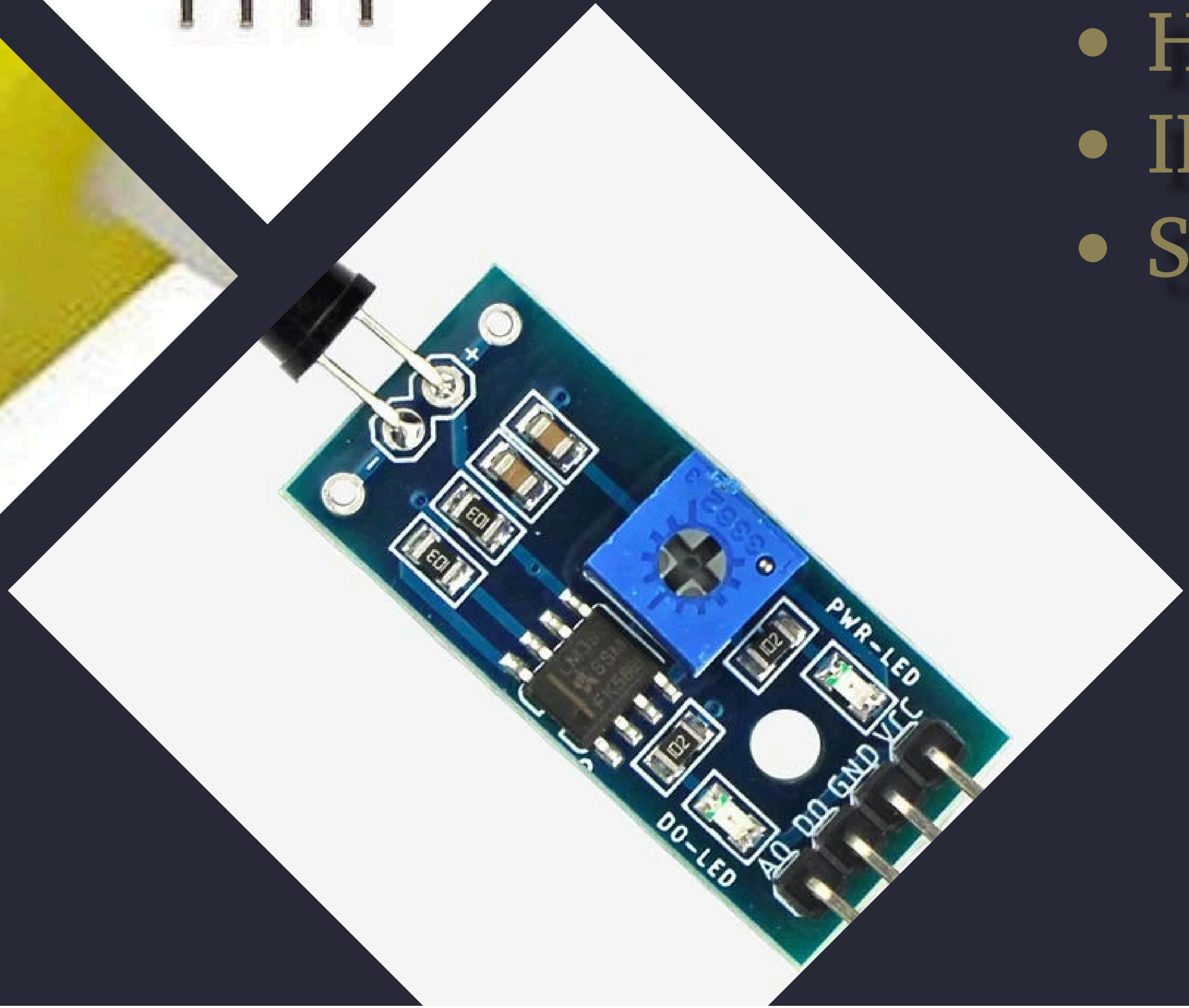
Obstacle Avoidance in Ground Robotics

Team - Mind Bogglers

Navishaa Agarwaal 2023101136
Virat Garg 2023101081
Bibek Dhody 2023101054
Adiya Gaur 2023101139





Components used

- ESP32
- L-Shaped 60 RPM BO Motor with 65X25 Wheel
- L2N3D/L298N Motor Driver
- HC - SR04 Ultrasonic Sensors
- IR sensors
- SG90 Servo Motor

Current progress

- Testing of infrared, ultrasonic sensors,DC motors.
- Collection of data from infrared and ultrasonic sensors
- Integration of the infrared, ultrasonic, and servo motors to operate in synchronization depending on the distance readings

https://drive.google.com/file/d/1cqwTWv88wJXULHHwoubFoiUYzlIzjyUN/view?usp=drive_link

Simulation and Testing


```
Distance: 229.67 cm
Distance: 178.21 cm
Distance: 366.84 cm
Distance: 178.22 cm
Distance: 366.82 cm
Distance: 195.36 cm
Distance: 126.69 cm
Distance: 0.00 cm
Distance: 620.23 cm
Distance: 593.94 cm
Distance: 385.00 cm
Distance: 92.44 cm
Distance: 195.36 cm
Distance: 0.00 cm
Distance: 126.77 cm
Distance: 1600.94 cm
Distance: 178.14 cm
Distance: 30.42 cm
Distance: 435.47 cm
Distance: 126.76 cm
Distance: 0.00 cm
Distance: 0.00 cm
Distance: 193.40 cm
Distance: 126.72 cm
Distance: 126.79 cm
Distance: 401.12 cm
Distance: 75.72 cm
Distance: 195.30 cm
Distance: 92.46 cm
Distance: 113.57 cm
Distance: 126.70 cm
Distance: 2538.70 cm
Distance: 141.02 cm
```

Sensor testing codes

```
#include <Servo.h>


// Create a Servo object
Servo myServo;

// Define the pin that connects to the servo motor
const int servoPin = 12;  // GPIO pin for servo

void setup() {
  // Attach the servo to the pin
  myServo.attach(servoPin);

  // Start serial communication
  Serial.begin(115200);
}

void loop() {
  // Move servo to 0 degrees
  myServo.write(0);
  Serial.println("Servo at 0 degrees");
  delay(1000);  // Wait for a second

  // Move servo to 90 degrees
  myServo.write(90);
  Serial.println("Servo at 90 degrees");
  delay(1000);  // Wait for a second

  // Move servo to 180 degrees
  myServo.write(180);
  Serial.println("Servo at 180 degrees");
  delay(1000);  // Wait for a second
}
```

```
// Define pin numbers for the ultrasonic sensor
const int trigPin = 12;
const int echoPin = 13;

void setup() {
  // Start the serial communication for debugging
  Serial.begin(115200);

  // Set the trigPin as OUTPUT and echoPin as INPUT
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
}

void loop() {
  // Variable to store the duration and distance
  long duration;
  float distance;

  // Clear the trigPin by setting it LOW
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);

  // Trigger the sensor by setting the trigPin HIGH for 10 microseconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Read the time it takes for the echo to return (duration)
  duration = pulseIn(echoPin, HIGH);

  // Calculate the distance in centimeters (speed of sound is 343 m/s)
  // Formula: distance = (duration * speed of sound) / 2
  // Divide by 58 to convert duration in microseconds to distance in cm
  distance = (duration * 0.0343) / 2;

  // Print the distance to the serial monitor
  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println(" cm");

  // Wait for a short time before taking another measurement
  delay(5000);
}
```

```
#include <ESP32Servo.h>

const int trigPin = 5;    // Updated Trig Pin to GPIO 5
const int echoPin = 14;   // Updated Echo Pin to GPIO 14
const int irPin = 27;     // IR Sensor Pin remains GPIO 27

Servo myServo;

// Define the pin for the servo motor
const int servoPin = 18;

// Variables to store the duration and distance
long duration;
float distance;

// Function to get distance from the ultrasonic sensor
float getDistance() {
  // Clear the trigPin by setting it LOW
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);

  // Set the trigPin HIGH for 10 microseconds to send the trigger signal
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Read the echoPin and measure the duration of the echo signal
  duration = pulseIn(echoPin, HIGH);

  // Calculate the distance in centimeters
  float distance = (duration * 0.0343) / 2;
  return distance;
}

// Function to get reading from the IR sensor
bool isPathClear() {
  int irValue = digitalRead(irPin);  // Read IR sensor value
  return irValue == HIGH;            // Assume HIGH means no obstacle (clear path)
}

void setup() {
  // Start serial communication
  Serial.begin(115200);

  // Set trigPin as OUTPUT and echoPin as INPUT
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);

  // Set the IR pin as INPUT
  pinMode(irPin, INPUT);

  // Attach the servo motor
  myServo.attach(servoPin);

  // Set the servo to the center position (90 degrees) initially
  myServo.write(90);
}

void loop() {
  // Get the current distance from the ultrasonic sensor
  distance = getDistance();
  Serial.print("Ultrasonic Distance: ");
  Serial.print(distance);
```

```
  Serial.println(" cm");

  // If ultrasonic sensor detects something close, proceed with avoidance
  if (distance < 30 && distance != 0) {
    Serial.println("Obstacle detected! Checking for clear path...");

    // Rotate the servo to the right in 30-degree increments
    for (int angle = 90; angle <= 150; angle += 30) {
      myServo.write(angle);  // Move servo to the angle
      delay(500);            // Wait for the servo to move

      // Measure the distance again at this angle
      distance = getDistance();
      Serial.print("Distance at ");
      Serial.print(angle);
      Serial.print(" degrees: ");
      Serial.print(distance);
      Serial.println(" cm");

      if (distance > 30 || distance == 0) {
        Serial.println("Clear path found on the right! Move forward.");
        myServo.write(90);  // Return the servo to center position
        delay(500);
        return;             // Exit the loop once a clear path is found
      }
    }

    // Rotate the servo to the left in 30-degree increments
    for (int angle = 90; angle >= 30; angle -= 30) {
      myServo.write(angle);  // Move servo to the angle
      delay(500);            // Wait for the servo to move

      // Measure the distance again at this angle
      distance = getDistance();
      Serial.print("Distance at ");
      Serial.print(angle);
      Serial.print(" degrees: ");
      Serial.print(distance);
      Serial.println(" cm");

      if (distance > 30 || distance == 0) {
        Serial.println("Clear path found on the left! Move forward.");
        myServo.write(90);  // Return the servo to center position
        delay(500);
        return;             // Exit the loop once a clear path is found
      }
    }

    // If no clear path is found, keep checking
    Serial.println("No clear path found! Continue scanning...");
  }
  // If ultrasonic sensor shows distance too large or 0, check IR sensor
  else if (distance == 0 || distance > 300) {
    if (isPathClear()) {
      Serial.println("No object detected. Path is clear.");
    } else {
      Serial.println("IR sensor detects an object! Proceed with avoidance.");
      // You can trigger the avoidance logic here if the IR sensor detects an object
    }
  }

  // Wait for a short time before taking another reading
  delay(500);
}
```

Real-World Applications

- **Autonomous Vehicles:** Self-driving cars use obstacle avoidance to detect and avoid pedestrians, other vehicles, and road obstructions.
- **Aerospace:** Drones and unmanned aerial vehicles (UAVs) rely on obstacle avoidance for safe navigation around buildings, trees, or other aircraft.
- **Agriculture:** Autonomous tractors and harvesters use obstacle avoidance to maneuver around crops, livestock, or other obstacles in the field.
- **Maritime Navigation:** Autonomous boats and submarines use obstacle avoidance to prevent collisions with other vessels or underwater structures.

Contributions

- Virat Garg(2023101081) - Code implementation and managing the github repository
- Bibek Dhody(2023101054) - Hardware implementation(servo motor, ultrasonic sensor, Infrared sensor)
- Navishaa Agarwaal(2023101136) - Preparing the slides and video, integrating dc motor with motor driver
- Aditya Gaur(2023101139) - Data collection

Future works invloving ambition

- Integrating the DC motors with a motor driver using DC power supply.
- Combining the vehicle base with servo motor and ultrasonic sensors.
- Employing IR and ultrasonic sensors for edge detection,preventing the vehicle from falling off tables or elevated surfaces.
- Also planning on showing the vehicle's route on software.

Challenges and Limitations

- Sensor Limitations: Sensors can have limited range, accuracy, or struggle in poor visibility conditions (e.g., fog, rain, or darkness).
- Dynamic Environments: Rapidly changing environments, such as crowded areas with moving obstacles, make real-time obstacle detection and response difficult.
- Terrain Variability: Uneven or slippery surfaces can challenge the robot's ability to navigate safely and maintain balance.
- Processing Delays: Real-time obstacle avoidance requires fast data processing; any delay can lead to collisions, especially in high-speed applications.
- Complex Obstacles: Irregularly shaped or transparent objects can be hard to detect and avoid accurately, leading to potential collisions.
- Power Constraints: Continuous obstacle detection and processing can drain power quickly, limiting the operational time of autonomous robots.

Thanks!