CA PROJECT REPORT

CLASS ID: (114740)

OBSTACLE AVOIDING ROBOT

GROUP MEMBERS:

12439 M.USMAN HUSSAIN

12409 MIRZA SARIM BAIG

12413 SAIFULLAH REHMANI

SUBMITTED TO:

SIR KASHIF BASHIR

TABLE OF CONTENT:

INTRODUCTON:

An obstacle-avoiding robot is an autonomous robot designed to move around and navigate its environment while detecting and avoiding obstacles in its path. It uses Ultrasonic sensor and drivers to perceive the surroundings, make decisions, and control its movement. This type of robot is widely used in various applications such as industrial automation, search and rescue missions, and autonomous vehicles.

COMPONENTS:

1) **Arduino UNO:** The brain of the robot, responsible for processing sensor data and controlling the motors.
2) **Ultrasonic Sensor (HC-SR04):** Used to detect obstacles by measuring the distance to objects using ultrasonic waves.
3) **Servo Motor:** Used to rotate the ultrasonic sensor to scan for obstacles in different directions.
4) **Motors:** Drive the wheels of the robot to enable movement.
5) **LM298N Motor Driver:** An interface between the microcontroller and the DC motors, allowing control of motor direction and speed.
6) **Power Supply:** Provides the necessary power to all components.

7) **Chassis:** The physical structure of the robot that houses all the components.
8) **Wheels:** Attached to the DC motors to facilitate movement.
9) **Jumper Wires:** Used for electrical connections between the components.

SOFTWARE:

- Arduino IDE.
- Proteus.

WORKING:

1) **Initialization:** The microcontroller initializes the motor pins, ultrasonic sensor pins, and the servo motor.

2) **Sensing:** The ultrasonic sensor emits ultrasonic waves and measures the time it takes for the waves to bounce back from an obstacle. The distance to the obstacle is calculated based on the duration of the returned waves.

3) **Decision Making:** If the distance to the obstacle is greater than 15 cm, the robot moves forward.
4) If an obstacle is detected within 15 cm, the robot stops and scans the surroundings using the servo-mounted ultrasonic sensor. Based on the scan, the robot decides to move backward briefly and then turn to find a clear path.

5) **Actuation:** The microcontroller sends signals to the motor driver to control the direction and movement of the motors, enabling the robot to move forward, backward, or turn.

6) **Continuous Loop:** The process of sensing, decision-making, and actuation is repeated continuously, allowing the robot to navigate its environment autonomously.

LIMITATIONS:

Limited Sensor Range: Ultrasonic sensors have a limited detection range, typically up to a few meters. This restricts the robot's ability to detect distant obstacles, especially in open environments.

Obstacle Size and Material: The effectiveness of obstacle detection can vary depending on the size, shape, and material of the obstacles. Small or soft objects may not be detected reliably, leading to potential collisions.

Sensor Interference: Ultrasonic sensors can be affected by environmental factors such as temperature, humidity, and acoustic interference. This interference can reduce the accuracy of distance measurements and lead to false detections.

Single Sensor Limitation: Many obstacle-avoiding robots use a single ultrasonic sensor for detection, which limits their ability to perceive obstacles from multiple directions simultaneously. This can result in blind spots and increase the risk of collisions, especially in complex environments.

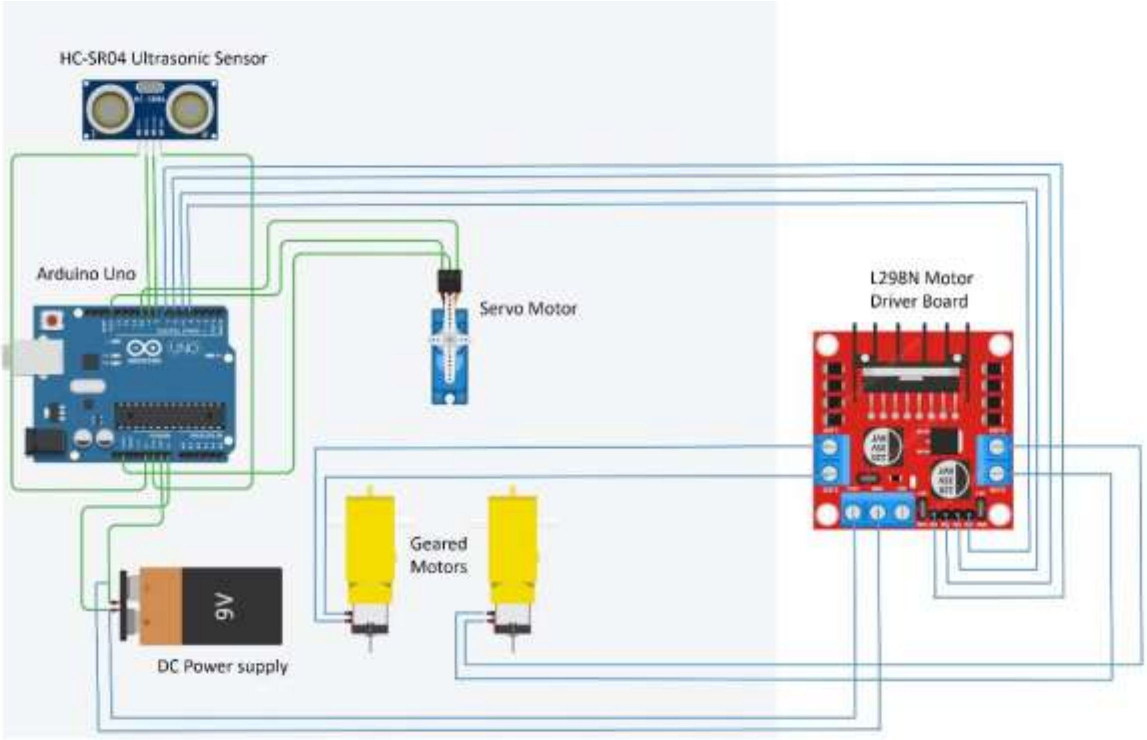
Limited Maneuverability: While obstacle-avoiding robots can navigate around obstacles, their maneuverability may be limited in confined spaces or cluttered environments. They may struggle to find alternative paths or become stuck in tight spaces.

Complex Environments: Obstacle-avoiding robots may face challenges in navigating through complex environments with dynamic obstacles, uneven terrain, or unpredictable obstacles. They may require advanced algorithms and sensors to handle such scenarios effectively.

Power Constraints: Battery-powered obstacle-avoiding robots have limited operating time due to power constraints. As the robot performs sensing, processing, and motor control tasks, it consumes energy, which can affect its runtime and overall performance.

Cost and Complexity: Building a robust obstacle-avoiding robot with advanced sensors and algorithms can be costly and require technical expertise. Balancing cost-effectiveness with performance and reliability is a common challenge in robot design.

CIRCUIT DIAGRAM:



CODE:

```
#include <Servo.h>
Servo Myservo;
#define trigPin 9           // Trig Pin Of HC-SR04
#define echoPin 8        // Echo Pin Of HC-SR04
#define MLa 4                   //left motor 1st pin
#define MLb 5                  //left motor 2nd pin
#define MRa 6               //right motor 1st pin
#define MRb 7               //right motor 2nd pin
long duration, distance;

void setup() {
  Serial.begin(9600);
  pinMode(MLa, OUTPUT);     // Set Motor Pins As O/P
  pinMode(MLb, OUTPUT);
  pinMode(MRa, OUTPUT);
  pinMode(MRb, OUTPUT);
  pinMode(trigPin, OUTPUT);       // Set Trig Pin As O/P To Transmit Waves
  pinMode(echoPin, INPUT);        //Set Echo Pin As I/P To Receive Reflected
Waves
  Myservo.attach(10);
}
void loop()
{
  Serial.begin(9600);
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);       // Transmit Waves For 10us
  delayMicroseconds(10);
  duration = pulseIn(echoPin, HIGH);        // Receive Reflected Waves
  distance = duration / 58.2;                       // Get Distance
  Serial.println(distance);
  delay(10);
  if (distance > 15)               // Condition For Absence Of
Obstacle
  {
    Myservo.write(90);
    digitalWrite(MRb, HIGH);       // Move Forward
    digitalWrite(MRa, LOW);
    digitalWrite(MLb, HIGH);
    digitalWrite(MLa, LOW);
  }
  else if ((distance < 10)&&(distance > 0))            // Condition For Presence
Of Obstacle
  {
    digitalWrite(MRb, LOW);     //Stop
    digitalWrite(MRa, LOW);
    digitalWrite(MLb, LOW);
    digitalWrite(MLa, LOW);
    delay(100);

    Myservo.write(0);
    delay(500);
    Myservo.write(180);
    delay(500);
    Myservo.write(90);
    delay(500);

    digitalWrite(MRb, LOW);     // Move Backward
    digitalWrite(MRa, HIGH);
    digitalWrite(MLb, LOW);
    digitalWrite(MLa, HIGH);
    delay(500);
    digitalWrite(MRb, LOW);        //Stop
    digitalWrite(MRa, LOW);
    digitalWrite(MLb, LOW);
    digitalWrite(MLa, LOW);
    delay(100);
    digitalWrite(MRb, HIGH);     // Move Left
    digitalWrite(MRa, LOW);
    digitalWrite(MLa, LOW);
    digitalWrite(MLb, LOW);
    delay(500);
  }

}
```