INFORMATION TECHNOLOGY UNIVERSITY

Line-following and Obstacle-avoiding robot

Submitted To: Sir Junaid Ashraf

Group Members

Name: Muhammad Umar Akbar (BSCE23061)

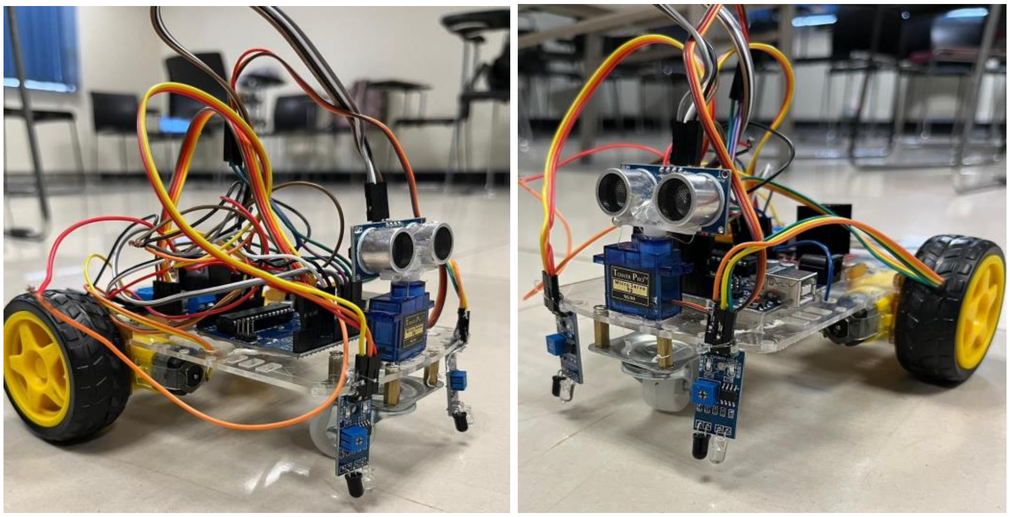
Date of Submission: 1/3/2024

Contents

Components	Tools
1x Arduino Uno	1x Glue Gun
1x Motor Driver(L298)	1x Double Tape
2x Infrared Sensor	1x Black Tape
1x Ultrasonic Sensor	1x Scissors
2x DC Motor(Gear Motor)	
Set of Jumper Wires	
3x Lithium Ion Cells(3.7 Volt each)	
1x Cell Holder	
2x Wheels	
1x Robot Chassis Board	
1x Servo Motor	
1x USB Cable	

Project Description

This Multi-Mode Robot is an innovative project designed to showcase the adaptability and versatility of robotic systems. This robot is engineered to seamlessly switch between two fundamental functions: line following and obstacle avoidance. The key feature that sets this robot apart is its ability to transition between these modes at the user's discretion, achieved by a simple toggle switch. This dual functionality enhances the robot's utility, making it suitable for a wide range of applications and scenarios.



Robot Picture

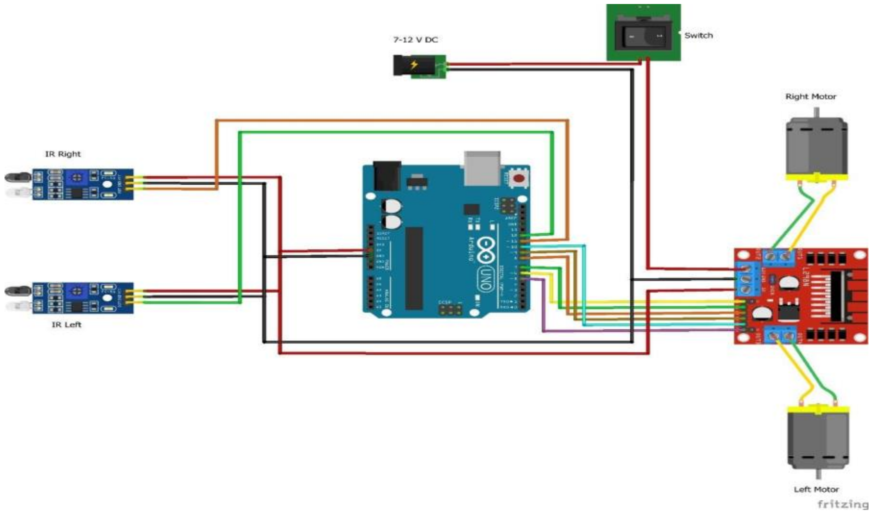
Line Following Robot

Working Principle

The line following robot operates based on the principle of detection of Infrared Waves of certain threshold values. The IR sensors continuously monitor the surface beneath the robot, and the Arduino Uno processes the sensor data to make decisions about the robot's movement. When a sensor detects the black line, the Arduino adjusts the motor speeds to keep the robot on the correct path. The line follower robot senses a black line by using a sensor and then sends the signal to Arduino. Then Arduino drives the motor according to sensors' output. We are using PWM to convert analog signals into digital signals for accurate functionality of the robot as it is helping to control motor speed.

Programming

The programming to achieve the objective of creating a line following robot involves creating algorithms for the robot's behavior. We initialized the pins before the void setup and void loop functions. After this, we declared the built-in functions of output of the motors and built-in functions for input of the IR Sensors. According to the code, motors and sensors will work with digital signals. The code is executed when we turn the switch on. The code includes conditions to read sensor values and it instructs the robot to move depending on those sensor values. If both sensors detect the presence of the black line, the robot stops. If the left sensor detects the black line but the right does not, the robot will move right because the output voltage on the right motor will be high and will be low on the left motor. If the right sensor detects the black line but left does not, the robot will move left because the output voltage on the left motor will be high and will be low on the right motor.



Schematic Diagram

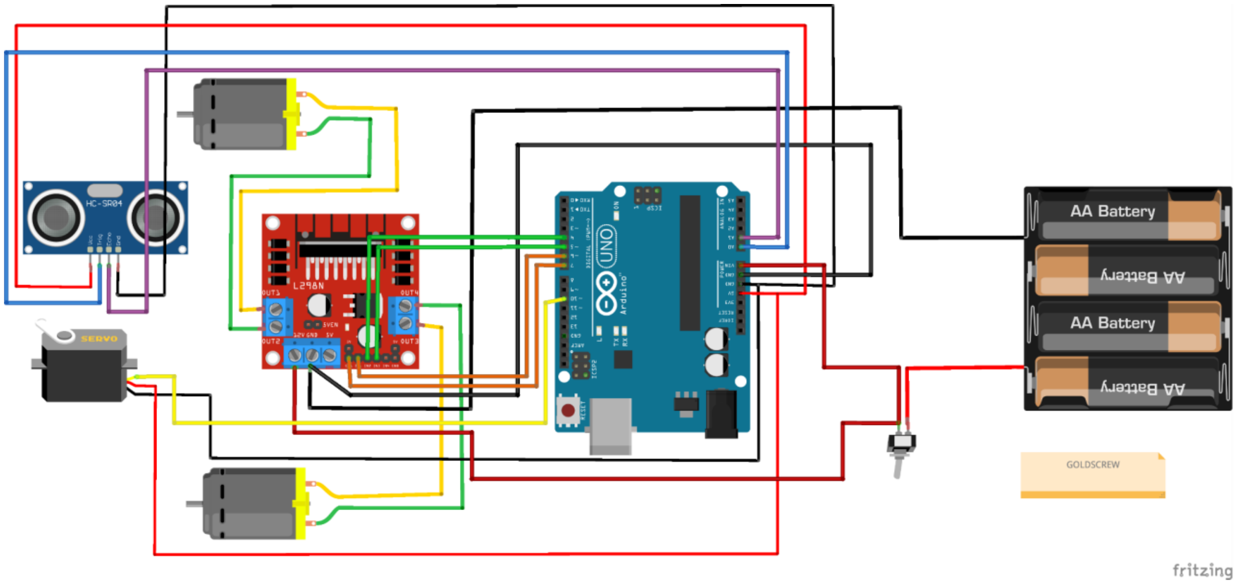
Obstacle Avoiding Robot

Working Principle

The Obstacle Avoiding Robot functions by employing the principle of sending out a sound wave. If the sound wave is received back, it indicates the presence of an obstacle in front of the robot, prompting it to stop. An ultrasonic sensor is utilized by the robot to measure the distance between itself and potential obstacles. The Arduino Uno processes the data from the sensor and, based on the proximity of obstacles, issues commands to the motors, directing the robot to navigate away from the detected obstacles. A servo motor is also attached with the ultrasonic sensor so that the sensor can easily rotate to check for any potential obstacles besides the robot.

Programming

The Arduino code manages the robot's actions by collecting and understanding distance data from the sensor. A servo motor moves the sensor in a sweeping motion, expanding its detection range. In real-time, the code decides what to do based on the measured distances. If an obstacle is close, the robot changes its path on the spot to avoid collisions, making sure it moves safely without obstacles in its way. This combination of the servo motor and ultrasonic sensor improves the robot's navigation skills in various environments.



Schematic Diagram

Line Following Code

```
#define IR_SENSOR_RIGHT 13
#define IR_SENSOR_LEFT 12
#define MOTOR_SPEED 65
int enableRightMotor=9;
int rightMotorPin1=2;
int rightMotorPin2=3;
int enableLeftMotor=10;
int leftMotorPin1=4;
int leftMotorPin2=5;

void setup()
{
  pinMode(enableRightMotor, OUTPUT);
  pinMode(righ tMotorPin1, OUTPUT);
  pinMode(rightMotorPin2, OUTPUT);
  pinMode(enableLeftMotor, OUTPUT);
  pinMode(leftMotorPin1, OUTPUT);
  pinMode(leftMotorPin2, OUTPUT);
  pinMode(IR_SENSOR_RIGHT, INPUT);
  pinMode(IR_SENSOR_LEFT, INPUT);
  rotateMotor(0,0);
}


void loop()
{
  int rightIRSensorValue = digitalRead(IR_SENSOR_RIGHT);
  int leftIRSensorValue = digitalRead(IR_SENSOR_LEFT);
  if (rightIRSensorValue == LOW && leftIRSensorValue == LOW)
  {
    rotateMotor(MOTOR_SPEED, MOTOR_SPEED);
  }
  else if (rightIRSensorValue == HIGH && leftIRSensorValue == LOW )
  {
      rotateMotor(-MOTOR_SPEED, MOTOR_SPEED);
  }
  else if (rightIRSensorValue == LOW && leftIRSensorValue == HIGH )
  {
      rotateMotor(MOTOR_SPEED, -MOTOR_SPEED);
  }
  else
  {
    rotateMotor(0, 0);
  }
}


void rotateMotor(int rightMotorSpeed, int leftMotorSpeed)
{

  if (rightMotorSpeed < 0)
  {
    digitalWrite(rightMotorPin1,LOW);
    digitalWrite(rightMotorPin2,HIGH);
  }
  else if (rightMotorSpeed > 0)
  {
    digitalWrite(rightMotorPin1,HIGH);
    digitalWrite(rightMotorPin2,LOW);
  }
  else
  {
    digitalWrite(rightMotorPin1,LOW);
    digitalWrite(rightMotorPin2,LOW);
  }

  if (leftMotorSpeed < 0)
  {
    digitalWrite(leftMotorPin1,LOW);
    digitalWrite(leftMotorPin2,HIGH);
  }
  else if (leftMotorSpeed > 0)
  {
    digitalWrite(leftMotorPin1,HIGH);
    digitalWrite(leftMotorPin2,LOW);
  }
  else
  {
    digitalWrite(leftMotorPin1,LOW);
    digitalWrite(leftMotorPin2,LOW);
  }
  analogWrite(enableRightMotor, abs(rightMotorSpeed));
  analogWrite(enableLeftMotor, abs(leftMotorSpeed));
}
```

Obstacle Avoidance Code

```
#include <Servo.h>
Servo myservo;
byte servostart = 105;
int distanceleft = 0;
int distanceright = 0;
long t, cm;

#define ENA 9
#define IN1 2
#define IN2 3

#define IN3 4
#define IN4 5
#define ENB 10

#define Trig 6
#define Echo 7

#define Speed 255

void setup() {
  myservo.attach(11);
  pinMode(ENA, OUTPUT);
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(ENB, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
  pinMode(Trig, OUTPUT);
  pinMode(Echo, INPUT);
  Serial.begin(9600);
  start();
}
void loop() {
  getdistance();
  Serial.println(cm);
  int leftdistance = 0;
  int rightdistance = 0;

  if (cm <= 20) {
    Stop();
    delay(200);
    leftdistance = leftsee();
    rightdistance = rightsee();

    if (leftdistance >= rightdistance) {
      turnleft();
      delay(200);
      Stop();
    } else {
      turnright();
      delay(200);
      Stop();
    }
  } else {
    forward();
    Serial.println("forward");
  }
}

void forward() {
  analogWrite(ENA, Speed);
  analogWrite(ENB, Speed);
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);
  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);
}
```

```
void turnright() {
  analogWrite(ENA, Speed);
  analogWrite(ENB, Speed);
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);
}
void turnleft() {
  analogWrite(ENA, Speed);
  analogWrite(ENB, Speed);
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, HIGH);
}
void Stop() {
  analogWrite(ENA, 0);
  analogWrite(ENB, 0);
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, LOW);
}

void start() {
  delay(3000);
  for (int a = 0; a < 4; a++) {
    myservo.write(servostart);
    delay(50);
    myservo.write(40);
    delay(50);
    myservo.write(90);
    delay(50);
    myservo.write(servostart);
  }
}

int leftsee() {
  myservo.write(servostart);
  delay(1000);
  myservo.write(175);
  delay(1000);
  distanceleft = getdistance();
  myservo.write(servostart);
  return distanceleft;
}
int rightsee() {
  myservo.write(servostart);
  delay(1000);
  myservo.write(5);
  delay(1000);
  distanceright = getdistance();
  myservo.write(servostart);
  return distanceright;
}
int getdistance() {
  digitalWrite(Trig, LOW);
  delayMicroseconds(4);
  digitalWrite(Trig, HIGH);
  delayMicroseconds(10);
  digitalWrite(Trig, LOW);
  t = pulseIn(Echo, HIGH);
  cm = t / 29 / 2;
  return cm;
}
```