Design and Implementation of an Ultrasonic Sensor Robot Car Controlled by L293D IC Motor Driver

Mahendra Khinchi

26/04/2024

Roll No. EE23BTECH11039

Design and Implementation of an Ultrasonic Sensor Robot Car Controlled by L293D IC Motor Driver

Abstract: This lab report details the planning, building, and testing of a robot automobile using an ultrasonic sensor that makes use of an L293D IC motor driver. The goal of the project was to build a flexible, autonomous vehicle that could navigate its environment by using ultrasonic sensors to identify impediments and an L293D motor driver for effective motor management. The components utilized, the circuitry design, the building procedure, the testing and performance evaluation outcomes are all described in the report.

Introduction: Automation and robotics are becoming more and more common in a variety of disciplines, from hobby projects to industrial applications. Because of their adaptability and potential for practical use, robot automobiles are a popular platform for investigating robotics concepts. Robots can be reliably detected by ultrasonic sensors, and their movement can be precisely controlled by motor drivers like the L293D.

Materials and Methods: The following materials were used in the construction of the ultrasonic sensor robot car:

1. Arduino Uno microcontroller board
2. L293D IC motor driver
3. Ultrasonic sensors (HC-SR04)
4. DC motors
5. Chassis

6. Wheels
7. Battery pack
8. Jumper wires
9. Breadboard

The L293D motor driver is used in the circuitry to link the DC motors and ultrasonic sensors to the Arduino Uno. The robot car's front was equipped with ultrasonic sensors for obstacle detection, and its wheels were powered by DC motors for control of movement. Based on input from the Arduino Uno, the L293D motor driver was used to control the motors' speed and direction.

To read data from the ultrasonic sensors, process the data, and adjust the motors accordingly, an Arduino code was created. The sketch comprised algorithms for obstacle avoidance and rudimentary navigation, and it interfaced with the ultrasonic sensors using the NewPing library.

Results: After building and programming were finished, the ultrasonic sensor robot automobile was able to successfully traverse its surroundings and avoid impediments that it had picked up on with its ultrasonic sensors. The robot car's ability to navigate was made possible by the L293D motor driver, which gave the DC motors smooth and accurate control.

Discussion: Using ultrasonic sensors in conjunction with the L293D motor driver worked out well to build an autonomous robot automobile that could avoid obstacles. Still, there are a few places that might use further work and experimentation:

1. optimizing the obstacle avoidance algorithm's performance in various settings.
2. incorporating extra sensors with broader perceptual capabilities, like infrared or camera modules.
3. investigating sophisticated motor control methods for more fluid motion and improved dexterity.

Conclusion:

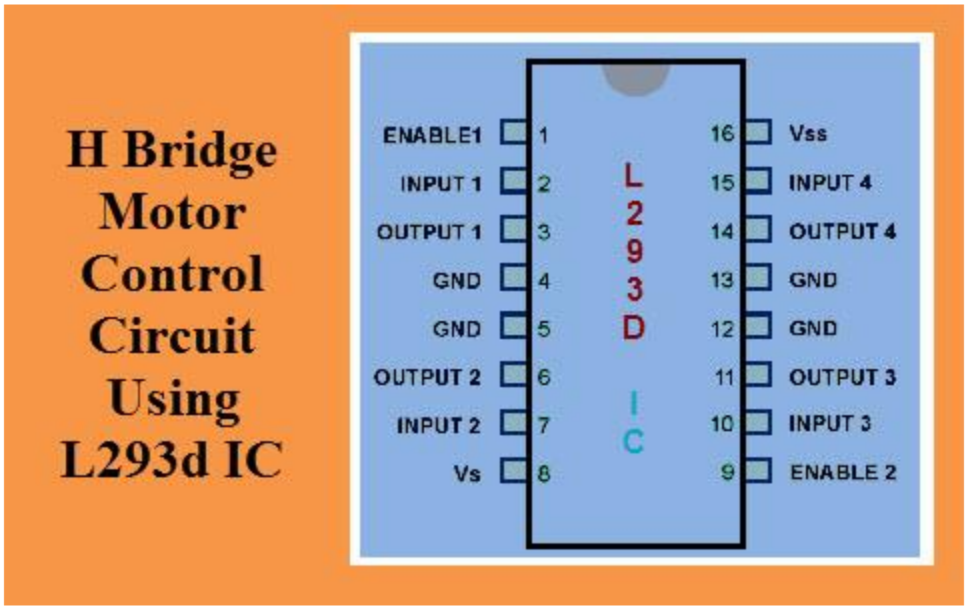
In conclusion, the L293D IC motor driver-controlled ultrasonic sensor robot car shows that it is possible to build autonomous robotic systems for a range of uses. around the integration of precise motor control and sensor inputs, the robot automobile is capable of maneuvering around its environment and dodging obstacles. Performance and functionality can be improved with additional experimentation and refining.

References: [1] L293D Datasheet.

[2] HC-SR04 Ultrasonic Sensor Datasheet.

[3] Arduino Uno Official Website.

[4] NewPing Library Documentation.



Code of Arduino uno:

```
#define IN1 8
#define IN2 9
#define ENA A5
#define IN3 10
#define IN4 11
#define ENB A1
#define IN6 4
#define IN5 5
#define IN7 6
#define IN8 7
#define ENC A2
#define END A3
#define TRIG_PIN 2
#define ECHO_PIN 3
double previousDistance = 400.0;
int speed =1024;
unsigned long lastTimeUltrasonicTrigger = millis();
unsigned long ultrasonicTriggerDelay = 60;

volatile unsigned long pulseInTimeBegin;
volatile unsigned long pulseInTimeEnd;
volatile bool newDistanceAvailable = false;
void movingforward(){
  digitalWrite(IN1,HIGH);
digitalWrite(IN2,LOW);
  digitalWrite(IN3,HIGH);
  digitalWrite(IN4,LOW);
  digitalWrite(IN5,HIGH);
  digitalWrite(IN6,LOW);
  digitalWrite(IN7,HIGH);
  digitalWrite(IN8,LOW);

}
void movingbackward(){
  digitalWrite(IN2,HIGH);
  digitalWrite(IN1,LOW);
  digitalWrite(IN4,HIGH);
  digitalWrite(IN3,LOW);
  digitalWrite(IN6,HIGH);
  digitalWrite(IN5,LOW);
  digitalWrite(IN8,HIGH);
  digitalWrite(IN7,LOW);

}
void movingleft(){
  digitalWrite(IN1,HIGH);
  digitalWrite(IN2,LOW);
  digitalWrite(IN3,HIGH);
  digitalWrite(IN4,LOW);
  digitalWrite(IN5,HIGH);
  digitalWrite(IN6,LOW);
digitalWrite(IN7,LOW);
  digitalWrite(IN8,HIGH);


}
void movingright(){
  digitalWrite(IN1,HIGH);
  digitalWrite(IN2,LOW);
  digitalWrite(IN3,HIGH);
  digitalWrite(IN4,LOW);
  digitalWrite(IN5,LOW);
  digitalWrite(IN6,HIGH);
  digitalWrite(IN7,HIGH);
  digitalWrite(IN8,LOW);


}
void avoidingobs(double dist){
  if(dist<40){
    movingright();

  }
  movingforward();

}
```

```
void Trig_pin(){
digitalWrite(TRIG_PIN,LOW);
  delay(2);
  digitalWrite(TRIG_PIN,HIGH);
  delay(10);
  digitalWrite(TRIG_PIN,LOW);
}


double getUltrasonicDistance()
{
  double durationMicros = pulseInTimeEnd - pulseInTimeBegin;
  double distance = durationMicros / 58.0; // cm (148.0: inches)
  if (distance > 400.0) {
    return previousDistance;
  }
  distance = previousDistance * 0.7 + distance * 0.3;
  previousDistance = distance;
  return distance;
}
void echoPinInterrupt()
{
  if (digitalRead(ECHO_PIN) == HIGH) { // start measuring
    pulseInTimeBegin = micros();
  }
  else { // stop measuring
    pulseInTimeEnd = micros();
    newDistanceAvailable = true;
  }
}

void setup() {
  Serial.begin(115200);
  pinMode(IN1, OUTPUT);
  pinMode(IN2,OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4,OUTPUT);
```

```
 pinMode(IN5, OUTPUT);
 pinMode(IN6,OUTPUT);
 pinMode(IN7, OUTPUT);
 pinMode(IN8,OUTPUT);
 pinMode(ENA,OUTPUT);
 pinMode(ENB,OUTPUT);
 pinMode(ENC,OUTPUT);
 pinMode(END,OUTPUT);
 digitalWrite(IN1,LOW);
 digitalWrite(IN2,LOW);
 digitalWrite(IN3,LOW);
 digitalWrite(IN4,LOW);
 analogWrite(ENA,0);
 analogWrite(ENB,0);
 analogWrite(ENC,0);
 analogWrite(END,0);
 attachInterrupt(digitalPinToInterrupt(ECHO_PIN),
                 echoPinInterrupt,
                 CHANGE);
                 Serial.println(IN1);
   Serial.print(IN2);
   Serial.println(IN3);
   Serial.print(IN4);
Serial.println(IN5);
   Serial.print(IN6);
   Serial.println(IN7);
   Serial.print(IN8);
   Serial.print(analogRead(ENA));


}

void loop() {
 analogWrite(ENA,speed);
 analogWrite(ENB,speed);
 analogWrite(ENC,speed);
 analogWrite(END,speed);
 movingforward();
```

```
  unsigned long timeNow=millis();
  if(timeNow-lastTimeUltrasonicTrigger>ultrasonicTriggerDelay){
    lastTimeUltrasonicTrigger+= ultrasonicTriggerDelay;
    Trig_pin();

  }
  if (newDistanceAvailable) {
    newDistanceAvailable = false;
    double distance = getUltrasonicDistance();

    avoidingobs(distance);
  }


}


// put your main code here, to run repeatedly:
```