PROJECT

MAZE SOLVING ROBOT

PROJECT REPORT

Submitted by

KURUBA BULLEY BHARADWAJ

Mail ID:

kbulleybharadwaj_eee180223@mgit.ac.in

ELECTRICAL AND ELECTRONICS ENGINEERING

In

MAHATMA GANDHI INSTITUTE OF TECHNOLOGY-HYDERABAD

TITLE: MAZE SOLVING ROBOT

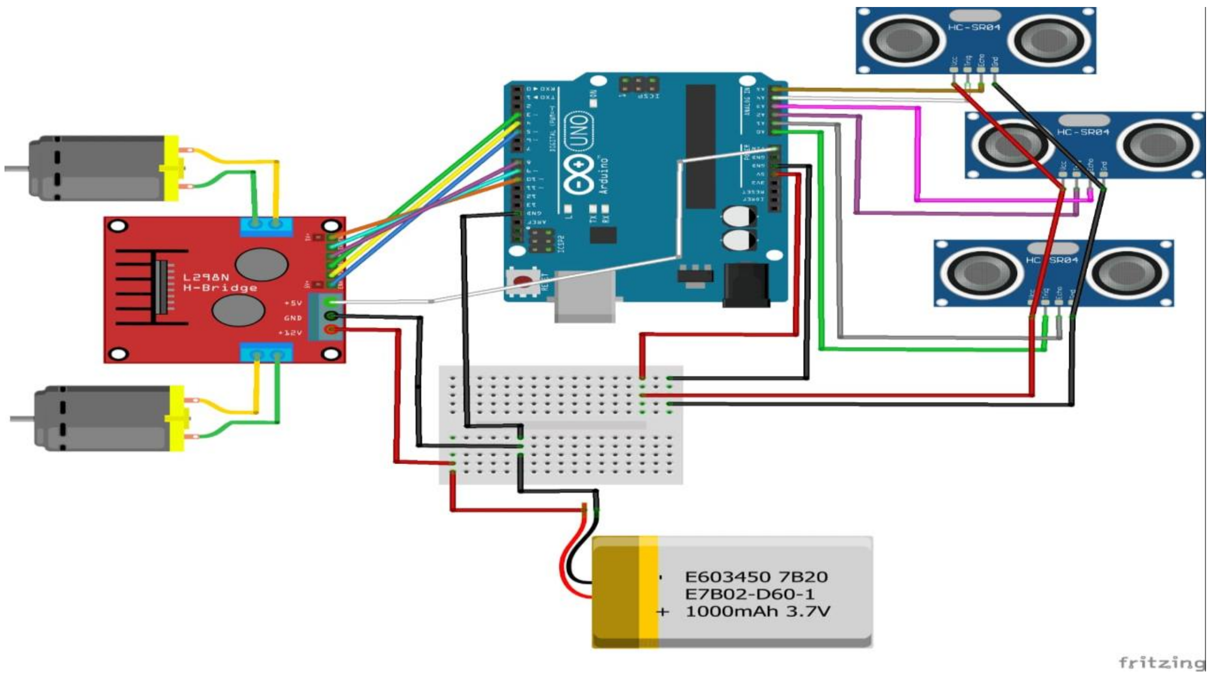
COMPONENTS REQUIRED :

- Arduino uno
- Arduino Uno Cable
- Ultrasonic Sensors
- L293D Motor Driver
- Two Wheel Robot Chassis
- TWO DC MOTORS
- JUMPER WIRES
- BREAD BOARD
- 9V BATTERY

SERVICES USED :

- ARDUINO IDE

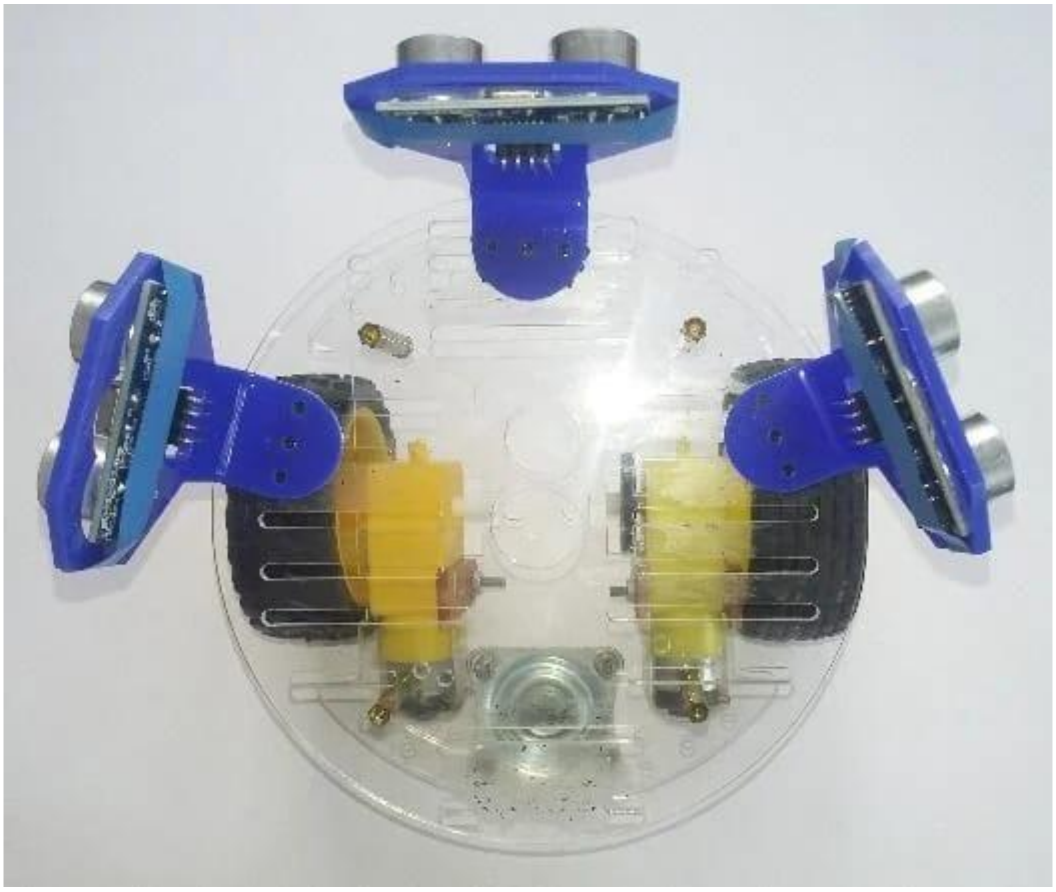
CIRCUIT DIAGRAM :



PROCEDURE :

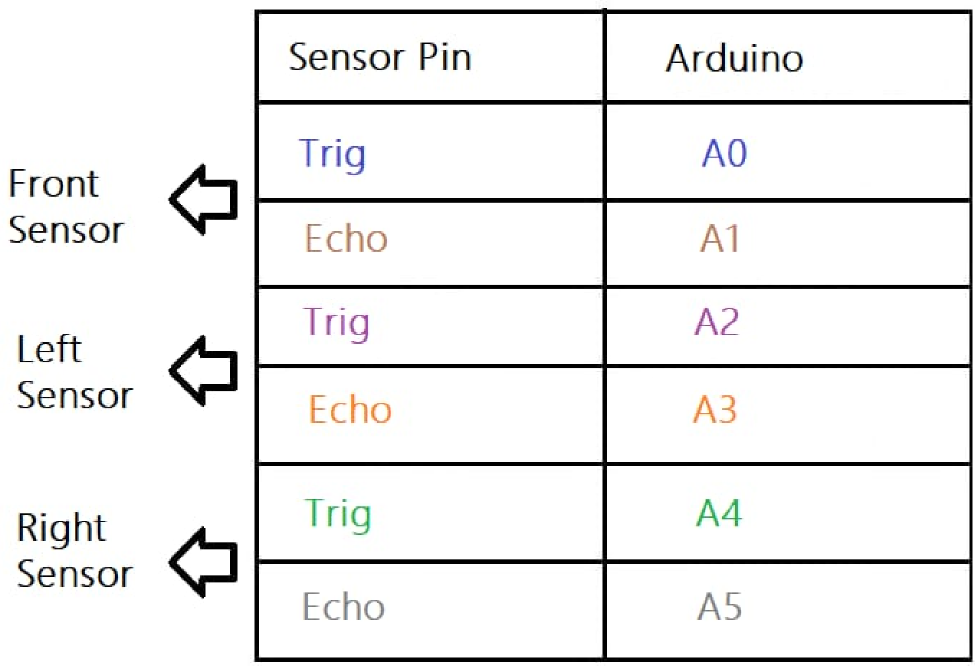
STEP – 1 : ASSEMBLE CHASSIS PARTS AND ULTRASONIC SENSORS

Attach the 3 utrasonic sensors in three different directions (front , right & left) . fix the robot chasis tightly and attach the sensors with glue.



STEP – 2 : INTERFACING ULTRASONIC SENSORS TO ARDUINO

Wire the trigger pin and echo pin of the sensors to analog pins of the Arduino. To power up the sensors, connect all the ground pins to a common ground rail and all the 5V pins to a common 5V rail on the breadboard.

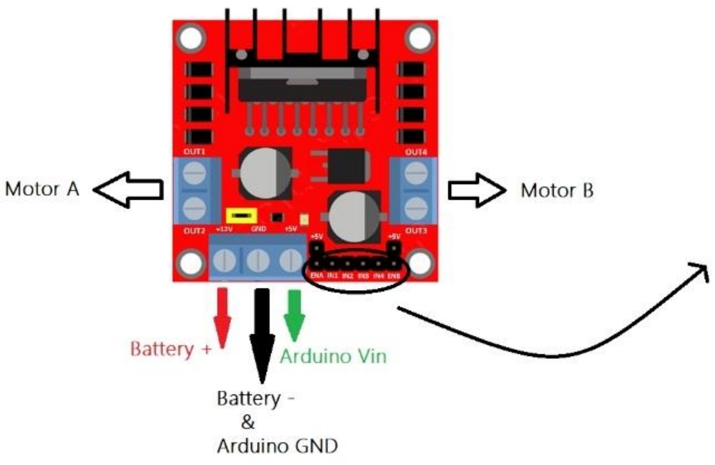


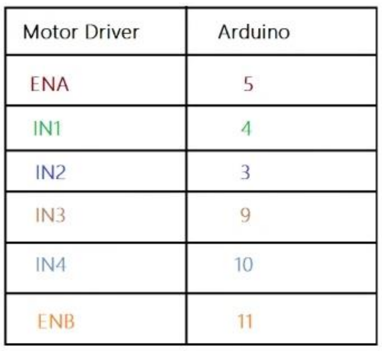
	Sensor Pin	Arduino
Front Sensor	Trig	A0
	Echo	A1
Left Sensor	Trig	A2
	Echo	A3
Right Sensor	Trig	A4
	Echo	A5

Here, the ultrasonic sensors are basically the eyes of the robot which help it to find any obstacles around it. Ultrasonic sensors measure the distance to the obstacle for the robot to be able to navigate autonomously. You can implement your maze solving robot with just one or two ultrasonic sensors also, but having three sensors ensures more precise movements.

STEP – 3 : INTERFACING THE MOTOR DRIVR TO ARDUINO

The motor driver has seven screw terminals, which makes the interfacing easy.





Motor Driver	Arduino
ENA	5
IN1	4
IN2	3
IN3	9
IN4	10
ENB	11

The L293D motor driver helps us to control the speed and direction of the two DC motors simultaneously. It acts as an interface between the motor and the Arduino. The motor driver has 4 input pins to control the rotational direction of the motor. 'Enable A' and 'Enable B' pins have the responsibility to enable and control the speed of the motors.

ARDUINO CODE :

```
const int trigPin1 = 6;//front

const int echoPin1 = 7;

const int trigPin2 = 10; //left

const int echoPin2 = 11;

const int trigPin3 = 8;//right

const int echoPin3 = 9;

const int in1 = 2;

const int in2 = 3;
const int in3 = 4;

const int in4 = 5;


#define DIS 7


void setup()

{

pinMode(trigPin1, OUTPUT);

pinMode(echoPin1, INPUT);

pinMode(trigPin2, OUTPUT);

pinMode(echoPin2, INPUT);

pinMode(trigPin3, OUTPUT);

pinMode(echoPin3, INPUT);

pinMode (in1, OUTPUT);

pinMode (in2, OUTPUT);

pinMode (in3, OUTPUT);

pinMode (in4, OUTPUT);


}

void loop()

{

  if (FrontSensor ()>DIS && RightSensor () >DIS && LeftSensor () >DIS)

  {forward();}

 //else if ( FrontSensor () > DIS && RightSensor () < DIS && LeftSensor ()< DIS)

//{

  //forward();
```

```
//}

else if ( FrontSensor() < DIS && RightSensor () <DIS && LeftSensor ()<DIS) // obstacle infront
of all 3 sides

{

reverse ();

delay(500);

if((LeftSensor())>(RightSensor()) )

turn_left();

else

turn_right();

//then reverse

}

else if (FrontSensor() <DIS && RightSensor () <DIS && LeftSensor ()>DIS) // obstacle on right
and front sides

{

turn_left (); // turn left side

}

else if (FrontSensor() <DIS && RightSensor () >DIS && LeftSensor ()<DIS) // obstacle on left
and front sides

{

turn_right (); // turn right side

}

else if (FrontSensor() <DIS && RightSensor () >DIS && LeftSensor ()>DIS) // obstacle on front
sides

{

turn_left ();

delay(500);
```

```
forward();// then turn right //*******************

}

else if (FrontSensor() >DIS && RightSensor () >DIS && LeftSensor ()<DIS) // obstacle on left
sides

{

turn_right(); // then turn right and then forward

delay(500);

forward();

}

else if (FrontSensor() >DIS && RightSensor () <DIS && LeftSensor ()>DIS) // obstacle on right
sides

{

turn_left (); // then turn left and then right

delay(500);

forward();

}

else

{

forward();

}

}

void forward ()

{

digitalWrite(in1, HIGH);

digitalWrite(in2, LOW);

digitalWrite(in3, HIGH);

digitalWrite(in4, LOW);
```

```
}

void turn_left ()

{

digitalWrite(in1, HIGH);

digitalWrite(in2, LOW);

digitalWrite(in3, LOW);

digitalWrite(in4, HIGH);


}

void turn_right ()

{

digitalWrite(in1, LOW);

digitalWrite(in2, HIGH);

digitalWrite(in3, HIGH);

digitalWrite(in4, LOW);


}

void reverse ()

{

digitalWrite(in1, LOW);

digitalWrite(in2, HIGH);

digitalWrite(in3, LOW);

digitalWrite(in4, HIGH);


}

void stop()
```

```
{

digitalWrite(in1, LOW);

digitalWrite(in2, LOW);

digitalWrite(in3, LOW);

digitalWrite(in4, LOW);


}

long FrontSensor ()

{

long dur;

digitalWrite(trigPin1, LOW);

delayMicroseconds(2); // delays are required for a succesful sensor operation.

digitalWrite(trigPin1, HIGH);

delayMicroseconds(10); //this delay is required as well!

digitalWrite(trigPin1, LOW);

dur = pulseIn(echoPin1, HIGH);

return (dur/58);// convert the distance to centimeters.

}


long RightSensor ()

{

long dur;

digitalWrite(trigPin2, LOW);

delayMicroseconds(2); // delays are required for a succesful sensor operation.

digitalWrite(trigPin2, HIGH);

delayMicroseconds(10); //this delay is required as well!
digitalWrite(trigPin2, LOW);

dur = pulseIn(echoPin2, HIGH);

return (dur/58);// convert the distance to centimeters.

}

long LeftSensor ()

{

long dur;

digitalWrite(trigPin3, LOW);

delayMicroseconds(2); // delays are required for a succesful sensor operation.

digitalWrite(trigPin3, HIGH);

delayMicroseconds(10); //this delay is required as well!

digitalWrite(trigPin3, LOW);

dur = pulseIn(echoPin3, HIGH);

return (dur/58);// convert the distance to centimeters.

}
```