
welcome
alamy
Image ID: DHWRRE
www.alamy.com

Maze Solving Robot using Arduino Uno

- Project represented by
- Name : Bishnu kumar
- Roll no : 2401730230
- Name : Amandeep Pradhan
- Roll no : 2401730233

Maze Solving Robot using Arduino Uno

Overview, Components, Uses, and Future Impact

Introduction

- Maze solving robots are autonomous machines designed to find the optimal path through a maze using sensors and algorithms. They are widely used in robotics education, competitions, and research.

Purpose to Build

- - Learn embedded systems and sensor integration
- - Understand pathfinding and obstacle avoidance algorithms
- - Enhance problem-solving and programming skills
- - Serve as a base for more advanced robotic applications

Key Points

- - Uses sensors to detect walls and navigate paths
- - Controlled by Arduino Uno for decision making
- - Efficient algorithm implementation (like Left/Right-hand rule)
- - Real-time path correction based on sensor input

Components Required

- - Arduino Uno
- - 3 Ultrasonic Sensors
- - L298N Motor Driver Module
- - DC Gear Motors (4)
- - Robot Chassis
- - Battery Pack (for power supply)
- - Jumper Wires and Breadboard
- - Wheels and Casters

Uses of Maze Solving Robot

- - Educational tool for robotics and coding
- - Pathfinding demonstrations in robotics competitions
- - Rescue and exploration missions in unknown terrains
- - Development platform for autonomous vehicle systems

Impact on Future Environment

- - Encourages green, automated transport systems
- - Enhances robotics research and smart navigation
- - Reduces human risk in hazardous area exploration
- - Promotes innovation in AI-powered mobility solutions

Programming language use

```
// Motor pins
#define IN1 8
#define IN2 9
#define IN3 10
#define IN4 11

// Ultrasonic sensor pins
#define trigFront 7
#define echoFront 6
#define trigLeft 5
#define echoLeft 4
#define trigRight 3
#define echoRight 2
```

```
long getDistance(int
trigPin, int echoPin) {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin,
HIGH);
   delayMicroseconds(10);
   digitalWrite(trigPin, LOW);
   long duration =
 pulseIn(echoPin, HIGH,
20000);
 long distance = duration *
 0.034 / 2;  return (distance == 0 ||
 distance > 400) ? 400 :
distance;}
```

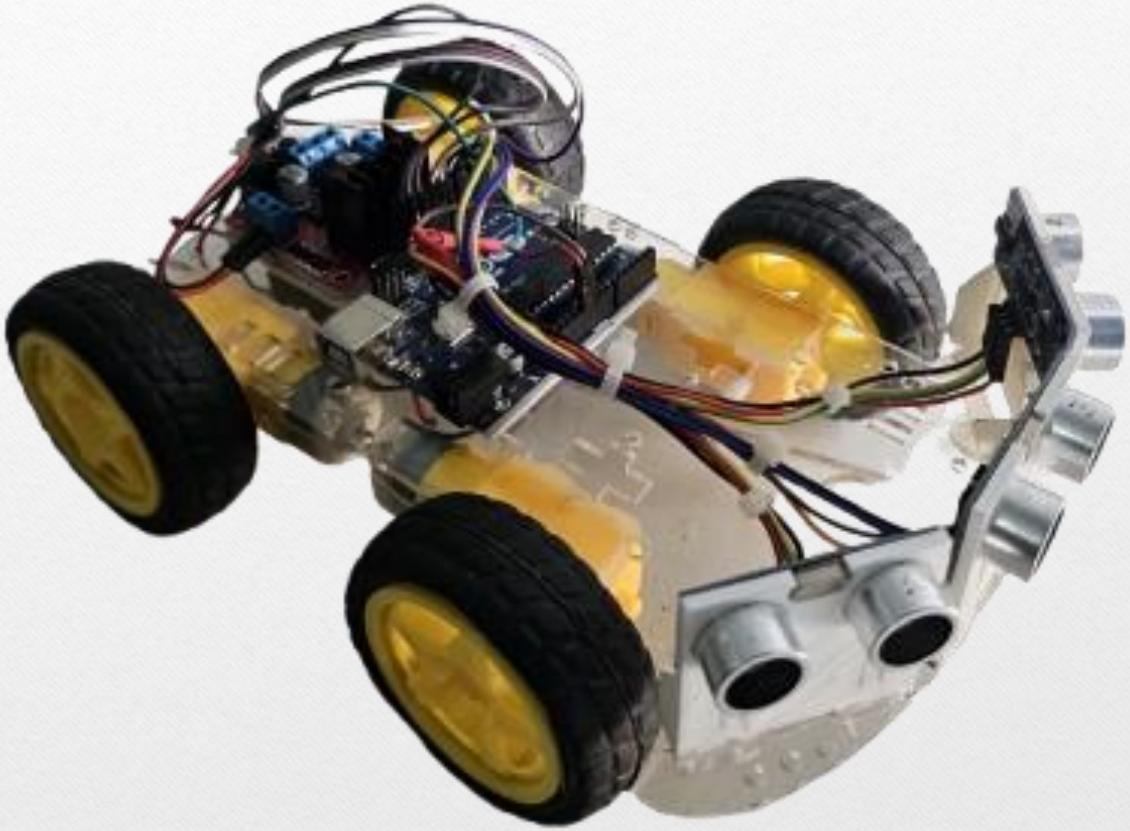
```
void moveForward() {
 digitalWrite(IN1, HIGH);
 digitalWrite(IN2, LOW);
 digitalWrite(IN3, HIGH);
 digitalWrite(IN4, LOW);
}
void turnLeft() {
digitalWrite(IN1, LOW);
digitalWrite(IN2, HIGH);
 digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);
}
void turnRight()
{    digitalWrite(IN1, HIGH);  digitalWrite(IN2, LOW);
      digitalWrite(IN3, LOW);  digitalWrite(IN4, HIGH);}
void stopMotors() {
     digitalWrite(IN1, LOW);  digitalWrite(IN2, LOW);
    digitalWrite(IN3, LOW);  digitalWrite(IN4, LOW);
}
```

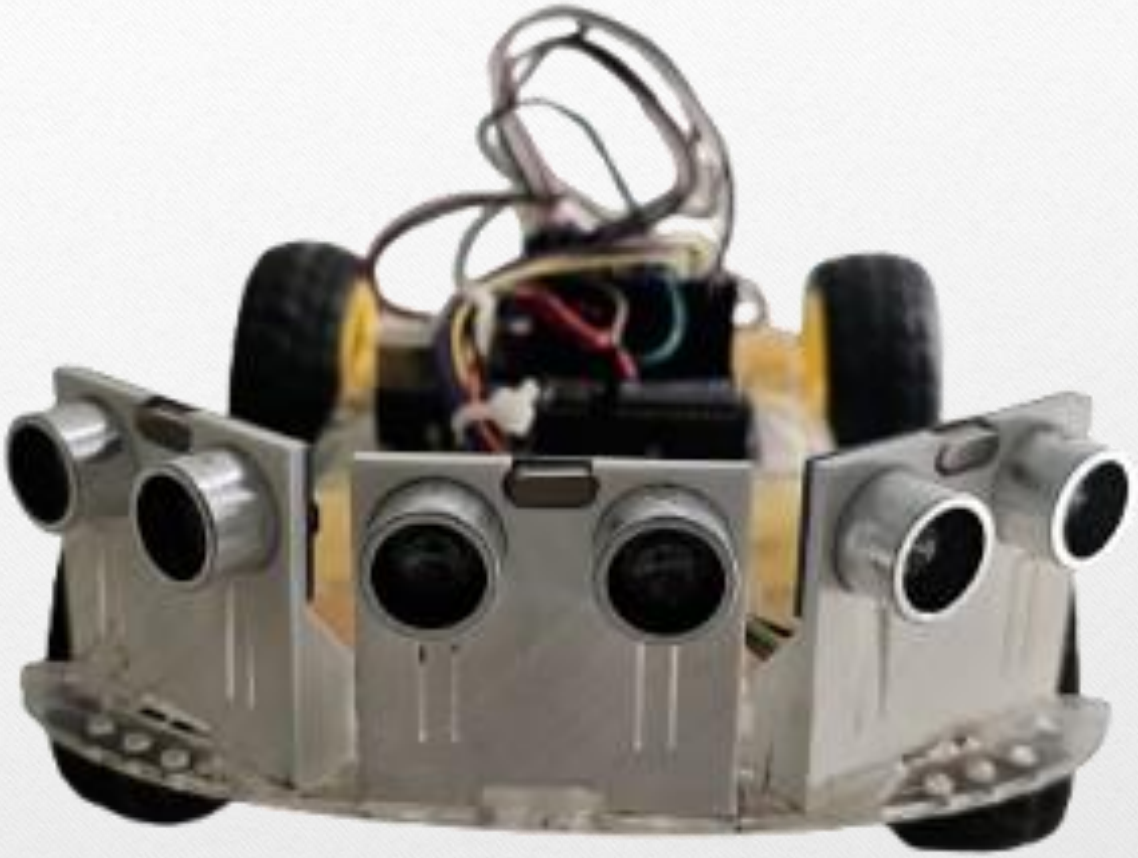
```
void stopMotors() {
    digitalWrite(IN1, LOW);  digitalWrite(IN2, LOW);
    digitalWrite(IN3, LOW);  digitalWrite(IN4, LOW);
}
void setup() {
    pinMode(IN1, OUTPUT);
    pinMode(IN2, OUTPUT);
    pinMode(IN3, OUTPUT);
    pinMode(IN4, OUTPUT);

pinmode(trigFront, OUTPUT);
 pinMode(echoFront, INPUT);
 pinMode(trigLeft, OUTPUT);
 pinMode(echoLeft, INPUT);
 pinMode(trigRight, OUTPUT);
 pinMode(echoRight, INPUT);
      Serial.begin(9600);
}
```

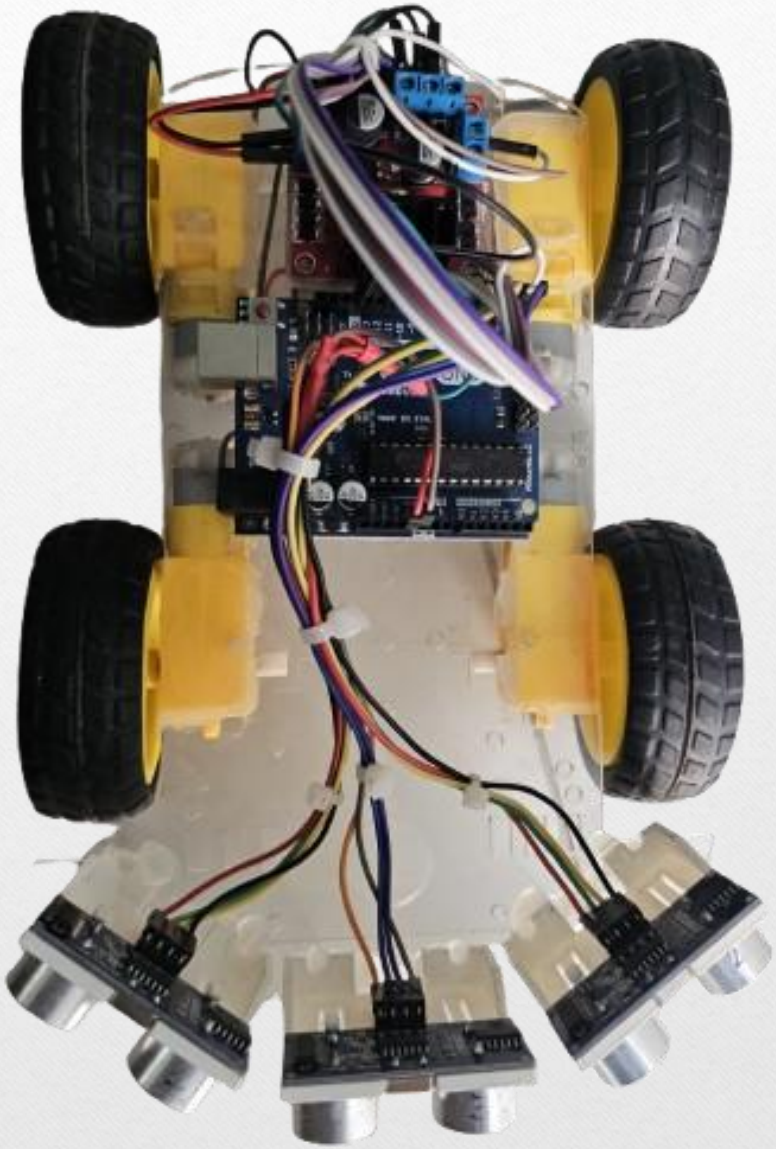
```
void loop() {
  long front = getDistance(trigFront, echoFront);
  long left = getDistance(trigLeft, echoLeft);
  long right = getDistance(trigRight, echoRight);
  Serial.print("F: "); Serial.print(front);
  Serial.print(" L: "); Serial.print(left);
  Serial.print(" R: "); Serial.println(right);
 if (front > 20) {    moveForward();  }
 else if (left > right) {    turnLeft();
  delay(400);  } else {    turnRight();
  delay(400);  }  delay(100);}
```

```
long getDistance(int
 trigPin, int echoPin) {
  digitalWrite(trigPin, LOW);
delayMicroseconds(2);
  digitalWrite(trigPin,
 HIGH);  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
```

Project-Image








THANK
YOU