

**MAZE LEARNING ROBOT**

SUBMITTED BY:

1. Parithosh Poojary (17BEC7050)
2. Deepsagar Mandal (17BCN7014)
3. Jatin Karthik (17BCE7106)
4. Piyush Yadav (17BME7043)
5. Kshitij Nair (17BCE7118)

**Vellore Institute of Technology, Amaravathi Campus**

Academic Session: 2018-19

Under the guidance of:  
Prof. Kalyan Chakravarthi

TABLE OF CONTENT(page no)

1. *Introduction and Background 2  
   1.1 Introduction 2  
   1.2 Advantages of the Project 2  
   1.3 Motivation 3*
2. *Problem Definition 4*
3. *Objectives 5  
   3.1 Model Objectives 5  
   3.2 Real Life Practical Objectives 5*
4. *Methodology 6  
   2.1 Components Required 6   
   2.2 Components Descriptions 6  
   2.3 Connections 8  
   2.4 Block Diagram 9*
5. *Result 10*
6. *Conclusion and Future Scope 10*
7. *References 11*
8. *Appendix (Coding) 12*
9. ***INTRODUCTION AND BACKGROUND***  
   **1.1 Introduction:**

In this project we have devised a robot which will move from any position in a maze to any target point. Our robot acts according to a code written purely in Arduino, which directs it to explore the maze containing random turns. The robot is autonomous and follows a derivative of wall following algorithm.

**1.2 Advantages of the project:**

Robots can be autonomous or semi-autonomous and range from humanoids such as Honda's Advanced Step in Innovative Mobility (ASIMO) and TOSY's TOSY Ping Pong Playing Robot (TOPIO) to industrial robots, medical operating robots, patient assist robots, dog therapy robots, collectively programmed swarm robots, UAV drones such as General Atomics MQ-1 Predator, and even microscopic nano-robots. By mimicking a lifelike appearance or automating movements, a robot may convey a sense of intelligence or thought of its own. Autonomous Things are expected to proliferate in the coming decade, with home robotics and the autonomous car as some of the main drivers. Ours is an autonomous robot.

**1.3 Motivation**

After the Thai Cave incident where a few kids were involved in an unfortunate birthday party. They were trapped in a cave for several days. The rescue teams could not figure out the way to get to the kids. Pondering upon this scenario we wondered if we could make a maze solving robot. We looked up previously researched papers on the topic and that was our inspiration.

1. ***PROBLEM DEFINITION***

**Background of the problem**

In this project, our core problem is to solve a given maze to find the shortest path to the end, i.e., out of the maze from the point of start.

**Possible Solution**

This problem can be solved using multiple algorithms such as:

1. Wall follower algorithm
2. Pledge algorithm
3. Dead-end Filling

And many more.

For our project, we are using the left-hand algorithm to traverse the maze in the first pass and make the robot automatically move in the shortest distance in the second pass.

**Benefits of the solution**

This solution ensures that the final path found is the shortest path.

1. ***OBJECTIVE***
   1. **Model Objectives**

The model should traverse the given maze, appropriately implementing the left-hand algorithm and learn the true shortest path when it reaches the end of the maze.

During the second run of the robot, it should travel only on the calculated shortest path form the Start to the End.

**3.2 Real Life Practical Objectives**

In a real life scenario, this algorithm can be set up in catastrophy-hit places (landslides, flooding, etc.) to find a way out of the affected location. Since it is nearly impossible for living beings to find a way out themselves, these robots can easily overcome that problem as they can be modified to work on any terrain or weather conditions, yet using the same algorithm to solve the problem.

1. ***METHODOLOGY***
   1. **Components Required**
2. Arduino UNO
3. DC Geared Motors, 60rpm
4. L298N Motor Driver
5. Infrared Sensor 2x
6. Ultrasonic Senor
7. 12V 3A Battery
8. Breadboard and jumper wires
9. 4 Wheel Chassis
   1. **Components Descriptions**
10. **Arduino UNO**

The Arduino UNO is an open-source microcontroller board based on the Microchip ATmega328P microcontroller and developed by Arduino.cc. The board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits. The board has 14 Digital pins, 6 Analog pins, and programmable with the Arduino IDE (Integrated Development Environment) via a type B USB cable. It can be powered by a USB cable or by an external 9 volt battery, though it accepts voltages between 7 and 20 volts.

1. **L298N Motor Driver**

The L298N H-bridge module can be used with motors that have a voltage of between 5 and 35V DC. With the module used in this tutorial, there is also an onboard 5V regulator, so if your supply voltage is up to 12V you can also source 5V from the board.

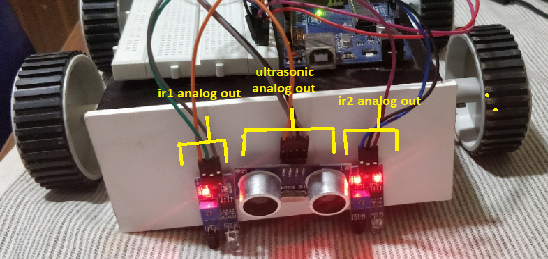
1. **Infrared sensor**

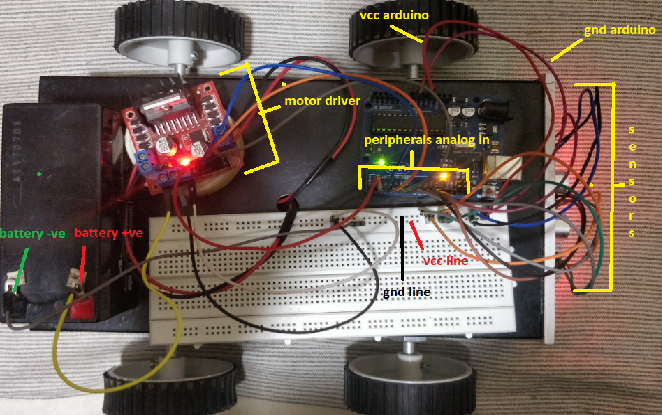
An infrared sensor is an electronic instrument that is used to sense certain characteristics of its surroundings. It does this by either emitting or detecting infrared radiation. Infrared sensors are also capable of measuring the heat being emitted by an object and detecting motion.

1. **Ultrasonic sensor**

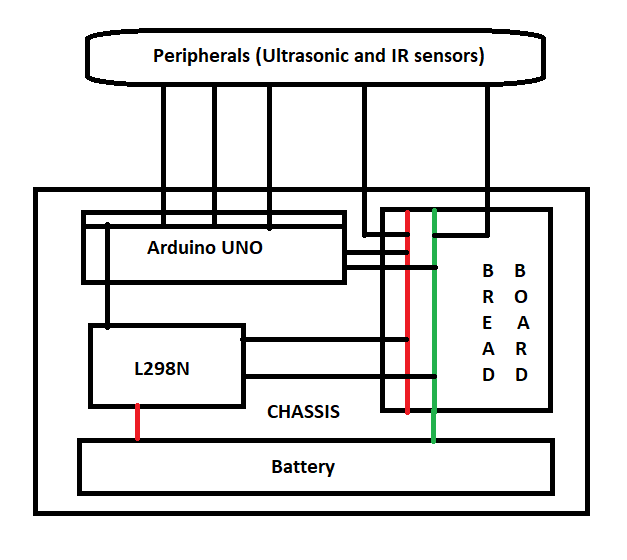
HC-SR04 Ultrasonic (US) senso**r** is a 4 pin module, whose pin names are Vcc, Trigger, Echo and Ground respectively. This sensor is a very popular sensor used in many applications where measuring distance or sensing objects are required. The module has two eyes like projects in the front which forms the Ultrasonic transmitter and Receiver.

**4.3 Connections**

**

****

**4.4 Block Diagram**

**

1. ***RESULT***

Any given perfect maze (without any loops) can be easily solved and traversed back in path using this algorithm.

1. ***CONCLUSION***

Maze solving algorithm via left hand approach works for all but close looped mazes.

**Future Scope**

1. The algorithm can be improved to include more ‘n’ junction-road variables and be able to solve a maze with known loops.
2. The robot can be modified according to the terrain that it has to be used on.
3. *REFERENCES:*

* https://www.arduino.cc/en/Guide/Introduction
* http://www.telegraph.co.uk/news/uknews/road-and-rail-transport/11403807/How-a-driverless-car-will-benefit-you.html
* https://learn.adafruit.com/adafruit-ultimate-gps-logger-shield/overview
* http://www.dummies.com/computers/arduino/arduino-projects-for-dummies-cheat-sheet/
* https://www.youtube.com/watch?v=BBwEF6WBUQs
* https://www.youtube.com/watch?v=HMksrzmAj88https://www.youtube.com/watch?v=HMksrzmAj88
* http://mertarduinotutorial.blogspot.in/2016/12/arduino-project-tutorial-11-obstacle.html
* Arduino: The Ultimate Beginner’s Guide to Learn by Daniel Jones

1. Chapter three: Data Types Found in Arduino
2. Chapter four: Local and Global Variables
3. Chapter six: Arduino Control Statements
4. Chapter seven: Arduino Loops
5. Chapter twelve: Arduino Arrays
6. Chapter thirteen: Input and Output Functions Found in Arduino

* Arduino® Projects for Dummies® Published by John Wiley & Sons, Ltd.
  + - 1. Part IV: Advanced Arduino Projects

Chapter 14: Building a Remote-Controlled Car

* + - 1. Part I: Getting Started with Arduino Projects

Chapter 1: Exploring the World of Arduino

Chapter 2: Setting Up Your Workspace and Tools

1. *APPENDIX (CODE)*

//dfws

const int irpin1 = 8;

const int irpin2 = 9;

const int trigPin = 10;

const int echoPin = 11;

uint32\_t period = 0.04 \* 80000L;

int irRead1 = HIGH;

int irRead2 = HIGH;

const int mtr\_R1 = 2;

const int mtr\_R2 = 3;

const int mtr\_L1 = 4;

const int mtr\_L2 = 5;

String initial = String("");

String solved = String("");

//dfws\_end

boolean black\_flag = true;

boolean end\_flag = false;

boolean sbflag = false;

int nblack = -1;

void setup() {

pinMode(irpin1, INPUT);

pinMode(irpin2, INPUT);

pinMode(mtr\_R1, OUTPUT);

pinMode(mtr\_R2, OUTPUT);

pinMode(mtr\_L1, OUTPUT);

pinMode(mtr\_L2, OUTPUT);

pinMode(trigPin, OUTPUT);

pinMode(echoPin, INPUT);

Serial.begin(9600);

}

//dfws

void right() {

digitalWrite(mtr\_L1, LOW);

digitalWrite(mtr\_L2, HIGH);

digitalWrite(mtr\_R1, LOW);

digitalWrite(mtr\_R2, LOW);

}

void left() {

digitalWrite(mtr\_R1, HIGH);

digitalWrite(mtr\_R2, LOW);

digitalWrite(mtr\_L1, LOW);

digitalWrite(mtr\_L2, LOW);

}

void forward() {

digitalWrite(mtr\_R1, HIGH);

digitalWrite(mtr\_R2, LOW);

digitalWrite(mtr\_L1, LOW);

digitalWrite(mtr\_L2, HIGH);

}

void reverse() {

digitalWrite(mtr\_R1, HIGH);

digitalWrite(mtr\_R2, LOW);

digitalWrite(mtr\_L1, HIGH);

digitalWrite(mtr\_L2, LOW);

}

void halt() {

digitalWrite(mtr\_R1, LOW);

digitalWrite(mtr\_R2, LOW);

digitalWrite(mtr\_L1, LOW);

digitalWrite(mtr\_L2, LOW);

}

//dfws\_end

void loop() {

Serial.println(initial);

Serial.println(solved);

//ultrasonic segment

long duration, distance;

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

duration = pulseIn(echoPin, HIGH);

distance = (duration/2) / 29.1;

//ultrasonic end

irRead1 = digitalRead(irpin1);

irRead2 = digitalRead(irpin2);

//solved segment begin

if(end\_flag) {

//Algorithm begin

String temp = String("");

int right\_lnum=0;

int left\_lnum=0;

temp = initial.substring(0,initial.length()-1);

for(int i = temp.length()-1; i>=0; i--) { //LLBLBLL

if(right\_lnum=0) {

if(temp.charAt(i)=='B') {

right\_lnum = temp.length()-1-i;

}

}

if(left\_lnum!=right\_lnum) {

i--;

if(temp.charAt(i)=='L') {

left\_lnum+=1;

}

if(temp.charAt(i)=='B') {

left\_lnum=0;

}

}

else {

solved.concat("R");

right\_lnum=0;

}

}

//Algorithm end

if (irRead1==HIGH && irRead2==HIGH) {

if(distance<15) {

halt();

delay(2000);

for(int i = 0; i<4; i++) {

for (uint32\_t tStart = millis(); (millis()-tStart)<period; ) {

reverse();

}

}

while(true) {

halt();

}

}

if(solved.substring(nblack,nblack+1)=="R"){

right();

}

else {

left();

}

if(sbflag) {

nblack+=1;

sbflag = false;

}

}

if (irRead1==LOW && irRead2==HIGH) {

left();

}

if (irRead1==HIGH && irRead2==LOW) {

right();

}

if (irRead1==LOW && irRead2==LOW) {

forward();

sbflag = true;

}

}

//solved segment end

//unsolved segment begin

if(!end\_flag) {

if (irRead1==LOW && irRead2==LOW) {

black\_flag = true;

if(distance>=15) {

forward();

}

else {

initial.concat("B");

for (uint32\_t tStart = millis(); (millis()-tStart)<period; ) {

reverse();

}

}

}

if (irRead1==LOW && irRead2==HIGH) {

left();

}

if (irRead1==HIGH && irRead2==LOW) {

right();

}

if (irRead1==HIGH && irRead2==HIGH) {

if(distance<15) {

end\_flag = true;

halt();

delay(10000);

}

else {

if(black\_flag) {

initial.concat("L");

black\_flag = false;

}

left();

}

}

}

//unsolved segment end

}