MAZE RUNNER

By

VUPPALA VAMSI MOHAN – 17BEC1087
SYED NAJAF HAIDER NAQVI – 17BEC1028
RITESH SINGH – 17BEC1065
ABHINAV SHIVNATH DUTTA – 17BEC1229

A project report submitted to
Dr.ASK

SCHOOL OF MECHANICAL AND BUILDING SCIENCE

for CAL in partial fulfilment of the requirements for the course

EXC1055 ROBOTICS CLUB
in
B.Tech. (Electronics and Communication Engineering)

VIT
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

Vandalur – Kelambakkam RoadChennai – 600127

FALL SEMESTER - 2018-19

October 2018

TABLE OF CONTENTS

BONAFIDE CERTIFICATE

Certified that this project report entitled "**MAZE RUNNER**" is a bonafide work of VUPPALA VAMSI MOHAN (17BEC1087), SYED NAJAF HAIDER NAQVI (17BEC1028), RITESH SINGH (17BLC1065) ABHINAV SHIVNATH DUTTA (17BEC1229) carried out the "J"-Project work under my supervision and guidance for ECE 2002 ANALOG ELECTRONIC CIRCUITS.

Dr.ASK

SCHOOL OF MECHANICAL AND BUILDING SCIENCE

VIT University, Chennai

Chennai – 600127.

ABSTRACT

Maze runner is a robot which solves mazes when placed in one, in a limited amount of time.

It comes to the centre of the maze in which it is placed in. Maze runner competitions are held world-wide, micro mouse, wall hugging robots etc. are famously used to achieve a maze-runner.

Our bot copies the nature in solving the maze, it uses the technique which bats use to move around in caves.

Our bot emits ultrasonic waves around and traverses the maze based on its reflections. It remembers the route in which it went in so that it can trace its steps back when necessary.

ACKNOWLEDGEMENT

We wish to express our sincere thanks and deep sense of gratitude to our project guide, **Dr. ASK,** School of Mechanical and Building Sciences, for his consistent encouragement and valuable guidance offered to us in a pleasant manner throughout the course of the project work.

We are extremely grateful to **Mr Siva Shankar,** Team leader, Maze runner, for guiding us through the options and solutions for the maze runner.

We also take this opportunity to thank all the faculty of the School for their support and their wisdom imparted to us throughout the course.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.

OBJECTIVE OF THE PROJECT

Objective of Maze Runner VITCC is to create a fully autonomous robot which can fit in a box of 20 x 20 x 20 cm, with at max of 4 sensors, which solves the maze in a limited amount of time. The bot should not touch the walls at any point of time during the competition. The robot should not be a wall hugger. The power of the bot at any point should not exceed 12v DC, 1.5A. The bot should be able to take a perfect 90 degree turn and should be able to move straight without hitting the wall. It should be able to trace back its steps if needed in a situation where it ends up in a dead-end. The bot should be able to recognize that it has reached its destination i.e. the centre of the maze.

INTRODUCTION

The bot consist of 4 sensors:
1) 3 x ultrasonic sensors
2) 1 x rotary encoder

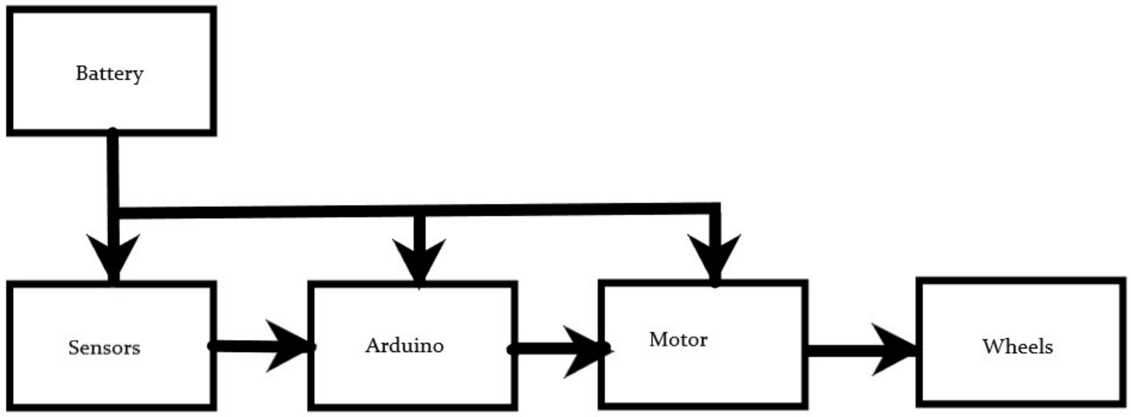
Ultrasonic sensors are placed at front of, to the right, and to the left of the bot. Ultrasonic sensors produce ultrasonic waves and tells us the distance of it from the walls. It helps the bot in locating if there is an obstacle in front, to the right and to the left of it, so as to make an appropriate turn at the right moment.

It also helps the bot in error correction (of its position with respect to the wall) by checking if the distance of it from the right wall and left wall are maintained equal so as to move in a straight line and not hit the wall.

Rotary encoder does the tracing of the route part, so that the bot does not get lost in the maze and keeps running into the same lane again and again.

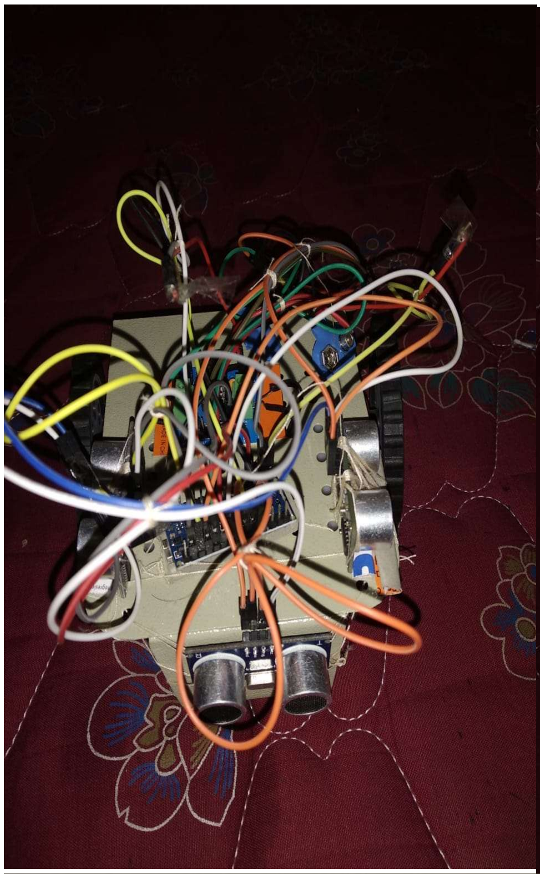
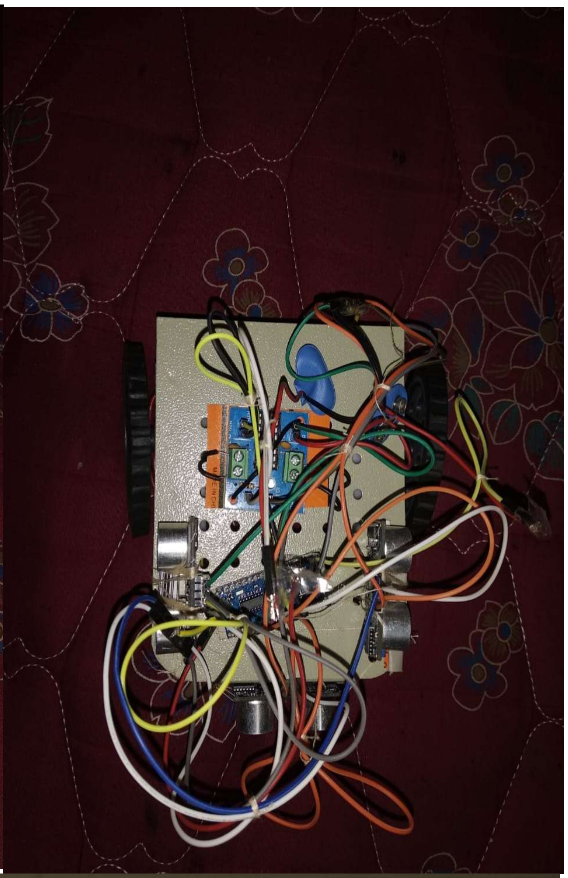
The motors are communicated with the Arduino via motor drivers.
The whole bot is powered by 2 9V batteries.

BLOCK DIAGRAM



The battery powers the microcontroller (Arduino), motors and all of the four sensors.
 Sensors sense the environment they are designed for and give their outputs to the arduino which then process the data and instructs the motor as programmed.
The motor turns the wheels which results in the movement of the bot in the required direction.

SNAPSHOT

HARDWARE ANALYSIS

Our Maze runner bot contain 4 sensors 3 ultrasonic and 1 rotary encoder.

Ultrasonic have 4 pins – Vcc, Gnd, Trig, Echo.

We have two 100 rpm dc motor and 2 9v battery.

We have used Arduino Nano.

We have used L298 to control dc motor.

When we power Arduino, Ultrasonic sensors start detecting walls based on that we decide where to move.

Four digital pins in motor driver control State of motor.

State of motor is set by motor driver.

Based on Its state we move forward, left, right or turn 180 degree.

Performing these Movement,bot move in maze.

THE CODE

```
///########################
/* Thing to notice
there will be four time duration used in movements.
time0 == time taken to move 1 unit forward
time1 == time taken to turn left or right.
time2 == time taken to turn 180.
time3 == time taken to turn slight left or slight right for error
correction.
___________________________________________________________
____

There will be two dist which will be used
unit_dist == dimension of 1 unit box
wall_dist == dist to be maintained from wall from both side
*/
int const time0=1800;  // == 3s
int const time1=1800;  // == 3s
int const time2=2500;  // == 5s
int const time3=500;  // == 1s
int constunit_dist=20;    //15cm
int constwall_dist=5;   // 8cm
// four digital pins for motor driver
int md1=2;
int md2=4;
int md3=7;
int md4=8;
//variables for Ultrasonic in forward direction
int trigf=3;
int echof=5;
long tmf;
int distf;
//variables for Ultrasonic in left direction
int trigl=6;
int echol=9;
long tml;
int distl;
//variables for Ultrasonic in right direction
int trigr=10;
int echor=11;
long tmr;
int distr;
int counter=0; // for setting preference
void setup()
{
  // put your setup code here, to run once:
pinMode(md1,OUTPUT);
pinMode(md2,OUTPUT);
pinMode(md3,OUTPUT);
pinMode(md4,OUTPUT);
pinMode(echof,INPUT);
pinMode(trigf,OUTPUT);
pinMode(echol,INPUT);
pinMode(trigl,OUTPUT);
pinMode(echor,INPUT);
pinMode(trigr,OUTPUT);
  //pinMode(led,OUTPUT);
Serial.begin(9600);
}
int calcf()  // function to calculate and return forward distance
{
digitalWrite(trigf,LOW);
delayMicroseconds(2);
digitalWrite(trigf,HIGH);
delayMicroseconds(10);
digitalWrite(trigf,LOW);
tmf=pulseIn(echof,HIGH);
distf=tmf*0.034/2;
  return distf;
}
int calcl()  // function to calculate and return left distance
{
digitalWrite(trigl,LOW);
delayMicroseconds(2);
digitalWrite(trigl,HIGH);
delayMicroseconds(10);
digitalWrite(trigl,LOW);
tml=pulseIn(echol,HIGH);
distl=tml*0.034/2;
  return distl;
}
int calcr()  // function to calculate and return right distance
{
digitalWrite(trigr,LOW);
```

```
delayMicroseconds(2);
digitalWrite(trigr,HIGH);
delayMicroseconds(10);
digitalWrite(trigr,LOW);
tmr=pulseIn(echor,HIGH);
distr=tmr*0.034/2;
  return distr;
}
void forward()  // function to move forward
{
digitalWrite(md1,HIGH);
digitalWrite(md2,LOW);
digitalWrite(md3,HIGH);
digitalWrite(md4,LOW);
  delay(time0);  // If we will give delay than only bot will get time to
perform movement
}
void left()     // function to move left
{
digitalWrite(md1,HIGH);
digitalWrite(md2,LOW);
digitalWrite(md3,LOW);
digitalWrite(md4,LOW);
  delay(time1);
}
void right()  // function to move right
{
digitalWrite(md1,LOW);
digitalWrite(md2,LOW);
digitalWrite(md3,HIGH);
digitalWrite(md4,LOW);
  delay(time1);
}
void turn()  // function to take 180 degree turn
{
digitalWrite(md1,LOW);
digitalWrite(md2,LOW);
digitalWrite(md3,HIGH);
digitalWrite(md4,LOW);
  delay(time2);   // If we will give delay than only bot will get time to
perform movement
}
int center(int f,intl,int r)  // path to be taken when it predicts that
box can be a centre
{
   //code for case When we enter centre block and
foward==left==unit_dist and right == wall dist
   if(((unit_dist-1)<f<(unit_dist+1)) && ((unit_dist-
1)<l<(unit_dist+1)) && ((wall_dist-1)<r<(wall_dist+1)))   // AS they
can't be exactly equal
   {
forward();
      if(((unit_dist-1)<l<(unit_dist+1)) && ((wall_dist-
1)<r<(wall_dist+1)))    // If this condition does not hold than else
will execute and we will come out of centre fn by returning 0 // If
condition hold it will move to check another below if condition
skipping else
      {
left();
forward();
```

```
     }
     else
       return 0;
     if(((unit_dist-1)<l<(unit_dist+1)) && ((wall_dist-
1)<r<(wall_dist+1)))
     {
      //write code to stop we have reached centre
digitalWrite(md1,LOW);
digitalWrite(md2,LOW);
digitalWrite(md3,LOW);
digitalWrite(md4,LOW);
delay(100000);   // Give longer delay for above values, because we
want to stop bot
     }
     else
       return 0;
  }
  //code for case When we enter centre block and
foward==right==unit_dist and left == wall dist
  if(((unit_dist-1)<f<(unit_dist+1)) && ((unit_dist-
1)<r<(unit_dist+1)) && ((wall_dist-1)<l<(wall_dist+1)))   // AS they
can't be exactly equal
  {
forward();
     if(((unit_dist-1)<r<(unit_dist+1)) && ((wall_dist-
1)<l<(wall_dist+1)))    // If just above condition does not hold than
else will execute and we will come out of centre by returning // If
condition hold it will move to check another below if condition
skipping else
     {
```

```
left();
forward();
     }
     else
        return 0;
     if(((unit_dist-1)<r<(unit_dist+1)) && ((wall_dist-
1)<l<(wall_dist+1)))
     {
       //write code to stop we have reached centre
digitalWrite(md1,LOW);
digitalWrite(md2,LOW);
digitalWrite(md3,LOW);
digitalWrite(md4,LOW);
delay(100000);   // Give longer delay for above values because we
want to stop bot
     }
     else
        return 0;
  }
}
void chk0(int f,intl,int r)  // It is for counter 0 prefence -- f,l,r   ---
>f,l,r will be calc. in void loop()
{
  if(f>wall_dist)
  {
forward();
  }
  else if(l>wall_dist)
  {
left();
```

```
forward();  // Because after taking turn we need to move 1 unit
forward (otherwise we will keep rotating on that place only)
  }
  else if(r>wall_dist)
  {
right();
forward();  // Because after taking turn we need to move 1 unit
forward (otherwise we will keep rotating on that place only)
  }
  else
  {
turn();
  }
  counter=counter+1;
}
void chk1(int l,intr,int f)  // It is for counter 1 prefence -- l,r,f
{
  if(l>wall_dist)
  {
left();
forward();   // Because after taking turn we need to move 1 unit
forward (otherwise we will keep rotating on that place only)
  }
  else if(r>wall_dist)
  {
right();
forward();    // Because after taking turn we need to move 1 unit
forward (otherwise we will keep rotating on that place only)
  }
  else if(f>wall_dist)
  {
forward();
  }
  else
  {
turn();
  }
  counter=counter+1;
}
void chk2(int r,intf,int l)  // It is for counter 2 prefence -- r,f,l
{
  if(r>wall_dist)
  {
right();
forward();     // Because after taking turn we need to move 1 unit
forward (otherwise we will keep rotating on that place only)
  }
  else if(f>wall_dist)
  {
forward();
  }
  else if(l>wall_dist)
  {
left();
forward();   // Because after taking turn we need to move 1 unit
forward (otherwise we will keep rotating on that place only)
  }
  else
  {
turn();
```

```
  }
  counter=counter+1;
}
void moverob(int f,intl,int r)   // function for movement of robot  //
it will chk main condition --> if it hold move forward other wise
according to counter value execute chk0, chk1 or chk2 function
respectively.
{
  /*if(f==l||f==r)   // If this condition hold than it will according to
path set in center function
  {
center(f,l,r);
  }*/
  if(f>wall_dist&&l<wall_dist&&r<wall_dist)
  {
forward();
  }
  else
  {
    if(counter%3==0)
    {
      chk0(f,l,r);
    }
    else if(counter%3==1)
    {
      chk1(l,r,f);
    }
    else if(counter%3==2)
    {
      chk2(r,f,l);
    }
  }
}
//Function for correcting path
void correction(int f,intl,int r)  //correction to path
{
  if(r>l)    //move slight RIGHT
  {
digitalWrite(md1,LOW);
digitalWrite(md2,LOW);
digitalWrite(md3,HIGH);
digitalWrite(md4,LOW);
    delay(time3);   // If we will give delay than only bot will get some
time to perform movement
  }
  else if(l>r)   //move slight LEFT
  {
digitalWrite(md1,HIGH);
digitalWrite(md2,LOW);
digitalWrite(md3,LOW);
digitalWrite(md4,LOW);
    delay(time3);   // If we will give delay than only bot will get some
time to perform movement
  }
}
void loop()
{
  // put your main code here, to run repeatedly:
distf=calcf();
distl=calcl();
```

```
distr=calcr();
moverob(distf,distl,distr);
  //correction(distf,distl,distr);  // correction in posn of bot will be
done after each movement
}
```

ComponentRequirement and Estimation

S. No	Component Name	Quantity	Cost (in Rs.)
1	Connecting wires	As required	-
2	Ultrasonic sensor	3	Rs. 100
3	Rotary encoder	1	Rs. 100
4	Arduino	1	Rs. 400
5	Motor driver	2	Rs. 100
6	Wheels	2	Rs. 150
7	9v battery	2	Rs. 40
8	Motor	2	Rs. 100

Overall cost of the Project: Rs.1500

CONCLUSION

- The maze runner bot was built and implemented.

- The bot is made to solve maze on its own.

- The sensor detects the walls and based on that algorithm run motor to solve maze

- The preliminary test results are promising.