Maze-Solving Robot using Infrared Sensors

Nawab Haider Ghani, Sazzad Hossain, Jahid Hasan Mamun, Tahsinur Rahman, Mazedur Rahman Riad

BRAC University, Department of Computer Science and Engineering, Mohakhali, Dhaka 1212, Bangladesh

{nawabhaider19, sazzadhossain1206, gregorian.zahid, tahsinurrahman5, riad.druvo } @gmail.com

Abstract. This project will be based on the development of a robot that will solve a maze by navigating through it and finding the exit of the maze. The robot will consist of a microcontroller, the Arduino Uno R3. It will also have motors, wheels, batteries, a motor driver, a mechanical chassis etc. Finally it will have three Infrared sensors (IR sensors) that would be used to guide the robot through the maze. The microcontroller (Arduino Uno) will contain the maze solving algorithm which would instruct the robot on how to move through the maze depending on the information received from the sensors [1]. The robot will traverse through the entire maze and at each intersection decide whether to go left, right, straight or make a 180º turn depending on the output of the IR sensors. It will keep traversing the maze until it reaches the end point, then it will stop.

Keywords: Maze solver, Infrared Sensors, Arduino Uno R3

1. Introduction

The functionality of a maze solving robot is to find its way out of a maze. The maze itself will be a simple one with black lines on a white background. The black lines would be black tape on a white paper. The robot will be programmed using IR sensors so that it travels over the black lines and stops if it comes across any white parts. The IR sensors will detect a specific light wavelength in the IR spectrum by using two LED's. The first LED will emit light at the wavelength that the sensor is looking for. When an object comes within the range of the sensor the light is returned back and the second LED receives it. That is how the IR sensors will be used to detect black lines on white background. The maze will have a start point and an end point; the objective will be for the robot to go from the start to end traversing through the entire maze and stopping when it finds the end point.

Now the maze itself will be very simple; it will contain straight paths, dead ends and intersections. However it would not contain any loops. The tapes will be overlapped to avoid white gaps and will have clean corners.

The robot will use three closely spaced IR sensors that will look down on the track and then be read by the program. The information from the IR sensors will be sent to the IR controller and then to the microcontroller (Arduino Uno R3) which will process it and send instructions to the motors via the motor driver. Depending on these instructions the motor will decide which way to move the robot. The power supply will come from a 9V battery and a 7.4V 2C Li-Po battery. All the components will be attached to a mechanical chassis.

The rest of the paper is organized as follows. Section 2 describes the proposed model with a block diagram; Section 3 demonstrates hardware implementation of proposed model; Section 4 contains circuit diagram; Section 5 comprises of flow diagram of the proposed model and the algorithm. Section 6 concludes the paper.

2. Proposed Model

Fig.1 shows a complete block diagram of the circuit of proposed model. Power supplied to the microcontroller (Arduino Uno R3) will be from the 9V battery. The power supply of the motor driver will be from a 7.4V 2C Li-Po battery. There will three IR sensors attached to the maze solver. One sensor will be placed at the centre while the other two will be placed to the right and left. The sensors will give different signals depending on whether it is detecting white or black color. If a particular sensor detects black color then it will be activated. So if the left sensor is activated then the robot will go left. Likewise, if the right sensor is activated then it will go right. If the centre sensor is activated then it will go straight. The output of the IR sensors is sent to the IR modules which in turn send it to the arduino uno. From the arduino uno it goes to the motor driver. This maze solver robot consists of two gear motors; one on the left and one on the right. The motors will move either forward or backward or be turned off entirely depending on the signal it receives from the motor driver.

The mechanical chasis of the proposed mode basically contains two wheels to the two sides. To these two wheels, two motors are attached. A plastic board is bolted down with the wheels using screws. All the components are then are attached to the chassis using either tape or glue.

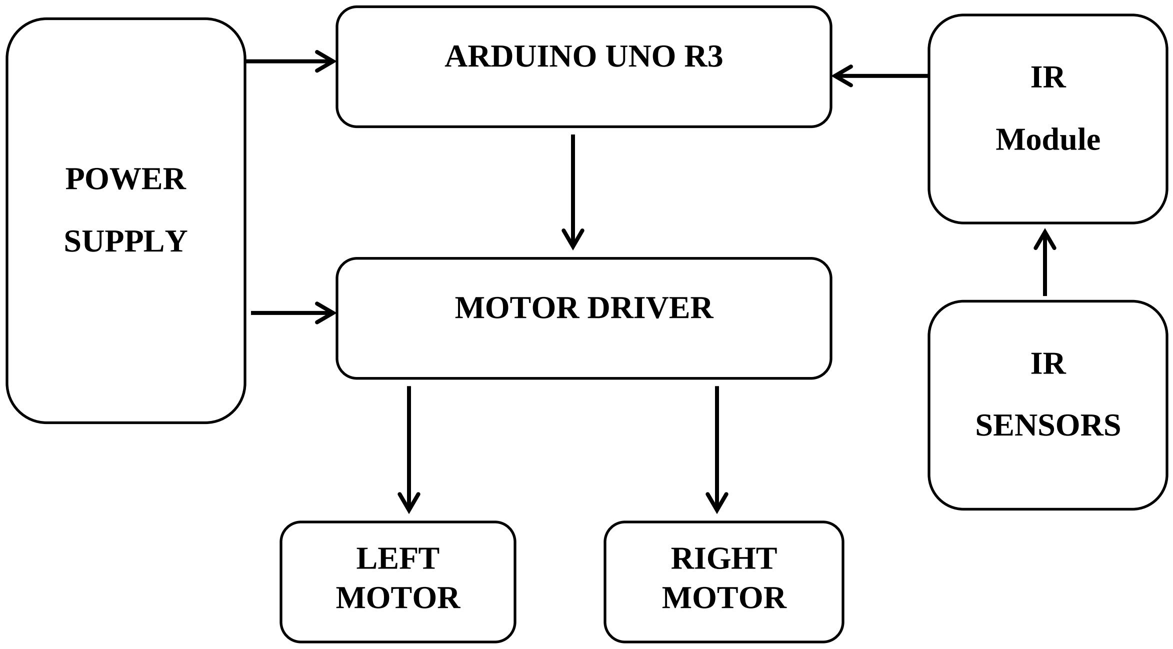
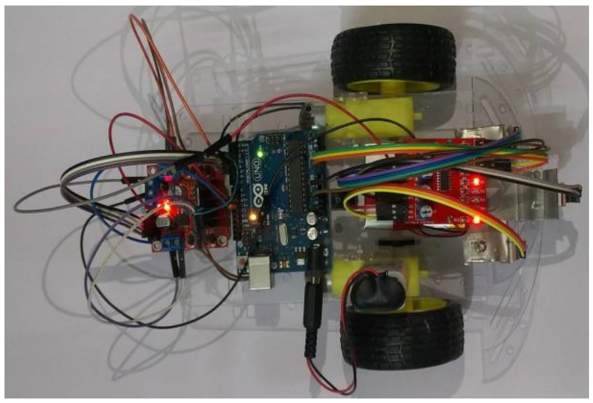


Fig. 1 Block diagram of proposed circuit



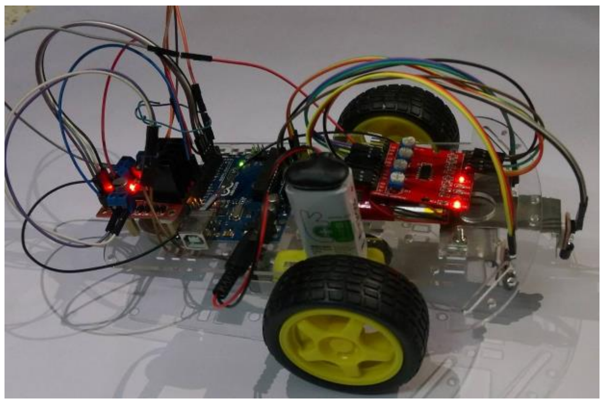


Fig. 2 Top view of Prototype

Fig. 3 Side View of Prototype

3. Hardware Implementation of Proposed Model

3.1 Arduino Uno R3

Arduino is the microcontroller board that will be used in this project. Both its hardware and software is open-source. Moreover, it is cheap, has lots of free libraries and easy to code in. There are different types of arduino boards are available; however for our purposes we would be using the Arduino Uno R3. It has an ATmega328P microcontroller and contains 14 digital pins which can be used as either input or output pins. 6 of these pins can be used as PWM outputs. It also contains 6 analog inputs, a 16MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. The DC current per I/O pin is 20mA. Clock speed is 16MHz. It has a flash memory of 32KB and SRAM of 2KB [2].

Fig. 4 shows the Arduino Uno

3.2 Motor Driver

Motor driver is needed because the vehicle is going to be bi-directional, so it will go both forwards and backwards. It can control 2 gear motors at the same time [3]. There are two enable inputs turning the device on/off. It has 15-lead Multiwatt and PowerS020 packages with bidirectional drive currents of up to 1A. Its operating voltage is 4.5V-36V and maximum power drive is 75W.

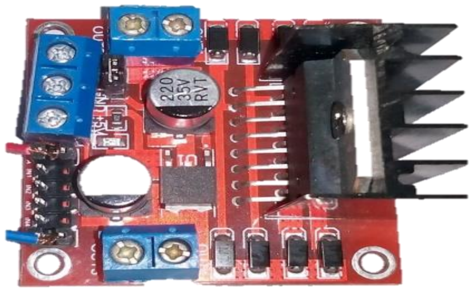


Fig.5 shows the motor driver

3.3 IR Sensor & Module

The IR sensors work by using reflected infrared light to detect the presence of a line or an edge when there is no reflection [3]. There is two LED's present in each sensor; one emits Infrared light and another receives it. In the figure we can see 4 separate IR sensors. The signal from each sensor is sent to the IR controller which in turn sends it to the microcontroller.

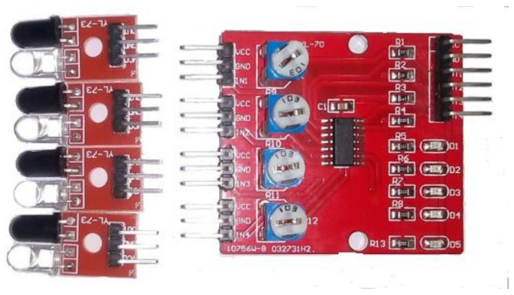


Fig.6 shows IR Sensors & Module

3.4 Two Gear Motors

A gear motor has a magnetic coil that produces a magnetic force when a current passes through it. This magnetic force turns the motor. The amount and direction of current decides the speed and direction of the motor [4]. 2.5 inch long, 0.85 inch wide and 0.7 inch thick. The wheel can be mounted on either side and the gear motor works between 4V to 7V (recommended 6 Volts).

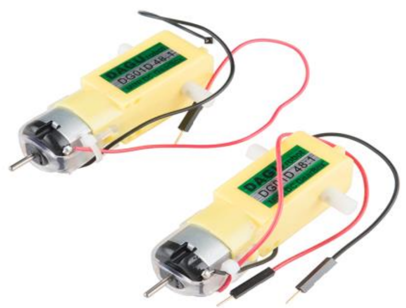


Fig.7 shows two gear motors

3.5 Other Components:

- **9V battery**
- **7.4V 2C Li-Po battery**
- **Jumper Wires**
- **Mechanical Chassis**
- **Breadboard**

4. Circuit Diagram

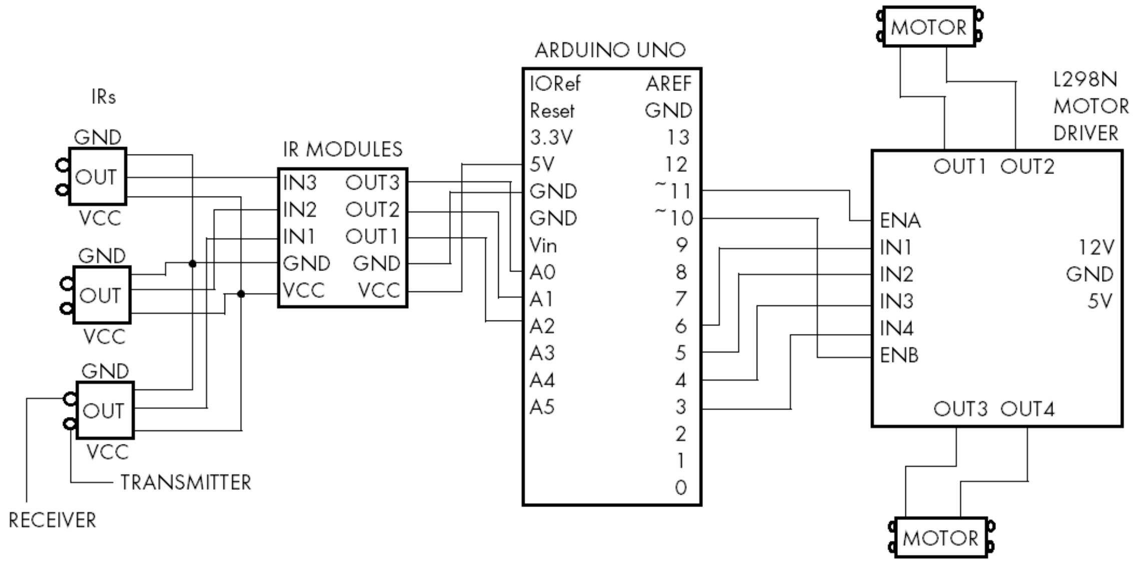


Fig. 8 shows circuit diagram of proposed model

Pin 10 and 11 of the arduino uno is connected to the motor driver and works as enable for the two motors. Pins 3, 4, 5 and 6 are the inputs of the L298N motor driver. Pins 6, 5, 4 and 3 are connected to IN1, IN2, IN3 and IN4 of the motor driver. OUT1 & OUT2 are connected to one motor while OUT3 and OUT4 are connected to another motor [5].

Pins A0, A1 and A2 of the arduino uno are connected to OUT3, OUT2 and OUT1 of IR Module (controller) respectively. GND of arduino is connected to GND of IR module and 5V of arduino is connected to Vcc of IR module. Each IR sensor has an output pin that is connected to IN3, IN2 and IN1 of the IR module. Also, each IR has a GND and Vcc pin that is connected to the GND and Vcc of the IR module. Finally, each IR sensor has a receiver and transmitter that is used to detect Infrared light.

5. Flow Diagram & Algorithm

By using the arduino IDE we would develop the code then compile and execute it. Then it is transferred it to the Arduino Uno R3.

The basic concept of the algorithm is that at any given point there are there possible routes to take; left, right or straight. Depending on the output of the IR sensors, the robot will decide which routes it can take of the three mentioned above. If there are a combination of possibilities to choose from then it would choose the one according to the algorithm. If it cannot take any of the routes, then it will make a 180º turn and go back the way it came from.

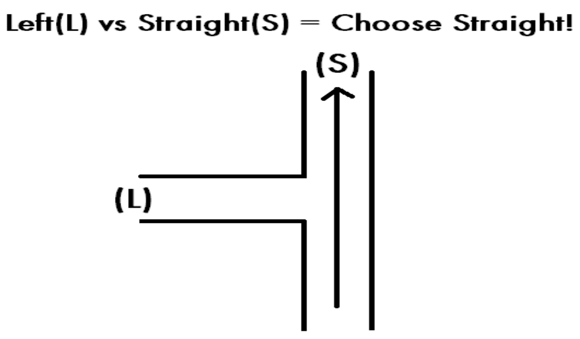


Fig.9 Left vs Straight

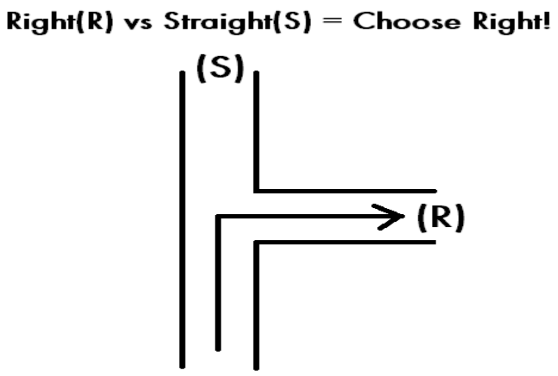


Fig.10 Right vs Straight

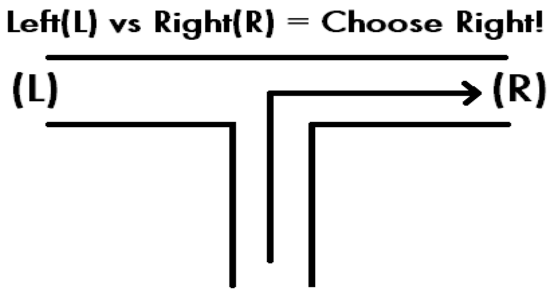


Fig.11 Left vs Right

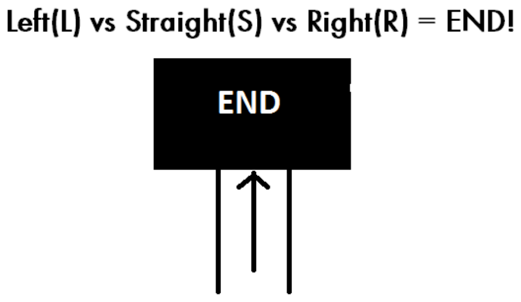


Fig.12 Right vs Left vs Straight

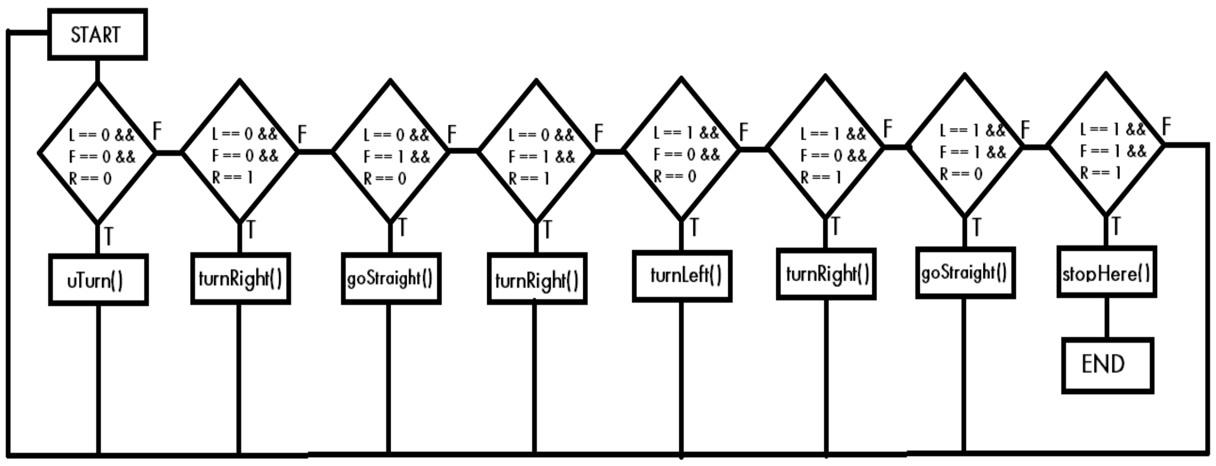

START
L == 0 && F == 0 && R == 0
uTurn()
L == 0 && F == 0 && R == 1
turnRight()
L == 0 && F == 1 && R == 0
goStraight()
L == 0 && F == 1 && R == 1
turnRight()
L == 1 && F == 0 && R == 0
turnLeft()
L == 1 && F == 0 && R == 1
turnRight()
L == 1 && F == 1 && R == 0
goStraight()
L == 1 && F == 1 && R == 1
stopHere()
END
T
F

Fig. 13 shows Flow Diagram of the algorithm

L represents the left sensor.

R represents the right sensor.

F represents the center sensor.

```
while(true){
        if( L==0 && F==0 && R==0)
            uTurn()
        if( L==0 && F==0 && R==1)
            turnRight()
        if( L==0 && F==1 && R==0)
            goStraight()
        if( L==0 && F==1 && R==1)
            turnRight()
        if( L==1 && F==0 && R==0)
            turnLeft()
        if( L==1 && F==0 && R==1)
            turnRight()
        if( L==1 && F==1 && R==0)
            goStraight()
        if( L==1 && F==1 && R==1)
            stopHere()
            break
}

goStraight(){
    leftMotor = FORWARD
    rightMotor = FORWARD
}
 turnRight(){
     leftMotor = FORWARD
     rightMotor = OFF
}
turnLeft(){
     leftMotor = OFF
     rightMotor = FORWARD
}
stopHere(){
     leftMotor = OFF
     rightMotor = OFF
}
uTurn(){
     leftMotor = FORWARD
     rightMotor = BACKWARD
}
```

When goStraight() method is called, the robot goes straight by turning both the left and right motor on. When turnRight() method is called, the robot moves 90º to the right. This is done by turning the right motor off and keeping the left motor on. When turnLeft() method is called, the robot moves 90º to the left by turning the left motor off and turning the right motor on.

If all three sensors have a value of 0, then uTurn() method is called. This method basically turns the robot 180º by setting the left motor forward and the right motor backward.

Finally, when all three sensors have a value of 1, stopThere() method is called which stops the robot by turning off both motors. In the arduino IDE, it enters an infinite loop where no operations are done but the arduino uno stays on.

A similar algorithm is used in [6], however we modified it to for our project. The algorithm used has short searching time and requires low space complexity so that’s the reason of using it. There are other methods and algorithms like dijkstra etc to solve even more complicated mazes

[7]. These other algorithms work by finding the shortest path from the start to end of a maze [8]. However this is the easiest method that we could find given our limited knowledge and expertise.

6. Conclusion

In doing this project we faced a lot of difficulties, combining the mechanical aspect with the electrical and computing aspects of the robot. Due to friction the wheels do not operate in the same way. Also it is very difficult to calibrate two analog motors together at the same voltage so their speed is different [9]. The speed of the motor also changes depending on the surface it is on. IR readings from the sensor can change due to different objects in the background. Moreover, there are a lot of connections between the Arduino Uno and other components. If one of these connections is loose, then the robot wouldn't work properly. We also had to keep in mind the discharging rate of the battery. As the battery loses its charge it might make the motors move at a slower speed. Furthermore, the maze itself had to be properly placed with the chart paper being completely flat. The black tapes had to be smoothly placed with clean corners and no white gaps in between [10].

Despite these problems we were able to properly implement the maze solving robot using IR sensors. The robot successfully traversed through the entire maze, going over the black lines and stopping or turning if it meets any white parts and stopping if it comes to the end.

References

[1] Siegwart, R., Nourbakhsh, I.R., & Scaramuzza, D. (2011). “Introduction to autonomous mobile robots”. Cambridge, MA: MIT Press.

[2] Kurniawan, A.(2015). “Arduino Uno: A Hands-On Guide for Beginners”, Retrieved from https://ebooklink.info/detail/B019A3B5YM/Arduino+Uno%3A+A+Hands-On+Guide+for+Beginner/

[3] Margolis, M. (2012). Make an Arduino-controlled robot. Sebastopol, CA: O’Reilly.

[4] Jokitulppo, M. (2015). “Arduino-Controlled Robot”, Retrieved from http://www.theseus.fi/bitstream/handle/10024/97523/thesis_matti_jokitulppo.pdf

[5]] Pandian, J. A., Karthick, R., Karthikeyan, B.(2011). “Maze Solving Robot Using Freeduino and LSRB Algorithm”, International Journal of Modern Engineering Research (IJMER).

[6] Vannoy, R. T.(2009). “Design a Line Solving Robot”. Retrieved from https://www.pololu.com/file/0J195/line-maze-algorithm.pdf

[7] Cai, J., Wan, X., Huo, M., & Wu, J. (2010). “An Algorithm of Micromouse Maze Solving”, Computer and Information Technology (CIT).

[8] Alsubaie, M. (2013). “Algorithms for Maze Solving Robot”, Manchester Metropolitan University

[9] Development environments for autonomous mobile robots: Kramer, J., Scheutz, M. (2006)

Retrieved from http://danwu.myweb.cs.uwindsor.ca/60-572-fall-2008/slides/Development%20environments%20for%20autonomous%20mobile%20robots-%20A%20Survey.pdf

[10] Sharma, K., & Munshi, C. (2015). “A Comprehensive and Comparative Study Of Maze-Solving Techniques by Implementing Graph Theory. IOSR Journal of Computer Engineering, 17(1). Retrieved from www.iosrjournals.org/iosr-jce/papers/Vol17-issue1/Version-4/E017142429.pdf