ARTIFICIALLY INTELLIGENT MAZE SOLVER ROBOT

Malkit Singh[1], Rajnish Kumar[2], Vaibhav Giradkar[3], Pallavi Bhole[4], Minu Kumari[5]

Department of Electronics and Telecommunication Engineering
Priyadarshini College of Engineering, Nagpur, India
malkitsingh1123@gmail.com[1], rajnish.singh8194@gmail.com[2], gvaibz@gmail.com[3], pallavibhole428@gmail.com[4], nituminu.89@gmail.com[5]

Abstract - Robotics is very important now days, especially due to its increasing practice in many industries. It was recently observed that there is a great difficulty being faced in the separation of the articles in the industries, like Textiles. Being a manual job it was also time consuming and boring. Therefore, for giving this problem a technological perspective, automation in such field was required. When a robot having its own sense of judgment to the path which it follows, would be introduced then a high efficiency in performance could be achieved along with increase in reliability and affordability of the manufacturers could be seen. The robot would be self-sufficient to take a note the paths through which it is moving, hence executing some complex maze-solving algorithms in its CPU core and taking its own decision on turnings and reaching its goal. It would certainly be a proof of a robot having its own "Brain-like" structural methodology having an access to the real-time inputs, making the prototype an Artificially Intelligent Robot.

Key Words: Line follower, Robot, Artificial Intelligence, Maze Solver, ATmega16, L293D

1. INTRODUCTION

Autonomous navigation is an important feature that allows a mobile robot to independently move from a point to another without an intervention from a human operator. Autonomous navigation within an unknown area requires the robot to explore, localize and map its surrounding. By solving a maze, the pertaining algorithms and behavior of the robot can be studied and improved upon. This paper describes an implementation of a maze-solving robot designed to solve a maze [1]. A maze is like a puzzle in the form of complex branching through which the robot has to travel. The robot basically have to perform two tasks are as follow-

i. To drive through the maze using right hand rule and reach at the center of maze (target).
ii. To optimize the shortest path for returning back by avoiding dead ends.

Maze-solving – although artificial in terms of the confine that the robot it subjected to – is a structured technique and controlled implementation of autonomous navigation which is sometimes preferable in studying specific aspect of the problem [1].

The maze-solving task is similar to the ones in the Micromouse [2] competition where robots compete on solving a maze in the least time possible and using the most efficient way. A robot must navigate from a corner of a maze to the center as quickly as possible. It knows where the starting location is and where the target location is, but it does not have any information about the obstacles between the two.

2. LITRATURE REVIEW

i. Micromouse by University of East London [5]

This version of Micromouse was developed by Michael Gims, Sonja Lenz and Dirk Becker from University of East London in year 1999. The design of the mobile robot is quite compact, but there is some improvement on the wiring. The algorithm applied is Wall Following Algorithm which is a non-graph theory algorithm. It does not move intelligent in the map and it could not solve maze with loop.

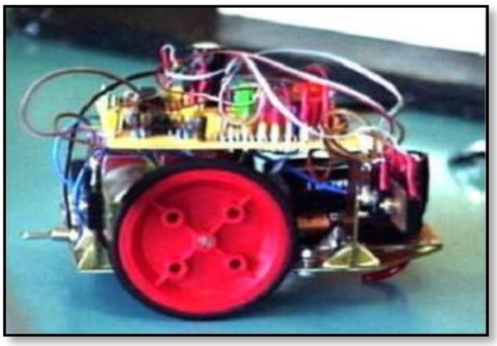

Fig - 1: Micromouse University of East London **[5]**

ii. Micromouse Maze Solving Robot [4]

This project is done by Chang Yuen Chung in year 2008/2009 academic session. The robot is designed in three layers so that the robot looks more compact and smaller size. The disadvantage of this robot is that it has too many connectors and it is very hard to troubleshoot if there is circuit faulty. The algorithm used in this project is Flood Fill Algorithm. It is one of the graph theory mazes solving algorithm. Chang claimed that this algorithm is able to find the shortest path but more memory is required for execution.

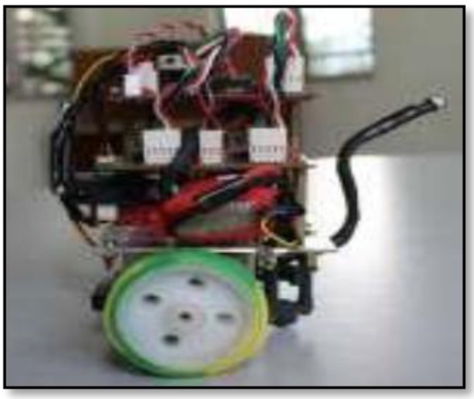

Fig - 2: Micromouse Maze Solving Robot **[4]**

iii. Maze Solving Robot [3]

This project is developed by Law Sei Cui in year 2010/2011 academic session. The objective is building a low cost mobile robot which can negotiate a maze. The author had used servo motor and IR sensor instead of costly stepper motor and analogue distance sensors in this project. The robot design is simple and stable which can move with faster speed. The algorithm applied is Dijkstra's Algorithm which is also one of the graph theory mazes solving algorithm. The author claims that although A* is advancement of Dijkstra's algorithm, Dijkstra's algorithm is easier to implement compare to A*.

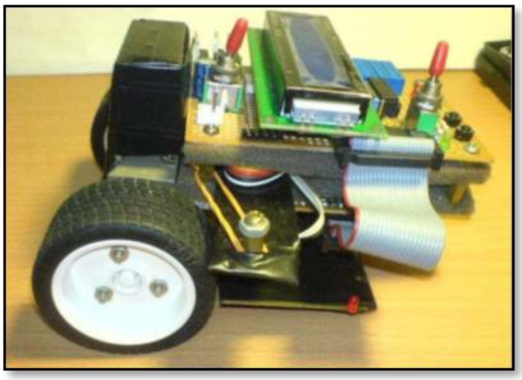

Fig - 3: Maze Solving Robot **[3]**

3. DESIGN METHODOLOGY

The design methodology of this prototype is based on mobile navigation of robot following the path provided in the arenna. It consists of designing six basic circuits as shown in bellow figure. Signals are passes from one circuit to another and necessary signal conversions are carried out by each circuit and enabling proper line flow.

In sensor circuit, four Infra-Red emitter-receiver pairs are used which distinguishes white line from black surface, as the black absorbs more infra-red than white colour. These sensors are called as line sensors. The information from a line sensor is passed on to the comparator, which is mainly used for comparing the input voltage with a reference threshold voltage. The digitalized output is then passed on to transistor circuit, which helps for maintaining constant voltage throughout the circuit.

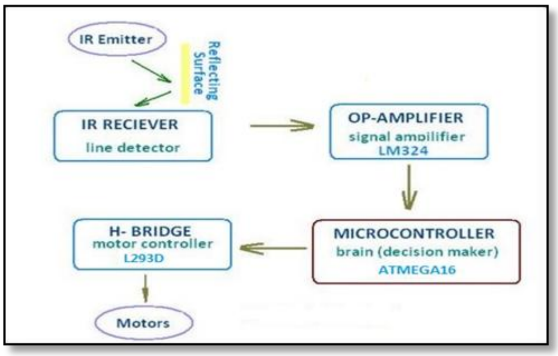

Fig 4: Design Process (Block representation)

The signal is then passed on to programmed microcontroller circuit which then takes decision based on received input. It processes the given input and sends corresponding output to motor driver circuit. Motor driver circuit is responsible for driving two independent motors. Each motor is operated independently by the motor driver, which is decided based on each motors input. Motor driver rotates the motor both in forward and reverse direction.

4. CIRCUIT DESIGN

The whole Prototype fits into a hollow cube of 20x20x20 cm^3. The entire circuit design is shown below in Fig. 4. The circuit consists of designing various individual circuits starting from sensor circuit to H-Bridge designing which has been discussed above.

The power supply of 12 V is gained from a 1200 mAH Li-Po battery. This battery is very light weight and is durable. The clock to the microcontroller has been supplied externally. Since ATMega16 microcontroller has been used so it is very fast in computation and has built-in ADC and DAC in it. It runs the DC motors of 300 RPM at a throttle of about 70 %. So the running RPM is nearly about 190 – 210.
The range of the photodiode is tuned nearly to 6 to 10 cm above the ground.

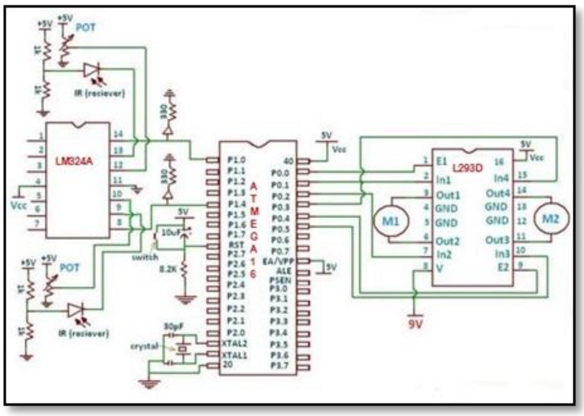

Fig - 4: Circuit Diagram

5. COMPONENT DESCRIPTION

i. Motor Driver IC – L293D

This prototype uses two dc geared motors, in order to control and drive them. The voltage of 12V is needed by the DC motors. The motor driver regulates in such a way as to provide speed and acceleration to the robot. L293D is a dual H-Bridge motor driver as shown in figure, so with one IC we can interface two DC motors which can be controlled in both clockwise and anticlockwise direction. L293D has output current of 600mA and peak output current of 1.2A per channel. Moreover for protection of circuit from back EMF output diodes are included within the IC. The output supply (VCC2) has a wide range from 4.5V to 36V, which has made L293D a best choice for DC motor driver.

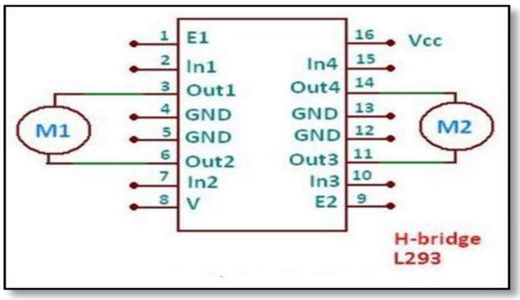

Fig - 5: Motor Driver IC - L293D

ii. Line Sensor Circuit

A Line sensor circuit is shown in Figure. It basically comprises of IR emitting diode, IR receiving transistor, resistors, potentiometer and an operational amplifier. The IR Sensor used is a general purpose proximity sensor. We are using 4 IR sensors. Here we use it for line detection and corner identification. The output of IR receiver goes low when it receives IR signal. Hence the output pin is normally low because, though the IR LED is continuously transmitting, due to no obstacle, nothing is reflected back to the IR receiver. The indication LED is off. When an obstacle is encountered, the output of IR receiver goes low, IR signal is reflected from the obstacle surface. This drives the output of the comparator low. This output is connected to the cathode of the LED, which then turns ON.

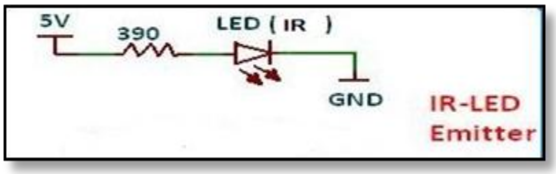

Fig - 6: Line Sensor (Emitter)

Similarly the receiver section of the line sensor is shown in the figure below.

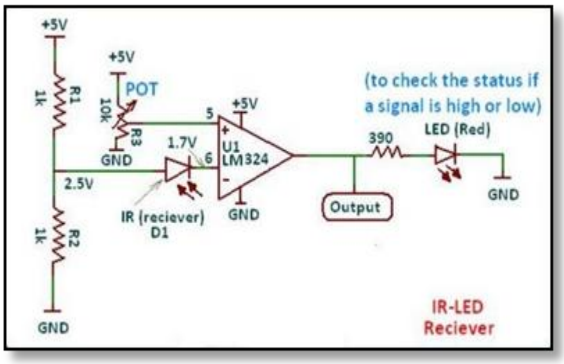

Fig - 7: Line Sensor (receiver)

6. MAZE SOLVING ALGORITHM

Right Hand Rule

Right Hand Rule states that RIGHT direction has highest priority compared to straight and left directions. Similarly straight has high precedence compared to left. The precedence order is as follows:

Right > Straight > Left

1. Always prefer a right turn over going straight ahead or taking a left turn.
2. Always prefer going straight over going left.

If the maze has no loops, this will always get you to the end of the maze

7. WORKING

The prototype is launched onto a given arena as shown in the Fig 8 below. This arena is a typical white line on black type of arena.

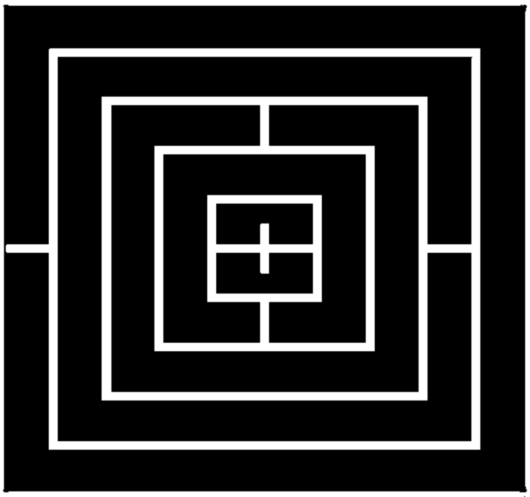

Fig - 8: The Arena

The arena is made up of several parts such as the T – Point, Bridges, + ends and the dead ends as shown in fig 9.

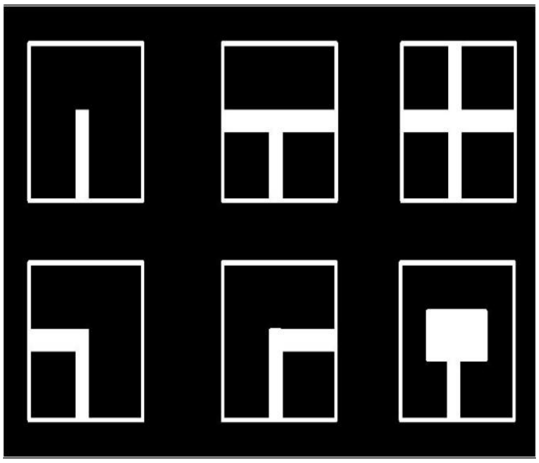

Fig - 9: Patterns used to make the arena

Logic table as shown in Table 1 is the basic truth table which was used in programming the prototype robot. Here the following annotations are used as:

0 is for low; 1 is high.

If the left motor has to be stopped and right motor has to be run then the truth table is 01 and so on.

LEFT SENSOR	RIGHT SENSOR	CODE	MOVEMENT
ON	ON	11	FORWARD
OFF	ON	01	LEFT TURN
ON	OFF	10	RIGHT TURN
OFF	OFF	00	STOP

Table - 1: Truth Table

According to the algorithm, the right hand rule tells always turn right over straight or left turn and go straight over left turn. So at first the robot will enter the first square. Then it will store the square number in an array along with the number of corners encountered in that particular square.

Likewise this data is stored for all the squares while moving from the outer to the inner square.

When a bridge is detected on the right hand side, the bot follow RHS rule and then enters into the inner square. By following all the above steps the prototype robot will reach the center of the arena.

Now, the main Artificial Intelligence concept plays on. While returning out of the arena, which is from Inner Square to the outer square, there is a comparison made with the no of corners noted while entering.

When the input is less than 2, then the bot takes the original path otherwise it takes the opposite path to come out of the shortest path of the arena, which was the desired problem statement.

8. RESULTS

After running the prototype robot it was seen that the robot worked fine after a few tuning in the variable resistance of the line sensor. This part was under the calibration of the line sensor, which was required from time to time and also dependent on the external whether conditions.

The prototype was made after taking into consideration that it could be upgraded to a better version in future so there has been provision made to attach any external peripherals to the prototype.

The Battery of 1800 mAH was sufficient for the prototype to work for about 16-18 cycles of the complete arena, making it more user friendly and mobile.

It was also observed that special arrangements have to be made for the robot along with the arena and it's material (eg. Flex as used for prototype), for certain conditions when there is an extreme climatic conditions and the Infra-Red emission is increased.

The 7 out of 10 demonstrations were successful resulting efficiency of the project to be nearly 70% and accuracy up to 80%.

It was also observed that there is some noise induced in the sensor if the robot jumped too high, leaving the ground level.

The figure shown below (Fig. 11 – 13) is the live demonstration of the prototype robot.

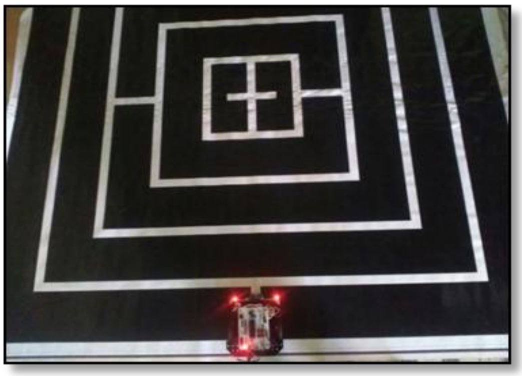

Figure 11: Starting of the game play

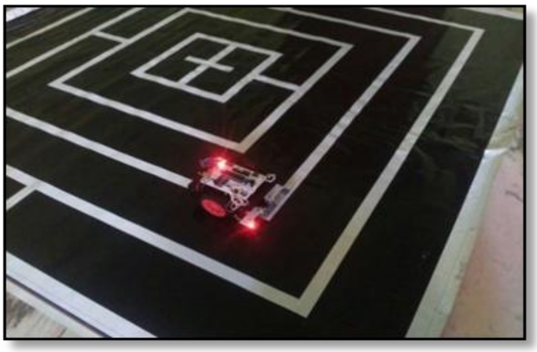

Figure 12: Middle of the game play

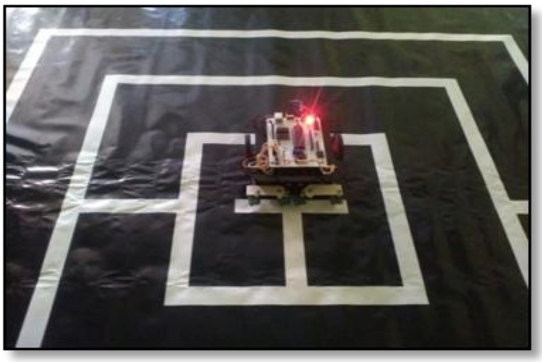

Figure 13: Centre of the Game Play

9. CONCLUSIONS

Thus, at the end of the game play it could be concluded that it was successful in achieving the goal of Artificial Intelligence in the prototype robot, that is giving a own brain-like feature to the robot so that it can work accordingly in any favorable condition and the arena provided.

Since, we were successful in providing the robot with its own sense of judgment, so it is probable that the robots in the industry could be developed with such a brain- like feature, so that they can have their own intellect to judge the path they follow, making it a helping hand in the textile industry.

REFRENCES

[1] Ibrahim Elshamarka and Abu Bakar Sayuti Saman, "Design and Implementation of a Robot for Maze-Solving using Flood-Fill Algorithm," International Journal of Computer Applications (0975 – 8887), Vol. 56, No. 5, 2012 pp. 8.

[2] Mohamed Alsubaie, "Algorithms for Maze Solving Robot", Manchester Metropolitan University, unit code-64ET3590, 2013, pp. 12-15

[3] S. C. Law, "Maze Solving Robot", Bachelor of Engineering, Universiti Teknologi, Malaysia, 2010

[4] Y. C. Chang, "Micromouse Maze Solving Robot," Bachelor of Engineering, Universiti Teknologi, Malaysia, 2009

[5] Michael Gims, "Micromouse - Microprocessor Controlled Vehicle," Bachelor of Engineering, University of East, S. L. D. B. 1999