ISSN: 2455 - 1597

ijergs
Open Access

International Journal of Engineering Research and Generic Science (IJERGS)
Available Online at www.ijergs.in

Volume - 5, Issue - 2, March – April - 2019, Page No. 173 - 185

Shortest Distance Maze Solving Robot

Sonam Gour, Mohini Dgal, Prahalad Jat

Assistant Professor, Department of ECE, Arya College of Engineering and Research Centre, Jaipur

Assistant Professor, Department of Physics, Arya College of Engineering and Research Centre, Jaipur

Abstract

In this paper, the design of maze solving robot which has the ability to navigate automatically in an unknown area based on its own decision is presented. For the proposed design algorithm, a wall following technique based on LSRB and RSLB algorithm is used. The designed robot obtains input from ultrasonic sensor, Infra-red sensor and wheel rotation encoders and then make decision for solving maze. It has the capability to solve the maze by taking the shortest distant path and it stores the details for the further reference also. The designed robot has the ability to learn any arbitrary maze and find the possible shortest route for solving it. The best application of this designed robot could be for navigational purposes

Keywords: Autonomous navigation, LSRB and RSLB algorithm, Maze Solving Robot

Introduction

A maze is a network of paths designed as puzzles through which one has to find a way. Another puzzle which is similar to maze is labyrinth which is unicursal i.e. has only a single, non-branching path. Solving a maze is more complex and difficult than solving a labyrinth. Different algorithms have been proposed to solve labyrinth but the same algorithm doesn‟t works efficiently for solving maze [1]. A maze solver must navigate from one end of the maze i.e. starting point to the other end of the maze i.e. end point. On its way, it must make important decisions at the intersections and dead ends. A maze solving robot is programmed in such a way that it will find its path without colliding the walls. It involves a two-step process in which the first step is to drive through the maze and find the end of it followed by optimization of path through which the robot can travel in the maze without going down any dead ends. So far many works have been done related with the autonomous maze solving robot. Special robots have also been designed for this particular purpose. These robots are capable of solving the maze very efficiently. But the only drawback is that their designed robot is capable of solving the maze only in one way and the path may not be same every time the robot solves the same maze [2]. The path taken by the robot is not fixed i.e. it may take the longer or it may also take the shorter route. So far there has also been no research on the shortest path maze solving robot. However, it is also important for the robot to travel in the shorter path in order to save time and distance. Based on this we propose the design of a robot which can solve any given maze in multi direction and accordingly by traversing by all the ways it finds the shortest distant path. This robot is capable of measuring the distance of the paths. And so by measuring the distance of the paths the designed robot finds and stores the shortest distant path. Once after storing the shortest distant path it travels to the shortest path every time it is made to solve the same maze. The designed robot uses two algorithms which are Left straight Right Back (LSRB) algorithm and Right Straight Left Back (RSLB) algorithm [3].The basic application of the designed robot can be for navigation purpose in which the robot once after traversing all the directions automatically follows the shortest path all

the time [4]. To solve the maze we use LSRB and RSLB algorithm. According to the LSRB algorithm first priority is given to the left direction i.e. if there is a junction the robot first turns left if there is no left turn goes straight or turns right or back as the case maybe. Hence, we give a constant priority for robot"s movement as per the given condition [5]. In most the papers freeduino board and has been used to solve the given maze by using only LSRB algorithm [6][7]. As compared in our paper arduino has been used in which we can give more functions to the board making the task easier and also our designed robot uses both LSRB and RSLB algorithm making it easier to find the shortest path. The designed robot not only solves the maze but it also finds the shortest distance to solve the maze. Firstly the robot is made to solve the maze by all possible ways using the mentioned algorithms and then by comparing the distance of different paths it stores the path of the shortest distance. The designed robot uses the wheel encoder to determine the distance travelled by it. This device allows the robot to measure the distance and the speed precisely and efficiently [8][9]. The robot will firstly move according to LSRB algorithm to search the end of the maze, then it will travel back to the starting position without going down to any dead end and distance covered in that path. Then the robot will move according to RSLB algorithm to search the end of the maze, then it will travel back to the starting position without going down to any dead ends and during its travelling back to the starting point it will measure the distance covered in that path. Then it will compare both the distances and find the shortest path between them and then it will continue to move in the shortest path[10].

Overview of Maze Solving Robot

The robot consists of 3 infrared sensors, 1 ultrasonic sensor, arduinouno microcontroller, pair of motors, pair of wheels and wheel rotation encoder. The robot is capable of scanning the area ahead for obstacles with the help of ultrasonic sensor.. The robot is also equipped with 2 infrared sensors facing right and left to detect the walls. One infrared sensor is placed facing the floor. A wheel rotation encoder is placed near each wheel to measure the extend of how much the wheel is rotating. By knowing the diameter of wheel, the rotation can be converted to distance travelled .

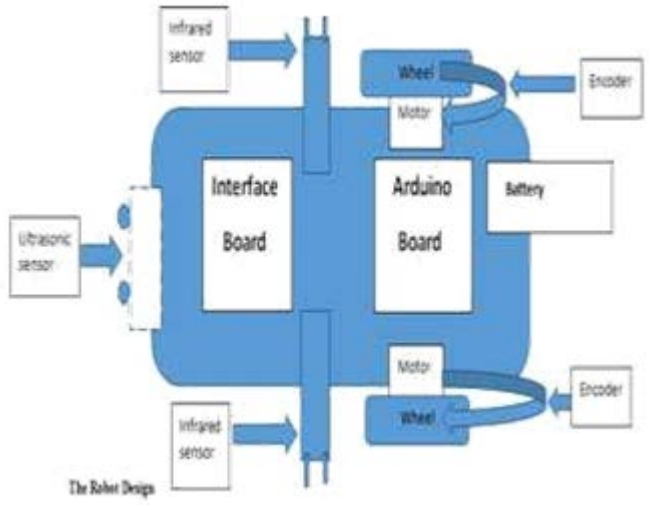

Fig. 1: Design of maze solving robot

Control Board

Arduino Uno microcontroller provides the processing power. Arduinouno is an open source microcontroller based on AtMega328P which has 14 digital input/output pins, 6 analog pins, 16MHz crystal oscillator etc. Arduinouno acts as the brain of our robot because all the decisions which is taken by the robot to solve the maze is governed by this board.

Obstacle Sensor

The robot is equipped with 1 ultrasonic sensor and 2 infrared sensors. Ultrasonic sensor transmits ultrasonic waves from its sensor head and again receives the waves reflected from the object. By measuring the length of time from the transmission to reception of sonic wave, it detects the position of the object. Ultrasonic sensor measures the distance between the robot and the obstacles in centimeters. The infrared sensors are placed on the left and right of the robot. Color and distance of reflecting surface are the two parameter on which the reflectivity of infrared light varies.IR sensor use reflective photo sensor module to detect the distance and color. The signal intensity received by IR sensor increases whenever a light colored object approaches which turns the onboard led indicator. The signal intensity decreases when dark colored object approaches which leads to turn off the onboard LED.

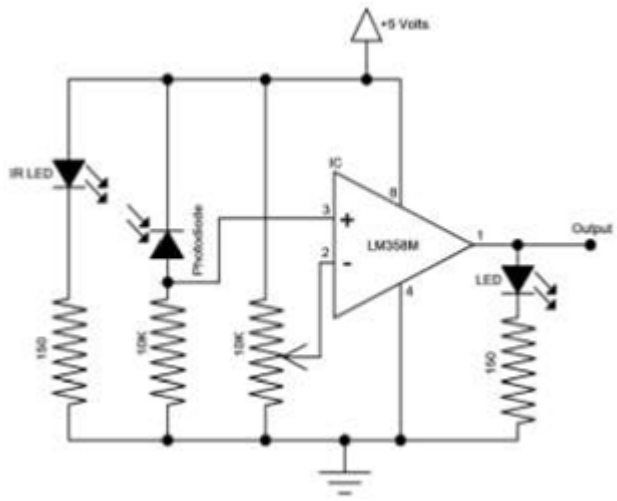

Fig. 2: Infrared sensor circuit

Infrared light is emitted by the infrared LED which is detected by the photodiode. LM 358 integrated circuit is used as voltage comparator and potentiometer is used for calibrating the output of the sensor. When the light emitted by the infrared LED falls on the photodiode after hitting the object, resistance of the photodiode falls down a huge value. One of the inputs of the operational amplifier is at the threshold value set by the potentiometer whereas the other input is connected to photodiode‟s series resistor. The voltage drop across the series resistor will be high when the incident radiation is more on photodiode. Both voltages are compared in the op-amp and if the voltage across the resistor series to photodiode is more than that of threshold voltage, the output will be high and onboard LED glows.

Wheel Rotating Encoder

A pair of infra-red transmitter and receiver is equipped on each wheel. By knowing the wheel diameter and counting the holes in the wheel, distance calculated by the robot can be calculated.

Motor Drive

Pair of dc motors are interfaced to the arduino through L293 H-Bridge to drive the wheels. L293 can drive two motors which can be controlled in both clockwise and anticlockwise direction with output current of 0.6A and peak current of 1.2 A per channel. The circuit is protected from back EMF at the outputs by in-build diodes. Supply voltage range varies from 4.5V to 36V, making L293 a flexible choice for the motor.

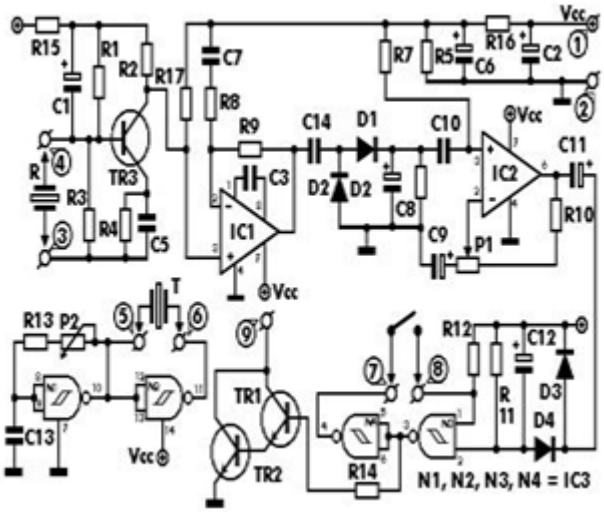

Fig. 3: Circuit diagram of ultrasonic sensor

Sonic transducer is used for ultrasonic proximity sensor which allows for alternate transmission and reception of sound waves. The sonic waves emitted by the transducer are reflected by the object and received back in the transducer. After emitting the sound waves, the ultrasonic sensor will switched to receive mode. The time elapsed between emitting and receiving determines the distance of object from the sensor.

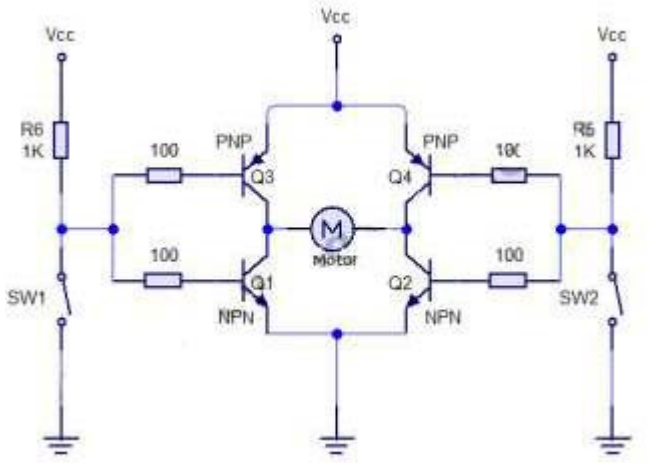

Fig. 4: Circuit diagram of L293 H-Bridge

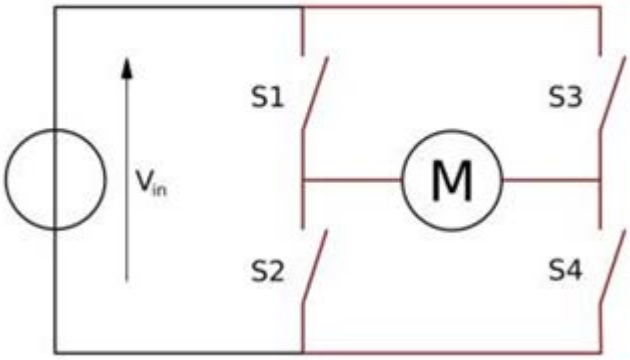

Fig. 5: Structure of H-Bridge

H-Bridge is an electronic circuit that enables voltage to be applied across a load in both directions. The following table summarizes operation, with S1-S4 corresponding to the diagram above.

Table 1: Operation of H-Bridge

S1	S2	S3	S4	Result
1	0	0	1	Motor turns right
0	1	1	0	Motor turns left
0	0	0	0	Motor coasts
0	1	0	1	Motor brakes
1	0	1	0	Motor brakes
1	1	0	0	Short circuit
0	0	1	1	Short circuit
1	1	1	1	Short circuit

Algorithm

LSRB Algorithm

The robot is programmed with an algorithm that helps it to navigate the entire maze until it finds the end. Our designed robot uses left hand on wall technique and right hand on wall technique. Both the algorithms will help robot to navigate the maze until it find the end of it. Left hand on wall technique states that when there is an intersection turn left if you

can, else go straight if you can, else turn right if you can, else turn around because you are at dead end. The LSRB algorithm is processed calculating the shortest path of the robot given below.

During the search process the robot travels in accordance with the algorithm and it will store its path in the memory. The robot remembers its path by storing each turn as a letter in the array. The second step in maze solving is taking the path the robot travelled and shortening it to correct path to the end of the maze without travelling down any dead ends. The movement gets stored as follows:

Left turn="L", Right turn="R", Turn around="B", Go straight="S"

Figure 6 denotes the starting point of first decision based on the LSRB priority so the robot will give higher priority to left and chose left direction at the intersection. This decision value is stored in the register as L.

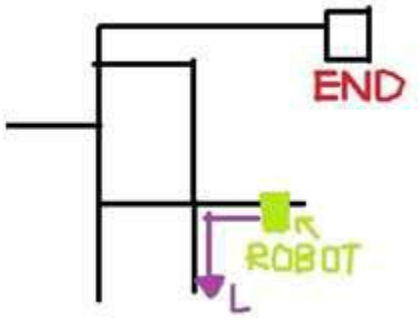

Fig 6: Path stored L

In the 7rd figure, there is no option available to turn right or left or go straight therefore it takes decision to go around or turn 360 degree. This decision value is stored in the register as B.

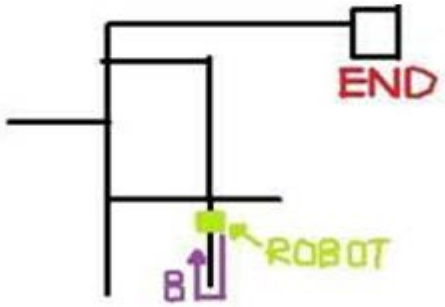

Fig.7: Path stored LB

Infigure 8,the decision is chosen to turn left because it will give higher priority to left direction in accordance with LSRB algorithm. The decision value is stored in the register as L

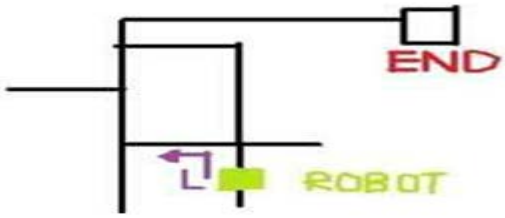

Fig. 8: Path stored LBL

In figure 9, the decision is chosen to turn left. The decision value is stored in the register as L.

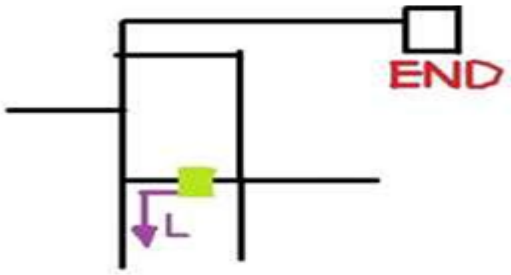

Fig. 9: Path stored LBLL

In figure 10, the decision is chosen to go around because it is a dead end and there are no options available. The decision value is stored in register as B.

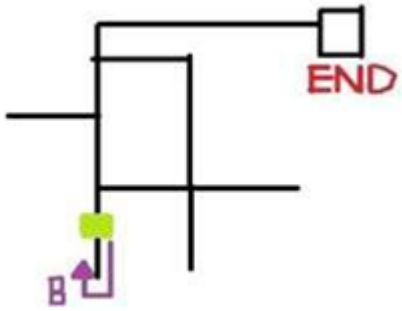

Fig.10: Path stored LBLLB

In figure 11,the decision is chosen to go straight. The decision value is stored in register as S.

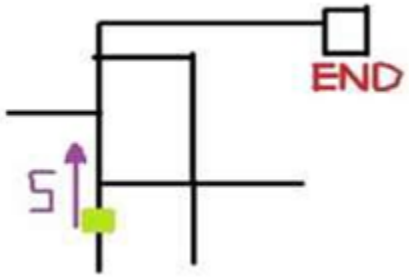

Fig. 11: Path stored LBLLBS

In figure 12, the decision is chosen to turn left according to the highest priority of left turn in LSRB algorithm. The decision value is stored in register as L.

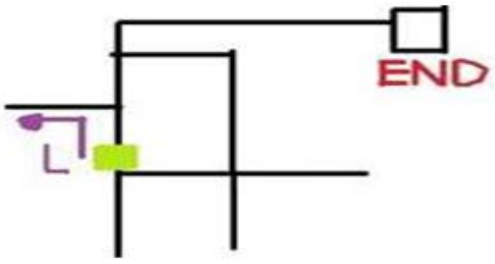

Fig. 12: Path stored LBLLBSL

In the figure 13, it encountered dead end so it will take the decision to go around. The decision value is stored in register as B.

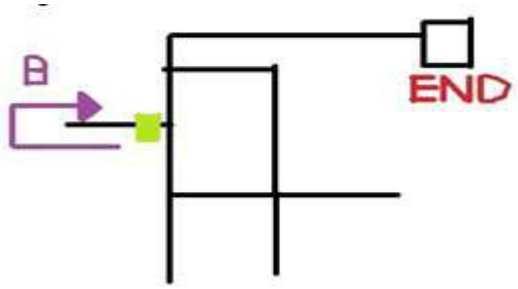

Fig. 13: Path stored LBLLBSLB

In figure 14, the decision is chosen to turn left. The decision value is stored in the register as „L".

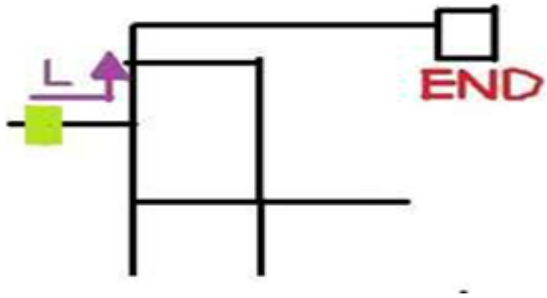

Fig 14: Path stored LBLLBSLBL

In figure 15, the decision is chosen to go straight. The decision value is stored in the register as S.

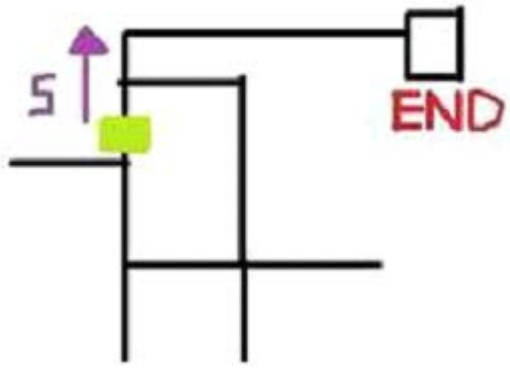

Fig. 15: Path stored LBLLBSLBLS

In figure 16, the decision chosen by robot is to turn right. The next decision value is stored in register as R.

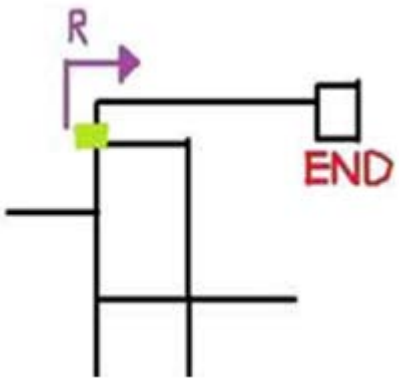

Fig. 16: Path stored LBLLBSLBLSR.

After the sixteenth step the searching process will be finished and finally the "LBLLBSLBLSR" value would be stored in the register. The next process is the travelling process. In this travelling process the robot will go to the destination without any searching and using the shortest path calculated below. „B" indicates that the path taken by the robot is wrong path. A perfect path would take the robot to a dead end and a „B" indicates a dead end was taken. So when shortening the path, the robot looks for these dead ends taken and tries to get rid of them by using a few things it is programmed to know. It is programmed with multiple 3 letter sequence which tells it what to replace wrong moves with. The three letter sequence is as follows:

Table 2: Equivalent path for the sequence

Sequence	Equivalent Letter
LBR	B
RBL	B
SBL	R
LBL	S
RBR	S
SBR	L
LBS	R
RBS	L
SBS	B

Now during shortening procedure whenever the above 3 letter sequence comes we have to replace it with equivalent letter mentioned above. The stored path is "LBLLBSLBLSR"

Continue shortening until all „B" are gone (LBL=S) LBSLBLSR

The new path will be SLBSLBLSR.

Continue shortening it S (LBS=R) LBLSR.

The new path will be SRLBLSR.

Continue shortening it SR (LBL=S) SR.

The new path will be SRSSR.

RSLB Algorithm

Right hand on wall technique says that when at intersection turn right if you can, else go straight if you can, else turn left if you can, else turn around because you are at dead end. Here the top priority is given to right turn. Fig. 17 denotes the starting point and the first decision based on the RSLB priority so the robot will give higher priority to right and chose right direction at the intersection. The decision value is stored in the register as R.

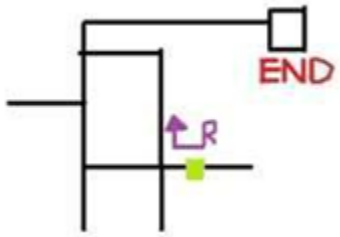

Fig. 17: Path stored R

In 18th figure, the decision taken by robot is to turn left. The decision value is stored in the register as L.

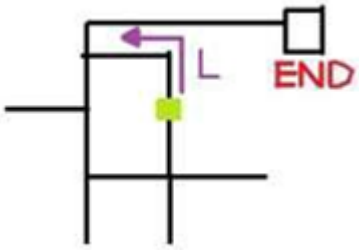

Fig 18: Path stored RL

Similarly the next decision it take after processing the previous decision is to turn right and the decision value is stored in the register as R. Finally the last decision it takes is to turn right. So the final path stored in the array is RLRR. Since there is no B in the stored path so no shortening is required and robot will move in the same path.

Working Of Maze Solving Robot

Initially it will send ultrasonic waves through ultrasonic sensor to detect the object in front of it. Then it will calculate the distance of the robot from the object usingultrasonic sensor. So the distance of the object from ultrasonic sensor is stored in a variable names as distance. Then it will take readings from both Infrared sensors fixed on left and right side of the robot. These Infrared sensors will scan for the objects on the side of the robot. The readings from both the sensors are stored in a variable named as leftsen and rightsen respectively. One infrared sensor is also fitted at the bottom of the robot to sense the black strip which is placed at the bottom of the robot. Two black strip are placed at the starting point and end point of the robot. When the robot is at startimg position, the bottom infrared sensor will sense the black strip and give a digital reading of 0. Initialize a temporary variable count to 0 and the value of the count variable will increase by 1 every time when there is change in reading in lower infrared sensor. So when the count value is 0 i.e. robot is at starting position. After getting all the readings from the sensors it will make a decision whether to go right or left or straight or go around according to the LSRB algorithm discussed above. The decision made by the robot at the intersection is based on

the LSRB algorithm and the readings it got from the sensors. During the searching process the robot will store all the decision taken by the robot at the intersections in an array names as path array. When the lower sensor reading is 0 again it means it has reached the end point of the maze and the value of the count will be 1. Turn the robot 180 degree. Now the lower sensor reading is 1 and the value of count is2. Now it will move in the shortest path calculated by LSRB algorithm without going down any dead ends.We have to short the array until all „B‟ are removed from the patharray and store it in a new array named as shortpathLSRB. Now reverse the elements of the array such that „L‟ is replaced by „R‟, „R‟ is replaced by „L‟ and no change in „S‟ i.e. if the shortpathLSRB has elements „SRSSR‟ so it will reverse this and store it in new array as „LSSLS‟. During travelling in the shortest path it will measure the distance of shortest path with the help of wheel rotating encoder and store the distance of this path in a variable named as shortdist1. When it travelled through the shortest path and reached the starting point, the lower sensor will detect the black strip and change its sensor value. So the count value will increase and robot will turn 180 degrees. When the count value is 3 the robot will move according to RSLB algorithm and the decision taken by robot at the intersection according to RSLB algorithm is stored in an array named as patharray2. When the robot will reach the end of the maze after searching the path using LSRB algorithm, the lower sensor would detect the black strip and count value will change to 4. It will then take a 180 degree turn and lower sensor reading will be 1 again thereby changing the count value to 5. Now it will move in the shortest path calculated by RSLB algorithm without going down any dead ends. Now the shortening and reversing of array takes place as discussed above. During its travelling in the shortest path calculated by RSLB algorithm, it will measure the distance of this shortest path with the help of wheel rotating encoder and store the distance in a variable named as shortdist2. When the lower sensor reading would be low, the value of count will be 6 and robot will take 180 degree turn. As value of count become 7, it will then compare the value of shortest distance stored in the variable shortdist1 and shortdist2. The variable which hasthe least value will be shortest path and the robot will continue to move to the corresponding path of the variable which has least value.

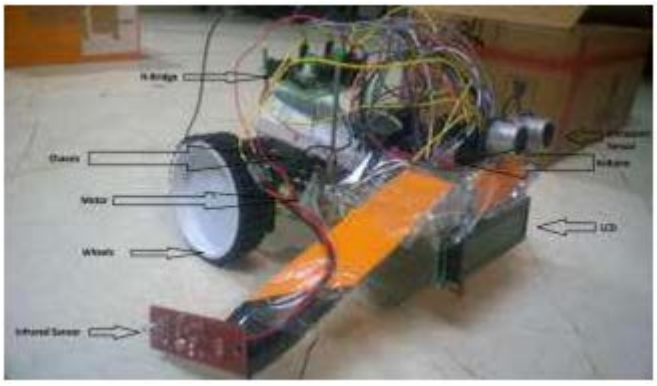

Fig. 19: Model of maze solving robot.

Fig. 20: Robot taking left turn according to LSRB algorithm

Fig. 21: Robot moving straight in accordance with LSRB algorithm

Fig. 22: Ultrasonic sensor detected the wall and robot is giving priority to left direction according to LSRB algorithm

Fig.23: Robot clearing the obstacle

Conclusion and Future Work

A maze solving robot using LSRB and RSLB algorithm has been designed and tested in real time. The designed robot has proved its capability of solving any arbitrary maze by finding the shortest path due to effectiveness of algorithm. The proposed algorithm has low space complexity, high performance and provides optimal solution to the maze.

References

1. BagusArthaya, Ali Sadiyoko&ArdeliaHadiwidjaja, The design of a maze solving system for a micromouse by using a potential value algorithm, World Transactions of Engineering and Technology Education, 2006 UICEE Vol. 5, No. 3, 2006

2. ShoujiangXu, "A Summary of the Maze Algorithm", China Computer & Communication, 2009

3. Swati Mishra, PankajBande, Maze solving algorithms for Micro Mouse, 2008, IEEE International Conference on Signal Image Technology and Internet based systems, 2008 IEEE, DOI 10.1109/SITIS.2008.104, p86-93

4. Adil M.J Sadik, MarufA.Dhali, Hasib M.A.B. Farid, Tafhim U Rashid, A. Syeed, A Comprehensive and Comparative Study of Maze-Solving Techniques by Implementing Graph Theory, 2010 International Conference on Artificial Intelligence and Computational Intellegence, DOI 10.1109/AICI.2010.18 2010 IEEE,p52-56

5. BabakHousseiniKazerouni, Mona BehnamMoradi and PooyaHosseiniKazerouni, "Variable Priorities in Maze-solving Algorithms for Robot‟s Movement", 2003

6. Kazuo Sugihara, John Smith, "GeneticalAlgortihms for adaptive motion planning of an autonomous mobile robots", Problems IEEE Trans, USA,SIM 1997

7. KazerouniB.H.Moradi, "Variable Priorities in Maze-Solving Algorithms for Robot‟s Movement" Seville: Proceedings IEEE International Conference on Industrial Informatics, vol.6, pp181-185,2003

8. Choudhury D.R, Process Control System in Modern Control Engineering 2004, PHI Learning Pvt. Ltf, p323-351

9. Jianping C, et al. A micromouse maze solving simulator in Future Computer and Communication (ICFCC), 2010, 2nd International Conference on 2010

10. Ahmed N.Mosa, AymanM.SaadMoustadaA.Mohamed, Micromouse: An Intellignet Maze Solving Robot", IEEE Egypt Student Paper Contest, September 2005, Cairo Egypt