Available online at www.sciencedirect.com
ScienceDirect
Procedia Computer Science 105 (2017) 57 – 61
Procedia Computer Science

2016 IEEE International Symposium on Robotics and Intelligent Sensors, IRIS 2016, 17-20 December 2016, Tokyo, Japan

Maze solving robot with automated obstacle avoidance

Rahul Kumar, Peni Jitoko, Sumeet Kumar, Krishneel Pillay, Pratish Prakash, Asneet Sagar, Ram Singh, Utkal Mehta*

School of Engineering & Physics, Faculty of Science, Technology & Environment, The University of the South Pacific (USP), Suva, Fiji.

Abstract

A quick development of innovation moves us to plan the best choice for an accurate mission. Numerous independent automated innovations are intimated in the lives of individuals making their work much easier. It has been seen that automated vehicles are presented so far, with shrewd abilities after enormous measures of cash spent yearly on the examination. Here in this paper, autonomous maze solving robot is developed with independent mapping and localization skill. Firstly, the maze solving vehicle is designed with three infrared sensors of which two is used for wall detection to avoid collision and the third is for obstacle detection for picking and placing the objects to clear its pathway with the help of robotic arm. Also, it desires to use robot where an environment unreachable for human. In addition, there are also places where use of robots is the only way to achieve a goal. For this, appropriate placement of sensory devices is very critical. We have successfully implemented a maze solving ability onto the robot so called MazeBot. It has been tested that the robot can solve the maze successfully without any interruption with the walls and the objects. In this design, the accuracy of measurements and the real-time processing allied with minimum processing power are the key components in overall embedded design.

Peer-review under responsibility of organizing committee of the 2016 IEEE International Symposium on Robotics and Intelligent Sensors(IRIS 2016).

Keywords Mechatronics; Mazebot; Microcontroller; Maze Solving Robot; Infrared Sensors; Robotic Arm.

1. Introduction

Dense network of city streets, disaster prone regions and war torn environments have common navigational difficulties. These shortcomings range from navigating through obstruction or wreckage on a possible route, navigating around dead ends to figuring out new and complex paths [1]. Specific missions, hazardous surroundings, inhabitable conditions serve as reason for the shift to autonomous technology and decision oriented mechatronics. Another reason to opt for autonomous technology or unmanned vehicles is that these are introduce conservative ways or rescue of survivors due to its impeccable, refined and numerously reviewed decision making and being able to enter a scene or an environment, locate objectives and exit the quickest and most effective way possible whilst eliminating obstructions. The primary worries for this type of technology are processing power, real-time processing and the measurement accuracy [1-3]. It is well-known that the idea of decision making algorithms for applications like maze solving leads an entrance to a goal. Regardless of the challenge of maze solving being close to three centuries old, it still holds a vital stature in robotics. Maze solving is part of the most imperative section of robotics design, which is the Decision Making Algorithm. If a robot is positioned in an unfamiliar environment, it should have a good/exceptional decision making algorithm in order to

*Corresponding author: Tel: (+679) 3232337
Email address: mehta_u@usp.ac.fj

Peer-review under responsibility of organizing committee of the 2016 IEEE International Symposium on Robotics and Intelligent Sensors(IRIS 2016).
doi:10.1016/j.procs.2017.01.192

successfully solve the maze. There has been development in the maze solving or rather decision making algorithms under the micro-mouse algorithm type. The micro-mouse algorithm has developed from wall follower algorithm to flood filling algorithm, where specific algorithms require either full vision of the maze or just mediate portions. It is reported that much needed integration of mechatronics systems is the maze solving robot with a 5 degree of freedom robotic arm for the elimination of obstacles on the robot's path. This is a new feature of a maze robot since previous versions of maze solving robots only concentrated on the navigation and not the physical elimination of obstruction to introduce another possible path to the goal.

In literature, it was demonstrated a maze-solving robot designed to solve a maze, based on the flood-fill algorithm [2], based on Partition-central Algorithm [3]. According to "Seven Bridges of Konigsberg" [7], maze has a mathematical solution to solve a problem that is more a geographical problem. It was explained with an example that there were two Islands in the city of Konigsberg, Prussia which had been connected to the mainland and each other by seven bridges. The challenge set was the possibility if all the seven bridges were able to be crossed only once and eventually return to the starting point. Meanwhile, majority of the algorithms used these days share a close relation to graph theory where there exists a perfect maze or simply a maze without loops which in graph theory equivalence is a tree. Generally, algorithms with relations to graph theory are more superior, efficient, accurate and proficient whilst compared to non-graph theory algorithms when implemented practically on geographic locations. There have been several algorithms utilized in modern day maze solving [4-9] such as wall follower, pledge algorithm, recursive backtracker algorithm and Trémaux's algorithm. In the Wall follower algorithm, either the right hand rule or the left hand rule can be used to solve the maze. This algorithm is very fast and uses no extra memory to solve the maze while having a simple logic of movements. The logic of the wall follower algorithm is observing being in a dark room and finding one's way using the walls of an enclosure and doing this (either with your left or right hand), the solver would eventually make its/his/her way out of the maze. The pledge algorithm is another such algorithm which is similar to the wall follower method and has the feature of solving imperfect mazes i.e. mazes which have loops in them. The pledge algorithm counts the number of turns it makes and if its sum comes to 0 (+1 being right turns and -1 being left turns), when it does come to 0, it leaves the island/loop and continues with using the right hand rule and counting turns. The Recursive backtracker algorithm uses stored coordinates in arrays and does not take the same path again since if it finds a wall ahead or a dead end, it marks it and does not go to those coordinates again but reverses to the last known option or rather detour and continues from there. This algorithm does not necessarily find the shortest path and takes up memory depending how big the maze is. Finally, the Trémaux's algorithm is similar to the previous algorithm (Recursive backtracker) but what this algorithm does is that it does not use memory to store coordinates but physically marks the maze. This algorithm only marks the routes leading to dead ends after locating one. This may ensure least memory used but may use limited materials for marking and may run out of materials like paint or ink.

2. Structure modeling and design

Any mechatronics application consists of four stages: Mechanical, Electrical, Computer and Information Systems. Firstly, it is desired to build mechanical structure and electronics sensor circuit. At the final stage, microcontroller is programmed as per the solving technique adopted. In following, all main tasks are discussed to assemble autonomous maze solving robot. Remember, the aim is not only to solve maze automatically but also to avoid obstacles in the way of propagation.

2.1 Design of overall structure or chassis

The main structure of the system consists of Perspex glass material and the design is circular in shape to incorporate all the components to be seated perfectly onto the base. The main objective is to pick and place the objects on its path and solve the maze successfully. The chassis hosts a robotic arm which is held intact by four screws and protected in-between the two Perspex plates is mounted the Arduino board, motor driver and distribution board. Under the chassis there are housed of two servo motors which control the direction of the fully autonomous robot. The servo rotates the vehicle at any desired direction by using differential drive logic. Fig. 1 shows the basic power platform and chassis for the maze solver robot, namely called here after 'MazeBot'.

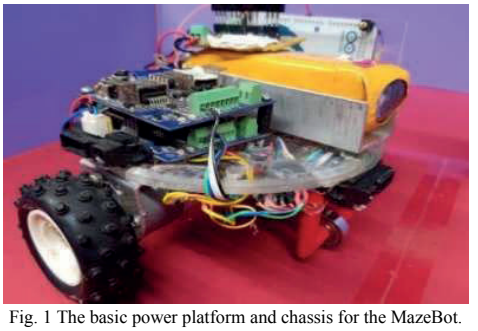

Fig. 1 The basic power platform and chassis for the MazeBot.

2.2 Vehicle Control

In this assembly, a vehicle is controlled by a pair of servo motors operated at 9V. As shown in figure, each motor has maneuvered by rubber tires fixed on the sides and bearing wheels mounted on the back and front for smooth and sharps turns.

2.3 Robotic Arm for Pick-n-Place

Figure 2 shows the main construction with robotic arm, completed after following three states of assembly and has two movable joins. It is used to remove the obstacle during maze solving process to clear the path way.

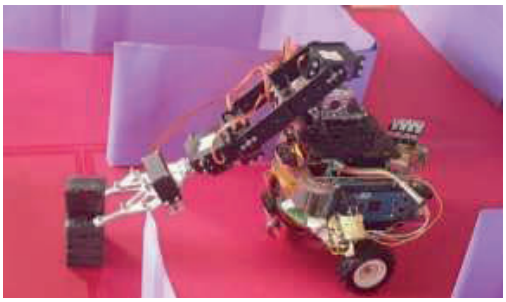

Fig. 2 Robotic arm and gripper.

First state includes the arm going downwards, opens the gripper, and picks up the object than closes the gripper. Second state is known as the medium level, from first state of the arm has already picked the object so in second state the arm with the object just vertically flips itself and goes backwards in its dropping position. From second state, the arm is flipped backwards than enters into third state where its function is to open the gripper than drops the object and returns to its initial state, keeping the gripper in an open position.

Table 1: State table for two servo joints.

	State 1	State 2	State 3
Joint 1	150°	50°	0°
Joint 2	100°	0°	150°

Table 1 shows the state table for two servo joints that govern the motion of the arm. In this arm, there are 2 axes of rotation and 1 gripper where it can consume up to 6.4 volts of voltage and a current of 6 amps. At a time 2 servo motors are powered up for picking and placing the object. The total PWM (pulse width modulation) range is from 0 to 100 and the rotating range goes from 0 to 180 degrees therefore PWM determines the rotation; for example a PWM of 128 will give a rotation of 90 degrees.

2.4 IR sensing logic for wall following and Object detection

In this assembly, we have used two infrared proximity sensors for detecting the walls. MazeBot would go on following the right wall till it finishes solving the maze. The sensor has a sweet spot of 8cm to 12cm. The third infrared sensor is fully dedicated for detecting obstruction on its path. The sensor senses the object at about a distance of 15cm than after it has sensed the object it than moves back of about 10cm while keeping in a straight position and not deviating from its path extends its arm, picks up the object and drops it accordingly. There are three levels of detection presented in this sensor logic. As shown in Fig. 3, the sensor logic helps to detect the wall and obstacles during solving the maze.

Fig. 1: Logic thresholds for navigation.

3. Sensor calibration

There are two types of IR sensors used to assemble the MazeBot, SHARP GP2D120 and SHARP GP2Y0A21YK0F. The GP2D120 IR sensor has an operating range of 4cm to 30cm. This sensor is placed on the side of the MazeBot to follow the wall. The datasheet provided a basic linearization of the sensor within the operating range; however, a calibration was done to get the distance in centimetres of the sensor. The GP2Y0A21YK0F IR sensor has an operating range of 10cm to 80cm. This sensor was placed in the front for wall detection and obstacle detection. The accuracy of the sensor could easily be manipulated using equations however the precision was a problem as the sensor had a precision of less than 60%. In order to overcome the precision problem, ten readings were taken and averaged in order to get a more stable value. This has increased the precision to more than 92%. This meant that there was only an error margin of 8% in existence. The following equation governs the sensor reading from the sensor:

$$Y = (-42 \times R) + 28 \tag{1}$$

Where, Y is the distance reading in cm, R is the percentage voltage reading from the sensor with respect to the reference. The same equation was used for both sensors; however, both sensors had different thresholds in the robot drive logic.

Another sensing circuit is embedded with DC servo motor as feedback encoder. The DC servo motors has a gear ratio of 1:29. These motors have a no load RPM of 10000 at 12V as per Pololu datasheet [10]. We have chosen operating voltage for both motors 9V. As such, the manufacturer's page suggests a linear relationship between the motor speed and terminal voltage. As such the maximum RPM for the motor would be 7500, reduced with a ratio of 29, the shaft RPM is around 258 RPM. The motors are fitted with 64 PPR optical encoders that will be detected through interrupts in the microcontroller. The encoders are used in this application for distance measuring and mapping. The encoder count is checked and reset every 1 millisecond and the count is used to gauge the present speed and also to map the distance system has covered in the maze. To get the distance covered, the following relation was used:

$$D = \frac{N}{18.56} \times 2\pi r \tag{2}$$

Where, D is the distance covered in centimetres, N is the number of encoder pulses recorded and r is the radius of the wheel. Each direction the robot moves in has a sentinel value which will enable the interrupt to update the distance covered in which direction by the robot. The location of the robot and the mapping of the maze will be done by creating a 2-dimensional array that will be updated whenever a sentinel reaches 30cm. The maze has a resolution of 30cm as described above. This implies that the maze can be treated as a 2D array with 30cm by 30cm large pixels.

4. Issues and solutions

Even we assume all components near perfect and performing as per data sheet, practically some issues are found to be common. In following we have listed issues related to power, servo operation, calibration etc. and proposed troubleshooting methods.

i. **Regulator voltage and current**: The issue was quite a severe one, as in the initial stages we were using a lab DC power supply and were supplying the arm with 6.4V at 6A but when it came to circuit implementation, the approaches to build a 6.4V regulator seemed too complex and available regulators had a max power rating of 1.5A each. To solve this issue, we used a simple circuit that we found online which uses a 5V regulator and is raised two diode drops (~1.4V) by connecting two diodes on ground pin of the regulator with the diode facing ground. This gave an output of 6.4V, but we required 6.4V at 6A, so 4 of these circuits were placed in parallel.
ii. **Arduino power up:** We noticed that the Arduino worked fine with the USB mini cable plugged but showed no sign of life when supplied with 6.4V on the input voltage pin. Upon reading online articles on the Arduino website, it was learnt that the built in regulator had been fried. A solution to this issue, by several forum users was to power up the Arduino directly by supplying 5V regulated to the high level of pins which worked without flaws.
iii. **Motors gear:** It was noticed that the left motor was a bit slower than the right motor so we initially gave plus 5 PWM to the left motor to counter this but we saw that the error accumulated which caused the robot to function different from expected. Upon visual examination, we noticed that the shaft for left motor was a bit wobbly than the right motor, so we disassembled the gearbox and located the culprit which was a broken gear hub. This was beyond repair so we replaced the gearbox
iv. **Robotic arm:** This was a tricky issue as the base servo had a single mount on one side only which made it unstable when it was moving and the base servo had not been properly designed to support the weight of the entire arm. To solve this, we placed an extra mount to securely hold the base servo and from our previous servo DIY kit, we retrieved a strong bearing mount which could very easily take the weight and placed it between the base servo and the rest of the arm to promote smooth operation.

5. Remarks

This project has successfully been able to automate the maze solving capability of a robot using maze-solving algorithms. In addition, the robot has also been able to pick up obstacles and move it out of the robots path when traversing the maze. These capabilities have been achieved by using IR sensors for obstacle and wall detection, motor control and using a robotic arm mounted

on the maze-bot. For the robot to make its decisions it relies on inputs from several sensors, namely IR sensor and encoder calculations. It has been noticed that the structure would able to correct its orientation errors arising from its physical motion within the maze.

Main applications of such automated maze solving robot are; automated crate moving within a warehouse, military applications may involve, robots traversing unknown terrain while avoiding or moving obstacles out of the way and in rescue or emergency scenarios whereby robots are required to go into hazardous or hard to reach places. In addition, it may useful in earthquake affected areas that it may deem unstable rubble. For future works, a full visual system may be incorporated, enabling he robot to realize its environment fully.

There were a few things that could be improved and implemented further in this system. For instance, the implementation of an encoder feedback memory system proposed in the analysis. A study may include but not limited to studying the robot's maze-solving capability in a bigger and more complex maze with reconfigurable feature. The next development if this robot should include a robustness study that actually attempts to implement this robot on a larger more realistic scale. The model needs to be scaled down for easier testing and faster debugging. Image processing could have been implemented and is a viable option related to this present prototype.

6. Acknowledgement

The project team would like to thank the School of Engineering and Physics, FSTE, USP for providing excellent lab support and materials to complete the robot.

References

1. Chatelais Q., Vultur H, and Kanellis E., "Maze Solving by an Autonomous Robot", *Aalborg University*, 2014.
2. Elshamarka and Saman A., "Design and Implementation of a Robot for Maze-Solving using Flood-Fill Algorithm", *International Journal of Computer Applications*, 2012, vol. 56, no. 5, pp. 8-13.
3. Cai J., Wan X., Huo M., and Wu J., "An Algorithm of Micromouse Maze Solving", *In Proceedings of the 2010 10th IEEE International Conference on Computer and Information Technology (CIT '10)*. IEEE Computer Society, Washington, DC, USA, 1995-2000.
4. Mishra S., and Bande P., "Maze Solving Algorithms for Micro Mouse", *IEEE International Conference on Signal Image Technology and Internet Based Systems*, 2008.
5. Dang H., Song J., and Guo Q., "An Efficient Algorithm for Robot Maze-Solving", in Proceedings of the 2010 Second International Conference on Intelligent Human-Machine Systems and Cybernetics IEEE Computer Society –vol. 02. p. 79-82.
6. Kern H., Through the Labyrinth: Designs and Meanings over 5000 years., Prestel Publishing, 2000.
7. Alexanderson G.L., "Euler and Königsberg's Bridges: A historical view," *Bulletin of the American Mathematical Society,* pp. 567-573, 2006.
8. Gims M., "MICROMOUSE: Microprocessor Controlled Vehicle," University of East London, London, 1999.
9. González F., Eusebio D., and Zazueta L., "Action selection and obstacle avoidance using ultrasonic and infrared sensors", Frontiers in Evolutionary Robotics, 2008, pp. 327-340.
10. Datasheet, "Pololu DC gearmotor", https://www.pololu.com/