Project Title: A Maze Solving Robot
Course Number: CSE310L
Course Title: Lab work based on CSE 310 **(Electronics II)**

Serial No	Name	ID	Theory Section	Lab Section
1*	Sabbir Hossain Zishan	1920490	1	1
2	Nusrat Jahan	1921793	N/A	1
3	Ahad Al Tanvir Ahmed	1920468	N/A	2
4	Farhan Noor Dehan	1920269	2	2
5	Rezuana Tasmin Chaity	1920492	N/A	2

Instructor Name: Azfar Hossain

Table of Contents

Abstract

The maze-solving robot, which is designed to find a path on its own without assistance from outside sources, is briefly described in this essay. This paper presents the design, detailed circuit schematic, and programming of a maze-solving robot. Robots that follow lines along a predetermined course are different from robots that solve mazes. The robot moves through the specified maze, which has turns and crossroads, using ultrasonic sonar sensors. A maze-exploring, maze-learning, and maze-solving algorithm is implemented in the maze-solving robot. These flexible, autonomous robots can be utilized for a range of jobs, such as bomb disposal, pipe inspection, material handling, and warehouse management, Medical emergencies, Rescue operations etc.

This work develops an autonomous maze-solving robot with independent mapping and localization abilities.

The maze-solving vehicle is equipped with three sonar sensors, all of which are used for wall detection to prevent collisions as well as obstacle detection to pick up and place objects to clear its path.

Additionally, it wants to utilize robots in environments that are inaccessible to humans.

There are also situations where using robots is the only way to accomplish a task.

The proper positioning of sensory equipment is essential for this.

The so-called Maze Solver robot now has the capacity to navigate mazes thanks to our effective implementation.

It has been demonstrated that the robot can successfully navigate the maze without being hindered by the walls or other obstacles.

Introduction

An autonomous robot that can navigate around a maze is called a maze solver robot.

Any task assigned to an autonomous robot should be completed without involving a human. Wall mazes and line mazes are two of the various types of mazes. A maze solver robot is an improvement of a line follower robot. We can add a wide variety of algorithms to make it easier for the robot to navigate the maze. This can have one or so more algorithms added to it to aid the robot in locating the maze's end. The goal of the maze-solving algorithm is to identify the shortest path with the highest level of efficiency possible. A maze is a path or set of paths that usually leads from an entrance to a certain objective. The idea of a maze dates back around a thousand years. It was originally created in Egypt. Since then, numerous algorithms have been developed by mathematicians to clear the maze. These days, labyrinth troubleshooting is a crucial topic of robotics. The "Decision Making Algorithm" branch of robotics, which is crucial, follows the path of maze issues. Given that the robots would be placed in an unknowable location, it must be able to make precise decisions. There are many robots that can solve mazes using multiple algorithms. This project's maze-solving robotic device strategy includes obstacle-avoidance ultrasonic sensors, followed by sensors that will study the walls. This robot will use wall-following algorithms like the left- or right-hand rule to get out of the maze. Robot for solving mazes is independent. An autonomous robot is a kind of robotic that can carry out tasks without the assistance of humans. It is a little autonomous robot that can navigate a maze in the least amount of time possible, from the recognized beginning function to the maze's central area. A robot designed to solve mazes makes many

trips through them. First, it draws a map of the maze graph and stores it in memory. Then, it searches for the shortest route.

Case Studies

(i) Importance of the project:

The applications of maze solving robots have significant importance. A maze solving robot is capable of reaching the destination point faster. The shortest path algorithm we used in our project helps the robot to achieve that. This same application can be used for other projects as well. This is important to help ambulances, fire-fighters or rescuing robots to reach their destination on time. This system works much faster than the traditional methods.

(ii) Real Life examples:

As mentioned before, maze solving robots try to reach the destination using the shortest path algorithm. These applications are used mostly for navigation:

a) Rescue operations: A great example is the work of fire fighters. It is just as important as an ambulance. Fire fighters has to work fast in order to save many lives. They need to reach their destination within time in order to prevent more chaos than it already is. A split-second difference here can be vulnerable. By implementing the systems of maze solving robot in their fire truck, the fire fighters can avoid obstacles and reach destination by taking the shortest possible path.

b) Meet urgent requirements: For other urgent requirements in day-to-day life the applications of maze solving robot can be implemented. This will help anyone to reach their destination much faster hence save time.

c) Intelligent traffic control: The applications used in this robot can be used to implement an intelligent traffic control system. This will help fire fighters, rescuing teams and ambulances to accurately find the shortest path.

Medical Use

It is very important for an ambulance to reach its destination as fast as possible in order to save a patient's life. Using the applications of the maze solver robot an ambulance can implement the system and find the possible path in which it can reach the hospital faster. This can save a patient's life.

Industrial Use

The use of maze solver robot is huge in industries. Using the applications of the maze solving robot it is possible to deliver the industrial goods faster from one place to another. Industries are always shifting their goods and it is important that they reach the destination in time. These autonomous bots can be used as transporters to move in maze and use the algorithm to find its destination faster.

(iii) Assistive uses of the project in Bangladesh

As discussed before, we can use the applications of this project in the same criteria discussed above for Bangladesh. In many cases a country like Bangladesh is need of projects like this. Maze solver can be used as a traffic controller as well. The traffic of Bangladesh has always been a great challenge for us. Also in medical needs Bangladesh can be benefited by implementing these applications to ambulances. For rescue operations, to reach the destination faster vehicles can implement it for faster routes. Definitely a project like this is beneficial for the assistance of Bangladesh.

Project Compilation

i) Methodology

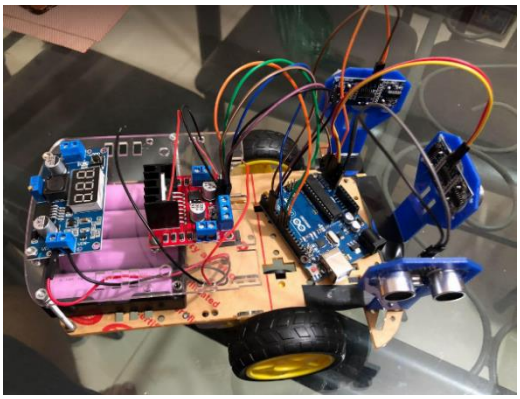

Figure 1 : Maze Solver Robot

At First, we took a plastic board which is designed for 'MAZE SOLVER ROBOT'. Then, using a screw and nut, we attached two motors to the plastic board. After connecting two wheels to the motor, we add a third wheel which is mainly used to change direction. Then we rotate the plastic board and then connect three sonar holders to the front side with glue after that, we install the sonar with the sonar holder. Then, using screws, we attach the Arduino Uno and battery holder to the plastic board. Then we install another half plastic board to it with two large nuts and screws and then we install buck converter and motor controller in upper plastic board.

Then we start connecting our motors to the motor controller using connecting wire and after that, we connect the buck converter to it, and then, we use male to female connecting wires to connect the motor controller and sonar to the Arduino Uno. Finally, we write a code and install it in the Arduino Uno.

Arduino Uno: A microcontroller board called Arduino UNO is based on the ATmega328P. It contains 6 analog inputs, a 16 MHz ceramic resonator, 14 digital input/output pins (six of which may be used as PWM outputs), a USB port, a power connector, an ICSP header, and a reset button. It comes with everything

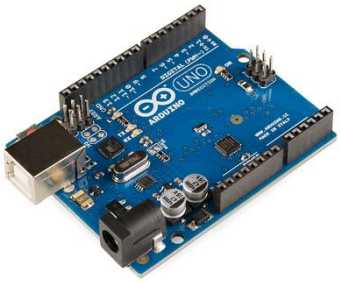

Figure 2: Arduino Uno

required to support the microcontroller; to get started, just use a USB cable to connect it to a computer or an AC-to-DC adapter or battery. It is very easy to use.

Figure 3: Buck Converter

Buck Converter: The buck converter is a quite simple type of DC-DC converter that produces an output voltage that is less than its input. The buck Converter is a particular kind of chopper circuit made to step-down convert the supplied dc input signal. In the case of buck converters, the fixed dc input signal is converted into a different, lower value, dc signal at the output. This indicates that it is built to have an output dc signal with a smaller magnitude than the applied input.

Figure 4: Sonar Sensor

Sonar Sensor: Sonar uses sound waves to detect and identify objects at a distance. In sonar, sound waves emitted by or reflected from the object are detected by the apparatus and analyzed for the information they contain, allowing for the identification of contours along their surfaces. Sonar and ultrasonic sensors fundamentally use the same technique.

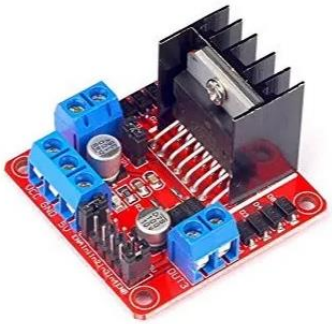

Figure 5: Motor Driver L298N

Motor Driver L2098N/L298N: For operating DC and Stepper Motors, the L298N Motor Driver Module is a high-power motor driver module. An L298 motor driver IC and a 78M05 5V regulator make up this module. Up to 4 DC motors or 2 DC motors with speed and direction control can be operated by the L298N Module.

ii) Flow Chart

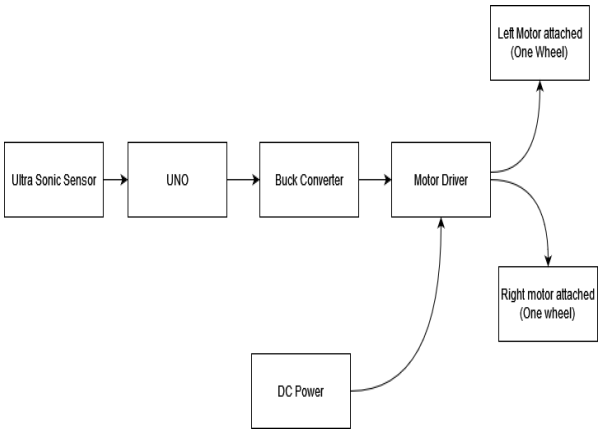

Figure 6: Flow chart of the project

iii) Discussion about the code, functions and libraries

As the program starts, the maze solver will start going forward. Until it finds an obstacle it will keep moving. If the robot is blocked or senses an obstacle it will turn left and start moving in that direction. If it comes across another obstacle the maze solver will then turn right and keep moving in that direction. However, If all sides are blocked, it will reverse it's direction and turn left and go forward. The system will repeat the same process.

pinMode(): This function has been used to configure a pin as an input or output. Pins 7,10 and 12 are used for Input and Pin 3,5,6,8,9,11,13 are used for output.

analogWrite(): The function is used to drive a motor at various speeds. It is also used for lighting LED's

delay(): This function has been used throughout the program. Delay function is used to pause the execution of a program for a specific time period.

delayMicroseconds(): It pauses the execution of the program for specific time in microseconds. The function was used for the three sonar sensors.

Some custom functions were created which included other functions in order to work with different sensors. sensor_front(), sensor_left(), and sensor_write() are made of other functions. Those are discussed below:

DigitalWrite(): This function is used to write a high or low value to a digital pin. For low mode it will be zero volt for High mode it will be set to the corresponding value.

For sensor_front() the digitalWrite mode is low on the pin 8, for sensor_left() digital write mode is low on pin 11, sensor_right() digital write mode is low on pin 13 which means that the corresponding values of these pins are zero.

In the created custom function pulseIn() function has also been used.

PulseIn(): This function allows us to block the program and wait for the signal to change from high to low with a timeout. The pin mode must be set as Input to work with the pulseIn function. Using HIGH or LOW mode in the pulseIn() function we can let the program know if it is close to the power supply or if the signal is close to zero.

Again, we used pin 7, 10 and 12 and set it to high respectively for sensor front, sensor_left() and sensor_right().

Function Stop_robot() is used to stop the bot when its closer to the wall or some block.

The function "reverse_robot()" is used to reverse the bot if left path is blocked it will reverse back and stop the bot if right path is open then it will go to the right track .

iv) Hardware Used

1.Arduino UNO -
2.Motor Driver L2098n-
4. Yellow body Kit (2 wheels)
5. Buck converter with display
6. Battery Holder Li-io (4 cells) -
7. Battery (li-Ion 3.7*4=14.8V) 18650 - 4*
8. Jumper wire Male to female
9. Sonar (3)
10. Sonar Holder (3)
11. Extra half board

v) Cost breakdown Table

Serial No.	Components	Price
1	Arduino UNO	950
2	Motor Driver L2098n	150
3	Yellow body Kit (2 wheels)	550
4	Buck converter with display	230
5	6. Battery Holder Li-io (4 cells)	50
6	Battery (li-Ion 3.7*4=14.8V) 18650 - 4*80	320
7	Jumper wire Male to female	70
8	Sonar (90*3)	270
9	Sonar Holder (40*3)	120
10	Li-ion charger	250
		Total Cost: 2960 BDT

Cost Analysis

The overall fetched of this extend is 2960 takas. It is exceptionally cheap cost for a robot. The cost of Arduino Uno is 950taka. But the circuit may be more complex and require outside fetched for run that chip. Ultrasonic sonar (HR-SR04) has been utilized in this extend. Per sensor taken a toll is as it were 90taka. But other sensors are more costly. DC engines 300rpm with wheels taken a toll as it were 550taka. 900mAh lipo batteries are utilized in this venture. It takes as it were 320 takas.

Possible future projections and recommendations

We wanted to build a robot that could move parallel to a wall at any distance for Tech Fest 2022. We concentrated on developing various algorithms and optimizing for the best results while increasing the robot's speed as much as possible based on test results. For this project, we had to put control system theory into practice, which led us to create the wall-following robot. The main goal of the robot, as the name implies, is to follow the wall within a set distance. We used Arduino to implement the PID controller. The PID controller is used to keep a certain distance from the wall. PID controller stands for Proportional, Integral, and Derivative Controller and is used to calculate and attempt to minimize the error between the measured and desired set points.

This robot still has some limitations, such as the fact that it can only follow an obstacle with a small change in angle. Because it can only read angles perpendicular to the robot, this is due to hardware limitations. When there is a significant change in the angle of the obstacle, the robot is unable to detect distance, resulting in an error.

To provide a wider field of view, the robot can be outfitted with more ultrasonic sensors.

The robot can currently only follow one side of the obstacle. To make it an autonomous car, we must ensure that both sides of the robot are clear of obstacles. As a result, the solution is to install ultrasonic sensors on both sides of the robot.

As a result, we can ensure that the robot is always in the middle of two obstacles, avoiding collisions.

Aside from that, the robot will only stop when an obstacle in front of it approaches a predetermined distance. To improve the robot even further, we can implement speed increments or decrement based on the obstacle in front of it. If there is no object in front, the robot will accelerate; if an object is detected, the robot will decelerate based on the distance between the object and the robot.

Conclusion

By applying maze-solving algorithms, this study was able to automate a robot's ability to navigate mazes. Finding the shortest path can be done with high efficiency by using a shortest path algorithm. Wall following algorithm was the initial algorithm. The fundamental approach yields an effective solution to the maze. But because it lacked self-intelligence, it was unable to navigate the maze quickly. And it was unable to clear the loop maze. Therefore, the flood fill algorithm approach has been employed as an effective way to discover the shortest path. A genuine maze was used to train the robot after every technique had been tried. To guarantee the robot's optimal performance, numerous tests have been carried out.

This study advances a variety of crucial robotics knowledge, including understanding of numerous decision-making systems. Learning about other electronics components, such as motor drivers, sensors, etc. is also beneficial. Future work will be significantly impacted by the knowledge obtained.

We'll aim to make the system capable of loop detection in upcoming work. Robots will be able to complete practically all maze characteristics with this modification. Following that, this robot can be used to a variety of uses. As an illustration, this robot may be fitted to a car's autopilot system or used to locate ruins in an underground mine.

References

1. *A comprehensive and comparative study of Maze-solving techniques by ...* (no date). Available at: https://www.researchgate.net/publication/224202469_A_Comprehensive_and_Comparative_Study_of_Maze-Solving_Techniques_by_Implementing_Graph_Theory (Accessed: November 30, 2022).

2.*panelRahulKumarPeniJitokoSumeetKumarKrishneelPillayPratishPrakashAsneetSagarRamSinghUtkalMehtaPersonEnvelope, A.links* open overlay *et al.* (2017) *Maze solving robot with automated obstacle avoidance, Procedia Computer Science*. Elsevier. Available at: https://www.sciencedirect.com/science/article/pii/S1877050917302107 (Accessed: November 30, 2022).

3.*What are the benefits of making a maze solving robot? - quora* (no date). Available at: https://www.quora.com/What-are-the-benefits-of-making-a-maze-solving-robot (Accessed: November 30, 2022).

4.Mjrovai and Instructables (2017) *Maze solver robot, using artificial intelligence with Arduino, Instructables*. Instructables. Available at: https://www.instructables.com/Maze-Solver-Robot-Using-Artificial-Intelligence-Wi/ (Accessed: November 30, 2022).

5. *Maze Solving Robot (3 IR sensors)* (no date) *Arduino Project Hub*. Available at: https://create.arduino.cc/projecthub/Varun2905/maze-solving-robot-3-ir-sensors-9ada3b (Accessed: November 30, 2022).

6. *(PDF) Autonomous Maze Solving Robot - Researchgate* (no date). Available at: https://www.researchgate.net/publication/342614026_Autonomous_Maze_Solving_Robot (Accessed: November 30, 2022).