Solving a Maze using the Wall Follower Algorithm

William Zhang
Department of Electrical and
Computer Engineering
University of Central Florida
Orlando, Florida 32816
Email: WilliamZhang@knights.ucf.edu

***Abstract*—There are numerous approaches to solving a maze. To autonomously navigate through a maze, the robot must be able to sense its environment. The robot must also be able to proceed with a course of action from its initial position. Mazes can be designed in a number of ways that allow for some approaches to be better than others in terms of efficiency and memory usage. This paper will analyze the use of the Left-hand Rule wall follower algorithm in approaching three different maze designs.**

General Terms
Autonomous, navigation, robotics

Keywords
Wall follower, maze solving, algorithm

I. Introduction

Mazes are a great puzzle game. They can be used to develop and/or sharpen problem-solving skills. Depending on how the maze is arranged, the robot may need to plan how to go about reaching the exit. Several methods have been developed that will allow the robot to solve the maze [1]. In this paper, the implementation will use the wall follower algorithm to demostrate one way to solve a simple maze.

The wall follower algorithm is a simple plan of action when the robot does not know the layout of the maze in the beginning. The robot simply chooses one side of the wall and follows it. The robot does not store any of its previous locations in memory. The robot should be able to reach the exit of the maze by following one side of the wall. By using this algorithm, the robot should be able to autonomous navigate the maze and reach the exit without any supervision.

In this implementation, the robot will try to solve a maze that is simply-connected [2]. The maze will contain no loops. The maze should not have any walls that are isolated from the rest. The walls must be connected together or to the maze's outer boundary. In order to sucessfully solve the maze, the robot must exit the maze. The exit must be on the perimeter of the maze. The exit cannot be within the area of the maze.

II. Method

The wall follower algorithm is a simple approach to solving some mazes. The robot uses the walls of the maze to traverse the path to the exit. Assuming that the maze is simply-connected and the exit is on the perimeter of the maze, the wall follower algorithm can gaurantee that the robot will find one solution to the maze. The solution may not be the most efficient solution in terms of steps taken to reach the exit from the starting location.

There are two variations of the wall follower algorithm. This paper will utilize the Left-hand Rule approach, rather than the Right-hand Rule approach. The Left-hand Rule approach utilizes the left-hand side of the robot to traverse the maze, while the Right-hand Rule utilizes the right-hand side of the robot to traverse the maze. Both approaches follow the same idea of following a wall from the start until the exit can be reached.

A. Left-hand Rule

The robot places its "left-hand" onto the wall of its left-hand side. The robot begins to walk forward. When the robot is able to move to its left, it turns left and moves in that direction. When there is a wall in front of the robot and the robot is unable to move left, the robot checks if it can move right. If the robot is able to move to its right, it makes a right turn and begins to move in that direction. When the robot is unable to move to its left, forward, or right, the robot asks if it has reached the exit. If it has not reached the exit, the robot can assume it is in a dead-end and turns around. If the robot has reached the exit, the robot has sucessfully solved the maze. Figure 1 shows a flowchart for the wall follower algorithm using the left-hand rule navigation.

The Left-hand Rule can be summaized as follows:

- The Robot uses its "left-hand"
- It puts its hand on the wall on its left
- Starts walking
- Never lets go of its left hand

In order to test the wall follower algorithm implementation, a maze must be created. The maze illustrates the obstacles the robot must sucessfully overcome in order to reach the exit. If the robot is able to keep following a wall, the robot should be able to solve the maze.

Three mazes were designed to showcase the operation of the Left-hand Rule wall follower algorithm. Maze One, in Figure

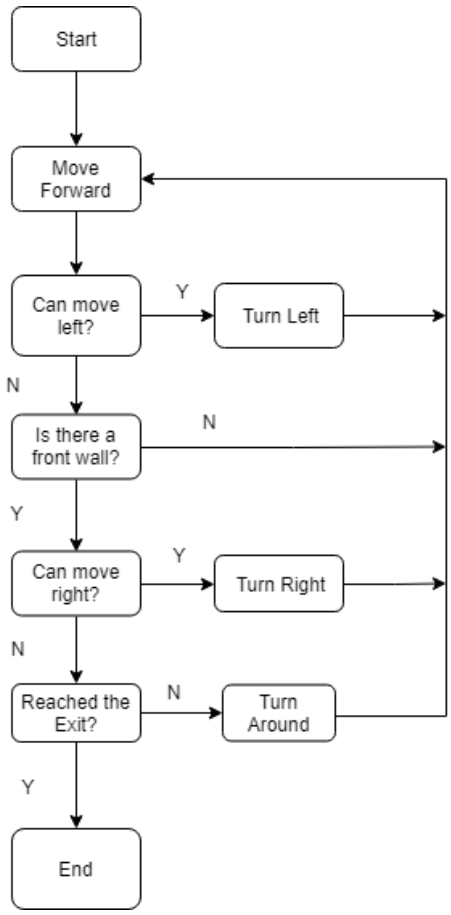

Fig. 1. Flowchart for Left-hand Rule wall follower algorithm

2, is a single path from the start to the exit. Maze Two, in Figure 4, is a more complicated maze for the robot to solve. Maze Three, in Figure 6, is an open maze. Each of the mazes were designed to test the advantages and disadvantages of the Left-hand Rule wall follower algorithm.

B. Maze One Design

For this maze, the robot just needs to follow the path in order to reach exit. If the robot is able to follow the path, the robot will successfully solve the maze. There are turns that the robot must complete. There are long narrow path that the robot must abide to in order safely reach the exit. As long as the robot follows the path, it will reach the exit.

C. Maze One Traversal

Figure 3 illustrates the trace that the Left-hand Rule wall follower algorithm is using to traverse the maze. As shown, the trace has successfully followed the long narrow path and made the appropriate turns in order to reach the exit. The algorithm is able to solve the maze. The Left-hand Rule wall follower algorithm did perform as expected. However, other algorithms can accomplish the exact same trace of the solution.

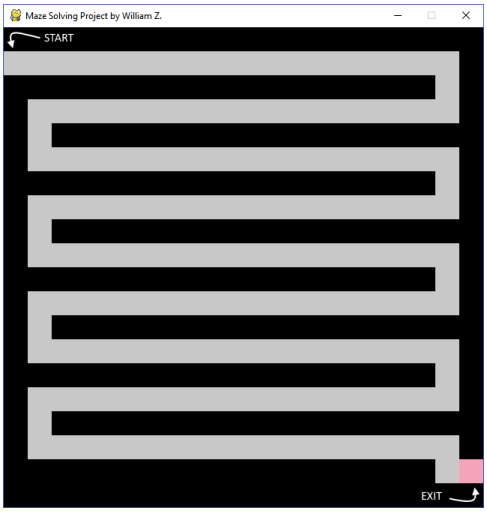

Fig. 2. Initial State of Maze One

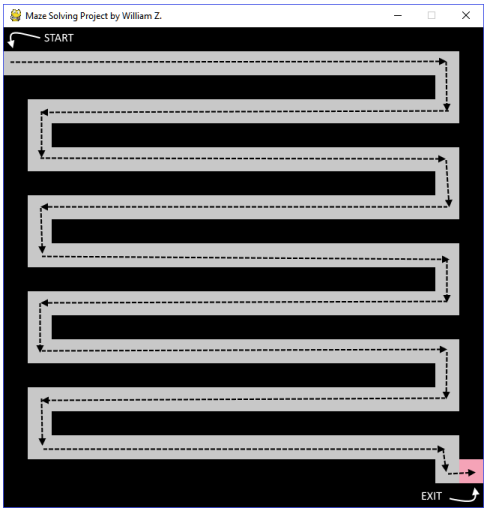

Fig. 3. Trace of the Left-hand Rule wall follower navigation

D. Maze Two Design

The design of Maze Two is different from Maze One. Maze Two is designed to showcase the potential of the Left-hand Rule wall follower algorithm in terms of its ability to follow a wall. There are many paths to reach the exit for this maze. The Left-hand Rule wall follower algorithm is able to reach the exit using one path. The path is not the most efficient path to the exit, but the algorithm is able to reach the path.

E. Maze Two Traversal

The traversal of Maze Two is more complicated than Maze One. For Maze Two, the robot must be able to sense whether it is able to move left, forward, or right in order to keep following the wall. There is also a dead-end in the path of the robot. The robot must be able to turn around for it to get out of the dead-end. The robot must also not get confused about the wall on its right-hand side. The Left-hand Rule wall follower algorithm strictly forces the robot to only follow one side of

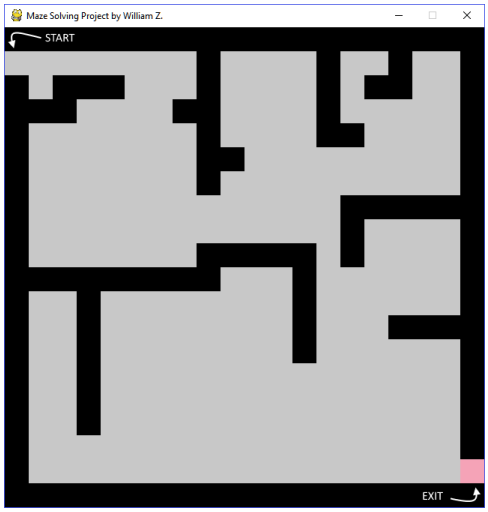

Fig. 4. Initial State of Maze Two

Fig. 6. Initial State of Maze Three

the wall. The robot should not switch which side of the wall it is following. Figure 5 illustrates the trace that the Left-hand Rule wall follower algorithm is using to traverse the maze.

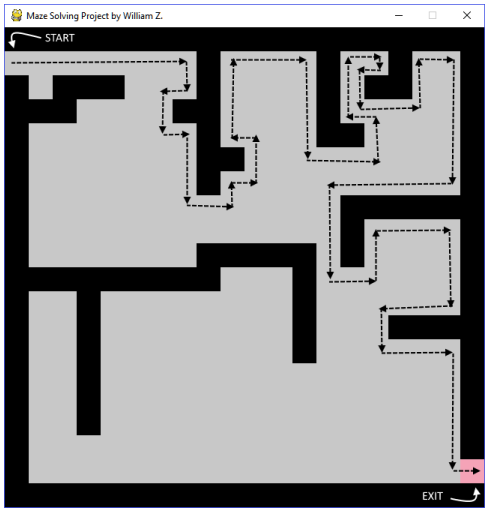

Fig. 5. Trace of the Left-hand Rule wall follower navigation

F. Maze Three Design

In Figure 6, the design of Maze Three is simply an open area. There is no obstacles for the robot to avoid. The robot must simply reach the exit in any path it chooses to do so. There are a number of possible solutions that the robot can use to solve the maze. This maze simply judges how well the algorithm is able to handle a maze with no interior walls.

G. Maze Three Traversal

Using the Left-hand Rule wall follower algorithm, the robot simply follows the wall on its left-hand side and begins moving. The robot does not stray away from the wall. It simply follows the algorithm. The path it chose did lead it to the exit. Figure 7 illustrates the trace that the Left-hand Rule wall follower algorithm is using to traverse the maze.

The most efficient path from the start to the exit is simply a straight path. The Left-hand Rule wall follower algorithm was unable to figure this out because the algorithm was made to follow the wall. This maze proved the short-comings of the Left-hand Rule wall follower algorithm in terms of efficiency.

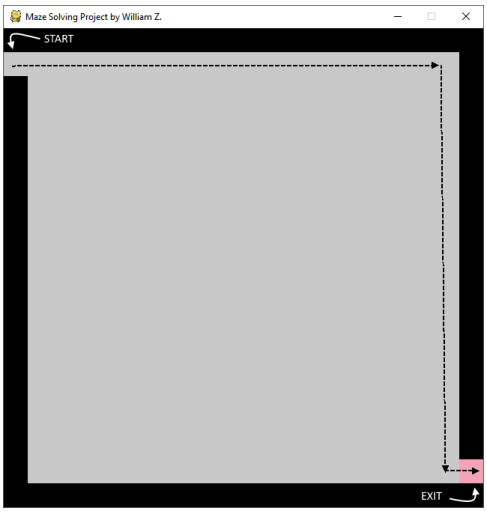

Fig. 7. Trace of the Left-hand Rule wall follower navigation

III. Discussion

The robot has successfully solved the mazes using the Left-hand Rule wall follower algorithm. The algorithm may not lead the robot to the most efficient path, but it does lead the robot to a path that the robot can take to arrive to the exit. Without storing any memory of the maze, the robot is able to traverse the maze. The algorithm is efficient in terms of space-complexitiy. But in terms of run-time complexity, the

algorithm is not efficient in regards to other maze solving algorithms, like the A* (A-Star) search algorithm.

A. Solving the Maze

The Left-hand Rule wall follower algorithm is able to solve the three mazes. But the algorithm does not gaurantee the fastest approach to solving the maze. But because the robot does not know how the maze is designed beforehand, the Left-hand Rule wall follower algorithm is a simple algorithm for the robot to find a path from start to exit. Even though other solutions may be possible, the Left-hand Rule wall follower algorithm does accomplish the task of solving the maze and without using any memory at all.

B. Memory

The Left-hand Rule wall follower algorithm does not rely on what the robot has previous seen. The robot only utilizes the direction what where it is heading and the choice of wall to follow. The robot does not store any information of the maze. It simply senses the environment and acts. There is no planning involved.

C. Maze Design

If the maze is designed under the circumstances that the exit is on the outer perimeter and the maze is simply-connected, a solution to the maze can be found from the algorithm. The robot can only find the exit if it is located close to the walls. The algorithm may not give the fastest approach to the exit from the start, but it does enable the robot to reach the exit, assuming that the maze is designed with the algorithm in mind.

D. Efficiency

Other algorithms are able to find the most efficient path from the start to the exit. These algorithms require memory of the maze in order to plan which path to undergo. This process of planning enables algorithms like the A* search algorithm to efficently traverse the maze in the small amount of steps taken.

E. Issues with the Left-hand Rule Wall Follower Algorithm

The Left-hand Rule wall follower algorithm only works on limited maze designs. If the exit is not located near the walls on the perimeter, the robot will not be able to reach the exit. The robot will simply follow the wall from the start and loop back to the start. The algorithm is dependent upon the design of the maze.

Only one solution can be found from the algorithm. The algorithm will not be able to tell how many possible solutions there are from the start to the exit. The algorithm simply follows the wall until an exit has been reached. If there are multiple exits or multiple paths to the those exits, the algorithm will only find a single solution.

F. Improvements on the Robot's Ability to Navigate

To improve the robot's ability to traverse the maze, several considerations should be given to the approach. The robot should plan on how to act within the maze with more intelligence. The Left-hand Rule wall follower algorithm is a simple algorithm enabling any robot to traverse the maze and find an exit. But, the algorithm lacks in its ability to the find the most efficient path.

The addition of more sensors for the robot to use can enable the robot to get a better understanding of its environment. The use of vision and proximity sensors can enable the robot to traverse the maze with greater efficiency.

IV. Conclusion

Several ways to improve the operation of the robot include the use of additional sensors and improved artifical intelligence. Instead of only relying upon use the of the robot's "left hand," the robot could utilize additional resources that allow it to get a better understanding of its environment. The maze solving algorithm should be refined to allow the robot more intelligence. Ways to refine the algorithm is to include the use of memory. Allowing the robot the ability to an overview of the layout of the maze will enable the robot to better traverse the maze. Rather than going in blind, the robot will be better able to traverse the maze.

Some interesting maze designs should be used to fully test the algorithm's performance in solving the maze. There are a variety of mazes that can be used. The use of 3D mazes can pose interesting responses for the algorithm's approach to solving the maze. Mazes with different routing paths from the start to the exit can entice interesting responses from the algorithm.

Even though the wall follower algorithm is not guaranteed to work for all mazes, it does allow the robot to find a solution of a maze given certain limitations. The algorithm focuses on the robot that is traversing the maze. It does not require to store any information of what the robot has seen beforehand. The robot should be able to follow a wall from start to the exit. The algorithm is a faster algorithm than a robot taking random directions into the maze. However, the use of memory could enable the robot to traverse the maze with a much faster solution.

References

[1] W. D. Pullen, Think Labyrinth: Maze Algorithms. [Online]. Available: http://www.astrolog.org/labyrnth/algrithm.htm. [Accessed: 28-Apr-2018].

[2] Schaaf, William L., Number game. [Online]. https://www.britannica.com/topic/number-game. [Accessed: 28-Apr-2018].

[3] Robomind, RoboMind Maze challenge. [Online]. http://www.robomind.net/downloads/Maze

[4] "About - wiki," About - pygame wiki. [Online]. Available: https://www.pygame.org/wiki/about. [Accessed: 28-Apr-2018].