Autonomous Maze Solving Robotics: Algorithms and Systems

Shatha Alamri[1], Shuruq Alshehri[1], Wejdan Alshehri[1], Hadeel Alamri[1], Ahad Alaklabi[1], and Tareq Alhmiedat [1,2]
[1] Faculty of Computers & Information Technology, University of Tabuk, Tabuk, Saudi Arabia
[2] Industrial Innovation & Robotics Center, University of Tabuk, Tabuk, Saudi Arabai
Email: {salaamrei, s_alshehri, w.alshehri, h.alamri, a.alaklabi}@ut.edu.sa, t.alhmiedat@ut.edu.sa

***Abstract*—In robotics, autonomous movement is an important feature that enables the robot to move independently from one location to another. Autonomous movement within an unknown area requires the robot to carry out investigations. The concept of solving a maze has an important place in the field of robotics, and is based on one of the most important areas of robotics, the Decision-Making Algorithm. In this paper, we discuss and analyse existing maze solving algorithms, and investigate the recent development of autonomous maze solving robotic systems. In addition, the work presented in this paper guides the researcher and developer for choosing an adequate maze solving algorithm to develop an efficient maze solving robotic system for a certain scenario.**

***Index Terms*—maze, autonomous robot, maze solving, solver robot, maze solving algorithms**

I. Introduction

Autonomous mobile robot navigation plays a critical role in diverse applications, including: warehouse robots, self-driving vehicles, smart wheelchairs and personal assistant robots. In many situations, autonomous robots are the best option for various missions. Autonomous navigation is a critical task in mobile robotics because it enables the mobile robot to independently perform movement tasks to get from one point to another. Autonomous mobile robot is an intelligent vehicle that is capable of travelling through several locations (point of interest), either following a predefined trajectory, or navigate itself in a specific area. In both cases, the mobile robot has to avoid obstacles and move forward to destination point [1].

One interesting topic in robotics is autonomous maze solving, where an autonomous robot tries to solve a maze in the shortest time possible in the most efficient way. The key mission of an autonomous maze solving robot is to reach a target location by navigating through a maze area. Maze solving robot is one of the most popular independent robots, where the solver robot is self-reliant and can move through a maze from the source-point to the destination-point. It follows the shortest route and take the least time possible. To achieve this goal, a solver robot tests several paths to create a map of the maze, and stores paths and routes and then runs the autonomous robot through the maze area along the most efficient route.

The solver robot knows the source-point and the destination-point in the maze area, but it does not have any information about the maze area structure or obstacles between the two locations. The implementation of autonomous vehicles ranges from employing robots to carry goods through factories, office buildings and other workspaces to using robots in dangerous situations or difficult to reach locations such as bomb sniffing, finding humans in wreckage, fire-fighting, in emergency situations [2].

Recently, various autonomous maze solving algorithms and techniques have been developed for this purpose, each one with its own advantages and shortcomings. The existing maze solving algorithms and systems have been reviewed in [3-5]. However, the work presented in this paper is different because the recent deployment of maze solving algorithms and systems are considered. Moreover, we discuss and analyse the performance of existing autonomous maze solving robotic algorithms and systems. The main contributions of this work lies on the following aspects:

1. Research and categorize the existing maze solving algorithms.
2. Research the recent developed maze solving robotic systems.
3. Evaluate the efficiency for the recent developed maze solving robotic systems.

This paper is divided into six sections. Section 2 presents a discussion of the potential applications for autonomous maze solving robotic systems. A classification and analysis of the existing maze solving algorithms are presented in Section 3. In Section 4, the existing autonomous maze solving robotic systems are presented and discussed. Section 5 presents a discussion and analysis of the existing autonomous maze solving robotic algorithms and systems. And finally, Section 6 draws conclusions.

II. Robotic Applications

The possible applications for maze solving vehicles range from simple tasks such as transferring goods through factories, office buildings, classrooms and other

Manuscript received February 17, 2021; revised April 12, 2021.

doi: 10.18178/ijmerr.10.12.668-675

workspaces, to hazardous tasks in difficult to reach areas like evacuating people from dangerous buildings, bomb sniffing, etc. Autonomous maze solving robotic systems can be applied to:

- Manufacturing: robots can be used to transport items and tools from one location to another in a fast and accurate manner across a complicated terrain, where paths must first be explored to accomplish the delivery. The Flexible Manufacturing System (FMS) that employs autonomous robots are currently dominated by the use of Automatic Guided Vehicle (AGV) [6, 7].
- Home automation: domestic robots are designed to assist human with tasks including: lawn moving, vacuum cleaning, and home monitoring. A mobile robot can be used as a vacuum cleaner, navigating itself effectively around the house whilst simultaneously cleaning [8].
- Traffic control: this is a real example of maze solving technology that enables fire fighters or paramedics to find the best route to an emergency [9]. Therefore, appropriate traffic control is a critical issue in highway work zone safety, in order to control the robotic system to safely travel from one location to another one [10].
- Rescue operations: rescue missions usually start with a search in the unknown environment before reaching the region where victims can be located. As rescuers progress along a particular path, therefore, they are required to report their location to the mission headquarters. This will help the rescuer to reach the final destination. Mobile robots have been employed widely in the search and rescue operations, such as searching victims in dangerous areas which is harmful for human, as to offer observation data for map building [11, 12].

III. Maze Solving Algorithms

In any maze solving system, the first stage is to compile a maze solving algorithm. This section discusses existing maze solving algorithms which may be employed in an autonomous maze solving robotic system. There are various maze solving algorithms which aim to find the path between the source-point and the destination-point. In this section, we classify the maze solving algorithms into two categories according to the deployment of the solver robotic system. The categories are as follows: known environment algorithms (solver robot has a prior knowledge about the maze area), unknown environment algorithms (solver robot solves the maze with no prior knowledge of the maze area). Fig. 1 shows the classification of maze solving algorithms.

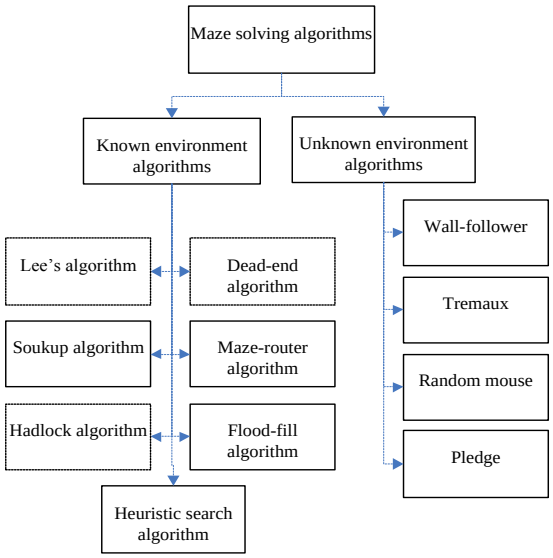

Figure 1. Classification of maze solving algorithms

A. Known-Maze Enviroments-Based Maze Solving Algorithms

This section describes the maze solving algorithms which can be deployed with known maze environments, where the solver robot can see the whole maze at once.

1) Dead-end filling algorithm

Dead-end algorithm works well in known mazes, where it fills all dead ends, leaving only the correct paths unfilled. By employing this method, the solver robot first finds all the dead ends in the maze and then fills in the path from each dead end up to the first junction. This algorithm scans the maze in an empty order, finds the dead ends and closes the associated paths [13].

2) Maze routing algorithm

Maze routing algorithm finds the possible paths between any two points in the maze area. This algorithm only works in known mazes where the maze area is already recognized and stored. The maze routing algorithm is able to explore the possible paths between the source and destination points and identify the shortest path between them [14].

3) Flood-fill algorithm

Flood-fill algorithm assigns a distance value between any point (intersection) and the centre-point (destination). The flood-fill algorithm floods the maze when the robot reaches a new node [15].

4) Lee's algorithm

The Lee's algorithm offers a simple solution for maze routing problems based on breadth first search strategy, and it is able to explore the possible paths between any two points if exit, and guarantees the minimum path. Through the Lee's algorithm, (a) an initial point is added to the queue, (b) then valid neighbouring cells are added to the queue, (c) next the queue element is removed from the queue and continue to the next element. The steps (a)-(c) are repeated till the queue is empty [16].

5) Soukup algorithm

Soukup algorithm combines both the breadth first and depth first search algorithms. Through the Soukup

algorithm, the line search is first conducted toward the destination-point, and then an expansion is processed to 'bubble' around the wall as soon as the line search detects a wall. The Soukup algorithm is 10-50 times faster than the Lee's algorithm [17].

6) Hadlock algorithm

Hadlock algorithm is a shortest path algorithm for grid graphs. Hadlock algorithm is an enhancement of Lee's algorithm which reduces the processing time through the expansion phase by biasing the search in the target's path. The Hadlock algorithm uses a value known as detour value, which is the path from the source to a grid-point that reveals the number of grids that this path has detoured away from the target-point. Hadlock algorithm guarantees that a shortest-path connection will be found if exist [18].

7) Heuristic search algorithm

The heuristic search algorithm is based on the concept of 'greedy best first' search, which is like the breadth-first search. The heuristic search algorithm explores multiple paths in parallel, and the best-first focuses on paths which are closest to the goal. The distance from the goal serves as a heuristic to direct the search. The major drawback for the heuristic algorithm is its space complexity, since it stores all generated nodes in memory.

B. Unknown-Maze Enviroments-Based Maze Solving Algorithms

The following algorithms can be deployed in unknown environments where the solver robot has no prior knowledge of the maze area. These algorithms include wall follower, Tremaux's algorithm, random mouse and Pledge.

1) Wall-follower algorithm

The wall follower is the most common maze solving algorithm, where its main idea is to follow walls in the maze area. The solver robot observes the right or left wall and moves throughout the maze area until it finds the way out. There are two possible rules in the wall follower, the left-hand rule and right-hand rule. The turning priority will be either to the left or to the right depending on the rule chosen [19, 20].

2) Tremaux's algorithm

This algorithm requires drawing lines on the floor to mark a path. It is guaranteed to work with all mazes that have well defined passages. Tremaux's algorithm works according to the following rules: if a junction that has no marks (unvisited), then choose an arbitrary unmarked path, follow it and then mark it as visited. If a junction has one mark, turn around and return along that path, marking it a second time as visited. This situation can occur when the robot reaches a dead end. If a junction has more than one mark, arbitrarily choose one of the remaining paths with the fewest marks, follow that path and mark it as visited. However, when the robot finally reaches the destination, paths marked just once will indicate a way back to the start [21, 22].

3) Random mouse algorithm

Random mouse algorithm can be implemented using any robot, and works in the same way as a mouse would solve a maze. The solver robot travels around the maze aiming to find the destination point. A solver robot follows a straight line until an obstruction is reached, where there is a choice of more than one path (intersection-point). At this point, the robot makes a random decision as to which path it takes. It continues straight ahead until it comes to the next intersection. At each intersection, the solver robot randomly turns right or left, however, it never goes back along the path it came from. This method can be implemented by an unintelligent robot [23].

4) Pledge algorithm

The wall follower algorithm is a part of the Pledge algorithm solution. The Pledge algorithm is a maze traversing algorithm used when the walls are disjointed and there are obstacles present. The robot keeps the obstacle either on its left or right-hand side, and keeps track of all turns using a counter, where a right-turn increases the counter and a left-turn decreases the counter. When the counter reaches zero, the solver robot has traversed the obstacle and it continues on its path for the right-hand side algorithm. The solver robot always begins by turning left, then for every movement it applies the following logic: if there is no obstacle on the right, it moves right; if there is an obstacle on the right, but no obstacle in front, it moves forward; if there are obstacles on the right and in front, then it turns left. The solver robot will continue this way, counting each turn until the counter reaches zero again [24, 25].

The above algorithms guarantee to solve different types of maze with no prior knowledge of the maze environment. However, whenever multiple paths exist these algorithms do not offer the shortest path possible between the source and destination points.

IV. Autonomous Maze Solving Robotic Systems

Various autonomous maze solving robotic systems have been proposed recently. Each system employs one of the maze solving algorithms discussed earlier. In this section, we discuss the existing autonomous maze solving robotic systems. In addition, we categorize the existing systems into two categories based on the navigation method employed: camera-based systems and sensor-based systems, as presented in Fig. 2.

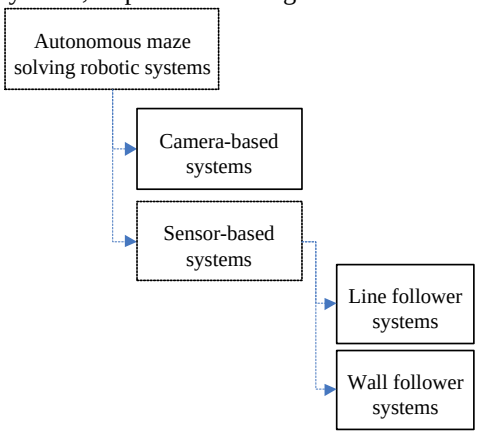

Figure 2. Classification of autonomous maze solving robotic systems

A. Camera-Based Systems

The image-processing-based solving robot systems require capturing the entire image of the maze area using a digital camera in order to analyse and process the maze's paths, determine the possible paths between the source and destination points, and identify shortest path possible. M. Aqel et al. [26] proposed a maze solving robotic system which is based on image processing technique and artificial intelligence algorithm. The entire image of the maze is captured using a web camera and then the maze solving process is performed completely outside the maze by a computer.

The work presented in [27] includes the design and development of a maze solving robotic system based on image processing and path finding techniques. O. Kathe et al. revealed that the proposed system works faster because the maze's data are acquired beforehand rather than travelling through the maze cell by cell. A. Chandak et al. [28] proposed a wave-front based algorithm to create a path for a robot to travel through an indoor environment. The 4-points and 8-points connectivity-based algorithms are presented for the purpose of path generation.

B. Rahnama et al. [29] introduced a maze solving algorithm which avoids the robot having to go through a long navigation process. This algorithm is based on image processing and shortest path algorithms. The proposed system works efficiently because of the pre-processing of the maze image data rather than going through the maze cell by cell. In reference [30], N. Barnouti et al. employed A* search algorithm to investigate the shortest path between the source and destination points through images that represent a map or a maze. The proposed algorithm was tested through different maze environments.

The work presented in [31] includes a wireless navigation mobile robotic system for path planning and trajectory execution within an indoor maze environment. The proposed system is based on a camera in order to capture live images for the mobile robot within the maze area. K. Lutvica et al. have developed an image processing and analysis algorithm which is able to determine the robot's position and orientation based on colour markers recognition. The results confirmed the robustness and effectiveness of the implemented robotic system.

B. Sensor-Based Systems

Existing sensor-based maze solving robotic systems can be classified into two sub-categories: rangefinder and line-follower. The former employs rangefinder technologies (such as infrared and ultrasonic sensors) to measure the distance between the solver robot and the facing wall, whereas the later employs colour (light intensity) sensors to follow the line drawn on the floor in order to explore the correct path to the destination points.

The rangefinder systems are discussed first. The work presented in reference [32] provides experimental and simulated results for an efficient maze solving algorithm which provides locally optimized path choices while maintaining the integrity of the potentials. The designed robotic system consists of five infrared proximity detectors which detect the presence of the walls. I. Elshamarka, & S. Abu Bakar [33] presented a maze solving robot algorithm designed to solve a maze based on the flood-fill algorithm. The algorithm consists of four main stages: Mapping, Flooding, Updating and Turning. Three ultrasonic sensors are deployed in order to estimate the distance to the walls surrounding the robotic system.

I. Arya et al. [34] presented an autonomous robotic system to navigate the unknown terrain and then explore the maze using multiple experimental mapping and navigation algorithms based on ultrasonic sensors. This work [35] introduced the concept of robot maze solving using the virtual maze environment. F. Annaz. examined the performance of various types of robots within various board algorithms, where two maze solving algorithms were implemented to solve mazes generated by users: a primary navigation algorithm for systematic discovery and a proximity algorithm for optimized rescue.

The work presented in [36] includes the design and development of an autonomous navigation robot, which is based on the Left Straight Right Back (LSRB) algorithm and the Right Straight Left Back (RSLB) algorithm. The developed robotic system employs infrared and ultrasonic sensors for navigation. I. Elshamarka & A. Saman [37] designed a maze solving robotic system using the flood-fill algorithm. The designed solver robot was able to detect walls using ultrasonic rangefinder sensors and was able to learn the maze, find all possible paths and solve the maze using the shortest path possible.

The work presented in [38] includes the design and development of an intelligent robotic system that is able to solve the maze using the shortest possible distance, where the maze area is divided into cells. The robot first navigates the area of interest and then finds the possible paths between the source-point and the destination-point. J. Su et al. [39] developed a half-size micro-mouse for international contests and mobile robot education. The J. Su et al. developed two new algorithms for enhancing the resolution of position and velocity estimations and a time-based diagonal maze solver.

C. Wu et al. [40] proposed a modified route-searching algorithm to find all potential paths in an unknown maze area. The maze area is thought of as divided into grids, where each grid has a maximum of four directions. In addition, the C. Wu et al. presented a modified searching algorithm based on the traditional depth-first graph traversal method.

In [41], R. Kumar et al. proposed an autonomous maze solving robot with independent mapping and localization operations, based on deploying three infrared sensors. The designed robotic system was tested and solved out the maze successfully without any interruption with walls and objects. On the other hand, a wall follower-based maze solving robotic system is presented in reference [42]. The mobile robot was capable of solving the maze and calculating the total distance travelled to reach the

final destination. The mobile robot was equipped with ultrasonic sensors in order to detect the presence of walls.

The work presented in [43] includes the design and development of an autonomous solver robot system which combines the Pledge algorithm with the Flood-fill algorithm, named as MazeRobot, and consists of ultrasonic range-finder sensors to detect the presence of walls and wheel rotation decoders to estimate the travelled distance.

L. Bienias et al. [44] proposed a maze exploration algorithm that consists of two main phases: the exploration of the whole maze area where all possible paths are determined, and then the runtime process. The proposed system integrates two maze solving algorithms: Wall-follower, and Tremaux's algorithms, and real experiments were conducted using Arduino-robot platform with proximity and rotation sensors.

The rest of this section discusses the line follower systems. Norbert-Brendan, K. & Marius, T.C. [45] proposed a line follower-based maze solving robotic system based on the left-hand rule and dead-end filling algorithms. The designed robotic system was able to offer a reasonable solution. In [46], S. Sakib et al. proposed a line follower algorithm for exploring and solving any kind of maze. The proposed maze mapping system is based on a coordinate system where the paths are calculated using the Dijkstra algorithm.

The work presented in [47] includes the design and development of a line maze solver robot based on a depth-first search algorithm. The robot was able to solve out a non-loop maze in an average time of 84.97 seconds and a loop-maze in an average of 63.40 seconds. On the other hand, R. Musridho et al. [48] proposed a line maze solving algorithm to explore and solve a maze area made up of curved and zigzag lines. The new algorithm successfully solved the maze with a curved and zigzag structure.

A. Khan et al. [49] proposed an autonomous maze solving robot with turning indicators and the ability to navigate itself through any environment. The work presented in [50] includes a simple autonomous robotic system (Lego Ev3) with limited on-board resources to employ a line follower maze solving algorithm.

V. Discussion

For any solver robot, the navigation process is the main task that intends to explore the possible paths between the source and destination points, and may then employ routing algorithms to find out the shortest path between any two points [51]. In this paper, the maze solving algorithms have been divided into two categories: algorithms for known environments and algorithms for unknown environments. This section analyses the existing maze solving algorithms and evaluates the existing solver robot systems.

The dead-end filling algorithm [13] finds all the dead ends in the maze area in order to explore the path from the source-point to destination-point. However, the dead-end algorithm is inefficient in robotics, since the robot has to look at the entire maze at once. On the other hand, the flood-fill algorithm is able to explore the path between the source-point and destination-point with no requirements or pre-assumptions, in addition, the flood-fill algorithm finds the shortest path to the centre of the maze. However, the flood-fill algorithm requires the maze area to be divided into cells, which requires a large memory and is expensive to update.

The maze routing algorithm [14] is able to find the possible paths between any two points in the maze area, and guarantees to find the connection between 2 terminals if exist, and promise to find out the shortest-path possible. However, maze routing algorithm is slow and requires large memory size for dense layout. On the other hand, the Soukup is an efficient maze routing solving algorithm, but it has three minor disadvantages: (1) the initial and the target points have to be specified, (2) the routes are suboptimal when the maze area becomes congested, and (3) the algorithm cannot guarantee the obtained path is the shortest one.

The maze solving algorithms for unknown environments were discussed. These include: wall follower, Tremaux's algorithm, random mouse and Pledge. The wall follower algorithm [19, 20] is easy to implement, there is no need to know the robot's location and orientation in the maze, and it only needs to follow the wall. However, the wall follower algorithm does not work in every maze, especially those containing loops. On the other hand, Tremaux's algorithm [21, 22] always finds a path to the destination-point and is able to explore the direct path to the destination-point. However, in robotics, it is hard to implement Tremaux's algorithm and large mazes require large memory space to store the visited paths.

Random mouse algorithm does not record where the robot has been, but only decides where to go next. The solver robot could wander for hours when taking wrong turns and there is a strong possibility that the robot will not find the exit in the time allocated. Although this method would always eventually find the solution, this algorithm can be extremely slow. Therefore, random mouse algorithm is inefficient in terms of speed and accuracy.

Wall follower algorithms can solve mazes when the entrance and exit are on the outer walls of the maze. However, wall follower algorithms fail to solve the maze if the robot starts inside the maze area. The Pledge algorithm can solve this problem, since it is designed to circumvent obstacles and requires a random direction to go through. The Pledge algorithm allows a robot with compass to find the path from any point in the maze area to the exit. However, this algorithm will not work in reverse, i.e., finding a path from the exit to the entrance. The Pledge algorithm is more powerful than wall follower algorithm and can solve mazes that wall follower cannot.

Table I presents a comparison of the maze solving algorithms discussed earlier. The maze solving algorithms have been compared according to the following issues:

- Memory size: the total size of memory required to process the maze solving algorithm through the robotic processor.
- Implementation with robotics: the ability to implement the maze solving algorithm in robotic systems.
- Algorithm complexity: this measures the amount of time it takes to run the maze solving algorithm.
- Obtain the shortest path: can the maze solving algorithm find the shortest path in the maze area when more than one path exists between the source and destination points.

In section 4, we discussed the existing solver robot systems and placed them into two main categories based on the method used to solve the maze: camera-based systems and sensor-based systems.

The existing image processing-based maze solving robotic systems [26-31] reduce the total time required to solve the maze. In addition, they offer efficient path planning and achieve reliable navigation in small environments (for instance 2 m × 2 m). However, image processing-based solutions require a camera to capture the maze area. Moreover, an efficient processor and memory are required in order to process and analyse the captured images and explore the possible paths. Finally, image processing-based systems are inapplicable in large maze areas and real situations where it is impossible to capture the maze area.

Sensor-based systems are efficient solutions in terms of cost and reliability. The rangefinder-based robotic systems [32-40] employ rangefinder sensors to determine the distance between the robot and the wall. These systems are easy to implement, inexpensive and efficient.

The line follower maze solving robotic systems [45, 46, 47, 48, 49, 50] are similar to the wall follower systems, however they differ in the technology used in the navigation process, since the line follower systems are based on sensing the colour placed on the floor in order to determine the direction of the solver robot. These systems are efficient in terms of cost, but they fail in environments with no light source. Table II presents a comparison between the existing solver robot systems where the systems are evaluated according to the following issues:

1. Efficiency: this evaluates the productivity of the maze solver robotic system in all environments.
2. Maze area: this assesses the efficiency of the solver robot in any size environment in order to accomplish the mission that the solver robot designed to.
3. Robot solver Complexity: this measures the solver robot design and implementation. The requirement for complicated sensors increases the complexity of the solver robot design.
4. Cost: the maze solver robot cost plays a significant role on the total efficiency of the maze robotic solver system.

TABLE I. A COMPARISON BETWEEN THE MAZE SOLVING ALGORITHMS

	Maze Solving Algorithm	Memory size	Implementation with robotics	Algorithm Complexity	Obtain the shortest path
Known environment algorithms	Dead-end	Large	Hard	$O(P^N)$	×
	Maze routing	Large	Medium	$O(r \times c)$ *r*: rows *c*: columns	√
	Flood-fill	Large	Medium	$O(r \times c)$	√
	Lee's	Large	Medium (requires grid structure)	$O(r \times c)$	√
	Soukup	Large	Medium (requires grid structure)	$O(r \times c)$ 10-50 times faster than Lee's algorithm	×
	Hadlock	Large	Medium (requires grid structure)	$O(r \times c)$	√
	Heuristic	Large	Hard	$O(b^d)$ *d: depth of solution b: the branching factor*	
Unknown environment algorithms	Wall-follower	Medium	Easy	$O(n^2)$ *n:* # of junctions in the maze area	×
	Tremaux's	Large	Hard	$O(n^2)$ *n:* # of junctions in the maze area	×
	Random mouse	Low	Easy	$O(n^2)$ *n:* # of junctions in the maze area	×
	Pledge	Medium	Medium	$O(n^2)$ *n:* # of junctions in the maze area	×

TABLE II. A COMPARISON BETWEEN THE MAZE SOLVER ROBOT SYSTEMS

Maze solver technique	Efficiency	Maze Area	System Complexity	Cost
Camera-based systems	high efficiency in small areas	Small areas	Requires processing for the captured images	High
Rangefinder systems	Efficient in areas with walls	Any maze area size	Sensing data can be processed using any microcontroller	Low
Line follower systems	Efficient in areas with lines drawn	Any maze area size	Sensing data can be processed using any microcontroller	Low

VI. Conclusion & Future Work

Several maze solving algorithms have been proposed with diverse structure and efficiency. However, there is no perfect maze solving algorithm for this purpose. The deployment of a certain maze solving algorithm depends on several factors including: area size, area complexity, area structure, deployment cost and complexity. In this paper, we investigated the maze solving algorithms and categorized them in terms of the deployed maze environment into two categories. In addition, we discussed the recent deployment of maze solving robotic systems and a critical analysis is presented, in order to guide researchers and developers for choosing the most suitable maze solving algorithm.

Conflict of Interest

The authors declare that there is no conflict of interest.

Author Contributions

Ms. Shatha reviewed the recent developed maze solving algorithms. Ms. Shuruq conducted the research on the existing maze solving robotic system, whereas Ms. Wejdan discussed the maze solving algorithms, Ms. Hadeel discussed the maze solving robotic systems, and Ms. Ahad performed a comparison study among the existing robotic systems, and Dr. Tareq Alhmiedat wrote the paper.

Acknowledgment

The authors wish to thank the Industrial Innovation and Robotics Center at the University of Tabuk.

References

[1] C. Wang, L. Meng, S. She, I. M. Mitchell, T. Li, F. Tung, W. Wan, M. Q. H. Meng, C. W. De Silva, "Autonomous mobile robot navigation in uneven and unstructured indoor environments," in *Proc. 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 109-116, September, 2017.

[2] A. Saman and I. Abdramane, "Solving a reconfigurable maze using hybrid wall follower algorithm," *International Journal of Computer Applications*, vol. 82, no. 3. 2013.

[3] S. Mishra and B. Pankaj, "Maze solving algorithms for micro mouse," in *Proc. 2008 IEEE International Conference on Signal Image Technology and Internet Based Systems*, pp. 86-93, 2008.

[4] B. Gupta and S. Sehgal, "Survey on techniques used in autonomous maze solving robot," in *Proc. 2014 5th International Conference-Confluence: The Next Generation Information Technology Summit*, 2014, pp. 323-328.

[5] N. Kumar and S. Kaur, "A review of various maze solving algorithms based on graph theory," *International Journal for Scientific Research & Development*, vol. 6, no. 12, 2019.

[6] R. C. Arkin and R. R. Murphy, "Autonomous navigation in a manufacturing environment," *IEEE Transactions on Robotics and Automation*, vol. 6, no. 4, pp. 445-454. 1990.

[7] P. J. Costa, N. Moreira, D. Campos, J. Gonçalves, J. Lima, P. L. Costa, "Localization and navigation of an omnidirectional mobile robot: The robot@ factory case study," *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje*, vol. 11, no. 1, pp. 1-9, 2016.

[8] H. Sahin and L. Guvenc, "Household robotics: Autonomous devices for vacuuming and lawn mowing [applications of control]," *IEEE Control Systems Magazine*, vol. 27, no. 2, pp. 20-96, 2007.

[9] M. L. Delle-Monache, J. Sprinkle, R. Vasudevan, and D. B. Work, "Autonomous vehicles: From vehicular control to traffic control," *HAL-Inria*, 2019.

[10] S. M. Farritor and M. E. Rentschler, "Robotic highway safety markers," in *Proc. ASME 2002 International Mechanical Engineering Congress and Exposition*, pp. 833-839, American Society of Mechanical Engineers Digital Collection, January, 2002.

[11] N. Ruangpayoongsak, H. Roth, and J. Chudoba, "Mobile robots for search and rescue," in *Proc. IEEE International Safety, Security and Rescue Rototics, Workshop, 2005*, pp. 212-217, June, 2005.

[12] H. Sugiyama, T. Tsujioka, and M. Murata, "Coordination of rescue robots for real-time exploration over disaster areas," in *Proc. 2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)* , 2008, pp. 170-177,.

[13] H. Abelson and A. DiSessa, *Turtle Geometry: The Computer as a Medium for Exploring Mathematics*, MIT press. 1986.

[14] M. Fattah, A. Airola, R. Ausavarungnirun, N. Mirzaei, P. Lileberg, J. Plosial, S. Mohammadi, T. Pahikkala, O. Mutlu, H. Tenhunen, "A low-overhead, fully-distributed, guaranteed-delivery routing algorithm for faulty network-on-chips," in *Proc. the 9th International Symposium on Networks on Chip*. ACM. 2015.

[15] A. Glassner, "Graphics gems I". *San Francisco: Morgan Kaufmann Publishers Inc*. 1990.

[16] C. Y. Lee, "An algorithm for path connections and its applications," *IRE Transactions on Electronic Computers*, vol. 3, pp. 346-365, 1961.

[17] J. I. R. I, Soukup, "Fast maze router," in *Proc. the 15th Design Automation Conference*, pp. 100-102, IEEE Press, 1978.

[18] F. O. Hadlock, "A shortest path algorithm for grid graphs," *Networks*, vol. *7, no.* 4, pp. 323-334. 1977.

[19] D. Hanafi, Y. Abueejela, and M. Zakaria, "Wall follower autonomous robot development applying fuzzy incremental controller," *Intelligent Control and Automation*, vol. 4, no. 01, p.18, 2013.

[20] M. Al-Khawaldah, O. Badran, and I. Al-Adwan, "Exploration algorithm technique for multi-robot collaboration," *Jordan Journal of Mechanical and Industrial Engineering*, vol. 5, no. 2. pp. 177-184. 2011.

[21] Public conference, December 2, 2010 – by professor Jean Pelletier-Thibert in Academie de Macon (Burgundy – France) – (Abstract published in the Annals academic, March 2011 – ISSN 0980-6032) Charles Tremaux (1859–1882) Ecole Polytechnique of Paris (X:1876), French engineer of the telegraph. 2010.

[22] E. Lucas,"Récréations Mathématiques," vol. I, 1882.

[23] M. Babula, "Simulated maze solving algorithms through unknown mazes," *Organizing and Program Committee*, vol. 13, 2009.

[24] D. Abelson. Turtle Geometry: The computer as a medium for exploring mathematics," 1980.

[25] S. Papert, "Uses of technology to enhance education," *MIT Artificial Intelligence Memo*, no. 298, 1973.

[26] M. Aqel, A. Issa, M. Khdair, M. ElHabbash, M. AbuBaker, and M. Massoud, "Intelligent maze solving robot based on image processing and graph theory algorithms," *International Conference on Promising Electronic Technologies (ICPET)*, pp. 48-53. 2017.

[27] O. Kathe, T. Varsha A. Jagtap, and G. Gidaye, "Maze solving robot using image processing," *IEEE Bombay Section Symposium (IBSS)*, pp. 1-5. 2015.

[28] A. Chandak, G. Ketki, G. Shalaka, A. Sumeet & P. Kulkarni. "Path planning for mobile robot navigation using image processing," *International Journal of Scientific & Engineering Research*, vol. 4, no. 6, pp, 1490-1495, 2013.

[29] B. Rahnama, A. Elçi, and S. Metani, "An image processing approach to solve labyrinth discovery robotics problem," in *Proc. The 36th Annual Computer Software and Applications Conference* pp. 631-636, 2012.

[30] N. Barnouti, S. Al-Dabbagh, M. Naser, "Pathfinding in strategy games and maze solving using A search algorithm," *Journal of Computer and Communications*, vol. 4, no. 11, p. 15. 2016.

[31] K. Lutvica, J. Velagić, N. Kadić, N. Osmić, G. Džampo, and H. Muminović, "Remote path planning and motion control of mobile robot within indoor maze environment," *International Symposium on Intelligent Control (ISIC)*, pp. 1596-1601, 2014.

[32] L. Wyard-Scott, and M. Meng, "A potential maze solving algorithm for a micromouse robot," in *Proc. IEEE Pacific Rim Conference on Communications, Computers, and Signal Processing. Proceedings*, pp. 614-618, 1995.
[33] I. Elshamarka and S. Abu Bakar, "Design and implementation of a robot for maze-solving using flood-fill algorithm," *International Journal of Computer Applications*, vol. 56, no. 5, 2012.
[34] I. Arya, G. Fiona K. Sohum, M. Karan, R. Daniella B. Jacob "Autonomously solving mazes with robots," *Thesis for Bachelor of Science in Electronics and Telecommunication Engineering*, 2017.
[35] F. Annaz, "A mobile robot solving a virtual maze environment," *International Journal of Electronics, Computer and Communications Technologies*, vol. 2, no. 2. 2012.
[36] A. Islam, F. Ahmad, and P. Sathya, "Shortest distance maze solving robot," *IJRET: International Journal of Research in Engineering and Technology*, vol. 5, no. 7. 2016.
[37] I. Elshamarka and A. Saman, "Design and implementation of a robot for maze-solving using flood-fill algorithm," *International Journal of Computer Applications*, vol. 56, no. 5, 2012.
[38] S. Tjiharjadi, C Wijaya, and E. Setiawan, "Optimization maze robot using A* and flood fill algorithm," *International Journal of Mechanical Engineering and Robotics Research*, vol. 6, no. 5. 2017.
[39] J. Su, X. Cai, C. Lee, and C. Chen. "The development of a half-size micromouse and its application in mobile robot education," in *Proc. 2016 International Conference on Advanced Robotics and Intelligent Systems (ARIS)*, 2016, pp. 1-6.
[40] C. Wu, D. Liaw, and H. Lee, "A method for finding the routes of mazes," in *Proc. 2018 International Automatic Control Conference (CACS)*, 2018, pp. 1-4.
[41] R. Kumar, P. Jitoko, S. Kumar, K. Pillay, P. Prakash, A. Sagar, R. Singh, and U. Mehta, "Maze solving robot with automated obstacle avoidance," *Procedia Computer Science*, vol. 105, pp. 57-61. 2017
[42] S. Chua, J. Dominguez, J. Limqueco, E. Lu, S. Que, J. Rosario, and D. Abuan, "Development of a maze solving mobile robot capable of tracking the distance it traversed," *Advanced Science Letters*, vol. 24, no. 11, pp. 8640-8646. 2018.
[43] S. Tjiharjadi, "Design and implementation of flood fill and pledge algorithm for maze robot," *International Journal of Mechanical Engineering and Robotics Research*, vol. 8, no. 4, 2019.
[44] L. Bienias, K. Szczepański, & P. Duch. "Maze exploration algorithm for small mobile platforms," *Image Processing & Communications*, vol. 21, no. 3, pp.15-26. 2016.
[45] K. Norbert-Brendan and T. C. Marius, "Autonomous line maze solver using artificial intelligence," in *Proc. 15th International Conference on Engineering of Modern Electric Systems (EMES)*, 2019, pp. 133-136.
[46] S. Sakib, A. Chowdhury, S. Ahamed, and S. Hasan, "March. maze solving algorithm for line following robot and derivation of linear path distance from nonlinear path," in *Proc. 16th International Conference on Computer and Information Technology*, pp. 478-483. 2014.
[47] A. Hidayatullah, A. Jati, and C. Setianingsih, "Realization of depth first search algorithm on line maze solver robot," in *Proc. 2017 International Conference on Control, Electronics, Renewable Energy and Communications (ICCREC)*, 2017, pp. 247-251.
[48] R. Musridho, F. Yanto, H. Haron, and H. Hasan, "Improved line maze solving algorithm for curved and zig-zag track," in *Proc. 7th ICT International Student Project Conference (ICT-ISPC)*, 2018, pp. 1-6.
[49] A. Khan, G. Rana, M. Rabin, F. Mitul, and M. Shahjahan, "Design and implementation of a robot for maze-solving with turning indicators using PID controller," in *Proc. 2013 International Conference on Informatics, Electronics and Vision (ICIEV)*, 2013, pp. 1-6.
[50] C. Saranya, M. Unnikrishnan, S. Ali, D. Sheela, and V. Lalithambika, "Path planning algorithm for maze: Design and implementation in LEGO mindstorms rover," in *Proc. 2017 International Conference on Inventive Computing and Informatics (ICICI)*, 2017, pp. 189-193.
[51] T. A. Alhmiedat, A. Abutaleb, and G. Samara, "A prototype navigation system for guiding blind people indoors using NXT Mindstorms," *International Journal of Online and Biomedical Engineering (iJOE)*, vol. *9*, no. 5, pp. 52-58. 2013.

Shatha Alamri is a Teaching Assistant in the Department of Computer Science, and she has been involved in the Industrial Innovation & Robotics Center at the University of Tabuk, Tabuk, Saudi Arabia, 2018. Her main research in the field of AI and robotics. Ms. Shatha is currently completing her master's degree in computer science at King Saud University, Riyadh, Saudi Arabia.

Shuruq Al-Shehri, is a Teaching Assistant in the Department of Computer Science, in the Faculty of Computers & Information Technology, at the University of Tabuk. She received a BSc degree in Computer Science from the Faculty of Computers and Information Technology, University of Tabuk , Tabuk , Saudi Arabia , 2019.

Wejdan Alshehri has completed her BSc degree in the Department of Computer Science from the Faculty of Computers & Information Technology, at the University of Tabuk, Saudi Arabia, 2019.

Hadeel Alamri has completed her BSc degree in the Department of Computer Science from the Faculty of Computers & Information Technology, at the University of Tabuk, Saudi Arabia, 2019.

Ahad Alaklabi has received BSc degree in Computer Science from the Faculty of Computers and Information Technology, University of Tabuk , Tabuk , Saudi Arabia , 2019.

Tareq Ahmiedat is an Associate Professor in the Department of Computer Science, and the head of Robotics & Artificial Intelligence Unit, in the Industrial Innovation & Robotics Center at University of Tabuk, Tabuk, Saudi Arabia. His research interests include tracking mobile targets through Wireless Sensor Networks, robot navigation and orientation, and robot vision systems. Dr. Alhmiedat has been involved in various research projects including: SafetyNET, IndoorTrack, Diabetic Robot (SARA), and WSN environment monitoring.