Design of Multi-Agent Pathfinding Robot Using Improved Flood Fill Algorithm in Maze Exploration

Semuil Tjiharjadi*
Computer System Department, Faculty of Engineering, Maranatha Christian University, Bandung, Indonesia
Centre for Robotics and Industrial Automation, Fakulti Teknologi Maklumat dan Komunikasi,
Universiti Teknikal Malaysia Melaka, Durian Tunggal, Melaka, Malaysia
Email: semuiltj@gmail.com

Sazalinsyah Razali*
Centre for Robotics and Industrial Automation, Fakulti Teknologi Maklumat dan Komunikasi,
Universiti Teknikal Malaysia Melaka, Durian Tunggal, Melaka, Malaysia
Email: sazalinsyah@utem.edu.my

H. A. Sulaiman
Centre for Adcanced Computing Technology, Fakulti Teknologi Maklumat dan Komunikasi,
Universiti Teknikal Malaysia Melaka, Durian Tunggal, Melaka, Malaysia
Email: asyrani@utem.edu.my

Gerry Fernando
Computer System Department, Faculty of Engineering, Maranatha Christian University, Bandung, Indonesia
Email: gerryfernando31@gmail.com

***Abstract*—Maze exploration is a field of science that continues to be researched and developed. Various algorithms, techniques, and scenarios result in faster trips and shortest distances. Various international competitions in the maze field are also held to encourage quicker and better labyrinth exploration innovations. Usually, maze exploration uses a mobile robot that moves autonomously in pathfinding to reach the target location. This study aims to generate and test a multi-agent algorithm to find the closest distance to reach a target in an unknown maze. So the question is how multi-agents can work together to reach the target using the shortest possible path. For this reason, a flood fill algorithm usually used for single-agent has been developed into an improved flood fill algorithm that can be applied to multi-agents. The result is an algorithm that can be applied to find targets through unknown maze exploration using multi-agent.**

***Index Terms*— maze robot, flood fill algorithm, multi-agent, pathfinding**

I. Introduction

Research on the use of the robot to explore the maze has been carried out for a long time and is still developing today. The application development of maze research exploration can be implemented in many areas, such as autonomous cars, traffic control, video games, warehouse robots, Unmanned Aerial Vehicles (UAV), search and rescue missions, and many other applications.

Autonomous navigation is an essential feature of mobile robotics. It enables the robot to move independently to a target location without being controlled. Many algorithms have been studied, each with its benefits and limitations.[1][2][3]

Usually, maze exploration uses a mobile robot that moves autonomously in pathfinding to reach the target location [4]. This research tested a single and multi-agent pathfinding robot to find the target location and return to the starting point in a 16 x 16 maze arena. It is considered sufficient for two robot experiments. More multi-agent pathfinding robots need broader maze arenas to give more chances to get more exciting data

Overall, the contribution of this research is the novelty of the Improved Flood Filling Algorithm in maze exploration, which has been successfully applied to multi-agents. The test was carried out to test the Improved Flood Fill algorithm, which was used to multi-agent pathfinding robots, and at the same time, the weaknesses of the application of the Flood Fill algorithm when carried out for multi-robots.

II. Literature Review

Maze exploration has long been a growing field of research. Its application in various industries, including games, robots, autonomous vehicles, and autonomous

Manuscript received March 25, 2022; revised June 27, 2022.

doi: 10.18178/ijmerr.11.8.631-638

warehousing, has attracted researchers worldwide. Overall, the maze problems can be classified into three scenarios: (1) the target location is unknown but has a maze schema; (2) the Labyrinth Schematic is unknown but has a target location; and (3) both the target location and the labyrinth scheme are unknown. Various approaches and ways of thinking have been developed to create new and improved methods and algorithms exploring mazes. [5]

In the game industry, the first scenario is frequently used, in which the path is known, but the enemy and target are unknown[6]. The robot must navigate a maze of unknown paths in the second scenario to reach the predetermined target[7]. The annual micro mouse competition uses this second scenario, in which the robot must explore the maze's path to the destination and then return to the starting point. The robot must then arrive at the target location as quickly as possible using the best distance calculated from the previous exploration results. Meanwhile, in the third scenario, the target's and the road's locations in the maze are unknown and must be searched[8][9][10]. The target location could be in the middle or at the edge of the maze, so the robot must explore the maze while mapping the maze based on its exploration.

Maze exploration can also be developed by using more than one robot. It is called multi-agent pathfinding (MAPF) [11][12], one of Multi-agent System development. There are two groups of MAPF algorithms:

1. MAPF will be simplified into understandable algorithms using a reduction-based solver. SAT [13], linear integer programming, and response set programming are part of the Reduction-based solver.
2. Search-based solvers, such as Flood Fill algorithm [3], Dijkstra's algorithm [14], A* algorithm [15], Pledge algorithm [16], Genetic algorithm [17], Trémaux's[18], Trees and Ant colony optimization [19], are the second group of MAPF development.

TABLE I. COMPARISON OF PATHFINDING MAZE ALGORITHM

Algorithm	Benefits	Limitation
A*	Cost-efficient	Requires a large amount of memory
Ant colony algorithm	Optimal time	More number of iterations
Genetic algorithm	Multiple solutions (beneficial when input is enormous)	Inconstant optimization response times
Depth First Search	Simple to implement	Large computing power
Breadth-First Search	When memory is not a problem	Large space complexity
Flood Fill	Optimal	High-cost updates

Maze exploration is a search-based solver, which belongs to the second group. A pathfinding algorithm based on search-based solvers aims to reduce costs from start to finish. Because it can reduce the heuristic cost as much as possible, the A* algorithm is frequently used in Artificial Intelligence that follows the path. Aside from A*, an ant colony algorithm also mimics ant behavior. The algorithm initially traverses nodes at random while updating the cost of each node. Then we can find the best route in terms of time. The disadvantage is that it necessitates numerous iterations. A Flood Fill algorithm is also available and is commonly used for maze exploration. In Flood Fill, each cell has a value representing its distance from the target. Of course, this necessitates regular updates. [20]

TABLE II. MAIN STUDIES ON MULTI-AGENT PATHFINDING [21]

No.	Author	Title	Env.	Agent	Task
1	V. Rahmani, et al, 2020	Multi-Agent Parallel Hierarchical Path Finding in Navigation Meshes (A-HNA*)[22]	Static - Known	Multi	No
2	Semenas, 2020	Modeling of Autonomous SAR Missions[23]	Static - Known	single	No
3	R. Bartak, 2019	Multi-agent Pathfinding on Real Robots[11]	Static - Known	Multi	No
4	Bogatarkan, 2019	Conference A Declarative Method for Dynamic MAPF[24]	Dynamic-Known	Multi	No
5	P.M. Chu, et al. 2019	Flood Fill based object segmentation and tracking for intelligent vehicles[3]	Static - Known	single	No
6	Y. Liu, 2019	Formation control and collision avoidance for a class of multi-agent systems[25]	Static - Known	Multi	No
7	K. Shetty, 2019	Drivable road corridor detection using Flood Fill road detection algorithm[26]	Static - Known	single	No
8	H. Shi, et al. 2019	Clustering-based task coordination to SAR teamwork of multiple agents[27]	Static - Known	Multi	No

9	A. Botea, et al. 2018	Solving Multi-agent pathfinding on strongly biconnected digraphs[28]	Static - Known	Multi	No
10	Robinson, 2018	An Efficient Algorithm for Optimal Trajectory Generation for Heterogeneous Multi-Agent Systems in Non-Convex Environments[29]	Static - Known	Multi	No
11	Wu, 2018	A Method for Finding the Routes of Mazes[4]	Static - Known	single	No
12	Gade, 2017	Design and Implementation of Swarm Robotics using Flood Fill algorithm	Static - Known	Multi	No
13	Ma, 2017	Overview Generalizations of Multi-Agent Pathfinding to Real-World Scenarios	Static - Known	Multi	Yes
14	Ambeskar, 2016	Pathfinding Robot Using Image Processing	Static - Known	single	No
15	Jabbar, 2016	Autonomous Navigation of Mobile Robots based on Flood Fill Algorithm	Static - Known	single	No
16	Surynek, 2016	Efficient SAT Approach to Multi-Agent Pathfinding Under the Sum of Costs Objective	Static - Known	Multi	No
17	Andre, 2015	Collaboration in Multi-Robot Exploration	Static-Unknown	Multi	No
18	Ansari, 2015	An optimized hybrid approach for pathfinding	Static - Known	single	No
19	Ozturk, 2015	Optimal bid valuation using pathfinding for multi-robot task allocation	Static - Known	Multi	Yes
20	Sharon, 2015	conflict based search for optimal multi-agent pathfinding	Static - Known	Multi	No
21	Singh, 2015	A New Shortest First Pathfinding Algorithm for a Maze Solving Robot	Static - Known	single	No
22	Law, 2013	Quantitative Comparison of Flood Fill and Modified Flood Fill Algorithms	Static - Known	single	No
23	Reddy, 2013	Pathfinding - Dijkstra's and A star algorithms	Static - Known	single	No
24	Xue, 2013	Formation Control and Obstacle Avoidance for Hybrid Multi-Agent Systems	Static - Known	Multi	No
25	Chauhan, 2012	Evaluation of Modified Flood Fill Algorithm for Shortest Path Navigation in Robotics	Static - Known	single	No
26	Elshamarka, 2012	Design and implementation of a Robot for Maze Solving using Flood Fill Algorithm	Static - Known	single	No
27	Zheng, 2011	Recursive Path Planning in a Dynamic Maze with Modified Tremaux's Algorithm	Static - Known	single	No
28	Jin, 2010	Multi-robot pathfinding with wireless multihop communications	Static - Known	Multi	No
29	Schurr, 2005	The Future of Disaster Response Humans Working with Multiagent Teams using DEFACTO	Static - Unknown	Multi	No

Table I summarizes various pathfinding maze algorithms. These algorithms are implemented as a single agent in the static maze environment. The most common algorithms used to implement a maze-solving robot are A* and Flood Fill.

The advantages and limitations of each existing algorithm are shown in Table I. The Flood Fill algorithm

was chosen for this study based on the benefits of this algorithm, which is optimal and is the best algorithm that is frequently used in the annual micro mouse competition.[30][31]

There are several pieces of research on the exploration of the maze and multi-agents. It shows in Table II.

III. Hardware Design

The robot used for experiments in this study uses a miniQ 2WD robot chassis. Fig. 1 depicts the robot chassis. It consists of a 122mm diameter robot chassis, a pair of wheels, ball casters, and a couple of Direct Current (DC) motors with a gearbox and DC motor bracket. This robot uses a rotary encoder connected to a DC motor to calculate wheel rotation, as shown in Fig. 2 [12].

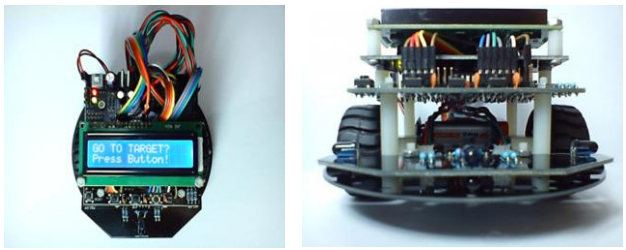

Figure 1. Mobil Robot from the overhead and side view.

It has three infrared sensors that detect the position of the maze wall in the front, right, and left. Driver L293D controls the speed and rotation of a DC Motor in this maze robot [13]. It also has a rotary encoder, which calculates the spin of both wheels. The robot is started by pressing a button.

Robots use DC motors to rotary the wheels. It would direct the robot to move forward, turn left or right, and rotate backward [14]. The AT Mega 328 microcontroller in this maze robot responds to input signals and runs the actuator based on processing algorithms [10]. It was designed to update the maze's path using ESP8266 Wifi MCU to and from the laptop.

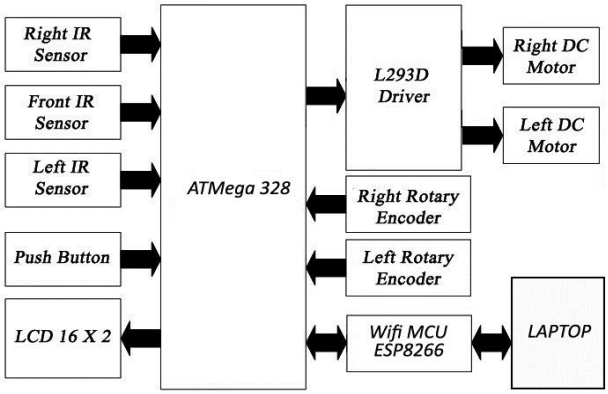

Figure 2. Maze a single robot's block diagram.

The maze size is 16x16 cells in size, as shown in Fig. 3. The maze was designed with two paths to solve it. One of the paths is significantly longer than the other. The Robot (Fig. 2) must determine which path is the shortest and then solve the maze using that path.

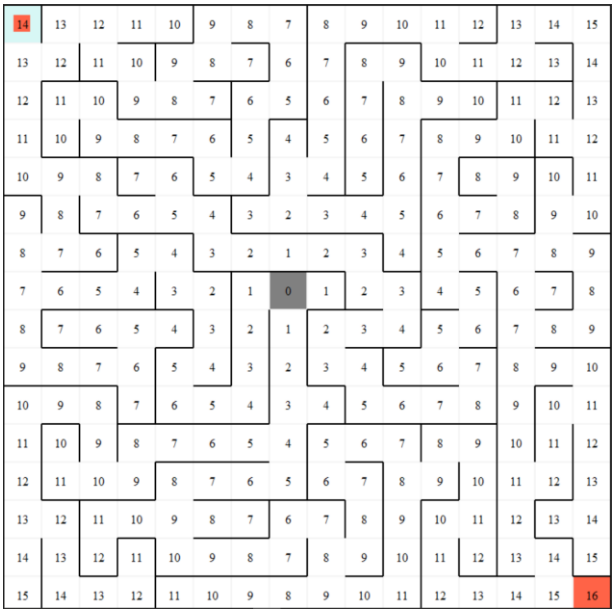

Figure 3. The layout of the maze.

IV. Algorithm

Several search-based solver algorithms can be implemented to solve the maze cases. However, this research chooses the Flood Fill algorithm as the primary algorithm. It is suitable for the maze with a target location in the middle of the maze. A flowchart of improved flood fill algorithm implemented for multi-agent can be seen in Fig. 4.

The Improved Flood Fill algorithm's primary goal is to update the cell value, share it with other agents, and update its wall map. If the other agent is more effective, then switch the job. As a result, the Flood Fill algorithm adapts to the wall conditions it encounters while exploring the maze. Maze exploration experiments with a single or two robots update the maze walls discovered during exploration.

V. Results and Discussion

There are several types of research on multi-agent and maze exploration. Table II shows the various studies on multi-agent pathfinding that have been carried out. Based on the multiple objectives and characteristics of these various studies, it was found that the majority of maze exploration is still single agents. This study focused on examining the use of multi-agent pathfinding in a maze environment. Flood Fill algorithm is usually used for single-agent but has now been improved to be used on multi-agent.

The experiment was conducted under two circumstances: only one robot was working, and both were working simultaneously. A program is used to simulate the experimental results to facilitate the description of the robot's journey.

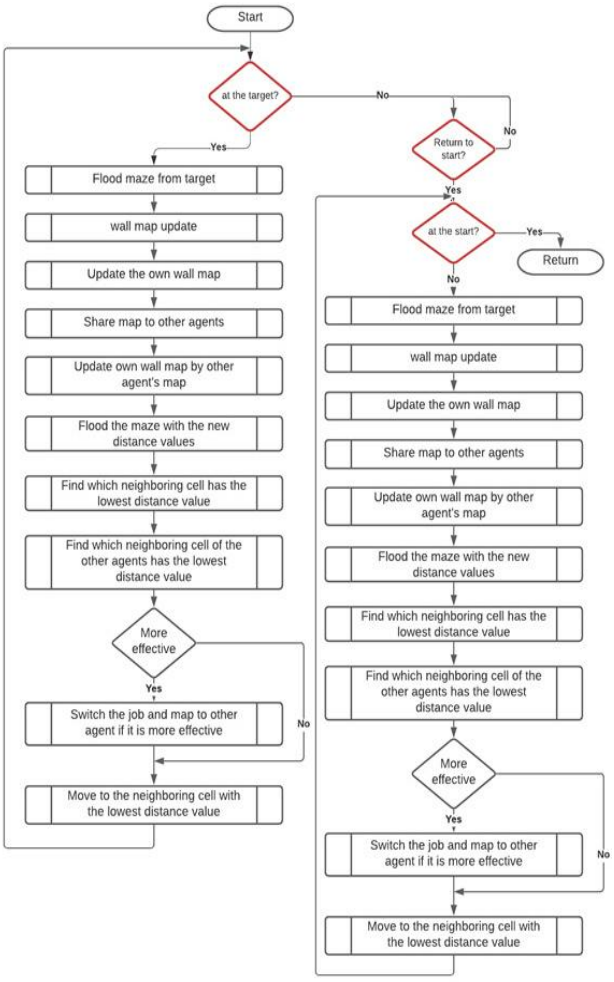

Figure 4. Flowchart of the main program installed in each robot.

Fig. 5 depicts the path taken by the first robot to reach the target location in the center. It begins in the upper left corner of the maze and then explores a target. The first robot takes 94 steps to reach the target.

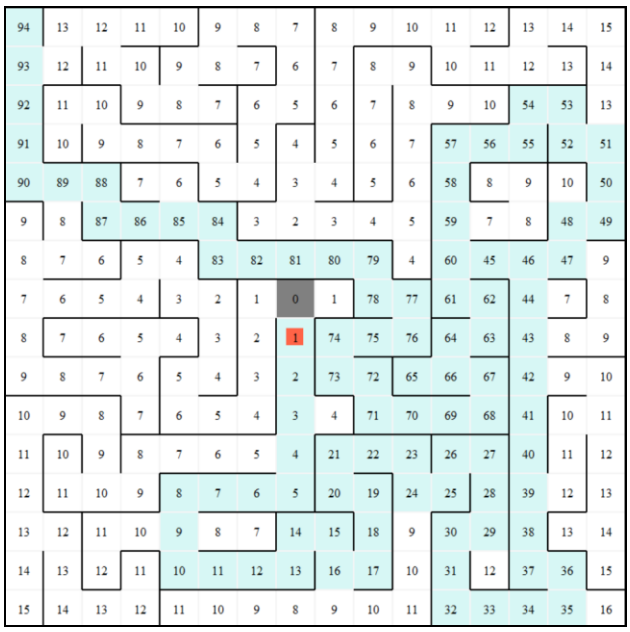

Figure 5. The journey of the first robot towards the target

The first robot's return journey, depicted in Fig. 6, reveals that it did not use the travel route when leaving but instead tried a new path. This route is shorter, with only 27 steps required to return to the starting point.

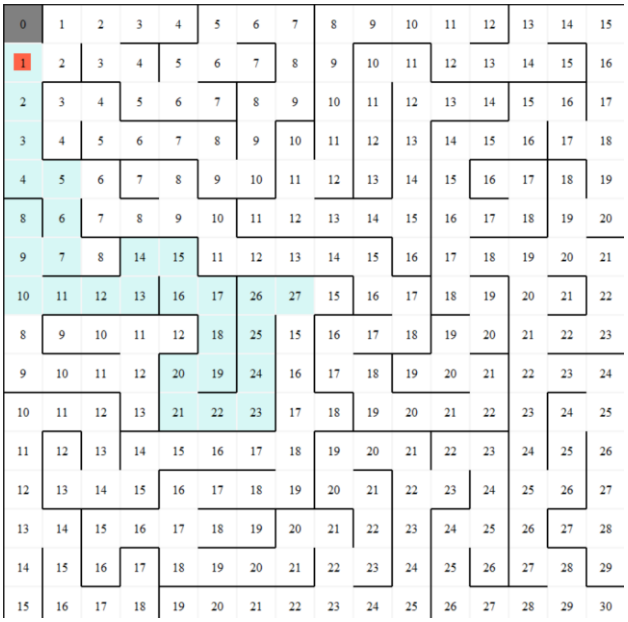

Figure 6. The journey of the first robot to come home from the target

The following experiment employs a second robot that moves from the lower right corner to the target in the center. Fig. 7 depicts the journey. It needs 36 steps to reach the target location.

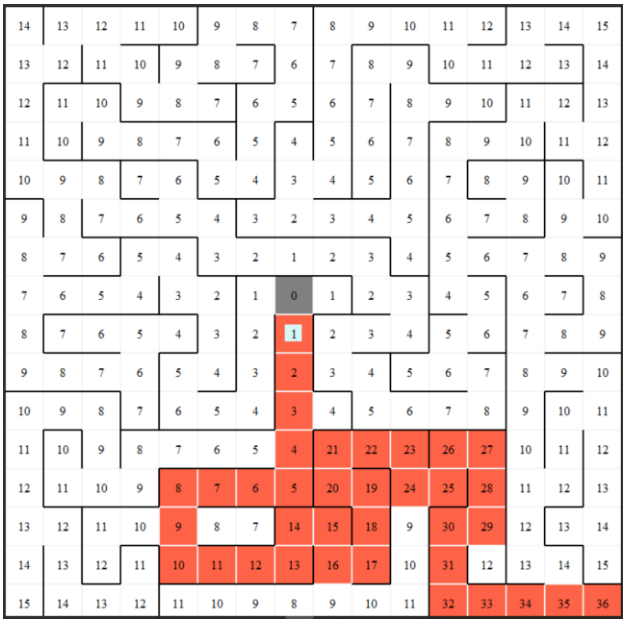

Figure 7. The journey of the second robot towards the target

The next test is the second robot's return journey from the target to the starting point. Fig. 8 depicts this. It uses the same path and needs 36 steps.

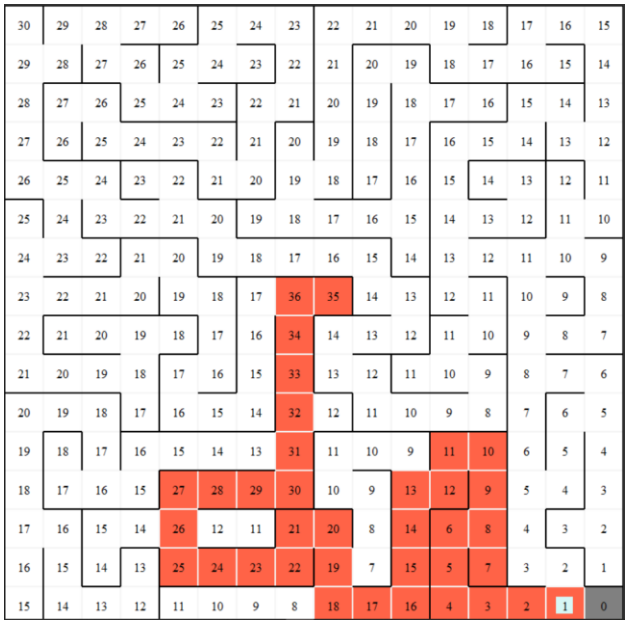

Figure 8. The journey of the second robot to come home from the target

The following experiment involved running two robots simultaneously, from the upper left corner and the second from the lower right corner. Because it is closer than the first robot, the second robot arrives first. Fig. 9 illustrates this.

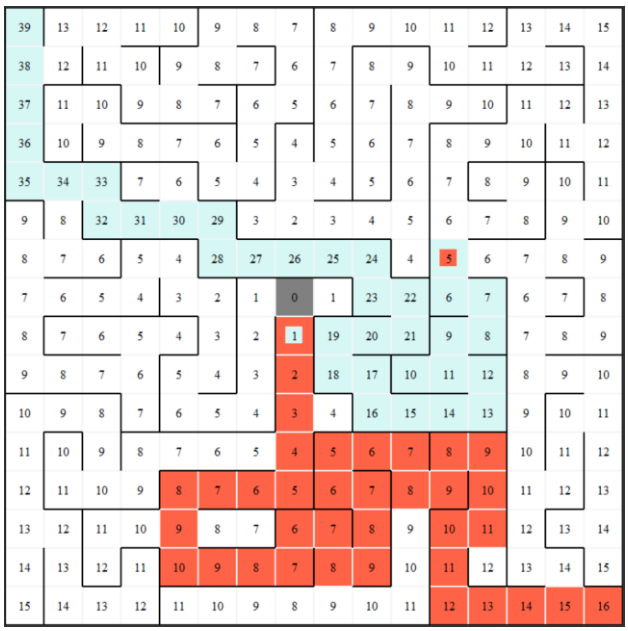

Figure 9. The journey of both robots to search pathfinding towards the target location.

Because the two robots move at different angles, the updated wall information from these two robots' trips does not assist the other robots in moving faster to the target. However, the information on the wall becomes more covered, which will aid in the next exploration.

The second robot's journey back is next, while the first robot is still looking for its way to the target. A collision occurred at one point. (Fig. 10)

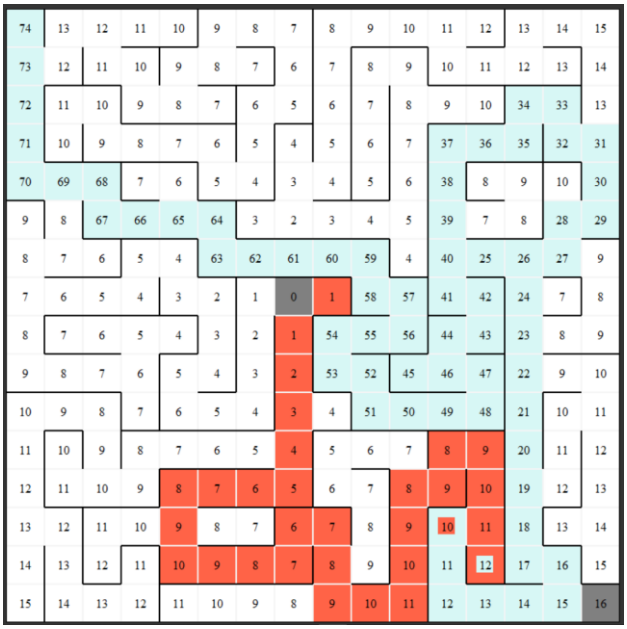

Figure 10. Collision of the robots

Since this collision state would almost certainly be challenging to overcome, it was decided to use task allocation to swap the tasks of the two robots. The task allocation implementation allows the first robot's work to be transferred to the second robot and the second robot's position to be transferred to the first robot. Of course, this is possible as long as the two robots perform nearly identical tasks, namely searching for the target location. The collision problem can be solved using this method.

Another experiment moves the second robot's location closer to the first robot's initial location. When both robots are executed, it turns out that they take the same path, where the leading robot finds a way and the other robot follows the same path as the leading robot. Fig. 11 shows the situation of both robots.

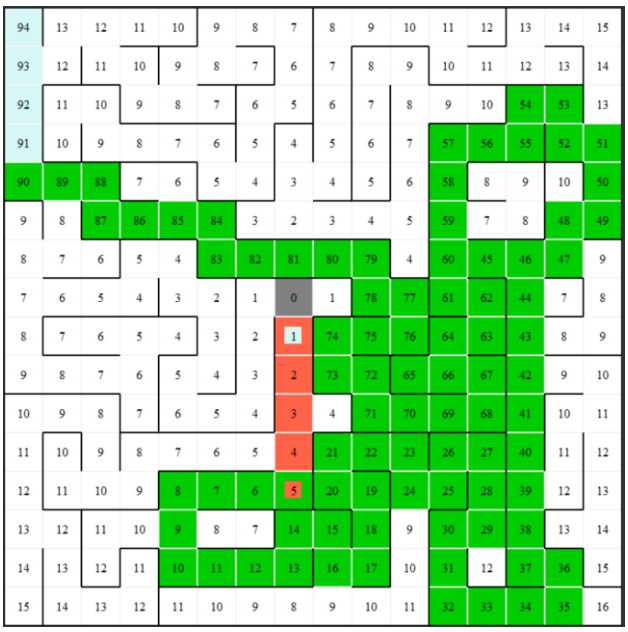

Figure 11. Scenario when the second robot starts close at the starting point of the first robot

VI. Summary

According to the results of the experiments, the two robots were able to communicate with each other and update the data wall they encountered on their journey. The two robots use the data to determine the quickest path to the target.

The two robots should be positioned in opposite directions to provide a better picture of the other robots, and both robots can use this information to determine the closest path.

The problem that most often occurs is when the two robots meet and block each other, for that task allocation is applied to share and exchange tasks. Overall, this facilitates the completion of both robots' tasks.

If the initial locations of the two robots are close together, there is a chance that the two robots will follow the same path. Of course, this reduces the effectiveness of using two robots to explore larger areas faster.

This research in this maze certainly has limitations in terms of maze size, only one agent can occupy the same cell, and the number of agents used is still limited to limit the complexity of the system described.

Future research could look into robotic maze-solving abilities in more extensive and more complex mazes with more robots. Research can also be directed at how many robots are effective for exploring a specific maze size so that the number of robots becomes more effective without actually decreasing effectiveness because of most robots. Furthermore, research can be directed toward establishing a more effective starting location for the robot.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

S. Tjiharjadi conducted the research; S. Tjiharjadi and S. Razali analyzed the data; S. Razali and H.A. Sulaiman supervised the research; S. Tjiharjadi wrote the paper; G. Fernando wrote the simulation program; all authors had approved the final version.

Acknowledgment

The authors wish to thank the Centre for Robotics and Industrial Automation (CeRIA) and Fakulti Teknologi Maklumat dan Komunikasi (FTMK) at Universiti Teknikal Malaysia Melaka (UTeM) for supporting the research and facilities. This work is supported in part by a scholarship from Maranatha Christian University.

References

[1] I. Elshamarka and A. B. S. Saman, "Design and implementation of a robot for maze-solving using flood-fill algorithm," *Int. J. Comput. Appl.*, vol. 56, no. 5, pp. 8–13, 2012.

[2] S. Tjiharjadi, "Performance comparison robot pathfinding uses flood fill-Wall follower algorithm and flood fill-Pledge algorithm," *Int. J. Mech. Eng. Robot. Res.*, vol. 9, no. 6, pp. 857–864, 2020.

[3] P. M. Chu, S. Cho, K. Huang, and K. Cho, "Flood-fill-based object segmentation and tracking for intelligent vehicles," *Int. J. Adv. Robot. Syst.*, vol. 16, no. 6, pp. 1–11, 2019.

[4] C. M. Wu, D. C. Liaw, and H. T. Lee, "A method for finding the routes of mazes," in *Proc. 2018 Int. Autom. Control Conf. CACS 2018*, pp. 1–4, 2019.

[5] X. Yu, Y. Wu, and X. M. Sun, "A navigation scheme for a random maze using reinforcement learning with quadrotor vision," in *Proc. 2019 18th Eur. Control Conf. ECC 2019*, pp. 518–523, 2019.

[6] G. Foderaro, A. Swingler, and S. Ferrari, "A model-based approach to optimizing Ms. Pac-Man game strategies in real-time," *IEEE Trans. Comput. Intell. AI Games*, vol. 9, no. 2, pp. 153–165, 2017.

[7] G. Law, "Quantitative comparison of flood fill and modified flood fill algorithms," *Int. J. Comput. Theory Eng.*, vol. 5, no. 3, pp. 503–508, 2013.

[8] T. Andre and C. Bettstetter, "Collaboration in Multi-Robot Exploration: To Meet or Not to Meet? " *J. Intell. Robot. Syst. Theory Appl.*, vol. 82, no. 2, pp. 325–337, 2016.

[9] J. Hu, J. Xu, and L. Xie, "Cooperative search and exploration in robotic networks," *Unmanned Syst.*, vol. 1, no. 1, pp. 121–142, 2013.

[10] K. A. M. Annuar, M. H. M. Zin, M. H. Harun, M. F. M. A. Halim, and A. H. Azahar, "Design and development of search and rescue robot," *Int. J. Mech. Mechatronics Eng.*, vol. 16, no. 2, pp. 36–41, 2016.

[11] R. Bartâk, J. Švancara, V. Škopková, D. Nohejl, and I. Krasičenko, "Multi-agent path finding on real robots," *AI Commun.*, vol. 32, no. 3, pp. 175–189, 2019.

[12] A. Botea, D. Bonusi, and P. Surynek, "Solving multi-agent pathfinding on strongly biconnected digraphs," *J. Artif. Intell. Res.*, vol. 62, pp. 273–314, 2018.

[13] P. Surynek, A. Felner, R. Stern, and E. Boyarski, "Efficient SAT approach to multi-agent pathfinding under the sum of costs objective," *Front. Artif. Intell. Appl.*, vol. 285, pp. 810–818, 2016.

[14] H. Du, Y. He, and Y. Cheng, "Finite-time synchronization of a class of second-order nonlinear multi-agent systems using output feedback control," *IEEE Trans. Circuits Syst. I Regul. Pap.*, vol. 61, no. 6, pp. 1778–1788, 2014.

[15] S. Tjiharjadi, M. C. Wijaya, and E. Setiawan, "Optimization maze robot using A* and flood fill algorithm," *Int. J. Mech. Eng. Robot. Res.*, vol. 6, no. 5, 2017.

[16] S. Tjiharjadi, "Design and implementation of flood fill and pledge algorithm for Maze Robot," *Int. J. Mech. Eng. Robot. Res.*, vol. 8, no. 4, 2019.

[17] R. Kala, "Multi-robot path planning using co-evolutionary genetic programming," *Expert Syst. Appl.*, vol. 39, no. 3, pp. 3817–3831, 2012.

[18] N. Z. Yew, K. M. Tiong, and S. T. Yong, "Recursive pathfinding in a dynamic maze with modified Tremaux's algorithm,"in *Proc. Int. Conf. Appl. Math. Eng. Math.*, vol. 5, no. 12, pp. 845–847, 2011.

[19] A. Viseras, R. O. Losada, and L. Merino, "Planning with ants," *Int. J. Adv. Robot. Syst.*, vol. 13, no. 5, pp. 1–16, 2016, DOI: 10.1177/1729881416664078.

[20] A. Ansari, M. A. Sayyed, K. Ratlamwala, and P. Shaikh, "An optimized hybrid approach for path finding," *Int. J. Found. Comput. Sci. Technol.*, vol. 5, no. 2, pp. 47–58, 2015.

[21] S. Tjiharjadi, S. Razali, and H. A. Sulaiman, "A systematic literature review of multi-agent pathfinding for maze research," *J. Adv. Inf. Technol.*, vol. 13, no. 4, 2022.

[22] V. Rahmani and N. Pelechano, "Multi-agent parallel hierarchical pathfinding in navigation meshes (MA-HNA*)," *Comput. Graph.*, vol. 86, pp. 1–14, 2020.

[23] R. Semenas and R. Bausys, "Modelling of autonomous search and rescue missions by interval-valued neutrosophic WASPAS Framework," *Symmetry (Basel).*, vol. 12, no. 1, 2020.

[24] A. Bogatarkan, V. Patoglu, and E. Erdem, "A declarative method for dynamic multi-agent path finding," vol. 65, pp. 54–39, 2019.

[25] Y. Liu, H. Yu, P. Shi, and C. C. Lim, "Formation control and collision avoidance for a class of multi-agent systems," *J. Franklin Inst.*, vol. 356, no. 10, pp. 5395–5420, 2019.

[26] K. Shetty and P. Kanani, "Drivable road corridor detection using flood fill road detection algorithm," *Int. J. Eng. Adv. Technol.*, vol. 9, no. 2, pp. 5011–5014, 2019.

[27] H. Shi, R. Zhang, G. Sun, and J. Chen, "Clustering-based task coordination to search and rescue team work of multiple agents," *Int. J. Adv. Robot. Syst.*, vol. 16, no. 2, pp. 1–9, 2019.

[28] A. Botea, D. Bonusi, and P. Surynek, "Solving multi-agent pathfinding on strongly biconnected digraphs," *IJCAI Int. Jt. Conf. Artif. Intell.*, vol. 2018-July, pp. 5563–5567, 2018.
[29] D. R. Robinson, R. T. Mar, K. Estabridis, and G. Hewer, "An efficient algorithm for optimal trajectory generation for heterogeneous multi-agent systems in non-convex environments," *IEEE Robot. Autom. Lett.*, vol. 3, no. 2, pp. 1215–1222, 2018.
[30] H. Dang, J. Song, and Q. Guo, "An efficient algorithm for robot maze-solving," in *Proc. 2010 2nd Int. Conf. Intell. Human-Machine Syst. Cybern. IHMSC 2010*, vol. 2, pp. 79–82, 2010.
[31] D. J. O. Hoetama, F. P. Putri, and P. M. Winarno, "Algoritma fisher-yates shuffle dan flood fill sebagai maze generator pada game labirin," *Ultim. Comput.*, vol. 10, no. 2, pp. 59–64, 2019.

Semuil Tjiharjadi currently serves as an assistant professor in Computer Engineering Department. His research focuses on Robotics, Computer automation, control, and security. He has authored several books, To Be a Great Effective Leader (Jogjakarta, Indonesia: Andi Offset, 2012), Multimedia Programming by SMIL (Jogjakarta, Indonesia: Andi Offset, 2008), Computer Business Application (Bandung, Indonesia: Informatics, 2006) and other. He contributed to the various academic bodies as Vice-Rector of human capital management, assets, and development, Head of Computer Engineering Department, Member: Senate of University, Member: APTIKOM, Member: MSDN Connection, Member: AAJI.

Sazalinsyah Razali has a Ph.D. in Computer Science, Masters in Computer Science, BSc. (Hons.) in Information Technology, a Certified Tester (Foundation Level - MSTB/ISTQB, 2013), and a Certified IT Professional (FE Module - MITPE/ITPEC, 2005). He joined Universiti Teknikal Malaysia Melaka in 2004 as a Lecturer at the Faculty of Information and Communication Technology and now a Senior Lecturer since 2014. He started his academic career in 2003 as a Lecturer at Kolej Universiti Kejuruteraan Utara Malaysia, now known as Universiti Malaysia Perlis. Before that, Dr. Sazalinsyah Razali was a Lead Web Developer in a local IT company and was involved in various web-based and e-commerce projects. He has professionally served as Program Committee and Reviewer in several international conferences & journals, and he is a Professional Member of several professional societies, such as IEEE, IEEE Robotics & Automation Society (IEEE-RAS), IEEE Computational Intelligence Society (IEEE-CIS), IEEE Systems, Man & Cybernetics Society (IEEE-SMC) and Malaysian Society for Engineering & Technology (MySET).

H. A. Sulaiman received his Bachelor and Master degree from Universiti Teknologi Malaysia (UTM), Malaysia, in 2007 and 2010, respectively. He received the Doctor of Philosophy (Ph.D.) in Mathematics with Computer Graphics from Universiti Malaysia Sabah in 2015. His research interest lies in Computer Graphics and Visualization, Mathematics, Computer Games Technology, Data Computation, and Computer Engineering. He has over 700 citations, with Google Scholar at H Index at 12 and SCOPUS H Index at 8. His publications concentrated on Computer Graphics and Visualization and various computer engineering fields. Currently, he holds the position as Head of Department in Interactive Media from 2020. He is actively involved in the management of the faculty. He is also a Professional Technologist from the Board of Technologist Malaysia (MBOT) recognition that held a title of Ts from 2018 until now.

Gerry Fernando received his Bachelor degree in Engineering from Computer System Dept., Maranatha Christian University, Bandung, Indonesia, in 2017 and a Master degree from the National Taiwan University of Science and Technology, Taipei Taiwan, in 2020. He works now as Data Scientist at PT. Bareksa Portal Investasi.