JURNAL INFOTEL
Informatics - Telecommunication - Electronics
Website: http://ejournal.st3telkom.ac.id/index.php/infotel
ISSN: 2085-3688; e-ISSN: 2460-0997

Genetic algorithm for finding shortest path of mobile robot in various static environments

Dyah Lestari[1,*], Siti Sendari[2], Ilham Ari Elbaith Zaeni[3]
[1,2,3]Universitas Negeri Malang
[1]Jl. Semarang, No. 5, Malang 65145, Indonesia
[*]Corresponding email: dyah.lestari.ft@um.ac.id

Received 6 May 2023, Revised 27 August 2023, Accepted 19 September 2023

Abstract — In conducting their work in the industry quickly, precisely, and safely, mobile robots must be able to determine the position and direction of movement in their work environment. Several algorithms have been developed to solve maze rooms, however, when the room is huge with several obstacles that could be replaced in other parts of the room, determining the path for a mobile robot will be difficult. This can be done by mapping the work environment and determining the position of the robot so that the robot has good path planning to get the optimal path. In this research, a Genetic Algorithm (GA) will be used to determine the fastest route that a robot may take when moving from one location to another. The method used is to design a mobile robot work environment, design genetic algorithm steps, create software for simulation, test the algorithm in 6 variations of the work environment, and analyze the test results. The genetic algorithm can determine the shortest path with 93 % completeness among the 6 possible combinations of the start point, target point, and position of obstacles. The proposed GA, it can be argued, can be used to locate the shortest path in a warehouse with different start and end points.

Keywords – genetic algorithm, mobile robot, shortest path, static environment

I. Introduction

A mobile robot is a type of robot used to help smooth work in the industry, such as receiving, picking, sorting, and packing [1]. To solve the task of carrying goods quickly, precisely, and safely, a mobile robot must be able to identify the environment's map, plan its paths, dynamically respond to its surroundings, and avoid obstacles autonomously [2], [3]. Path planning is the process of choosing the quickest route for mobile robots to take from their starting point to their desired location. Path planning becomes a key challenge for robots when faced with various static and moving obstacles that can be placed anywhere in a large warehouse. For many applications, including navigation systems, it is essential to have algorithms for determining the shortest distance between two points. By finding the shortest path, mobile robots can save time and use as little energy as possible. Moreover, in a dynamic environment, the robot must continuously detect changes in the position of obstacles and apply its algorithms to avoid them and find the shortest path to the target point. As described in [4], four performance criteria can be used to assess the performance of these algorithms. These criteria are completeness, optimum, time complexity, and space complexity. The ability of the algorithm to identify the entire route connecting the start and the destination node is known as completeness. Meanwhile, optimum performance is defined as finding the best course with the least effort or expense. The amount of required computation to discover the best path is called time complexity. When determining the best path, space complexity refers to the overall amount of computational memory being used.

Path planning can be either local planning [5] or global planning [6]. While global planning is based on prior knowledge of the workplace, local planning is used in an unknowable environment. The shortest distance between the start and target points is ideal for both planning scenarios. The optimal path's definition, though, may change circumstances. For instance, the amount of required time to compute the best path is important if the tasks are sequentially given to the robots. To achieve the optimal path, it will be challenging to guarantee task continuity when the computation time is too long. Therefore, researchers should adopt the appropriate algorithm for obtaining the desired optimal criterion.

The Breadth-First Search (BFS) [7], Depth-First Search (DFS), Best-First Search, and Dijkstra algorithms [8] are the most popular classical methods for path planning, enabling identification of the shortest path between nodes in a graph and grid-based structures or known as single-source shortest path (SSSP) problem [9]. Greedy BFS [10] and A* algorithm [11], [12] are two heuristic, goal-oriented search algorithms that are faster than the aforementioned algorithm. However, they can only come up with solutions for situations involving single-source, single-target pathfinding. Sampling-based techniques, such as the rapidly exploring random tree (RRT) algorithm [13], are another group of popular strategies, particularly in robotics. RRT-based techniques work well in continuous spaces but are less effective than Dijkstra or A* on grids or in challenging situations like mazes [14]. Additionally, classical search algorithms can give the best robot path. Still, they might not be appropriate for larger and more complex environments and demand a lot of processing time and memory space. Besides, heuristic search algorithms such as Greedy-BFS and RRT can also give path solutions, but they may not always come up with the best answers.

To solve the path planning problems faced when using classical and heuristic algorithms, metaheuristic optimization algorithms are employed. Particle Swarm Optimization (PSO), for instance, is used in conjunction with other algorithms to solve multi-objective path planning issues for mobile robots in radioactive environments [15] or to get smooth path planning [16]–[18]. A simulated Annealing algorithm was used by [19] for dynamic path planning. The shortest path length and the fewest number of turn times were obtained using the Ant Colony Optimization method (ACO) [20]–[22]. However, the main issues with these methods are their high computing complexity and the possibility of them being trapped at local optima.

In addition, the Genetic Algorithm (GA) approach appears to be a promising method to address these issues, as it is a multidimensional search technique capable of simultaneously handling multiple solutions. One metaheuristic algorithm that draws inspiration from biological evolution is GA. Based on Darwin's theory of the survival of the fittest in nature, GA is a well-recognized evolutionary algorithm. J.H. Holland first proposed GA in 1992. Chromosome representation, fitness selection, and biologically inspired operators like crossover, mutation, and selection make up the fundamental components of GA.

According to a study by Lamini *et al.* [23], an improved crossover operator can be used with genetic algorithms in a static environment to solve path-planning issues. The number of turns in the ideal path and the average iteration numbers are decreased in comparison to other methods when applied to 4 different maps of varying sizes. The recommended crossover operator delays early convergence and generates viable paths with higher fitness values than its parent paths, causing the algorithm to converge more quickly.

A genetic algorithm is also used by De Camargo *et al.* [24] to attain more than one optimal path. Using a 20 $\times$ 20 grid environment, there are four various paths generated by the algorithm. The research findings indicate that a low number of individuals in a genetic algorithm result in a non-converge solution to the problem or it may take many generations to converge. Conversely, larger populations are more likely to promote convergence, although it should be noted that convergence is slower with larger populations. Then, genetic algorithms can learn and produce suitable paths without any past understanding of the environment.

Elhoseny *et al.* [25] combines Bezier curve-based approach and a modified genetic algorithm (MGA) for path planning in several fields. Using MGA, the best smooth path that minimizes the overall distance between the start and finish locations is found by looking for the points that will serve as the control points of the Bezier curve. This research uses two scenarios, namely a 100 $\times$ 100 environment and a 200 $\times$ 200 environment with some obstacles and six benchmark maps. This approach provides the shortest path length based on the research's findings.

Four new domain knowledge-based operators were added to the genetic algorithm by Sarkar *et al.* [26] to overcome the path planning issue with single independent targets. The method was deployed on several environments, namely six mid-scale environments with 15 $\times$ 15 grids and large-scale environments with 50 $\times$ 50 grids. According to the experimental findings, the method delivered superior outcomes in terms of consistently determining the shortest path across all scenarios.

However, researchers [23]–[26] applied GA to specific maps of environments with certain sizes and several obstacles permanently placed so that if the designed GA is applied in different work environments, it will not necessarily get the same results. This study aimed to simulate the proposed GA in a warehouse-like environment with randomly placed obstacles. GA is implemented to measure two performance criteria, namely completeness and optimum. However, that study has several weaknesses, such as the environment being in the form of 5 $\times$ 10 grids, the number of obstacles being two, which can be placed anywhere in the environment, and the maximum population size being 100.

In this paper, the related research is presented initially, followed by the proposed algorithm's design,

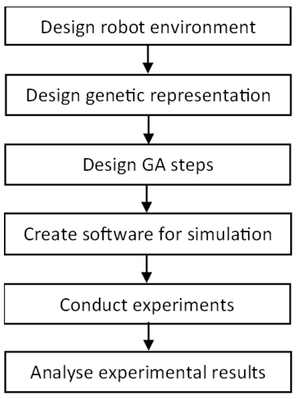
Design robot environment
Design genetic representation
Design GA steps
Create software for simulation
Conduct experiments
Analyse experimental results

Fig. 1. Research stages.

the experiment's findings, and discussion. The last part consists of a conclusion and recommendation related to the research results.

II. Research Method

The research stage is illustrated in Fig. 1. The research starts with designing the robot working environment, designing the gene representation, and designing the GA steps. After all the designs were completed, software was created to simulate the GA with Visual Studio. After the software is completed, experiments are carried out and the results obtained from the experiments are analyzed.

With each path having a single source and a single target, we have developed GA-based solutions to the path planning issue for robots. In terms of path planning, the advantages of GA are faster and more efficient compared to the classical method, can provide a list of reasonable solutions, and provides optimization on large-scale spaces. The GA algorithm starts by initializing the population, calculating fitness, implementing genetic operators consisting of tournament selection, crossover, and mutation, and applying the elitist strategy.

A. *Representation of Environment*

In this work, the robot's environment has been divided into several orderly numbered grids of the same size and shape using a grid-based decomposition technique. The environment is modeled as a rectangle with 50 points consisting of 5 rows and 10 columns. The robot will travel between these points to reach the target point. Fig. 2 shows the modeling of a robot's work environment, which has point identity information (I) in the robot's work which has 50 points (J = 50), 5 rows (M = 0 - 4), and 10 columns (N = 0 - 9). Obstacle (H) placed at a certain point will change its value: 0 for a free point and 1 for an obstacle.

Fig. 3 represents connections of a point in the work area. From one location to another, the robot can move by paying attention to the connection information (K) at the point where the robot is located. Each point (I) has four connections, consisting of:

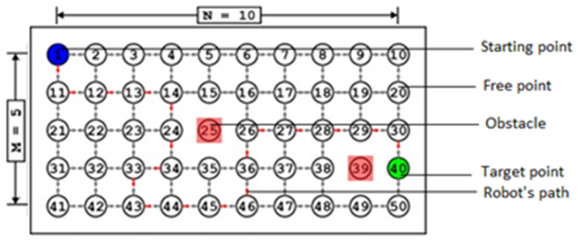

Fig. 2. Robot's environment.

- Connection to the top point ($K1$)
- Connection to the right point ($K2$)
- Connection to the bottom point ($K3$)
- Connection to the left ($K4$)

If the present point becomes an obstacle (H = 1), then the conditions for the four connections are inactive ($K1$ = 0, $K2$ = 0, $K3$ = 0, and $K4$ = 0). However, if there are no obstacles at a point (H = 0), then the connection conditions are active ($K1$ = 1, $K2$ = 1, $K3$ = 1, and $K4$ = 1). As an exception, the connection value will be assigned a value of 0 at a point on the boundaries of the work environment.

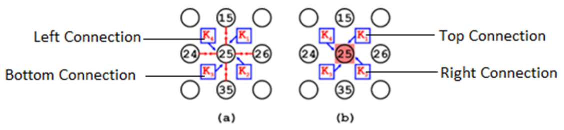

Fig. 3. Point connections (a) a free point and (b) an obstacle.

B. *Genetic Representations*

A chromosome represents a potential solution to the path planning problem. Row position (M), column position (N), obstacle (H), top connection ($K1$), right connection ($K2$), bottom connection ($K3$), and left connection ($K4$) make up a chromosome or a path. The matrix representation for each point is expressed by (1).

$$C = [M \quad N \quad H \quad K1 \quad K2 \quad K3 \quad K4] \qquad (1)$$

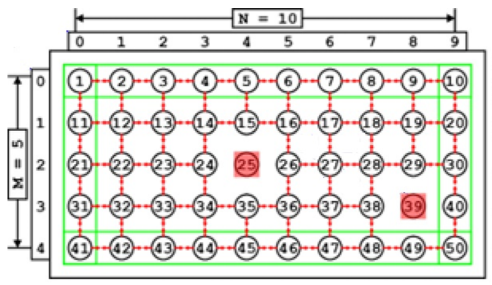

Fig. 4. Example of robot environment with two obstacles and possible connections for each point.

Fig. 4 shows all the points in the robot's environment and their possible connections. There are also two examples of obstacles placed at point 25 and point 39. In accordance with (1), the chromosome representation for point 4, which is on the edge and a free point, is [0 3 1 0 1 1 1], the chromosome representation for point 24 which is a free point, namely [2 3 1 1 0 1 1], while

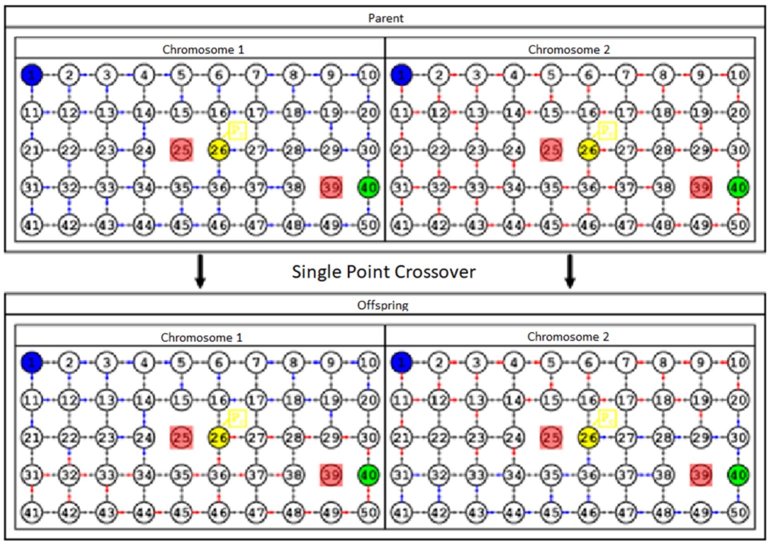

Fig. 5. An example of single point crossover.

the chromosome representation for point 25 which is an obstacle is [2 4 0 0 0 0 0].

C. Initialization of Population

In this investigation, the first population's individuals are generated at random. A generation is determined by setting the following parameters:

- Chromosomes in the form of a 50 $\times$ 7 matrix, where row indices 0 to 49 represent point identity information (1 to 50) and column indices 0 to 6 contain information of the chromosome.
- The probability of the number of obstacles (Ph) is 0.04.
- Connections to chromosomes are determined by rules of the connection has a value of 0 or is not active if a point become an obstacle or at a point on the boundaries of the work environment and the connection will be generated with a connection probability (Pk) 0.25 if a point is free.

D. Fitness Function

Finding the best path between a start node and a target node is the aim of the path planning problem. A population's chromosomal quality is assessed using the fitness function. Since distance has been used as the optimization criterion, it is necessary to minimize the fitness function (f), as stated in (2) and (3). The total distances between each node along a path are used to define the fitness function value.

$$f = \sum_{i=1}^{n} \delta(p_i, p_i + 1) \tag{2}$$

$$\delta(p_i, p_i+1) = \sqrt{(x_{(i+1)} - x_i)^2 + (y_{(i+1)} - y_i)^2} \tag{3}$$

In which f represents the fitness function, n represents the chromosome length, δ represents the distance between two points, p_i represents the ith gene of the chromosome, x_i and y_i represent the robot's current horizontal and vertical locations, while $x_{(i+1)}$ and $y_{(i+1)}$ represent robot's next horizontal and vertical locations.

E. Genetic Operators

1) Tournament selection

The GA's primary principle is that the best chromosomal genes should survive and be passed down to future generations. At this point, to choose the best chromosomes, a selection process must be carried out. There are three steps in the selection procedure. The objective function values of each chromosome are discovered in the first step. The second stage involves assigning chromosomal fitness ratings based on the values of their objective functions. The rank-based fitness assignment method was employed in this study in place of the proportional assignment method. As a result, the population is kept from having a few superior chromosomes become dominant. In the final step, chromosomes are chosen based on fitness scores and placed in a mating pool to create new chromosomes.

2) Crossover

Two-parent chromosomes exchange information with one another during the crossover process to produce two offspring for the following generation. In this study, a single-point crossover was used, with the single point selected at random from the parent chromosome with the shorter path. Fig. 5 shows an example of the crossover process of two selected parent chromosomes.

3) Mutation

A mutation process is required to produce a new gene that is not on a chromosome. The gene on the desired chromosome is the connection from the active point. A gene on a chromosome has a probability of going through a mutation process. The new gene will

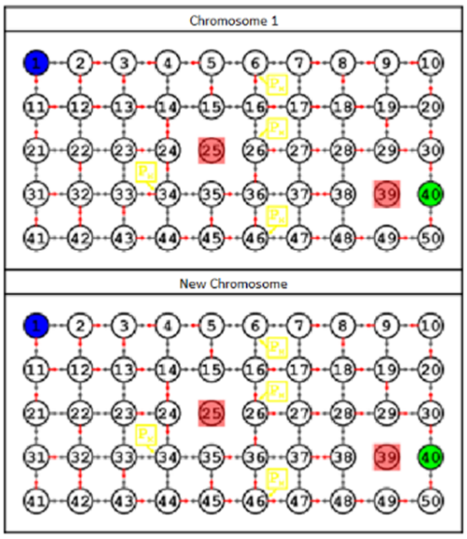

Fig. 6. An example of a mutation process.

ensure that it is not at a location with obstacles. The mutation process will change the value of the selected gene, when the gene has a value of 0 it will be changed to a value of 1, and vice versa. The result of this mutation process will produce new chromosomes in the population. Fig. 6 shows an example of a chromosome mutation process.

F. Elitist Strategy

In order to prevent the best chromosomes from the previous generation from being lost if they are not picked or if crossover or mutation modifies them, the elitist technique seeks to conserve them in the current generation.

G. Termination Condition

Although there isn't a single finishing point for GA, the process will end when there has been 400 generations total.

III. Results

This section discusses parameter and experimental settings.

A. Parameter Setting

Aside from constructing the algorithm, parameter adjustment is also essential. Investigations were done on the effects of population size and the number of iterations on the path length. The shortest path between the start and target points in the environment is used to determine the ideal parameter. In this experiment, the crossover, mutation, and elitist rates were 0.6, 0.39, and 0.01, respectively.

1) Population size

The population size or number of solutions must be sufficient to allow for the exploration of the search space. This section examined the impact of population size on the length of the proposed model's path when there were 40 iterations and 40, 60, 80, or 100 populations. Fig. 7 shows the results. As illustrated in

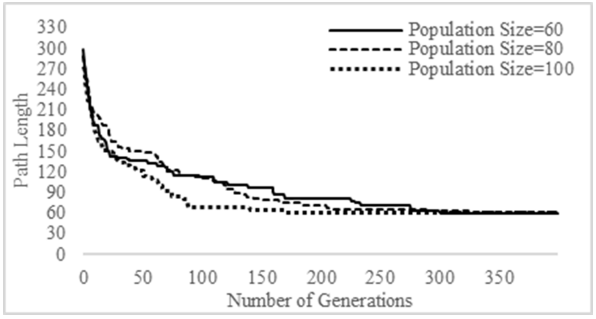

Fig. 7. Effect of population size on the proposed model's path length.

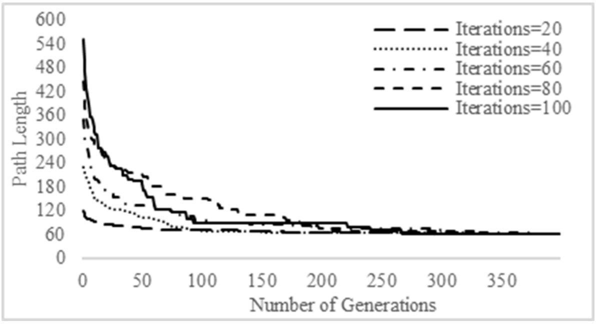

Fig. 8. Effect of iterations on the path length of the proposed model.

the figure, as the population increases, the path length will get better and converge to the optimal or nearly optimal solutions more quickly. For the experiment, we use 100 as the population size.

2) Number of iterations

The suggested model's performance is also impacted by the number of iterations. This section examined the impact of this parameter on the effectiveness of the suggested model. In this experiment, the population size was 100, while the number of iterations varied from 20, 40, 60, 80, and 100. Fig. 8 presents a summary of the experiment's findings. From the figure, the best number of iterations is 40 since it converges faster than other number of iterations.

B. Experimental Settings

We build software to simulate the proposed algorithm using Microsoft Studio. By setting up random obstacle placements in the field with various start and target places, we created six scenarios to test the effectiveness of our suggested methods. We conducted the examination ten times for each scenario, reporting the mean findings. We used 100 population size, 40 iterations, and the maximum generation was 400. Fig. 9 shows an illustration of the scenarios. The starting point is marked with a black circle, the target point is marked with a white circle, and obstacles are marked with black squares. Aside from the start point and target point distances varying from close to far apart, two environmental obstacles' locations were also altered. The result of the experiments is shown in Table 1.

IV. Discussion

In terms of the optimum path length, average path length, average number of generations, percentage of

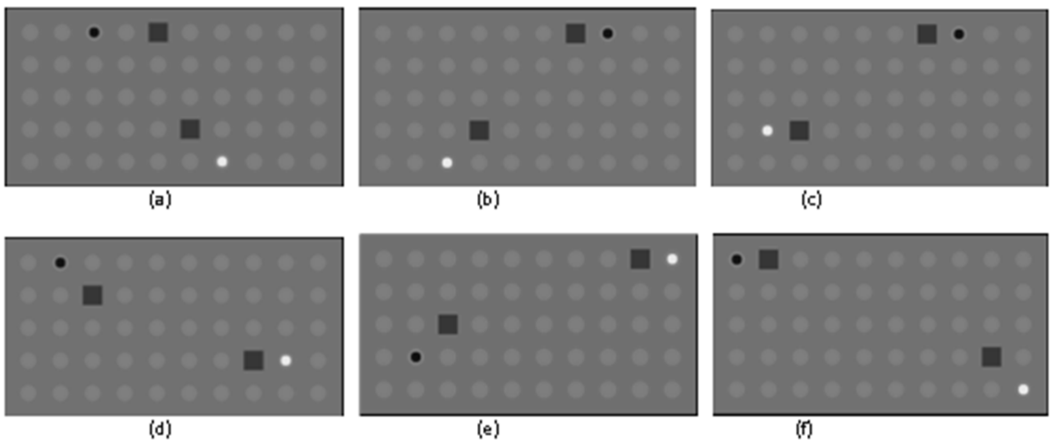

Fig. 9. Six scenarios used in the experiments with two obstacles and different start and target point (a) Scenario1, (b) Scenario2, (c) Scenario 3, (d) Scenario 4, (e) Scenario 5, and (f) Scenario 6.

Table 1. Result of Experiments for Scenario 1 - 6

Scenario	**Best Path Length**	**Average Path Length**	**Average Number of Generations**	**Percentage of Completeness**	**Number of Path Variations**
1	23.948	23.948	14.1	100 %	5
2	29.796	29.827	16.6	100 %	2
3	30.429	30.829	53.5	100 %	4
4	35.742	35.771	53.6	100 %	3
5	43.203	43.449	87.9	80 %	5
6	61.52	64.025	160.7	80 %	8

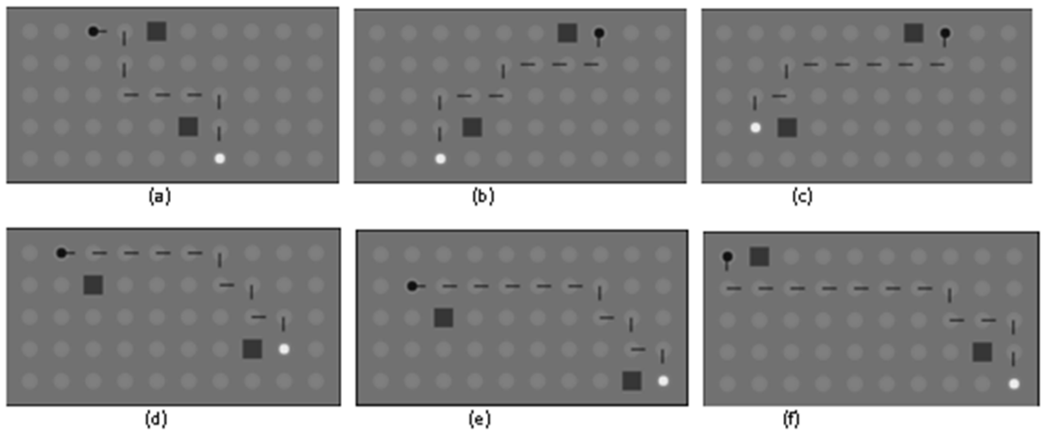

Fig. 10. The shortest path generated from experiments (a) Scenario1, (b) Scenario2, (c) Scenario 3, (d) Scenario 4, (e) Scenario 5, and (f) Scenario 6.

completeness, and number of path modifications, Table 1 shows the results of testing from scenarios 1 through 6. For scenario 1, out of 10 trials, the path lengths obtained are identical even though the resulting paths vary. The percentage of completeness is 100 %, with an average number of generations of 16.6. For scenarios 2, 3, and 4, out of 10 trials, there are 2 to 3 path length variations and two path variations with a 100 % success rate. For scenarios 5 and 6, out of 10 trials, there were eight successful trials with an average resulting path length of 43.449 and 64.025, which were not too far from the obtained best path length. For scenarios 5 and 6, the average number of generations needed for convergence is relatively high—87.9 and 160.7. Additionally, there were two failed attempts, indicating that the robot was unable to reach the point target. The results of this study follow the results of studies [23], [25], and [26], which state that GA can provide the shortest path length in various tested environments. Fig. 10 shows one of the shortest paths that has been obtained from the experiments in the six scenarios. The shortest path for each scenario is marked with a dashed black line from the starting point to the target point.

Table 1 also shows that the farther path the robot must take results in a more significant number of generations, which affects the required time to reach convergence. However, because the population initialization in GA is random, the required time to reach convergence in each trial is also different, with some of them requiring many generations and others requiring a small number of generations.

V. Conclusion

This paper proposed GA as the algorithm for finding the shortest path between the start point and the target point with a predetermined amount of obstacles in various static environments. The result of the experiments shows that the proposed GA can find the shortest path within a certain number of generations in multiple environments. The number of generations increases following the distance that the robot must travel, which impacts the period it takes to attain convergence. Since the current work only considered static obstacles, the proposed GA should be assessed for path planning in a dynamic environment with moving obstacles.

References

[1] A. Yildirim, H. Reefke, and E. Aktas, "Mobile robot automation in warehouses: A framework for decision making and integration," in *Palgrave studies in logistics and supply chain management. Cham: Imprint: Palgrave Macmillan*, 2023. doi: 10.1007/978-3-031-12307-8.

[2] N. Horňáková, L. Jurík, H. Hrablik Chovanová, D. Cagáňová, and D. Babčanová, "AHP method application in selection of appropriate material handling equipment in selected industrial enterprise," *Wirel. Netw.*, vol. 27, no. 3, pp. 1683–1691, Apr. 2021, doi: 10.1007/s11276-019-02050-2.

[3] Z. Abidin, K. Joni, and A. F. Ibadillah, "Rancang bangun robot penghindar halangan berbasis kamera menggunakan deteksi kontur," *Jurnal Infotel*, vol. 9, no. 3, pp. 248–256, Aug. 2017, doi: 10.20895/infotel.v9i3.279.

[4] M.-K. Ng, Y.-W. Chong, K. Ko, Y.-H. Park, and Y.-B. Leau, "Adaptive path finding algorithm in dynamic environment for warehouse robot," *Neural Comput. Appl.*, vol. 32, no. 17, pp. 13155–13171, Sep. 2020, doi: 10.1007/s00521-020-04764-3.

[5] D. V. Lyridis, "An improved ant colony optimization algorithm for unmanned surface vehicle local path planning with multi-modality constraints," *Ocean Eng.*, vol. 241, p. 109890, Dec. 2021, doi: 10.1016/j.oceaneng.2021.109890.

[6] X. Guo, M. Ji, Z. Zhao, D. Wen, and W. Zhang, "Global path planning and multi-objective path control for unmanned surface vehicle based on modified particle swarm optimization (PSO) algorithm," *Ocean Eng.*, vol. 216, p. 107693, Nov. 2020, doi: 10.1016/j.oceaneng.2020.107693.

[7] H. K. Tripathy, S. Mishra, H. K. Thakkar, and D. Rai, "CARE: A collision-aware mobile nobot navigation in grid environment using improved breadth first search," *Comput. Electr. Eng.*, vol. 94, p. 107327, Sep. 2021, doi: 10.1016/j.compeleceng.2021.107327.

[8] I. P. Sari, M. F. Fahroza, M. I. Mufit, and I. F. Qathrunad, "Implementation of dijkstra's algorithm to determine the shortest route in a city," *J. Comput. Sci. Inf. Technol. Telecommun. Eng.*, vol. 2, no. 1, pp. 134–138, Mar. 2021, doi: 10.30596/jcositte.v2i1.6503.

[9] M. Amr, A. Bahgat, H. Rashad, and A. Ibrahim, "Comparison of path planning between improved informed and uninformed algorithms for mobile robot," *Int. J. Adv. Comput. Sci. Appl.*, vol. 13, no. 6, pp. 239–248, 2022, doi: 10.14569/IJACSA.2022.0130629.

[10] D. Xiang, H. Lin, J. Ouyang, and D. Huang, "Combined improved A* and greedy algorithm for path planning of multi-objective mobile robot," *Sci. Rep.*, vol. 12, no. 1, p. 13273, Aug. 2022, doi: 10.1038/s41598-022-17684-0.

[11] B. Wang, "Path planning of mobile robot based on A* algorithm," in *2021 IEEE International Conference on Electronic Technology, Communication and Information (ICETCI)*, Changchun, China: IEEE, Aug. 2021, pp. 524–528. doi: 10.1109/ICETCI53161.2021.9563354.

[12] V. Suryani, K. Agustriana, A. Rakhmatsyah, and R. R. Pahlevi, "Room cleaning robot movement using A* algorithm and imperfect maze," *Jurnal Infotel*, vol. 15, no. 1, pp. 75–81, Mar. 2023, doi: 10.20895/infotel.v15i1.901.

[13] H. Ryu and Y. Park, "Improved informed RRT* using gridmap skeletonization for mobile robot path planning," *Int. J. Precis. Eng. Manuf.*, vol. 20, no. 11, pp. 2033–2039, Nov. 2019, doi: 10.1007/s12541-019-00224-8.

[14] J. Pak, J. Kim, Y. Park, and H. I. Son, "Field evaluation of path-planning algorithms for autonomous mobile robot in smart farms," *IEEE Access*, vol. 10, pp. 60253–60266, 2022, doi: 10.1109/ACCESS.2022.3181131.

[15] D. Zhang, Y. Yin, R. Luo, and S. Zou, "Hybrid IACO-A*-PSO optimization algorithm for solving multiobjective path planning problem of mobile robot in radioactive environment," *Prog. Nucl. Energy*, vol. 159, p. 104651, May 2023, doi: 10.1016/j.pnucene.2023.104651.

[16] S. Dian, J. Zhong, B. Guo, J. Liu, and R. Guo, "A smooth path planning method for mobile robot using a BES-incorporated modified QPSO algorithm," *Expert Syst. Appl.*, vol. 208, p. 118256, Dec. 2022, doi: 10.1016/j.eswa.2022.118256.

[17] L. Xu, M. Cao, and B. Song, "A new approach to smooth path planning of mobile robot based on quartic Bezier transition curve and improved PSO algorithm," *Neurocomputing*, vol. 473, pp. 98–106, Feb. 2022, doi: 10.1016/j.neucom.2021.12.016.

[18] B. Song, Z. Wang, and L. Zou, "An improved PSO algorithm for smooth path planning of mobile robots using continuous high-degree Bezier curve," *Appl. Soft Comput.*, vol. 100, p. 106960, Mar. 2021, doi: 10.1016/j.asoc.2020.106960.

[19] K. Shi, Z. Wu, B. Jiang, and H. R. Karimi, "Dynamic path planning of mobile robot based on improved simulated annealing algorithm," *J. Frankl. Inst.*, vol. 360, no. 6, pp. 4378–4398, Apr. 2023, doi: 10.1016/j.jfranklin.2023.01.033.

[20] L. Wu, X. Huang, J. Cui, C. Liu, and W. Xiao, "Modified adaptive ant colony optimization algorithm and its application for solving path planning of mobile robot," *Expert Syst. Appl.*, vol. 215, p. 119410, Apr. 2023, doi: 10.1016/j.eswa.2022.119410.

[21] W. Hou, Z. Xiong, C. Wang, and H. Chen, "Enhanced ant colony algorithm with communication mechanism for mobile robot path planning," *Robot. Auton. Syst.*, vol. 148, p. 103949, Feb. 2022, doi: 10.1016/j.robot.2021.103949.

[22] C. Miao, G. Chen, C. Yan, and Y. Wu, "Path planning optimization of indoor mobile robot based on adaptive ant colony algorithm," *Comput. Ind. Eng.*, vol. 156, p. 107230, Jun. 2021, doi: 10.1016/j.cie.2021.107230.

[23] C. Lamini, S. Benhlima, and A. Elbekri, "Genetic algorithm based approach for autonomous mobile robot path planning," *Procedia Comput. Sci.*, vol. 127, pp. 180–189, 2018, doi: 10.1016/j.procs.2018.01.113.

[24] J. T. F. De Camargo, E. A. F. De Camargo, E. V. Veraszto, G. Barreto, J. Cândido, and P. A. Zibordi Aceti, "Route planning by evolutionary computing: an approach based on genetic algorithms," *Procedia Comput. Sci.*, vol. 149, pp. 71–79, 2019, doi: 10.1016/j.procs.2019.01.109.

[25] M. Elhoseny, A. Tharwat, and A. E. Hassanien, "Bezier curve based path planning in a dynamic field using modified genetic algorithm," *J. Comput. Sci.*, vol. 25, pp. 339–350, Mar. 2018, doi: 10.1016/j.jocs.2017.08.004.

[26] R. Sarkar, D. Barman, and N. Chowdhury, "Domain knowledge based genetic algorithms for mobile robot path planning having single and multiple targets," *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 34, no. 7, pp. 4269–4283, Jul. 2022, doi: 10.1016/j.jksuci.2020.10.010.