International Journal of Robotics and Control Systems
Vol. 5, No. 1, 2025, pp. 407-425
ISSN 2775-2658
http://pubs2.ascee.org/index.php/ijrcs

Improving Efficiency and Effectiveness of Wheeled Mobile Robot Pathfinding in Grid Space Using a Genetic Algorithm with Dynamic Crossover and Mutation Rates

Dyah Lestari [a,1], Siti Sendari [a,2,*], Ilham Ari Elbaith Zaeni [a,3], Samsul Arifin [b,4], Rina Dewi Indah Sari [b,5]

[a] Universitas Negeri Malang, Jalan Semarang No. 5, Malang, 65145, Indonesia
[b] Institut Teknologi dan Bisnis Asia Malang, Jalan Rembuksari No. 1A, Malang 65113, Indonesia
[1] dyah.lestari.2205349@students.um.ac.id; [2] siti.sendari.ft@um.ac.id; [3] ilham.ari.ft@um.ac.id; [4] samsul@asia.ac.id; [5] rinadewi@asia.ac.id
* Corresponding Author

ARTICLE INFO

Article history
Received August 14, 2024
Revised December 22, 2024
Accepted January 10, 2024

Keywords
Wheeled Mobile Robot;
Pathfinding;
Grid Space;
Genetic Algorithm;
Dynamic Crossover and Mutation Rates

ABSTRACT

Incorrect parameter tuning of crossover and mutation rates in Genetic Algorithms (GA) can negatively impact their effectiveness and efficiency in mobile robot pathfinding. This study focuses on improving the performance of wheeled mobile robots in grid-based environments by introducing a Dynamic Crossover and Mutation Rates (DCMR) strategy within the GA framework. The primary contribution of this research is enhancing the efficiency and effectiveness of mobile robot pathfinding, resulting in shorter average path lengths and faster convergence times. Additionally, this method addresses the challenge of selecting appropriate GA parameters while increasing the algorithm's adaptability to different phases of the search process. The DCMR approach involves linearly increasing the crossover rate by 10% (from 0% to 100%) and decreasing the mutation rate by 10% (from 100% to 0%) over every 10 generations during the GA's evolution. Unlike fixed parameter tuning or exponential and sigmoid parameter tuning—both of which require trial and error to determine optimal values—the DCMR method provides a systematic and efficient solution without additional computational cost. Experiments were conducted across eight scenarios featuring varying distances between the start and target points, with two obstacles randomly placed in the robot's environment. The results showed that implementing the DCMR method consistently identified the optimal path, reduced average path lengths by 0.99%, and accelerated algorithm convergence by 48.39% compared to fixed parameter tuning. These findings demonstrate that the DCMR method significantly enhances the performance of GAs for mobile robot pathfinding, offering a reliable and efficient approach for navigating complex environments.

1. Introduction

The application of pathfinding in autonomous robots has been utilized in various fields [1]-[7]. Pathfinding is crucial for mobile robots because it enables them to navigate their environment and reach specific destinations without human intervention [8]-[15]. Some key reasons why pathfinding

is important are that pathfinding allows robots to move autonomously through complex environments, effective pathfinding ensures that robots take the most efficient routes, a good pathfinding algorithm helps robots avoid obstacles, and pathfinding involves both global and local planning.

Pathfinding in mobile robots faces several significant challenges, such as unstructured environments [16], collision avoidance [17], excessive turns [18], dynamic and real-time adaptation [19], dan computational complexity [20]. Genetic Algorithms (GA) can help address these pathfinding challenges by optimising paths over generations to adapt to unpredictable terrains, evolving diverse path options, smoothing out paths to reduce unnecessary turns, by rapid re-computation, and by using hybrid approaches it can reduce computational complexity while maintaining pathfinding efficiency [18], [21].

Numerous researchers have used GA in mobile robot pathfinding investigations by modelling it as a Travelling Salesman Problem (TSP) [22]-[25]. Other researchers modelled the robot environment in the form of a grid [26]-[29]. Research conducted by [28], [30]-[33] all modified the GA or combined the GA with other methods for mobile robot pathfinding. In addition to pathfinding, GA is also used for solving a variety of challenging optimization issues. Researchers across a wide range of fields, including computer networks [34], software engineering [35], image processing [36], healthcare [37], educational games [38], machine learning [39], essentially used GA extensively all the time. In GA, without any mathematical explanation, optima developed from generation to generation. It is a discontinuous, non-linear process not guided by mathematics. GA became a significant tool to resolve issues with various solutions and intricate search spaces. This enabled the resolution of various optimization problems and the discovery of multiple solution structures, particularly in parameter tuning.

For mobile robot pathfinding, parameter tuning in Genetic Algorithms (GA) is crucial due to its significant impact on the algorithm's efficiency and effectiveness. Several problems related to parameter tuning are population size, crossover and mutation rates, and fitness function. The relationship between these GA parameters directly impacts solution quality. Maintaining balanced parameter values will enhance GA performance. Incorrect these parameter settings can result in convergence to a sub-optimal solution or slow down the evolutionary process. Crossover and mutation rates are two parameters that must be carefully chosen when designing GA. Crossover helps introduce new variations in the population by combining genes from two different individuals, allowing the algorithm to explore new solution regions and increase genetic diversity. Mutation is done by changing a small portion of an individual chromosome, thus introducing new traits that can help in finding a better optimal solution [29]. Improper crossover rates might fail to generate diverse offspring, limiting the exploration of new paths. Similarly, low mutation rates can result in slow convergence and stagnation, while high mutation rates can disrupt good solutions.

Some researchers who conducted pathfinding research with mobile robots solved the tuning parameter problem through fixed parameter tuning [21], [40]-[43] by setting different values for crossover rates and mutation rates. Other studies set the tuning parameters exponentially or sigmoidally [44], [45] by setting a certain range of values for crossover and mutation rates. Although these studies provide good results in mobile robot pathfinding performance, the use of fixed parameters for crossover and mutation rates in mobile robot pathfinding can cause several shortcomings such as premature convergence, limited diversity, and non-optimal performance [9], [18], [46]. Similar to fixed parameter tuning, exponential parameter tuning for crossover and mutation rates in mobile robot pathfinding can lead to several drawbacks such as overemphasis on exploration or exploitation, complexity in tuning because it requires careful calibration, increased computational load, and risk of instability due to sudden shifts in mutation and crossover rates [47]-[49]. Both methods still require a trial-and-error process to find the right value or range of crossover and mutation rates. On the other hand, at different stages in the pathfinding process, different parameter values may be more suitable.

To overcome the problem of determining the right parameters and improving the efficiency and effectiveness of GA in mobile robot pathfinding, in this study, dynamic parameter tuning is applied

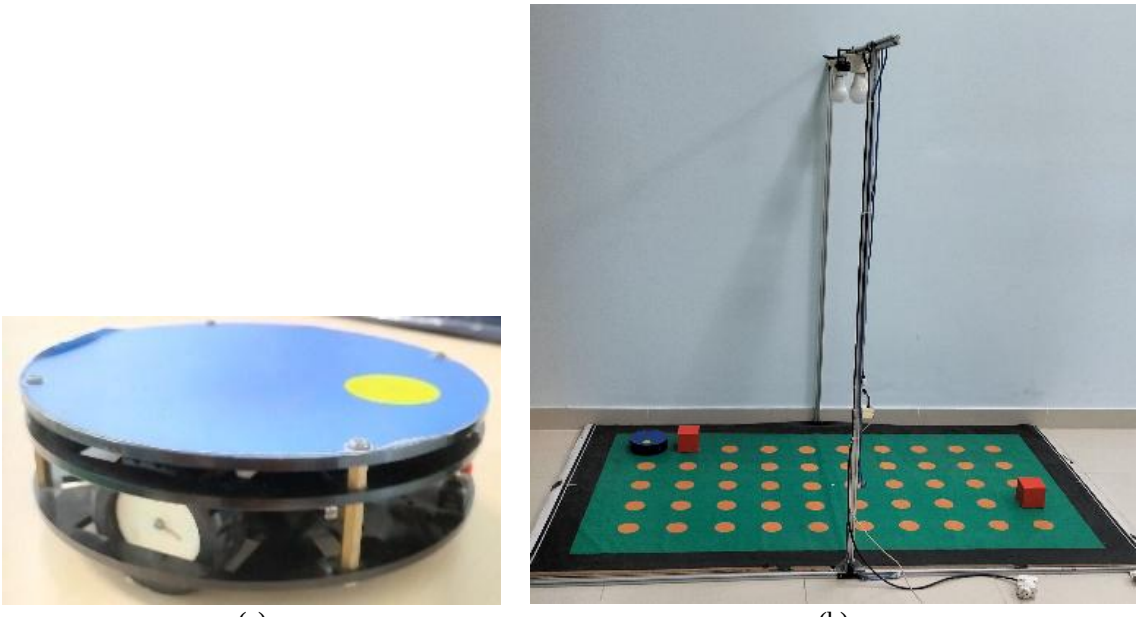
by applying linear changes in crossover and mutation rates every 10 generations that will not cause additional computational cost. This research aims to improve mobile robot pathfinding performance by dynamically adjusting crossover and mutation rates within the GA in terms of completeness, path lengths, and convergence time. The main contribution of this research is demonstrating that dynamically adjusting crossover and mutation rates in GA improves pathfinding efficiency and effectiveness, resulting in shorter average path lengths and faster convergence. The second contribution of the research is that this study addresses the challenge of selecting the right parameters for GA and making the algorithm more adaptable to different stages of the search process. Robots will perform better because of this research when it comes to determining routes and obtaining the shortest routes between the start and target points at different distances. This paper starts with the relevant research and then moves on to the proposed methodology, examines the experiment's results, and its discussion. A summary and suggestions on the study's findings make up the final section.

2. Methods

2.1. Environmental Modelling

The result of this research will be tested with a wheeled mobile robot so that the area of the robot's working environment is limited according to the existing hardware. As can be seen in Fig. 1 (a), the mobile robot used is tubular with a diameter of 15 cm and height of 5 cm and has 2 motors to drive 2 motion wheels. The top of the robot is coloured blue. The small yellow circle at the top of the robot is used to determine the front of the robot.

The robot's working environment is designed according to the size of the robot, which is a flat field with an active area measuring 175 cm × 95 cm which is coloured green. To simplify the representation of the robot's working area and obstacles, the real robot's working environment is created in the form of a grid [50], [51]. The grid-based model was chosen because of its simplicity and effectiveness in handling various obstacles and providing a straightforward framework for applying optimization algorithms. The working area was divided into 50 grids consisting of 5 rows and 10 columns with a distance between the grid midpoints of 16 cm. Each grid is marked with an orange circle with a diameter of 7 cm. The obstacle is made in the form of a red cube with a side length of 8 cm. The camera is placed at the top of the centre of the robot environment 150 cm above the robot's working area. The robot's real working environment can be seen in Fig. 1 (b).



(a) (b)

Fig. 1. Environmental modelling (a) the wheeled mobile robot and (b) the real working environment of the robot

In this system, the camera will capture images of the robot's work area, then determine the coordinates of each grid in the robot's work area, the start point of the robot, the position of the obstacles to be processed into the GA on the computer which will find the shortest path for the robot to traverse to the destination point. After the algorithm finds the shortest path, the robot will receive a command from the computer to move towards the destination point.

In this study, to reduce the complexity of navigation, help avoid collisions with obstacles, and be efficient in pathfinding, in the developed software, the working environment of the robot is modelled as in Fig. 2. It can be seen in Fig. 2, that the region the robot can move through is indicated by the white grid, the obstacles in the environment are indicated by the red grid, the start point position is indicated by the blue grid, and the goal point location is indicated by the yellow grid. The primary purpose of assigning numbers to the grids is to facilitate the coding of solutions in post-sequence algorithm programs that represent the ultimate solution path. A global static path that is ultimately planned and shown on the grid as a set of numbers.

The robot environment is represented as fifty grids (J=50), which consist of five rows (M=0-4) and ten columns (N = 0-9). If a grid becomes an obstacle (red grid), then O=1, and if the grid is free (white grid), then O=0. The connections for each grid in the environment are shown in Fig. 3. The robot can move by focusing on the connection information (C) in the location it is to determine its next step. CT is the connection to the top grid, CR is the connection to the right grid, CB is the connection to the bottom grid, and CL is the connection to the left grid. Every free grid (like in Fig. 3 (a)) has value 1 for CT, CR, CB, and CL, except for the topmost grid, CT=0, the rightmost grid, CR=0, the bottommost grid, CB=0, and the leftmost grid, CL=0. For a red grid or an obstacle like in Fig. 3. (b), CT, CR, CB, and CL become 0. Since the number of connections provided is only 4, while from one point is only allowed to go to one other point, the possible connections allowed are 1/4 or 0.25. If a grid is not an obstacle, then a connection in that grid will be generated with a connection probability (P_{CO} = 0.25).

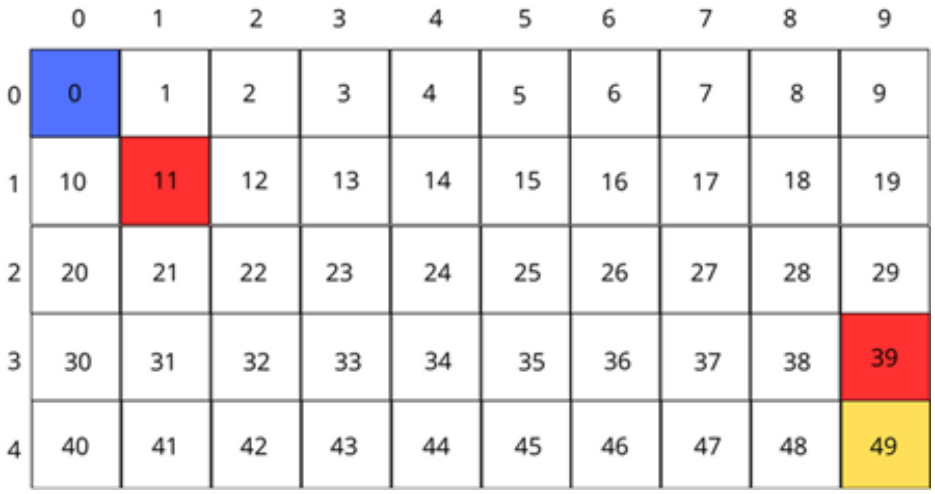


Fig. 2. Grid map of the robot environment

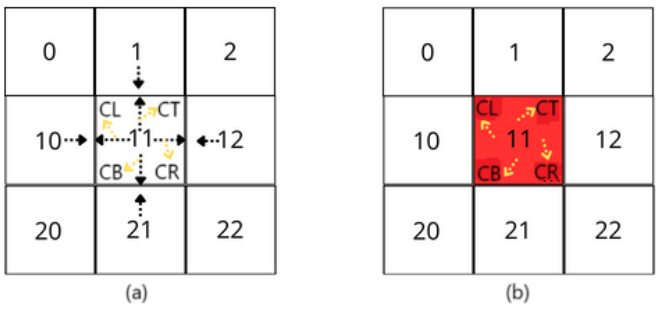


Fig. 3. Connection conditions for (a) a free grid and (b) an obstacle

2.2. Genetic Representation

All the information is available for each grid in the robot's working environment and represented in a matrix with 50 rows (J=50) and 7 columns (D=7) where the seven columns contain information about M (0-4), N (0-9), O (0-1), CT (0-1), CR (0-1), CB (0-1) and CL (0-1) as shown in (1). The formation of genetic representation in the matrix is used to provide information on the direction of the path that may be travelled by the mobile robot and to avoid obstacles.

$$I_{[J,D]} = \begin{bmatrix} i_{00} & i_{01} & i_{02} & \cdots & i_{0d} \\ i_{00} & i_{01} & i_{02} & \cdots & i_{1d} \\ & & \vdots & & \\ i_{j0} & i_{j1} & i_{j2} & \cdots & i_{jd} \end{bmatrix} \quad (1)$$

For example, using the working environment in Fig. 3 (b), grid number 1 which is a free grid will give a genetic representation [0 1 0 0 1 0 1] where the numbers in the first and second columns indicate the position of the grid in the 0th row and 1st column, then the number in the third column is given the value 0 because the grid is free. The numbers in the 4th through 7th columns indicate no connection above because the grid is at the upper boundary of the work environment, the connection on the right is 1, the connection below is 0 because there is an obstacle below, and the connection on the left is 1. Another example, grid number 11 which is an obstacle has a genetic representation of [1 1 1 0 0 0 0] where the numbers in the first and second columns indicate the position of the grid in the 1st row and 1st column, then the number in the third column is given a value of 1 if the grid is an obstacle. The numbers in the 4th through 7th columns indicate there are no connections above, the right, below, and the left because the grid is an obstacle.

2.3. Fitness Function

The pathfinding problem seeks to identify the shortest route between the target and start points. With the fitness function, the chromosomal quality of a population is evaluated. When the GA generates chromosomes randomly, each chromosome will be calculated for its fitness function. Each gene will be calculated as its distance from the target point using Euclidean distance with (2). Euclidean distance was chosen as the metric for the fitness function because it allows the GA to calculate the shortest path without being affected by orientation and direction constraints. Equation (3) will sum the distance of each gene in a chromosome into a fitness function. To obtain the shortest distance, the chromosome that has the minimum fitness function must be selected.

$$\delta(p_i) = \sqrt{(x_n - x_i)^2 + (y_n - y_i)^2} \quad (2)$$

$$f = \sum_{i=1}^{n \leq 50} \delta(p_i) \quad (3)$$

The distance between two points is denoted by δ, the i^{th} gene of the chromosome by p_i, the position of i^{th} gene horizontal and vertical locations by x_i and y_i, and the target horizontal and vertical locations by x_n and y_n, and the fitness function is denoted by f. The maximum number of genes in a chromosome is limited to the number of grids, which is 50. For example, as in Fig. 4, if the start point is grid 0, and the target point is grid 42, and a chromosome gene generates paths 0-10-20-30-40-41-42, the length of the chromosome is 7. The fitness value or f for this example can be seen in (4).

$$f = \sqrt{20} + \sqrt{13} + \sqrt{8} + \sqrt{5} + \sqrt{4} + \sqrt{1} + \sqrt{0} = 16{,}142 \quad (4)$$

2.4. Genetic Operator

The genetic operator is part of the GA shown in Fig. 5. In this study, the population size was set to 100 based on the results of previous studies which showed that a population of 100 was sufficient to produce optimal and rapidly convergent paths.

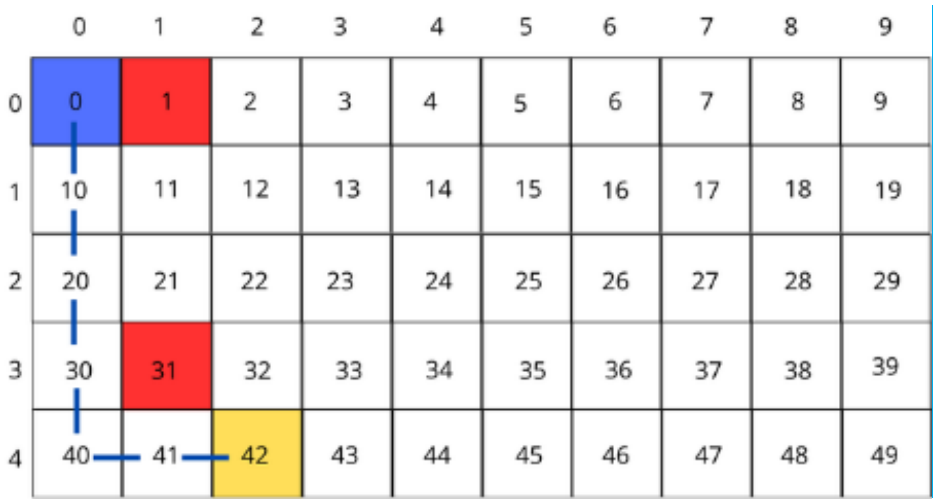


Fig. 4. An example of a chromosome that gives a path from grid 0 to grid 42

After population initialization as generation 0, a fitness function will be calculated to sort the chromosomes in the population from those with the closest distance to the target to those with the farthest distance to the target. Starting from generation 1, the genetic operator will be applied for each generation: (1) selection of two chromosomes that become parents using the tournament selection method with the probability of chromosome selection (P_S = 0.1/5 of 50 chromosome) to pass the crossover process, resulting in a new chromosome for the population in the next generation based on the probability of crossover (P_C), (2) crossover process of two chromosomes selected as parents using the one point crossover method (1 of 50 gene chosen as the point) which has a probability (P_{CG} = 0.02), (3) mutation process of chromosomes in the population to produce new chromosomes for the population in the next generation based on the probability of mutation (P_M) with the probability of selecting genes that pass the mutation process (P_{MG} = 0.1/5 of 50 gene will changed), (4) chromosome elitism process to maintain chromosomes with the best fitness function using the rank selection method for the population in the next generation based on the elitism probability (P_E = 0.02/1 of 50 chromosome). After running the genetic operator, GA will continue with the calculation of the fitness function, and the formation of the current generation population. The GA will stop when the number of generations reaches maximum generation.

2.5. Dynamic Approach of Crossover and Mutation Rate (DCMR)

In this study, DCMR is applied where crossover rates are gradually increased, and mutation rates are gradually decreased as the GA evolves. This approach helps balance exploration and exploitation, which are critical to GA's success. Early exploration helps the algorithm search across a broader solution space, avoiding traps in local optima. This phase uses high mutation to introduce novelty and low crossover to reduce premature refinement. Late exploitation focuses on polishing the solutions by recombining strong candidates (high crossover) while minimizing disruptive randomness (low mutation). This phase ensures convergence to an optimal or near-optimal solution. Along with the crossover rate increases from 0% to 100%, the mutation rate decreases every 10 generations from 0% to 100%. The decrease in crossover rate and the increase in mutation rate were made equal at 10%. Upon algorithm startup, the mutation rate (P_M), gene mutation rate (p_{MG}), and crossover rate (P_C) are adjusted linearly depending on the number of generations in total as well as the number of current generations by (5), (6), and (7). CG denotes the number of generation levels (the current generation, when the ratio is determined), G_n represents the total number of generations, and p_{MG} represents the probability of a gene being selected through the mutation process.

$$P_M = \frac{CG}{G_n} \tag{5}$$

$$p_{MG} = P_{MG}\ x\ P_M \tag{6}$$

$$P_C = 1 - \frac{CG}{G_n} \tag{7}$$

The rates for the GA run are shown in Table 1. According to Table 1, the crossover rate increases by 10% at every 10-generation level, reaching 100% at the last generation. At every 10-generation level, starting from 100%, the mutation rate is decreased by 10% to achieve 0% at the end of the GA cycle. The crossover rate at each generation is equal to the complement of the mutation rate, as shown by (5) and (7). At the same generation level, the crossover rate plus the mutation rate adds up to 100%. The goal is to dynamically increase population variety by using changes in crossover and mutation rates.

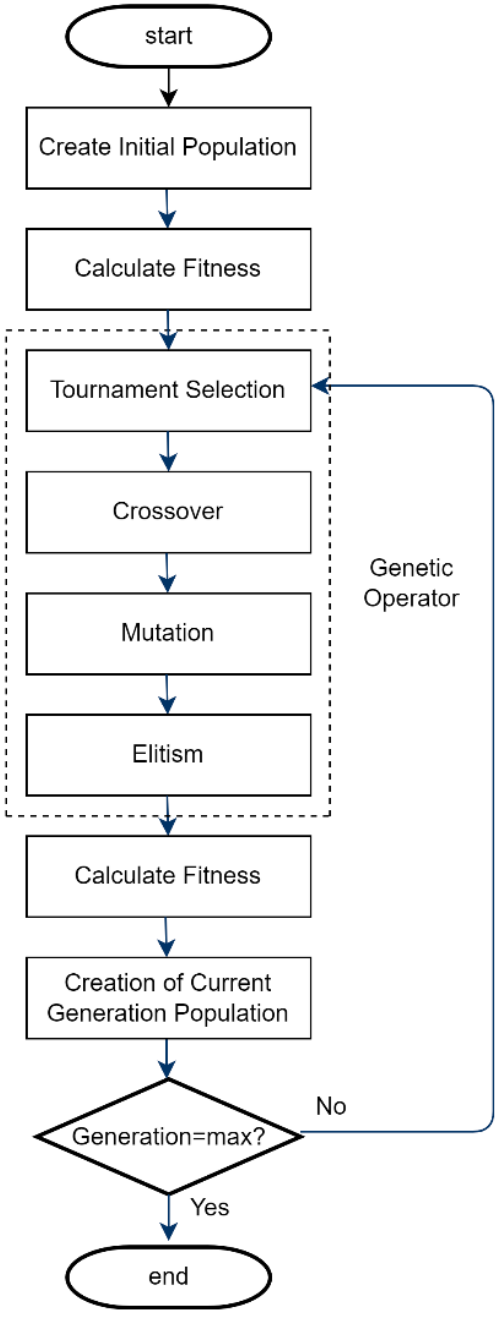


Fig. 5. Flowchart of designed GA

2.6. Fixed Parameter Tuning Method (0.9PC0.03PM)

For comparison, the fixed parameter tuning method is implemented. As mentioned in the introduction, researchers apply different crossover and mutation rates in their GA designs using the try-and-error method. In this research, a fixed value for crossover rates is $P_C = 0.9$, as used by several researchers [21], [44]. For the mutation rate we choose $P_M = 0.03$, which is used by several researchers [45]. For every generation level of the experiment, there is an equal number of mutant chromosomes and crossover participants.

Table 1. Crossover, mutation, and mutation gen rates during the GA run

Generation	P_C (%)	P_M (%)	p_{MG} (%)
1-9	0	100	10
10-19	10	90	9
20-29	20	80	8
30-39	30	70	7
40-49	40	60	6
50-59	50	50	5
60-69	60	40	4
70-79	70	30	3
80-89	80	20	2
90-99	90	10	1
100-109	100	0	0

2.7. Experimental Method

We use Microsoft Visual Studio C# to create software to simulate the algorithm. Using a robot working environment consisting of 50 grids of 5 rows and 10 columns, the GA algorithm was tested with 8 variations of scenarios to see if the DCMR helps find paths faster with varying start-target distances and different obstacles placed. The distance between the start point and the target point from one scenario to the next increases by 1 grid to the right. Scenario 1 has the shortest distance between the start point and the target point, while Scenario 8 has the longest distance between the start point and the target point. The target point is on the left if the start point is on the right, and vice versa. In addition, if the target point is at the bottom, the start point is at the top. Two obstacles were relocated, in addition to the start point and target point distances changing from near to far apart. Varying the start-target distances incrementally across scenarios will determine how effective the algorithm is under varying levels of complexity. Placing the target on the opposite side of the grid (left-right or top-bottom) ensures the pathfinding algorithm navigates the entire grid space. Moving obstacles between scenarios creates a dynamic and realistic simulation environment. To make it easier to understand the experimental setup, an illustration of the 8 scenarios which have different positions of start point, target point, and obstacles is presented in Fig. 6. A blue grid is used to indicate the start point, a yellow grid is used to indicate the target points and red grids are used to indicate obstacles.

The number of steps used is equal to the number of grids in the robot's working environment. According to previous research [40], a population of 100 was sufficient to generate optimal and quickly converging pathways. The maximum number of generations is set according to the decrease in crossover and increase in mutation every 10 generations.

For every scenario, we ran the analysis ten times and reported the average results. Ten runs provide a balance between resource efficiency and ensuring robust results. The four performance criteria used to assess the performance of path-finding algorithms are completeness, optimality, time complexity, and space complexity [52]. We consider the completeness, optimality in the form of best path length, and time complexity represented by the number of generations to convergence. After conducting experiments, first, we analyse the maximum fitness, the minimum fitness, and the average fitness for every scenario. Then, the completeness, best path length, average and standard deviation of path length and number of generations to converge data for all scenarios from ten runs after implementation of DCMR and 0.9PC0.03PM are shown in a table. Next, the average path length and

the average number of generations to converge for all scenarios after the implementation of DCMR and 0.9PC0.03PM were tested for differences using a t-test. To see how much difference between the application of the DCMR method and the 0.9PC0.03P method in the designed GA on the average path length and average number of generations of Scenario 8, Cohen's d effect size is calculated.

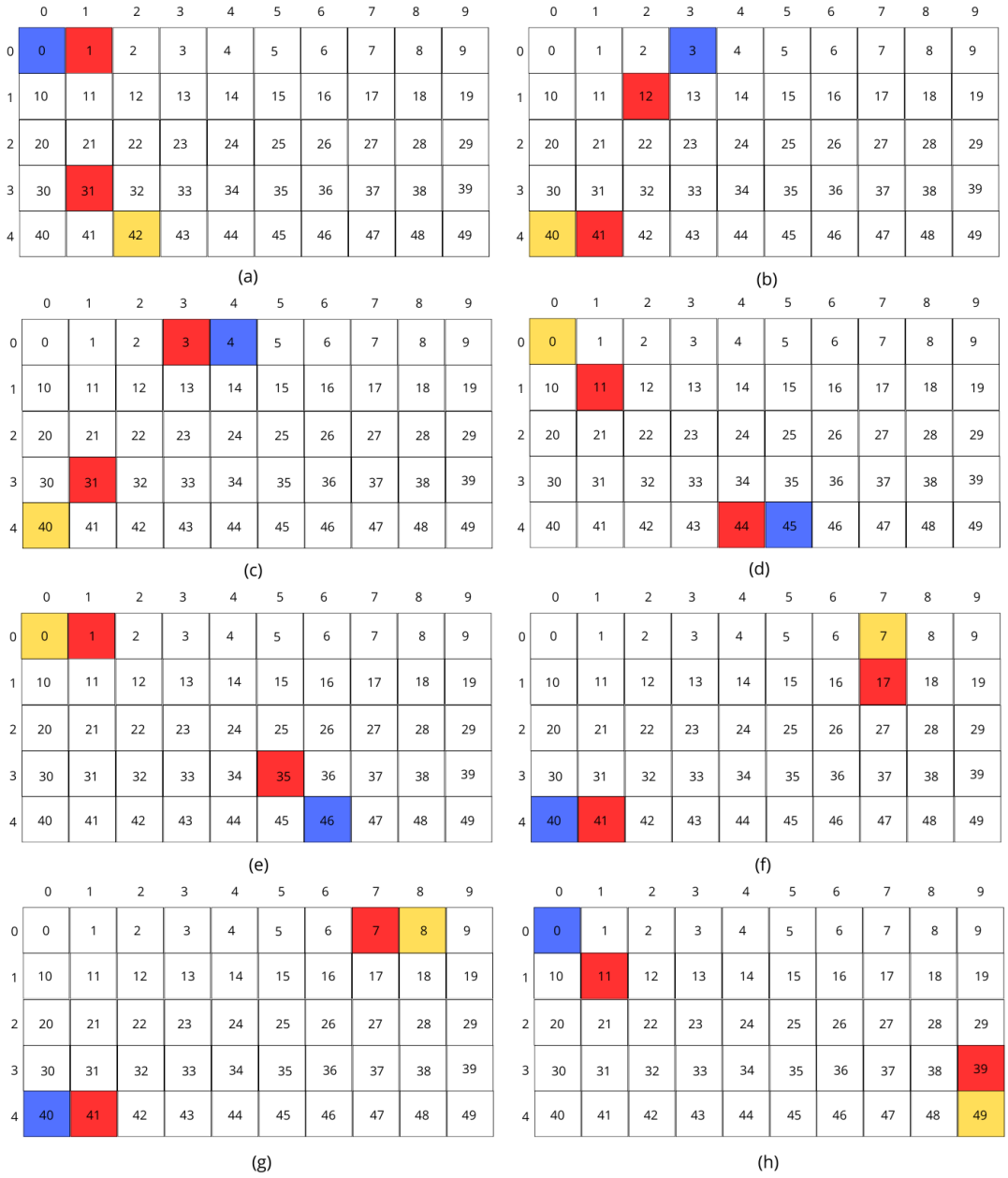


Fig. 6. Eight scenarios were used in the experiments (a) scenario 1, (b) scenario 2, (c) scenario 3, (d) scenario 4, (e) scenario 5, (f) scenario 6, (g) scenario 7, and (h) scenario 8

For the experiment, we utilized 50 steps, a population size of 100, and a maximum generation of 109.

3. Results and Discussion

3.1. Main Findings of The Present Study

As mentioned in the method section, the fitness function directly impacts the path evaluation. For example, in Fig. 7. with the same start point and target point in Scenario 3, the fitness values obtained

for different chromosomes can be different. Fig. 7 (a) shows a shorter path with a fitness value of 26.779 compared to Fig. 7 (b) which shows a longer path with a fitness value of 33.567. From the path evaluation results, the path with the smaller fitness value will be selected as the better path.

Fig. 8 shows the maximum fitness, minimum fitness, and average fitness values for Scenarios 3 and 6. As seen in Fig. 8 (a) and Fig. 8 (b), the maximum fitness can vary because the paths generated by the algorithm are random, meaning that the population varies. Since the population evolves in each generation, as can be seen in Fig. 8 (a) and Fig. 8 (b), the minimum fitness value or best path length tends to become lower as the generation increases until it converges in a certain generation. This is due to the implementation of all genetic operators so that there will be more population variations that can reach the target point. The average fitness is also in line with the minimum fitness, which tends to decrease from generation to generation because more populations have fitness values that are closer to the minimum fitness value.

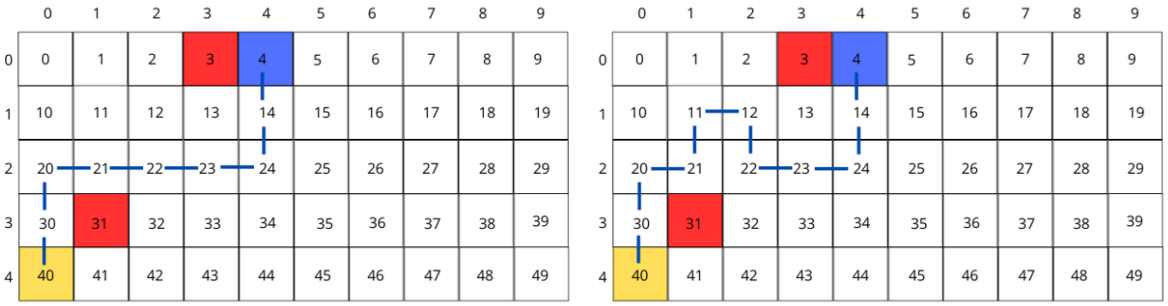


Fig. 7. Comparison of fitness value from scenario 3 (a) shorter path with fitness value of 26,799, (b) longer path with fitness value of 33,567

Table 2 shows the completeness, best path length from 8 scenarios, the mean and standard deviation of path length and number of generations to converge for 10 runs of pathfinding performance when GA employs DCMR and 0.9PC0.03PM. The experimental results show that both methods can 100% find the target points in all scenarios. The best path length obtained by applying both methods is also the same, meaning that the dynamic approach of crossover and mutation rate and the fixed parameter tuning method get the same best path for all scenarios. The best path length for 8 scenarios can be seen in Fig. 9. Although the results of applying the DCMR and 0.9PC0.03PM methods obtained the same completeness and best path length in all scenarios, the average path length and convergence by applying the DCMR method were slightly better than applying 0.9PC0.03PM.

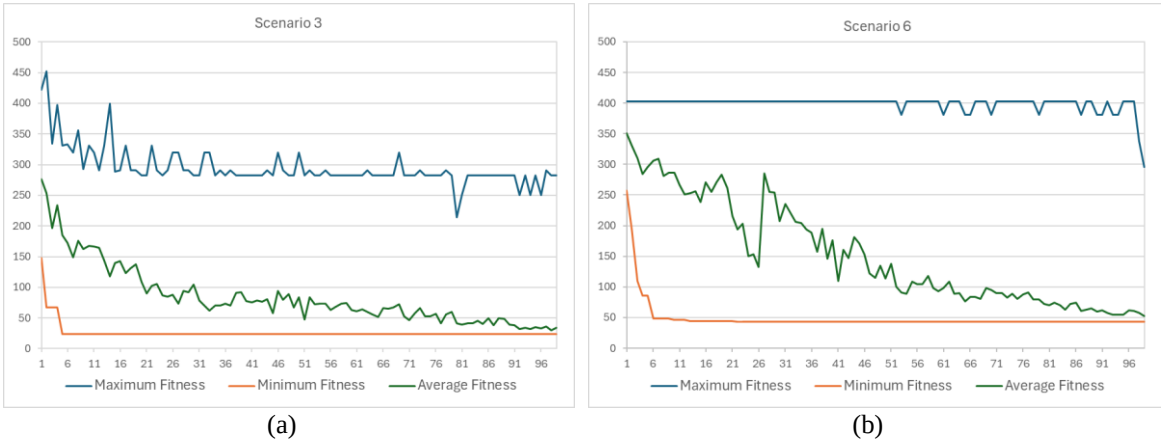


Fig. 8. Maximum fitness, minimum fitness, and average fitness values for (a) scenario 3 and (b) scenario 6

Fig. 10 (a) compares the average path length with DCMR and 0.9PC0.03PM implementation. From 10 trials, the average path length obtained when implementing DCMR is mostly slightly lower

than when implementing 0.9PC0.03PM. In Fig. 10 (b), for all scenarios, the average number of generations required to find the path from the start point to the target point is lower when applying DCMR than when applying 0.9PC0.03PM.

Table 2. Performance of DCMR and 0.9PC0.03PM on pathfinding

Scenario	Method	Complete-ness (%)	Best Path Length	Path Length		Number of Generations to Converge	
				Average	Standard Deviation	Average	Standard Deviation
1	DCMR	100	14.283	14.283	0.000	0.8	0.789
	0.9PC0.03PM	100	14.283	14.283	0.000	1.4	1.838
2	DCMR	100	18.202	18.228	0.081	2.9	2.025
	0.9PC0.03PM	100	18.202	18.339	0.353	3.5	3.808
3	DCMR	100	23.948	23.948	0.000	2.1	2.234
	0.9PC0.03PM	100	23.948	23.948	0.000	3.9	3.071
4	DCMR	100	29.796	29.907	0.146	6	2.828
	0.9PC0.03PM	100	29.796	30.005	0.287	16.3	14.945
5	DCMR	100	35.309	35.503	0.412	8.8	6.033
	0.9PC0.03PM	100	35.309	35.481	0.399	15.5	7.169
6	DCMR	100	43.472	43.739	0.367	5.2	2.530
	0.9PC0.03PM	100	43.472	44.815	2.233	15.7	5.851
7	DCMR	100	51.788	52.471	1.534	11.8	4.492
	0.9PC0.03PM	100	51.788	53.935	3.676	34.4	18.216
8	DCMR	100	58.752	59.030	0.440	14.4	8.553
	0.9PC0.03PM	100	58.752	60.194	2.214	24.8	12.647

The results of testing the difference in pathfinding performance using DCMR and 0.9PC0.03PM for Scenario 8 on the average path length and average number of convergent generations can be seen in Table 3. In Table 3, the difference in experimental results is also tested with t-test and Cohen's d effect size.

3.2. Comparison with Other Studies

The A*, D*, RRT, and GA are widely used approaches for mobile robot pathfinding, each with distinct strengths and limitations. A* is a graph-based heuristic search algorithm that guarantees optimal paths in static and structured environments. It is efficient for known maps but struggles with large-scale or dynamic scenarios due to high computational overhead [53]. D* extends A* for dynamic and partially unknown environments, adapting to changes in real-time. While more flexible than A*, it has increased complexity and computational cost. RRT (Rapidly-exploring Random Tree) excels in high-dimensional, unstructured spaces and rapidly finds feasible paths. However, it does not guarantee optimality and is less suited for structured environments [54]. On the other hand, GA employs evolutionary optimization principles, making it adept at solving complex and multi-objective path-planning problems in dynamic and unpredictable environments. GAs can handle non-linear problems but are computationally intensive and sensitive to parameter tuning, such as mutation and crossover rates [46]. GA can overcome the high computational cost and optimality issues faced when applying A*, D*, and RRT by applying appropriate parameter tuning methods [55]. The DCMR method applied in this study does not require high computational cost, but the path length obtained can be optimized with faster convergence time as shown in Table 2.

In GA, parameter tuning strategies such as fixed and exponential or sigmoid can significantly impact optimization performance. Fixed parameter tuning applies mutation and crossover rates constant throughout the algorithm's execution. It lacks the flexibility to adapt to different stages of the optimization process, which may result in suboptimal performance for complex problems [56], [57]. Exponential tuning adjusts parameters using an exponential function which parameters change rapidly at the beginning of the algorithm and stabilize later. This strategy helps in quickly exploring the solution space early on and focuses on fine-tuning as the algorithm progresses [44], [45]. DCMR

method is linear parameter tuning which modifies parameters in a steady, incremental manner over generations. This method balances exploration and exploitation effectively and doesn't need a trial-and-error process to find the right value or range of crossover and mutation rates.

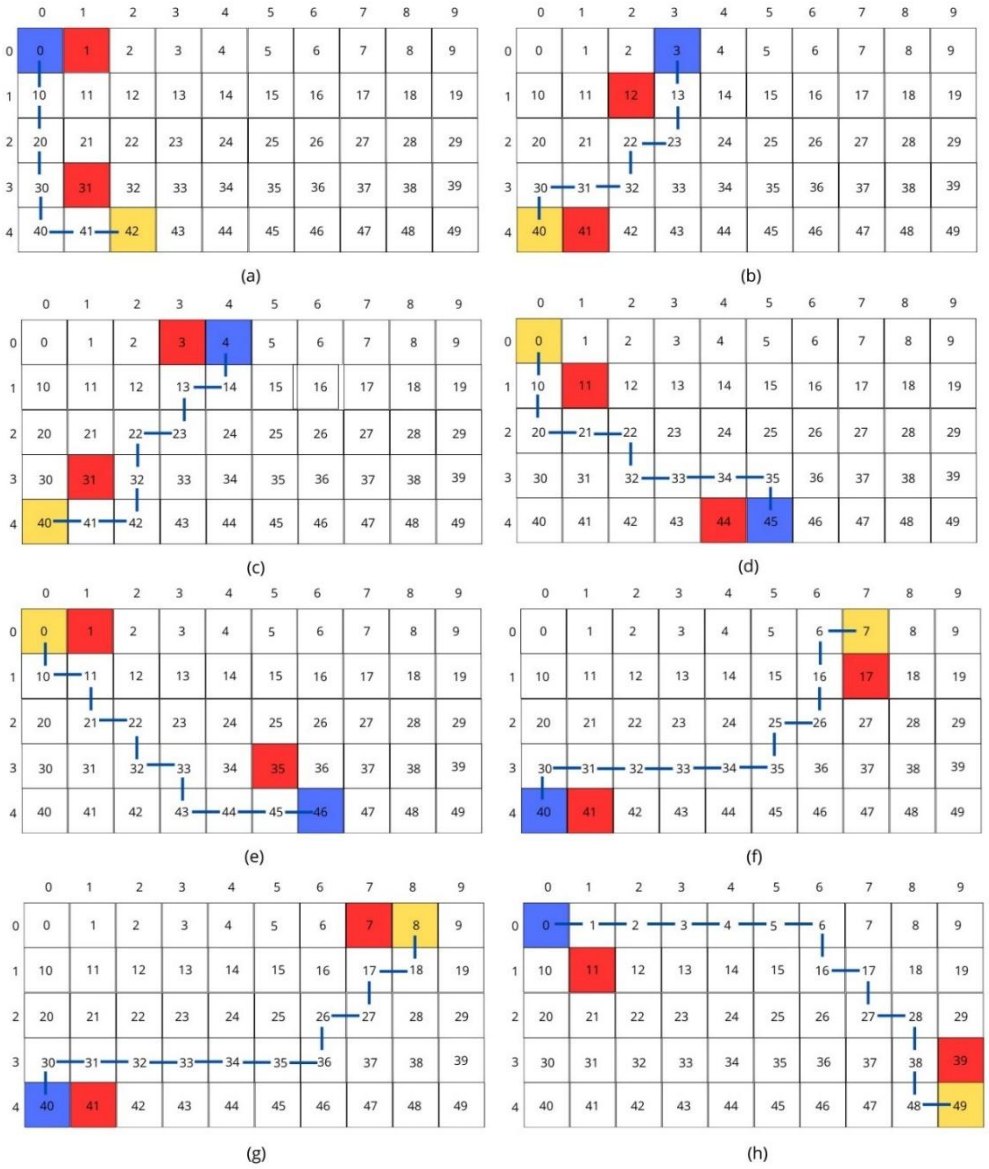
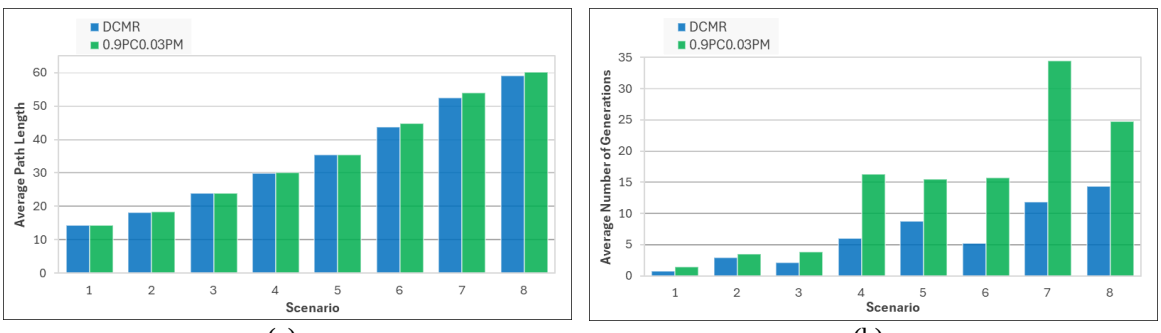


Fig. 9. Best path length obtained for (a) scenario 1, (b) scenario 2, (c) scenario 3, (d) scenario 4, (e) scenario 5, (f) scenario 6, (g) scenario 7, and (h) scenario 8



(a) (b)

Fig. 10. Comparison of path finding performance by applying DCMR and 0.9PC0.03PM (a) average path length and (b) average number of generations

Table 3. The difference test result of pathfinding performance by applying DCMR and 0.9PC0.03PM for Scenario 8 on (a) average path length and (b) average number of generations to converge

(a)		
Method	**DCMR**	**0.9PC0.03PM**
Mean	59.030	60.194
Standard Deviation	0.440	2.214
p-value	-	0.071
Cohen's d	-	0.729
(b)		
Method	**DCMR**	**0.9PC0.03PM**
Mean	14.4	24.8
Standard Deviation	8.553	12.647
p-value	-	0.006
Cohen's d	-	0.963

3.3. Implication and Explanation of Findings

The maximum fitness shown in Fig. 8 shows varying values because the population is generated by the GA randomly. Randomness in GA is crucial as it ensures diversity within the population, leads to broader solution spaces, and improves the chances of finding a global optimum. Good quality randomness prevents premature convergence to suboptimal solutions. In Scenario 3 (Fig. 9 (a)), the highest value of maximum fitness is 452.276 where the generated path is circuitous and hence the fitness is high. In Scenario 6 (Fig. 9 (b), where the start point is at grid 40 and one of the obstacles is at grid 41, the highest value of maximum fitness is 403.113 which appears in many chromosomes in many generations because the start point is in the lower left corner and there is an obstacle to the right, no path is generated, meaning the robot will remain at the start point. This condition also occurs in Scenario 1 and Scenario 7 which have the same start point conditions as Scenario 6.

The conditions seen in Scenarios 1, 6 and 7, show that the placement of obstacles greatly affects the effectiveness of the pathfinding algorithm because obstacles in strategic locations can increase the number of points that need to be evaluated by the algorithm, slowing down the pathfinding process [58]. In addition, complex or adjacent obstacles force the algorithm to find alternative routes, which may be longer or non-optimal [59].

The completeness results shown in Table 2, which reached 100%, show better results compared to those obtained in previous research [40] where for Scenario 6 to Scenario 8, 2 out of 8 trials could not find the target point. The best path obtained when applying DCMR and 0.9PC0.03PM obtained the same results because the mutation rates used in both approaches range from 0 to 0.1, which will aid in boosting population variety and preventing premature convergence. The fact that the optimum path length found using both approaches is the same value serves as evidence for this. This shows that the DCMR method can be applied when using GA algorithms in real-world applications.

It is shown in Fig. 10 (a), that of the 8 scenarios, 6 scenarios have small different average path lengths between the application of DCMR and 0.9PC0.03PM in GA. The result of Table 3 (a) shows that the t-test results for average path length from 10 trials for Scenario 8 obtained a p-value of 0.071. This p-value is greater than the significance level (α) of 0.05, so the path lengths obtained by applying DCMR and 0.9PC0.03PM are the same. Nevertheless, in scenario 8, the average path length that implemented the 0.9PC0.03PM method is 0.729 standard deviations greater than the average path length that implemented the DCMR method or there is a medium effect size of the DCMR method to the average path lengths.

For every scenario shown in Fig. 10 (b), the average number of generations needed to determine the path from the start point to the target point is less when DCMR is applied than when 0.9PC0.03PM is applied. This result is also supported by the t-test results of the number of generations to converged of the DCMR and 0.9PC0.03PM methods for Scenario 8 in Table 3 (b) which shows a p-value of 0.006. Since the p-value obtained is smaller than the significance level (α) of 0.05, it can be concluded that when applying the DCMR method, the proposed GA algorithm finds target points faster than

when applying the 0.9PC0.03PM method. In addition, in Scenario 8, the average number of generations to converge for GA using the 0.9P method is 0.963 standard deviations higher than the number of generations to converge for GA using the DCMR method, or the DCMR method has a large impact on the average number of generations to converge. Practically, it validates that the DCMR method improves the GA's performance, enabling faster convergence in pathfinding.

Genetic operators play a key role in determining the performance and behaviour of GA in terms of solution quality and convergence. The crossover operator combines genetic material from two parents to create offspring, potentially producing better solutions by merging advantageous traits. The crossover operator increases exploration of the solution space and prevents local optima traps when well-tuned. The mutation operator introduces random changes to individual solutions to maintain diversity in the population [60]. It helps avoid premature convergence by exploring new areas of the solution space. Applying crossover and mutation operators produces high-quality solutions that have better fitness which leads to faster convergence.

Fitness value convergence in GA is influenced by multiple factors such as population diversity and selection pressure. High population diversity ensures the exploration of a larger solution space, avoiding premature convergence and local optima. Moderate selection balances exploitation and exploration, promoting stable and effective convergence.

In accordance with other research [61], [62], applying the DCMR method means applying high mutation rates and low crossover rates at the beginning of GA which promotes exploration of the solution space, preventing premature convergence and helping to discover diverse candidate solutions. This is beneficial when the initial population lacks sufficient diversity. As the algorithm progresses, lowering the mutation rate and increasing the crossover rate emphasizes exploitation, allowing the algorithm to refine and converge toward an optimal or near-optimal solution by combining the best traits of the fittest individuals. This dynamic adjustment balances exploration and exploitation, improving GA performance by adapting to the problem's needs during different evolutionary stages and leading to faster convergence. With the combination of dynamic crossover and mutation rates throughout the generations, it is expected that with a different robot environment that is larger and has a different number and placement of obstacles, it is expected that the robot can find a path quickly.

Since the DCMR approach requires fewer iterations to get an optimal or nearly optimal solution, its faster convergence suggests that it has computational efficiency that can result in energy efficiency. The DCMR method's marginally improved average path lengths result in more effective routes in path-planning applications, which can lower delivery times and mobile robot operating expenses.

3.4. Strengths and Limitations

Although the DCMR method applied in the GA gave good results in this study, it should be further improved or tested when applied in more complex environments because too frequent adjustments can destabilize the search, causing the GA to lose diversity or fail to effectively explore promising areas. Applying the GA with larger populations may also cause the DCMR method to have little effect as larger populations already maintain diversity, reducing the need for high mutation rates [63].

4. Conclusions

To dynamically modify the crossover and mutation rate parameters, we present a unique deterministic approach in the design of GA and parameter selection in GA. The crossover rate and mutation rate parameters vary linearly without human input or feedback, according to the DCMR mathematical equation. Ten trials in eight different robot work environment settings show that the DCMR method majority outperforms 0.9PC0.03PM in pathfinding. Both methods produced the same completeness and optimal path length in every scenario. Scenario 5-8 obtained a better average shortest path when applying the DCMR method. The time to convergence achieved by implementing the DCMR method in GA was somewhat superior to those acquired by using the 0.9PC0.03PM

method without additional computational costs. In comparison to the 0.9PC0.03PM method, results in every scenario showed that applying the DCMR method in GA consistently found the best path, shortened average path lengths by 0.99%, and sped up algorithm convergence by 48.39%. DCMR method can reduce computing overhead making it useful in applications like robotics and autonomous vehicle navigation where making decisions in real time is crucial.

By using the DCMR approach in GA, the problem of choosing the appropriate GA parameters was resolved and pathfinding effectiveness and efficiency were increased. This approach will produce efficient routes and computational efficiency for mobile robot path planning. The practical implications of DCMR's advancements in convergence speed and path optimisation are multifaceted, spanning industries like robotics, logistics, healthcare, and telecommunications.

It is important to conduct a comparative analysis between the path-finding performance achieved through GA and that achieved through other mobile robot path-finding techniques. Several pathfinding techniques such as A*, D*, RRT, and PSO can be compared with GA to see which technique performs better, highlights the robustness of each technique in handling certain problems, and looks at the possibility of combining these techniques to solve certain problems. DCMR method in GA scales effectively to larger and more complex environments due to their adaptability and focus on balancing exploration and exploitation. However, challenges such as computational overhead and the need for precise parameter tuning must be addressed. Future research should continue to explore and combine the DCMR approach with other methods, particularly in dynamic and complex environments, to fully unlock its potential in optimising pathfinding and other applications.

Author Contribution: Dyah Lestari: Conceptualization, Methodology, Investigation, Formal analysis, Writing – original draft; Siti Sendari: Supervision, Writing – review and editing, Validation; Ilham Ari Elbaith Zaeni: Supervision, Writing – review and editing, Validation; Samsul Arifin: Software; Rina Dewi Indah Sari: Visualization.

Funding: This research was funded by Universitas Negeri Malang (UM).

Acknowledgements: The authors would like to acknowledge UM which provides financial support for this research activity in 2023 and UPT Publikasi Ilmiah UM which is organizing Writing Warrior Camp in 2024.

Conflicts of Interest: The authors declare no conflict of interest.

References

[1] S. Sendari *et al*., "Grasping and Repositioning Objects Using Inverse Kinematic Method for Arm Robot Based on Pixel Position Regression," *Journal of Advanced Manufacturing Technology (JAMT)*, vol. 16, no. 3, 2024, https://jamt.utem.edu.my/jamt/article/view/6409

[2] S. Tahmasebi, N. Rasouli, A. H. Kashefi, E. Rezabeyk, and H. R. Faragardi, "SYNCOP: An evolutionary multi-objective placement of SDN controllers for optimizing cost and network performance in WSNs," *Computer Networks*, vol. 185, p. 107727, 2021, https://doi.org/10.1016/j.comnet.2020.107727.

[3] S. Sendari *et al*., "Dual-Stage Identifying Data of Arm Robot for Recognizing and Sorting Objects with Double Faces Permitted-Prohibited Area," *Journal of Advanced Manufacturing Technology (JAMT)*, vol. 18, no. 1, 2024, https://jamt.utem.edu.my/jamt/article/view/6635.

[4] S. Sendari *et al*., "Movement of Trolley Robot using Fuzzy Logic Control and Camshift Algorithms in Following Similar Object," *International Journal of iRobotics*, vol. 5, no. 2, pp. 10–15, 2022, https://iroboticsjournal.org/index.php/irobotics/article/view/125.

[5] W. Dawid and K. Pokonieczny, "Methodology of Using Terrain Passability Maps for Planning the Movement of Troops and Navigation of Unmanned Ground Vehicles," *Sensors*, vol. 21, no. 14, p. 4682, 2021, https://doi.org/10.3390/s21144682.

[6] Z. A. Adeola-Bello and N. Z. Azlan, "Power Assist Rehabilitation Robot and Motion Intention Estimation," *International Journal of Robotics and Control Systems*, vol. 2, no. 2, pp. 297-316, 2022, https://doi.org/10.31763/ijrcs.v2i2.650.

[7] R. Sukwadi *et al*., "Comparative Analysis of Path Planning Algorithms for Multi-UAV Systems in Dynamic and Cluttered Environments: A Focus on Efficiency, Smoothness, and Collision Avoidance," *International Journal of Robotics and Control Systems*, vol. 4, no. 4, pp. 1602-1616, 2024, http://dx.doi.org/10.31763/ijrcs.v4i4.1555.

[8] J. R. Sánchez-Ibáñez, C. J. Pérez-del-Pulgar, and A. García-Cerezo, "Path Planning for Autonomous Mobile Robots: A Review," *Sensors*, vol. 21, no. 23, p. 7898, 2021, https://doi.org/10.3390/s21237898.

[9] L. Liu, X. Wang, X. Yang, H. Liu, J. Li, and P. Wang, "Path planning techniques for mobile robots: Review and prospect," *Expert Systems with Applications*, vol. 227, p. 120254, 2023, https://doi.org/10.1016/j.eswa.2023.120254.

[10] Z. Yu, Z. Si, X. Li, D. Wang and H. Song, "A Novel Hybrid Particle Swarm Optimization Algorithm for Path Planning of UAVs," *IEEE Internet of Things Journal*, vol. 9, no. 22, pp. 22547-22558, 2022, https://doi.org/10.1109/JIOT.2022.3182798.

[11] C. Miao, G. Chen, C. Yan, and Y. Wu, "Path planning optimization of indoor mobile robot based on adaptive ant colony algorithm," *Computers & Industrial Engineering*, vol. 156, p. 107230, 2021, https://doi.org/10.1016/j.cie.2021.107230.

[12] C. Qu, W. Gai, J. Zhang, and M. Zhong, "A novel hybrid grey wolf optimizer algorithm for unmanned aerial vehicle (UAV) path planning," *Knowledge-Based Systems*, vol. 194, p. 105530, 2020, https://doi.org/10.1016/j.knosys.2020.105530.

[13] M. M. J. Samodro, R. D. Puriyanto, and W. Caesarendra, "Artificial Potential Field Path Planning Algorithm in Differential Drive Mobile Robot Platform for Dynamic Environment," *International Journal of Robotics and Control Systems*, vol. 3, no. 2, pp. 161-170, 2023, http://dx.doi.org/10.31763/ijrcs.v3i2.944.

[14] M. G. Mohanan and A. Salgaonkar, "Robotic Motion Planning in Dynamic Environments and its Applications," *International Journal of Robotics and Control Systems*, vol. 2, no. 4, pp. 666-691, 2022, https://doi.org/10.31763/ijrcs.v2i4.816.

[15] A. N. Jasim and L. Chaari Fourati, "Guided Genetic Algorithm for Solving Capacitated Vehicle Routing Problem With Unmanned-Aerial-Vehicles," *IEEE Access*, vol. 12, pp. 106333-106358, 2024, https://doi.org/10.1109/ACCESS.2024.3438079.

[16] L. Wijayathunga, A. Rassau, and D. Chai, "Challenges and Solutions for Autonomous Ground Robot Scene Understanding and Navigation in Unstructured Outdoor Environments: A Review," *Applied Sciences*, vol. 13, no. 17, p. 9877, 2023, https://doi.org/10.3390/app13179877.

[17] K. Katona, H. A. Neamah, and P. Korondi, "Obstacle Avoidance and Path Planning Methods for Autonomous Navigation of Mobile Robot," *Sensors*, vol. 24, no. 11, p. 3573, 2024, https://doi.org/10.3390/s24113573.

[18] H. Qin, S. Shao, T. Wang, X. Yu, Y. Jiang, and Z. Cao, "Review of Autonomous Path Planning Algorithms for Mobile Robots," *Drones*, vol. 7, no. 3, p. 211, 2023, https://doi.org/10.3390/drones7030211.

[19] L. Yang *et al*., "Path Planning Technique for Mobile Robots: A Review," *Machines*, vol. 11, no. 10, p. 980, 2023, https://doi.org/10.3390/machines11100980.

[20] M. Reda, A. Onsy, A. Y. Haikal, and A. Ghanbari, "Path planning algorithms in the autonomous driving system: A comprehensive review," *Robotics and Autonomous Systems*, vol. 174, p. 104630, 2024, https://doi.org/10.1016/j.robot.2024.104630.

[21] N. S. Abu, W. M. Bukhari, M. H. Adli, and A. Ma'arif, "Optimization of an Autonomous Mobile Robot Path Planning Based on Improved Genetic Algorithms," *Journal of Robotics and Control*, vol. 4, no. 4, pp. 557–571, 2023, https://doi.org/10.18196/jrc.v4i4.19306.

[22] D. Debnath and A. F. Hawary, "Adapting Travelling Salesmen Problem for Real-Time UAS Path Planning Using Genetic Algorithm," *Intelligent Manufacturing and Mechatronics*, pp. 151–163, 2021, https://doi.org/10.1007/978-981-16-0866-7_12.

[23] D. Debnath, F. Vanegas, S. Boiteau, and F. Gonzalez, "An Integrated Geometric Obstacle Avoidance and Genetic Algorithm TSP Model for UAV Path Planning," *Drones*, vol. 8, no. 7, p. 302, 2024, https://doi.org/10.3390/drones8070302.

[24] K. Sriniketh *et al*., "Robot-aided human evacuation optimal path planning for fire drill in buildings," *Journal of Building Engineering*, vol. 72, p. 106512, 2023, https://doi.org/10.1016/j.jobe.2023.106512.

[25] Y. Li, J. Zhao, Z. Chen, G. Xiong, and S. Liu, "A Robot Path Planning Method Based on Improved Genetic Algorithm and Improved Dynamic Window Approach," *Sustainability*, vol. 15, no. 5, p. 4656, 2023, https://doi.org/10.3390/su15054656.

[26] J. Shao, "Robot Path Planning Method Based on Genetic Algorithm," *Journal of Physics: Conference Series*, vol. 1881, no. 2, p. 022046, 2021, https://doi.org/10.1088/1742-6596/1881/2/022046.

[27] Z. Ke-qi, "Path Planning of Mobile Robots Based on Improved Genetic Algoritm," *International Journal of Engineering Continuity*, vol. 2, no. 1, pp. 40-48, 2022, https://doi.org/10.58291/ijec.v2i1.84.

[28] J. Ma, Y. Liu, S. Zang, and L. Wang, "Robot Path Planning Based on Genetic Algorithm Fused with Continuous Bezier Optimization," *Computational Intelligence and Neuroscience*, vol. 2020, no. 1, pp. 1–10, 2020, https://doi.org/10.1155/2020/9813040.

[29] A. Hussain and Y. S. Muhammad, "Trade-off between exploration and exploitation with genetic algorithm using a novel selection operator," *Complex & Intelligent Systems*, vol. 6, no. 1, pp. 1–14, 2020, https://doi.org/10.1007/s40747-019-0102-7.

[30] P. K. Nandi and A. K. Dutta, "Global Path Planning for Mobile Robots with optimization through Advanced Neuro-Genetic Algorithms: A Cutting-Edge Exploration," *SAMRIDDHI: A Journal of Physical Sciences, Engineering and Technology*, vol. 15, no. 04, pp. 398–405, 2023, https://doi.org/10.18090/samriddhi.v15i04.03.

[31] K. S. Suresh, R. Venkatesan, and S. Venugopal, "Multi-objective optimization for solving mobile robot path planning problem employing hybridization of algorithms," *International Journal Of Health Sciences*, vol. 6, no. S5, p. 1973–1992, 2022, https://doi.org/10.53730/ijhs.v6nS5.9058.

[32] S. A. Fatemi Aghda and M. Mirfakhraei, "Improved routing in dynamic environments with moving obstacles using a hybrid Fuzzy-Genetic algorithm," *Future Generation Computer Systems*, vol. 112, pp. 250–257, 2020, https://doi.org/10.1016/j.future.2020.05.024.

[33] P. T. Zacharia and E. K. Xidias, "AGV routing and motion planning in a flexible manufacturing system using a fuzzy-based genetic algorithm," *The International Journal of Advanced Manufacturing Technology*, vol. 109, no. 7, pp. 1801–1813, 2020, https://doi.org/10.1007/s00170-020-05755-3.

[34] F. Chen, "Optimal Design of Computer Network Security Performance Based on Genetic Algorithm," *Journal of Physics: Conference Series*, vol. 1852, no. 3, p. 032054, 2021, https://doi.org/10.1088/1742-6596/1852/3/032054.

[35] A. Alhroob, W. Alzyadat, A. Tareq Imam, and G. M. Jaradat, "The Genetic Algorithm and Binary Search Technique in the Program Path Coverage for Improving Software Testing Using Big Data," *Intelligent Automation & Soft Computing*, vol. 26, no. 4, pp. 725–733, 2020, https://doi.org/10.32604/iasc.2020.010106.

[36] U. K. Acharya and S. Kumar, "Genetic algorithm based adaptive histogram equalization (GAAHE) technique for medical image enhancement," *Optik*, vol. 230, p. 166273, 2021, https://doi.org/10.1016/j.ijleo.2021.166273.

[37] H. Ansar, A. Jalal, M. Gochoo, and K. Kim, "Hand Gesture Recognition Based on Auto-Landmark Localization and Reweighted Genetic Algorithm for Healthcare Muscle Activities," *Sustainability*, vol. 13, no. 5, p. 2961, 2021, https://doi.org/10.3390/su13052961.

[38] S. Sendari, A. B. Putra Utama, N. S. Fanany Putri, P. Widiharso, and R. J. Putra, "K-Means and Fuzzy C-Means Optimization using Genetic Algorithm for Clustering Questions," *International Journal of*

Advanced Science and Computer Applications, vol. 1, no. 1, pp. 1–9, 2021, https://doi.org/10.47679/ijasca.v1i1.2.

[39] K. M. Hamdia, X. Zhuang, and T. Rabczuk, "An efficient optimization approach for designing machine learning models based on genetic algorithm," *Neural Computing and Applications*, vol. 33, no. 6, pp. 1923–1933, 2021, https://doi.org/10.1007/s00521-020-05035-x.

[40] D. Lestari, S. Sendari, and I. A. E. Zaeni, "Genetic algorithm for finding shortest path of mobile robot in various static environments," *Jurnal INFOTEL*, vol. 15, no. 3, pp. 281–287, 2023, https://doi.org/10.20895/infotel.v15i3.961.

[41] M. N. Ab Wahab *et al*., "Improved genetic algorithm for mobile robot path planning in static environments," *Expert Systems with Applications*, vol. 249, p. 123762, 2024, https://doi.org/10.1016/j.eswa.2024.123762.

[42] K. Li, Q. Hu, and J. Liu, "Path Planning of Mobile Robot Based on Improved Multiobjective Genetic Algorithm," *Wireless Communications and Mobile Computing*, vol. 2021, no. 1, p. 8836615, 2021, https://doi.org/10.1155/2021/8836615.

[43] Z. Chen, J. Xiao, and G. Wang, "An Effective Path Planning of Intelligent Mobile Robot Using Improved Genetic Algorithm," *Wireless Communications and Mobile Computing*, vol. 2022, no. 1, p. 9590367, 2022, https://doi.org/10.1155/2022/9590367.

[44] T. Feng *et al*., "The Optimal Global Path Planning of Mobile Robot Based on Improved Hybrid Adaptive Genetic Algorithm in Different Tasks and Complex Road Environments," *IEEE Access*, vol. 12, pp. 18400-18415, 2024, https://doi.org/10.1109/ACCESS.2024.3357990.

[45] K. Hao, J. Zhao, B. Wang, Y. Liu, and C. Wang, "The Application of an Adaptive Genetic Algorithm Based on Collision Detection in Path Planning of Mobile Robots," *Computational Intelligence and Neuroscience*, vol. 2021, no. 1, p. 5536574, 2021, https://doi.org/10.1155/2021/5536574.

[46] W. Rahmaniar and A. Rakhmania, "Mobile Robot Path Planning in a Trajectory with Multiple Obstacles Using Genetic Algorithms," *Journal of Robotics and Control (JRC)*, vol. 3, no. 1, pp. 1–7, 2021, https://doi.org/10.18196/jrc.v3i1.11024.

[47] A. Loganathan and N. S. Ahmad, "A systematic review on recent advances in autonomous mobile robot navigation," *Engineering Science and Technology, an International Journal*, vol. 40, p. 101343, 2023, https://doi.org/10.1016/j.jestch.2023.101343.

[48] M. S. Abed, O. F. Lutfy, and Q. F. Al-Doori, "A Review on Path Planning Algorithms for Mobile Robots," *Engineering and Technology Journal*, vol. 39, no. 5A, pp. 804–820, 2021, https://doi.org/10.30684/etj.v39i5A.1941.

[49] M. N. A. Wahab, S. Nefti-Meziani, and A. Atyabi, "A comparative review on mobile robot path planning: Classical or meta-heuristic methods?," *Annual Reviews in Control*, vol. 50, pp. 233-252, 2020, https://doi.org/10.1016/j.arcontrol.2020.10.001.

[50] H. K. Tripathy, S. Mishra, H. K. Thakkar, and D. Rai, "CARE: A Collision-Aware Mobile Robot Navigation in Grid Environment using Improved Breadth First Search," *Computers & Electrical Engineering*, vol. 94, p. 107327, 2021, https://doi.org/10.1016/j.compeleceng.2021.107327.

[51] F. H. Ajeil, I. K. Ibraheem, A. T. Azar, and A. J. Humaidi, "Grid-Based Mobile Robot Path Planning Using Aging-Based Ant Colony Optimization Algorithm in Static and Dynamic Environments," *Sensors*, vol. 20, no. 7, p. 1880, 2020, https://doi.org/10.3390/s20071880.

[52] 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 Computing and Applications*, vol. 32, no. 17, pp. 13155–13171, 2020, https://doi.org/10.1007/s00521-020-04764-3.

[53] Q. Cai, "A Comparison Between A* and RRT Algorithm in Path Planning for Mobile Robot," *Highlights in Science, Engineering and Technology*, vol. 97, pp. 282–287, 2024, https://doi.org/10.54097/2stv5y97.

[54] J. Ding, Y. Zhou, X. Huang, K. Song, S. Lu, and L. Wang, "An improved RRT* algorithm for robot path planning based on path expansion heuristic sampling," *Journal of Computational Science*, vol. 67, p. 101937, 2023, https://doi.org/10.1016/j.jocs.2022.101937.

[55] M. Kara, "Evaluation of popular path planning algorithms," *International Journal of Electronics and Telecommunications*, vol. 70, no. 1, pp. 13–22, 2024, https://doi.org/10.24425/ijet.2023.147699.

[56] G. Joy, C. Huyck, and X.-S. Yang, "Review of Parameter Tuning Methods for Nature-Inspired Algorithms," *Benchmarks and Hybrid Algorithms in Optimization and Applications*, pp. 33–47, 2023, https://doi.org/10.1007/978-981-99-3970-1_3.

[57] P. Sebastjan and W. Kuś, "Method for Parameter Tuning of Hybrid Optimization Algorithms for Problems with High Computational Costs of Objective Function Evaluations," *Applied Sciences*, vol. 13, no. 10, p. 6307, 2023, https://doi.org/10.3390/app13106307.

[58] G. Airlangga, "A comparative analysis of pathfinding algorithms in static environments: modified A*, PSO, and FLA," *Jurnal Mantik*, vol. 7, no. 4, pp. 3697-3976, 2024, https://www.ejournal.iocscience.org/index.php/mantik/article/view/4795/3428.

[59] L. Wang and L. Sun, "Path Planning Algorithm Based on Obstacle Clustering Analysis and Graph Search," *Symmetry*, vol. 15, no. 8, p. 1498, 2023, https://doi.org/10.3390/sym15081498.

[60] E. Shem-Tov, M. Sipper, and A. Elyasaf, "Deep Learning-Based Operators for Evolutionary Algorithms," *Neural and Evolutionary Computing*, 2024, https://doi.org/10.48550/arXiv.2407.10477.

[61] M. A. Albadr, S. Tiun, M. Ayob, and F. AL-Dhief, "Genetic Algorithm Based on Natural Selection Theory for Optimization Problems," *Symmetry*, vol. 12, no. 11, p. 1758, 2020, https://doi.org/10.3390/sym12111758.

[62] S. T. Shishavan and F. S. Gharehchopogh, "An improved cuckoo search optimization algorithm with genetic algorithm for community detection in complex networks," *Multimedia Tools and Applications*, vol. 81, no. 18, pp. 25205–25231, 2022, https://doi.org/10.1007/s11042-022-12409-x.

[63] J. Lengler and J. Meier, "Large population sizes and crossover help in dynamic environments," *Natural Computing*, vol. 23, no. 1, pp. 115–129, 2024, https://doi.org/10.1007/s11047-022-09915-0.