Article

Improved Genetic Algorithm for Solving Robot Path Planning Based on Grid Maps

Jie Zhu [1] and Dazhi Pan [1,2,*]

[1] College of Mathematic and Information, China West Normal University, Nanchong 637009, China; zjiemath@stu.cwnu.edu.cn
[2] Sichuan Colleges and Universities Key Laboratory of Optimization Theory and Applications, Nanchong 637009, China
* Correspondence: pdzzj@cwnu.edu.cn

Abstract: Aiming at some shortcomings of the genetic algorithm to solve the path planning in a global static environment, such as a low efficiency of population initialization, slow convergence speed, and easy-to-fall-into the local optimum, an improved genetic algorithm is proposed to solve the path planning problem. Firstly, the environment model is established by using the grid method; secondly, in order to overcome the difficulty of a low efficiency of population initialization, a population initialization method with directional guidance is proposed; finally, in order to balance the global and local optimization searching and to speed up the solution speed, the proposed non-common point crossover operator, range mutation operator, and simplification operator are used in combination with the one-point crossover operator and one-point mutation operator in the traditional genetic algorithm to obtain an improved genetic algorithm. In the simulation experiment, Experiment 1 verifies the effectiveness of the population initialization method proposed in this paper. The success rates in Map 1, Map 2, Map 3, and Map 4 were 56.3854%, 55.851%, 34.1%, and 24.1514%, respectively, which were higher than the two initialization methods compared. Experiment 2 verifies the effectiveness of the genetic algorithm (IGA) improved in this paper for path planning. In four maps, the path planning is compared with the five algorithms and the shortest distance is achieved in all of them. The two experiments show that the improved genetic algorithm in this paper has advantages in path planning.

Keywords: path planning; genetic algorithm; grid method; directional guidance; non-common point crossing operator

MSC: 68W50; 90C59

Citation: Zhu, J.; Pan, D. Improved Genetic Algorithm for Solving Robot Path Planning Based on Grid Maps. *Mathematics* **2024**, *12*, 4017. https://doi.org/10.3390/math12244017

Academic Editor: José Antonio Sanz

Received: 19 November 2024
Revised: 16 December 2024
Accepted: 18 December 2024
Published: 21 December 2024

1. Introduction

As the level of machine intelligence increases, the application scenarios of robots [1] become richer and richer, such as cleaning robots [2] for ground waste disposal, space robots [3] for sample collection and environmental surveys on Mars and the Moon, and so on. In the navigation of mobile robots, path planning is very important, which refers to the robot planning a safe and collision-free path from the start point to the target point according to the actual needs in a static environment. An efficient path requires not only obstacle avoidance, but also shorter distances, number of turns, etc. in order to reduce energy costs.

1.1. Literature Review

In recent years, researchers have proposed to apply many algorithms to robot path planning. Meta-heuristic algorithms [4] are widely used, examples of which are particle swarm algorithm [5], ant colony algorithm [6], genetic algorithm [7], A* [8], and so on.

Zeng et al. [9] proposed switching local evolutionary PSO (SLEPSO) based on the non-homogeneous Markov chain and analyzed differential evolution for the path planning problem. The velocity updating equation of the presented SLEPSO algorithm jumps from one mode to another based on the non-homogeneous Markov chain, which can overcome the contradiction between local and global search. Mac et al. [10] presents a new hierarchical global path planning method for robots moving in complex environments. The method has a three-level structure to obtain feasible, safe and optimal paths that are solved using the PSO algorithm. Wang et al. [11] proposed the clustering guidance multi-objective particle swarm algorithm, which employs an improved particle clustering strategy, a guided particle selection strategy, and an external archiving mechanism, and uses it in path planning. Abaas et al. [12] proposed a PSO algorithm with inertia weight variables for optimal path planning. The algorithm has cleverly designed the inertia weights and applied them to complex path planning, which can realize the path planning quickly. Tao et al. [13] proposed a bi-population PSO algorithm with a random perturbation strategy (BPPSO), which divides particles into two subpopulations. The first subpopulation enhances global search capabilities by considering the quality of particles and the optimal solution of a randomly selected particle when updating velocities. The second subpopulation strengthens local search using a linear cognitive coefficient adjustment strategy. Finally, the algorithm is applied to robot path planning.

In recent years, ant colony algorithms are widely used in path planning. Gao et al. [14] proposed an improved ant colony algorithm (EH-ACO), which, firstly, improves the heuristic distance in the local visibility formula by considering the heuristic distance from the ant's neighborhood point to the target. Second, a new pheromone diffusion gradient formula is designed, which emphasizes that pheromones leaving a path diffuse into a region where the pheromone density is distributed in a gradient. Third, a backtracking strategy is introduced to enable ants to find new paths when their search is blocked. Zheng et al. [15] proposed an adaptive ant colony algorithm, which gives an adaptive heuristic function to update the heuristic information using the shortest practical distance passed by the ants. The reward and punishment rules are introduced to optimize the local pheromone updating strategy. A pseudo-random state transfer rule is used to optimize the state transfer function. Cui et al. [16] proposed a multi-strategy adaptable ant colony optimization algorithm (MsAACO). The algorithm has four strategies. First, a direction guidance mechanism is proposed to improve the performance of node selection; second, an adaptive heuristic function is introduced to reduce the length and the number of circles of the optimal path solution; meanwhile, a deterministic state transfer probability criterion is used to improve the convergence speed of the ACO algorithm; and lastly, a non-uniform pheromone initialization is utilized to enhance the ACO algorithm's ability of selecting the favorable region. Yang et al. [17] proposed an improved ant colony algorithm based on expanding neighborhood, this algorithm extends the search neighborhood from 8 to 10, then improves the initial pheromone by positional information, and finally improves the pheromone updating rule by combining the advantages of Maximum-Minimum Ant System (MMAS) and Elite Ant Model.

In addition to this, there are a number of other algorithms. Silva et al. [18] proposed a simplified cost function heuristic that improves on the traditional A* algorithm. The heuristic reduces the computational effort of the search, the number of expansion units and mainly the time required for target localization. Batik Garip et al. [19] uses the A* algorithm for multi-robot paths. The path planning for the shortest distance has been performed using A* algorithm in dynamic frame between robot–object and object–target point, respectively. Hassani et al. [20] proposed a free segment and steering point algorithm. The aim of the turning point approach is to search a safe path for the mobile robot to make the robot moving from a starting position to a destination position without hitting obstacles. This proposed algorithm handles two different objectives which are the path safety and the path length. Yunqiang et al. [21] proposed an improved artificial potential field algorithm for solving multi-objective path planning problems. The method takes the path length and path security as the optimization targets, and the obstacle repulsion coefficient as the decision variable. The objective function is calculated by establishing the artificial potential field performance model, and the NSGA-II algorithm is used for multi-objective optimization to obtain the optimal path. Xu et al. [22] proposed a new globally optimal artificial bee colony algorithm, squarely addressing the slow convergence speed of the artificial bee colony algorithm and the one-dimensional search strategy, limiting its advantages in separable functions. A co-evolutionary framework is introduced into the artificial bee colony algorithm to improve the convergence speed and dimension dependence of the algorithm. Muthukumaran et al. [23] proposed a new meta-heuristic optimization algorithm, Dragonfly Algorithm (DA). This new meta-heuristic dragonfly algorithm is inspired by the static and dynamic swarming behaviors of dragonflies in nature and applies it to the navigation of autonomously moving robots in unknown and complex environments filled with static obstacles. Ajeil et al. [24] proposed a hybrid particle swarm algorithm (PSO-MFB) that combines the advantages of particle swarm and bat algorithms to overcome the shortcomings of traditional methods such as grid methods. Ab Wahab et al. [25] proposed a hybrid meta-heuristic algorithm named PSOFS between Particle Swarm Optimization and Edge Search Algorithms and applied the algorithm in robot path planning in unknown environments. Zhang et al. [26] proposed an algorithm that mixes the genetic algorithm and firefly algorithm. The core idea of this new algorithm is that when the FA falls into the local optimal solution, the local optimal fireflies would be regarded as a group, and the group is subjected to the selection, crossover and mutation operations in the GA. Finally, the optimal firefly individual can be obtained from genetic operations. Duan et al. [27] proposed an adaptive path-smoothening optimization method is proposed in this study, which combines neural network, genetic algorithm, and Bézier curve to effectively resolve the problems of strong subjectivity, cumbersome steps, and thus low efficiency in the selection process of control points.

1.2. Research Motivation and Contribution

Traditional genetic algorithms face these shortcomings in solving robot path planning problems: inefficient population initialization, easy-to-fall-into local optimums, and slow convergence. To address the shortcomings of genetic algorithms in robot path planning, the main contribution of this paper is as follows:

1. A population initialization method with directional guidance is designed, which is able to find the target point faster, overcome the difficulties of population initialization, and improve the quality of the initial population.
2. A new crossover operator is developed that does not rely on a common point, which solves the difficulty of one-point crossover operators that require a common point and increases the search capability.
3. Pairing the designed operators with traditional genetic operators balances the local and global search capabilities of the genetic algorithm.

1.3. Paper Organization

The rest of the paper is organized in the following manner. Section 2 introduces the concept and use of genetic algorithms, as well as the application of genetic algorithms in path planning and their improvement by the rest of the scholars. Section 3 introduces the construction of the robot's mobile environment. Section 4 introduces the genetic algorithm constructed in this paper, including chromosome construction, each genetic operator, and the flow of the genetic algorithm constructed. Section 5 introduces the results of the simulation experiments, which are mainly compared with the traditional genetic algorithm(GA), the algorithm from the literature [28] (CGA), the algorithm from the literature [29] (BGA), the algorithm from the literature [30] (CSAGA) and Bitterling fish optimization algorithm [31] (BFO), and the experimental results show that the improved genetic algorithm(IGA) in this paper is able to achieve robot path planning with high quality.

2. Genetic Algorithm

The genetic algorithm [32] is an optimization algorithm based on the principles of natural selection and genetics, created by Professor Holland [33] and their students, and commonly used to solve complex optimization problems. It gradually evolves individuals with higher fitness from an initial population by simulating the process of biological evolution, using operations such as selection, crossover and mutation. The basic steps of the algorithm include generating an initial population, evaluating individual fitness, selecting superior individuals for crossover and mutation, and then forming a new generation of population. This process is repeated until the stopping condition is satisfied.

With regard to the developments that have been made so far, the genetic algorithm has been improved to the point where it has a wide range of applications in mobile robot path planning in order to solve its shortcomings in path planning. Elhoseny et al. [34] proposed an improved genetic algorithm (GADPP) that solves path planning problems in dynamic environments. The proposed method uses Bezier Curve to refine the final path according to the control points identified by GADPP. Patle et al. [35] proposed a new variant of genetic algorithm using the binary codes through matrix for mobile robot navigation. The path planning strategy is established using the trace theory as the optimum controller, Sylvester Law of Inertia (SLI) and matrix simulation. The Matrix-Binary Code-based Genetic Algorithm (MGA) transforms the environment from chaos to array. Kwaśniewski et al. [36] proposed an obstacle avoidance algorithm which uses the genetic path finding algorithm. The actual version is based on the 2D map which is built by the robot and the second-degree B-spline is used for the path model. The algorithm achieves high performance using only one processor thread in most cases and can also be easily multi-threaded. Suresh et al. [37] proposed a multi-objective genetic algorithm (MRPS-MOGA) for robot path planning. MRPS-MOGA is designed with the novelty of a genetic algorithm namely tournament selection, ring crossover, and adaptive bit string mutation to find the optimal path. Tournament selection is applied to select the best individual path from the population to satisfy the fitness criterion. Bao et al. [29] explored the effect of crossover and mutation operations on the solution results in solving robot path planning with genetic algorithms. Ab Wahab et al. [38] proposed an improved genetic algorithm. As an enhanced GA, a Linear Rank-based, or Clearance-based Probabilistic Road Map (CBPRM), technique is proffered to overcome inefficient population initialization and low-quality solutions.

3. Environment Modeling

Environment modeling is very important for the path planning of mobile robots. In this paper, the grid method is used to model the static environment. The established map is a two-dimensional space, the size and position of all obstacles are known, the robot is regarded as a point in the space, and the range of each movement is the neighboring eight grids. As shown in Figure 1, take the 5 × 5 grid map as an example, the white area of the grid map is the free area, and the black is the obstacles; each grid is uniquely numbered from the lower left corner to the upper right corner; the number of each grid can be converted with the coordinates of each other, for example, when the number of the grid is 6, the corresponding coordinates are (2,2). Grid numbers are converted to coordinates using Equations (1) and (2). E_y is the number of columns, $mod()$ is the residual function, $fix()$ is the downward rounding function, and t is for the grid number.

$$\begin{cases} x = mod(t, E_y) + 1 \\ y = fix(t/E_y) + 1 \end{cases} \tag{1}$$

$$t = (y - 1) * E_y + x - 1 \tag{2}$$

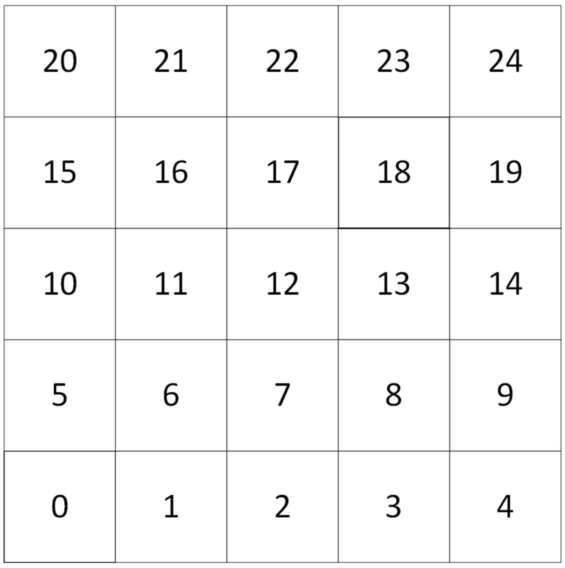

Figure 1. Example of grid map.

4. Improved Genetic Algorithm

This section describes the various aspects of the improved genetic algorithm constructed in this paper.

4.1. Chromosome Construction & Population Initialization

In genetic algorithm path planning, a chromosome $P(p_1, p_2, \cdots, p_n)$ represents a complete route from the start point (S) to the target point (T). In order to present the complete path information, a chromosome of variable length is encoded, where the genes of the chromosome consist of grid serial numbers, the first gene of the chromosome is the start point, and the last gene is the target point. In order to ensure the efficiency of the path, a qualified chromosome does not allow duplicate grid numbers as well as numbers that correspond to obstacles. Figure 2 is used as an chromosome example $(0, 1, 7, 12, 18, 24)$ of a feasible path.

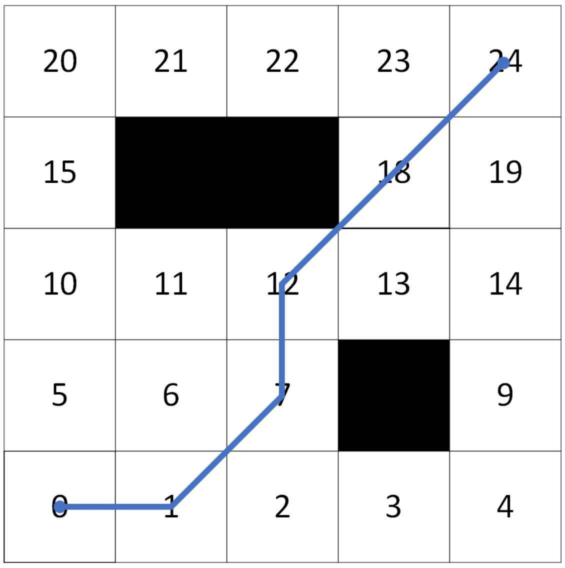

Figure 2. Examples of a feasible path.

Initial population has an important impact on the efficiency of the genetic algorithm. Traditional genetic algorithms generate a complete path by random walks. The start grid will be added to the path, a random search for walkable grid to join the path will be carried out, until the target grid is found. Then, the path creation is complete, but the chances of successfully generating a complete path are low. Alajian et al. [28] proposed a greedy method to initialize the population based on Euclidean distance, which calculates each grid score using Equation (3) and then selects the grid based on the score using Roulette Wheel Selection (RWS). In this paper, we improve on this method and propose a population initialization method with directional guidance for improving the efficiency of population initialization. The method with directional guidance focuses on using vector guidance to increase the score of the advantage grids, thus making it more favorable to find the target grid. $sgn()$ is the symbolic function.The pseudo-code for the method in this paper is shown in Algorithm 1.

$$score = \frac{1}{d_{(current,target)}} \tag{3}$$

$$d_{(current,target)} = \sqrt{(x_{current} - x_{target})^2 + (y_{current} - y_{target})^2} \tag{4}$$

$$score_{new} = \begin{cases} 3 * score\ ,\ sgn((x_{target} - x_{current}) * (x_{next} - x_{current})) \\ \qquad\qquad + sgn((y_{target} - y_{current}) * (y_{next} - y_{current})) = 2 \\ 2 * score\ ,\ sgn((x_{target} - x_{current}) * (x_{next} - x_{current})) \\ \qquad\qquad + sgn((y_{target} - y_{current}) * (y_{next} - y_{current})) = 1 \\ score \quad ,\ others \end{cases} \tag{5}$$

As shown in Figure 3, the starting grid is 0 and the target grid is 24, with three paths $P_1(0,1,7,12)$, $P_2(0,1,7,13)$ and $P_3(0,1,6,11,7)$ being created. The next set of walkable grids for P_1 is $(6,11,13,18)$. The other set of walkable grids for P_2 is $(9,12,14,18,19)$. The final set of walkable grids for P_3 is $(2,3,12,13)$. Each walkable grid is score-adjusted using Equation (5), and the probability of each grid being selected is shown in Table 1. As shown in Table 1, we make the selection probability of advantageous grids larger and the selection probability of disadvantageous grids smaller. Thus, the purpose of creating chromosomes quickly is achieved.

Algorithm 1: Population initialization method with directional guidance.

```
1  Define the start grid ($S$) and target grid ($T$)
2  Initialize $chromosome = []$, $gridlist = []$, $current = S$
3  Calculate each grid score according to Equation (3)
4  Append $current$ into $chromosome$
5  for $i \leftarrow 1$ to $10,000$ do
6      Find the grid around $current$ to add to $gridlist$
7      if $find\ T\ in\ gridlist$ then
8          Append $T$ into $chromosome$
9          Initialization successful
10         Break
11     end
12     if $gridlist\ is\ empty$ then
13         Initialization Failed
14         $chromosome = []$
15         Break
16     end
17     Adjust the grid score in $gridlist$ according to Equation (5)
18     Use RWS to select $nextgrid$ in the $gridlist$ based on the adjusted score
19     Append $nextgrid$ into $chromosome$
20     $current = nextgrid$
21     $gridlist = []$
22 end
```

Table 1. Example of grid selection probability.

	Grid	Score	Adjusted Score	Traditional GA's Methods	Literature [28]'s Method	Paper's Method
Figure 3a	6	0.2357	0.2357	0.25	0.1414	0.0668
	11	0.2774	0.2774	0.25	0.1664	0.0786
	13	0.4472	0.8944	0.25	0.2682	0.2535
	18	0.7071	2.1213	0.25	0.4243	0.6011
Figure 3b	9	0.3333	0.3333	0.2	0.1152	0.0546
	12	0.3536	0.3536	0.2	0.1222	0.058
	14	0.5	1	0.2	0.1728	0.1639
	18	0.7071	1.4142	0.2	0.2443	0.2318
	19	1	3	0.2	0.3455	0.4917
Figure 3c	2	0.2236	0.2236	0.25	0.1765	0.0889
	3	0.2425	0.2425	0.25	0.1914	0.0964
	12	0.3536	0.7072	0.25	0.2791	0.2812
	13	0.4472	1.3416	0.25	0.353	0.5335

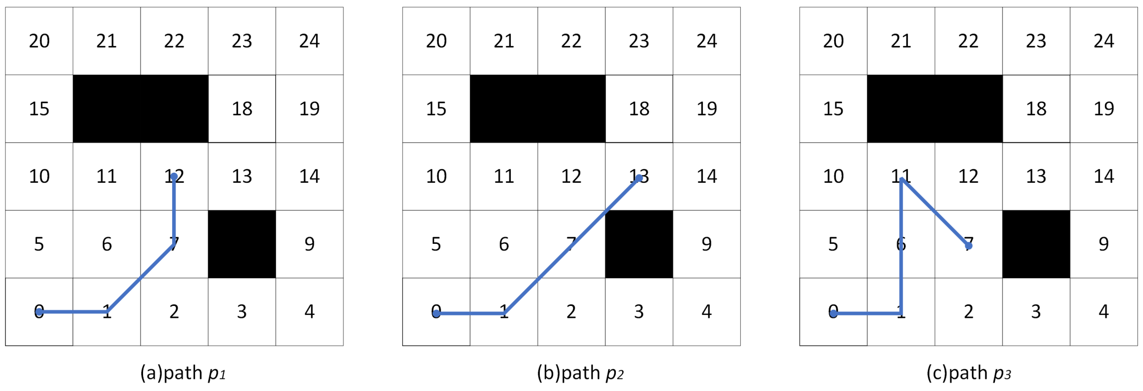

Figure 3. Example of gird selection.

4.2. Construction of the Fitness Evaluation Equation

Considering the energy consumption of the mobile robot operation, the robot is required to move in the path of the distance as short as possible and the route as smooth as possible. The distance and smoothness are the requirements of the objective optimization. Therefore, the multi-constraint fitness function, shown in Equation (6), is designed to measure the quality of the path. Equation (7) outlines the path distance objective, and Equation (8) outlines the smoothness objective. where ω is the weight of distance and $c1, c2, c3$ are used to measure the energy consumption paid for the three turning angles in the robot movement $(0 < c1 < c2 < c3)$.

$$Fit(P) = \frac{1}{\omega * fit_1(P) + (1-\omega) * fit_2(P)} \tag{6}$$

$$fit_1(P) = \sum_{i=2}^{n} |\overrightarrow{P_i P_{i-1}}| = \sum_{i=2}^{n} \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2} \tag{7}$$

$$fit_2(P) = \sum_{i=1}^{n-2} \Gamma(p_i) \tag{8}$$

$$\Gamma(p_i) = \begin{cases} 0 \ , & \arccos\langle \overrightarrow{P_i P_{i+1}}, \overrightarrow{P_{i+1} P_{i+2}} \rangle = \pi \\ c1 \ , & \arccos\langle \overrightarrow{P_i P_{i+1}}, \overrightarrow{P_{i+1} P_{i+2}} \rangle = \frac{3}{4}\pi \\ c2 \ , & \arccos\langle \overrightarrow{P_i P_{i+1}}, \overrightarrow{P_{i+1} P_{i+2}} \rangle = \frac{1}{2}\pi \\ c3 \ , & \arccos\langle \overrightarrow{P_i P_{i+1}}, \overrightarrow{P_{i+1} P_{i+2}} \rangle = \frac{1}{4}\pi \end{cases} \tag{9}$$

4.3. Selection Operator

In biological genetic evolution, better adapted individuals will be more likely to participate in the next round of evolution, and good populations can speed up the convergence of genetic algorithms. In this paper, we will adopt the roulette wheel approach for population selection. After completing one iteration, according to the fitness value, the population for the next iteration will be selected by the Roulette Wheel Selection (RWS). In order to keep the best individual, the best individual in the previous generation will replace the worst individual in the constructed population after each iteration is completed. The Roulette Wheel Selection (RWS) is implemented according to Equation (10), where $pb(i)$ is the probability of being selected, $Fit(P_i)$ is the fitness value, and $popnum$ is the number of individuals in the population.

$$pb(i) = \frac{Fit(P_i)}{\sum_{j=1}^{popnum} Fit(P_j)} \tag{10}$$

4.4. Crossover Operator

In genetic algorithms, the crossover operator is a key genetic operation that is mainly used to generate new individuals to maintain the diversity of the population and to explore the solution space. In this paper, we use two kinds of crossover operators, which are the one-point crossover operator and the non-common point crossover operator.

The first crossover operator is the one-point crossover operator, which selects two-parent chromosomes, removes the start and end points to form a common set, and then randomly selects a point from the common set to crossover to obtain the child chromosome, or does not crossover if the point set is empty. Figure 4 shows an example of a one-point crossover operator.

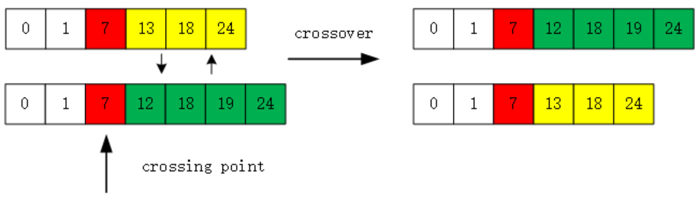

Figure 4. Example of one-point crossover operator.

The second crossover operator is the non-common point crossover operator. This crossover operator does not rely on a common point for crossover, which solves the difficulty of not being able to crossover when there is no common point and facilitates the solution search. Path continuum processing aims to determine whether the adjacent nodes in the chromosome P are continuous according to Equation (11). If $D = 1$, it is continuous, and there is no need to insert a new node; if $D > 1$, calculate the coordinates of the node that needs to be inserted according to Equation (12). $max()$ is the maximum value function, and $abs()$ is the absolute value function. The pseudo-code for the non-common point crossover operator in this paper is shown in Algorithm 2.

$$D = \max\{abs(x_{i+1} - x_i), abs(y_{i+1} - y_i)\} \tag{11}$$

$$\begin{cases} x_{new} = int(\dfrac{x_{i+1} + x_i}{2}) \\ y_{new} = int(\dfrac{y_{i+1} + y_i}{2}) \end{cases} \quad (12)$$

As shown in Figure 5, the two-parent chromosomes are $P_1(0,5,10,16,17,23,24)$ and $P_2(0,1,2,8,14,19,24)$. In the parent chromosomes, the max grid number of the second column is 16, and the min grid number is 1, so grid 6 is randomly selected from set $(1,6,11,16)$ to be added to the child chromosome P. In accordance with this rule, we obtain the child chromosome $P(0,6,17,18,24)$. Finally, we perform the path continuum process on the child chromosome $P(0,6,11,17,18,24)$.

Algorithm 2: The non-common point crossover operator

```
1  Selection of two-patent chromosomes $P1$ and $P2$
2  Initialize $P = []$
3  Append start grid($S$) into $P$
4  for $i \leftarrow 2$ to $E_y$ do
5      Select the $max$ grid number and the $min$ grid number in column $i(p1\&p2)$
6      Randomly select a grid ($R$) that lies in column $i$ with $min$ and $max$
7      Append grid ($R$) into $P$
8  end
9  Append target grid ($T$) into $P$
10 if Successful of $P$ using path continuum processing then
11     2 from $P1$,$P2$, and $P$ are selected to become child chromosomes using RWS
12 end
13 else
14     crossover failure
15 end
```

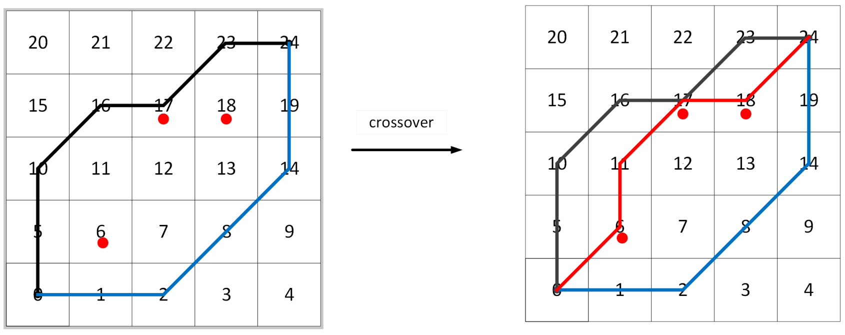

Figure 5. Example of the non-common point crossover operator.

4.5. Mutation Operator

In genetic algorithms, the mutation operator is another important genetic operation, whose main roles include increasing randomness, exploring solution space, and avoiding local optimality. In this paper, two kinds of mutation operators are used, which are the one-point mutation operator and the range mutation operator.

The first mutation operator is the one-point mutation operator, which enhances the exploration of localization by randomly changing one gene value of an individual. (i) one-parent chromosome $P(p_1, p_2, \cdots, p_n)$ is selected, and one node p_m in the parent chromosome other than the start point and the target point is randomly chosen for mutation; (ii) then, this node is deleted to find the set Y of optional points that can make p_{m-1} and p_{m+1} consecutive; (iii) a node is randomly selected from Y in place of p_m. Figure 6 shows an example of one-point mutation operator.

The second mutation operator is the range mutation operator, which balances the local and global search capabilities by randomly changing multiple gene values of an individual, increasing the diversity of the population and preventing premature convergence. (i) randomly select two nodes p_i and p_j in the parent chromosome $P(p_1, p_2, \cdots, p_n)$ for the range mutation, and delete the path in the middle of these two nodes; (ii) set p_i as the start point and p_j as the target point. Determine a new path by using the method of population initialization, and if a new path is not found, the mutation fails. Figure 7 shows an example of the range mutation operator.

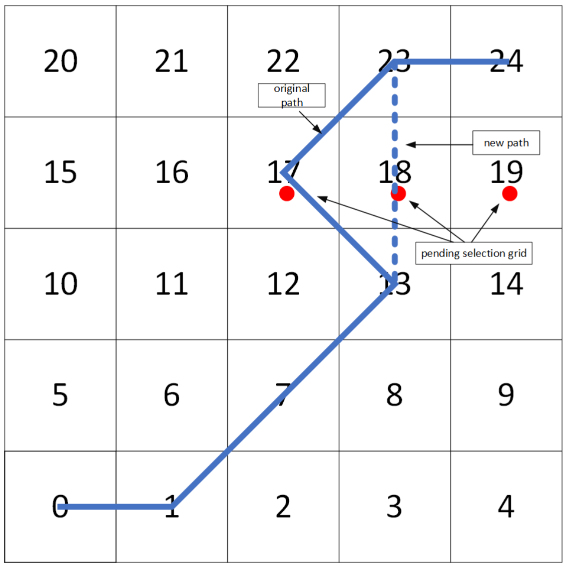

Figure 6. Example of one-point mutation operator.

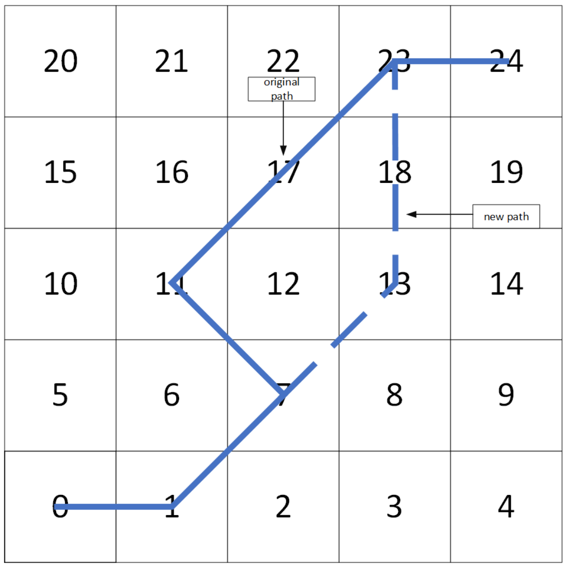

Figure 7. Example of range mutation operator.

4.6. Simplification Operator

In the improved genetic algorithm of this paper, a simplification operation will be performed after completing the variance difference operation. The simplification operator is mainly used to remove unwanted or redundant path nodes in the parent chromosome, which is used to reduce the complexity of the solution and improve the performance of the solution. (i) randomly select a node p_t in the parent chromosome $P(p_1, p_2, \cdots, p_n)$ except the end point for simplification; (ii) then, set s to $t+1$, and sequentially search from node p_{t+2} until the target point p_n is finished. The current search point is recorded as p_{now}; (iii) if the distance between p_t and p_{now} is less than 2; s is set to now. The path in the middle of

p_t and p_s is deleted after searching is completed, and the simplification fails if $s = t + 1$. Figure 8 shows an example of the simplification operator.

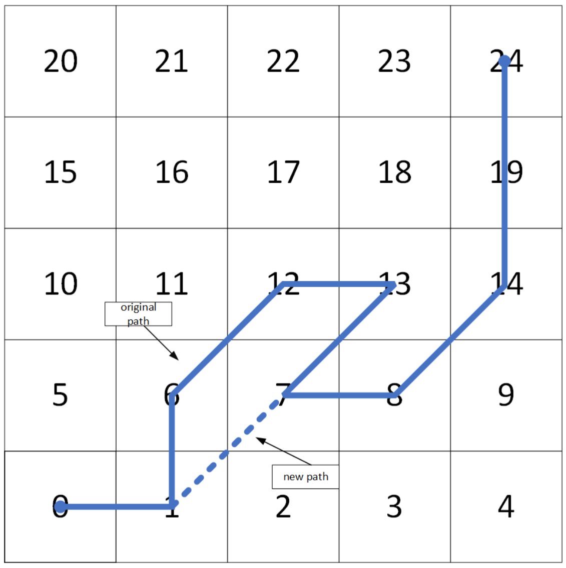

Figure 8. Example of simplification operator.

4.7. Optimization Process Steps

Traditional genetic algorithms (GAs) use a random walk population initialization method, and genetic operators are the selection operator, one-point crossover operator, and one-point mutation operator. The difference between the genetic algorithm used in the literature [28] (CGA) and the traditional genetic algorithm is the population initialization method, where the one from the literature [28] uses Roulette Wheel Selection (RWS) in the selection of grid based on the grid score. The difference between the genetic algorithm used in the literature [29] (BGA) and the traditional genetic algorithm is the one-point mutation operator, where the one from the literature [29] uses the range mutation operator. The algorithm in the literature [30] (CSAGA) combines the simulated annealing algorithm with the genetic algorithm. The following is the construction process of the genetic algorithm in this paper (IGA).

Begin:

Step 1: Initialize the GA and set the parameters' population size *popnum*, number of iterations N, crossover probability p_c, mutation probability p_m, and simplification probability p_s. Step 2: Generate initial populations using the method in this paper.

Step 3: Perform selection, crossover, and mutation. Each iteration randomly selects half of the populations to use one-point crossover and the other half to use non-common point crossover; after crossover is complete, each iteration randomly selects half of the populations to use the one-point mutation and the other half to use the range mutation.

Step 4: Perform simplification operations.

Step 5: If the maximum number of iterations is reached, terminate the execution of the algorithm; otherwise, turn to Step 3.

5. Results Simulation and Performance Evaluation

This test is mainly divided into two experiments, according to different purposes. The first experiment is carried out to test the effect of the initialized population proposed in this paper; the second experiment is carried out to test the effect of the improved genetic algorithm in this paper in terms of the path planning optimization. The maps used for the experimental test are derived from four maps in the literature [29], as shown in Figure 9. The experimental simulation was performed using a MatlabR2023a compilation algorithm, 2.30 GHz, 4 GB, 64-bit operating system computer.

Experiment 1 verifies the performance of the proposed improved population initialization method (IGA), which is compared with the population initialization method of the algorithm used in the literature [28] (CGA) and the population initialization method of the traditional genetic algorithm (GA) in four grid maps, respectively. Repeat the experiment 30 times. The start point is the grid sequence number 0, and the target point is the grid sequence number 399. Table 2 has four evaluation metrics, respectively, which are the number of 100 initialization successes, the time taken to create the 200 populations, the average path length of the created population, and the standard deviation of the average path length of the created population.

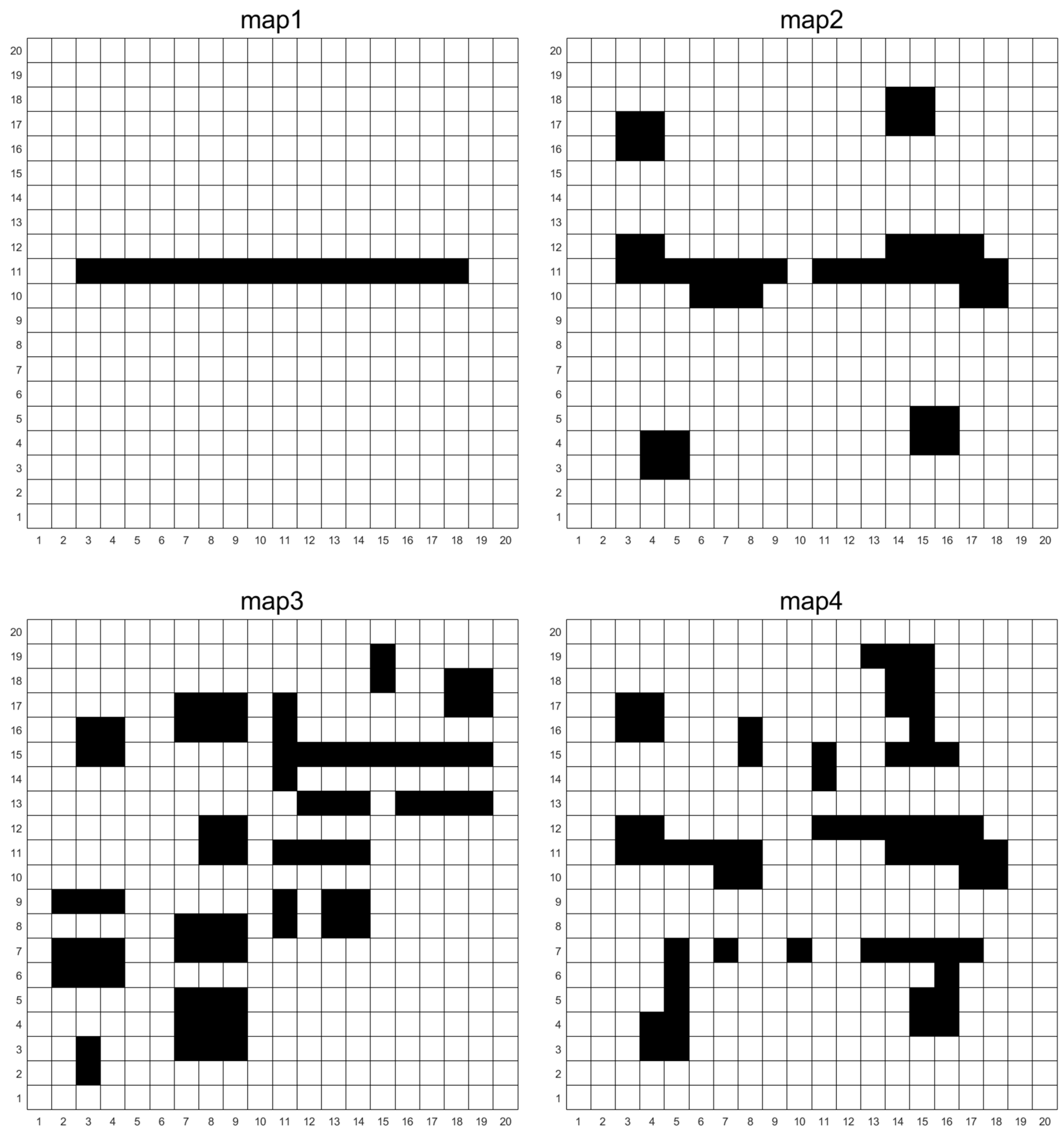

Figure 9. Example of test maps.

From the results of Experiment 1, the success rate of this paper's method in initializing Map 1, Map 2, Map 3, and Map 4 is 56.3854%, 55.851%, 34.1%, and 24.1514%, respectively, which is higher than the other two initialization methods. The time to complete the initialization of population number 200 was 0.3359, 0.3203, 0.4452, and 0.7312, respectively, which is lower than the other two initialization methods. The average path lengths of the created populations were 82.0458, 78.1052, 71.3699, and 81.2861, which were shorter than the other two initialization methods, and the standard deviation was also lower than the other two initialization methods. This paper's method achieved some advantages in all four maps, indicating that the population initialization method with direction guidance proposed in this paper was effective.

Table 2. Experiment 1's simulation results.

Map	Method	Average Number of Successes	Average Processing Time	Population Average Path Length	Sd of Population Average Path Length
Map 1	GA	4.733	2.8011	134.4501	35.4558
	CGA	8.3667	2.6622	119.6382	33.8084
	IGA	56.3854	0.3359	82.0458	26.1825
Map 2	GA	6.4432	2.6111	122.7112	28.9864
	CGA	7.8354	2.0790	109.6144	31.7533
	IGA	55.851	0.3203	78.1052	19.7578
Map 3	GA	3.5667	3.0364	99.4183	25.4193
	CGA	5.4224	2.0871	93.6177	26.8966
	IGA	34.1	0.4452	71.3699	16.3846
Map 4	GA	3.033	4.3500	115.7864	27.2270
	CGA	4.1225	4.1708	102.2958	27.3611
	IGA	24.1514	0.7312	81.2861	18.9799

Experiment 2 verifies the performance of the algorithm proposed in this paper in terms of path planning, which is performed 30 times on four grid maps with the traditional genetic algorithm (GA), the algorithm from the literature [28] (CGA), the algorithm from the literature [29] (BGA), the algorithm from the literature [30] (CSAGA), the algorithm from the literature [31] (BFO), and the algorithm proposed in this paper (IGA). The parameters are set as shown in Table 3. The start point is the grid sequence number 0, and the target point is the grid sequence number 399. Figure 10 shows the fitness profile of Map 4, and the results are also recorded in Table 4. Figure 11 shows the path planning results of the four algorithms in the grid maps. Table 4 has five evaluation metrics, which are average path length, standard deviation of average path length, average smoothness, standard deviation of average smoothness, and minimum path length.

Table 3. Parameter setting.

	Popnum	N	p_c	p_m	p_s	ω	$c1$	$c2$	$c3$
GA	200	50	0.8	0.4	-	-	-	-	-
CGA	200	50	0.8	0.4	-	-	-	-	-
BGA	200	50	0.8	0.4	-	-	-	-	-
CSAGA	200	50	0.8	0.4	-	-	-	-	-
BFO	200	50	-	-	-	-	-	-	-
IGA	200	50	0.8	0.4	0.4	0.7	1	2	3

The results of Experiment 2 show that the minimum path length of IGA in the four maps is 31.5536, 27.4558, 30.9706, and 30.3848, respectively. Although the minimum path length of CSAGA, BGA and BFO is equal to that of IGA, the standard deviation of the average path length is higher than that of IGA.The average smoothness of IGA in the four maps is 4.2333, 2.3, 8.9333, and 9, lower than that of the other five algorithms, and the standard deviation of average smoothness is also lower than that of the other five algorithms. From Figure 10, IGA converges faster than the other five algorithms. Results from Experiment 2 show that the genetic algorithm in this paper has some advantages over the other algorithms compared to the other five algorithms in terms of path length and smoothness. It has the advantages of fast convergence speed, not-easy-to-fall-into local optimum, and strong global search ability, which can accurately and efficiently achieve robot path planning.

Table 4. Experiment 2's simulation results.

Map	Algorithm	Average Path Length	Sd of Path Length	Average Smoothness	Sd of Smoothness	Minimum Path Length
Map 1	GA	38.4305	3.6074	22.4667	5.8057	35.2194
	CGA	33.4206	1.6147	11.9667	3.5378	31.9678
	BGA	31.7907	0.2919	8.0333	1.6291	31.5536
	CSAGA	32.7678	1.2274	12.5333	2.9094	31.5536
	BFO	31.9469	0.3871	7.4000	1.7340	31.5536
	IGA	31.7011	0.2701	4.2333	1.3817	31.5536
Map 2	GA	34.2434	4.6513	17.7	5.2071	30.2842
	CGA	29.2603	1.139	8.5667	3.0477	28.87
	BGA	27.4568	0.0013	3.1333	0.5457	27.4558
	CSAGA	28.5769	1.6553	5.8667	3.2242	27.4558
	BFO	27.4561	0.0003	2.7667	0.6302	27.4558
	IGA	27.4558	0	2.3	0.4661	27.4558
Map 3	GA	34.3555	1.29	13.6	3.5487	33.799
	CGA	32.254	0.7173	10.2333	1.6955	31.799
	BGA	31.5895	0.3526	12	3.5428	30.9706
	CSAGA	31.5873	0.9156	12.8333	2.0356	30.9706
	BFO	31.211	0.4181	9.1667	3.9748	30.9706
	IGA	31.0096	0.1486	8.9333	2.2118	30.9706
Map 4	GA	37.1787	3.7141	20.0667	5.5456	33.5864
	CGA	33.6244	2.3946	13.3	2.9496	32.1421
	BGA	30.7948	0.5579	12.9667	2.6325	30.758
	CSAGA	31.4702	1.5720	12.1333	2.8007	30.3848
	BFO	31.1854	0.8632	10.0333	1.9085	30.3848
	IGA	30.5215	0.4533	9	0.6948	30.3848

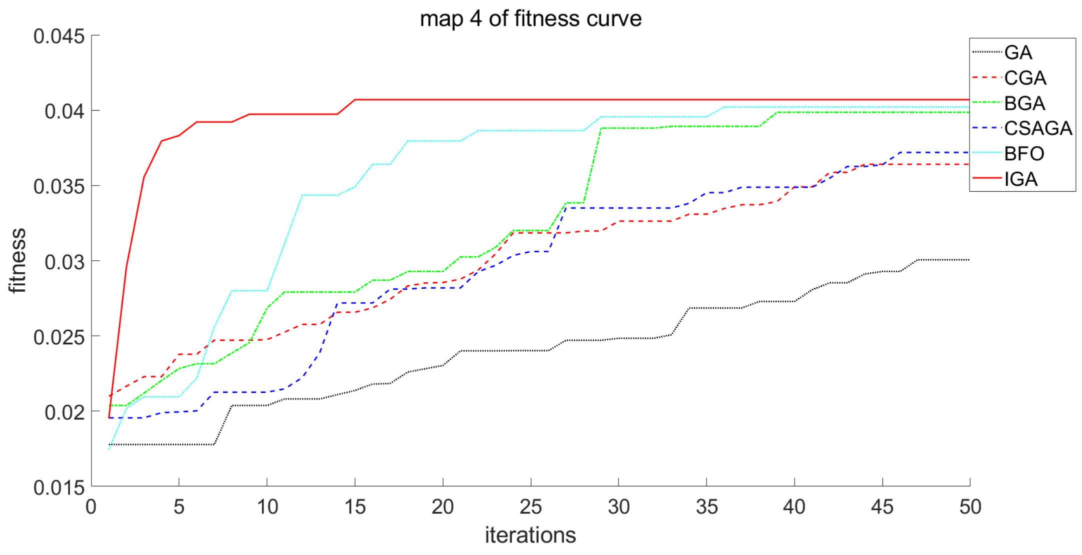

Figure 10. Map 4 of fitness curve.

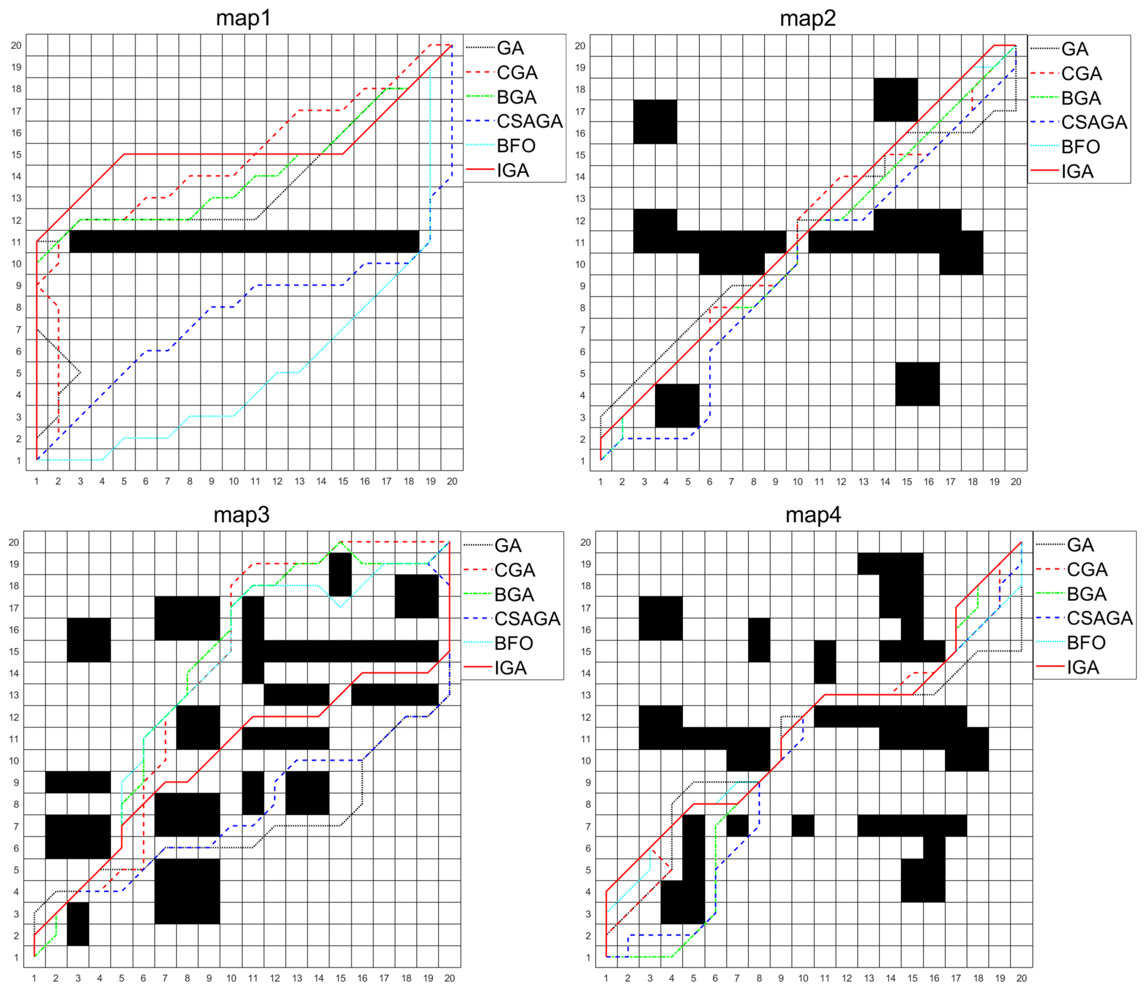

Figure 11. Example of path planning.

6. Conclusions

In this paper, an improved genetic algorithm is proposed to address the shortcomings of the genetic algorithm in terms of population initialization's low efficiency, low quality of the initial population, easy-to-fall-into local optimum, and poor global search ability. Compared with the traditional genetic algorithm, this paper adopts the population initialization method with direction guidance to improve the efficiency of population initialization and the quality of the initial population; then, it combines the non-common point crossover operator, the range mutation operator, the traditional one-point crossover operator, and the traditional one-point mutation operator, which balances the local and global optimization searching; finally, it corrects the whole path by using the simplification operator. Simulation experiments on robot path planning under grid maps are carried out, and the experimental results show that the algorithm in this paper has certain advantages in population initialization efficiency, path optimization, and convergence speed.

We do not use the algorithm in this paper for the robot path planning problem in a dynamic environment and the multi-robot path planning problem. For these two problems, this will be our next research direction.

Author Contributions: Conceptualization, D.P. and J.Z.; methodology, D.P.; software, J.Z.; validation, D.P.; formal analysis, D.P.; data curation, J.Z.; writing—original, draft preparation, J.Z.; writing—review, and editing, D.P.; visualization, J.Z.; supervision, D.P.; project administration, D.P.; funding acquisition, D.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China, grant number 11871059; the Natural Science Foundation of Sichuan Education Department, grant number 18ZA0469; the Sichuan Science and Technology Program, grant number 2022NSFSC0558; and the Research Project on Graduate Education Reform of China West Normal University, grant numbers 2022XM24 and 2024XM05.

Data Availability Statement: All data used in this study can be provided upon request from the corresponding author.

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

References

1. Pena, A.; Tejada, J.C.; Gonzalez-Ruiz, J.D.; Sepúlveda-Cano, L.M.; Chiclana, F.; Caraffini, F.; Gongora, M. An evolutionary intelligent control system for a flexible joints robot. *Appl. Soft Comput.* **2023**, *135*, 110043. [CrossRef]
2. Wang, Y.; Hernandez, A.; Shen, L.; Zhang, H. Path planning of water surface garbage cleaning robot based on improved immune particle swarm algorithm. *AIP Adv.* **2024**, *14*, 025217. [CrossRef]
3. Stenning, B.E. *Path/Action Planning for a Mobile Robot*; University of Toronto: Toronto, ON, Canada, 2013.
4. Sanin-Villa, D.; Rodriguez-Cabal, M.A.; Grisales-Noreña, L.F.; Ramirez-Neria, M.; Tejada, J.C. A Comparative Analysis of Metaheuristic Algorithms for Enhanced Parameter Estimation on Inverted Pendulum System Dynamics. *Mathematics* **2024**, *12*, 1625. [CrossRef]
5. Mo, H.; Xu, L. Research of biogeography particle swarm optimization for robot path planning. *Neurocomputing* **2015**, *148*, 91–99. [CrossRef]
6. Zeng, M.R.; Xi, L.; Xiao, A.M. The free step length ant colony algorithm in mobile robot path planning. *Adv. Robot.* **2016**, *30*, 1509–1514. [CrossRef]
7. Amirjanov, A.; Sobolev, K. Changing range genetic algorithm for multimodal function optimisation. *Int. J. Bio-Inspired Comput.* **2015**, *7*, 209–221. [CrossRef]
8. Zuo, L.; Guo, Q.; Xu, X.; Fu, H. A hierarchical path planning approach based on A$*$ and least-squares policy iteration for mobile robots. *Neurocomputing* **2015**, *170*, 257–266. [CrossRef]
9. Zeng, N.; Zhang, H.; Chen, Y.; Chen, B.; Liu, Y. Path planning for intelligent robot based on switching local evolutionary PSO algorithm. *Assem. Autom.* **2016**, *36*, 120–126. [CrossRef]
10. Mac, T.T.; Copot, C.; Tran, D.T.; De Keyser, R. A hierarchical global path planning approach for mobile robots based on multi-objective particle swarm optimization. *Appl. Soft Comput.* **2017**, *59*, 68–76. [CrossRef]
11. Wang, X.; Yan, Y.; Gu, X. Spot welding robot path planning using intelligent algorithm. *J. Manuf. Process.* **2019**, *42*, 1–10. [CrossRef]
12. Abaas, T.F.; Shabeeb, A.H. Autonomous mobile robot navigation based on PSO algorithm with inertia weight variants for optimal path planning. In Proceedings of the IOP Conference Series: Materials Science and Engineering, Penang, Malaysia, 17–18 April 2020; IOP Publishing: Bristol, UK, 2020; Volume 928, p. 022005.
13. Tao, B.; Kim, J.H. Mobile robot path planning based on bi-population particle swarm optimization with random perturbation strategy. *J. King Saud Univ.-Comput. Inf. Sci.* **2024**, *36*, 101974. [CrossRef]
14. Gao, W.; Tang, Q.; Ye, B.; Yang, Y.; Yao, J. An enhanced heuristic ant colony optimization for mobile robot path planning. *Soft Comput.* **2020**, *24*, 6139–6150. [CrossRef]
15. Zheng, Y.; Luo, Q.; Wang, H.; Wang, C.; Chen, X. Path planning of mobile robot based on adaptive ant colony algorithm. *J. Intell. Fuzzy Syst.* **2020**, *39*, 5329–5338. [CrossRef]
16. Cui, J.; Wu, L.; Huang, X.; Xu, D.; Liu, C.; Xiao, W. Multi-strategy adaptable ant colony optimization algorithm and its application in robot path planning. *Knowl.-Based Syst.* **2024**, *288*, 111459. [CrossRef]
17. Yang, S.; Hong, T.; Zhu, L.k. Forest Fire Prevention Mobile Robot Path Planning Based on Improved Ant Colony Algorithm. *For. Eng.* **2024**, *40*, 152–159.
18. Silva, J.B.; Siebra, C.A.; Nascimento, T.P. A simplified cost function heuristic applied to the A*-based path planning. *Int. J. Comput. Appl. Technol.* **2016**, *54*, 96–105. [CrossRef]
19. Batik Garip, Z.; Karayel, D.; Ozkan, S.; Atali, G. Path planning for multiple mobile robots using A* algorithm. *Acta Phys. Pol. A* **2017**, *132*, 685–688. [CrossRef]
20. Hassani, I.; Maalej, I.; Rekik, C. Robot path planning with avoiding obstacles in known environment using free segments and turning points algorithm. *Math. Probl. Eng.* **2018**, *2018*, 2163278. [CrossRef]
21. Yunqiang, H.; Wende, K.; Lin, C.; Xiaokun, L. Research on multi-objective path planning of a robot based on artificial potential field method. *Int. J. Wirel. Mob. Comput.* **2018**, *15*, 335–341. [CrossRef]
22. Xu, F.; Li, H.; Pun, C.M.; Hu, H.; Li, Y.; Song, Y.; Gao, H. A new global best guided artificial bee colony algorithm with application in robot path planning. *Appl. Soft Comput.* **2020**, *88*, 106037. [CrossRef]
23. Muthukumaran, S.; Sivaramakrishnan, R. Optimal path planning for an autonomous mobile robot using dragonfly algorithm. *Int. J. Simul. Model.* **2019**, *18*, 397–407. [CrossRef] [PubMed]

24. Ajeil, F.H.; Ibraheem, I.K.; Sahib, M.A.; Humaidi, A.J. Multi-objective path planning of an autonomous mobile robot using hybrid PSO-MFB optimization algorithm. *Appl. Soft Comput.* **2020**, *89*, 106076. [CrossRef]
25. Ab Wahab, M.N.; Lee, C.M.; Akbar, M.F.; Hassan, F.H. Path planning for mobile robot navigation in unknown indoor environments using hybrid PSOFS algorithm. *IEEE Access* **2020**, *8*, 161805–161815. [CrossRef]
26. Zhang, T.W.; Xu, G.H.; Zhan, X.S.; Han, T. A new hybrid algorithm for path planning of mobile robot. *J. Supercomput.* **2022**, *78*, 4158–4181. [CrossRef]
27. Duan, S.; Qi, Y.; Zhang, L.; Li, Y.; Wang, F.; Liu, G. Novel Adaptive Path-Smoothening Optimization Method for Mobile Robots. *Int. J. Comput. Methods* **2024**, *21*, 2450005. [CrossRef]
28. Alajlan, M.; Koubaa, A.; Chaari, I.; Bennaceur, H.; Ammar, A. Global path planning for mobile robots in large-scale grid environments using genetic algorithms. In Proceedings of the 2013 International Conference on Individual and Collective Behaviors in Robotics (ICBR), Sousse, Tunisia, 15–17 December 2013; pp. 1–8.
29. Bao, Y.Y.; Liu, Y.; Wang, J.S.; Liu, J.X. Genetic Algorithm Based on Grid Maps for Solving Robot Path Planning Problem. *Eng. Lett.* **2023**, *31*, 1635–1648.
30. Kou, L.; Wang, Y.k.; Zhang, F.F. A Chaotic Simulated Annealing Genetic Algorithm with Asymmetric Time for Offshore Wind Farm Inspection Path Planning. *Int. J.-Bio-Inspired Comput.* **2024**. [CrossRef]
31. Zareian, L.; Rahebi, J.; Shayegan, M.J. Bitterling fish optimization (BFO) algorithm. *Multimed. Tools Appl.* **2024**, *83*, 75893–75926. [CrossRef]
32. Prakash, S.; Vidyarthi, D.P. A hybrid immune genetic algorithm for scheduling in computational grid. *Int. J. Bio-Inspired Comput.* **2014**, *6*, 397–408. [CrossRef]
33. Hayes-Roth, F. Review of "Adaptation in Natural and Artificial Systems by John H. Holland", The U. of Michigan Press, 1975. *Acm Sigart Bull.* **1975**, *53*, 15. [CrossRef]
34. Elhoseny, M.; Shehab, A.; Yuan, X. Optimizing robot path in dynamic environments using genetic algorithm and bezier curve. *J. Intell. Fuzzy Syst.* **2017**, *33*, 2305–2316. [CrossRef]
35. Patle, B.; Parhi, D.; Jagadeesh, A.; Kashyap, S.K. Matrix-Binary Codes based Genetic Algorithm for path planning of mobile robot. *Comput. Electr. Eng.* **2018**, *67*, 708–728. [CrossRef]
36. Kwaśniewski, K.K.; Gosiewski, Z. Genetic algorithm for mobile robot route planning with obstacle avoidance. *Acta Mech. Et Autom.* **2018**, *12*, 151–159. [CrossRef]
37. Suresh, K.; Venkatesan, R.; Venugopal, S. Mobile robot path planning using multi-objective genetic algorithm in industrial automation. *Soft Comput.* **2022**, *26*, 7387–7400. [CrossRef]
38. Ab Wahab, M.N.; Nazir, A.; Khalil, A.; Ho, W.J.; Akbar, M.F.; Noor, M.H.M.; Mohamed, A.S.A. Improved genetic algorithm for mobile robot path planning in static environments. *Expert Syst. Appl.* **2024**, *249*, 123762. [CrossRef]