Hindawi
Journal of Control Science and Engineering
Volume 2020, Article ID 5287189, 12 pages
https://doi.org/10.1155/2020/5287189

Research Article

A Novel Parallel Ant Colony Optimization Algorithm for Warehouse Path Planning

Junqi Yu,[1] Ruolin Li,[1] Zengxi Feng ,[1] Anjun Zhao,[1] Zirui Yu,[1] Ziyan Ye,[1] and Junfeng Wang[2]

[1] *Xi'an University of Architecture and Technology, Xi'an 710055, China*
[2] *Huafa Architectural Design Consulting Ltd., Zhuhai 519000, China*

Correspondence should be addressed to Zengxi Feng; 23616006@qq.com

Received 14 November 2019; Accepted 7 July 2020; Published 1 August 2020

Academic Editor: Radek Matušů

In order to improve the working efficiency of automated guided vehicles (AGVs) and the processing efficiency of fulfilling orders in intelligent warehouses, a novel parallel ant colony optimization algorithm for warehouse path planning is proposed. Through the interaction of pheromones among multiple subcolonies, the coevolution of multiple subcolonies is realized and the operational capability of the algorithm is improved. Then, a multiobjective function with the object of the shortest path and the minimum number of turns of the AGV is established. And the path satisfying this objective function is obtained by the proposed algorithm. In addition, the path is further smoothed by reducing the number of intermediate nodes. The results show that the stability and convergence rate of the algorithm are faster and more stable, compared to other algorithms, in generating paths for different complexity maps. The smoothing treatment of the path significantly reduces the number of turns and the path length in the AGV driving process.

1. Introduction

Nowadays, automated guided vehicles (AGVs) are often used to move goods in common logistics storage spaces such as intelligent warehouses and automated wharfs. AGVs are required to search for a best path in a given working environment according to certain goals (e.g., shortest time, shortest energy consumption, shortest distance). A reasonable AGV driving path not only improves the goods turnover rate and order fulfillment efficiency of the warehouse, but also makes the AGV more stable during driving. Therefore, the path planning of AGVs in intelligent warehouses is mainly studied in this paper.

In path planning, the goal is to find a collision-free path from the starting position to the target position while optimizing one or more objectives (e.g., path length, smoothness, security) with a reasonable method [1]. The applications of path planning are very extensive. Scholars have reported results on mobile robot path planning using a variety of approaches including the A$*$ algorithm [2], Dijkstra's algorithm [3, 4], ant colony optimization [5, 6], genetic algorithm [7, 8], particle swarm optimization [9], fuzzy control algorithm [10], and other intelligent optimization algorithms. Each algorithm has its own advantages and limitations under different performance indicators.

The ant colony optimization (ACO) algorithm, initially proposed by Marco Dorigo, is a metaheuristic approach inspired by ants' foraging behavior and used to solve the traveling salesman problem (TSP) [11]. Later, various improved ant colony algorithms have been proposed and applied to the path planning problem. The ACO algorithm has the advantages of strong robustness, excellent distributed computer mechanism, and easy integration with other methods [12]; it is a commonly used method to solve the path planning problem. The ACO algorithm adopts a positive feedback approach in which a roulette selection is used to choose the direction of the ants. The algorithm converges when the iterations reach a certain number of times. However, in this positive feedback algorithm, achieving fast convergence and avoiding premature stagnation are contradictory. Many scholars are seeking the balance between the two problems. The max–min ant system (MMAS) proposed in [13] avoids the stagnation of the algorithm to some extent by restricting the upper and lower limits of pheromones on the path. However, when the solution is sparsely distributed, the convergence speed slows down. To address the convergence problem, numerous solutions have been proposed. The algorithm proposed in [14] divides a total ant colony into several subcolonies. Each ant colony uses the MMAS algorithm and completes a multicolony coevolution through a migration operator connection; this improves the convergence speed of the algorithm. However, because of too much dependence on the elite ants, the pheromone local growth is too fast and the solving ability is reduced. An alternative algorithm [15] reconstructs the heuristic information of an ant colony by utilizing the potential field force in the artificial potential field force method. The authors propose an improved potential field force ant colony algorithm. The convergence speed of the algorithm is improved, but the algorithm can readily stagnate prematurely. According to the zero point theorem, [16] assigns different initial pheromones to different raster positions, which reduces the blindness of the ant colony search and improves the search capability of the algorithm. However, the algorithm is too dependent on the initial pheromone distribution and is not adaptable to complex maps. The algorithm proposed in [17] firstly initializes pheromones of an ant colony with random values and then performs a crossover mutation on pheromones in the later stage. This improves the search capability of the algorithm but reduces the convergence speed.

The previous work about improving the operation effect of the ACO algorithm focused on the intervention of certain parameters to change the node selection mode of ants in the population or to influence the pheromone updating mode of the ant population. Less consideration has been given to the pheromone interactions between multiple ant colonies and the cogrowth of multiple subcolonies in the iteration process, so as to improve the overall path search capability of the ant colonies by using the search capability of the subcolonies themselves. Furthermore, most of the relevant research only pursues the shortest path of the AGV and does not consider the problem that the AGV is not smooth enough due to too many turns in the operation process. Therefore, research is needed to effectively reduce the number of turns in the AGV driving process.

Based on the above analysis, a warehouse map model is firstly built and a novel algorithm is proposed to generate the shortest AGV path with the least number of turns. It is a parallel-ranking ant colony optimization (P-RACO) algorithm. Compared with other algorithms, it is proven that the algorithm has faster convergence speed and better stability. Then, on the basis of the first stage of the path generation, the algorithm proved to be faster and more stable. By reducing the intermediate nodes, the path is smoothed, the number of turns and the length of the path are reduced, and the actual driving path of AGV is more stable and feasible.

2. Problem Description and Model Building

2.1. Problem Description and Assumptions. In an intelligent warehouse, when receiving the task, an AGV is required to begin from the starting point, bypassing barriers to complete the order. Therefore, a shortest path with minimum number of turns is needed to ensure the efficiency of the warehouse operation and the driving stability of the AGV.

The following four assumptions are made in this research: ① the warehouse map is known; ② the warehouse grid map is divided into passable and nonpassable grids; ③ the AGV is in good condition; ④ the starting point and end point of a task are known.

2.2. Model Building. The objective function is to minimize the number of turns and the length of the path. A multiobjective path optimization model is established. The P-RACO is used to solve the problem to obtain the path which can meet the requirements of these two constraints to the greatest extent.

2.2.1. Warehouse Environment Model. Through the analysis of an actual warehouse environment, the AGV working environment is determined to be a two-dimensional static environment that is divided into shelves (barriers) and AGV driving channels. The grid method is simple and effective; the use of a grid map can greatly reduce the complexity of the warehouse environment modeling. Therefore, the grid method is used to divide the working environment. In the simulation program, the driving channel is a passable grid, which is identified by 0, and the obstacle is represented by a nonpassable grid, which is represented by 1. The grids are marked (passable, nonpassable) and identified using two-dimensional rectangular coordinates. Figure 1 illustrates a $10 * 10$ grid. The problem of path planning can be simplified to the problem of finding a better subset of the drivable grid. The connection of the center point of the ordered grids is the path planned by the algorithm.

The barrier grids are black, barrier-free grids are white, S is the grid's identifying number, and L is the edge length of the grid. The corresponding relationship between the grid number and the grid center coordinate is as follows:

$$\begin{cases} x = \bmod(S, L) - 0.5, \\ y = S + 0.5 - \operatorname{ceil}\left(\dfrac{S}{L}\right). \end{cases} \quad (1)$$

2.2.2. Objective Function of Minimum Number of Turns.

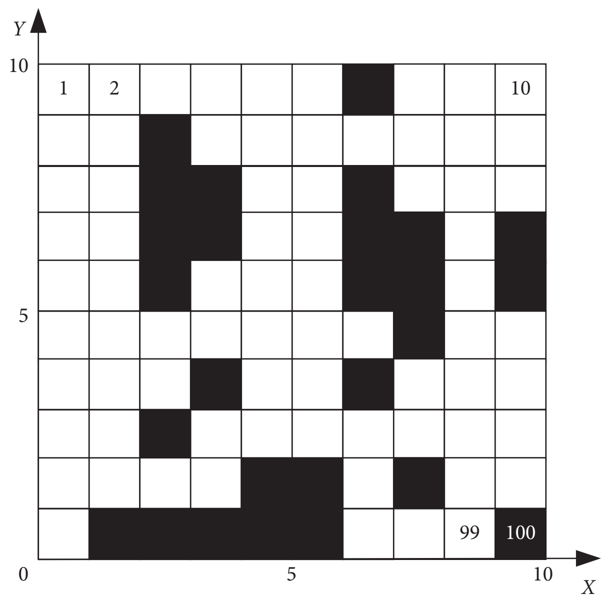

Figure 1: Grid coordinates and grid number arrangement.

In practical environments, fewer turns on paths can reduce the overall mechanical loss and prolong the service life of an AGV. The objective function is as follows:

$$\min N(d) = \sum_{n=1}^{n=1} d_n, \tag{2}$$

$$\mathrm{d}_n = \begin{cases} 0\ , & n = 1 \text{ or } k_n = k_{n-1}, \\ 1, & k_n \neq k_{n-1}, \end{cases} \tag{3}$$

$$k_n = \frac{y_{n+1} - y_n}{x_{n+1} - x_n}, \tag{4}$$

$$k_{n-1} = \frac{y_n - y_{n-1}}{x_n - x_{n-1}}, \tag{5}$$

where $N(d)$ is the number of turns in the path; d_n is the number of routes turning on the nth node; (x_n, y_n), (x_{n+1}, y_{n+1}) are the coordinates of the adjacent grid centers passing through the path; k is the slope; and n is the sequence number of the grid centers of the path.

2.2.3. Objective Function of Shortest Path. In practical environments, an AGV can save time and improve the efficiency of the whole warehouse by seeking the shortest driving path. The objective function is as follows:

$$\min L(P) = \sum_{i=1}^{N(d)} L(P_i, P_{i+1}) = \sum_{i=1}^{N(d)} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}, \tag{6}$$

where $L(P)$ denotes the sum of the shortest paths; $L(P_i, P_{i+1})$ is the distance between point P_i and P_{i+1}; (x_i, y_i), (x_{i+1}, y_{i+1}) are the coordinates of the center point of the current grid and the next point; and $N(d)$ is the number of turns in the path.

2.2.4. Utility Function. In order to simplify the model, the length of each grid in the grid map is set to be one unit length (the dimension of the objective function should be unified according to the actual unit length in its practical application). The objective function is calculated by linear weighting, so that the solution can satisfy both the shortest path and the minimum number of turns. The relationship between the above two objective functions and the utility function $Z(L, N)$ is established. Through this coordination, the multiobjective problem is transformed into a traditional single-objective solving problem. The utility function is as follows:

$$Z(L, N) = \omega_1 L(P) + \omega_2 N(d), \tag{7}$$

where ω_1 is the path length's weight coefficient; ω_2 is the turn number's weight coefficient; $L(P)$ is the summation of the shortest path; and $N(d)$ is the number of turns in the path.

3. Proposed Algorithm: Parallel-Ranking Ant Colony Optimization

As an evolutionary algorithm, the ACO algorithm has shown great potential to solve combinatorial optimization problems. However, like other evolutionary algorithms, it has shortcomings in terms of its convergence speed and its ease of falling into a local minimum solution. In order to address these problems, it is necessary to redesign the search strategy of the ant colony.

3.1. Algorithms for Solving the Shortest Path. In genetic algorithm, a ranking selection mechanism is used to improve the search speed. First, the population is classified according to fitness; then, the probability of being selected depends on the order of the individuals. The higher the fitness, the better the individual is, and the higher the probability of it being selected in the next iteration. This ranking and selection concept of genetic algorithms is extended to the ant colony algorithm. After all of the ants complete an iteration, a selection is made. The w-1 ants ranked first in the ant colony in addition to the ants constituting the best solution (up to the current iteration) are selected, and the pheromones of the paths of the w ants are updated. This algorithm is called the ranking ant colony optimization (RACO).

However, an ant colony algorithm based on the ranking optimization accumulates numerous pheromones in the local area very early. Although the speed is improved, it reduces the diversity of solutions in each generation. Consequently, the algorithm can readily fall into a local optimum. To address this problem, the ant colony is divided into several subcolonies to grow together, and the pheromone of the better individual in a subcolony is transmitted to another subcolony. This is accomplished through an information interaction between the subcolonies. The transmission ensures that the pheromone accumulation of each subcolony has the correct direction. The flowchart of the algorithm for a single subcolony is shown in Figure 2, and the schematic diagram of the

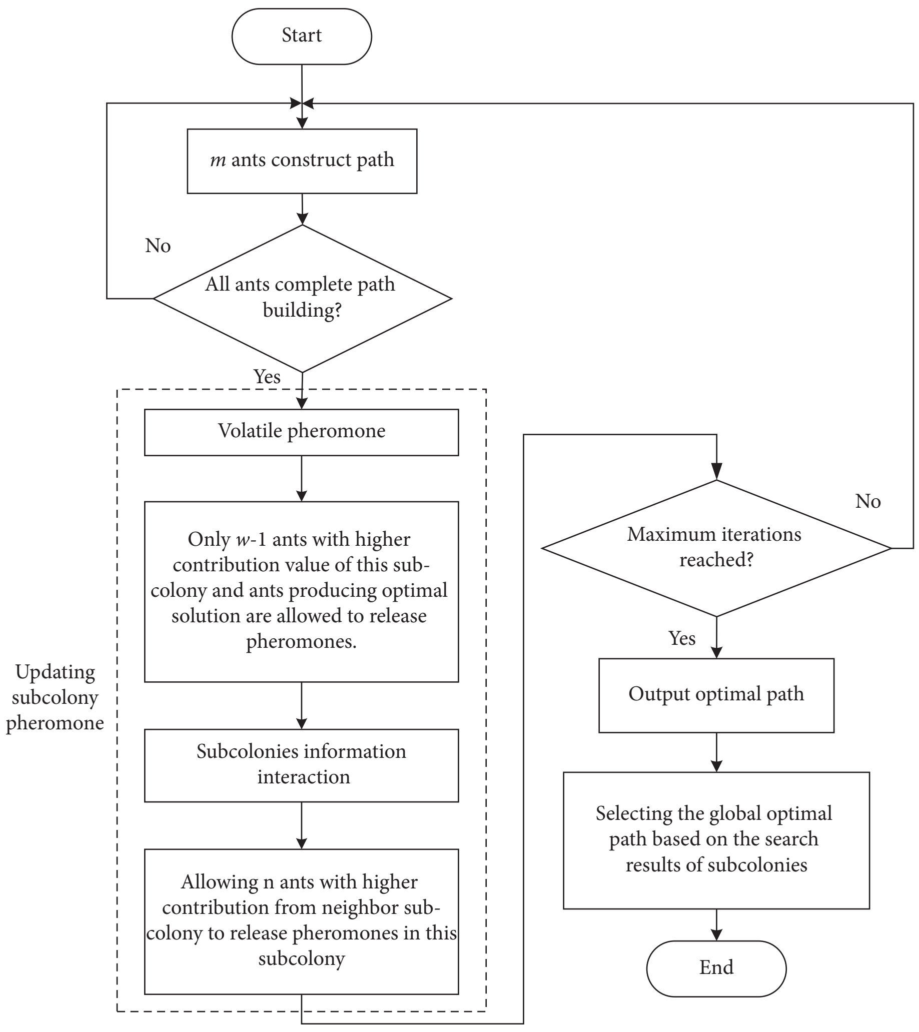
Start
m ants construct path
No
All ants complete path building?
Yes
Volatile pheromone
Only *w*-1 ants with higher contribution value of this subcolony and ants producing optimal solution are allowed to release pheromones.
Updating subcolony pheromone
Subcolonies information interaction
Allowing n ants with higher contribution from neighbor subcolony to release pheromones in this subcolony
No
Maximum iterations reached?
Yes
Output optimal path
Selecting the global optimal path based on the search results of subcolonies
End

Figure 2: Single sub-ant colony operational flowchart.

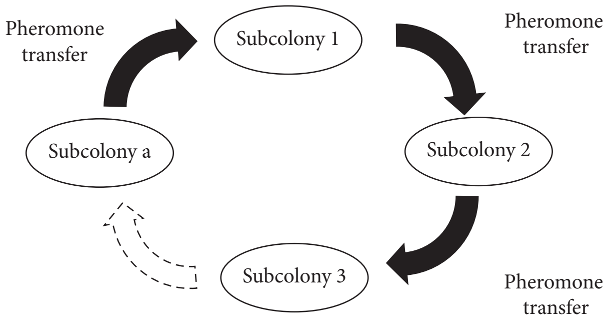

Figure 3: Diagram of subcolony information interactions.

information interaction for each subcolony is shown in Figure 3. All the steps of the parallel-ranking ant colony optimization (P-RACO) are described in more detail below.

Before the path construction, a large ant colony is divided into several subcolonies such that each subcolony has *m* ants. When the ants in each subcolony construct the path, the ants choose the next node by roulette. When ant *k* is in node *i*, the probability of selecting node *j* (node *j* is not visited) is as follows:

$$P_{ij}^{k} = \frac{\left[\tau_{ij}\right]^{\alpha}\left[\eta_{ij}\right]^{\beta}}{\sum_{i\in N_i^k}\left[\tau_{ij}\right]^{\alpha}\left[\eta_{ij}\right]^{\beta}}. \quad (8)$$

If the node has been accessed, $P_{\mathrm{ij}}^{k} = 0$. τ_{ij} is the pheromone on edge (i, j) and η_{ij} is the heuristic information of edge (i, j). For a general path search problem, η_{ij} takes the reciprocal of the path length, and α and β are the algorithm parameters.

When each ant generates a path, it volatilizes part of the pheromone that exists on the path before updating the pheromone, as follows:

$$\tau_{ij} \longleftarrow (1-\rho)\tau_{ij}, \quad (9)$$

where ρ is a pheromone volatile factor. The ants of a subcolony are sorted according to the value of the constructed path's utility function ($Z_1 \leq Z_2 \leq \ldots \leq Z_{\mathrm{m}}$). At the same time, according to the length of the path constructed by the ants in the subcolony, n ants (i.e., the first n-1 ants with a higher contribution, and one ant with the best path constructed so far) are selected from each subcolony and passed to the neighboring ant colony. Moreover, the path information of the n superior ants is received from the neighboring ant colonies. This parallel-ranking ant colony optimization system allows n superior ants to release pheromones on the path of their own colony. The pheromone release rule of this system is related to the order and contribution value of the ants: the amount of pheromones released by the ants is proportional to their ranking. In addition, the amount of pheromones transmitted by neighbors is obtained according to the ranking. The pheromone update formula is as follows:

$$\tau_{ij} \longleftarrow \tau_{ij} + \sum_{r=1}^{w-1} (w-r)\Delta\tau_{ij}^{r} + w\Delta\tau_{ij}^{bs} + \sum_{q=1}^{n-1} (n-q)\Delta\tau_{ij}^{q} + n\Delta\tau_{ij}^{nbs}, \tag{10}$$

where $\Delta\tau_{ij}^{r}$ is the amount of pheromones released by the first ant in its path. When edge (i, j) is on the path constructed by ant r, then $\Delta\tau_{ij}^{r} = 1/C^{r}$. When edge (i, j) is not on the path constructed by ant k, then $\Delta\tau_{ij}^{r} = 0$. When the edge (i, j) is on the optimum path up to now, then $\Delta\tau_{ij}^{bs} = 1/C^{bs}$. Otherwise, $\Delta\tau_{ij}^{r} = 0$, $\Delta\tau_{ij}^{q}$ is the amount of pheromones released by the first n-1 ants in their path. Furthermore, $\Delta\tau_{ij}^{nbs}$ is the best solution composed by the neighboring ant colony up to now; r is the ranking sequence of the ants in their own colony; and q is the ranking sequence of the neighboring ants in their own colony.

3.2. Avoiding the "Dead Corner of Path" Problem. Warehouse environments vary with the complexity of their functions. In a complex environment, ants may fall into a dead corner state in the process of searching for solutions (i.e., no target point is found and no mobile nodes are present). This state is illustrated in Figure 4.

Figure 4 shows that ants fall into a dead corner at the point P. To solve this problem, [18] adopts the method of an early death to make the ants in a dead corner die. As a result, the pheromones on this path are not updated. However, this method is not conducive to the search of a global optimal solution when more ants fall into dead corners in the ant colony. One approach proposed to address this issue is to abandon the generation of the path and begin a new search from the starting point [19]. However, this method increases the search time of the algorithm and cannot avoid the algorithm falling into the dead corner again. Therefore, this paper proposes that when an ant is in a dead corner state, it is allowed to retreat one step and update the search tabu table. This approach allows the current ant to reselect a mobile node and punish the pheromone on the edge. The pheromone penalty function is as follows:

$$\tau_{rs} = (1-\lambda)\tau_{rs}, \tag{11}$$

where $(1-\lambda)$ is the corresponding penalty coefficient and τ_{rs} is the pheromone on the return path.

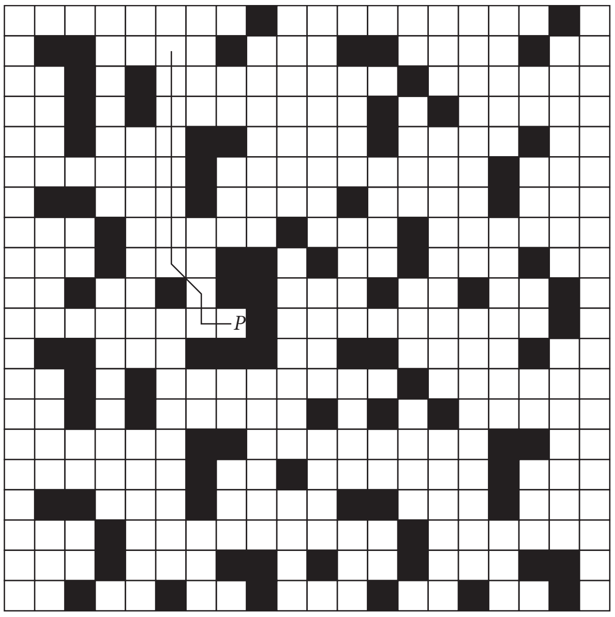

Figure 4: Dead corner of a path state diagram.

This solution can improve the global search capability of the algorithm and effectively avoid ants falling into the dead corner at the same location.

3.3. Path Smoothing. A smooth and executable path with less turns is an important part of warehouse AGV path planning. Too many turns during the operation of the AGV will significantly increase the mechanical wear of the machine and reduce its service life. In order to reduce this impact and make the obtained path more applicable to the actual robot, it is necessary to smooth the path [20].

As the warehouse map model is a grid map, the path generated in the first stage is a broken line composed of straight lines; these connect the centers of each grid. In practice, when there are no barriers on the road, several grids can be crossed directly. The center points of nonadjacent grids can be connected to reduce the number of turns and the length of the path, so as to improve the overall efficiency of the warehouse. The flowchart is shown in Figure 5, in which $\{P_N\}$ is the initial path sequence.

4. Experimental Results and Analysis

Firstly, the path solving process is introduced, and the performance of the proposed algorithm is tested with the classical TSP model and compared with other algorithms. Then, the proposed algorithm is used to solve the AGV path planning in the 30 ∗ 30 and 35 ∗ 35 warehouse grid graphs and compared with other multiobjective algorithms. Finally, the generated path is smoothed and the data before and after the path smoothing is compared. The computing environments are Windows 10, i5 CPU, 8 GB memory, and MATLAB 2018.

4.1. Path Solving Process

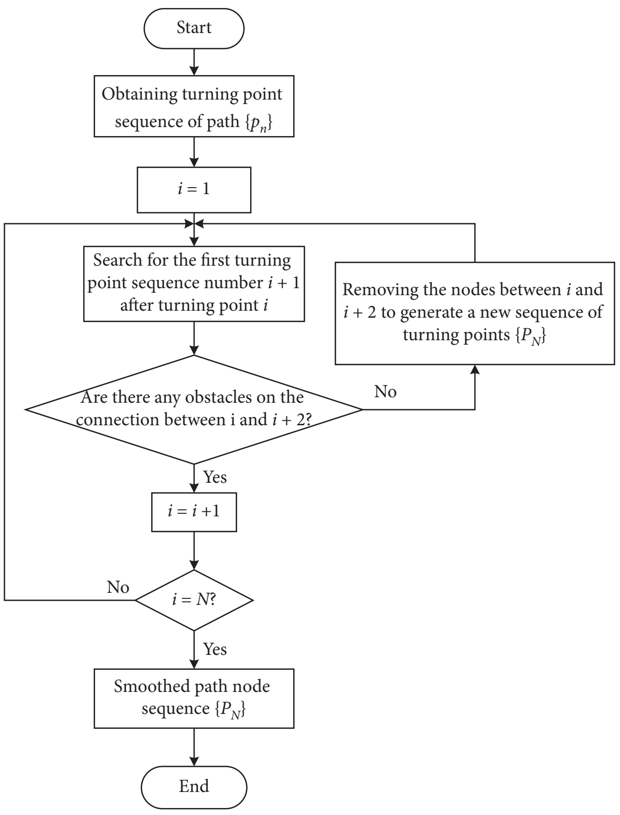
Start
Obtaining turning point sequence of path $\{p_n\}$
$i = 1$
Search for the first turning point sequence number $i + 1$ after turning point i
Removing the nodes between i and $i + 2$ to generate a new sequence of turning points $\{P_N\}$
Are there any obstacles on the connection between i and $i + 2$?
No
Yes
$i = i + 1$
No
$i = N$?
Yes
Smoothed path node sequence $\{P_N\}$
End

Figure 5: Path smoothing processing flowchart.

Step 1. According to the actual warehouse environment, map the distribution of obstacles in the intelligent warehouse to a grid graph and build a warehouse map model.

Step 2. Use the P-RACO algorithm to obtain the optimal path.

Step 3. Output the path if the calculation reaches the predetermined termination condition; otherwise, return to Step 2.

Step 4. Smooth the path generated by Step 2 by deleting invalid intermediate nodes.

Step 5. Verify the performance of the P-RACO in solving multiobjective warehouse problems, and compare it with other algorithms.

Step 6. End.

4.2. Performance Test of Algorithms. The TSP is a typical NP-hard problem, and it is often used to test the performance of various optimization algorithms. Most of the research on picking operations abstracts the picking path into the TSP model.

Firstly, the TSP model is used to test the optimization performance of the P-RACO algorithm. The test results are compared with those of ACO, RACO, and the improved ant colony (IAC) proposed in the literature [17]. In the TSP model, the number of cities is set to 31, the number initial ant colonies is set to 45, the maximum number of iterations is 200, the number of subcolonies is set to 5, and the number of ants per subcolony is 35.

Each algorithm runs 50 times and records the maximum value (Max), minimum value (Min), average value (Avg), and standard deviation (Sd) of the runs.

When solving the TSP, it can be seen from Table 1 that the convergence speed and stability of ACO are poor. RACO can quickly accumulate local pheromones, and the convergence speed is relatively fast. However, the accumulated direction of pheromones in the initial stage of the algorithm is prone to errors, so the algorithm is not stable enough. In IAC, the convergence speed and stability are improved, due to the promotion of the initial pheromone. In this case, the optimal solution is rapidly found in the fifth generation. The approach proposed in this paper further improves the IAC algorithm. Because of the cooperation of multiple populations, the early pheromone accumulation has better directionality, the algorithm converges quickly, and the optimal solution can be obtained as soon as the third generation. The average value of the path length and the number of iterations obtained by P-RACO are smaller than those of the other three algorithms. This shows that the overall convergence speed of the P-RACO algorithm is faster than those of the other three algorithms. In addition, the proposed algorithm's ability to search for the optimal solution is also better than those of the other three algorithms. The standard deviation of the path length and iteration times

Table 1: 50-simulation results of four algorithms for the TSP model.

	Path length				The number of iteration			
	Max	Min	Avg	Sd	Max	Min	Avg	Sd
ACO	3369.457	3198.345	3141.356	58.754	96	13	17.65	15.678
RACO	3280.290	3089.658	3189.875	29.747	57	7	10.35	5.869
IAC	3179.658	3086.897	3124.121	27.864	25	6	8.64	5.786
P-RACO	3154.785	3048.856	3104.457	21.654	18	3	5.89	3.879

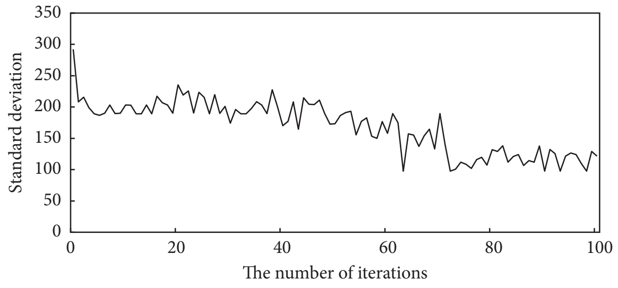

Figure 6: Diversity curve.

for all of the algorithms indicates that the stability of the P-RACO algorithm is better.

4.3. Diversity Analysis of Solutions. After each iteration of the RACO algorithm, pheromones are rewarded to the ants with higher contributions. Although this behavior makes ants approach the optimal solution faster through the rapid accumulation of pheromones, it also reduces the diversity of the ant solutions. In this paper, P-RACO uses a multiple-ant-colony search to increase the diversity of the whole ant colony solution.

The standard deviation $D\,(p)$ of the solution searched by all of the ant colonies in each iteration is used as a function to analyze the diversity of the algorithm. The calculation formula is as follows:

$$D\,(P) = \sqrt{\frac{\sum_{r=1}^{100}\left(L_r^p - \overline{L_p}\right)^2}{M-1}}, \qquad (12)$$

where L_r^p denotes the length of the path searched by the r ant in the pth iteration generation, and $\overline{L}_p$ denotes the average value of the path searched by all of the ants in the r generation. The diversity curves of the first 100 iterations of the algorithm are shown in Figure 6.

Figure 6 shows that, in the initial stage of the algorithm, the solution has a high diversity. As the iterating of the algorithm progresses, the diversity of the solution gradually decreases, but the standard deviation is always greater than 0. This shows that the algorithm does not appear to enter a stagnation state, and the solutions of each generation have good diversity.

4.4. Initial Path Generation and Algorithmic Comparison. In this section, multiobjective path planning is carried out on the simplified two-dimensional grid map of the warehouse, and the model is established according to the common warehouse size. The warehouse maps are set to $30 * 30$ and $35 * 35$ sizes. The number of barrier grids is 200 and 400, respectively. Four algorithms (ACO, TACO, IAC, and P-RACO) are used to solve the problem.

4.4.1. Experiments on the $30 * 30$ Map. For the $30 * 30$ environment, Figure 7 shows the path planning results of the four algorithms, Figure 8 shows the convergence curve of the four algorithms, and Table 2 shows the results of the 50 runs of the four algorithms.

From Figure 8, it can be seen that the ACO algorithm has obviously fallen into a local optimum, mainly because of the uneven distribution of pheromones in the initial iteration of the algorithm, the weak positive feedback effect, and the inability to choose a better global path. Consequently, there are many turns, a longer path, and a slower convergence speed. The RACO and IAC algorithms can avoid the relatively poor paths at the beginning of the iterations and improve the convergence speed of the algorithm. However, because of the low diversity of the search solutions, they still fall into local optimal paths. In this paper, the P-RACO algorithm can complete the solution quickly; from Table 2, it can be seen that the shortest time to search for the optimal solution is eight iterations. The multi-subcolony search of the P-RACO significantly improves the search capability of the algorithm, avoids the algorithm falling into a local optimum, and makes the quality of the solution better. From Table 2, it can be seen that the P-RACO has a smaller standard deviation, indicating that the stability of the algorithm is better.

4.4.2. Experiments and Results of a $35 * 35$ Complex Map. For the $35 * 35$ environment, Figure 9 shows the path planning results of the four algorithms, Figure 10 shows the convergence curve of four algorithms, and Table 3 shows the results of the 50 runs of the four algorithms.

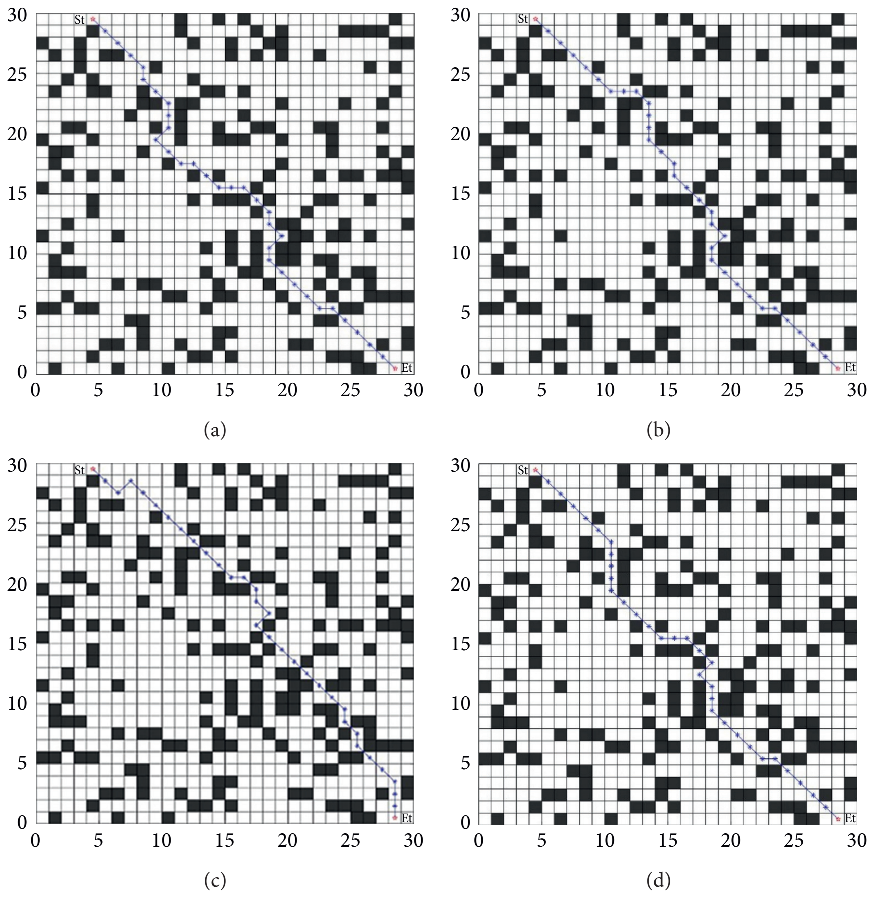

Figure 7: Path planning result diagrams of the four algorithms in the 30 ∗ 30 map. (a) ACO. (b) RACO. (c) IAC. (d) P-RACO.

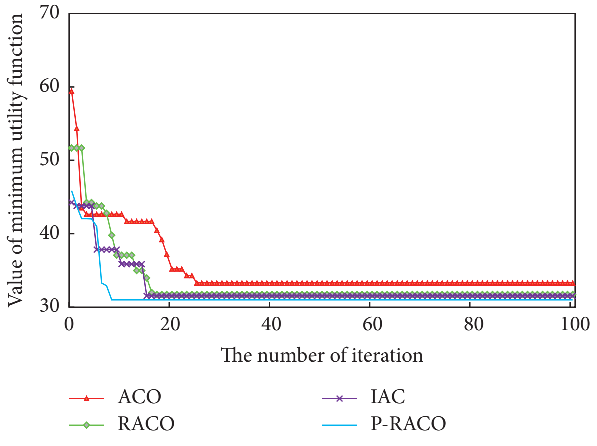

Figure 8: Convergence curves of the four algorithms in the 30 ∗ 30 map.

Table 2: 50-simulation results of the four algorithms in the 30 ∗ 30 map.

	Max	Min	Avg	Sd	The number of iterations			
					Max	Min	Avg	Sd
ACO	36.578	30.726	32.889	1.446	33	20	24.98	3.346
RACO	34.768	30.890	32.776	1.267	25	17	17.49	2.879
IAC	34.960	30.352	31.097	1.188	26	14	16.75	2.245
P-RACO	34.086	29.872	30.463	0.902	10	8	8.89	0.508

With the increase of map complexity, the stability of the ACO algorithm becomes worse. Figure 10 shows that the ACO and RACO algorithms still fall into a local optimum when searching for solutions. The IAC algorithm can search for the optimal solution better through pheromone cross-mutation in the later stage of the algorithm. The P-RACO algorithm can obtain smaller initial values in the early stage, which shows that the algorithm has a better direction when pheromones accumulate. In addition, the convergence speed of the P-RACO algorithm is higher. From the mean and standard deviation of the three experiments in Table 3, when the map complexity increases to 35 ∗ 35, the average convergence algebra of the ACO algorithm increases by 7.51 generations, and the RACO and IAC algorithms increase by 7.11 generations and 6.59 generations, respectively, while the P-RACO algorithm only increases by 1.36 generations. From Table 3, it can be seen that, with the increase of environmental complexity, the solution effect or the stability of the P-RACO algorithm is obviously better than that of the other three algorithms.

The experimental results show that the algorithm in this paper converges after 100 iterations at most when solving the path planning problem of different size warehouses.

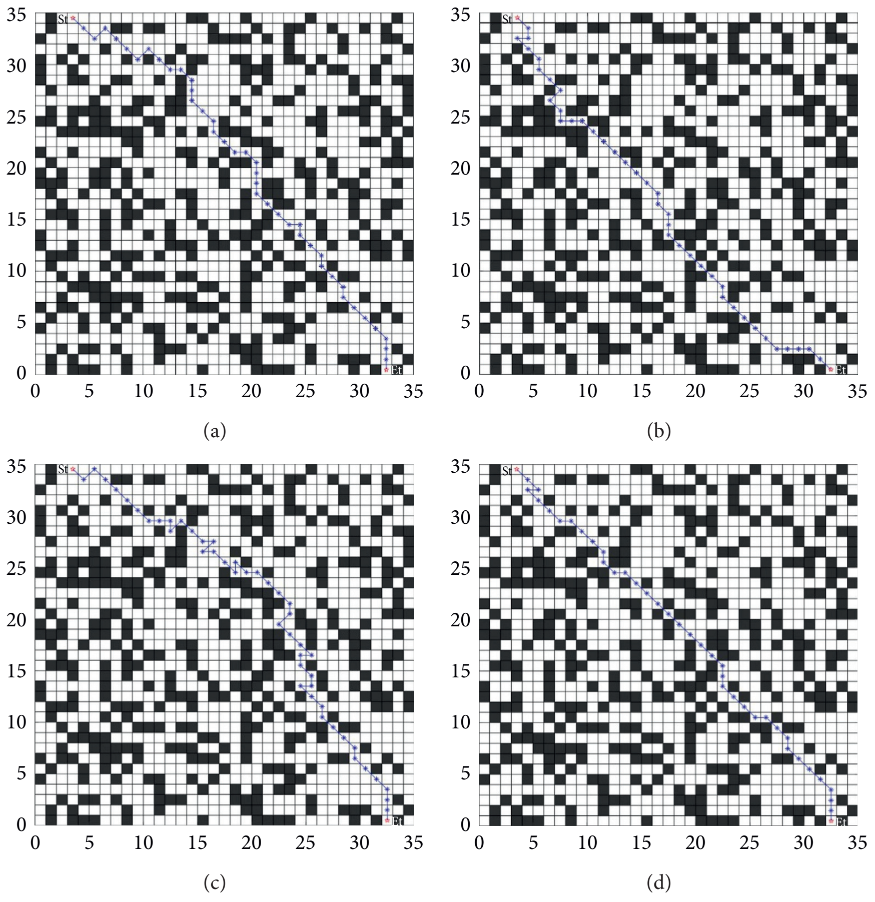

Figure 9: Path planning result diagrams of the four algorithms in the 35 ∗ 35 map. (a) ACO. (b) RACO. (c) IAC. (d) P-RACO.

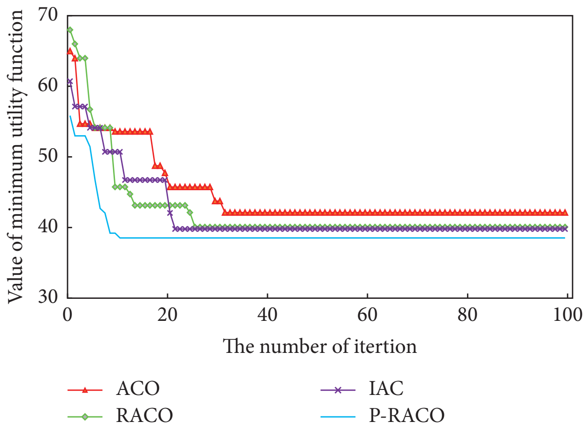

Figure 10: Path planning result diagrams of the four algorithms in the 35 ∗ 35 map.

After 100 iterations, the ACO algorithm does not search for even better paths, and the value of the minimum utility function does not change any more. Therefore, the maximum number of iterations of the experimental results is set to 100.

4.5. Path Smoothing. In this experiment, the turning center of the generated optimal path is processed to reduce the number of invalid paths and the number of turns of the AGV. The intermediate nodes of the generated optimal path are processed as illustrated in Figure 5, and a comparison of the smoothed path is shown in Figure 11. Table 4 compares the data before and after the path smoothing improvements.

Table 3: 50-simulation results of the four algorithms in the 35 ∗ 35 map.

	Max	Min	Avg	Sd	The number of iterations			
					Max	Min	Avg	Sd
	49.345	40.125	45.430	3.568	42	27	32.49	2.970
ACO	45.384	40.345	42.456	2.346	34	19	24.60	2.097
	47.293	38.872	42.236	1.870	28	19	23.34	1.083
P-RACO	43.986	38.262	40.346	0.866	12	9	10.20	0.533

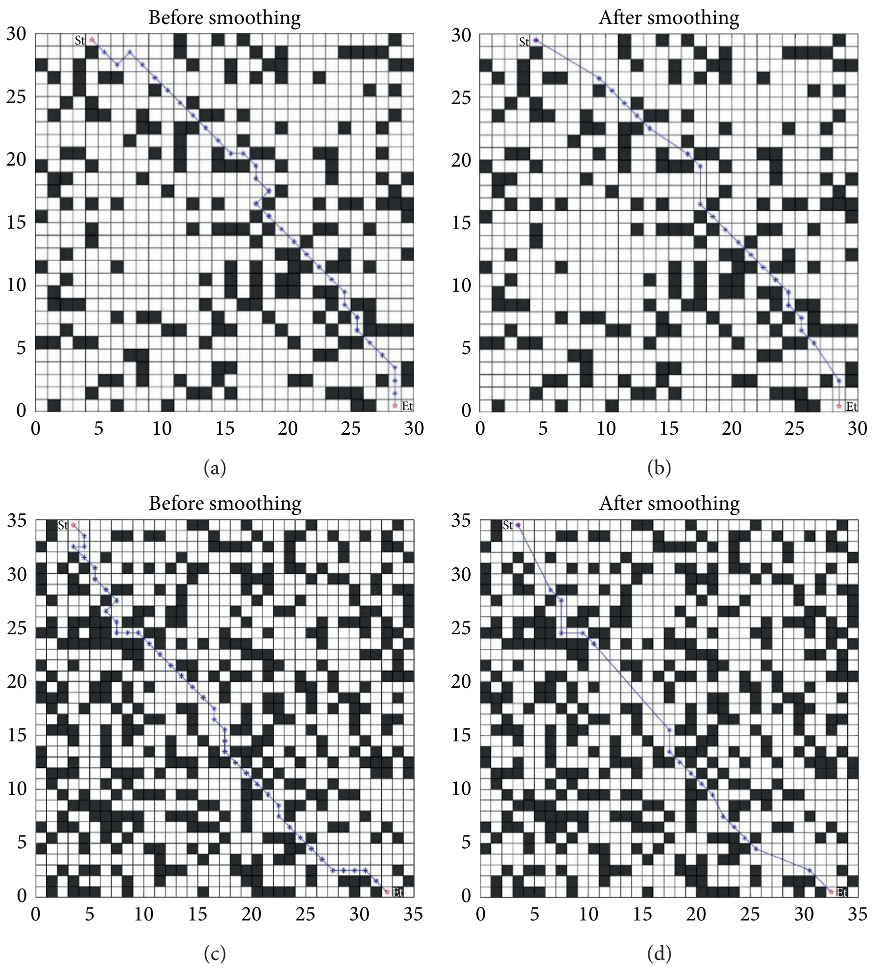

Figure 11: Smoothing route comparison. (a) Comparison of route smoothing in the 30 ∗ 30 map. (b) Comparison of route smoothing in the 30 ∗ 35 map.

Table 4: Data comparison before and after the path smoothing.

Map size	Length before smoothing	Length after smoothing	Reduction rate	Turns before smoothing	Turns after smoothing	Reduction rate (%)
30 ∗ 30	42.3553	39.8410	5.9	13	11	15.3
35 ∗ 35	51.1838	47.5159	7.2	18	11	39.0

For the 30 ∗ 30 and the 35 ∗ 35 maps, Table 4 shows that the smoothed path length decreases by 5.9% and 7.2%, respectively; the number of turns for these maps decreases by 15.3% and 39.0%, respectively. This shows that the method can effectively reduce the path length and the number of turns in the AGV driving path.

The research of this paper is based on a general practical warehouse that is simplified to a two-dimensional grid map model. Therefore, the algorithm in this paper is universal in the warehouse AGV path planning domain. This path planning method can be applied to a variety of different types and sizes of warehouses, which can effectively improve the efficiency of warehouse operation.

5. Conclusion

Aimed at the problem of AGV path planning in automated warehouses, a P-RACO for path planning is proposed. The method of reducing invalid intermediate nodes is used to smooth the generated path. Firstly, the TSP problem is used to test the performance of the algorithm, and then the algorithm is verified on the two-dimensional grid warehouse map model. The following conclusions are drawn:

(1) When the P-RACO is used to solve the path planning multiobjective problem of the warehouse model, the path obtained can integrate two objectives: the shortest path and the least number of turns. Compared with approaches that only consider the shortest path, the P-RACO algorithm improves the efficiency and reduces the number of turns.

(2) Compared with the ACO, RACO, and IAC algorithms, the P-RACO algorithm proposed in this paper accumulates more pheromones in the early stage of solving the path planning problems under the TSP and for warehouses with different complexities. The P-RACO algorithm has better directionality, which can avoid the algorithm falling into a local optimum; from the standard deviation of many experiments, the proposed algorithm has better stability.

(3) By analyzing the standard deviation of the first 100 iterations when the P-RACO algorithm solves the TSP, it can be seen that the solutions of each generation of the algorithm have better diversity and the algorithm does not appear to stagnate.

(4) In this paper, the initial path is smoothed by reducing the intermediate nodes. This method significantly reduces the number of turns and the length of path in AGV driving. Consequently, it improves the overall operational efficiency of the warehouse and the service life of the AGVs.

In this paper, we mainly consider the operation of a single AGV in a warehouse. The dynamic collision avoidance problem resulting from multiple AGVs working simultaneously in an environment is an interesting and valuable direction for future research.

Data Availability

The data used to support the findings of this study are included in the supplementary information file.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This paper was supported by the Key R&D Projects in Shaanxi Province (2017ZDCXL-SF-03-02); National Key Research and Development Project of China, entitled "New Generation Intelligent Building Platform Techniques" (No. 2017YFC0704100); Shaanxi Provincial Science and Technology Department Special Research Project (2017JM6106); and Brain Research Fund of the University (JC1706).

Supplementary Materials

The data used to support the findings of this study are included in the supplementary information file. (*Supplementary Materials*)

References

[1] J. Liang, *Research on Path Planning and Scheduling Algorithm of AGV system*, Beijing University of Posts and Telecommunications, Beijing, China, 2018.

[2] N.K. Kang, H.J. Son, S.-H. Lee, Modified A-star algorithm for modular plant land transportation," *Journal of Mechanical Science and Technology*, vol. 32, no. 12, pp. 5563–5571, 2018.

[3] Le Yang and J. Gong, "An efficient implementation of dijkstra shortest path algorithms," *Journal of Wuhan Technical University of Surveying and Mapping*, vol. 24, no. 3, pp. 209–212, 1999.

[4] S. Wang and A. Li, "Multi-adjacent-vertexes and multi-shortest-paths problem of dijkatra algorithm," *Computer Science*, vol. 41, no. 6, pp. 217–224, 2014.

[5] X. Chen, X. Fang, and Q. Wu, "A fast two-stage ACO algorithm for robotic path planning," *Neural Computing and Applications*, vol. 22, no. 2, pp. 313–319, 2013.

[6] H. Yang, J. Qi, Y. Miao et al., "A new robot navigation algorithm based on a double-layer ant algorithm and trajectory optimization," *IEEE Transactions on Industrial Electronics*, p. 99, 2018.

[7] Q. He, Y. Wu, and T. Xu, "Application of improved genetic simulated annealing algorithm in TSP optimization," *Control and Decision*, vol. 33, no. 2, pp. 219–225, 2018.

[8] Q. Jia, X. Zhang, Y. Yuan et al., "Guided trajectory planning method for tractor autopilot system," *Transactions of The Chinese Society of Agricultural Machinery*, vol. 49, no. 4, pp. 36–44, 2018.

[9] M. S. Couceiro, J. A. Tenreiro Machado, R. P. Rocha, and N. M. F. Ferreira, "A fuzzified systematic adjustment of the robotic Darwinian PSO," *Robotics and Autonomous Systems*, vol. 60, no. 12, pp. 1625–1639, 2012.

[10] L. Kong, P. Li, and Q. Du, "The closed-loop Control system design of hexapod robot autonomous navigation based on Fuzzy neural network," *ROBOT*, vol. 40, no. 1, pp. 16–23, 2018.

[11] M. Dorigo and L. M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53–66, 1997.

[12] D. Jin, B. Yang, J. Liu, D.-Y. Liu, and D.-X. He, "Ant colony optimization based on random walk for community detection in complex networks," *Journal of Software*, vol. 23, no. 3, pp. 451–464, 2012.

[13] T. Stützle and H. H. Hoos, "Hoos ant system," *Future Generation Computer Systems*, vol. 16, no. 8, pp. 889–914, 2000.

[14] L. Liu, G. Yuan, and Y. Dai, "Multi-ant colony virtual parallel optimization algorithm," *Computer Engineering*, vol. 33, no. 23, pp. 199–201, 2007.

[15] J. Liu, J. Yang, H. Liu, X. Tian, and M. Gao, "An improved ant colony algorithm for robot path planning," *Soft Computing*, vol. 21, no. 19, pp. 5829–5839, 2017.

[16] X. Wang, L. Yang, Y. Zhang et al., "Robot path planning based on improved ant colony algorithm with potential field heuristic," *Control and Decision*, vol. 33, no. 10, pp. 50–56, 2018.

[17] Y. Zhang, F. Wang, F. Fu, and Z. Su, "Multi-AGV path planning for indoor factory by using prioritized planning and improved ant algorithm," *Journal of Engineering and Technological Sciences*, vol. 50, no. 4, pp. 534–547, 2018.

[18] D. Wang and H. Yu, "Path planning of mobile robot in dynamic environments," in *Proceedings of the Intelligent Control and Information Processing (ICICIP) 2011 2nd International Conference*, IEEE, Harbin, China, 2011.

[19] B. Sun, P. Jiang, G. Zhou et al., "Application of improved genetic algorithm in path planning of mobile robots," *Computer Engineering and Applications*, vol. 55, no. 17, pp. 162–168, 2019.

[20] H. Chen, J. Fei, Y. Liu et al., "Smooth path planning method based on dynamic feedback A $*$ ant colony algorithm," *Transactions of The Chinese Society of Agricultural Machinery*, vol. 48, no. 4, pp. 34–40, 2017.