Robot Path Planning Model Based on Improved A* Algorithm

Jing Xie*, Chunyuan Xu, Qianxi Yang
School of Mechanical and Electrical Engineering, Nanyang Vocational College of Agriculture, Nanyang 473003, China

***Abstract*—Robot path planning is a key technology for achieving autonomous navigation and efficient operation of robots. In order to improve the autonomous navigation capability of mobile robots, a global path planning model based on an improved A* algorithm and a local path planning model based on an improved artificial potential field method were designed. The results showed that the turns in the optimal path under the improved A* algorithm were 8, 5, 9, and 5, respectively. The improved artificial potential field method achieved a maximum planning time of 0.17s and a minimum planning time of 0.11s. The designed global and local path planning models for mobile robots have good performance and can provide technical support for improving the autonomous navigation capability of mobile robots for industrial manufacturing.**

***Keywords*—Robot; path; planning; A* algorithm; artificial potential field method; SA**

I. Introduction

With the development of information technologies and the concerned policies, China's industrial sectors are transitioning towards informatization and intelligence. Mobile robots play an important role in intelligent workshops, effectively enhancing industrial production efficiency and reducing production costs. Navigation, dynamic obstacle avoidance, and localization are key technologies in mobile robotics, with path planning being a major focus within the navigation technologies [1-2]. Global planning and local path planning are two categories. Common methods for the global path planning include the A* algorithm, Dijkstra's algorithm, and Floyd's algorithm. However, these algorithms have some shortcomings, such as the long search time of the A* algorithm and the lack of path smoothness in the generated paths [3]. With the development of intelligent biomimetic algorithms, more researchers have applied these methods to global path planning and made improvements to address specific shortcomings. Common methods at present include the artificial potential field (APF) and dynamic window approach for the local path planning. However, the former is prone to local minimum, the dynamic window approach heavily relies on weight coefficients. The rapid exploration random tree (RRT) exhibits randomness in node expansion, which may lead to path failure [4-5]. Therefore, the research question is how to improve the existing A* algorithm and APF method to enhance the global and local path planning performance of mobile robots and strengthen their autonomous navigation capabilities. To enhance the autonomous navigation capability of mobile robots, research has been conducted from two perspectives: global and local path planning. As a result, a variety of path planning methods have been designed. An enhanced version of the A* algorithm has been developed for global path planning purposes. In the context of local path planning, an enhanced APF method was devised. It can be seen that the purpose of the research is to design and optimize global and local path planning models for mobile robots, and to improve the adaptability and stability of the algorithm. The objective of this research is to enhance the autonomous navigation capability of mobile robots, improve their operational efficiency, and reduce industrial production costs. The contribution of the research is the enhancement of the A* algorithm and APF method in path planning. This is achieved by improving their performance in global or local path rules, reducing the number of optimal path turns, and lowering the time required for path planning. The research has two main innovations. First, it is the improvement of the repulsive potential field (RPF) in the APF. Second, simulated annealing algorithm (SA) and Doppler cooling strategy are introduced in the APF. The novelty of the research is reflected in the improvement of the heuristic function of the traditional A* algorithm through Manhattan distance and angle-based breaking of the "tie" strategy, making the algorithm more cost-effective and effectively reducing the number of turns on the optimal path. Meanwhile, the APF has been optimized through the enhancement of RPF, SA, and Doppler cooling strategy. This has led to the successful resolution of the local minimum problem and the enhancement of adaptability and stability in path planning. The significance of this research is that the results will help to improve the autonomous navigation ability of mobile robots in industrial manufacturing and other fields. It can reduce the production cost, improve the production efficiency, and provide technical support for intelligent workshop and automated production. In addition, the improved algorithms and strategies proposed in the study also provide new ideas and methods for research in the field of path planning. The research is further divided into five sections. The second section provides an overview of relevant literature on mobile robot path planning. The third section presents the specific design of the global planning and the local path planning models. The fourth section validates and analyzes the experimental results of the global and local path planning models. The fifth section presents a discussion of the research, which combines a literature analysis with a review of previous studies to provide a more comprehensive account of the comparative analysis results and details. The sixth section concludes the research, highlights shortcomings, and provides future perspectives.

II. Related Works

With the advancement of industrial manufacturing, mobile robots are increasingly used in intelligent industrial workshops and play an important role. Many researchers have conducted studies on path planning for mobile robots. Hossain et al.

*Corresponding Author.

designed a local algorithm combined with the follow-the-gap method to improve the obstacle avoidance function of mobile robots. The algorithm successfully generated collision-free trajectories and demonstrated good performance without encountering a local minimum [6]. Wang et al. addressed the autonomous navigation of unmanned aerial vehicles by proposing fixed charge set and discrete charge set problems. The fixed charge set problem was solved by using a two-stage traveling salesman problem method. Graph transformation techniques were used to handle the discrete charge set. Experimental results showed that both the fixed charge set and discrete charge set problems enabled unmanned aerial vehicles to operate continuously [7]. Hu et al. developed a motion planning framework for wheeled robots that incorporated the RRT algorithm. The study also introduced a path deformation strategy and posture-based motion control laws. Experimental results showed that the framework had computational advantages and generated smoother and shorter trajectories for wheeled robots [8]. Liu et al. addressed the complete coverage problem in hilly regions by proposing a path planning algorithm based on an energy consumption model. The SA solution was employed for traversing the optimal sequences for the fields. Moreover, a functional relationship between the driving angle and the energy consumption was established in the turning area. Experimental results indicated that considering energy consumption in the turning area reduced the minimum energy consumption [9].

Wang X et al. proposed a guided fast-exploring random tree algorithm and an improved discrete adaptive differential evolution algorithm to obtain the shortest collision-free path for arc welding robots. In addition, the welding environment of arc welding robots was modeled using packing and lattice methods. Experimental results showed that the invented algorithm optimized the paths of arc welding robots with good performance [10]. Liu A et al. presented a pigeon-inspired optimization algorithm improved by the logistic chaotic beetle algorithm for the path planning of mobile robots. This method reduced the iterations and search time, which optimized the path evaluation function. Experimental results demonstrated that the proposed improved pigeon-inspired optimization algorithm quickly found the global optimum solution and generated smoother paths [11]. Sun Y et al. developed two B-spline-based fast-searching random tree methods to generate collision-free trajectories in cluttered environments. The first method introduced dynamic feasible regions and designed two guiding functions. The second method guided the rapid growth of the tree in the first method through a fast marching path. Experimental results showed that the proposed algorithm had good performance and effectiveness [12]. Zhang Z et al. proposed an improved hybrid A* algorithm for path planning of spherical mobile robots. This study also designed a feasible and reachable path method that satisfied kinematic constraints and introduced the optimal minimum rotation region for robots. Experimental results demonstrated that the proposed method had good performance in path planning for spherical mobile robots and improved search efficiency to some extent [13].

In summary, there have been studies on mobile robot path planning, and different algorithms have been discussed. However, commonly used global path planning algorithms such as the A* algorithm, Dijkstra's algorithm, and Floyd's algorithm have their own drawbacks. The A* algorithm suffers from a long search time and produces less smooth paths, the Dijkstra algorithm is less efficient, and the Floyd algorithm is not suitable for computationally large graphs. Therefore, this research will design methods from global and local path planning to form a complete path planning method for mobile robots. Firstly, a global path planning model with the improved A* algorithm is designed for global path planning. Secondly, in terms of local path planning, a local path planning model based on an improved APF method is designed to enhance the autonomous navigation capability of mobile robots.

III. Mobile Robot Global and Local Path Planning Model Design

Models are designed separately for the global path planning and the local path planning for mobile robots. In the global path planning, an improved A* algorithm is used, with modifications made to the heuristic function. The improvement of heuristic functions is mainly reflected in two aspects. Firstly, the existing heuristic function is optimized by using the minimum difference between the Manhattan distance and the horizontal and vertical coordinates of the current node and the next node. Second, the angle-based strategy of breaking the "tie" is adopted to further optimize the heuristic function based on the first improvement. This is somewhat different from previous work, which mainly improved the A* algorithm through dynamic heuristic weight adjustment, incremental reprogramming, predictive obstacle avoidance, and redundant node removal. In the local path planning, an improved APF is employed to address the local minimum for both single and multiple obstacles. First, the study improves RPF in APF. Then the improved RPF exponentially decays with distance within the range of obstacle influence and maintains the continuity of the function outside the range of obstacle influence. Afterwards, SA is introduced to solve the local minimum. Meanwhile, an improved APF method and SA are combined to form the final hybrid algorithm. This approach diverges from previous works, which primarily enhanced the APF algorithm through the following methods: dynamic repulsion function design, hybrid algorithm architecture design, enhancement of dynamic environment adaptability, and implementation of path smoothing and optimization.

A. Design of Global Path Planning Model Hiring Improved A Algorithm for Mobile Robot*

Models for both global path planning and local path planning are developed to improve the autonomous navigation capability of mobile robots. First, the research designs a global path planning method for mobile robots. The A* algorithm is a common and widely used method in global path planning, which is known for its good search accuracy and performance in efficiently planning the global optimal path. However, the A* algorithm has certain limitations, such as longer search time and unbalanced optimal paths. Therefore, an improved A* algorithm is adopted for designing the global path planning model of mobile robots, focusing on modifications to the heuristic function and the strategy to break ties. In the global path planning, the environment map is static and known. The traditional A* algorithm is one of the commonly used methods for solving global path planning problems, known for its good

performance. The traditional A* algorithm utilizes heuristic search techniques. The choice of the evaluation function in this search technique affects the efficiency of the algorithm. The evaluation function used in the traditional A* algorithm is expressed as Eq. (1) [14-15].

$$f(x)=g(x)+h(x) \tag{1}$$

In Eq. (1), x represents the current node. $g(x)$ represents the cost already incurred from the start node s to x. $h(x)$ is the estimated cost from x to the target node t, which is a heuristic function. The value of $h(x)$ affects the computational efficiency of the A* algorithm. Therefore, this value should be as close as possible to the actual cost from x to t when designing the heuristic function. The Manhattan distance and Euclidean distance are the most common heuristic functions in the A* algorithm, with the calculation of the Manhattan distance shown in Eq. (2) [16].

$$d_M=|x_2-x_1|+|y_2-y_1| \tag{2}$$

In Eq. (2), (x_1,y_1) and (x_2,y_2) represent the coordinates of different nodes. The calculation of the Euclidean distance is shown in Eq. (3) [17].

$$d_{ogld}=\sqrt{(x_2-x_1)^2+(y_2-y_1)^2} \tag{3}$$

Generally, the Euclidean distance is more effective. However, the Manhattan distance may be more appropriate in certain scenarios, such as warehouse environment, where movement is limited to north-south or east-west directions. The original A* algorithm is shown in Fig. 1.

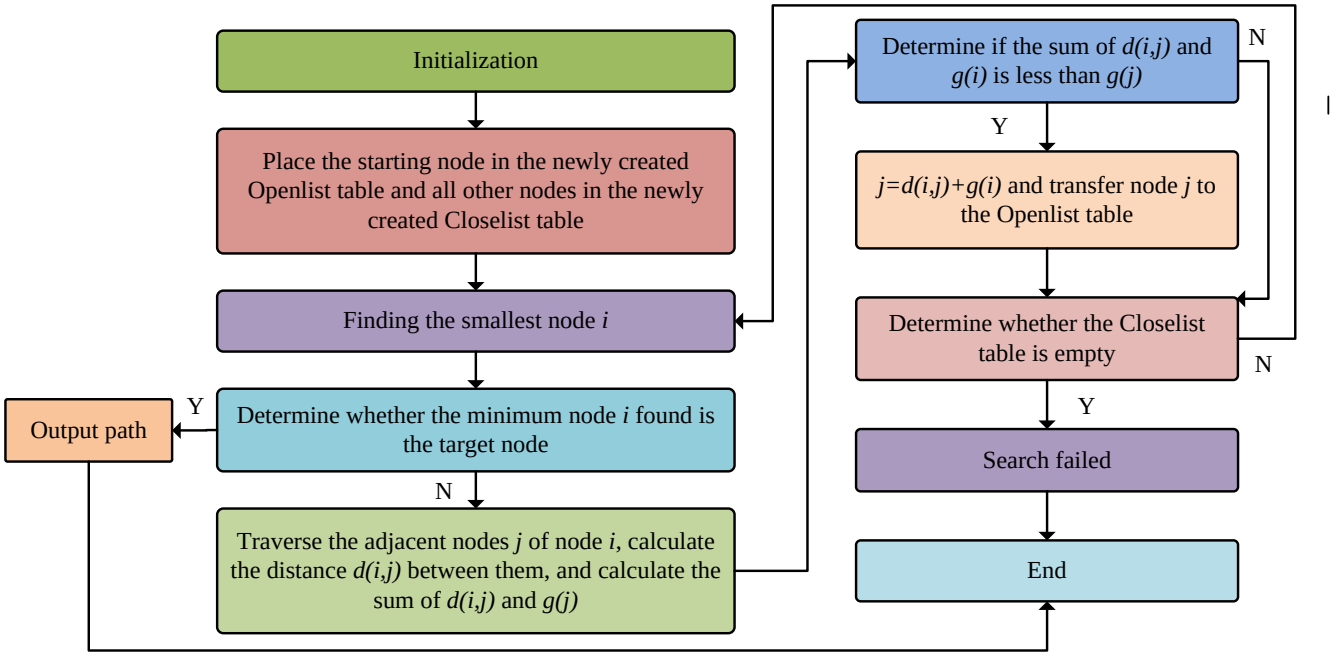


Fig. 1. The Operational process of traditional A* algorithm.

In Fig. 1, the first step of the traditional A* algorithm is to initialize the environment map and input the start and target nodes. The second step is to place the start node in a newly created Openlist and all other nodes in a newly created Closelist. The third step is to find the smallest value node. The fourth step is to check if the found node i is the target node. If i is the target node, the search is successful and the path needs to be outputted. The process ends. If i is not the target node, the next step is carried out. The fifth step is to iterate through the neighboring nodes of the smallest node j, calculate the distance $d(i,j)$ between the nodes, and calculate the sum of $d(i,j)$ and $g(i)$. $g(i)$ represents the cost incurred from the start node s to i. The sixth step is to check if the sum of $d(i,j)$ and $g(i)$ is less than $g(j)$. $g(j)$ represents the cost incurred from the start node s to j. If the sum of $d(i,j)$ and $g(i)$ is less than $g(j)$, j is assigned the sum of $d(i,j)$ and $g(i)$, and the node j is moved to the Openlist. Otherwise, the next step is carried out. The seventh step is to check if the Closelist is empty. If the Closelist is empty, it means the search has failed and the process ends. Otherwise, the process goes back to the third step. The relationship between the Manhattan distance and Euclidean distance used in the traditional A* algorithm and the actual cost is shown in Fig. 2.

In Fig. 2, there is a certain gap between both the Manhattan distance and the Euclidean distance and the actual cost, especially for the Manhattan distance, which has the largest gap. Therefore, the heuristic function of the traditional A* algorithm is improved to ensure that the heuristic function is near the actual cost, which is displayed in Eq. (4).

$$h'(x)=\sqrt{2}h_d(x)+\left(h_M(x)-2h_d(x)\right) \tag{4}$$

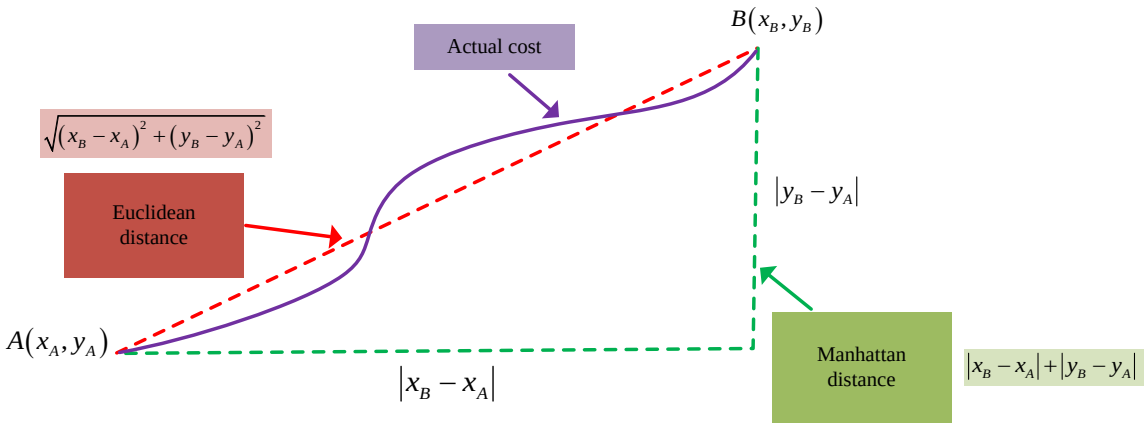


Fig. 2. The relationship between Manhattan distance and Euclidean distance and actual costs.

In Eq. (4), $h_M(x)$ represents the Manhattan distance. $h_d(x) = \min(|x_B - x_A|, |y_B - y_A|)$. (x_A, y_A) represents the coordinate of the current node. (x_B, y_B) represents the coordinate of the next node. The randomness of the traditional A* algorithm may lead to a failure in finding the optimal path when dealing with the "draw" situation. The specific schematic of the "draw" situation is shown in Fig. 3.

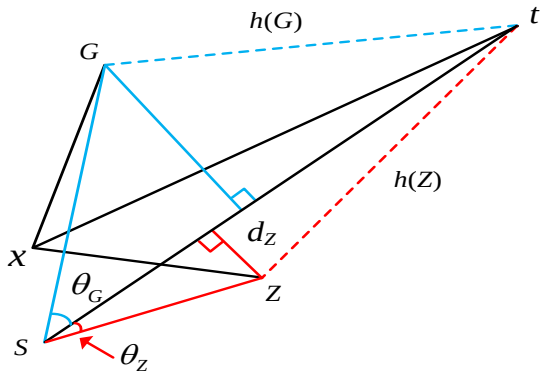


Fig. 3. Specific schematic of the 'draw' situation.

From Fig. 3, st represents the line between the start node s and the target node t. Nodes G and Z are the expansion nodes of the current node x. Their distances to the target node t are the same, i.e. $h(G) = h(Z)$. d_G and d_Z represent the vertical distances from nodes G and Z to st. θ_G and θ_Z represent the angles between nodes G and Z and the start node s. Therefore, a strategy is designed to deal with the "draw" situation. This strategy is to choose the expansion node with the shortest vertical distance between the line connecting the start node s and the target node t. The heuristic function is further improved using this strategy. The further improved heuristic function is shown in Eq. (5).

$$h''(x) = \alpha\theta h'(x) \tag{5}$$

In Eq. (5), α represents the weight coefficient, which is within [0, 1]. θ represents the angle between the line connecting the start and target nodes and the line connecting the expansion and start nodes. $\theta \in (0, 90^\circ)$. The evaluation function of the improved A* algorithm is shown in Eq. (6).

$$f(x) = g(x) + h''(x) = g(x) + \alpha\theta h'(x) \tag{6}$$

B. Design of Mobile Robot Local Path Planning Model with Improved APF

The research designs the local path planning method for mobile robots after designing the global path planning method to form the complete path planning method for mobile robots. Local path planning plays a crucial role in obstacle avoidance and autonomous movement in the autonomous navigation of mobile robots. Local path planning involves avoiding sudden obstacles in the optimal path obtained from global path planning. Therefore, the environment faced in local path planning is dynamic and unknown [18]. The APF, a commonly used approach, is chosen when designing the local path planning model for mobile robots. However, the APF method tends to get stuck in the local minimum [19]. Solutions are developed for both single and multiple obstacles to address this issue. The basic idea of APF is to create potential fields at the obstacles and at the target location. Meanwhile, obstacles are controlled and avoided by using these potential fields. Attractive forces are generated by the attractive potential field at the target point in the generated potential fields. Repulsive forces are generated by the RPF at the obstacles. Obstacle avoidance for the robot is achieved by the combined effect of these forces. The attractive potential field is generally set at the target point, and its magnitude is calculated using Eq. (7) [20].

$$U_{att} = \frac{1}{2} K_{att} \|X - T\|^2 \tag{7}$$

In Eq. (7), K_{att} represents the attractive gain coefficient, $T(x_t, y_t)$ is the target site coordinate, U_{att} represents the attractive potential field exerted on the mobile robot, and $X(x, y)$ represents the coordinate of the mobile robot. The force exerted on the mobile robot due to the attractive potential field is calculated using Eq. (8).

$$F_{att} = -K_{att} \|X - T\| \tag{8}$$

The force F_{att} can be decomposed into the coordinate axes. F_{att} can be transformed into a vector to improve computational efficiency. The representations of F_{att} along the x-axis and y-axis are given in Eq. (9).

$$\begin{cases} F_{attx} = F_{att}\cos\partial \\ F_{atty} = F_{att}\sin\partial \end{cases} \tag{9}$$

In Eq. (9), F_{attx} represents the component of F_{att} along the x-axis. ∂ is the angle between the vector F_{att} and the positive direction of the x-axis. F_{atty} represents the component of F_{att} along the y-axis. The RPF is primarily set at the obstacles. Once the mobile robot enters the influence area of an obstacle, it will be affected by the repulsive force. The calculation of the magnitude of the RPF is shown in Eq. (10) [21].

$$U_{rep} = \begin{cases} \frac{1}{2}K_{rep}\left(\frac{1}{\|O-X\|}\right)^2, \|O-X\| \le R \\ 0, \|O-X\| > R \end{cases} \tag{10}$$

In Eq. (10), K_{rep} represents the repulsion gain coefficient. $O(x_o, y_o)$ represents the coordinate of the obstacles. R represents the range of repulsion. $\|O-X\|$ represents the distance between the obstacle and the mobile robot. The force exerted on the mobile robot by the RPF is calculated as shown in Eq. (11).

$$F_{rep} = \begin{cases} K_{rep}\left(\frac{1}{\|O-X\|}-\frac{1}{R}\right)\frac{1}{\|O-X\|^2}, \|O-X\| \le R \\ 0, \|O-X\| > R \end{cases} \tag{11}$$

The force F_{rep} can be decomposed into the coordinate axes. F_{rep} can be converted into a vector to improve computational efficiency. The representations of the force on the x-axis and y-axis are shown in Eq. (12).

$$\begin{cases} F_{repx} = F_{rep}\cos\beta \\ F_{repy} = F_{rep}\sin\beta \end{cases} \tag{12}$$

In Eq. (12), F_{repx} represents the component of F_{rep} on the x-axis. β represents the angle between the vectors F_{rep} and the positive x-axis direction. F_{repy} represents the component of F_{rep} on the y-axis. The attractive potential field and multiple RPFs together form the superposed potential field, which is expressed in Eq. (13).

$$U = U_{att} + U_{rep} = U_{att} + \sum_{\varepsilon=1}^{N} U_{rep}^{\varepsilon} \tag{13}$$

In Eq. (13), N represents the number of obstacles, ε represents the ε th obstacle, and U_{rep}^{ε} represents the RPF of the ε th obstacle. The magnitude of the resultant force exerted on the mobile robot is calculated as shown in Eq. (14).

$$F = F_{att} + \sum_{\varepsilon=1}^{N} F_{rep} \tag{14}$$

Although the APF method produces relatively smooth paths, it is prone to local minimum [22]. Therefore, methods are developed to address the local minimum problems. Improvements are made to the RPF to address the local minimum for a single obstacle. Meanwhile, SA is applied to the method together with the Doppler cooling strategy. The improved RPF is expressed in Eq. (15).

$$U_{rep} = \begin{cases} \frac{1}{2}K_{rep}e^{-\|O-X\|^2}, \|O-X\| \le R \\ \frac{1}{2}K_{rep}e^{-R^2}, \|O-X\| > R \end{cases} \tag{15}$$

The core function of SA is to solve the local optimum values that occur during optimization. The SA uses the Metropolis criterion to avoid local minimum values. The main process of SA is shown in Fig. 4 [23].

In Fig. 4, the first step of SA is to initialize the parameters, including initial temperature, cooling rate, final temperature, and the iterations. The second step is to randomly generate an initial solution and compute the objective function. The third step is to generate a new solution and calculate the objective function of the new solution. The fourth step is to calculate the difference in objective function between the new solution and the initial solution. The fifth step is to determine whether the difference is less than zero. If the difference is less than zero, the new solution is accepted. Otherwise, SA proceeds to the next step. The sixth step is to determine whether the calculated probability is greater than or equal to a randomly generated number between 0 and 1. If the probability is greater than or equal to the random number, the new solution is accepted. Otherwise, SA returns to the third step. The seventh step is to lower the temperature. The eighth step is to determine if the termination conditions are met. If the termination conditions are met, the process terminates. If not, it returns to the third step. The Doppler cooling strategy is introduced to accelerate the convergence of SA. The combined algorithm of the improved APF method and SA is shown in Fig. 5.

From Fig. 5, the first step of the combined algorithm is to use the APF method to search for the path. The second step is to determine whether the algorithm falls into the local minimum. If the algorithm doesn't fall into the local minimum, it returns to the first step and continues the search until reaching the target point. Then the process ends. Otherwise, it proceeds to the next step. The third step is to use SA. The fourth step is to determine whether the algorithm escapes from the local minimum. If the algorithm escapes from the local minimum, it returns to the first step and continues searching until it finds the target point. Then the process ends. Otherwise, it returns to the third step. For the local minimum in the multiple obstacles, the combined algorithm of the improved APF method and SA may also encounter situations, where the target point cannot be reached. To solve this problem, a strategy of adding a virtual target point

is introduced. The main idea of this strategy is to use the attractive field of the virtual target point to help the mobile robot escape from the local minimum area of multiple obstacles [24]. The virtual target point strategy is shown in Fig. 6.

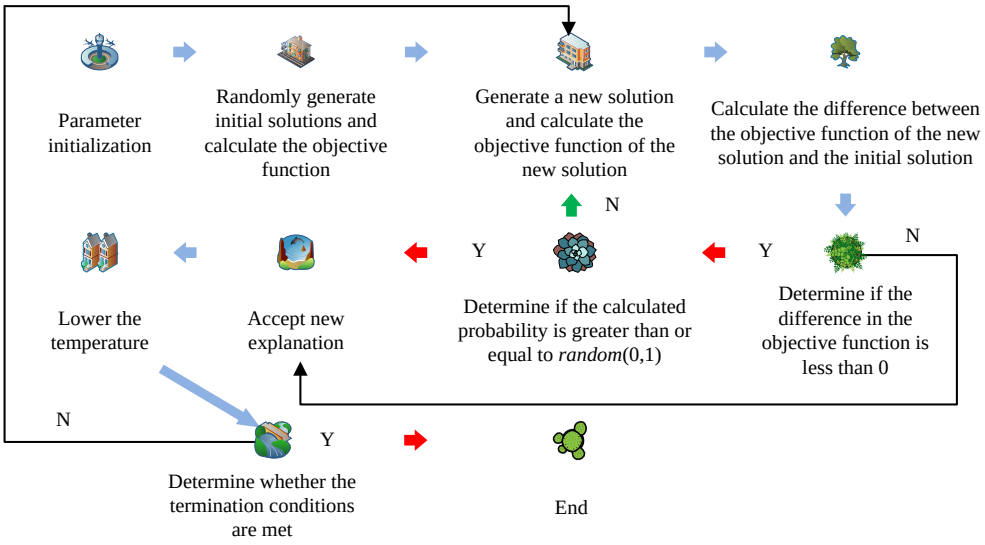


Fig. 4. The main process of SA.

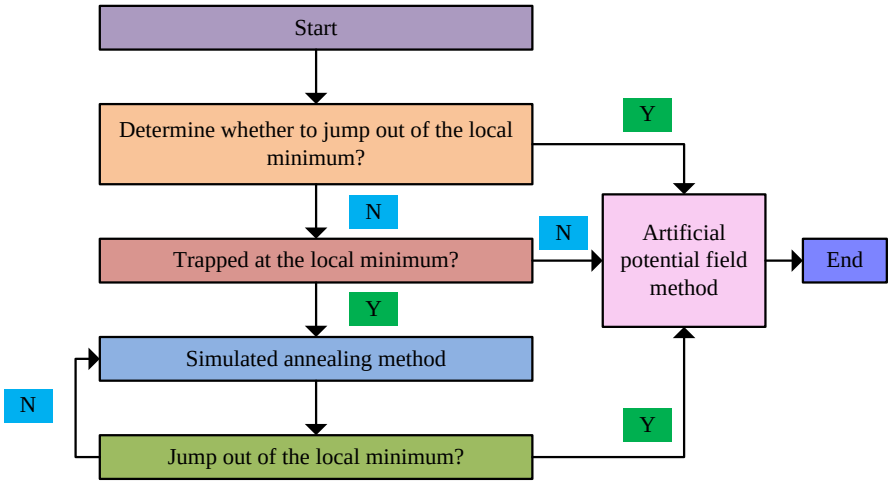


Fig. 5. The main process of combining the improved APF method and SA.

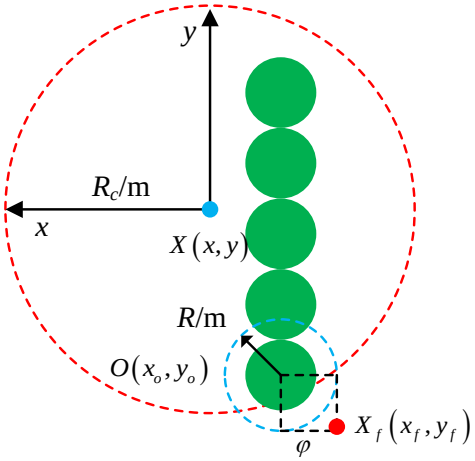


Fig. 6. Virtual target point strategy.

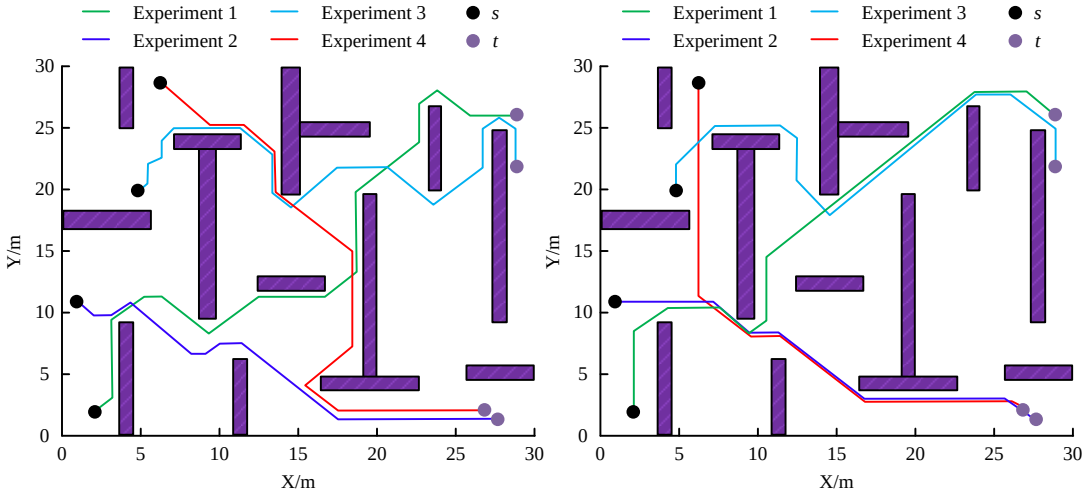
In Fig. 6, $X_f\left(x_f, y_f\right)$ represents the coordinate of the virtual target point. R_c represents the detection radius of the obstacles. φ is a constant, which needs to be set to ensure that the mobile robot can escape from the local minimum range of multiple obstacles. Once the mobile robot leaves the local minimum range of multiple obstacles, the attractive field of the virtual target point weakens. The mobile robot continues the search under the original potential field.

IV. Analysis of Global and Local Path Planning Results for Mobile Robots

In this section, the performance of the improved A* algorithm was verified in terms of path length, turns, and iterations. The performance of the combined algorithm of the improved APF method and SA was validated based on simulation results of the planning time and local minimum.

A. Analysis of Results for Global Path Planning based on the Improved A* Algorithm

The traditional A* algorithm was used to verify the performance of the improved A* algorithm. Simulation experiments were conducted using MATLAB R2019b software. The experiments were conducted on an Intel Core i5-11600K processor with 128GB memory, running on Windows 10 operating system. Four experiments were conducted, with different start and target nodes for each experiment. The start and target nodes for experiment 1 were (4, 4) and (29, 27), respectively. For experiment 2, they were (3, 12) and (29, 3). For experiment 3, they were (6, 21) and (29, 23). For experiment 4, they were (8, 29) and (28, 4). The evaluation metrics include path length, turns, and iterations, with a simulation map size of 30m*30m. The comparison of the turns for the optimal path between the pre-improved and post-improved A* algorithms under different experiments is shown in Fig. 7.



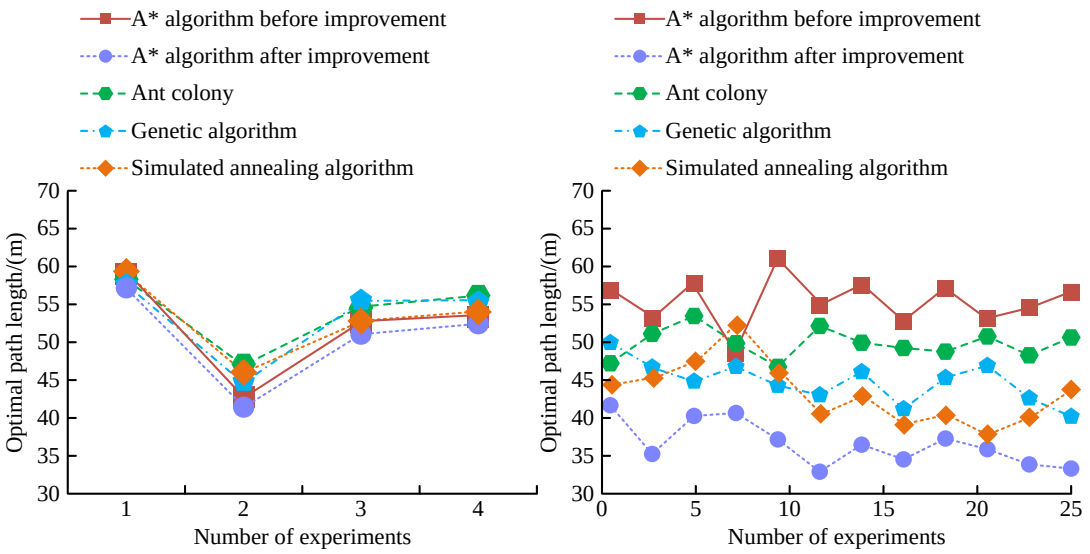
(a) The number of turns for the traditional A * algorithm

(b) The number of turns for the improved A * algorithm

Fig. 7. Comparison of the turns in the optimal path of A* algorithm before and after improvement in different experiments.

From Fig. 7(a), the turns in the optimal path for experiment 1 were 13 in the original A* algorithm, 8 for experiment 2, 16 for experiment 3, and 8 for experiment 4. Fig. 7(b) shows that the turns in the optimal path for experiments 1, 2, 3, and 4 were reduced to 8, 5, 9, and 5 after improving the A* algorithm, respectively. In experiments 1, 2, 3, and 4, the difference in the optimal number of turns for the A * algorithm before and after improvement was 5, 3, 7, and 3 times, respectively. The turns in the optimal path significantly decreased after improving the A* algorithm, indicating better performance compared to the original A* algorithm. Other path planning algorithms were also selected for comparison in the study to better validate the performance of the improved A* algorithm. Additional comparative algorithms include ant colony algorithm, genetic algorithm, and SA. In addition, the study also selected other path planning simulation maps for experimental verification, which were obtained from researchers such as Lai X [25]. The comparison of optimal path lengths for different algorithms on different simulation maps is shown in Fig. 8.

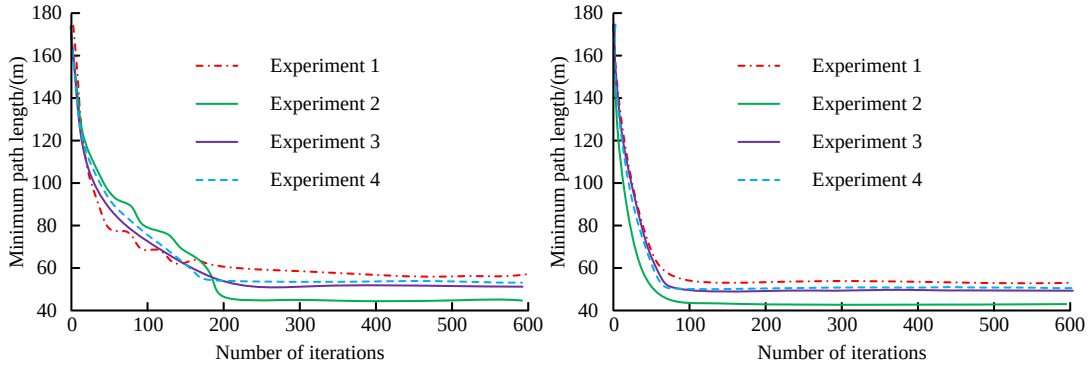
From Fig. 8(a), the maximum optimal path length for the pre-improved A* algorithm, ant colony algorithm, genetic algorithm, SA, and improved A* algorithm was 59.56m, 58.03m, 57.88m, 59.89m, and 56.39m, respectively, under four experiments, while the minimum value was 42.49m, 47.03m, 44.97m, 45.17m, and 40.83m, respectively. The maximum optimal path length of the A* algorithm, ant colony algorithm, genetic algorithm, and simulated SA before improvement was 3.17m, 1.64m, 1.49m, and 3.50m longer than that of the improved A* algorithm, respectively. According to Fig. 8(b), on the simulation map designed by Lai X et al., the maximum optimal path length for the five algorithms was 61.3m, 53.2m, 50.0m, 52.5m, and 41.9m, respectively. The maximum optimal path length of the improved A* algorithm was 19.4m, 11.3m, 8.1m, and 10.6m less than the maximum values of the other four algorithms, respectively. This also demonstrated that the performance of the improved A* algorithm was better. The convergence curves of the original and improved A* algorithms in the four experiments were compared, as shown in Fig. 9.



(a) The simulation map used in the paper

(b) On the simulation map designed by Lai X et al

Fig. 8. Comparison of the optimal path length planned by two algorithms under four experiments.



(a) Convergence curve before the improvement of A* algorithm

(b) Convergence curve after the improvement of A* algorithm

Fig. 9. Comparison of convergence curves of A* algorithm before and after improvement in four experiments.

From Fig. 9(a), the original A* algorithm required nearly 180 iterations to converge in experiment 1, and around 200, 210, and 190 iterations to converge in experiments 2, 3, and 4, respectively. In Fig. 9(b), the improved A* algorithm converged after approximately 95, 90, 100, and 85 iterations in experiments 1 to 4, respectively. In experiments 1, 2, 3, and 4, the number of iterations required for the enhanced A* algorithm to reach a convergence state was found to decrease by 85, 110, 110, and 105 times, respectively, in comparison to the original A* algorithm. The improved A* algorithm achieved faster convergence compared to the original A* algorithm, indicating its superior performance. The comparison of path planning time for different algorithms is shown in Table I.

TABLE I. COMPARISON OF PATH PLANNING TIME FOR DIFFERENT ALGORITHMS

Algorithm	**Runs**			
	10	***20***	***30***	***40***
Original A* algorithm	2.371s	2.397s	2.412s	2.435s
Ant colony	2.246s	2.269s	2.280s	2.297s
Genetic algorithm	2.017s	2.043s	2.067s	2.081s
Simulated annealing algorithm	1.943s	1.966s	1.987s	1.996s
Improved A* algorithm	1.732s	1.755s	1.773s	1.782s

From Table I, as the runs increased, the running time of all algorithms also increased synchronously. When the runs increased from 10 to 40, the maximum and minimum values of all algorithms were 2.435s and 1.732s, respectively, which appeared on the original A* algorithm and the improved A* algorithm. In addition, the running time of the improved A* algorithm was always lower than that of the compared algorithms. For example, when running 40 times, the improved A* algorithm had a running time that was 0.653s, 0.515s, 0.299s, and 0.214s lower than the other four algorithms, respectively. The improved A* algorithm took less time to plan the path and determined the optimal path more quickly. The study also placed three obstacles of different sizes on the simulation map, namely minor, moderate, and multiple obstacles. The configuration of obstacles at this time is shown in Table II.

In Table II, a limited number of obstacles were comprised of six rectangular obstacles of varying lengths, all with a width of 1.5 meters. Furthermore, the MATLAB code fragment intended to simulate map layout is displayed in Fig. 10.

As illustrated in Fig. 10, the MATLAB format code for simulating map layout comprised of five primary components: map size, obstacle configuration, map drawing, obstacle drawing, and adding networks and labels. The comparison of path planning results using different methods is shown in Fig. 11.

TABLE II. THE CONFIGURATION OF OBSTACLES

Obstacle configuration type	Number of obstacles	Obstacle shape	Obstacle size (width/m)
A small amount	6	Rectangle	1.5
Medium	10	Rectangle	1.5
More	16	Rectangle	1.5

```
% Map size
mapSize = [10, 10];

% Obstacle configuration
obstacleConfig = {
   'Minor', 6, 'rectangle', 1.5, [ (0,16), (0,17) ];
   'Moderate', 10, 'rectangle', 1.5, [ % Obstacle positions need to be specifically defined
   ];
   'More', 16, 'rectangle', 1.5, [ % Obstacle positions need to be specifically defined
   ];
};

% Create a new figure
figure;
axis([0, mapSize(2), 0, mapSize(1)]);
hold on;
set(gca, 'YDir', 'reverse');

% Draw obstacles
for i = 1:size(obstacleConfig, 1)
   obstacleType = obstacleConfig{i, 1};
   obstacleCount = obstacleConfig{i, 2};
   obstacleShape = obstacleConfig{i, 3};
   obstacleSize = obstacleConfig{i, 4};
   obstaclePositions = obstacleConfig{i, 5};

   for j = 1:obstacleCount
      position = obstaclePositions(j, :);
      if strcmp(obstacleShape, 'rectangle')
         rectangle('Position', [position(1), position(2), obstacleSize, obstacleSize], ...
            'Curvature', [0, 0], 'FaceColor', 'r', 'EdgeColor', 'k');
      end
   end
end

% Add grid and labels
grid on;
xlabel('X (m)');
ylabel('Y (m)');
title('Simulation Map');
hold off;
```

Fig. 10. The MATLAB format code snippet for simulating map layout.

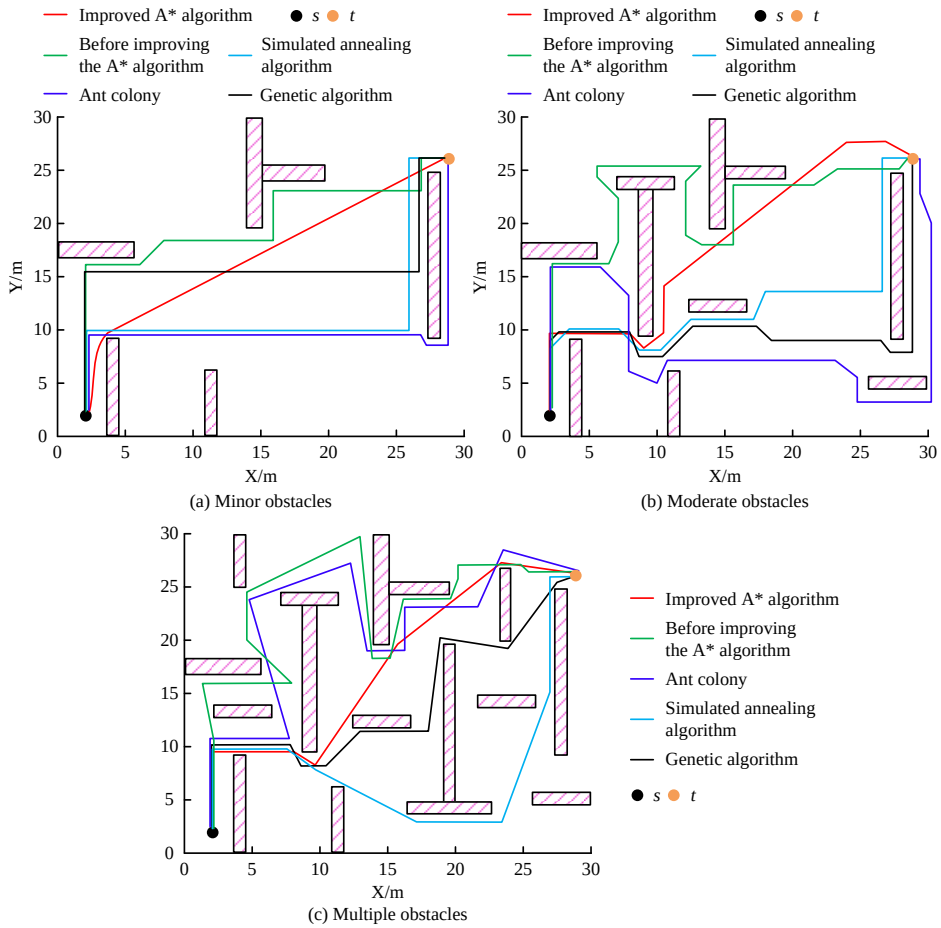


Fig. 11. Comparison of path planning results using different methods.

From Fig. 11(a), when the number of obstacles was small, the path smoothness of the improved A* algorithm, ant colony algorithm, genetic algorithm, SA, and improved A* algorithm was 0.532, 0.651, 0.702, 0.735, and 0.956, respectively. Moreover, the path smoothness of the enhanced A* algorithm was 0.424, 0.305, 0.254, and 0.221 greater than that of the original A* algorithm, the ant colony algorithm, the genetic algorithm, and the simulated SA, respectively. From Fig. 11(b), the improved A* algorithm planned shorter paths and had higher smoothness in moderate obstacle environments, with a value of 0.948, which was significantly better than the comparison algorithms. From Fig. 11(c), the path smoothness of the five algorithms was 0.498, 0.572, 0.698, 0.703, and 0.937, respectively, in environments with many obstacles. It can be concluded that when there were many obstacles, the path smoothness of the improved A* algorithm was 0.439, 0.365, 0.239, and 0.234 higher than the other four algorithms, respectively. In summary, the improved A* algorithm performed better. The comparison of central processing unit (CPU) utilization and memory usage of different algorithms under different numbers of obstacles is shown in Table III.

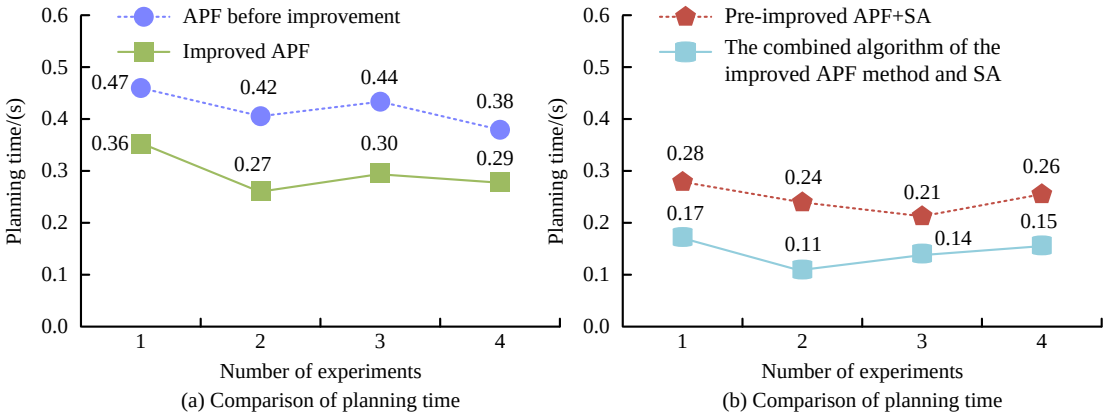
TABLE III. COMPARISON OF CPU UTILIZATION AND MEMORY USAGE OF DIFFERENT ALGORITHMS UNDER DIFFERENT NUMBERS OF OBSTACLES

Algorithm	CPU utilization/%			Memory usage/%		
	Scale of obstacles			Scale of obstacles		
	Minor	*Moderate*	*Multiple*	*Minor*	*Moderate*	*Multiple*
Before improving the A* algorithm	27.12	33.54	38.07	26.31	32.46	36.97
Ant colony	25.88	32.09	35.11	23.55	28.49	33.62
Genetic algorithm	23.35	25.73	30.71	21.82	26.61	30.04
Simulated annealing algorithm	21.09	23.86	29.58	19.51	22.33	26.97
Improved A* algorithm	12.01	15.46	18.73	13.96	16.23	18.02

In Table III, the CPU utilization rates of the improved A* algorithm were 12.01%, 15.46%, and 18.73%, respectively, under a small number of obstacles, moderate obstacles, and a large number of obstacles, which were significantly lower than the comparison algorithm. For example, under a small number of obstacles, the CPU utilization rates of the improved A* algorithm, ant colony algorithm, genetic algorithm, and simulated SA were 15.11%, 13.87%, 11.34%, and 9.08% higher than those of the improved A* algorithm, respectively. Meanwhile, under different numbers of obstacles, the memory consumption of the improved A* algorithm was significantly lower than that of the comparison algorithm, and the value remained below 20%. In summary, the improved A* algorithm had lower CPU and memory consumption, better performance, and more advantages in practical applications of path planning.

B. *Analysis of Local Path Planning Results with Improved APF*

Simulations were conducted using MATLAB R2019b software to verify the performance of the combined algorithm of the improved APF method and SA. The algorithms compared include the original APF, the improved APF, and the combined algorithm of the improved APF method and SA. The size of the simulation map was 10m*10m. The compared indicators were the planning time of the algorithm and the simulation results of the local minimum. The comparison of planning time for different algorithms is shown in Fig. 12.



(a) Comparison of planning time (b) Comparison of planning time

Fig. 12. Planning time of different algorithms.

In Fig. 12(a), the planning time of the original APF method ranged from 0.38s to 0.47s. The planning time of the pre-improved APF method ranged from 0.27s to 0.36s. The maximum and minimum planning times of the improved APF method were both 0.11s lower than before the improvement. In Fig. 12(b), the planning time for the combined algorithm of the pre-improved APF method and SA ranged from 0.21s to 0.28s. The planning time for the combined algorithm of the improved APF method and SA ranged from 0.11s to 0.17s. The combined algorithm of the improved APF method and SA had a significantly lower planning time compared to other algorithms, indicating better performance. This study selected APF, SA, dynamic window method (DWM), the RRT algorithm, and ant colony optimization with adaptive mechanism (ACOAM) for comparative verification to better verify the performance of the combined algorithm of the improved APF method and SA. Three types of obstacles of different scales were set up on the simulation map, namely minor, moderate, and multiple obstacles. The comparison of the number of path turns and average convergence times of different algorithms is shown in Table IV.

TABLE IV. COMPARISON OF PATH TURNING TIMES AND AVERAGE CONVERGENCE TIMES OF DIFFERENT ALGORITHMS

Algorithm	Path turning times			Average convergence times		
	Scale of obstacles			Scale of obstacles		
	Minor	*Moderate*	*Multiple*	*Minor*	*Moderate*	*Multiple*
SA	8	11	12	19.7	20.9	22.4
APF	9	10	13	19.5	21.6	23.4
DWM	10	13	14	21.5	22.9	26.1
RRT	9	12	13	20.7	22.3	25.8
ACOAM	7	9	11	17.9	18.8	20.3
Manuscript	6	8	10	14.7	15.1	17.3

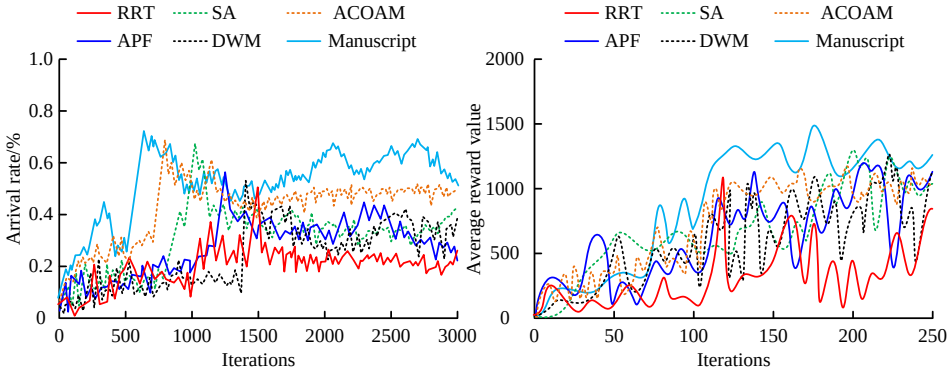
From Table IV, the path turning times and average convergence times of the designed algorithm were always smaller than those of the comparison algorithms under obstacles of different scales. The path turns for the designed algorithm were 6, 8, and 10, with an average convergence of 14.7, 15.1, and 17.3 under minor, moderate, and multiple obstacles, respectively. The performance of the ACOAM algorithm was closest to that of the designed algorithm, with 7, 9, and 11 path

transitions, and an average convergence rate of 17.9, 18.8, and 20.3, respectively. In summary, the designed algorithm had better performance and performed well when facing obstacles of different scales. The comparison of running time and path length of different algorithms under different obstacle scales is shown in Table V.

TABLE V. COMPARISON OF RUNTIME AND PATH LENGTH OF DIFFERENT ALGORITHMS UNDER DIFFERENT OBSTACLE SCALES

Algorithm	Running time/s			Path length/cm		
	Scale of obstacles			Scale of obstacles		
	Minor	*Moderate*	*Multiple*	*Minor*	*Moderate*	*Multiple*
SA	2.987	4.659	5.742	33.715	54.263	72.645
APF	3.012	4.583	6.776	36.854	57.312	75.791
DWM	3.227	5.317	7.462	38.213	59.621	76.373
RRT	3.452	5.472	7.550	38.336	60.082	77.591
ACOAM	2.632	4.447	5.576	32.176	52.537	71.998
Manuscript	1.941	3.294	4.733	30.386	50.558	68.168

From Table V, the minimum running time was 1.941s. The minimum path length was 30.386cm under minor obstacles. Both of them appeared in the combined algorithm. The running time of this combined algorithm under moderate and multiple obstacles was 3.294s and 4.733s, respectively. The path length was 50.558cm and 68.168cm. At different obstacle scales, the running time and path length of the combined algorithm were significantly lower than those of the comparison algorithms, indicating that the algorithm had strong path planning ability. The study compared the arrival rates and average rewards of different algorithms to further verify the performance of the combined algorithm of the improved APF method and SA. The comparison results are shown in Fig. 13.



(a) Comparison of arrival rates (b) Comparison of average rewards

Fig. 13. Comparison of arrival rates and average rewards of different algorithms.

Fig. 13(a) shows that the arrival rates of the different algorithms increased synchronously with the number of iterations. After more than 700 iterations, the combined algorithm of the improved APF method and SA showed a significant improvement in the arrival rate, which was significantly higher than the comparison algorithm. The maximum arrival rate of this hybrid algorithm was 72.3%, and the maximum arrival rates of the SA, APF, DWM, RRT, and ACOAM algorithms were 67.51%, 58.12%, 55.37%, 52.73%, and 69.34%, respectively. The research on building hybrid algorithms had a strong ability to approach the target point. In Fig. 13(b), with the increase of training times, the average rewards of different algorithms showed a synchronous increasing trend overall. Specifically, after 100 iterations, the average reward oscillation of the combined algorithm tended to stabilize at a smaller amplitude, which was significantly faster than the comparison algorithms. SA did not show oscillation stability in the limited training iterations, and the amplitude of the oscillation was relatively large. The combined algorithm was able to explore paths faster and more stably. To further verify the performance of the combined algorithm, the simulation results of the local minimum in the presence of a single obstacle for different algorithms were compared, as shown in Fig. 14.

From Fig. 14(a) and Fig. 14(b), both the original and the improved APF methods failed to reach the target node when facing a single obstacle and get trapped in the local minimum. In Fig. 14(c), the combined algorithm of the original APF method and SA reached the target node after iterating nearly 60 times to escape from the local minimum. In Fig. 14(d), the combined algorithm of the improved APF method and SA also reached the target node and escaped from the local minimum after only about 8 iterations. This demonstrated that the combined algorithm of the improved APF method and SA quickly escaped from the local minimum in the presence of a single obstacle, indicating better performance. The effectiveness of the strategy of adding a virtual target point was validated. Experiments were also conducted to escape the local minimum in the presence of multiple obstacles. The results are shown in Fig. 15.

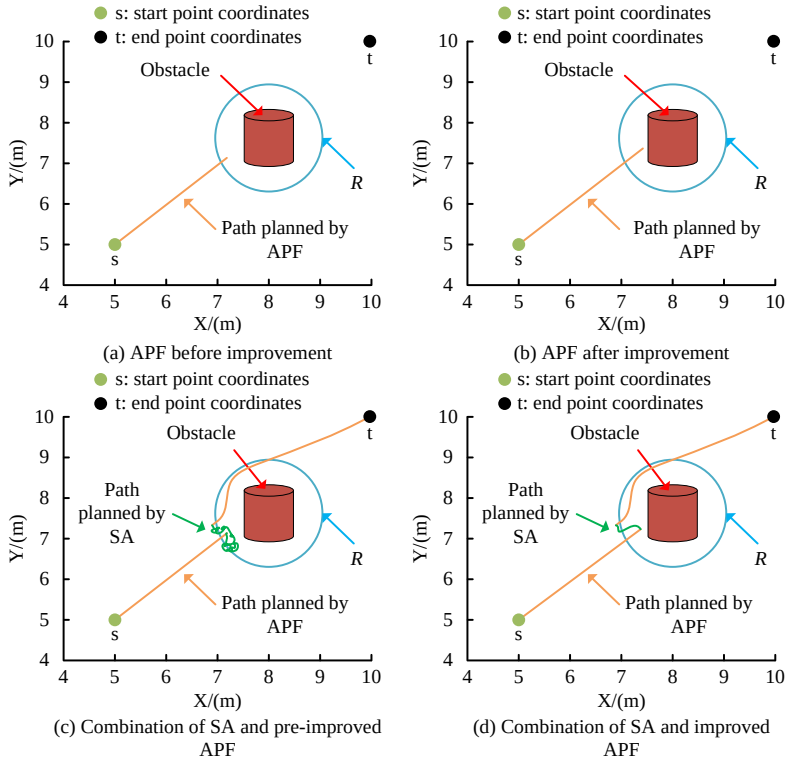


Fig. 14. Comparison of simulation results of different algorithms for local minimum values under a single obstacle.

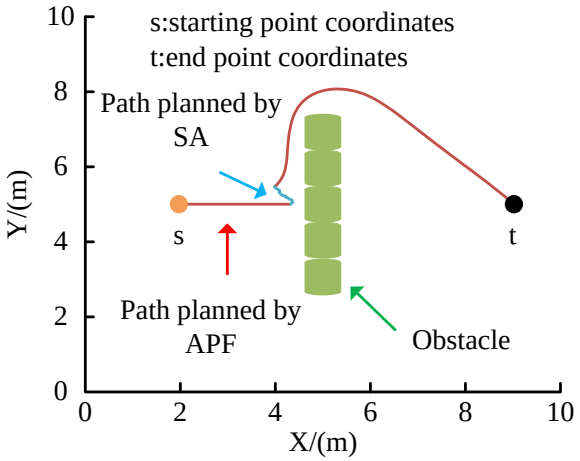


Fig. 15. Results of detachment from local minimum values under multiple obstacles.

In Fig. 15, the combined algorithm of the improved APF method and SA under multiple obstacles, with the introduction of the strategy of adding a virtual target point, successfully reached the target node. This method escaped from the local minimum under multiple obstacles after iterating for about 10 times. The strategy of adding a virtual target point helped the combined algorithm of the improved APF method and SA to escape from the local minimum under multiple obstacles, indicating the effectiveness of this strategy.

V. Discussion

To improve the autonomous navigation capability of mobile robots, an improved A* algorithm for global path planning and an improved APF method for local path planning have been studied and designed. The improvement of the A* algorithm in this study mainly started with the optimization of the heuristic function, which achieved a reduction in the optimal path length, and the shortest time in multiple experiments was 1.732 seconds. Qi S et al. improved the A* algorithm by introducing geomagnetic information entropy into the fitness function, achieving a 42.02% reduction in path length [26]. Xiang Y et al. enhanced the A* algorithm by developing a novel hybrid heuristic function based on Euclidean distance and projection distance, thereby optimizing the path length through the potential field function of the APF algorithm [27]. This study presented an improvement to the APF algorithm that optimized the RPF, introduced the simulated annealing method to facilitate escape from local minima, and combined the improved APF with the simulated annealing method to achieve a reduction in running time. Firdos I et al. developed an improved APF that combined Q-learning and combined dynamic and static reward functions, achieving a 67.25% improvement in path length [28]. The more comprehensive comparative analysis results and details of different studies are shown in Table VI.

In Table VI, the A* and APF algorithms for path planning problems have undergone significant advancements, resulting a reduction in optimal path length and enhanced obstacle avoidance efficacy. Meanwhile, there was still room for improvement in the reduction of path length in research. The comparison of path smoothness and path diversity across studies is shown in Table VII.

TABLE VI. MORE COMPREHENSIVE COMPARATIVE ANALYSIS RESULTS AND DETAILS OF DIFFERENT STUDIES

Number	Advantage	Disadvantage
[26]	The path length has been shortened by 42.02%, and the number of turns has been reduced by 92.31%	Difficulty in obtaining and processing geomagnetic data
[27]	Reduced the search nodes of the A * algorithm and improved the obstacle avoidance effect	There is a possibility of losing the optimal solution
[28]	A 67.25% path length improvement was achieved, with an average performance improvement of about 14.68%	There may be a conflict issue with reward signals
Manuscript	The maximum reduction in path length for the improved A * and APF algorithms is 5.32% and 10.06%, respectively.	Not considering dynamic obstacle avoidance yet

TABLE VII. COMPARISON OF PATH SMOOTHNESS AND DIVERSITY IN DIFFERENT STUDIES

Number	Path smoothness					Path diversity				
	Number of experiments					Number of experiments				
	1	*2*	*3*	*4*	*5*	*1*	*2*	*3*	*4*	*5*
[26]	0.851	0.899	0.894	0.860	0.891	0.856	0.894	0.875	0.861	0.889
[27]	0.890	0.881	0.890	0.868	0.862	0.885	0.897	0.879	0.906	0.866
[28]	0.902	0.887	0.919	0.892	0.894	0.916	0.887	0.884	0.887	0.897
Improved A*	0.982	0.974	0.933	0.961	0.952	0.973	0.953	0.944	0.943	0.953
Improved APF	0.985	0.957	0.962	0.954	0.971	0.948	0.972	0.987	0.957	0.970

The smoothness of a path was measured by standardization, with a value range of [0, 1], and the larger the value, the smoother the path. Path diversity was achieved by measuring the similarity of paths generated from multiple runs, with a value range of [0, 1], and the larger the value, the higher the diversity. As illustrated in Table VII, the mean path smoothness values reported in earlier studies [26], [27], and [28] were 0.879, 0.878, and 0.899, respectively. In this study, the average path smoothness values of improved A * and improved APF were 0.960 and 0.966, respectively, which were significantly better than previous studies. In addition, in terms of path diversity, the average values of the five methods were 0.875, 0.887, 0.894, 0.953, and 0.967, respectively. In summary, the designed algorithm was demonstrated to exhibit higher path smoothness and diversity, generate paths with reduced sharp turns and acceleration changes, and possess strong randomness and adaptability. These characteristics had the potential to enhance the probability of robots identifying feasible paths.

VI. CONCLUSION

A global path planning model based on the improved A* algorithm and a local path planning model based on the improved APF algorithm were developed to improve the autonomous navigation capability of mobile robots. The experimental results showed that the turns in the optimal path for the original A* algorithm in the four experiments were 13, 8, 16, and 8, respectively. For the improved A* algorithm, the turns in the optimal path were reduced to 8, 5, 9, and 5, respectively. There was a reduction of 5, 3, 7, and 3 turns compared to the original algorithm. The length of the optimal path for the original A* algorithm in the four experiments was 59.56m, 42.49m, 53.15m, and 53.32m, respectively. For the improved A* algorithm, the length of the optimal path was reduced to 56.39m, 40.83m, 50.56m, and 50.83m, respectively. There was a reduction of 3.17m, 1.66m, 2.59m, and 2.49m compared to the original algorithm. The performance of the improved A* algorithm was superior to the original A* algorithm. The maximum planning time for the original APF method was 0.47s. The maximum planning time for the improved APF method was 0.36s. The maximum planning time for the combined algorithm of the original APF method and SA was 0.28s. The maximum planning time for the combined algorithm of the improved APF method and SA was 0.17s. The combined algorithm of the improved APF method and SA had better performance. The combined algorithm of the improved APF method and SA reached the target node and avoided being trapped in the local minimum under the single obstacle. In addition, the strategy of adding a virtual target point can help the combined algorithm of the improved APF method and SA to escape from the local minimum under multiple obstacles. However, there are still limitations in this study. The designed local path planning model mainly focuses on static obstacle environments. Future research can explore the avoidance of dynamic obstacles that may appear unexpectedly. In addition, although the improved algorithm performs well in terms of path planning performance, there is still room for improvement in terms of computational resource consumption. Future research can reduce the computational cost of the algorithms and improve their real-time performance and applicability through algorithm optimization, hardware acceleration, and other methods. Finally, current path planning methods mainly focus on finding optimal paths. However, in some practical scenarios, it may be necessary to generate multiple feasible paths for selection. Future research could explore ways to enhance the algorithm's ability to generate path diversity to meet different task requirements. The research makes a significant contribution to the field by optimizing the performance of global and local path planning for mobile robots. This is achieved by improving the A* algorithm and APF method, thereby reducing the number of turns and path length of the optimal path. Additionally, the research reduces path planning time, improves the adaptability and stability of the algorithm, and provides efficient technical support for

autonomous navigation of robots in industrial manufacturing and other fields. Mobile robot path planning technology has a wide range of applications and is important in the field of industrial manufacturing, such as automatic material handling and component distribution in scenarios such as intelligent workshops and automated warehouses. However, in real-world implementation, the designed planning algorithm must be adapted to address issues such as dynamic environment adaptability, sensor accuracy and reliability, computational resource limitations, multi-robot collaboration, and real-time requirements. These are also issues that need to be addressed in practical applications. In the real-world, the global and local path planning models developed by the research can be directly applied to the autonomous navigation system of mobile robots. Specifically, the algorithm must first be integrated into the robot's embedded system, and real-time environmental data can be obtained through sensors, such as using LiDAR for obstacle detection and mapping. Then, by combining the kinematic model and the dynamic constraints of the robot, the algorithm is optimized and adapted to ensure the feasibility and real-time performance of the path planning. In terms of dynamic environment perception, deep learning techniques can be introduced. In addition, by combining it with hardware acceleration technology, the algorithm's computational resource consumption can be further reduced, which is expected to significantly improve its real-time performance and applicability in practical applications.

REFERENCE

[1] S. Ziadi and M. Njah, "PSO-DVSF~2-mt: An optimized mobile robot motion planning approach for tracking moving targets," Int. J. Robotics Autom., vol. 37, no. 5, pp. 421-430, January 2022.

[2] J. Wang, M. T. H. Fader, and J. A. Marshall, "Learning-based model predictive control for improved mobile robot path following using Gaussian processes and feedback linearization," J. Field Robotics, vol. 40, no. 5, pp. 1014-1033, February 2023.

[3] G. F. Chai and Y. Z. Xia, "Multi-robot path optimization and simulation for multi-route inspection in manufacturing," Int. J. Simul. Model., vol. 22, no. 1, pp. 121-132, March 2023.

[4] J. Li, J. Sun, L. Liu, and J. Xu, "Model predictive control for the tracking of autonomous mobile robot combined with a local path planning," Measure. Control, vol. 54, no. 9, pp. 1319-1325, October 2021.

[5] X. Li, G. Zhao, and B. Li, "Generating optimal path by level set approach for a mobile robot moving in static/dynamic environments," Appl. Math. Model., vol. 85, no.2, pp. 210-230, September 2020.

[6] T. Hossain, H. Habibullah, R. Islam, and R. V. Padilla, "Local path planning for autonomous mobile robots by integrating modified dynamic window approach and improved follow the gap method," J. Field Robotics, vol. 39, no.4, pp. 371-386, December 2021.

[7] Q. Wang, H. Chen, L. Qiao, J. Tian, and Y. Su, "Path planning for UAV/UGV collaborative systems in intelligent manufacturing," IET Intell. Transp. Syst., vol. 14, no. 2, pp. 1475-1483, May 2020.

[8] B. Hu, Z. Cao, and M. Zhou, "An efficient RRT-based framework for planning short and smooth wheeled robot motion under kinodynamic constraints," IEEE Trans. Ind. Electron., vol. 68, no. 4, pp. 3292-3302, April 2021.

[9] T. Liu, J. Li, S. X. Yang, Z. Gong, Z. L. Liu, and H. Zhong, "Optimal coverage path planning for tractors in hilly areas based on energy consumption model", Int. J. Robotics Autom., vol. 38, no. 1, pp. 20-31, 2023.

[10] X. Wang, Z. Xia, X. Zhou, J. Wei, X. Gu, and H. Yan, "Collision-free path planning for arc welding robot based on IDA-DE algorithm," Int. J. Robotics Autom., vol. 37, no. 6, pp. 476-485, 2022.

[11] A. Liu and J. Jiang, "Solving path planning problem based on logistic beetle algorithm search-pigeon-inspired optimisation algorithm," Electron. Lett., vol. 56, no. 21, pp. 1105-1108, September 2020.

[12] Y. Sun, C. Zhang, and C. Liu, "Collision-free and dynamically feasible trajectory planning for omnidirectional mobile robots using a novel B-spline based rapidly exploring random tree," Int. J. Advanced Robotic Syst., vol. 18, no. 3, pp. 473-493, June 2021.

[13] Z. Zhang, Y. Wan, Y. Wang, X. Guan, W. Ren, and G. Li, "Improved hybrid A* path planning method for spherical mobile robot based on pendulum," Int. J. Advanced Robotic Syst., vol. 18, no. 1, pp. 671-680, February 2021.

[14] S. Laaroussi, A. Baataoui, A. Halli, and K. Satori, "Dynamic mosaicking: combining A* algorithm with fractional Brownian motion for an optimal seamline detection," IET Image Process., vol. 14, no. 13, pp. 3169-3180, November 2020.

[15] A. M. Usman and M. K. Abdullah, "An assessment of building energy consumption characteristics using analytical energy and carbon footprint assessment model," Green Low-Carbon Econ., vol. 1, no. 1, pp. 28-40, March 2023.

[16] S. Kansal and R. Tripathi, "A new adaptive histogram equalization heuristic approach for contrast enhancement," IET Image Process., vol. 14, no. 6, pp. 1110-1119, 2020.

[17] J. Chen, S. J. Song, Y. Gu, and S. X. Zhang, "A multisensor fusion algorithm of indoor localization using derivative Euclidean distance and the weighted extended Kalman filter," Sens. Rev., vol. 42, no. 6, pp. 669-681, October 2022.

[18] J. Akshya and P. L. K. Priyadarsini, "Graph-based path planning for intelligent UAVs in area coverage applications," J. Intell. Fuzzy Syst., vol. 39, no.6, pp. 8191-8203, December 2020.

[19] W. Zhang, G. Xu, Y. Song, and Y. Wang, "An obstacle avoidance strategy for complex obstacles based on artificial potential field method," J. Field Robotics, vol. 40, no. 5, pp. 1231-1244, May 2023.

[20] A. A. Ansari and E. I. Abouelmagd, "Gravitational potential formulae between two bodies with finite dimensions," Astron. Nachr., vol. 341, no. 6, pp. 656-668, June 2020.

[21] W. Zhang, S. Wei, J. Zeng, and N. Wang, "Multi-UUV path planning based on improved artificial potential field method," Int. J. Robotics Autom., vol. 36, no. 4, pp. 231-239, 2021.

[22] A. A. Ibrahim and R. O. Abdulaziz, "Analysis of titanic disaster using machine learning algorithms," Eng. Let., vol. 28, no. 4, pp. 1161-1167, November 2020.

[23] M. Jimenez-Martinez and M. Alfaro-Ponce, "Fatigue life prediction of aluminum using artificial neural network," Eng. Let., vol. 29, no. 2, pp. 704-709, June 2021.

[24] M. F. Cifuentes-Molano, B. S. Hernandez, and E. Giraldo, "Comparison of different control techniques on a bipedal robot of 6 degrees of freedom," Iaeng. Int. J. Ap. Mat., vol. 51, no. 2, pp. 300-306, May 2021.

[25] X. Lai, D. Wu, D. Wu, J. H. Li, and H. Yu, "Enhanced DWA algorithm for local path planning of mobile robot," Ind. Robot., vol. 50, no. 1, pp. 186-194, August 2022.

[26] S. Qi, B. Qiang, and T. Yude, "Path planning of improved A * algorithm based on geomagnetic matching aided navigation," J. Jiangsu Univ. (Nat. Sci. Ed.)., vol. 44, no. 6, pp. 696-703, 2023.

[27] Y. Xiang, J. Chen, D. Sirui, and D. Qianrui, "Path planning for improvement of A* algorithm and artificial potential field method," J. Syst. Simul., vol. 36, no. 3, pp. 782-794, 2024.

[28] I. Firdos, R. Firas, and N. Ahmed, "Path planning improvement using a modified Q-learning algorithm based on artificial potential field," Int. J. Intell. Eng. Syst., vol. 17, no. 4, pp. 411-423, July 2024.