Journal of Network Intelligence ©2025 ISSN 2414-8105 (Online)
Taiwan Ubiquitous Information Volume **10**, Number **2**, May **2025**

A Novel Path Planning Algorithm Incorporating Improved A* Algorithm and Fitting Potential Field Method

You-Yuan Zhang, Li-Sang Liu*, Cheng-Yang Ke

School of Electronic Electrical Engineering and Physics
Fujian University of Technology, Fuzhou 350118, China
2221905038@smail.fjut.edu.cn, liulisang@fjut.edu.cn, 2211905005@smail.fjut.edu.cn

Bin Wang

Huadian International Shiliquan Power Plant, Zaozhuang 277100, China
2201905138@smail.fjut.edu.cn

Kai-Qi Guo

State Grid Fujian Electric Power Co., Ltd.
Putian Power Supply Company, Putian 351100, China
2211905052@smail.fjut.edu.cn

*Corresponding author: Li-Sang Liu

Received December 25, 2023, revised April 28, 2024, accepted December 13, 2024.

Abstract. *This paper presents an algorithm to enhance mobile robot path planning by improving speed, smoothness, and navigability. It integrates an improved A* algorithm and a fitted potential field method. Firstly, it constructs a contour grid map that extends the robot's workspace while obtaining a globally optimal path. Aiming at the problems of the traditional A* algorithm, the sub-node screening rule is redesigned so that the robot does not need to re-select the way after encountering an obstacle, thus improving the real-time performance. Secondly, the traditional potential field method is enhanced using interval point search, range judgment strategy, and adaptive iterative function. This reduces the number of local optimal and unreachable target points and solves the data redundancy problem. In addition, a least squares method is added to fit the paths, which improves the smoothness. Finally, the improved A* algorithm, the included potential field method, and the fusion algorithm are compared with the conventional algorithm on various maps. The results show that the proposed algorithms provide better passability, smoothing, and robustness, and reduced planning time.*

Keywords: Path planning; A* algorithm; Contour map; Artificial potential field method; Least squares method

1. **Introduction.** With the rapid development of artificial intelligence technology, mobile robots have been applied across multiple domains [1–3]. Among these domains, path planning constitutes one of the most critical techniques for mobile robots. This process encompasses perception of the surrounding environment, establishment of an environmental model, control of movement speed, and various other functionalities [4]. Its fundamental objective is to adhere to specific path planning principles within environments containing irregular obstacles, thereby generating a smooth and collision-free path that meets designated criteria [5]. Presently, numerous research efforts have been dedicated to algorithms

associated with path planning, spanning from traditional approaches to biomimetic algorithms. Prominent algorithms in this field encompass the A* algorithm [6], the dynamic window approach [7], artificial potential field methods [8], the rapidly random-exploring tree (RRT) algorithm [9], ant colony algorithms [10], and particle swarm algorithms [11], in addition, there are some improved optimization algorithms that can be applied in this field, such as the improved rolling grass optimization algorithm (CTOA) [12] and the stage population evolution (CPPE) algorithm [13]. As the integration of mobile robots into daily life becomes more pervasive, the demands on path planning complexity have correspondingly escalated, thereby imposing greater stringency on algorithmic requirements [14].

Path planning comprises both global and local path planning strategies. Concerning global path planning, Dutch scientist Dijkstra introduced an algorithm in 1959 to address the single-source shortest path problem, which marked the inception of the first global path planning approach [15] and laid the groundwork for the A* algorithm. In 1968, Hart et al. [16] introduced the heuristic A* algorithm to address global path optimization. However, this algorithm computes a substantial number of irrelevant nodes during the path planning process, resulting in challenges such as excessive computation, increased memory usage, and prolonged path planning times [17]. Yang [18] proposed an A* algorithm incorporating a jump point search strategy. This algorithm designates a specific category of pivotal nodes as jump points within the grid map and utilizes these points to eliminate a significant number of unnecessary nodes from the algorithm's calculations, thereby reducing the node count. However, as this strategy focuses only on searching for crucial points within a subset of nodes instead of conducting a comprehensive global search, the algorithm's performance lacks stability. Addressing deviations between the heuristic function of traditional A* algorithms and actual data in scenarios with complex obstacles, as well as the low precision in calculating costs for neighboring nodes, Chang [19] proposed an A* path planning algorithm based on an expanded search neighborhood. Although this algorithm yields smoother paths compared to conventional methods, its optimization is developed for relatively small map environments. Since the algorithm requires knowledge of the entire environmental layout, its computational complexity significantly increases for larger or more intricate map areas.

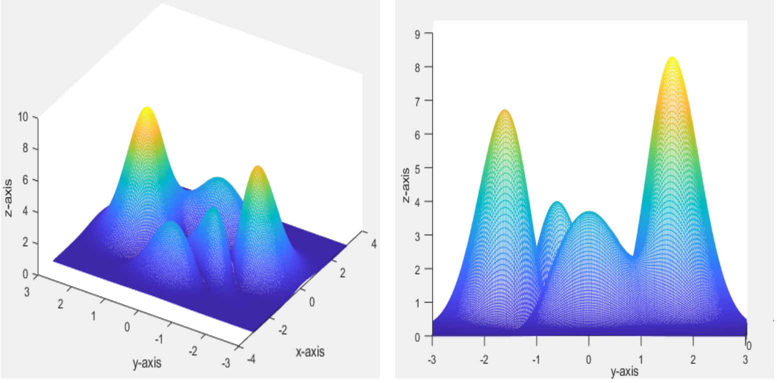
In the domain of local path planning, the artificial potential field method was introduced by Khatib in 1986 as a heuristic search algorithm applied to local maps [20]. This method simulates the map environment as a physics-based system of attractions, where a virtual attractive field spans the entire map toward the destination point. Within a certain range centered around obstacles, the mobile robot experiences a virtual repulsive force that varies with distance. However, the algorithmic rules of traditional potential field methods often lead to the robot becoming trapped in local optimal points or oscillating around the target. Addressing the limitations of the artificial potential field method, Yin and Xiang [21] proposed a fusion algorithm that combines the artificial potential field method with the grey wolf algorithm. This innovative fusion narrows the impact range of static obstacles and performs path planning between every pair of adjacent path points. While this fusion algorithm enhances path planning efficiency and reduces path turning points, its limited search scope can result in inadequate dynamic obstacle avoidance when multiple obstacles interact simultaneously. To simplify collision prediction with obstacles, Li and Wang [22] presented an approach that decides before the robot enters a local minimum point and simplifies obstacles when the robot exhibits oscillatory behavior. This algorithm effectively addresses the artificial potential field method's limitations in multi-obstacle environments and reduces the probability of the algorithm getting stuck in local optima. However, in dynamic environments, this algorithm fails to achieve efficient obstacle avoidance and often causes the robot to move in the opposite direction towards the target, rendering it unreachable. To overcome some of these challenges, Zhou [23] introduced modifications to the repulsive function by incorporating a target position function and altering the resultant force angle. These modifications improved the artificial potential field method by mitigating issues related to unreachable goals and becoming trapped in local minima. However, while tackling the problem of unreachable objectives, the increased iteration count led to longer path planning times, particularly evident in more complex maps.

In summary, to address the existing shortcomings of the algorithms, this paper proposes an improved path planning algorithm that combines the A* algorithm and the artificial potential field method. Firstly, in the initial phase of environment modeling for path planning, a novel map construction method is introduced. This method integrates the contour map drawing into a grid-based approach, resulting in more comprehensive map data. Secondly, in the process of global path planning, to overcome the issue of the traditional A* algorithm returning to the starting point for re-planning after encountering obstacles, this paper enhances the A* algorithm's list data and filtering conditions based on contour grid map data. This enhances both the speed of path planning and path traversability. Lastly, building upon global path planning, improvements are made to the traditional potential field method through the introduction of adaptive iteration functions, interval point search strategies, and range assessment strategies. Additionally, a least squares fitting technique is employed to enhance the smoothness of locally planned paths. This enhanced potential field method addresses the problems of local optimality and unreachable target points that are prevalent in traditional methods. Consequently, it enhances the mobile robot's ability to dynamically navigate around obstacles in local spaces.

2. Construction of contour-based grid maps.

Mobile robots rely on geometric models and use their positional information to generate maps of the environment. Common map models encompass grid maps [24], semantic maps [25], and topological maps [26]. Grid maps are commonly favored in the field due to their inherent capability to represent real obstacle dimensions and facilitate labeling. Thus, this paper utilizes two-dimensional grid maps for environmental modeling. Traditional A* algorithm, rooted in 2D grid-based environments, focuses primarily on finding the shortest paths to reach the target, neglecting the robot's traversability issues during the journey. To address this issue, the present study introduces the use of contour map construction principles [27]. As shown in Figure 1(a), the boundary of the obstacle is used as the dividing line and the height of the obstacle boundary is defined as 10. The height of the node increases in a gradient from the boundary of the obstacle to the centre of the obstacle, and conversely decreases in a gradient until it reaches zero. In scenarios where two overlapping obstacle locations with non-zero heights overlap (Figure 1(b)), the higher of the two is selected. The entire grid map is traversed systematically, mapping the elevation data of each grid to the corresponding node, thus generating a comprehensive map dataset. Incorporating the principles of contour-based grid mapping, this methodology yields a more nuanced representation of terrain features and elevation changes, contributing to a more accurate portrayal of the robot's traversable environment. By combining geometric models with elevation data, mobile robots can better navigate complex landscapes with enhanced awareness of real-world obstacles and passability considerations.

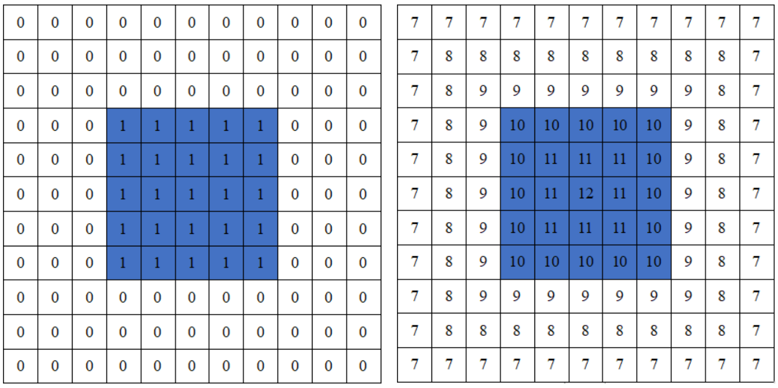
In traditional grid maps, nodes are typically assigned only two values: 0 and 1. Here, obstacle nodes are denoted as 0, while traversable nodes are represented as 1. As shown



(a) Side view representation (b) Rear view representation

FIGURE 1. Contour principle to handle the map environment

in Figure 2(a), although such grid maps are computationally efficient, they fail to capture the specific details of obstacles. Consequently, to address this limitation, the integration of contour data is introduced. This integration involves transforming the binary values (0 and 1) of traditional grid maps into corresponding height values for each node, as exemplified in Figure 2(b). This transformation enhances the representation of the map by incorporating elevation information.



0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

(a) Traditional grid maps

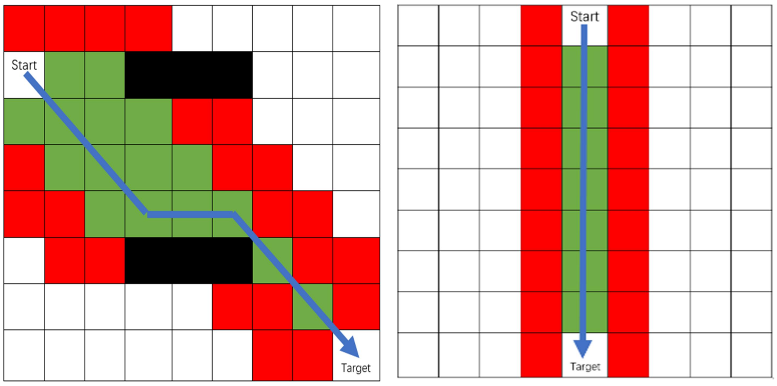
7	7	7	7	7	7	7	7	7	7	7
7	8	8	8	8	8	8	8	8	8	7
7	8	9	9	9	9	9	9	9	8	7
7	8	9	10	10	10	10	10	9	8	7
7	8	9	10	11	11	11	10	9	8	7
7	8	9	10	11	12	11	10	9	8	7
7	8	9	10	11	11	11	10	9	8	7
7	8	9	10	10	10	10	10	9	8	7
7	8	9	9	9	9	9	9	9	8	7
7	8	8	8	8	8	8	8	8	8	7
7	7	7	7	7	7	7	7	7	7	7

(b) Contour-based grid maps

FIGURE 2. Comparison of traditional raster maps and contour raster maps

3. The A* algorithm and its enhancements.

3.1. The traditional A* algorithm. The A* algorithm is a heuristic search technique employed for pathfinding in graph-based environments. It accomplishes path planning is achieved through the iterative evaluation of the cost associated with each adjacent node, iteratively progressing from parent to child nodes. The parent node represents the current position of the mobile robot, while the child nodes consist of the eight neighboring points. Figure 3 illustrates the fundamental principles of the traditional A* algorithm. As the algorithm traverses the graph, it selects nodes with the lowest combined cost (f value) to progress towards the goal while minimizing the traversal cost. This has made the A* algorithm a widely used path planning algorithm in applications ranging from robotics to computer games.



(a) A* algorithm for path planning with obstacles (b) A* algorithm for path planning without obstacles

Figure 3. Schematic diagram of traditional A* algorithm

As depicted in Figure 3, the visual representation illustrates different aspects of path planning using the A* algorithm in environments with and without obstacles. The utilization of color-coded nodes provides valuable insights into the behavior of the algorithm in these particular scenarios. In Figure 3(a), where the environment contains obstacles represented by black nodes, the A* algorithm navigates around obstacles by evaluating nodes and determining viable path points. Green nodes represent nodes deemed suitable for path consideration, while red nodes indicate nodes that have been evaluated but have not been selected as optimal path nodes. When the algorithm encounters obstacles, it dynamically employs a backtracking mechanism to recalibrate the path towards the initial starting point. This process continues until the path exploration width surpasses the width of the obstacles, allowing the robot to find a way through. Subsequently, the algorithm proceeds forward in the path planning process until it reaches the goal node. In Figure 3(b), where the environment is obstacle-free, the traditional A* algorithm engages in point-to-point path planning. The algorithm's computation and analysis involve minimal node quantities, resulting in a more streamlined and direct path planning process. It's evident that in obstacle-rich environments (Figure 3(a)), the A* algorithm adapts by exploring alternative paths around obstacles. In contrast, in obstacle-free environments (Figure 3(b)), the algorithm efficiently identifies a direct path. However, the traditional A* algorithm is associated with certain drawbacks. The algorithm frequently performs calculations and analyses multiple nodes that do not contribute in an optimal way, resulting in prolonged path planning durations and increased memory usage. Additionally, abrupt changes in path direction can arise, necessitating frequent acceleration and deceleration of the robot to adjust its velocity and direction, subsequently impacting the robot's overall average movement speed.

The core evaluation function of the traditional A* algorithm is:

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

$f(n)$ is the cost value from the current position to the start and end points, $h(n)$ is the cost value from the mobile robot from the current position to the target point position, and $g(n)$ is the cost value from the start point to the current position of the mobile robot.

When it comes to the cost functions $h(n)$ and $g(n)$, commonly used metrics include Euclidean distance and Manhattan distance [28]. The actual distance between the coordinates of two points in space corresponds to the Euclidean distance, whereas the sum of the absolute differences between the coordinates of the two points corresponds to the

Manhattan distance. Although the computation of Manhattan distance is relatively fast, its accuracy is not high. Therefore, in this study, the more accurate Euclidean distance is employed. The evaluation function for the Euclidean distance is given by.

$$\begin{aligned} g(n) &= \sqrt{(X_n - X_s)^2 - (Y_n - Y_s)^2} \\ h(n) &= \sqrt{(X_t - X_n)^2 - (Y_t - Y_n)^2} \end{aligned} \tag{2}$$

In the equation, (X_n, Y_n) represents the position of the current point, (X_s, Y_s) represents the position of the starting point, and (X_t, Y_t) represents the position of the target point.

3.2. **Improved A* algorithm.** Improvements to the A* algorithm include defining the current node occupied by the mobile robot as the parent node. The open list is defined as a repository for storing parent nodes along with neighboring nodes centered around the parent nodes. The closed list is defined as a collection for storing feasible path points for the mobile robot to traverse.

Once the map data has been acquired, enhancements to the A* algorithm are initiated with modifications to the data format and quantity constraints of the open list. The height value of nodes is included in the screening criteria, with a preference for selecting adjacent points with the lowest height values as path waypoints. In cases of identical heights, secondary screening is performed using a scoring function applied to nodes with equivalent height attributes. In cases where the evaluation function is used for node screening, priority is given to nodes with minimum cost values, thereby designating them as optimal path waypoints.

The obstacle node cost parameter is set to infinity in this study. Concurrently, a modification is introduced to the algorithm's structure wherein obstacle nodes continue to be included during node selection. The decision logic of the A* algorithm entails sorting the cost values of neighboring nodes to derive child nodes. As the cost value of obstacle nodes has been established as infinite, the enhanced A* algorithm inherently screens out obstacle nodes when applying node cost parameters for child node selection. Consequently, when the mobile robot encounters obstacles, it will no longer resort to revisiting the vicinity of the starting point for reinitiating path planning.

The refined A* algorithm comprises two distinct phases: the parameter adjustment phase, and the algorithmic pathfinding phase. The flowchart, demonstrating the path planning process, is depicted in Figure 4.

The parameter adjustment phase comprises four principal components:

(1) Processing of the contour map of the environment;
(2) Mapping of height data to corresponding grid points;
(3) Initialization of algorithm parameters;
(4) Placement of the starting point within the Closed list, and inclusion of the starting point and its adjacent nodes within the Open list.

The outcomes of the path planning via the improved A* algorithm are illustrated in Figure 5. In the figure, the black grid nodes represent obstacle points, the green nodes signify feasible path points, and the red nodes denote nodes that were initially included in the Open list for computation but subsequently filtered out. When comparing the path generated in Figure 5 exhibits a more rational trajectory, successfully circumventing the 'wall hugging' deficiency.

4. **Artificial potential field method and its enhancements.**

4.1. **Artificial potential field method.** The From the perspective of an individual obstacle's repulsive field or the attractive field generated by a target point, the influence of these potential fields on a mobile robot varies along the gradient direction. The artificial

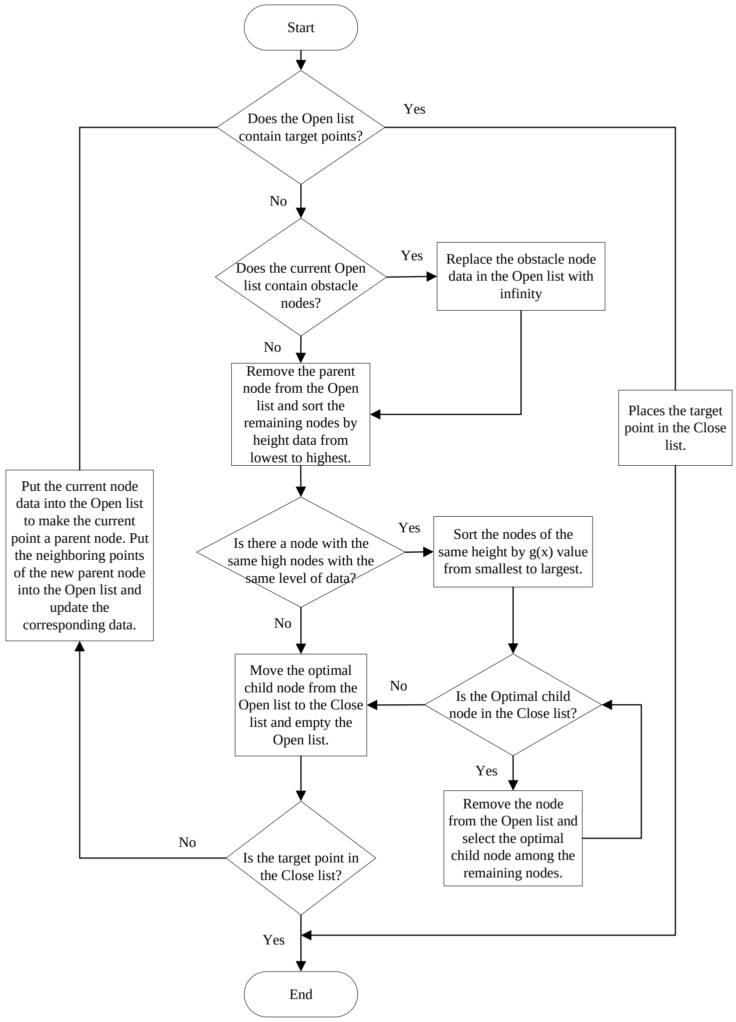

Start
Does the Open list contain target points?
Yes
No
Does the current Open list contain obstacle nodes?
Yes
Replace the obstacle node data in the Open list with infinity
No
Remove the parent node from the Open list and sort the remaining nodes by height data from lowest to highest.
Places the target point in the Close list.
Put the current node data into the Open list to make the current point a parent node. Put the neighboring points of the new parent node into the Open list and update the corresponding data.
Is there a node with the same high nodes with the same level of data?
Yes
Sort the nodes of the same height by g(x) value from smallest to largest.
No
Move the optimal child node from the Open list to the Close list and empty the Open list.
No
Is the Optimal child node in the Close list?
Yes
Remove the node from the Open list and select the optimal child node among the remaining nodes.
No
Is the target point in the Close list?
Yes
End

Figure 4. Improved A* algorithm path planning flow chart

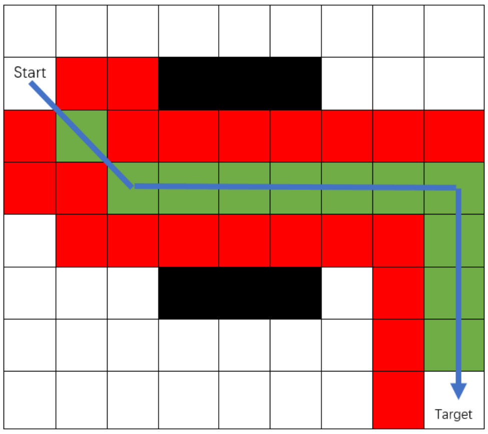


Figure 5. Improved A* algorithm path planning diagram

potential field method has found application within the field of dynamic obstacles, as evidenced by numerous scholars [29]. This methodology revolves around the application of repulsive forces from obstacles on the mobile robot and attractive forces originating from the target point. The repulsive force is oriented from the obstacle towards the mobile robot, while the attractive force is oriented from the mobile robot towards the target point. The cumulative impact of repulsive forces from multiple obstacles and the attractive force of the target point converge upon the mobile robot, giving rise to an overall force field. The force field experiences real-time modifications as the mobile robot traverses, a dynamic evolution described by Equation (3).

$$U_{att}(x) = \frac{1}{2}K\|(P_s - P_t)\|^2 \tag{3}$$

Where $U_{att}(x)$ is the gravitational potential field generated by the target on the mobile robot, K is the coefficient of action of the gravitational field, and $\|(P_s - P_t)\|^2$ is the Euclidean distance between the current point and the endpoint.

The gravitational force is the negative gradient of the gravitational potential field, as shown in Equation (4).

$$F_{att}(x) = \nabla U_{att}(x) = \frac{1}{2}K\|(P_s - P_t)\|^2 \tag{4}$$

Therefore, the magnitude of the repulsive field is inversely proportional to the distance between the mobile robot and the target point, as shown in Equation (5).

$$U_{rep}(x) = \begin{cases} K_{rep}\left[\frac{1}{\|(P - P_{obs})\|} - \frac{1}{P_0}\right]^2 & \|(P - P_{obs})\| \leq P_0 \\ 0 & \|(P - P_{obs})\| > P_0 \end{cases} \tag{5}$$

Where U_{rep} is the repulsive force field of the obstacle, K_{rep} is the coefficient of action of the repulsive force field, $\|(P - P_{obs})\|$ is the Euclidean distance between the mobile robot and the obstacle, P_0 and is the critical distance of the repulsive force of the obstacle. When the distance between the mobile robot and the obstacle is greater than P_0, the repulsive force on the mobile robot is zero. Meanwhile, the negative gradient of the repulsive force is shown in Equation (6).

$$F_{rep}(x) = -\nabla U_{rep}(x) = \begin{cases} K_{rep}\left[\frac{1}{\|(P - P_{obs})\|} - \frac{1}{P_0}\right]\frac{1}{\|(P - P_{obs})\|^2} & \|(P - P_{obs})\| \leq P_0 \\ 0 & \|(P - P_{obs})\| > P_0 \end{cases} \tag{6}$$

The algorithmic rules of the traditional potential field method lead to the fact that if there is an obstacle near the target point, the repulsive force is significantly stronger than the gravitational force of the target point, which can cause the mobile robot to be unable to approach the target point or even move in the opposite direction. When the mobile robot moves to the point where the repulsive force generated by the obstacle is less than the gravitational force generated by the target point, the mobile robot continues to travel toward the target location and repeats the process. If the mobile robot moves to a position where the combined force of the repulsive and gravitational forces is zero, the mobile robot will fall into a local optimum point. If there are many obstacles around the target point, when the mobile robot reaches this position, it will continue to move around the target point but will not be able to reach it.

4.2. **Improved potential field method.** To address the aforementioned issues, this paper proposes an interval point search strategy. Firstly, the turning points of the global path planning are identified, and the interval points are selected based on the specified distances. For the problem of how to determine the turning points, this paper adds a judgment function, as shown in Equation (7).

$$\begin{aligned} k_1 &= \frac{X_{n-1} - X_n}{Y_{n-1} - Y_n} \\ k_2 &= \frac{X_n - X_{n+1}}{Y_n - Y_{n+1}} \end{aligned} \tag{7}$$

Where (X_n, Y_n) is the current point, (X_{n+1}, Y_{n+1}) is the next path point, and (X_{n-1}, Y_{n-1}) is the previous path point. If $k_1 \neq k_2$, then the point (X_n, Y_n) is the path turning point. The algorithm replaces the point (X_n, Y_n) with (X_{n-1}, Y_{n-1}) until the objective point is reached, thus looping. If , then the algorithm continues until the goal point or turning point is reached. Finally, all the turning points of the global path are extracted and the interval points are determined as illustrated in Figure 6.

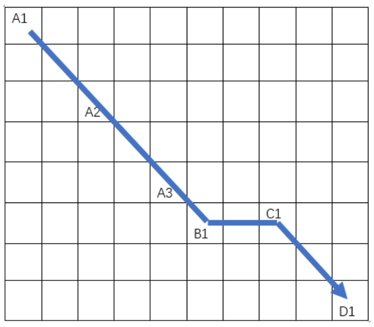


Figure 6. Principle of spacer selection

As shown in Figure 6, the starting point A1 is input as the first interval point, then the distance parameter R is defined, and the interval points A2 and A3 between A1 and B1 are selected according to the parameter. If the distance between the current interval point and the next turning point is less than the distance parameter R, the next turning point is used as an interval point. For example, if the Euclidean distance between A3 and B1 is less than R, B1 is used as the interval point after A3; if the distance between two consecutive turning points is less than the distance parameter R, the two turning points are used as interval points. For example, if B1 to C1, the distance between the two points is less than the distance parameter R, then there is no longer an interval point between B1 and C1. Therefore, the two points B1 and C1 are used as interval points. And so forth, until the desired point is included in the set of interval points.

$$\begin{aligned} \sqrt{(X - X_c)^2 - (Y - Y_c)^2} = R^2 \\ \frac{X_c - X}{Y_c - Y} = \frac{X_z - X_c}{Y_z - Y_c} \end{aligned} \tag{8}$$

Where (X_c, Y_c) is the current interval point and the default starting point (X, Y) is the first interval point. (X_z, Y_z) is the next interval point selected. If the distance (X_c, Y_c) to (X_z, Y_z) is less than , it (X_z, Y_z) will be used as the interval point. is the distance to the custom interval point.

After getting all the interval points, they are systematically arranged, with the initial point as the first and the ending point as the last. Based on the interval point list, this paper proposes an interval point, as depicted in Figure 7.

In the context of local path planning and dynamic obstacle avoidance, the enhanced potential field method uses the interval point S as a temporary starting point, while the point S+2 is designated as the temporary endpoint. The enhanced potential field method

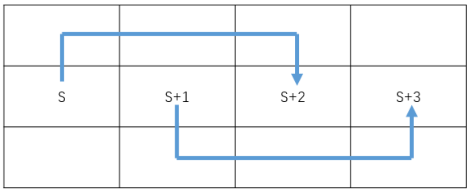


FIGURE 7. Interval point path planning diagram

continuously monitors in real-time whether the mobile robot has reached or surpassed the point S+1. If it has reached or passed the point S+1, the point S+1 is taken as the new temporary starting point, the point S+3 is taken as the new temporary endpoint, and so on, until the penultimate interval point is reached. When the penultimate interval point is reached, it is designated as the new temporary starting point, while the target point is assigned as the temporary endpoint.

The gravitational force generated on the target point in the traditional potential field method will decrease as the distance from the mobile robot to the target point decreases, while the gravitational force generated by the obstacle will increase as the distance between the mobile robot to the obstacle decreases, which will lead to an unreachable target point of the mobile robot if there is an obstacle near the target point, so this paper proposes to incorporate a range judgment strategy. When the mobile robot enters the range with the target point as the center and the predetermined distance parameter as the radius, the size of the gravitational force and the repulsive force are conducted. If the gravitational force is greater than the combined force of the repulsive force, the path planning continues, and if the repulsive force is greater than the gravitational force, the gravitational force function is replaced with Equation (9).

$$F_{att}(x) = \nabla U_{att}(x) = \left(\frac{1}{2}K\|P_s - P_t\|^2\right)^{-1} \tag{9}$$

Simultaneously replace the repulsion function with Equation (10).

$$F_{rep}(x) = -\nabla U_{rep}(x) = \begin{cases} \left(K_{rep}\left[\frac{1}{\|(P-P_{obs})\|} - \frac{1}{P_0}\right]\frac{1}{\|(P-P_{obs})\|^2}\right)^{-1} & \|(P-P_{obs})\| \le P_0 \\ 0 & \|(P-P_{obs})\| > P_0 \end{cases} \tag{10}$$

In the conventional potential field approach, the iteration count is set to a fixed value. Too many iterations will lead to long search time and increased data redundancy, whereas insufficient iterations can impede the robot's ability to reach the target point and increase the local optimum, so in the improved potential field method, an algorithm that adaptively adjusts the number of iterations based on the current position of the mobile robot, as depicted in Equation (11). The number of iterations changes in real-time according to the current position of the robot.

$$L = K\sqrt{(R_x - T_{ex})^2 + (R_y - T_{ey})^2} \tag{11}$$

Where K is the reference value, set to 10 here, and the Euclidean distance corresponding to each reference value is 1. (R_x, R_y) is the current position of the mobile robot and (T_x, T_y) is the coordinates of the current temporary target point. The Euclidean distance between the current position of the mobile robot and the temporary target point is rounded upwards. The number of iterations is adjusted once when the mobile robot changes the temporary start point and the temporary endpoint.

The path obtained by the improved potential field method will be discontinuous because there may be a turning point between the temporary starting point and the temporary ending point in the path planning process of the improved potential field method. As shown in Figure 8, J2 and J3 are turning points and interval points. When J2 is the temporary starting point, the path will be discontinuous. Therefore, as long as there is a turning point in the global path, the improved potential field method will have path discontinuity after smoothing. Therefore, this paper proposes to add the least squares method to the improved potential field method to solve the problem of path discontinuity, to obtain a continuous smooth path [30].

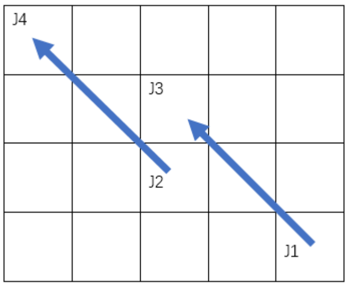


FIGURE 8. Path discontinuity

The central idea of the least squares method is to reduce the variance [29]. The expression for the variance of the hypothetical regression values and the actual values is given in Equation (12).

$$\varphi(x) = a_0 + a_1x + a_2x^2 + ... + a_kx^k \tag{12}$$

Where a is the polynomial coefficient to be determined, k is the highest order of the polynomial, and x is the horizontal coordinate of the path point. The sum of squared deviations of the function values is shown in Equation (13).

$$M = \sum_{i=0}^{n}[y_i - (a_0 + a_1x_i + a_2x_i^2 + ... + a_kx_i^k)]^2 \tag{13}$$

Where M represents the sum of squared errors, y_i is the distance from each point to the fitted curve. A partial derivation of the coefficients to be determined in the regression Equation (13) yields Equation (14).

$$\begin{pmatrix} 1 & x_1 & \dots & x_1^k \\ 1 & x_2 & \dots & x_2^k \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \dots & x_n^k \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_k \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \tag{14}$$

The coefficient matrix can be obtained by matrix operation in Equation (14). The coefficient matrix is brought into Equation (15) to obtain the fitted curve.

$$XA = Y \rightarrow A = (X^TX)^{-1}X^TY \tag{15}$$

5. **Simulation experiment and analysis.**

5.1. **Comparative analysis of simulation of improved A* algorithm.** This paper applies MATLAB for simulating and comparing algorithms. The simulation was conducted on Windows 10 Matlab 2019b, AMD Ryzen 5 5600H, using Radeon graphics @ 3.30GHz and 16GB of memory. When selecting the coordinates of the starting and ending points on the map, simplification is performed on both the x and y coordinates to ensure that both the starting and ending points can fall on integer points. As shown in Figure 9, a map of size 20×20 has been constructed. The red X marks represent obstacle areas on the map.

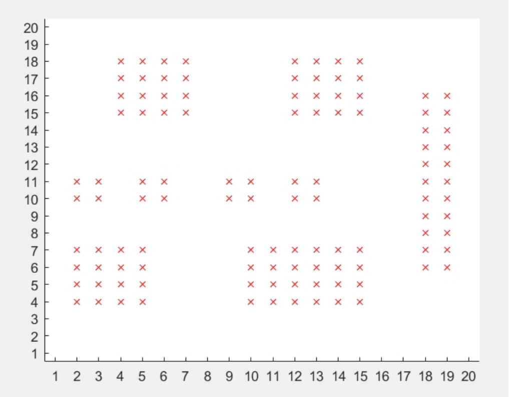
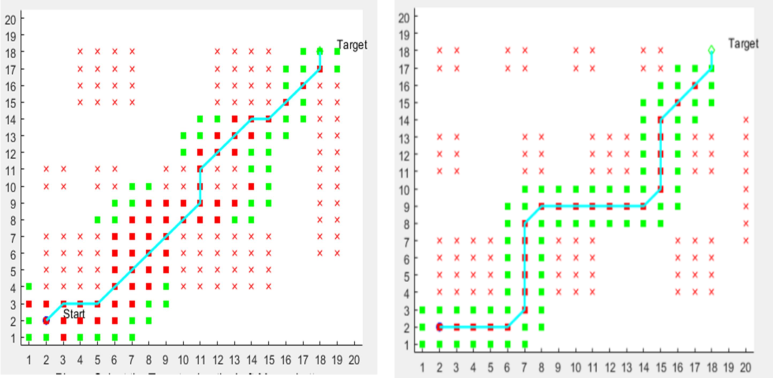


Figure 9. Building an experimental map environment

Firstly, different start and end points are set in Figure 9, and simulations are conducted using both the traditional A* algorithm and the improved A* algorithm. The experimental results are depicted in Figures 10 and 11. The green squares represent path points that have been calculated but are not the optimal child nodes, whereas the red squares represent feasible path points that have been computed. The blue lines represent the global path.

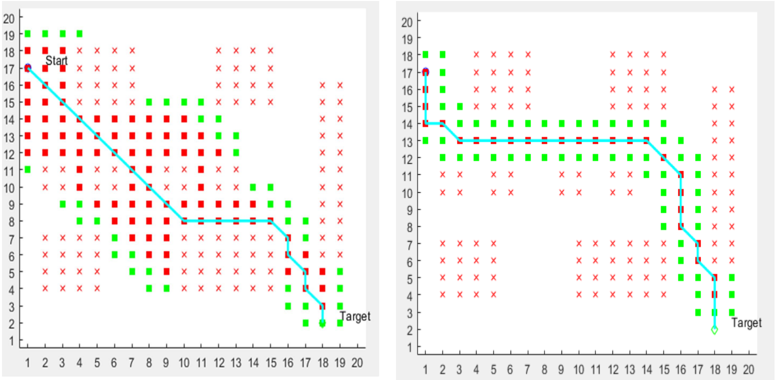


(a) Simulation results for the traditional A* algorithm

(b) Simulation results for the improved A* algorithm

Figure 10. Simulation comparison between traditional A* algorithm and improved A* algorithm based on lower-left starting point and upper-right ending point

As shown in Figure 10(b) and Figure 11(b), the improved A* algorithm successfully fulfills the requirements for path planning in the same environment with identical starting and ending points. Furthermore, as demonstrated in Figure 10(b) and Figure 11(b), where

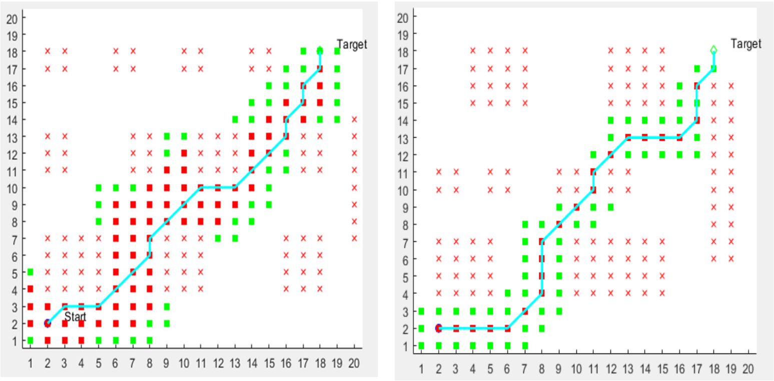


(a) Simulation results for the traditional A* algorithm (b) Simulation results for the improved A* algorithm

Figure 11. Simulation comparison between traditional A* algorithm and improved A* algorithm based on upper-left starting point and lower-right ending point

$x \in (10, 14)$, due to the traditional A* algorithm seeking a single path with the minimum cost, the resulting path hugs closely to obstacles, leading to poor traversability for the moving robot. Within this interval, the path generated by the improved A* algorithm lies between the two obstacles, thus providing better navigational capability. In Figure 11(b), the traditional A* algorithm computes 106 nodes, while the improved A* algorithm, due to its modification in the selection criteria for optimal child nodes, calculates only 83 nodes. Consequently, the improved A* algorithm reduces the number of nodes to be computed by 21.7%, and as the complexity of the map environment increases, the advantage of reduced node computation becomes even more pronounced.

To validate the robustness and speed of the improved A* algorithm, this paper conducts further simulations, using different map environments, to compare it with the traditional A* algorithm once more.



(a) Simulation results for the traditional A* algorithm (b) Simulation results for the improved A* algorithm

Figure 12. Simulation comparison based on other map environments

As shown in Figure 12, even after modifying the map environment, the improved A* algorithm continues to meet the requirements for path planning. Due to the reduced number of nodes that the improved A* algorithm needs to compute, a comparison of path planning times is made based on Figure 12(a) and Figure 12(b). As shown in Table

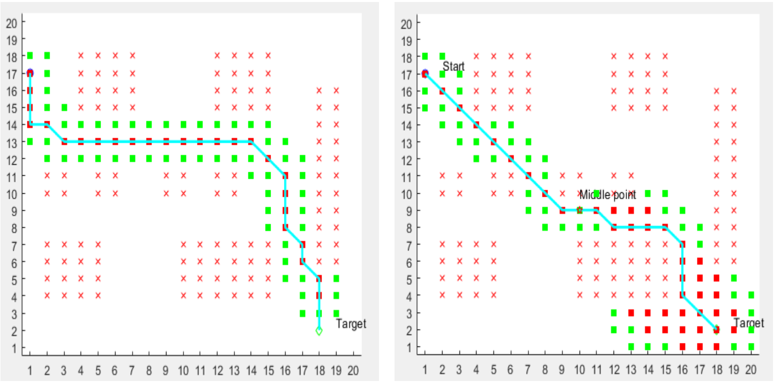
Table 1. Comparison of path planning time based on Figure 12

Algorithm	Number of simulations										Average time (s)
	1	2	3	4	5	6	7	8	9	10	
Traditional A* algorithm	0.653	0.634	0.64	0.624	0.648	0.638	0.633	0.655	0.631	0.636	0.639
Improved A* algorithm.	0.429	0.442	0.447	0.439	0.435	0.434	0.439	0.437	0.431	0.44	0.437

* The data in the table represents the path planning time for the algorithm in various simulation runs, measured in seconds (s).

1, the average time taken by the improved A* algorithm is 0.437 seconds, while the traditional A* algorithm takes an average of 0.639 seconds. The path planning time of the improved A* algorithm is reduced by 31.6% compared to the traditional algorithm.

In addition, this paper conducts a simulation-based comparative analysis between the improved A* algorithm and the bidirectional A* algorithm, while ensuring that the map environment and the starting and ending points of the mobile robot remain the same.



(a) Simulation results for the improved A* algorithm

(b) Simulation results for the bidirectional A* algorithm

Figure 13. Simulation comparison of improved A* algorithm and bidirectional A* algorithm

Table 2. Comparison of path planning time based on Figure 13

Algorithm	Number of simulations										Average time (s)
	1	2	3	4	5	6	7	8	9	10	
Traditional A* algorithm	0.429	0.442	0.447	0.439	0.435	0.434	0.439	0.437	0.431	0.440	0.437
Improved A* algorithm.	0.537	0.568	0.531	0.546	0.561	0.539	0.535	0.530	0.536	0.534	0.437

* The data in the table represents the path planning time for the algorithm in various simulation runs, measured in seconds (s).

Based on the path planning results shown in Figure 13, both the bidirectional A* algorithm and the improved A* algorithm are capable of fulfilling the path planning requirements, with the improved A* algorithm demonstrating superior traversability. As indicated in Table 2, the average path planning time for the bidirectional A* algorithm is 0.542 seconds, while the improved A* algorithm averages 0.437 seconds for path planning. The improved A* algorithm achieves approximately 19.4% time savings compared to the bidirectional A* algorithm.

5.2. **Comparative analysis of the simulation of the improved potential field method.** Since each grid point in the simulated map of this paper has a size of 1, a predefined distance parameter of 2 is chosen. As shown in Figure 14, the target point is (17, 18), with four obstacles placed around it at positions (16, 19), (16, 17), (18, 17), and (18, 19). In this scenario, the traditional potential field method, due to equal attraction and repulsion forces, results in the target being unreachable.

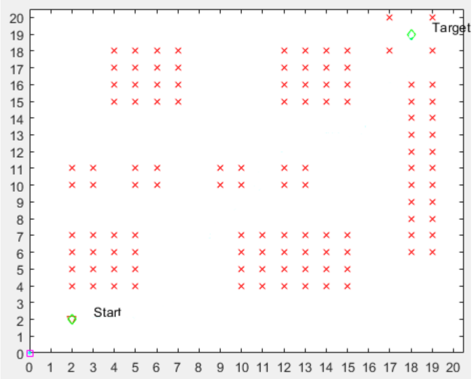
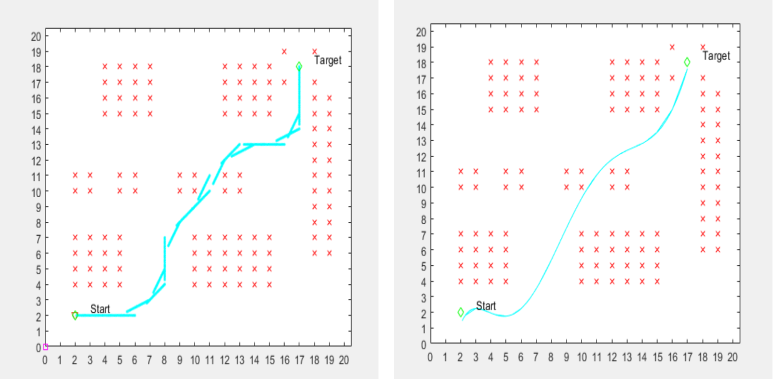


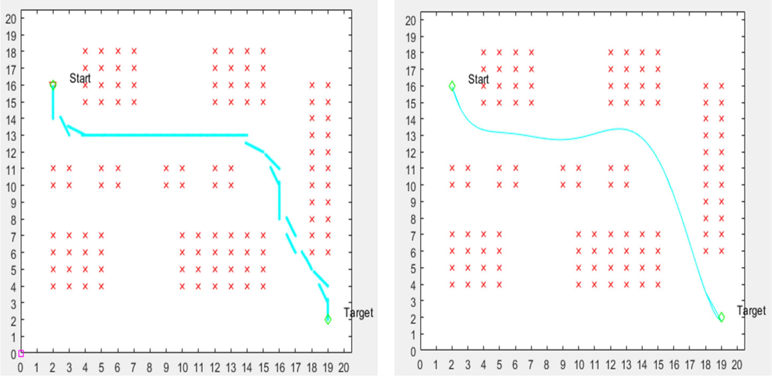
Figure 14. Obstacles are set up at the target point



(a) Path planning (b) Path fitting

Figure 15. Improved potential field method path planning

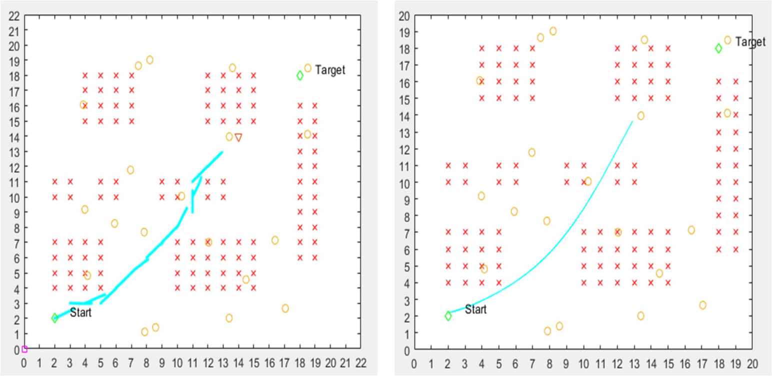
As shown in Figure 15, when obstacles are present near the target point, the improved potential field method can effectively navigate around obstacles and solve the problem of unreachable targets when the mobile robot's path planning reaches the range centered at the target point (17, 18). Additionally, the smoothed path is more favorable for the mobile robot's path tracking, allowing the robot to avoid frequent acceleration and deceleration at turns, thus reducing the time required to reach the destination. In the interval, the improved potential field method moves in the opposite direction of obstacles, creating more



(a) Path planning (b) Path fitting

Figure 16. Improved potential field method for path planning with different starting and stopping points on the same map

space for movement by avoiding obstacles and increasing turning speeds. As depicted in Figure 16, after redefining the starting and ending points, the improved potential field method can still meet the path planning requirements and maximize the mobile robot's walking space. Considering the possibility of dynamic obstacles in the environment, this paper introduces 20 random obstacles on the map to simulate dynamic objects in the map environment, with these obstacles represented by yellow circles in Figure 17.

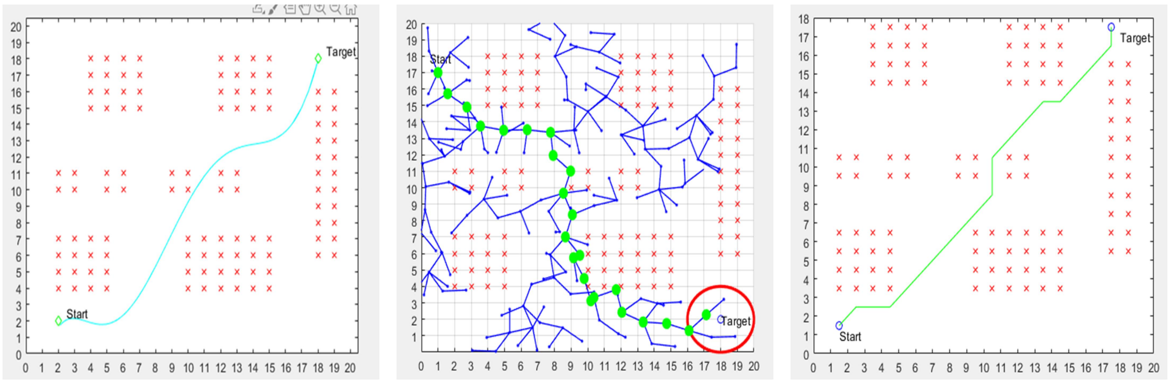


(a) Dynamic path planning (b) Dynamic path fitting

Figure 17. Adding dynamic obstacle path planning

As shown in Figure 17, during the path planning process, when dynamic obstacles appear in the map environment, the mobile robot responds quickly by avoiding these obstacles, ensuring that it moves toward a safe area while progressing toward the target position. As depicted in Figure 17(b), the mobile robot does not encounter the issue of small turning angles caused by avoiding dynamic obstacles.

5.3. **Comparative analysis of fusion algorithm with RRT algorithm and ant colony algorithm simulation.** Based on the comparison of path planning times presented in Figure 18(a) and Figure 18(b), as indicated in Table 3, the average path planning time for the proposed fusion algorithm is 0.857 seconds, whereas the average path planning time for the RRT algorithm is 3.124 seconds. Compared to the RRT algorithm, the fusion algorithm reduces path planning time by approximately 72.6%. Despite both the fusion algorithm and the RRT algorithm achieving the same path planning requirements,



(a) Fusion algorithm simulation graphs (b) RRT algorithm simulation graphs (c) Ant colony algorithm simulation graphs

Figure 18. Fusion algorithm with ant colony algorithm, RRT algorithm simulation diagram

Table 3. Comparison of path planning time based on Figure 18

Algorithm	Number of simulations										Average time (s)
	1	2	3	4	5	6	7	8	9	10	
The fusion algorithm proposed in this paper	0.834	0.884	0.817	0.876	0.887	0.829	0.848	0.885	0.852	0.860	0.857
RRT algorithm	3.134	3.092	3.098	3.186	3.054	3.124	3.132	3.169	3.143	3.104	3.124
Ant colony algorithm	2.015	2.043	2.031	2.138	2.051	2.044	2.012	2.052	2.049	2.052	2.049

* The data in the table represents the path planning time for the algorithm in various simulation runs, measured in seconds (s).

the fusion algorithm exhibits a significant advantage in terms of path planning time. Additionally, the fusion algorithm maintains a greater distance from obstacles, enhancing the robot's traversability. Based on the path planning times depicted in Figure 18(a) and Figure 18(c), the average path planning time for the ant colony algorithm is 2.049 seconds. In contrast, the fusion algorithm reduces path planning time, by approximately 58.2%. In summary, although both the fusion algorithm and the ant colony algorithm serve the purpose of path planning, the fusion algorithm completes the task in less time.

6. Conclusion.

This paper first integrates the principles of contour maps into a grid map environment, creating a contour grid map environment. Subsequently, the subnode selection rules of the traditional A* algorithm are restructured, and the cost evaluation function of the traditional A* algorithm is used as an auxiliary, resulting in the proposed improved A* algorithm. This approach addresses the issues of the traditional algorithm, which tends to return to the starting point and replan the path when encountering obstacles, as well as the limited workspace for the mobile robot. Next, within the traditional potential field method for local path planning, interval point path planning strategies are introduced. These strategies involve extracting interval points from the global path and using temporary starting and ending points instead of fixed ones for local path planning and dynamic obstacle avoidance. This modification resolves the problem of the traditional potential field method often converging to local optima. Additionally, a range determination strategy is incorporated, combined with an improved repulsion function and attraction function, to address the issue of unreachable target points. The fixed iteration parameters of the traditional potential field method are replaced with adaptive iteration functions, reducing computational complexity and data redundancy. The inclusion of the least-squares method for path fitting results in smoother and continuous movement routes. Experimental results demonstrate that, compared to traditional algorithms, the A* optimization algorithm reduces path planning time by 31.6% while improving path traversability under consistent map environments and starting, and ending points. Based on experimental data validation and analysis, the proposed fusion algorithm enhances path planning speed, and path smoothness, and reduces path curvature. The fusion technology enables mobile robots to quickly and effectively find safe paths in different map environments. Furthermore, comparative analyses are conducted with the optimization A* algorithm, bidirectional A* algorithm, RRT algorithm, and ant colony algorithm, confirming the advanced and robust nature of the fusion algorithm. Next, the authors will consider expanding the map environment and including more complex obstacles in the map to further verify the robustness and generalization of the optimized potential field method proposed in this paper.

Acknowledgment. This work is partially supported by the University-Industry Cooperation Project "Research and Application of Intelligent Traveling Technology for Steel Logistics Based on Industrial Internet", grant number 2022H6005, Natural Science Foundation of Fujian Provincial Science and Technology Department, grant number 2022J01952, and Research Start-up Projects, grant number GY-Z12079.

REFERENCES

[1] L. Liu, B. Wang, and H. Xu, “Research on path-planning algorithm integrating optimization a-star algorithm and artificial potential field method,” *Electronics*, vol. 11, no. 22, p. 3660, 2022.

[2] L. Liu, J. Liang, K. Guo, C. Ke, D. He, and J. Chen, “Dynamic path planning of mobile robot based on improved sparrow search algorithm,” *Biomimetics*, vol. 8, no. 2, p. 182, 2023.

[3] J. Liang and L. Liu, “Optimal path planning method for unmanned surface vehicles based on improved shark-inspired algorithm,” *Journal of Marine Science and Engineering*, vol. 11, p. 1386, 2023.

[4] Z. Liu, L. He, and H. Yuan, Liang aand Zhang, “Path planning of mobile robot based on tgwo algorithm,” *XAJT*, vol. 56, no. 10, pp. 49–60, 2022.

[5] C. Zhao, Y. Liu, L. Chen, F. Li, and Y. Man, “Research and development trend of multi-uav path planning based on metaheuristic algorithm,” *KZYC*, vol. 37, no. 5, pp. 1102–1115, 2022.

[6] Y. Wei, F. Jin, K. Dong, J. Son, W. Mo, and Y. Hui, “Improved global path planning a* algorithm based on node optimization,” *Computer Measurement and Control*, vol. 31, no. 6, pp. 143–148, 2023.

[7] W. Yongxiong, Y. Tian, X. Li, and L. Li, “Adaptive dynamic window method for traversing dense obstacles,” *Control and Decision*, vol. 34, no. 5, pp. 927–936, 2019.

[8] G. Chen, Z. Zhang, X. Huang, H. Hua, and Z. Shi, “Research on path planning of improved artificial potential field method in unknown environment,” *Machinery and Electronics*, vol. 39, no. 5, pp. 74–80, 2021.

[9] X. Wang, J. Wei, Z. Xia, and X. Gu, “Aeb-rrt*: an adaptive extension bidirectional rrt* algorithm,” *Auton Robot*, vol. 46, p. 685–704, 2022.

[10] H. Xue, K. T. Kim, and H. Y. Youn, “Dynamic load balancing of software-defined networking based on genetic-ant colony optimization,” *Sensors*, vol. 19, no. 2, p. 311, 2019.

[11] A. Antonakis, T. Nikolaidis, and P. Pilidis, “Multi-objective climb path optimization for aircraft/engine integration using particle swarm optimization,” *Applied Sciences*, vol. 7, no. 5, p. 469, 2017.

[12] T.-Y. Wu, A. Shao, and J.-S. Pan, “Ctoa: Toward a chaotic-based tumbleweed optimization algorithm,” *Mathematics*, vol. 11, no. 10, p. 2339, 2023.

[13] T.-Y. Wu, H. Li, and S.-C. Chu, “Cppe: An improved phasmatodea population evolution algorithm with chaotic maps,” *Mathematics*, vol. 11, no. 9, p. 1977, 2023.

[14] J. Yao, G. Xiong, and C. Zhang, “Design of automatic control system for obstacle avoidance of crawler robot,” *Journal of Nanchang University(Engineering and Technology)*, vol. 41, no. 4, pp. 385–390, 2019.

[15] E. Dijkstra, “A note on two problems in connexion with graphs,” *Numerische Mathematik*, vol. 1, pp. 269–271, 1959.

[16] P. E. Hart, N. J. Nilsson, and B. Raphael, “A formal basis for the heuristic determination of minimum cost paths,” *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[17] C. Cheng, X. He, J. Li, Z. Zhang, and G. Sun, “Global dynamic path planning based on fusion of improved a* algorithm and dynamic window approach,” *Journal of Xi’an Jiaotong University*, vol. 51, no. 11, pp. 137–143, 2017.

[18] J. Yang, “Research on path planning strategy of intelligent vehicle based on improved a-star algorithm,” Master’s thesis, Hubei Automobile University, Hubei, April 2021.

[19] C. Chang, “Research and application of path planning based on improved a-star algorithm,” Master’s thesis, Nanjing University, Nanjing, May 2020.

[20] O. Khatib, “The potential field approach and operational space formulation in robot control,” in *Adaptive and Learning Systems: Theory and Applications*. Springer, 1986, pp. 367–377.

[21] L. Yin and F. Xiang, “Path planning combined with improved grey wolf optimization algorithm and artificial potential field method,” *Electronic Measurement*, vol. 45, no. 3, pp. 43–53, 2022.

[22] E. Li and Y. Wang, “Research on obstacle avoidance trajectory of mobile robot based on improved artificial potential field,” *Computer Engineering and Applications*, vol. 58, no. 6, pp. 296–304, 2022.

[23] K. Zhou, “Research on path planing of moble robot in indoor complex environment,” Master’s thesis, Shanghai University of Engineering and Technology, Shanghai, April 2021.

[24] D. Li, T. Hong, and B. Zhang, “A grid-semantic fusion map construction method for mobile robot,” *Journal of Hangzhou Dianzi University (Natural Sciences)*, vol. 42, no. 2, pp. 21–26+63, 2022.

[25] Z. Li, “Research on real time semantic mapping based on uav visual localization,” Master’s thesis, Guangxi University, Guangxi, June 2022.

[26] Y. Wang, X. Zhang, F. Yan, and Y. Zhuang, “Large-scale outdoor road detection and topological map building for vision-based uavs,” *Unmanned Systems Technology*, vol. 4, no. 4, pp. 30–39, 2021.

[27] F. Wang and H. Pan, “Color processing technology of contour map based on programming,” *Highway*, vol. 66, no. 5, pp. 55–59, 2021.

[28] Y. Bian, X. Ma, F. Gao, and Y. Zhou, “Agv global path planning based on the improved a-star algorithm,” *Mechatronics*, vol. 25, no. 6, pp. 9–15, 2019.

[29] Z. Shi, S. Mei, Y. Shao, R. Wan, Z. Song, M. Xie, and Y. Li, “Research status and prospect of path planning for mobile robots based on artificial potential field method,” *Journal of Chinese Agricultural Mechanization*, vol. 42, no. 12, pp. 182–188, 2021.

[30] Q. Gao, L. Wang, and R. Wang, “Research on least square curve fitting and optimization algorithm,” *Industrial Control Computer*, vol. 34, no. 11, pp. 100–101, 2021.