Research Article

INTERNATIONAL JOURNAL OF ADVANCED ROBOTIC SYSTEMS

International Journal of Advanced Robotic Systems
May-June 2021: 1–13

DOI: 10.1177/17298814211026449
journals.sagepub.com/home/arx
SAGE

Research on path planning of three-neighbor search A* algorithm combined with artificial potential field

Jiqing Chen[1,2], Chenzhi Tan[1], Rongxian Mo[1], Hongdu Zhang[1], Ganwei Cai[1,2] and Hengyu Li[3]

Abstract

Among the shortcomings of the A* algorithm, for example, there are many search nodes in path planning, and the calculation time is long. This article proposes a three-neighbor search A* algorithm combined with artificial potential fields to optimize the path planning problem of mobile robots. The algorithm integrates and improves the partial artificial potential field and the A* algorithm to address irregular obstacles in the forward direction. The artificial potential field guides the mobile robot to move forward quickly. The A* algorithm of the three-neighbor search method performs accurate obstacle avoidance. The current pose vector of the mobile robot is constructed during obstacle avoidance, the search range is narrowed to less than three neighbors, and repeated searches are avoided. In the matrix laboratory environment, grid maps with different obstacle ratios are compared with the A* algorithm. The experimental results show that the proposed improved algorithm avoids concave obstacle traps and shortens the path length, thus reducing the search time and the number of search nodes. The average path length is shortened by 5.58%, the path search time is shortened by 77.05%, and the number of path nodes is reduced by 88.85%. The experimental results fully show that the improved A* algorithm is effective and feasible and can provide optimal results.

Keywords

A* algorithm, artificial potential field, path planning, obstacle avoidance

Date received: 21 February 2021; accepted: 23 May 2021

Topic Area: Mobile Robots and Multi-Robot Systems
Topic Editor: Nak-Young Chong
Associate Editor: Genci Capi

Introduction

With the development and progress of science and technology, the development and applications of robots are becoming increasingly mature. Compared with industrial robots in factories, mobile robots are obviously closer to people's lives. One of the core problems of a mobile robot is how to analyze and calculate a collision-free shortest path for the mobile robot to walk on in the map. The path planning problem should consider the obstacle distribution in the external environment and the calculation of path planning.

[1] School of Mechanical Engineering, Guangxi University, Nanning, China
[2] Manufacturing System and Advanced Manufacturing Technology Key Laboratory, Guangxi Univerisity, Nanning, China
[3] School of Mechatronic Engineering and Automation, Shanghai University, Shanghai, China

Corresponding author:
Hengyu Li, School of Mechatronic Engineering and Automation, Shanghai University, Shanghai 200444, China.
Email: lihengyu@shu.edu.cn

The more complex and changeable the external environment is, the higher the computational complexity is. Modern mobile robots are equipped with high-performance processors and sensors to explore, analyze, and process external unknown environments. When the specific situation of the external environment is known, only by combining path planning algorithms we can find the best path through in a short time. The environmental information obtained by external sensors can be used for environmental modeling, and the visual method is among the main methods of environmental modeling, which also include the Voronoi diagram method, the free space method, the topology method, the grid method, etc. Among these methods, the principle of the grid method is simple, and it is easy to realize the modeling, updating, and processing of the external environment. The grid method is one of the most widely researched and applied environmental modeling methods.[1,2]

In the past few decades, many scholars around the world have conducted research on path planning and derived many path planning algorithms, including the famous A* algorithm, the road map method, the probability method, the artificial potential field method, and the biologically inspired algorithm. The proposal of the A* algorithm in 1968 attracted the attention of the world.[3] The A* algorithm is an efficient heuristic search algorithm that performs well on grid maps, using the estimated cost of the node and the size of the cost as the priority in selection. When going to the next node, the node with the least cost is selected first, and the optimal path can be obtained by selecting nodes in turn for calculation.[4] The A* algorithm can quickly reach the shortest path. However, with the improvement of grid accuracy and the expansion of map size, repeated search and evaluation of useless nodes will cause the search time of the A* algorithm to increase exponentially.

Many researchers optimize the A* algorithm to improve efficiency. Uthus et al. proposed the iterative deepening A* algorithm for depth-first search by adjusting the threshold.[5] Botea et al. proposed a hierarchical method (hierarchical pathfinding A*) to reduce the complexity of the problem in raster map pathfinding.[6] Harabor and Grastien used corresponding node search strategies to identify valuable nodes in the surrounding neighborhood and optimized the A* algorithm while retaining the A* algorithm framework to achieve the effect of jumping point search.[7] Li et al., by constructing three indexes, used these indexes to calculate the range of the distance from the vertex to the target point and deleted some points that are not in the interval, thereby improving the performance of the algorithm.[8] Koenig et al. combined the idea of artificial intelligence by increasing or decreasing the number of obstacles over time, adding and deleting grid points, etc. In many cases, it is more efficient than reusing A* to search again.[9] Chabini and Lan extended the A* method to the shortest path to the network, where the propagation time arc is related to search again. By calculating the effective adaptation between nodes and multiple departure times on the starting node, the A* method was extended to the shortest path problem.[10] To reduce the calculation time of path planning, Shang et al. proposed an A* algorithm based on variable step length, which changes the step length with the distribution of obstacles.[11]

The artificial potential field also shows good performance in path planning.[12] The artificial potential field constructs a common gravitational field and repulsion field around the target location and obstacles. Then, by searching the downward direction of the potential function, the shortest collision-free path is planned. The entire potential field force is composed of gravity and repulsion. The efficient real-time controllability of the artificial potential field method can realize real-time path planning and smooth trajectory processing, so it has also been widely used. However, when multiple obstacles appear in the potential field space at the same time, the zero potential energy point easily appears, and the path planning task cannot be performed. This problem has been optimized. Pang et al. changed the function of the artificial potential field by combining visual information and corresponding auxiliary means to avoid the appearance of local minimum points.[13] Agirrebeitia et al. improved the artificial potential field by establishing a potential density along the geometric shape of the obstacle and solved the path planning problem to a certain extent.[14] Zhang et al. solved the local minimum problem in the process of obstacle avoidance by constructing a synthetic artificial potential field composed of a field and rotation vector.[15] Weerakoon et al. used partial information from obstacles to modify the force field function of the artificial potential field, which overcomes the problem of easily falling into a local minimum in path planning.[16] Zhang et al. proposed adding following walls and approaching target behaviors to help the robot escape the current trap when the robot falls into a local minimum.[17] Wu et al. proposed an artificial potential field method based on geometric modeling of obstacles, which greatly reduces local minima and trajectory oscillations.[18]

Some scholars try to combine the artificial potential field algorithm with the A* algorithm to achieve the effect of improving efficiency. Raheem and Abdulkareem constructed an enhanced roadmap through a new combination of probabilistic roadmaps and artificial potential fields and then found the optimal path in the enhanced roadmap through the A* algorithm.[19] Pan et al. solved the local minimum problem in path planning by setting intermediate target points in path planning. The intermediate target information is obtained through the A* algorithm, combined with the improved artificial potential field, to obtain the global optimal path.[20] Wang et al. obtained the optimized path points by using the quadratic A* search method and global path planning guided by these points. Local path planning is carried out by using an improved artificial potential field and by adding virtual subtargets to avoid

local minimum problems.[21] Ju et al. skillfully combined the artificial potential field and the A* algorithm, where the A* algorithm finds the shortest path, while the artificial potential field avoids obstacles.[22] The combination of the artificial potential field and the A* algorithm can break through the limitations of a single algorithm and demonstrate better results in path planning, but the specific optimization effect is determined by the superiority of the combined improved method.

This article proposes a three-neighbor search A* algorithm combined with an artificial potential field to solve the problem of repeated search of useless nodes by the A* algorithm and the local minimum of the artificial potential field method. The path planning of the improved algorithm consists of two main parts: fast forward and local precise obstacle avoidance. Under the action of an artificial potential field, the target point will generate a gravitational force to guide the mobile robot. Obstacles will not generate repulsive force to block the movement of the mobile robot or change its forward direction. When approaching obstacles, mobile robot will be converted to the A* algorithm of the three-neighbor search to avoid obstacles. Different from the traditional A* algorithm of eight-neighbor search, the A* algorithm of three-neighbor search will search only three or fewer of the eight surrounding grids and can avoid repeated searches of grids, which can greatly reduce the calculation time for obstacle avoidance while ensuring the optimal path. In addition, this research also screened irregularly distributed obstacles on the map and integrated them into regular obstacle areas, helping to optimize the trajectory of the mobile robot, avoid searching for useless grids, or making the mobile robot enter the concave obstacle trap.

Methodology

Environmental modeling

Before path planning, a map that the mobile robot can understand must be constructed. This article uses the grid method for environmental modeling. The shape of the grid includes a triangular grid, square grid, and hexagonal grid.[23] The traveling direction of the mobile robot is also divided into four-neighbor search, eight-neighbor search, and even variable-neighbor search,[24] and the accuracy of the grid method is divided into equal precision grid and variable precision grid.[25] This article uses a square grid with eight neighbors to search for environmental modeling. When modeling, the external environment is divided into equal rows and columns. Each time the mobile robot moves, the distance is a grid. Grids are divided into free grids and obstacle grids. The size of the grid can reflect the fineness of the environmental modeling. The smaller the grid is, the finer the environmental modeling is, but the longer the processing time is.

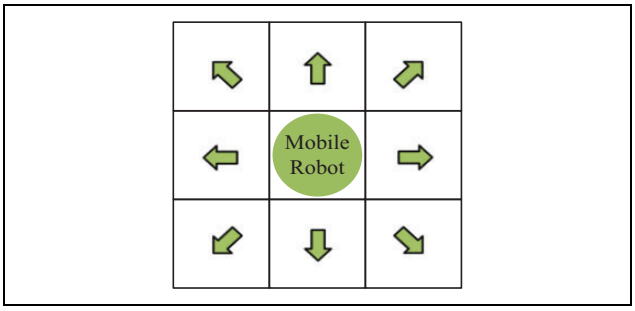

Figure 1. Schematic diagram of feasible progress.

Environmental modeling can be expressed by mathematical models. Let the grid map be $\boldsymbol{r}$ rows and $\boldsymbol{c}$ columns. The initial grid is (a_0, b_0). The traversed grid is recorded as (a_k, b_k). The target grid is (a_n, b_n). The length of the path that the mobile robot walks is calculated by the number of grids traversed. $\boldsymbol{E}$ represents the collection of grids in the grid

$$E = \{(i,j)|0 < i \le r, 0 < j \le c, i,j \in z\} \tag{1}$$

$B_{(i,\ j)}$ indicates that the information represented by grid (i, j) is a free grid or an obstacle grid

$$B_{(i,j)} = \begin{cases} 0, & \text{if grid}(i,j) \text{ is free grid}, (i,j) \in E \\ 1, & \text{if grid}(i,j) \text{ is obstacle grid}, (i,j) \in E \end{cases} \tag{2}$$

$D_{(i,\ j)}$ indicates that when the mobile robot is located at grid (i, j), the set of grids available for selection in the next step is shown in Figure 1

$$D_{(\mathrm{i,\,j})} = \begin{cases} (i+1,j),(i+1,j+1),(i,j+1),(i-1,j+1)|(i,j) \in E \\ (i-1,j),(i-1,j-1),(i,j-1),(i+1,j-1)|(i,j) \in E \end{cases} \tag{3}$$

$R(n)_{(i,\ j)}$ represents the grid that the mobile robot can select when starting from grid (i, j) and passing through the $\boldsymbol{n}$th step

$$R(n)_{(i,j)} = (a_k, b_k), (a_k, b_k) \in D_{R(n-1)_{(i,j)}} \cup B_{(i,j)} = 0 \tag{4}$$

P represents the set of all feasible grids

$$P = \{R(1)_{(a_k,b_k)}, R(2)_{(a_k,b_k)}, \ldots, R(n)_{(a_k,b_k)}|R(k)_{(a_k,b_k)} \in E, \\ k = 1, 2, \ldots, n \tag{5}$$

L represents the set of all feasible paths connecting the starting point and the target point. When solving the path planning problem, the path length J needs to reach the minimum value

$$J = \min\{\text{length}(L), L \in P\} \tag{6}$$

Equation (3) means that the mobile robot can select one of the grids in eight adjacent directions at a time as the next target grid. Equation (4) means that the grid selected by the mobile robot each time must be a free grid. Equation (5)

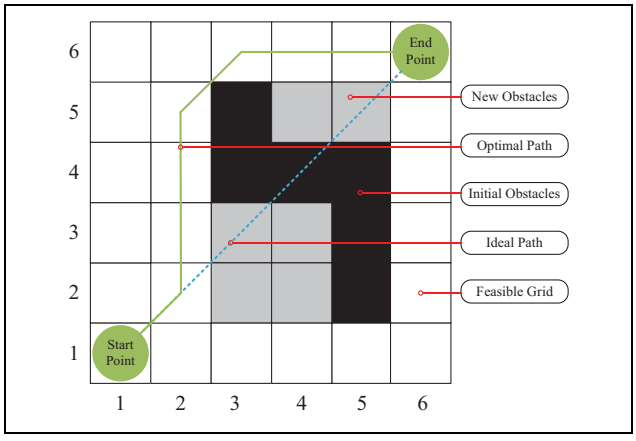

Figure 2. Schematic diagram of obstacle handling.

means that the feasible solution of the problem must start from the starting grid and finally reach the target grid.

In the actual path planning process, the disorderly distribution of obstacles may induce mobile robots to search the interior of some concave obstacles, which will increase the search time and reduce the efficiency of path planning. Therefore, this article proposes a method for initializing obstacles in the grid map. For the obstacles on the entire map, only the obstacles encountered in the direction of travel are processed. Every time an obstacle in a local area is processed, the next obstacle that should be processed will be replanned. The algorithm can regularize irregular obstacles and divide a local area for irregular obstacles in the forward direction. The algorithm uses a rectangular area to contain all irregular obstacles in this local area and does not perform any treatment on obstacles outside the forward direction to avoid meaningless operations. The effect is shown in Figure 2. The starting point coordinates are [1,1], and the target point coordinates are [6,6]. First, the starting point and the target point are connected, and the connecting line intersects the obstacle at position [4,4]. At this time, all adjacent obstacles around [4,4] are regarded as a whole area. A rectangle is used to contain the obstacle area, and all grids in the rectangle are regarded as obstacle grids.

The principle and implementation process of the algorithm are as follows. First, create two lists, Finish and Intersect, to connect the current position point with the target point, Determine the obstacle coordinates that intersect with the connecting line and store them in the transit list Intersect. To avoid repeated processing, the intersection needs to be purified, that is, only one obstacle coordinate is retained in an area, so that only one obstacle processing work is performed in an area, shortening the running processing time. Then, the depth-first search idea is used to search for each element in the Intersect list. Determine the obstacle information around the Intersect list to find all adjacent obstacles, and save the coordinate values in the Finish list after all searches are completed. The core step code is as follows:

Algorithm 1. Obstacle process.

```
Intersect← (a_i, b_j)
While Intersect not empty
    List←D_(i,j)
    If B_(List) ==1
      Intersect←B_(List)
    End
    (a_i, b_j) from Intersect to Finish
    To check next
End
```

Now, in Finish, the coordinate values of all obstacle points in the current local area are needed. Connect the maximum and minimum values of the horizontal and vertical coordinates in Finish to construct a rectangular area R_t

$$R_t = \begin{cases} \text{Line}(X_{\max}, Y_{\max}),\ X_{\max}, Y_{\max} \in \text{Finish} \\ \text{Line}(X_{\max}, Y_{\min}),\ X_{\max}, Y_{\min} \in \text{Finish} \\ \text{Line}(X_{\min}, Y_{\max}),\ X_{\min}, Y_{\max} \in \text{Finish} \\ \text{Line}(X_{\min}, Y_{\min}),\ X_{\min}, Y_{\min} \in \text{Finish} \end{cases} \tag{7}$$

All grids in the rectangular area R_t are considered obstacles

$$B_{(i,j)} = 1, \quad i,j \in R_t \in E \tag{8}$$

The last step is to process the coordinates of all obstacles in the Finish list, as shown in Equation (7) and use this information to construct the four vertices of the rectangular area. All grids in this rectangle will be regarded as impassable obstacles.

A* algorithm and improvement

The A* algorithm is a heuristic algorithm that can efficiently search for the correct path. Its main principle is to judge the direction of travel by judging the cost of the current grid to the next grid. The A* algorithm will give priority to the grid with the smallest cost value, and the cost value of the surrounding eight grids will be evaluated for each grid. The cost value of the grid is usually composed of two parts in the A* algorithm, which can be expressed as follows

$$F(n) = H(n) + G(n) \tag{9}$$

where $F(n)$ is the total cost value, which is the cost value from the initial grid to the nth grid, and $G(n)$ represents the real cost value from the initial grid to the nth grid

$$G(n) = \sum_{k=1}^{p-1} d(k) \tag{10}$$

where $p \geq 1$ is the number of grids from the initial grid to grid n, and $d(k)$ is defined as the Euclidean distance from

the kth grid to the $k+1$th grid. Specifically, the coordinates of the kth node are (x_k, y_k) and then $d(k)$ as follows

$$d(k) = \sqrt{(x_k - x_{k+1})^2 + (y_k - y_{k+1})^2} \tag{11}$$

$H(n)$ is the cost value of the best path from grid n to the target grid. The calculations generally include the Manhattan distance, Chebyshev distance, and Euclidean distance.[26] This article uses the Manhattan distance to calculate the following expression

$$H(n) = \text{abs}(n_x - \text{goal}_x) + \text{abs}(n_y - \text{goal}_y) \tag{12}$$

The implementation process of the A* algorithm is shown in Figure 3.

Although the A* algorithm can accurately find a correct path, due to the limitations of the algorithm principle, the surrounding eight neighbors will be searched regardless of whether the grid found in the path search process is an obstacle grid or a free grid, and the same grid will be searched repeatedly, increasing the time of path calculation. The node search method of the A* algorithm is shown in Figure 4. The mobile robot searches for a total of 24 nodes when moving from coordinate points [3,2] to [3,5] and repeats searches of some grids twice or even three searches (such as [2,3], [4,3] in Figure 4), greatly increasing the search time required by the A* algorithm when the map size is large.

To solve the problems of the A* algorithm searching for too many nodes and repeated searching for some nodes, this article improves the A* algorithm, changing the eight-neighbor search method of the A* algorithm, avoiding repeated search of nodes, and achieving three-neighbor search or even fewer neighborhood searches. The improvement principle is shown in Figure 5. Figure 5(a) is the operating principle diagram of the three-neighbor search. At the beginning, the mobile robot is located at coordinates [3,2]. Before searching the surrounding grid, first create a vector. The function of the vector is to indicate the target direction of the mobile robot. The direction of the vector is from the current position to the target point. Then, according to the direction of the position vector, determine the grids to be searched as [2,3], [3,3], and [4,3], that is, the grids that should be searched are the grids along the direction of the position vector and their left- and right-side grid. At this time, only the grids of three neighbors are searched, which avoids the search of useless grids. Obstacle discrimination and cost evaluation were performed on the grids at [2,3], [3,3], and [4,3], and it was determined that the mobile robot should go to the grid at [3,3] in the next step and proceed in sequence. Only nine grids were searched to reach the target point. In Figure 5(b), constructing the position vector at [3,2] shows that the grids to be searched are [2,3], [3,3], and [4,3], but these three grids are all obstacles. When the grids to be searched are all obstacle grids or grids that have been searched, the pose vector will be rotated 90 degrees counterclockwise and then searched along the new position vector. At this time, the repeated search is meaningful and can effectively avoid obstacles.

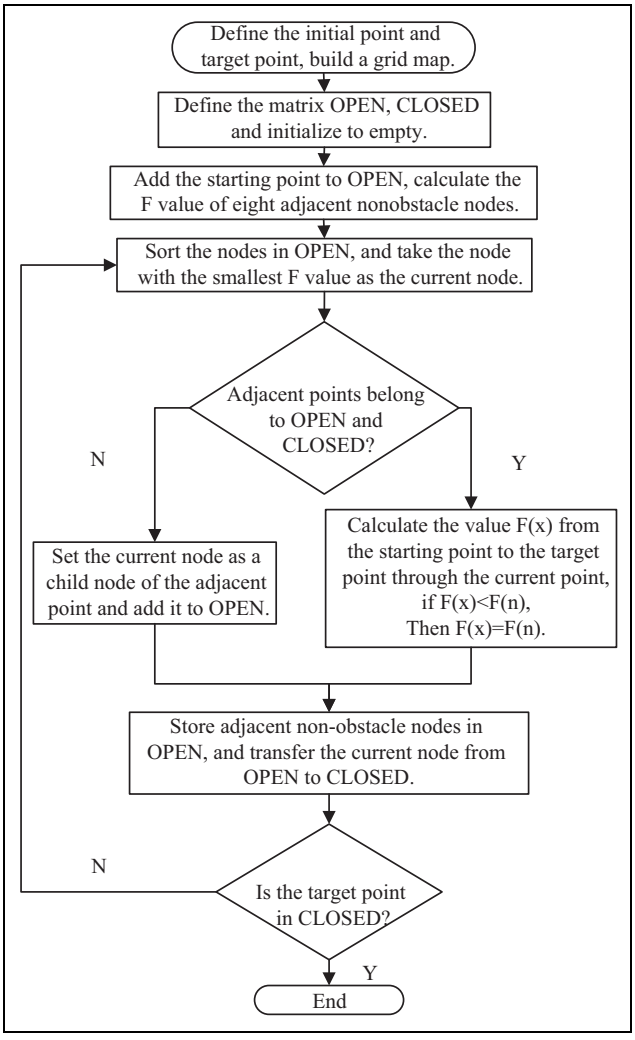

Figure 3. A* algorithm flow chart.

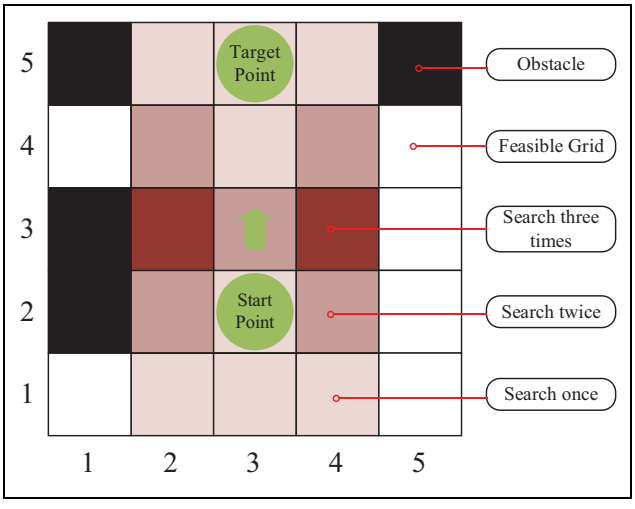

Figure 4. Schematic diagram of node search of A* algorithm.

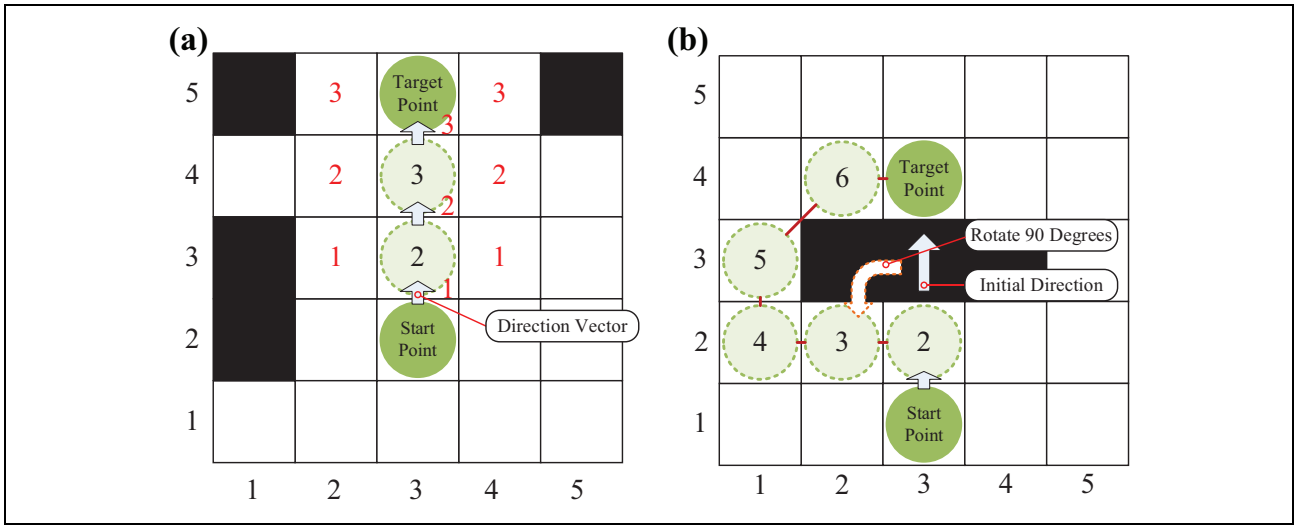

Figure 5. (a) and (b) Schematic diagram of the three-neighbor search A* algorithm.

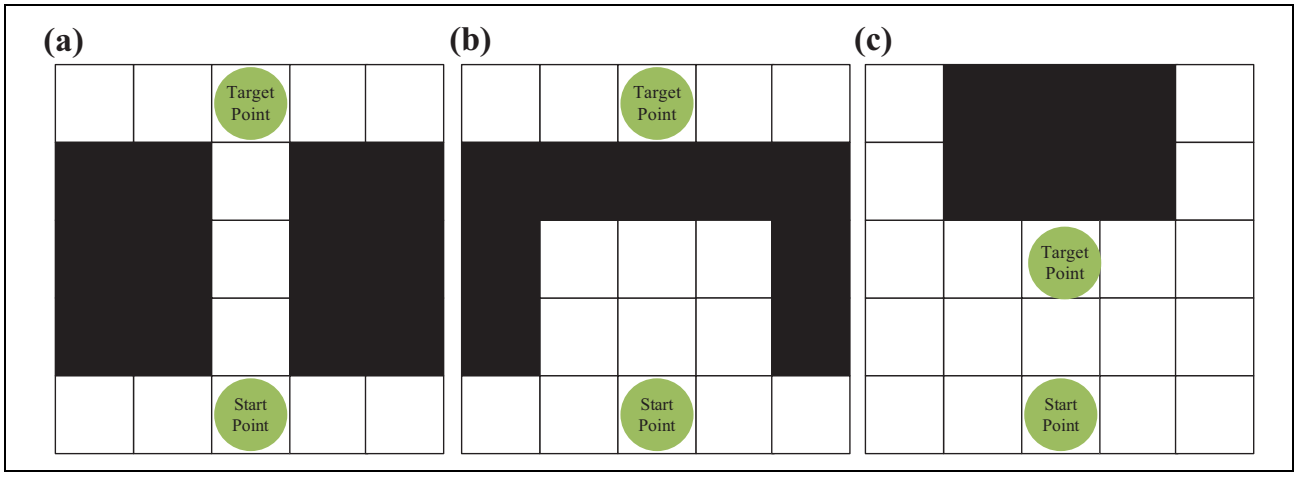

Figure 6. (a) to (c) Disadvantages of the artificial potential field method.

Introduction of partial artificial potential field

However, in practice, when there is a large blank area in the grid map, even the A* algorithm with three-neighbor search appears redundant. In this case, compared to the A* algorithm, the artificial potential field has the advantages of faster calculation speed, lower hardware requirements, and simple mathematical principles. The artificial potential field is applied to the path planning and smooth trajectory of mobile robots, which can reduce the redundancy of the A* algorithm, so this article introduces part of the artificial potential field algorithm to assist in pathfinding. The artificial potential field is one of the most commonly used methods in mobile robot path planning and obstacle avoidance. This method constructs two potential fields around the target point and obstacles. The target point is attractive to the mobile robot, and the obstacle radiates repulsive force around it. Then, by searching the downward direction of the potential function, the optimal path without collision is planned. As shown in Figure 6, although the mathematical principle of the artificial potential field algorithm is simple and the amount of calculation is small, the artificial potential field algorithm also has a fatal flaw. When multiple obstacles appear in the potential field space at the same time, the zero potential energy point appears easily, making the potential energy method enter the local minimum, causing confusion, and making the artificial potential field algorithm unable to carry out the path planning task in the potential field space. As shown in Figure 6(a), when the obstacle and the target points of the mobile robot are on the same straight line, the attractive and repulsive forces of the mobile robot will be equal at a certain position, the direction is opposite, and the resultant force is zero. At this time, the robot will enter the local minimum point and stop or vibrate severely at that point. As shown in Figure 6(b), when the robot is in an environment

surrounded by multiple obstacles, it may fall into the trap area and cannot escape autonomously. As shown in Figure 6(c), when the mobile robot approaches the target point due to the obstacle repelling field and when the repulsive force of the mobile robot is greater than the gravitational force provided by the target point, causing the robot to constantly oscillate or stop at the critical point, the mobile robot has the problem of an unreachable target point.

In the artificial potential field, the gravitational function and the repulsive force function need to cooperate with each other to complete the correct path planning calculation, but it is precisely because of the mutual cooperation of the gravitational function and the repulsive force function that the above defects occur. Therefore, in this article, only the gravitational field function is used for the artificial potential field, and obstacles will not generate a repulsive potential field. The gravitational field function can make the mobile robot reach the target point within a relatively short distance and less calculation time.

The core of the artificial potential field is the gravitational field function and the repulsive field function. The parameter changes in the function will directly affect the pathfinding effect.

The gravitational field function is as follows

$$U_{\text{att}}(X) = \frac{1}{2} * m_1 * d^2(X, X_g) \tag{13}$$

where m_1 is the gain coefficient of the gravitational potential field and $d(X, X_g)$ is the distance from the current mobile robot position to the target point.

The expression of the potential field force can be obtained by solving the potential field function with a negative gradient, where the expression of gravity is as follows

$$F_{\text{att}}(X) = m_1 * d(X, X_g) * \frac{\partial d(X, X_g)}{\partial X} \tag{14}$$

By searching the downward trend of the gravitational potential function to achieve the effect of fast-forwarding in the barrier-free area, the entire potential field force is composed of only the gravitational part and not the repulsion part. Therefore, there will be no zero potential energy point, and there will be no corresponding defects of the artificial potential field. As shown in Figure 7, the mobile robot is attracted to the target point, and the direction is from the starting point to the target point. Obstacles in the map will not generate repulsive force to interfere with the resultant force of advancement. As long as the robot moves in the direction of attraction, then the path taken must be the shortest path.

The gravitational field function will make the mobile robot reach the target point through the optimal path in a short calculation time, but it cannot achieve obstacle avoidance. Therefore, the gravitational field function needs to cooperate with the improved A* algorithm to complete the pathfinding work.

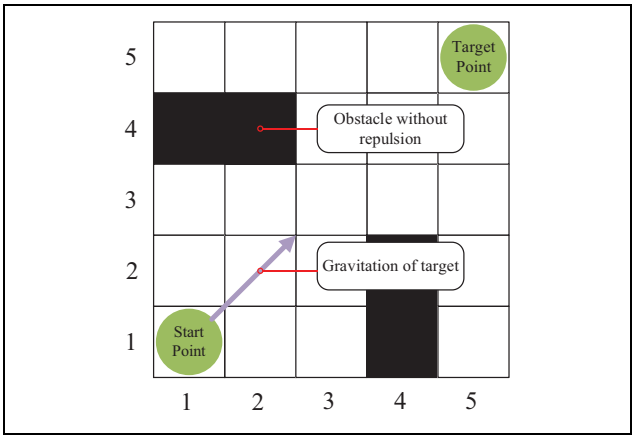

Figure 7. Illustration of part of the artificial potential field.

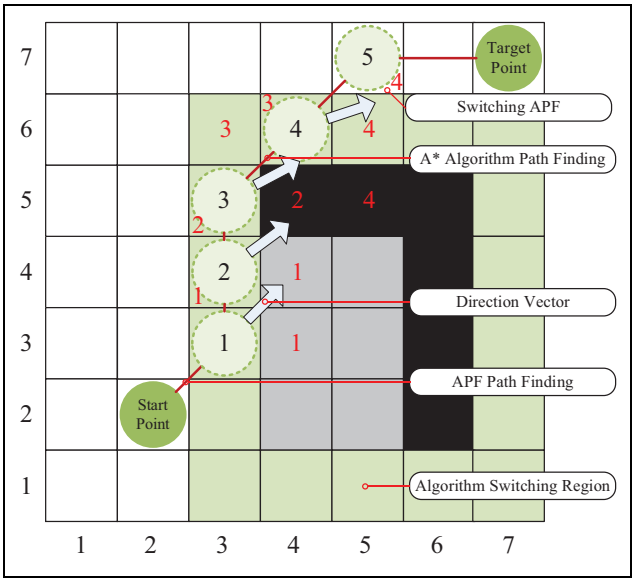

Figure 8. Schematic diagram of the three-neighbor search A* algorithm combined with the artificial potential field method.

Three-neighbor search A algorithm combined with artificial potential field*

Combining the advantages of the A* algorithm to find the path accurately and the artificial potential field to find the path quickly would be desirable. This article proposes a three-neighbor search A* algorithm combined with an artificial potential field. The algorithm can be divided into three parts, as shown in Figure 8. In Figure 8, [2,2] and [7,7] are the starting point and the target point, respectively. The same number as the number in the circle represents the grid searched at that position, and the light green area represents the algorithm switching area. When the mobile robot enters the area where the algorithm is switched, the

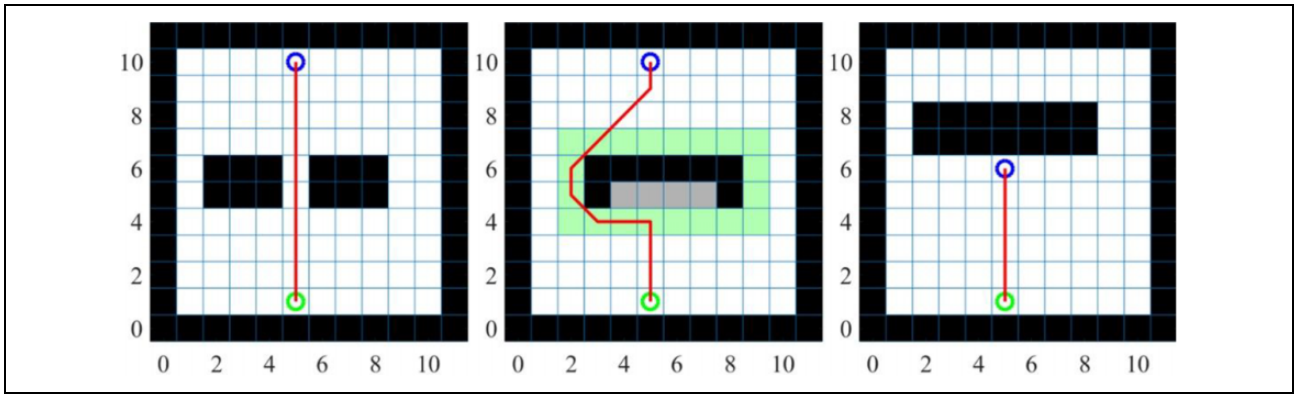

Figure 9. Part of the artificial potential field will not have a local minimum.

robot's entry will change the pathfinding algorithm from the artificial potential field method to the three-neighbor search A* algorithm. When leaving the green area, the pathfinding algorithm will change from the three-neighbor search A* algorithm to the artificial potential field. The black area is the obstacle distribution area, and the gray area is the virtual obstacle added after the obstacle is processed. The arrow in the figure, generated when the three-neighbor A* algorithm is running, represents the pose vector pointing to the target point at the current position.

The main steps of the algorithm proposed in this article are as follows: The first step is to process the constructed grid map. In this step, the irregular obstacles in the forward direction are processed, and the algorithm switching area is set around the processed obstacles. When the mobile robot enters or leaves the algorithm switching area, the pathfinding algorithm will switch. The second step is to use a partial artificial potential field to guide the mobile robot to move forward quickly in the area where there are no obstacles. At this time, the attraction of the target point to the mobile robot guides the mobile robot to move forward quickly. However, at this time, the robot does not have the ability to avoid obstacles. The third step is to use the three-neighbor search A* algorithm to accurately avoid obstacles in areas with obstacles. By switching back and forth between the second step and the third step, fast collision-free motion of the mobile robot can be realized.

Simulation

The simulation experiment will conduct a detailed comparison experiment between the proposed algorithm and the A* algorithm. The simulation experiment environment is as follows: the system environment is Windows 10, the processor is Intel Core I5-7300HQ, and the running memory is 16 GB; the compilation environment is matrix laboratory (MATLAB) 2016b. The gain coefficient of the gravitational potential field is $m1 = 5$, the grid method is used to construct the map environment, and the A* algorithm uses the eight-neighbor search method to search for the path, advancing one grid distance each time. The starting point and target point of the mobile robot are represented by green circles and blue circles, respectively, as shown in Figure 9. Although the algorithm proposed in this article uses the artificial potential field to guide the mobile robot forward, because only the gravitational part is used, the defect of the artificial potential field does not appear.

Under the condition that the ratio of the number of grids occupied by obstacles to the total number of grids is defined as the obstacle ratio, two sets of comparative experiments were carried out on grid maps with obstacle ratios of 0.15, 0.25, and 0.35. The size of the grid map is 20×20, and the size of the grid is 0.1×0.1 m. The improved algorithm is compared with the A* algorithm. As shown in Figure 10, the green grid in the graph of the improved algorithm represents the switching area of the improved artificial potential field and the improved A* algorithm, the light red area represents the number of nodes that need to be searched during the pathfinding process, and the dark red area is the node after being searched many times. The more times searched, the darker the color. The blue line segment in the figure is the path searched by the A* algorithm, and the green line segment is the path searched by the improved algorithm.

The results of the simulation experiment show that the proposed algorithm has a certain optimization effect on the selection of the path. The optimal path length becomes shorter, and the path search time is correspondingly shorter, which reduces the number of nodes expanded when searching for a path and improves the efficiency of the algorithm. As shown in Figure 10(a), in the vicinity of the [4,3]–[5,5] area, because the area is processed, the algorithm avoids entering the area for searching and reducing the search time and the number of search nodes.

As shown in Figure 11, in grid maps with different obstacle ratios, the algorithm proposed in this article optimizes the A* algorithm in grid maps with different obstacle ratios. The proposed algorithm has a very significant effect in optimizing the number of search nodes and search time, and the optimization of path length also has a certain effect. The quantitative analysis before and after the improvement

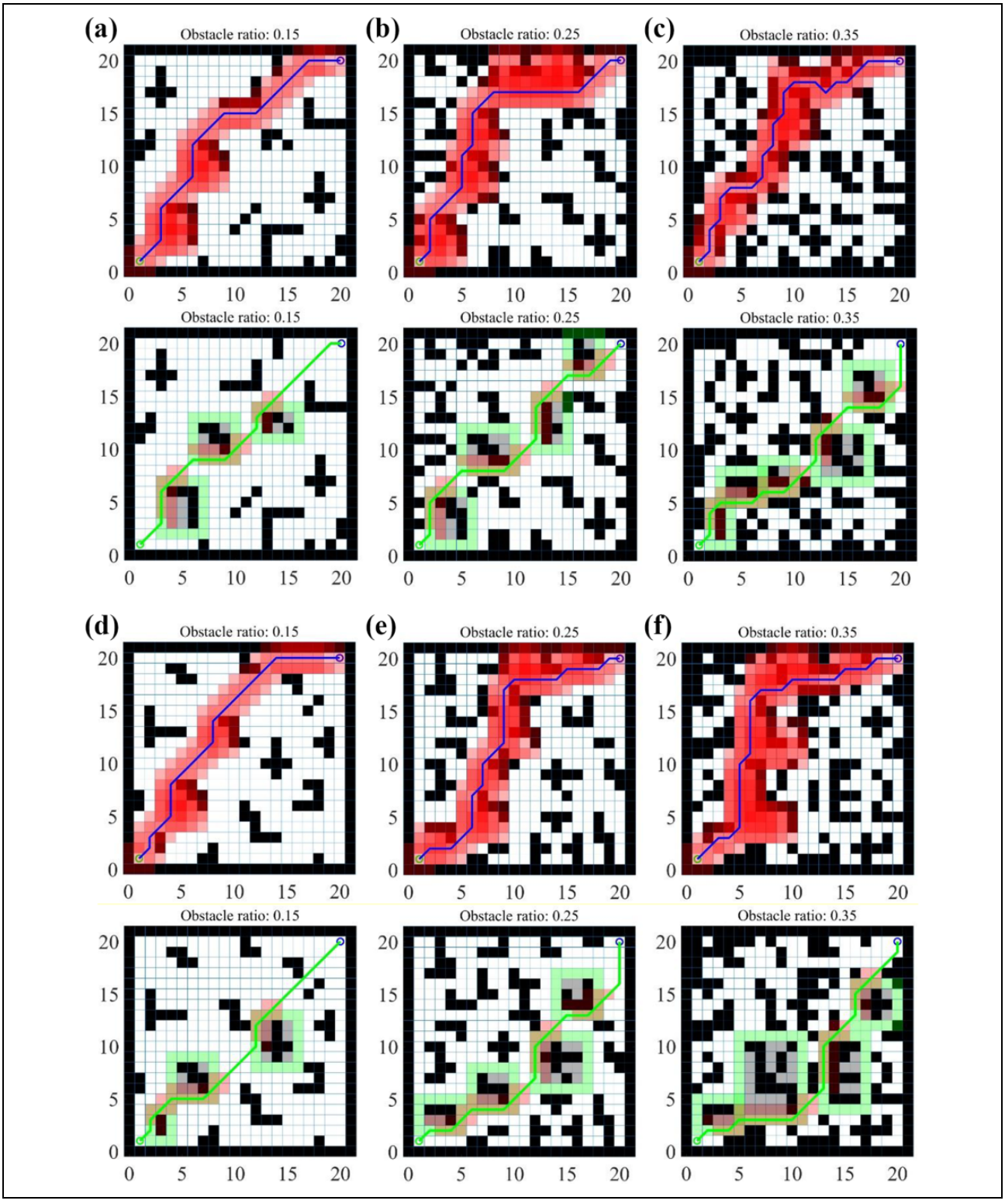

Figure 10. (a) to (g) Comparison of simulation experiment results.

of the algorithm in Figure 10 yields Table 1, where the unit of path length is centimeter(s) (cm), the unit of algorithm running time is second (s), and the number of path nodes represents the number of nodes searched by the mobile robot, and the obstacle ratio reflects the density of obstacles in the grid map.

Table 1 compares the A* algorithm and the algorithm proposed in this article for path planning with grid maps of

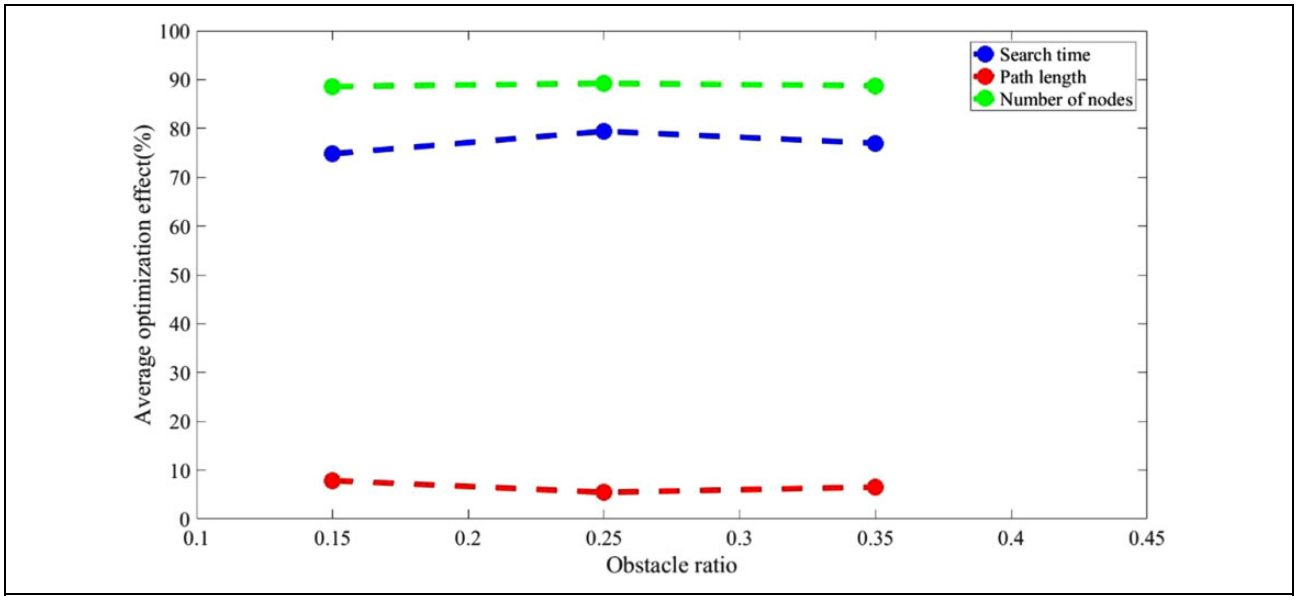

Figure 11. Optimization effect diagram.

Table 1. Simulation experiment results.

					Optimization effect (%)		
Grid map	Search time (s)	Path length (cm)	Number of nodes	Obstacle ratio	Time	Length	Nodes
A* algorithm (a)	3.275	303.85	288	0.15	77.43	3.86	88.89
Proposed algorithm (a)	0.739	292.12	32				
A* algorithm (b)	5.943	321.42	464	0.25	81.39	5.47	89.65
Proposed algorithm (b)	1.106	303.85	48				
A* algorithm (c)	4.903	335.56	368	0.35	74.93	5.96	86.96
Proposed algorithm (c)	1.229	315.56	47				
A* algorithm (d)	2.818	303.85	256	0.15	72.14	5.79	88.28
Proposed algorithm (d)	0.785	286.27	30				
A* algorithm (e)	4.913	327.28	392	0.25	77.41	5.37	88.78
Proposed algorithm (e)	1.110	309.71	44				
A* algorithm (f)	6.947	333.13	496	0.35	78.99	7.03	90.52
Proposed algorithm (f)	1.459	309.71	47				

different sizes and different obstacle distributions as well as the planned path length and the number of path nodes. According to the analysis in Table 1, compared with the A* algorithm and the experimental grid map, the path search time of the new algorithm is reduced by 72.14%–81.39%, the path length is shortened by 3.86%–7.03%, and the number of path nodes is reduced 86.96%–90.52%. The proposed algorithm can smooth the path to a certain extent and reduce turning points. The results show that the new path planned by the algorithm has better performance, achieving the expected effect, which proves that the proposed algorithm is effective.

Experiment

The experimental research is based on various related function packages of the Ubuntu 14.04 system and the Indigo version of the robot operating system (ROS). Experiments are carried out on the Turtlebot robot platform, YUJIN, and the proposed algorithm is finally verified. Turtlebot's hardware environment is composed mainly of a Kobuki mobile base, YUJIN, Rplidar A1 lidar is manufactured by Shanghai SLAMTEC Co., Ltd., netbook connected to Turtlebot, and workstation for establishing remote connections, as shown in Figure 12.

In the laboratory environment, the remote control workstation is connected to the netbook, and the Gmapping function package of ROS is combined with the Rplidar A1 lidar to scan the surrounding environment and generate a map model. The main parameters of A1 lidar are presented in Table 2.

Next, Turtlebot's global path planning task is completed by using Global_planner in the Move_base function package. The implementation principle block diagram of the

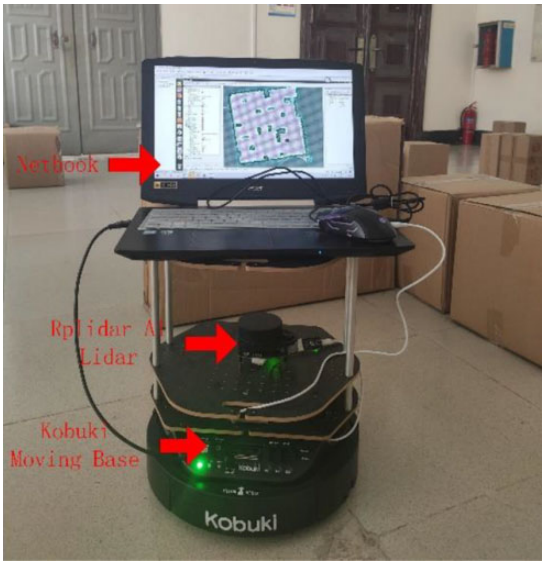

Figure 12. Turtlebot robot and its hardware.

Table 2. The main parameters of Rplidar A1 lidar.

Parameters	Performance
Ranging range	0.15–12 m
Scanning angle	0–360°
Single ranging time	0.5 s
Scanning frequency	5–10 Hz
Detection frequency	2–8 KHz

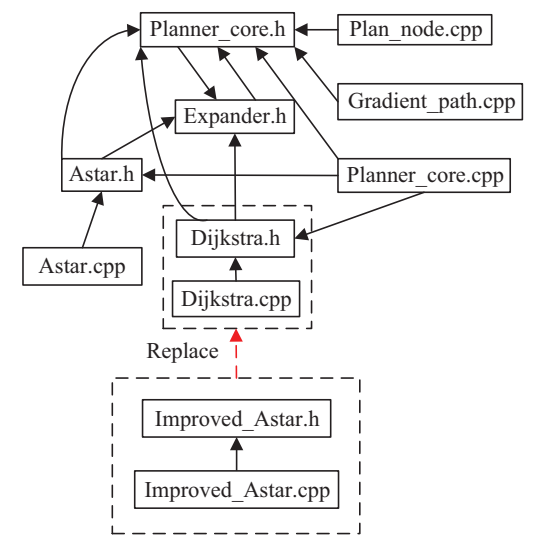

Figure 13. Global_planner framework schematic diagram.

Global_planner module is shown in Figure 13, where Planner_core.cpp and Planner_core.h are the C++ source code and header file of the global path planning algorithm. Expander.h is the header file of the node expansion algorithm. Astar.cpp and Astar.h are the C++ source code of the A* algorithm, and the header file Gradient_path.cpp is the C++ source code of the path gradient selection algorithm. Plan_node.cpp is the C++ source code of the calculation node cost algorithm, and Dijkstra.h and Dijkstra.cpp are the C++ source code and header files of the Dijkstra algorithm. To facilitate the comparative experiment, the original Dijkstra algorithm is replaced with the improved A* algorithm proposed in this article.

Figure 14. Experimental corridor.

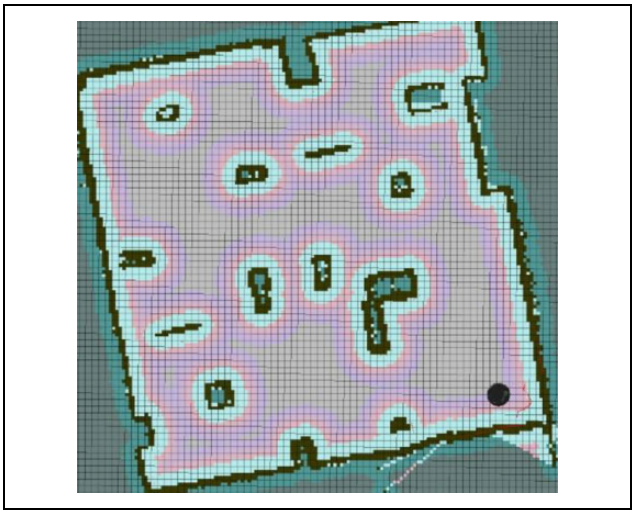

Figure 15. (a) to (c) Corridor map constructed by Rplidar A1.

In this article, part of the laboratory corridor is used as the experimental site for Turtlebot path planning, with a size of 5.5 × 6 m, as shown in Figure 14.

At the same time, the overall environment of the experimental corridor is scanned by the Rplidar A1 lidar external to Turtlebot, and the Rviz visualization tool in ROS is used to obtain and create a two-dimensional map of the experimental environment, as shown in Figure 15. Path planning

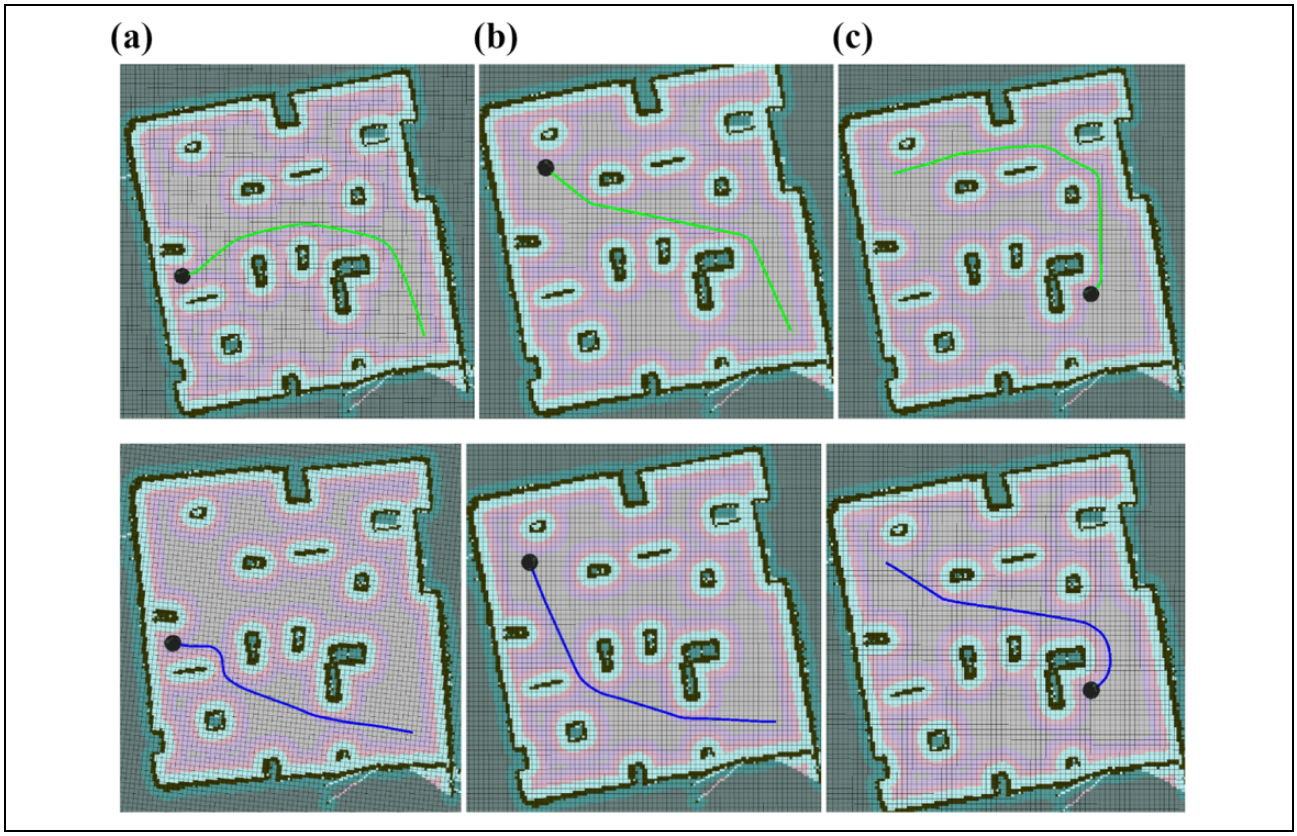

Figure 16. (a) to (c) Comparison experiment diagram of the A* algorithm and this article algorithm.

Table 3. Comparison of experimental results.

	Path length (cm)	Calculating time (s)
A* algorithm	686.140	6.785
Proposed algorithm	658.698	6.018
Optimization effect (%)	3.999	11.304

experiments are performed on the built map and the trajectory routes planned by different pathfinding algorithms under the same experimental environment are compared to verify the effectiveness of the proposed algorithm.

The path planning result of the comparative experiment is shown in Figure 16, and the grid size is 0.1×0.1 m. This picture shows the two-dimensional map and path planning results displayed in Rviz, the visualization tool of ROS. The light gray area in the figure is the two-dimensional form of the laboratory on the map, the black area is the area where the obstacle is located, the light blue area and the lavender area are the expansion areas of the obstacle, and the starting and ending points of the comparison test are the same. Figure 16(a) is the path trajectory planned by the A* algorithm, and Figure 16(b) is the path trajectory planned by the algorithm proposed for this article. Table 3 lists the comparison of the experimental results of the two algorithms in the same experimental environment. The proposed algorithm reduces the search time by 11.304% and the path length by 3.999% compared with the A* algorithm. The algorithm proposed in this article can effectively shorten the path length, reduce the time required for path planning, and reduce the risk of collision. Considering the influence of Turtlebot's own positioning error, this article believes that the improved algorithm can generate a better path than the A* algorithm.

Conclusions

This article proposes a three-neighbor search A* algorithm combined with an artificial potential field, which can quickly plan paths from a known map. First, the obstacles in the forward direction in the grid map are processed to avoid having the mobile robot enter unnecessary concave areas to search. Then, part of the artificial potential field and the three-neighbor search A* algorithm complement each other. The artificial potential field guides the mobile robot to move forward quickly in the area without obstacles. In the vicinity of obstacles, the A* algorithm of three-neighbor search is used for precise obstacle avoidance. The main contribution of this article is to process the grid map, optimize and improve the shortcomings of the traditional artificial potential field, and optimize the traditional eight-neighbor search A* algorithm. Compared with the traditional A* algorithm, the proposed algorithm has the

advantages of a smaller amount of calculation and faster calculation speed. MATLAB simulation experiments and Turtlebot mobile robot experiments verified the effectiveness of the proposed algorithm. Experimental results show that the proposed algorithm has good performance in reducing the path search time, reducing the number of search nodes, shortening the path length, and making the motion closer to the movement of mobile robots. In the future, fast path planning under unknown circumstances will also be considered.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work is supported by the National Nature Science Foundation of China [Grant Nos 61703116, 51765005, and 62073209), Guangxi Science and Technology Major Project [Grant No. AA19254021], and Guangxi Science and Technology Base and Talent Special Project [Grant No. AD19110034].

ORCID iDs

Rongxian Mo https://orcid.org/0000-0002-2505-5062
Ganwei Cai https://orcid.org/0000-0002-1936-5048
Hengyu Li https://orcid.org/0000-0002-2243-5908

References

1. Murray AT and Hong I. Assessing raster GIS approximation for Euclidean shortest path routing. *Trans Gis Tg* 2016; 20: 570–584.
2. Panov AI, Yakovlev KS, and Suvorov R. Grid path planning with deep reinforcement learning: preliminary results. *Procedia Comput Sci* 2018; 123: 347–353.
3. Hart Peter E, Nilsson Nils J, and Bertram R. A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans Syst Sci Cybern* 1968; 4: 28–29.
4. Duchoň F, Babinec A, Kajan M, et al. Path planning with modified A star algorithm for a mobile robot. *Procedia Eng* 2014; 96: 59–69.
5. Uthus DC, Riddle PJ, and Guesgen HW. Solving the traveling tournament problem with iterative-deepening A*. *J Schedul* 2012; 15: 601–614.
6. Botea A, Muller M, and Jonathan S. Near optimal hierarchical path-finding. *J Game Dev* 2004; 1: 7–28.
7. Harabor DD and Grastien A.Online graph pruning for path-finding on grid maps. In: *AAAI conference on artificial intelligence*, San Francisco, California, USA, 7–11 August 2011, pp. 1114–1119.
8. Li Y, Zhang H, Zhu H, et al. IBAS: index based A-star. *IEEE Access* 2018; 6: 11707–11715.
9. Koenig S, Likhachev M, and Furcy D. Lifelong planning A*. *Artif Intell* 2004; 155: 93–146.
10. Chabini I and Lan S. Adaptations of the A* algorithm for the computation of fastest paths in deterministic discrete-time dynamic networks. *IEEE Press* 2002; 3: 60–74.
11. Shang E, Dai B, Nie Y, et al. Guide-line and key-point based A-star path planning algorithm for autonomous land vehicles. In: *International conference on intelligent transportation systems (ITSC)*, Rhodes, Greece, 20–23 September 2020, pp. 1–7.
12. Khatib O. Real-time obstacle avoidance system for manipulators and mobile robots. *Int J Robot Res* 1986; 5: 90–98.
13. Pang M, Meng Z, Zhang W, et al. MGRO recognition algorithm-based artificial potential field for mobile robot navigation. *J Sens* 2016; 2016(2015-11–30): 1–7.
14. Agirrebeitia J, Avilés R, Bustos IFD, et al. A new APF strategy for path planning in environments with obstacles. *Mech Mach Theory* 2005; 40: 645–658.
15. Zhang J, Yan J, and Zhang P. Fixed-wing UAV formation control design with collision avoidance based on an improved artificial potential field. *IEEE Access* 2018; 6: 78342–78351.
16. Weerakoon T, Ishii K, and Nassiraei AAF. An artificial potential field based mobile robot navigation method to prevent from deadlock. *J Artif Intell Soft Comput Res* 2015; 5: 189–203.
17. Zhang T, Zhu Y, and Song J. Real-time motion planning for mobile robots by means of artificial potential field method in unknown environment. *Ind Robot* 2013; 37: 384–400.
18. Wu Z, Hu G, Feng L, et al. Collision avoidance for mobile robots based on artificial potential field and obstacle envelope modelling. *Assem Autom* 2016; 36: 318–332.
19. Raheem FA and Abdulkareem MI. Development of A* algorithm for robot path planning based on modified probabilistic road map and artificial potential field. *J Eng Sci Technol* 2020; 15: 3034–3054.
20. Pan H, Guo C, and Wang Z. Research for path planning in indoor environment based improved artificial potential field method. In: *Chinese Intelligent Automation Conference*, 2018, 273–281.
21. Wang H, Hao C, Zhang P, et al. Path planning of mobile robots based on A* algorithm and artificial potential field algorithm. *China Mech Eng* 2019; 20: 2489–2496.
22. Ju C, Luo Q, and Yan X. Path planning using artificial potential field method and A-star fusion algorithm. In: *2020 global reliability and prognostics and health management (PHM-Shanghai)*, Shanghai, China, 16–18 October 2020.
23. Nash A. *Any-angle path planning*. Los Angeles, CA: University of Southern California, 2012.
24. Hidalgo-Paniagua A, Vega-Rodríguez MA, and Ferruz J. Applying the MOVNS (multi-objective variable neighborhood search) algorithm to solve the path planning problem in mobile robotics. *Expert Syst Appl* 2016; 58: 20–35.
25. Xiong H, Guo J, and Liu L. A study of improvement of D* algorithms for mobile robot path planning in partial unknown environments. *Kybernetes* 2010; 39: 935–945.
26. Zhang H, Ming L, and Yang L. Safe path planning of mobile robot based on improved A* algorithm in complex terrains. *Algorithms* 2018; 11: 44–62.