Overview of Improvement Methods for Three Evaluation Indicators of Path Planning A * Algorithm

Xiwen Li

Department of Electricity and Intelligence of Architecture, Xi'an University of Architecture and Technology, Xian, China

panzf@puredesign.com.cn

Abstract. In the era of rapid development of industrial automation in the 21st century, robots are currently the focus of global technological development. The efficiency and safety of robot work are indicators that people evaluate their work, and most robots need to move during work, so it is necessary to study the problem of robot movement. The key to robot movement problem is route planning, and the research topic of this article is three improvement methods according to the conventional A * algorithm. This article first introduces the conventional A * algorithm, and then analyzes three evaluation indicators of path search time, amount of turning points, and turning flatness of the traditional A * algorithm. It is found that these indicators are the main factors affecting the efficiency of car search. Thus, several typical improvement methods were demonstrated, such as dynamically adjusting the cost estimation function method, dynamically adjusting the search domain method, and geometric optimization method. After mathematical modeling and formula analysis of these three methods, this article concludes that the traditional A * algorithm is not suitable for working in actual working conditions with randomly distributed obstacles. After improving the above methods, the improved A * algorithm has significant implications for path search efficiency and safety.

Keywords: A* algorithm, path planning, improvement, robot.

1. Introduction

In recent years, with the continuous progress of industrial automation, the effect of robots in industrial production is becoming increasingly important. Robots have great significance of enhancing the transportation velocity of goods, improving accuracy of goods transportation and alleviating personal pressure. Robots are widely used in plenty of high-tech-industries such as transportation, cargo distribution disaster relief and so on. And path planning is a major research topic in industrial robots. The meaning of path planning is robot find the shortest path from initial point to destination in a place with lots of random barriers [1]. Some researchers divided the question of path planning into two categories. The first one is traditional search algorithms and the second one is intelligent optimization algorithms [2]. In traditional search algorithms, it is divided into graphs based search algorithms [3], sampling based search algorithms and artificial potential field method [3]. Traditional Dijkstra algorithms and A* algorithms are categories for graph-based search algorithms, whereas RRT and PRM algorithms are the two main categories for sampling-based search algorithms. Intelligent optimization algorithms are divided into intelligent biomimetic algorithms like genetic algorithms、ant colony algorithms and deep reinforcement algorithms like DQN algorithms. the A * algorithm is often more efficient in obtaining the optimal path when the environment is known, so it is widely used in research on global path planning [4]. A popular heuristic global path search method based on state space is the classic A * algorithm [5]. The A * algorithm rasterizes the search area and simplifies it into a two-dimensional array, where each element in the two-dimensional array corresponds to each grid in the search area. The elements are divided into 1 and 0, where 1 represents the area that can be driven and 0 represents the area that cannot be driven. The path that the vehicle must take can be determined by locating the grid between the starting point and the destination.

However, traditional A * algorithms have problem of weak path smoothness processing, complex computation, and high memory consumption in practical application scenarios. This will result in the inability to achieve a high search efficiency in the path planning process [6]. So, there are lots of

researchers start to improve A* algorithms. Lin et al. combined the theory of JPS algorithm with the traditional A * algorithm and improved it by obtaining global key points, effectively solving the U-shaped trap problem of the traditional A * algorithm. The classic A * algorithm served as the foundation for Zhang et al.'s bidirectional search mechanism, which significantly increased robot path search efficiency. Jiang et al. modified the weight of the heuristic function h(n) on the basis of the traditional A * algorithm to reduce the turning points of the car. Similarly, Zhang et al. reduced the problem of excessive turning spots in the path by expanding the search Neighborhood and adopting a turning penalty function. Sun et al. combined A * algorithm with DWA algorithm to enable robots to achieve local avoidance while ensuring global optimization.

The improved A * algorithm has made varying degrees of progress in evaluation indicators such as search time, length of the path, quantity of turning spots, and smoothness of the turns. This article will summarize and compare some classic improved A * algorithms, aiming to find pain points and inspiration for future researchers.

2. Traditional A* algorithm

2.1. Environmental Modeling

There are generally three methods for environmental modeling in path planning: geometric method, grid method, and topological method [1]. The mesh method has the characteristics of being intuitive, flexible, and being prone to observe and calculate in environmental modeling, and is widely used in path planning research. As shown in Figure 1 [4], the field environment is divided into several grids with the same area. The white area represents the accessible area, and the black area represents the area where obstacles are located. Robot can move up, down, left, right, upper left, upper right, lower left, and lower right in eight directions within each square, as shown in Figure 2 [4].

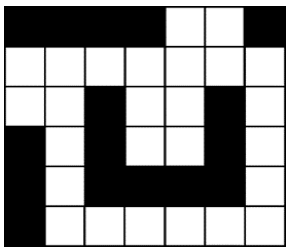


Fig. 1 Grid map [4]

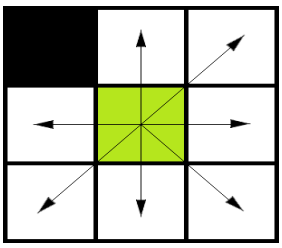


Fig. 2 Directions in which the car can pass [4]

2.2. A * Algorithm Principle Description

Combining the benefits of the Dijkstra and Greedy Best First Search algorithms, the A * algorithm is a popular classical heuristic search algorithm [7]. Finding the least expensive path from the beginning point to the destination point in a known region is the main function of the A * algorithm. A cost estimating function can be used to depict the cost of traveling from the starting point to the destination. The function is as followed:

$$f(n)=g(n)+h(n) \quad (1)$$

The cost of the journey from the starting point to the current location is denoted by g(n), and the estimated cost—also referred to as the heuristic function—from the starting point to the target point

is represented by h(n). There are other ways to value h(n), such as the Manhattan, Chebyshev, and Euclidean distances. The length of a straight line between two places is known as the Euclidean distance, while the Manhattan distance is the total of the level and perpendicular distances between two sites. The following are the expressions for the Manhattan, Euclidean, and Chebyshev distances.

The expression for Manhattan distance is:

$$h_1(n) = |x1 - x2| + |y1 - y2| \tag{2}$$

The Euclidean distance expression is:

$$h_2(n) = \sqrt{(x1 - x2)^2 + (y1 - y2)^2} \tag{3}$$

The Chebyshev distance expression is:

$$h_3(n) = \max(|x1 - x2|, |y1 - y2|) \tag{4}$$

In the formula, (x1, y1) and (x2, y2) respectively represent the coordinates of the current point and the target point. The advantages of Manhattan distance are simple calculation, no directional constraints, and high computational efficiency. The advantage of Euclidean distance is its practicality, simplicity, and ease of use. The advantage of Chebyshev distance is that it considers both straight and diagonal paths, which is suitable for problems that require the integration of multiple directional factors and has higher applicability in practical application scenarios [6]. In the execution process of traditional A * algorithm, two key data structures are first set: Open table and Close table. The Open table is mainly used to store all adjacent nodes and their child nodes of the current node in four directions (or eight directions), and record the value of the evaluation function f (n) for each node [8]. The Close table is used to store the final selected nodes, connecting all nodes in the Close table to form the final path of the car.

The traditional A * algorithm mainly uses f(n) as the evaluation criterion, and the search steps of the A * algorithm are shown in the following Figure 3:

3. Improvement of A * algorithm

As mentioned earlier, path search time, number of turning points, and smoothness of turning are important indicators for evaluating the A * algorithm. This article will introduce several commonly used solutions and demonstrate a comparison between the improved and pre improved algorithms.

3.1. Improvement of Path Search Time

As per equation (1), the A * algorithm's search velocity and results are primarily determined by the variation in h (n). The A* method prioritizes the real cost of h (n) to guarantee the shortest path when the fraction of h (n) is minimal. However, this may result in the expansion of redundant nodes, hence increasing runtime [9]. When h (n) is excessively big, the vehicle will concentrate on looking for routes close to the goal point in an attempt to locate a faster path, even though the one found might not be the best option. So in order to obtain a path with a short search time and the best possible path, the proportion of h (n) is very important. In reference [9], Tang et al. adopted an adaptive dynamic adjustment method by adding an adaptive dynamic coefficient before g (n) and h (n). The enhanced cost functions are as follows:

$$f(n)=(1-a)*g(n)+a*h(n) \tag{5}$$

$$a=1/(1+\frac{m}{L}) \tag{6}$$

In equation (6), m and L represent the Manhattan distances from the current node and the initial node to the target node, respectively. As the car continues to search, the value of m decreases and the value of a increases. This means that at the beginning of the search, g (n) should be the main focus to ensure a quick approach to the target node. To guarantee the best path selection when approaching

the destination node, the h (n) function should be utilized as the primary function. The comparison between the improved A * algorithm and the traditional A * algorithm was shown in the Table 1 [9]:

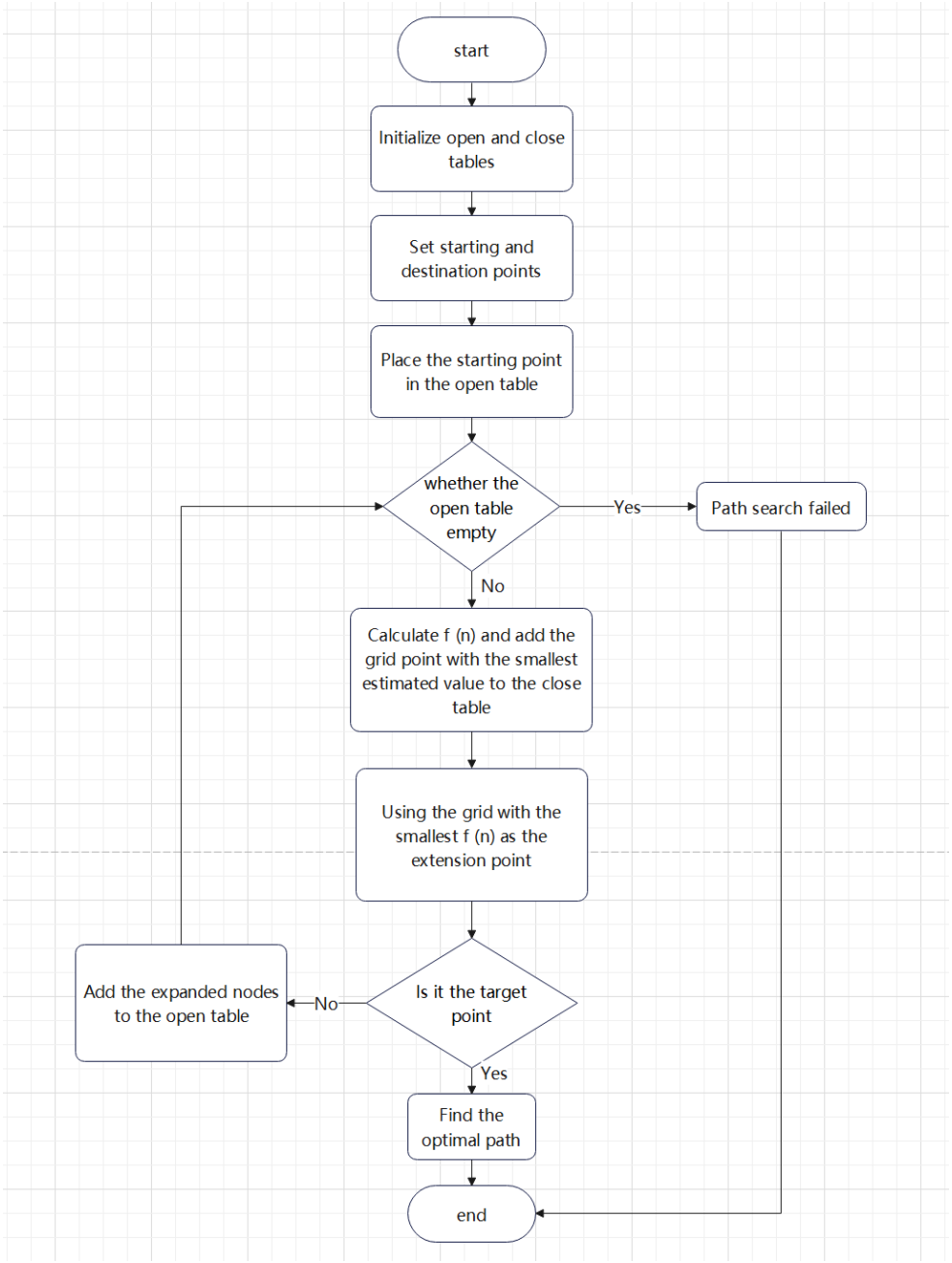


Fig. 3 Traditional A * algorithm flowchart (Photo credited: Original)

Table 1. Comparison of search performance between literature and traditional A*[9]

Scene size	algorithm	Searching time (ms)	path length	Number of traversal nodes
20*20	A * algorithm	2.01	29.8	163
	Improved algorithm[9]	0.99	30.2	78
30*30	A * algorithm	16.38	43.6	241
	Improved algorithm[9]	6.99	43.2	121
50*50	A * algorithm	17.27	74.0	634
	Improved algorithm[9]	11.81	74.0	222

From this table, It is evident that the search time has been cut in half thanks to the enhanced A * algorithm.

3.2. Improvement of The Number of Turning Points

The classic A * algorithm has some shortcomings in path planning, as it is prone to generating nodes that are independent of the final path, and the motion path has multiple turning points, lacking smoothness. In addition, the algorithm performs poorly in terms of path search accuracy, making it difficult to meet the motion requirements of mobile robots under actual working conditions [10]. The traditional A * algorithm generally uses a 4-neighborhood or 8-neighborhood search algorithm in the process of expanding nodes [11], as shown in the following Figure 4:



(a) Four neighborhood search algorithm (b) Eight neighborhood search algorithm

Fig. 4 Traditional A * algorithm searches for Neighborhood (The black area represents the search range) (Photo credited: Original)

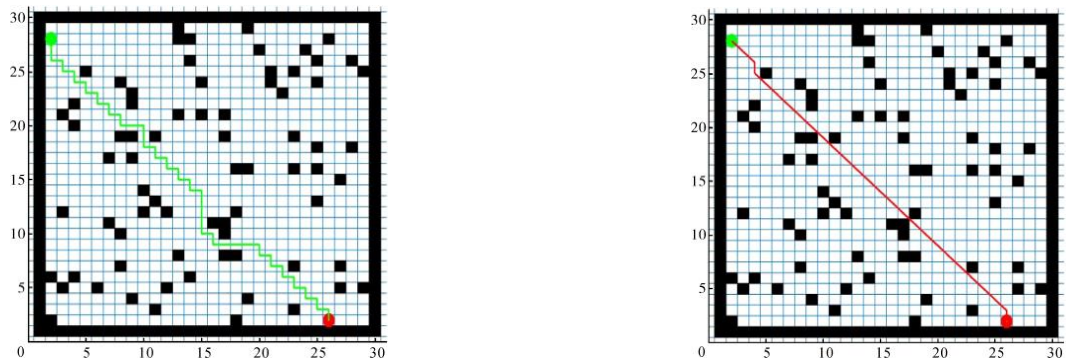
The angle between the search direction of the car in Figure (a) is 90 degrees, which can result in unnecessary turning points when the car is searching for a path. (b) The search direction angle of the car in the picture is 45 degrees, and there may be situations of driving close to or even colliding with obstacles. A real-time method for improving the neighborhood of car search was proposed in reference to address this issue [10]. The biggest difference between the method proposed in reference and traditional algorithms is that the car can switch between the four Neighborhood search algorithm and the eight neighborhood algorithm in real-time during driving [10]. The main steps are as follows:

Step 1: Mark the nodes to be expanded in eight directions of the current node of the car. If obstacles are detected, mark 1. If no obstacles are detected, mark 0.

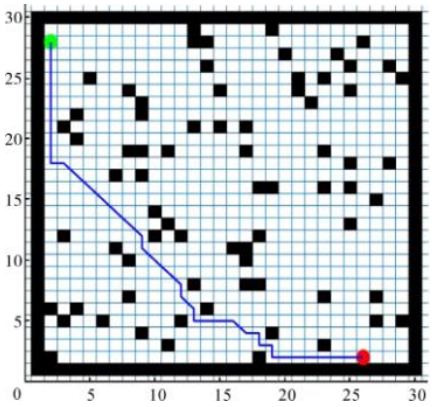
Step 2: Select a Neighborhood search algorithm by determining the tag value. When a marker value of 1 is detected, the four Neighborhood search algorithm is used, and subsequent processing is performed on the expanded node. If a marker value of 0 is detected, the eight Neighborhood search algorithm is used to calculate the f (n) value of the extended nodes in all eight directions. The node with the minimum value is selected and added to the close table.

From Figure 5(a), it can be seen that the traditional four neighborhood search algorithm has many inflection points in the path and poor smoothness of the road. Figure 5(b): Although the traditional eight neighborhood search algorithm has few inflection points and high road smoothness, it is close to obstacles and even passes through the middle of two obstacles, which is not suitable for practical application. Figure 5(c) uses a dynamic switching search algorithm to obtain a satisfactory search path, reducing the number of inflection points while also achieving better path smoothness.

The following are the simulation results from reference [10]:



(a) Four Neighborhood search algorithm (b) Eight Neighborhood search algorithm



(c) Improved algorithm simulation diagram

Fig. 5 Comparison diagram between the improved algorithm and the traditional algorithm in terms of path [10, 12]

3.3. Improvement of Turning Smoothness

Due to the inability of the improved A * algorithm function to abruptly change its mechanical states such as speed and acceleration during turns, the car must decelerate and accelerate during sharp turns, thereby increasing movement time and energy consumption [11]. Smoothing the path's corners is required to guarantee the robot moves smoothly when in motion [12]. The commonly used methods to improve turning smoothness include Bezier curve method, interpolation method, etc. This article introduces an improved method with low computational complexity and simple understanding. As shown in Figure 6:

The path taken by the car when turning is NA+AM. Assuming the coordinates of point N are (x1, y1), point M are (x2, y2), point A is (x, y), and point B is (a, b), $\theta+\alpha=\pi$. The specific calculation formula is as follows:

The length of NA:

$$\sqrt{(x1-x)^2+(y1-y)^2} \tag{7}$$

The length of AM:

$$\sqrt{(x2-x)^2+(y2-y)^2} \tag{8}$$

The length of AM+NA:

$$\sqrt{(x1-x)^2+(y1-y)^2}+\sqrt{(x2-x)^2+(y2-y)^2} \tag{9}$$

The length of radius R:

$$R=\frac{\sqrt{(x1-x)^2+(y1-y)^2}}{\tan(\frac{\alpha}{2})} \tag{10}$$

The length of arc NM:

$$L=2\pi R\theta=2\pi\frac{\sqrt{(x1-x)^2+(y1-y)^2}}{\tan(\frac{\alpha}{2})}\theta \tag{11}$$

Comparing the arc length NM and NA+AM, it can be concluded that the value of arc length NM is smaller, and the improved path of the car is significantly shortened at the turning point.

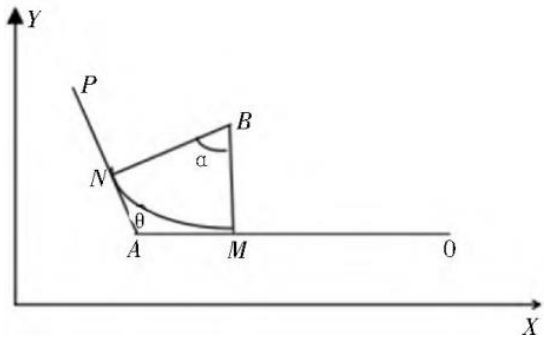


Fig. 6 Mathematical model at inflection point [13]

4. Discussions

4.1. Improvement Suggestions for Long Search Time of Car Paths

Changing the classic A * algorithm's cost estimation function is essential to get a car with high path search efficiency. The weights of g(n) and h(n) in the cost estimate function of the conventional A * algorithm are constant throughout the search. However, the relative importance of g(n) and h(n) may change during the search process itself. That is to say, the automobile must concurrently examine both to the same amount because the typical A * algorithm ignores the difference in relevance between g(n) and h(n). Consequently, this naturally causes the car's path search speed to drop. By changing the different weights of g(n) and h(n), the A * algorithm can be more biased towards one side. If the weight of the moving cost is too high, path planning will become slower, but the optimal path can still be found; If the weight of the estimated cost is too large, the path planning speed is fast but it cannot guarantee finding the best path [14]. So it is very important to change the proportion of h(n) and g(n) in the A * algorithm. However, the value of g(n) remains constant, so improvements to the cost estimation function are all focused on h(n). For example, references and both used similar adaptive dynamic coefficient methods [10, 14], with the specific formula as follows:

Improved cost estimation function in reference [10]:

$$f(n) = g(n) + w*h(n) , w \geq 1 . \quad (12)$$

The cost estimation function used in reference [14]:

$$f(n)=g(n)+(1+r/R)*h(n) \quad (13)$$

The above literature has achieved faster path search speed after improvement. From this, it can be concluded that dynamically changing the weight of h (n) can make the car more inclined towards h (n) or g (n) at different time periods, allowing the car to constantly balance between path planning speed and finding the best path, thereby reducing the time for path search.

4.2. Improvement Suggestions for Excessive Number of Inflection Points

The phenomenon of cars generating too many turning points during the search process is related to the search domain of the algorithm itself. This article suggests that addressing the issue of a large number of inflection points can be achieved by changing the field of car search. This is because the traditional A * algorithm expands nodes in a certain direction and adds them to the close table in a certain order, which means that under the condition of randomly distributed obstacles, the car is prone to making S-shaped turns, as shown in the following Figure 7:



Fig. 7 Schematic diagram of S-turn generated by the car (Photo credited: Original)

When the car travels from A to B, it will follow the yellow path above. However, when the car goes from point B to point A, it returns to the lower part and walks slightly. As the search area of the car is mostly in 4 or 8 fields, the car is not aware of the existence of obstacles below. Therefore, the process of walking downwards will create extra turning points. So expanding the search field appropriately can not only improve search efficiency, but also remove excess child nodes [7]. For example, literature [7] and literature [12] mentioned earlier. Reference [7] adopts a 5 * 5 search domain and improves the method of expanding nodes. Reference [7] uses a 5 * 5 search domain and if obstacles are detected during the search process, the evaluation function of the node is set to infinity, adjusting and expanding nodes in other directions. However, reference [12] adopted a method of dynamically adjusting the search domain of the car. Dynamically switch the search range between 4 and 8 domains. When obstacles are detected, use the 4 domain search algorithm, and when no obstacles are detected, use the 8 domain search algorithm. The specific steps are described in the previous text. So in order to reduce the turning points in the car search process and improve search efficiency, it is necessary to improve the search field.

4.3. Suggestions for Improving Turning Smoothness

This article believes that we can improve the smoothness of turning by using the Bessel curve or inscribed circle method, and the improvement of smoothness is necessary. For the traditional A * algorithm, the car can only take turns during the turning process, not curves. This leads to longer search times and increased energy consumption for the car, and in some practical working conditions, the car is required to turn smoothly. Both references and used geometric model optimization methods [13, 15], while references and used the Bessel curve method [6, 11], which greatly improved the smoothness of the car at turns. As shown in the following Figure 8.

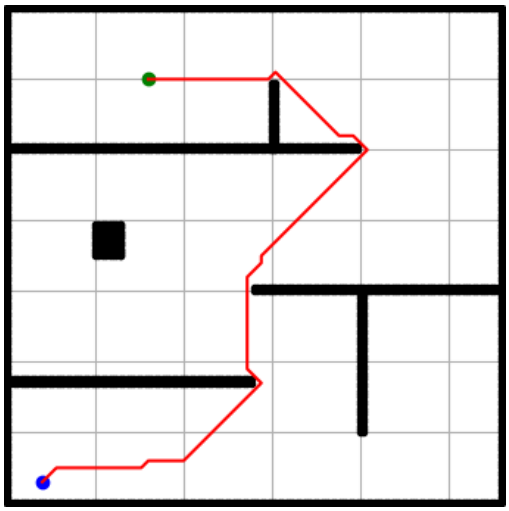


Fig. 8 Trajectory of Traditional A * [6]

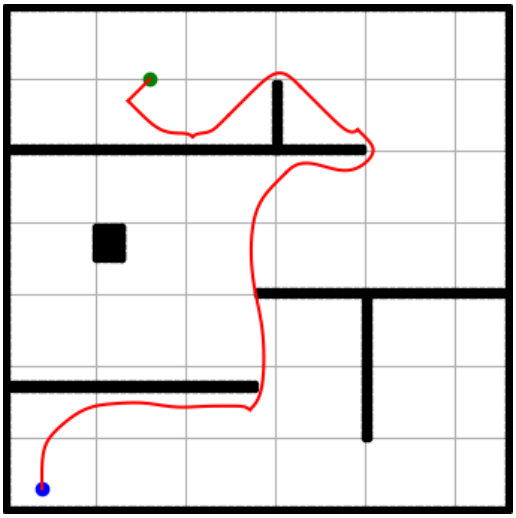


Fig. 9 The improved trajectory in reference [6]

From the Figure 9, it can be seen that the curve processed by turning smoothness increases the distance between the car and obstacles at the turning point, resulting in an improved safety factor for the car in actual working environments.

5. Conclusion

This article is based on the three shortcomings of the traditional A * algorithm, and introduces the improvement methods of current researchers for the three shortcomings of the traditional A * algorithm: long search time, large number of turning points, and poor turning smoothness. And a detailed introduction was given to the adaptive dynamic coefficient method for solving the cost estimation function with long path search time, the dynamic switching search domain method for solving the problem of too many turning points, and the geometric optimization method for solving the problem of poor turning smoothness. Therefore, this article finds that in the working condition where obstacles are randomly distributed, traditional A * algorithms are difficult to meet actual production needs, which can lead to many problems, such as crossing between two adjacent obstacles, turning against obstacles, and not being able to find the optimal path. Therefore, this article's primary contribution consists of analyzing and improving the three flaws in the conventional A * algorithm while also offering recommendations and techniques for future study. The shortcomings of this article are that it does not analyze all the shortcomings of traditional A * algorithms, such as the typical U-shaped trap problem and the problem of easily falling into local optima. It only provides a detailed demonstration of several typical improvement methods and does not systematically demonstrate the combination of traditional A * algorithms with other algorithms, such as JPS, DWA, Morphin, etc. In subsequent research, these three parts should be systematically supplemented.

References

[1] Jiang Chengjie, Zhu Hua, Xie Yao, et al. Path planning for mobile robots based on improved A * algorithm [J/OL]. Manufacturing Technology and Machine Tool, 2024: 1-5.

[2] Li Gang, Zhi Hongxin. Overview of Path Planning Methods for Electric Power Inspection Robots [J/OL]. Electric Power Science and Engineering, 2024: 1-10.

[3] Feng Ao, Yang Jinyu, Xie Yuyang, et al. Robot path planning based on improved PRM algorithm [J]. Computer Technology and Development, 2024, 34 (02): 127-133.

[4] Lin Guijuan, Li Zihan, Wang Yu. Research on Improved A * Algorithm for Global Path Planning Based on Global Key Point Extraction [J/OL]. Journal of System Simulation, 2024: 1-12 .

[5] Weng Xing Global path planning algorithm and experimental research for wheeled intelligent vehicles [D]. Hebei University of Technology, 2019.

[6] Zhang Xuefeng, Hu Weipeng, Shi Junhua, et al. Research on path planning for mobile robots based on improved bidirectional A * algorithm [J/OL]. Journal of Chongqing University of Technology (Natural Science Edition), 2024: 1-13.

[7] Zhao Yunlong, Wang Xinjie, Cheng Kui et al. A Fusion Improved A * Algorithm for AGV Path Planning [J]. Journal of Sichuan University of Light Industry and Chemical Technology (Natural Science Edition), 2024, 37(01): 71-78.

[8] Zhao Jiang, Zhang Yan, Ma Zewen, et al. Improvement and validation of the A-star algorithm for AGV path planning [J]. Computer Engineering and Applications, 2018, 54 (21): 217-223.

[9] Tang Yaling, Mu Ruru, Zhang Xuefeng, et al. Research on Path Planning of Rescue Robots Based on Improved A * Algorithm [J/OL]. Journal of Chongqing University of Technology (Natural Science Edition), 2024: 1-8.

[10] Chen Jiabao, Wen Jiayan, Xie Guangming. Mobile robot path planning based on improved A~* algorithm [J]. Journal of Guangxi University of Science and Technology, 2022, 33(01): 78-84.

[11] Xu Jishuang, Jiao Jun, Li Miao et al. A Path Planning and Motion Control Method for Disease and Death Livestock Transport Robots by Integrating Improved A~* Algorithm and Fuzzy PID [J]. Smart Agriculture (Chinese and English), 2023, 5(04): 127-136.

[12] Xu Wei, Tang Junwei, Zhang Chi. Path planning for mobile robots using improved A~* and dynamic window algorithms [J]. Computer Simulation, 2023, 40 (03): 447-452.

[13] Huang Chenxi, Han Tao, Wu Xueqiong, et al. Robot path planning and obstacle avoidance technology based on improved A~* algorithm [J]. Electronic Design Engineering, 2024, 32 (01): 24-28.

[14] Su Chang, Yuan Quan, Li Ziqiang, et al. Research on Improved A * Algorithm in Path Planning of Spray Robots [J]. Journal of Heilongjiang University of Technology (Comprehensive Edition), 2024, 24 (01): 145-150.

[15] Lu Wanlin, Lei Jingsen, Shao Yan. Path planning for mobile robots based on improved A~* algorithm [J]. Journal of Ordnance Equipment Engineering, 2019, 40(04): 197-201.