Acta Technica 62 No. 1A/2017, 401–410

Application and simulation analysis of C++ in sweep robot control program

Xiaorong You[1,2], Hao Pei[1]

Abstract. The purpose of this study is to further optimize the planning path, to improve the existing D*Lite algorithm. The reverse search connection path optimization method is used, and the existing D*Lite algorithm in the steering angle of 45 integer multiple limit is canceled. The smooth path of the planning process is to reduce the ground robot movement process of the number of steering and path length. And through simulation to verify the effectiveness of the optimized D*Lite algorithm. At the end, in the Visual C++ environment, the path planning of optimized D*Lite algorithm is programmed on the environment map. The multiple obstacles are added on the environment map, and the path planning capability of the optimized D*Lite algorithm is verified in the complicated environment. The results show that the security improvement strategy and path optimization method can improve the security and path optimization. Therefore, we concluded that the optimized D*Lite algorithm has advantages in ensuring the safety of the sweeping robot and reducing the length of the path.

Key words. C++, sweeping robot, D*Lite algorithm, path optimization.

1. Introduction

With the gradual integration of intelligent appliances into the family life, the application of the ground mobile robot in the life gradually increases, which intelligent sweeping robot is widely used in the home [1]. As a service robot, sweeping robot can replace the traditional manual cleaning work, and the market prospect is broad [2]. At present, domestic sweeping robot rarely has navigation and positioning function, and it lacks effective path planning [3]. It is cheap but is characterized by inefficient cleaning. How to make the sweeping robot moves safely and reliably in a dynamic environment and quickly calculates the shortest path has become an important issue in sweeping robot research.

At present, many researchers in the world study the path planning of sweeping robot. The D*Lite algorithm is an optimized version of the LPA* algorithm [4]. The

[1]Department of Electrical and Mechanical Engineering, Changzhou Textile Garment Institute, No. 5 Gehu Road Changzhou, Jiangsu, 213164, China

[2]Corresponding author

search direction is changed from the target position to the current robot position. And it is more suitable for the robot dynamic path planning in the process of moving [5]. Based on the study of existing D*Lite algorithm, two kinds of space can be improved, including the security improvement of path planning and the optimization of path. D*Lite algorithm uses search method of eight grid, the planned path angle is an integer multiple of 45°, and it is not suitable for the practical application of the sweeping robot. This paper gives the detection and optimization methods, the optimized path reduces the number of steering and the distance of the robot. And the steering angle can be any angle. Based on the existing grid map, the characteristics of the simulated laser scanning range finder are programmed in the Visual C++ environment. And the path calculation of the optimized D*Lite algorithm is completed. Based on the path calculated in Visual C++ on the sweeping robot, the actual verification of path planning is completed.

2. Materials and methods

2.1. Sweeping robot path planning algorithm

The Sweeping robot path planning algorithm was a safe optimal path from the current location to the target location in the environment map. If the environment was all known, and the location and size of the obstacle information were known, the path planning algorithm could be used to calculate the shortest path and avoid the optimal path, which was the global path planning. If only part of the environmental information was known, or the environment was not known, at this time, the robot configuration sensor only could be relied to perceive the surrounding environment. In the process of walking the robot, the detection of obstacles to avoid obstacles is the local path planning.

The first one is the A* path planning algorithm principle. A* algorithm is a heuristic search algorithm, using the evaluation function $f(n)$ to analyze the current location to the destination location of the path. Priority search of the highest-valued path nodes, in rare cases, it needed to search the entire environment map in order to get the best path. A* algorithm path assessment function was as follows: $f(n) = g(n) + h(n)$, where $f(n)$ evaluates whether the position node n reaches the destination, $g(n)$ is the actual value required from the starting point to the position n and $h(n)$ is the mobile cost estimate of position n to the target location.

The second one is the LPA* algorithm that was used to maintain the three parameters, including $g(s)$, $\mathrm{Rhs}(s)$ and $h(s)$ values in path planning. Here, $g(s)$ is the distance from node s to the starting position. And this is equal to G value in the A* algorithm. Symbol $\mathrm{Rhs}(s)$ is the estimated value of the parent node of the current node. And the $g(s)$ value is assigned to the node when the grid $\mathrm{Rhs}(s)$ value is calculated. Symbol $h(s)$ is the same as the H value in A* and it represents the estimated value of the current node to the target position. The $\mathrm{Rhs}(s)$ value is a new variable introduced by the LPA* algorithm, which is calculated as follows. If

Pred(s) is the parent node of node s, then

$$\mathrm{Rhs}(s) = \min[\mathrm{Pred}(s) + 1]\,.$$

The third is the D*Lite algorithm, whose implementation process is shown in Fig. 1.

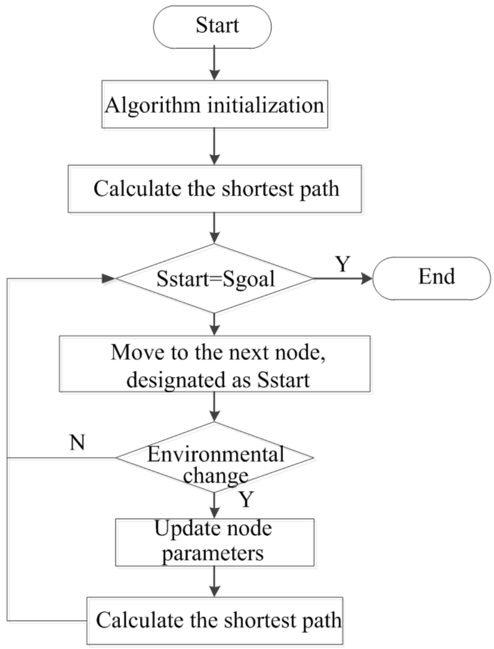

Fig. 1. Path planning flow chart of D*Lite

2.2. *Optimized D*Lite algorithm*

After analysis on the existing D*Lite algorithm, there are two main problems about the algorithm. Firstly, the algorithm for robot safety considerations were not perfect, and the algorithm of the path of the results had the existence of security risks. Secondly, the path of the D*Lite plan had too many twists and turns, and it did not optimize and smooth the planned path. In addition, the limit of the robot path was an integer multiple of 45 ° and did not conform to the actual motion characteristics of the robot.

2.2.1. Path optimization method. In the raster map, there were complex obstacles, such as concave obstacles. The first path search entered the concave barrier internal search path, finally, the conclusion could be gotten that this road cannot pass. In order to reduce the number of grids that were unnecessary to enter the recesses and reduce the number of path searches, the temporary barrier grid mark was used to avoid unnecessary raster search. Path optimization used a strategy that was similar to D*Lite to optimize the shortest path that had been planned from the target grid to the starting grid. The steps were as follows.

First, the target location grid and the following two path nodes were analyzed. And the three nodes were analysed to check whether the two paths of the three nodes can be optimized as a path and check the security path rules. Second, if the new path met the security path rule described in the previous section, then the two paths in the three nodes were changed to two nodes and one path. And the two nodes were kept and the next node was added to continue checking whether three new nodes and two paths could be optimized. Third, if the new path did not conform to the path safety rule, therefore, there was an obstacle ridge and the barrier grid binding position. Then the optimization of the three shortest path nodes was completed. The first joined node was removed (the path node closest to the target grid) and the next node was added. Fourth, the steps 2 and step 3 were carried out circularly to check whether there was an optimized path until the originating node was reached. Finally, when the path changed, from the path change node to the surrounding affected by the path of the node could be carried out the search and optimization of the optimized path until the need to pick up the need to optimize the path so far.

The reverse path was the "path straightened" within the visible range of the path, it aimed at reducing the number of turns and the length of the path. Before the path fusion, firstly determine whether the fusion path to meet the safety rules to ensure the safety of the path. The direction selection of the path fusion was consistent with the search direction of the path. That was the searching fusion from the target position to the robot's current position. Therefore, in the process of sweeping robots was close to the target, the need to merge and update the path was decreasing. At the same time, path fusion was the path optimization based on the existing D*Lite algorithm, which can guarantee the optimality and correctness of the path planning results.

*2.2.2. Optimized D*Lite search process.* The optimized D*Lite algorithm was roughly consistent with the D*Lite algorithm [6]. The optimized D*Lite algorithm checked for unsecured paths during path calculations and improved to a secure path. After completing the shortest path planning, the reverse search link optimization was made to paths, so that the path length reduced, the path smoothness improved and the number of steering purposes reduced. The complete algorithm was executed as follows.

First, the $h(s)$ value for all rasters was initialized and the target node was placed in the queue. The $h(\text{start})$ value of starting position is 0, and the starting position was incremented by one for each of the eight rasters of the surrounding grid. From one grid to another, the vertical or diagonal grid $h(s)$ value difference was 1. Until the full value of the entire raster was initialized. Initialized all grid $\text{Rhs}(s)$ values and $g(s)$ values were infinite. The target node was placed in the priority queue. The second was to calculate the shortest path. The target node became the first locally discontinuous node. From the target node, the $k(s)$ parameter of the eight rasters around was calculated, and $\text{Rhs}(s)$ was assigned to $g(s)$ if $g(s)$ was greater than Rhs(s). Before choosing the next grid to be extended, it was firstly determined whether there was a problem with the two safety path rules that violated the combination of the barrier through the barrier grid and the sharp corners passing through the obstacle grid. In addition, the grid with the smallest $k(s)$ was selected from the priority queue again as the grid to be expanded. And it was judged whether the new grid to be expanded conformed to the security path rule. The node extension that had the smallest $k(s)$ and conformed to the safe path rule was repeatedly selected until the Rhs(start) was equal to the g(start). The third was the path generating. It was moved from the current position to the grid with the smallest $g(s') + c(s', s\text{start})$ value, and s' is Succ(s). The fourth was the path optimization. From the target position, each time the three consecutive nodes were performed to judge the path optimization, and the security judgments was made to optimized path. If obeyed the path security rules, then the path optimization was given up. The first access to the path node was removed and the next path node was added until it was optimized to the originating node. The last was to check the environmental changes. During the movement of the swept robotic robot, the change of the raster map was detected, the $k(s)$ of the changing grid was updated, and the raster recalculation parameter was changed by the raster map change. Then according to the steps 3 to find the new path light and following the step 4 to complete the path optimization.

*2.2.3. Optimized D*Lite algorithm simulation.* In order to make a clear comparison between the two aspects of security improvement and path optimization for the optimized D*Lite algorithm, the results of the improved D*Lite algorithm for path planning security improvement were firstly verified, see Table 1. And then based on the result of path planning that was totally improved, the path optimization was carried out.

Table 1. Comparison of the security improved D*Lite algorithm and the original algorithm planning path

Evaluation parameters	Existing D*Lite algorithm	D*Lite algorithm with security strategy
The number of turns	8	10
Steering angle (°)	495	675
Path length (m)	15.6	21.6

The fully improved path avoided the security issues mentioned above, but the path length and the number of turns were increased while the security path was improved. According to the characteristic of robot in the practical application that it can complete any steering angle, based on the path security, the reverse connection path fusion method was used. Starting from the target node, each time three nodes were selected to determine whether they can be merged into one path. After merging, the three path nodes will be reduced to two nodes, and the intermediate nodes were removed. If the original two paths were folded, then the path after the fusion will no longer be reduced, which reduced the number of paths and the path length of the path. The robot walked in accordance with the fusion path after the degree of turning, it was no longer limited to the integer multiple of the 45 °. The path evaluation parameters after the execution path optimization with the existing D*Lite algorithm and the security improved D*Lite algorithm were compared in Table 2.

Table 2. Optimized D*Lite algorithm and existing algorithm and path comparison results using security policy

Evaluation parameters	Existing D*Lite algorithm	D*Lite algorithm with security strategy	Path-optimized D*Lite algorithm
The number of turns	8	10	2
Steering angle (°)	495	675	169
Path length (m)	15.6	21.6	15

The final optimization D*Lite path planning algorithm used a security strategy while the reverse search path optimization was performed on the path results. The data in table 2 showed that compared with the existing D*Lite algorithm, the optimized D*Lite algorithm had fewer steering and steering angles. Especially, when the number of turns was reduced to only twice and the steering angle was reduced to less than half of the algorithm. The path length was required to be bypassed by the use of a safety-improved path in the position where the position of the obstruction combination and the sharp corners of the obstacle. So that the path was not significantly reduced, and it was slightly lower than the path length of the original algorithm. In practice, the length of the walking path of the sweeping robot, the steering angle and the number of turns affected the length and energy consumption of the sweeping robot to the target. Therefore, the optimized D*Lite algorithm had an effective path optimization for the existing D*Lite algorithm aiming at the practical characteristics of sweeping robots [7].

2.3. Platform introduction

The laser scanning range finder was added into the basic hardware platform, which was in-depth study to the environment modeling, path planning and other directions. In the sweeping robot path planning research, the sensor system was one of the core [8]. The verification platform used a notebook as the center of information collection, processing and control. The lower computer was mainly responsible for driving the motor movement, collecting infrared proximity sensors and ultrasonic range finder data. Algorithm validation platform PC used a laptop computer to install the windows operating system and Vissual C++ 2008 development environment. The laptop was connected directly to the laser range finder via the USB port. The lower unit was equipped with six ultrasonic and infrared sensors and gyroscopes, acceleration sensors. In order to ensure the safety of the robot body, the grid map expansion amount was about the same as the length of the robot body and the larger value in the width of the robot. In the process of the path planning environment map construction, the size of the robot was the basis for the obstacle expansion of the grid map. In order to protect the safety of the robot body, the grid map expansion was about half the value of the robot body length and the larger width. In this paper, after installing the laser range finder at the front of the robot, the length was about 460 mm and the width was about 450 mm.

*2.4. Experimental results of the optimized D*Lite algorithm*

The experiment used the optimized D*Lite algorithm to do the final verification work of the path planning on the robot platform. First, according to the laboratory environment to draw the grid map, the data was acquired in Vissual by programming simulation laser scanning range finder, the optimized D*Lite algorithm in the drawing of the grid map to calculate the shortest path from the starting point to the target location. And then the path planning results were imported into the sweeping robot to test the actual walking effect of the optimized D*Lite algorithm.

2.4.1. Experimental grid map. The laboratory environment was divided into a number of small squares, according to the body size 48 cm of the length and 46 cm of the width, the square grid that the length and width were 10cm was selected. The obstruction will extend to the surroundings by two grids that cannot pass through the grid. It was important to prevent the robot from moving and obstruct the collision, and the robot was narrowed to occupy a grid.

*2.4.2. Optimize the running result of the D*Lite algorithm in the raster map.* It can be seen from the experiment that after the acquisition of the environment map became smaller, the path turning point that was calculated by the algorithm became more and more. It can be seen that the amount of environmental information that was obtained by the laser sensor that was installed by the sweeping robot had a great influence on the path planning process during the movement of the sweeping robot. When the sweeping robot moved to the target position, the laser range finder constantly detected the new environmental information to join the raster map. When the obstacle appeared on the path to be walking, the path planning algorithm updated the parameters and the calculation of the raster node. The effective maximum measurement distance of the laser sensor used in this experiment was 4095 mm, the high measurement accuracy could be gotten in the effective measurement range. When the laser scanning range finder used the maximum measurement distance, the amount of environmental information was obtained. And the path planning had less steering times and path lengths, and the path planning results were better. At the same time, in the experiment, when the obstacle was extended, the margin of the ground robot was set aside, and the unsafe path was improved accordingly. In order to obtain better path planning results, the next test used the laser scanning rangefinder to measure the maximum distance measurement of environmental information. The environment map in the experiment was the path planning under the condition that the global raster map was known. The path planning algorithm was relatively simple to complete the path planning on the known raster map. The ground moving robot traveling in the practical application environment can only rely on the distance data measured by the laser sensor. And the data measured by the laser sensor was the obstacle to face the data of the front of the laser sensor and cannot obtain the appearance dimension of the obstacle. The result of the shortest path was based on the acquisition of the global environment map. Therefore, the laser scanning range finder will affect the planning result of the environment. The self-localization of the sweeping robot was also a problem in practical application.

3. Results

In the known raster map, the starting position and target position of the ground robot was specified. The optimized D*Lite algorithm calculated the shortest path from the start point to the end point. Because the detection environment was large, the most of the obstacle information could be gotten during the movement. The path that was planned by optimized D*Lite algorithm was close to the shortest path, the number of path steering was only three times, and the path smoothness was high. Path planning details was in Fig. 2. When the path planning algorithm detected that there is an obstacle on one of the two sides of the next node in the process of going to the target, and the optimized algorithm changed the path from the neighboring grid.

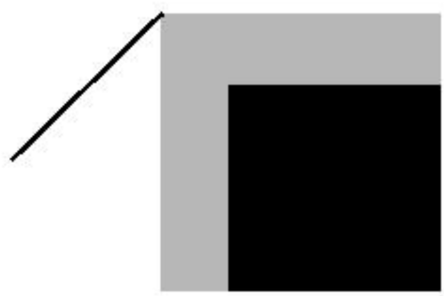

Fig. 2. Optimized D*Lite algorithm for planning the security path

On the basis of the map in Fig. 2, the raster map of the barrier joint point was added, and the algorithm of the optimization of the barrier grid was verified. The calculated path is shown in Fig. 3. The optimized algorithm regards the barrier of the barrier grid as non-accessible when the optimized D*Lite algorithm detects that there is the barrier junction when it moves to the next position.

Simple grid map of the algorithm validation was not very full, the actual environment had more obstacles. Then the path of the planning employed more practical significances. Figure 4 is the map after adding obstacles.

When the ground robot moved to the target position, the laser range finder constantly detected the new environment information to join the grid map. When the obstacle appeared on the path to be walking, the path planning algorithm updated

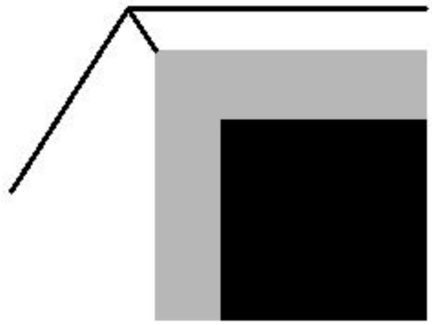

Fig. 3. Optimized D*Lite algorithm for planning a secure path (adding a barrier junction)

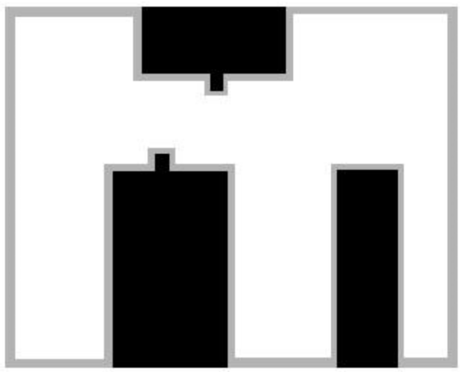

Fig. 4. Map after adding obstacles

the parameters and calculation of the raster node. From the experiment, it can be seen that the optimized D*Lite algorithm can still be used to plan a feasible shortest path in the chaotic environment.

The optimization of the D*Lite algorithm was carried out by using the optimized D*Lite algorithm to simulate the data acquisition characteristics of the laser scanning range finder on the grid map with the global environment. A new obstacle was added to the map to verify the effect of optimizing the D*Lite algorithm on path planning on complex barrier maps. In the simulation, the amount of environmental information of the laser scanning rangefinder was adjusted, and the comparison result showed that the path calculation algorithm of the sweeping robot was shorter and smoother. Because the limitation of the environment to obtain information and the accuracy of the robot self-positioning were not high, so the mapping laboratory environment map was used. The simulation results of the algorithm were simulated in Visual C++, and the optimized D*Lite algorithm was simulated. The route was imported into the sweeping robot to verify the safety and optimization of the moving route from the departure location to the target location. The experimental results showed that the improved strategy and path optimization method can improve the path security and path optimization.

4. Conclusion

In this paper, the path planning algorithm in unknown environment is studied, and the improvement of path planning security and path optimization are made. Based on the existing D*Lite algorithm, the total improvement and path optimization are carried out. The experimental results show that the total improvement strategy and the path optimization method can be used to improve the path.

It can also be concluded that the optimized D*Lite algorithm can calculate the optimal path from the current position to the target location according to the known environmental information. The simulation results show that the path of the algorithm has higher security. The path length, the number of steering and the steering angle are reduced and the smoothness of the path is improved. At the same time, the optimized D*Lite algorithm has advantages in ensuring the safety of the sweeping robot and reducing the length of the path

References

[1] W. Zhong, X. Tao, J. Lu: *Study of key technology for pipeline cleaning robot.* Modular Machine Tool & Automatic Manufacturing Technique (2013), No. 4.

[2] M. M. Bian, L. J. Zhang, L. L. Bian: *Design of mobile robot remote control.* Advanced Materials Research *1030–1032* (2014), Chapter 6, 1453–1456.

[3] A. Ulusoy, S. L. Smith, X. C. Ding, C. Belta, D. Rus: *Optimality and robustness in multi-robot path planning with temporal logic constraints.* International Journal of Robotics Research *32* (2013), No. 8, 889–911.

[4] H. Lee, A. Banerjee: *Intelligent scheduling and motion control for household vacuum cleaning robot system using simulation based optimization.* Proc. IEEE Winter Simulation Conference (WSC), 6–9 Dec. 2015, Huntington Beach, CA, USA, IEEE Conference Publications (2015), 1163–1171.

[5] Y. Zhang, D. W. Gong, J. H. Zhang: *Robot path planning in uncertain environment using multi-objective particle swarm optimization.* Neurocomputing *103* (2013), 172 to 185.

[6] L. Yang, J. Qi, D. Song, Y. Xia: *Survey of robot 3D path planning algorithms.* Journal of Control Science and Engineering (2016), No. 1, 1–22.

[7] M. R. K. Ryan: *Exploiting subgraph structure in multi-robot path planning.* Journal of Artificial Intelligence Research *31* (2008), No. 1, 497–542.

[8] X. Wang, Y. Shi, D. Ding, X. Gu: *Double global optimum genetic algorithm–particle swarm optimization-based welding robot path planning.* Engineering Optimization *48* (2016), No. 2, 299–316.

Received June 6, 2017