This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3021073, IEEE Access

Date of current version August 30, 2020.

Research and Optimization of D-Start Lite Algorithm in Track Planning

KAILI XIE, JIE QIANG, AND HAITAO YANG

College of Aerospace Information, Space Engineering University; Beijing 101416 , China;

Corresponding author: Haitao Yang (e-mail: yanghtt@126.com).

ABSTRACT With the widespread application of the aircraft, how to achieve the safety and efficiency of the aircraft during flight has attracted widespread attention. The track planned by the traditional D-Start Lite algorithm is very close to the obstacle. Due to the large size of the aircraft itself, it is very likely that the aircraft will collide with the obstacle when operating in a real environment. However, the existing D-Start Lite algorithm is only a single solution to the safety problem of trajectory planning or to improve the efficiency of the algorithm ' s trajectory search. Therefore, this article optimizes the existing D-Start Lite algorithm. First, this paper adopts the strategy of selecting priority level by child nodes. The child nodes in the vertical direction are selected as the first-level nodes. The state of the first-level node determines whether the second-level node on the diagonal needs to be expanded, so that a safe distance is generated between the planned path and the obstacle, thus improving the safety of the path. However, a certain space is reserved between the track and the obstacle, which results in the increase of the number of nodes in the path and the low efficiency of search. This paper improves and optimizes the valuation function and heuristic function in the algorithm. Therefore, the number of expansion nodes and the number of inflection points are reduced, and the search efficiency of the algorithm is improved. Therefore, the improved algorithm improves the efficiency of trajectory search while ensuring the safety of trajectory planning. And the effectiveness and feasibility of the algorithm are proved by the MATLAB platform.

INDEX TERMS D-Start Lite algorithm, Heuristic function, Selection of child nodes, Trajectory planning, Valuation function.

I. Introduction

In recent years, many achievements have been made in track planning[1]. Currently, the trajectory planning is mainly divided into two parts: environment modeling and algorithm optimization search. Canny proved that trajectory optimization search is an NP problem[2]. Among them, the intelligent track search algorithm is divided into two categories: One is a deterministic search algorithm, the other is a random search algorithm. As the core of the deterministic algorithm, the A-Star (A*) algorithm[3] of the graph search algorithm solves the problem of trajectory planning in a static environment. And Introduce Dynamic A*(D*) algorithm to solve the problem of low efficiency in dynamic environment[4].In order to solve the problem that the D* algorithm is more complicated and difficult to understand, in 2004, Sves Koening and others introduced environmental incremental information and developed the Lifelong Planning A* (LPA*) algorithm on this basis[5],which greatly improved search efficiency by multiplexing environmental information.However, when people use the LPA* algorithm, they need to re-plan the route from the starting point to the target point, so in 2005, Sves Koening et al. proposed the D-Start Lite(D*Lite) algorithm[6,7]. The D*lite algorithm only needs to consider the path information from the current node to the target point during path re-planning, so the search efficiency is higher. Reference [7], a path planning algorithm based on fast D* Lite is proposed. The algorithm can completely or partially avoid re-planning and calculation of some nodes, which reduce search time and improve search efficiency. In reference [8], in order to solve the collision problem of the path planned by D* Lite, a method combining lazy line of sight algorithm and distance transformation is proposed, which makes the paths under different environments very smooth and safe. Reference [9] uses an improved D*Lite search algorithm to design a three-dimensional fast track planning method. This method uses an improved cost evaluation function to combine the constraints of trajectory planning with an improved search algorithm,which can better solve the problem of trajectory planning when the target moves in an uncertain environment.Reference[10]in this paper,it introduce a novel multiobjective incremental algorithm, multiobjective D* Lite

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3021073, IEEE Access

(MOD*Lite) built upon a well-known path planning algorithm, D*Lite. The improved algorithm solves the problem of obtaining better execution time without sacrificing the optimality of too much path length. Reference[11]present an interpolation-based planning and re-planning algorithm for generating smooth paths through non-uniform cost grids. It is particularly well suited to planning smooth, least-cost trajectories for mobile robots.

From the above documents, it can be found that The improvement of the original D*Lite algorithm is to use fast re-planning, reduce the search nodes of the algorithm to improve search efficiency, or combine with other algorithms to achieve path security.With the complexity of the application environment and the development of intelligent technology. How to improve the search efficiency of the algorithm in the process of trajectory planning while ensuring the safety of the planned path is the main improvement direction of the D*lite algorithm.

In this paper, through the analysis of D * Lite algorithm, it is found that when operating in a real environment, due to the relatively large size of the aircraft itself, there will be a collision situation with obstacles.Therefore, this paper proposes to improve the selection of child nodes, and adopt the node priority strategy to improve the D*lite algorithm,which makes the planned path is the optimal path while being safe and feasible.At the same time,the evaluation function in the D*lite algorithm is optimized and optimized to reduce the search time of the algorithm and improve the search efficiency of the algorithm.Under the two-dimensional map environment, the improved algorithm is simulated and tested by MATLAB platform to verify its effectiveness and feasibility.

II. Related work

A. Establish the environment model

In the space environment modeling of aircraft trajectory planning, the amount of calculation and engineering applicability need to be considered. And it is required to use simple rule diagrams for modeling as much as possible. Therefore, this paper choose the grid method to model among the grid network, the configuration space method and the topology map method. The grid method is currently the most widely studied method of planning space, which divides the space for low-flying aircraft into multiple simple areas. These simple areas are called grids. Use grid cost based description to describe obstacles or threats in the planned space. This description method is based on the description of the entire planning space, using grids to divide the environment, using a two-dimensional grid to represent the environment, and increasing the value of a specific grid to describe threats or obstacles. Usually, the projection of the threat zone on the aircraft flight path in the vertical plane is composed of one or more polygon rings. Simulate the threat area of aircraft flight, and approximate it as a combination of multiple squares.Set the threat area as follows:

- If the threat area is not full of a grid, it is counted as a grid;
- If there is a hollow part in the threat area, set this part as the overall threat area;

If the physical distance between adjacent threat areas is small, the middle area of the two threat areas is also regarded as a threat area, otherwise, it is regarded as two different threat areas.As shown in Figure 1,the generation value of each gray grid is set to 1, which indicates that there is a threat;The value of non-gray area is set to 0, which means that there is no threat, and it is a feasible area for the aircraft.The task of planning is to avoid the gray threat area and search for a path from the starting grid to the target grid.

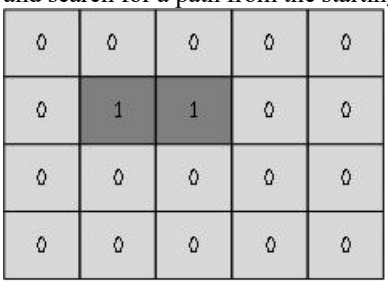


Figure 1 Raster network diagram

As shown in Figure 2, the raster map used in this article.Set the threat zone as an obstacle and indicate it with a red circle, and the substitute value of the obstacle is set to -1.The target generation value is set to 0, represented by a cyan circle.The moving body is represented by a purple circle, its generation value is set to 1.And the free space cost value is 2 indicating that there is no threat, which is a feasible area for aircraft.

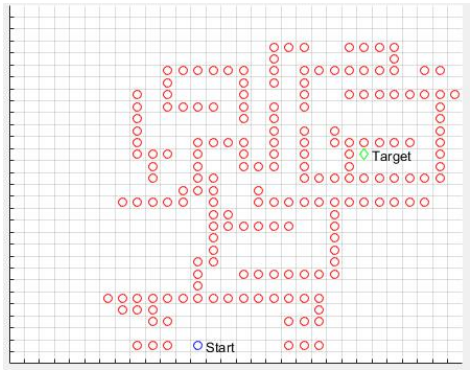


Figure 2 The grid network diagram used in this paper

B. Selection of D-Start Lite algorithm

Compared with other heuristic search algorithms, D*lite algorithm can deal with the situation of unknown environment very well. When a new unknown obstacle appears in the environment, you can quickly update the information of the nodes around the obstacle, put the nodes that are not continuous due to the appearance of new obstacles into the open list again, and then quickly re-planning.

TABLE I.COMPARISON OF A* AND LPA* ALGORITHMS

Algorithm	Application of Environment	The Method of Search	Valuation function
A* algorithm	High search efficiency in static environment, and not suitable for dynamic environment	heuristic search	F(s)=g(s)+h(s)
LPA* algorithm	Both static and dynamic environments are applicable	Incremental search for rapid re-planning in a dynamic environment	Introduce the rhs(s) variable as the minimum cost estimate

It can be seen from Table I that the A* algorithm has a high path search efficiency in a static environment, but the LPA* algorithm has a significantly higher search efficiency than the A* algorithm in a dynamic environment. The cost function of the A* algorithm is f=g+h, and the priority of the extended node is determined by the size of the f value. LPA* algorithm is an incremental version of A* algorithm.LPA* algorithm maintain two estimates g value g(s) and rhs value rhs(s) of the starting distance g*(s) of each vertex s, by determine the node status by comparing the values of g(s) and rhs(s).The LPA* algorithm uses a two-dimensional key, and the order of expanding nodes is determined based on the size of the key value.

D*lite algorithm is deepened on the basis of LPA* algorithm idea. The version presented in LPA* algorithm searches from the starting vertex to the target vertex, and the starting point is fixed.However, the search direction of the D*lite algorithm is opposite to the LPA* algorithm. It searches from the target vertex to the starting vertex, and the starting node can be changed. In addition, the definitions of variables such as h, g, and rhs are also the exact opposite of LPA*.As shown in Table II, D*lite algorithm has advantages over LPA* algorithm.

TABLE II COMPARISON OF LPA* AND D*LITE ALGORITHMS

Algorithm	Application of Environment	The Method of Search	Re-planning operations
LPA* algorithm	Both static and dynamic environments are applicable	Forward search from the starting node to the target node	Update the path information of the node from the starting point
D*Lite algorithm	Applicable environment is more flexible,and the starting point can change with time	Reverse search from target node to starting node	Update the path information of the current node

Based on the comparison between the above algorithms,the D*Lite algorithm will become the research direction of many scholars, and optimize and improve the algorithm from the efficiency and accuracy of path search.

C. Principle of D-Start Lite algorithm

The D*lite algorithm is extended based on the idea of LPA* algorithm, they are all based on A* algorithm.D*lite algorithm and LPA* algorithm add rhs variable on the basis of A* algorithm,and the core of the D*lite algorithm lies in the incremental search path based on the unknown area as a free space.The D*lite algorithm selects the minimum rhs value as the best expansion direction each time, and iterative searches and calculates the cost estimates of the 8 surrounding lattices until it finds the target position [12].In the D* Lite algorithm, g(s) and rhs(s) are the minimum cost estimates of the starting point of the node s distance.Since the search direction of the D*lite algorithm is opposite to the A* algorithm and the LPA* algorithm, the g(s) value will be recalculated when expanding 8 adjacent grids around the grid.Update the g(s) value and choose the g(s) with the smallest generation value.Rhs(s) is calculated from the g value of its forward point, and its calculation formula is as follows:

$$\mathrm{rhs}(s)=\begin{cases}0, s=s_{goal}\\ \min\limits_{s'\in Pred(s)}(g(s')+c(s',s)), others\end{cases} \tag{1}$$

Where c(s', s) represents the edge cost of nodes s’ to s, which is generally recorded as 1.S’∈pred(s) denotes the set of predecessors of vertex s ∈ S.When g(s)=rhs(s), it means that nodes are consistent, otherwise it is non-uniform. Among them, g(s)>rhs(s) is called local over-consistency, and g(s)<rhs(s) is called local under-conformity.When evaluating the estimated value of grid points, D* Lite also introduces key(s) value for comparison, where key(s) contains two values key(s)=[key(s1); key(s2)],which satisfy the following formulas:

$$\text{key1(s)=min(g(s)，rhs(s))+h(s)} \tag{2}$$

$$\text{key2(s)=min(g(s)，rhs(s))} \tag{3}$$

k1 and k2 as the priority queue arrangement parameters.The size of the key(s) value is used as the expansion priority, which determines the expansion order of the nodes in the queue.First compare the size of the K1 value, select the smallest k1 value grid as the next grid, and when the k1 values are equal, consider the size of the k2 value.The h(s) heuristic function is the same as the heuristic function in the LPA* algorithm, which represents the estimated value between the current node and the starting point, causing the object to always move toward the target point.The value of h(s) is very important, it determines the speed and accuracy of the algorithm search.The search heuristic function of node s includes two parts: the cost c (s, s') from node s to node s' and the value of the heuristic function from node s' to the target point s_{goal}.The calculation formula of h(s) is:

$$h(s,s_{start})=\begin{cases}0, s=s_{goal}\\ c(s',s)+h(s',s_{goal}), others\end{cases} \tag{4}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3021073, IEEE Access

The description of D*Lite path planning algorithm is shown in Table III [13]

TABLE III D * LITE ALGORITHM

D*lite algorithm

Procedure CalculateKey(s)
{01'}return[min(g(s),rhs(s))+h(S_{starts})+km;min(g(s),rhs(s))];
Procedure Initialize()
{02'}U=ϕ;
{03'}km=0;
{04'}for all s$\in$S rhs(s)=g(s)=∞;
{05'}rhs(s_{goal})=0;
{06'}U.Insert(s_{goal},CalculateKey(s_{goal}));

Procedure Update Vertex(u)
{07'}if (u≠s_{goal}) rhs(u)=min$_{s'\in Succ(u)}$(c(u,s')+g(s'));
{08'}if(u$\in$U)U.Remove(u);
{09'}if(g(u)≠rhs(u)) U.Insert(u,CalculateKey(u));

Procedure Compute ShortesPath()
{10'}while(U.TopKey()<CalculateKey(s_{start})OR rhs(s_{start}≠g(s_{start})
{11'}kold=U.TopKey();
{12'}u=U.Pop();
{13'}if(kold<CalculteKey(u))
{14'}U.Insert(u,CalcculateKey(u));
{15'}else if(g(u)>rhs(u))
{16'}g(u)=rhs(u);
{17'}for all s$\in$Pred(u) Update Vertex(s);
{18'}else
{19'}g(u)=∞;
{20'}for all s$\in$Pred(u)$\cup${u} Update Vertex(s);

Procedure Main()
{21'}s_{last}=s_{start};
{22'} Initialize();
{23'}ComputeShortesPath();
{24'}while (s_{start}≠s_{goal})
{25'}s_{start}=argmin$_{s'\in Succ(s_{start})}$(c(s_{start},s')+g(s'));
{26'}/*if(g(s_{start})=∞ then there is no known path*/
{27'}Move to s_{start};
{28'}Scan graph for changed edge costs;
{29'}if any edge costs changed
{30'}km=km+h(s_{last},s_{start});
{31'}s_{last}=s_{start};
{32'}for all directed edges(u,v)with changed edges costs
{33'}Update the edge cost(u,v);
{34'}Update Vertex(u)
{35'}Compute ShortesPath();

III. Optimization of D-Start Lite algorithm

The D*Lite algorithm is based on the LPA * algorithm, which is improved to make it possible to re plan quickly when facing obstacles[7],thus improving the planning efficiency. There have been many studies on the application of D*lite algorithm in trajectory planning. How to improve the efficiency of the algorithm and the security of algorithm application is the main direction of algorithm improvement. However, in practical applications, planning algorithms need to consider the safety of planned trajectories while improving the efficiency of the algorithm.

The traditional D*lite algorithm effectively expands the 8 nodes around the node with the parent node as the center.Put the expanded node into the priority queue (including obstacle nodes), and select the node with higher priority in the priority queue as the current node.Traverse the surrounding nodes again until reaching the target node. The final planned path is easy to be close to obstacles However, the D*lite algorithm itself still has deficiencies. The path planned by D*lite is close to obstacles, when operating in a real environment, due to the large size of the aircraft itself, if the aircraft is very close to the obstacle during the re-planning operation, then the possibility of collision between the aircraft and the obstacle is very high, which making D*lite plan Path is not safe. In order to solve this problem, this paper improves the expansion mode of sub nodes, so that the planned path is safe and feasible. In addition, in order to improve the efficiency of D*lite algorithm in trajectory planning,the weighted valuation function in the algorithm is improved.

A. Valuation function

The D*Lite algorithm uses the priority queue U to store the non-uniform nodes to be expanded, and the expansion order of the node S is determined by the valuation function.In order to reduce search time and improve planning efficiency, heuristic function weighting is a commonly used method to improve the valuation function.In the valuation function of D*Lite, min(g, rhs) is the width-first component, while the depth-first component is represented by h.You can use w to adjust the relative weights of the two components min(g, rhs) and h in key1 = min(g, rhs)+w*h(1≤w≤∞),thus, on the basis of maintaining the optimization of the algorithm as much as possible, the search efficiency of the algorithm is improved.

Where the evaluation function key (s) = [key (S1); key (S2)], key1 (s) is equivalent to the evaluation function f (s) in the A* algorithm, and key2 (s) is equivalent to the heuristic function g (s) in the A* algorithm.Through reading the literature, it is found that the number of nodes expanded by the A* algorithm to select key2(s) ≈ h(s) is significantly less than key2(s) ≈ h(s).Therefore, this paper uses an improved weighted valuation function [13]:

$$\mathrm{key}(s)=\begin{cases}[g(s)+h(s),\text{-}1], g(s)<rhs(s)\\ [rhs(s)+w*h(s),h(s)], otherwise\end{cases} \quad (5)$$

At the same time, this paper replaces the Euclidean distance in the original algorithm by Chebyshev distance.The aircraft can move along the direction of the corner line, solving the problem of relatively high turning frequency within a small range of the planned path.In this way, the distance between the current node and the target point can be determined by the following relationship:

$$D_{chess}(Goal,n)=h_{diagonal}+h_{straight} \quad (6)$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3021073, IEEE Access

Among them,$h_{diagonal}$ and $h_{straight}$ are:

$$h_{diagonal}= \min((x_{goal}-x_1),(y_{goal}-y_1)) \quad (7)$$

$$h_{straight}=abs(abs(y_{goal}-y_1)-abs(x_{goal}-x_1)) \quad (8)$$

Where (x_{goal}, y_{goal}) is the position coordinate of the target node, and (x1, y1) is the position coordinate of the current extended node.In this paper, the cost in the vertical and horizontal directions is 1, the cost in the diagonal direction is 1.4, and the cost in the two directions is substituted into:

$$h(s)=\min((x_{goal}-x_1),(y_{goal}-y_1))+1.3abs(abs(y_{goal}-y_1)-abs(x_{goal}-x_1)) \quad (9)$$

B. Selection of child nodes

The D*lite algorithm is similar to the A* algorithm, and it is based on the current node. It uses an iterative search method to expand to the 8 neighboring nodes and push the resulting non-uniform nodes into the open queue. But the path planned in this way will be close to obstacles. Due to the large size of the aircraft, it is very likely to collide with the obstacle in the application of the actual environment, and there are hidden safety risks. In order to solve this problem, this paper improves the selection of child nodes of the D*lite algorithm.

As shown in the figure, the priority selection strategy is adopted for node selection [14].The child node in the vertical and horizontal directions is selected as the first-level priority node, and the child node in the opposite diagonal direction is the second-level priority node.Select the second priority node by the status of the first priority node: If node of 5 is an obstacle, child nodes 3 and 8 in the diagonal direction cannot be selected, and paths 0-3 and 0-8 are excluded. The same analogy, if there are obstacles in nodes(2, 4, 7), the corresponding diagonal sub-nodes cannot be selected.

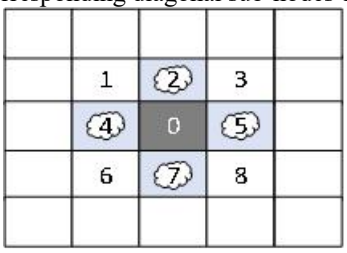


Figure 3. priority distribution of node 0

C. The process of improving the D-Start Lite algorithm

In the actual living environment, the aircraft itself has a large size and is prone to appear close to obstacles. This paper proposes the strategy of adopting node priority in generating nodes. Under the function of node priority, the nodes with obstacles around are not put into the scope of node search calculation, and a certain safety distance is reserved to meet the application in the actual environment. Although the phenomenon of being close to obstacles is avoided, the number of path nodes and turning points have increased. In this paper, Chebyshev distance is used to replace the Euclidean distance in the original algorithm, and the diagonal function is used to improve the heuristic function, so that the turning point of the planned path is reduced. At the same time, heuristic function weighting is used to optimize the valuation function to reduce the number of expansion of path nodes. The search time is reduced, and the search efficiency is improved.

Figure 4 is a flowchart of the improved D* Lite path planning algorithm:

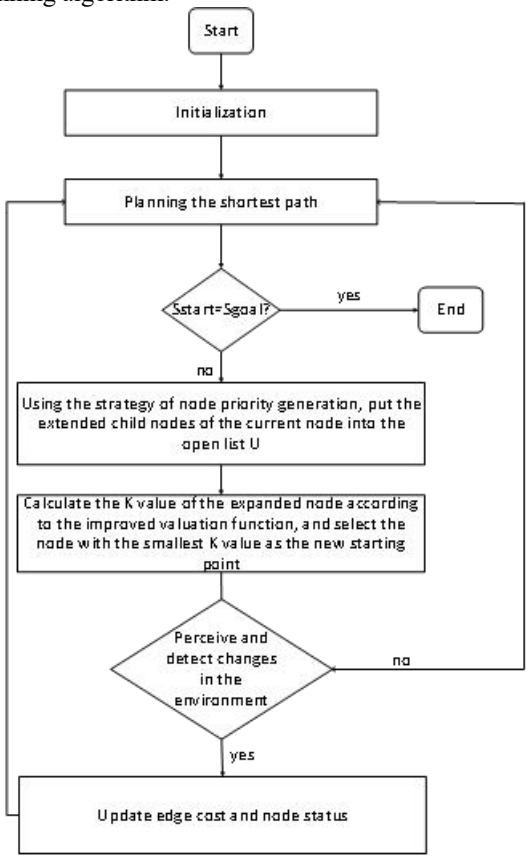


Figure 4 Improved D* Lite algorithm flow chart

The specific steps are:

(1) Initialization: The g(s) and rhs(s) values of all grids are infinite. First put the local inconsistent node sgoal into the priority queue.

(2) Calculate the shortest path: Starting from the target point, adopt the priority grading strategy generated by the node. Put the currently needed expansion node into the priority queue, and calculate the size of g(s), rhs(s) and k value according to the above improved valuation function and heuristic function. Select the node with the smallest k value as the next expansion node. Loop this step until the target point is equal to the starting point, then the search is terminated.

(3) Generate the shortest path: Moving from the current node to the grid with the smallest estimated value, generating a path to the target node.

(4) The environmental information changes: When it is found that the next node of the path is an obstacle, the environmental information will change accordingly, such as the value of edge cost. At this time, update the k value of the node and the state of the surrounding critical point affected

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3021073, IEEE Access

by the node, then go back to step 3 and find a path according to step 3.

IV. Experiments

In order to verify the feasibility of the improved D*lite algorithm, this paper conducts a simulation experiment on the improved D*lite algorithm in the MATLAB platform. The environment used in the simulation experiment is that the PC memory is 8G, the operating system is Windows 10 64-bit operating system, and MATLAB R2016b is used as the simulation software.The experimental results are compared and analyzed as follows: (1) Under the simple map,set obstacles to be randomly distributed in a grid of 30× 30 units. (2) Set the grid map of 100×100 units , which the distribution of obstacles is more complex. The feasibility of the improved algorithm in this paper is verified through comparative experiments.

Set obstacles randomly distributed in a 30×30 unit raster. As shown in Figure 5(a), it is the path simulation diagram of the original algorithm, where the red circle represents an obstacle, and when traversing to the threat area, the red circle becomes solid; the blue circle represents the trajectory of the aircraft. The yellow position is the starting point, and the cyan grid is the position of the target point .It can be seen from the figure that the path planned by the original algorithm has redundant nodes and trajectory segments that are close to obstacles. In the actual environment, considering the size of the aircraft itself, collisions will inevitably occur in the planned path. Therefore, the node priority strategy is adopted in the node generation link to improve the D*lite algorithm. The improved path is shown in Figure 5(b).

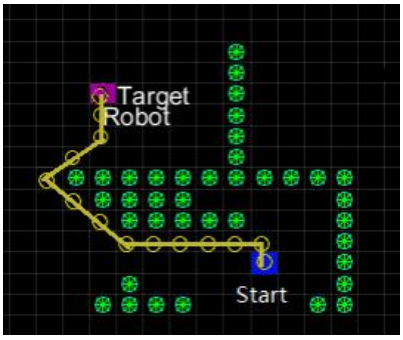


Figure 5(a) Simulation diagram based on the original D*lite algorithm

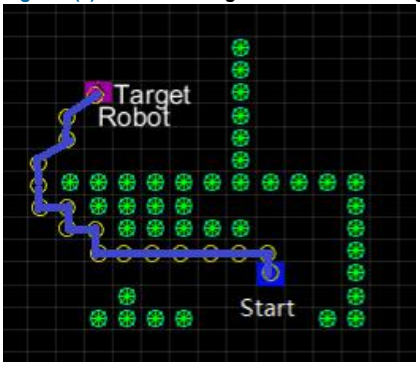


Figure 5(b) Path simulation diagram of adding node priority strategy

From the above two pictures, when the improved D*lite algorithm is close to the obstacle area, according to the node priority strategy, the node on the side with the obstacle is not expanded. It avoids the aforementioned phenomenon of clinging to obstacles and can be better applied to the actual environment.

Since a certain space is reserved between the trajectory and obstacles, the number of nodes in the path increases, making the search efficiency low. Therefore, this paper uses Chebyshev distance to replace Euclidean distance in the original algorithm, and the diagonal function is used to improve the heuristic function. So that the turning point of the planned path is reduced. At the same time, heuristic function weighting is used to optimize the valuation function, which can reduce the number of expansion of path nodes and search time, and improve the search efficiency.

In the evaluation function of the improved D*Lite algorithm, use the variable w to adjust the relative weights of width priority component min (g, rhs) and depth priority component h in key1 = min (g, rhs) +w * h($1 \leq \omega \leq \infty$).This paper takes different values of w for path planning experiments, and the corresponding results are shown in the following table:

TABLE IV. THE LENGTH OF THE SHORTEST PATH, THE NUMBER OF EXTENDED NODES AND THE ELAPSED TIME IN THE D*LITE ALGORITHM UNDER DIFFERENT WEIGHTS

w	The length of the shortest path	the number of extended nodes	the elapsed time
w=1.0	52.21	1580	16.013
w=1.1	52.796	1391	13.556
w=1.2	53.382	1231	11.907
w=1.3	53.968	1103	10.750
w=1.4	54.796	983	9.251

It can be seen from the table that as w increases, the path length increases but does not exceed w times the shortest path. The number of expanded nodes of the path is significantly reduced, so the search speed of the improved D*lite algorithm is faster. The search time is significantly reduced, and the search efficiency of the algorithm is improved. In order to make the effect of the planned path relatively optimal, the algorithm in this paper takes w=1.2. The simulation result is shown in Figure 6:

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3021073, IEEE Access

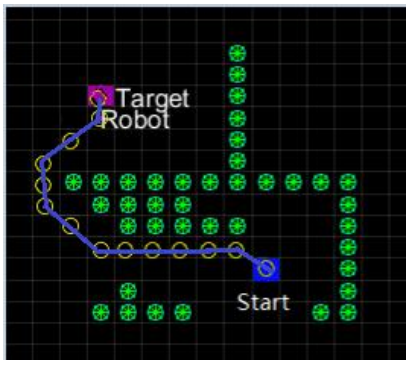
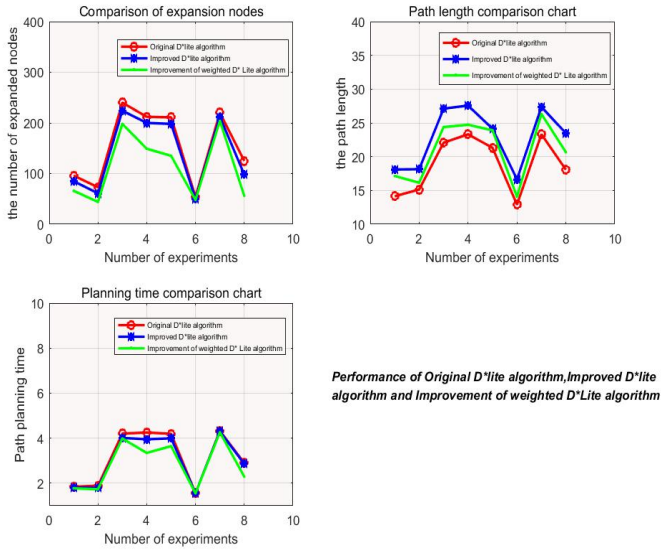


Figure 6 the weighted D*lite algorithm with improved child node selection method

In order to verify the feasibility of the improved algorithm, this paper conducts 8 experiments to compare the path length, the number of expanded nodes and the planning time of the planned path. As shown in Figure 7.



Performance of Original D*lite algorithm,Improved D*lite algorithm and Improvement of weighted D*Lite algorithm

Figure 7 Comparison of trajectory length, number of expansion nodes and traversal time under different algorithms under 8 sets of experiments

Table V lists the specific experimental data of the three algorithm trajectory planning in the simple map.According to the data in the table, compared with the above two algorithms, the improved algorithm in this paper has significant advantages in planning time and expanding the number of nodes:

TABLE V COMPARISON OF ALGORITHM EXPERIMENT RESULTS

Simple map	The length of the shortest path	the number of extended nodes	the elapsed time
Original D*lite algorithm	18.782	153.875	3.145
the D*lite algorithm with improved child node selection method	22.819	141	3.034
the weighted D*lite algorithm with improved child node selection	20.9155	128	2.022

Perform planning experiments on three sets of algorithms in the same environment. The planning time of the improved D*lite algorithm is significantly reduced compared to the original D*lite algorithm. However, due to the change in the selection method of the child nodes, there are redundant nodes in the path of the planning site. The path length planned by the improved D*lite algorithm has only slightly increased. And the number of expansion nodes and turning points of the path have been significantly reduced, which the resulting path is safe and effective. Therefore, the improved D*lite algorithm is effective.

Compare the three performances of the original D*lite algorithm, the D*lite algorithm with improved child node selection method, and the weighted D*lite algorithm with improved child node selection method mentioned above, in three different sizes of unknown maps.Figure 8 compares the target-oriented navigation of the three algorithms in unknown terrain. In 3 areas of different sizes, randomly generated terrain of each size whose obstacle density varies from10 to 40 percent. This graph shows the operating conditions of the three performances in different algorithms. The improved D*Lite in this paper performs better than the other two algorithms with respect to all three measures.And the figure also shows the corresponding 95 percent confidence intervals to demonstrate that our conclusions are statistically significant.

Performance of Original D*lite algorithm，Improved D*lite algorithm and Improvement of weighted D*Lite algorithm

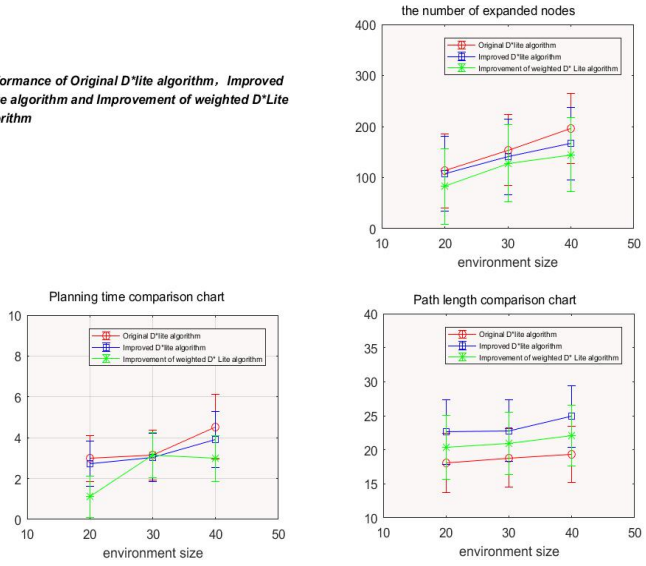


Figure 8 Comparison of Original D*lite algorithm,Improved D*lite algorithm and Improvement of weighted D* Lite algorithm

At the same time, in order to further verify the scalability of the improved D*lite algorithm, a more complex environment is selected for simulation.This article uses the same experimental environment settings as the previous simulation experiments with three different algorithms in the unknown terrain with a more complex environment.The verification result is shown in Figure 9:

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3021073, IEEE Access

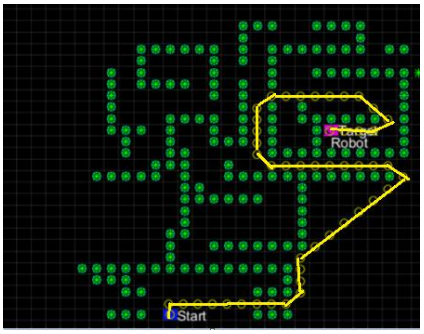


Figure 9(a)Original D*lite algorithm in complex environment

Figure 9(a) optimized D*lite algorithm in complex environment

As shown in Figure 9, there are fewer redundant points and inflection points in the path, so the trajectory is smoother than the path planned by the original algorithm. Therefore, in the actual environment, it is possible to avoid collision between the aircraft and obstacles during navigation and steering. In order to further verify the effectiveness of the algorithm, in the same complex environment map above, select different starting points and target points, and conduct 10 different experimental simulations. The data is averaged after the experiment, and the comparison results are shown in Table V.

TABLE V COMPARISON OF ALGORITHM EXPERIMENT RESULTS

Complex map	The length of the shortest path	the number of extended nodes	the elapsed time
Original D*lite algorithm	52.796	1580	16.013
the D*lite algorithm with improved child node selection method	56.981	1432	15.725
the weighted D*lite algorithm with improved child node selection	53.953	1350	13.978

From the statistical data in the table, it can be seen that the path length of the improved algorithm is increased by 2%. However, compared with the d * Lite algorithm, which only improves the selection of sub nodes, it reduces by 4.3%. At the expense of a certain path length, the security of the path is enhanced. Due to the improved heuristic function and optimization of the valuation function, compared with the original D*lite algorithm, the improved D*lite algorithm reduces the number of expanded nodes by 14.5% and the time consumed by the algorithm by 12.7%.Therefore, in a complex environment, the improved D*lite algorithm has a short planning path and high algorithm efficiency under the premise of safe planning trajectory, which further proves the feasibility of the improved D*lite algorithm.

According to the above analysis, three algorithms are simulated in the same environment map.It can be concluded from the experimental simulation results that,the improved algorithm in this paper solves the problem that the aircraft is close to the obstacle because of its large size.On unknown maps with different obstacle densities, compare the three algorithms in terms of trajectory planning length, number of expansion nodes, and planning time.With the complexity of the environment map, the advantages of the improved algorithm used in this paper in expanding the number of nodes and planning time have increased significantly.On the basis of sacrificing a certain trajectory planning length, which improve the search efficiency of the planning algorithm.Therefore, the improved algorithm used in this paper not only improves the search efficiency of the algorithm, but also solves the safety of the planned flight path.And meets the feasibility and scalability of the D*lite algorithm in practical engineering.

V. Conclusions

The existing D*lite algorithm can only single solution the safety problem of trajectory planning alone or improve the search efficiency of the algorithm.Therefore, in order to solve the problem of ensuring the safety of the trajectory planning path while improving the efficiency of the algorithm.This paper proposes an improved D*lite algorithm based on the node priority grading strategy, which makes the planned trajectory away from obstacles. In the actual application environment, the possibility of collision between aircraft and obstacles is avoided. Through the improvement and optimization of the heuristic function and the evaluation function and the planning time is reduced, and the planning search efficiency is improved. This paper compares the improved D*lite algorithm with the original algorithm to verify the effectiveness of the algorithm and its adaptability to different environments. Although the D*lite algorithm still has insufficient planning for the local environment, which is also a problem to be solved in the next step.

REFERENCES

[1] Lei Lei.Application research of intelligent optimization algorithm in UAV track planning[J].Ship Electronic Engineering,2017(02):34-37.
[2] J Canny.A Computational Approach to Edge Detection[J].IEEE Trans on PAMI,1986,8(6):679-698.
[3] Hart E , Nilssos J, Raphael B .A formal basis f or the heuristic determination of minimum cost paths [J] .IEEE Transactions of systems science and cybernetics, 1968 , 4 (2).
[4] Stentz A. The focussed D* Algorithm for Real-Time Re-planning[C]//International Joint Conference os Artificial Intelligence. 2000:3310-3317.

[5] Koenig S, Likhachev M, Furcy D. Lifelong planning A [J]. Artificial Intelligence, 2004, 155(1-2): 93-146
[6] Koenig S, Likhachev M. D* Lite [C]//AAAI/IAAI. 2002: 476-483
[7] Koenig S, Likhachev M. Fast re-planning for navigation unknown sterrais[J]. Robotics IEEE Transactions on, 2005, 21 (3):354-363
[8] Yanan Zhang , Fengcai Sun , Xuhua Shi. Mobile Robot Path Planning Based on Fast D*Lite Algorithm[J]. Data Communication,2018(01):46-51.
[9] Lu Huang, Feidong Zhou. Mobile Robot Path Planning Based on Path Optimization D*Lite Algorithm[J].Control and Decision,2020,35(04):877-884.
[10] Oral T, Polat F. MOD* Lite: an incremental path planning algorithm taking care of multiple objectives. IEEE Transactions on Cybernetics. 2015 Feb 27;46(1):245-57.
[11] Ferguson D, Stentz A. The Field D* algorithm for improved path planning and re-planning in uniform and non-uniform cost environments. Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-05-19. 2005 Jun.
[12] Xia Chen,Dong Liu.Three-dimensional trajectory planning of unmanned aerial vehicle during target movement using D* Lite algorithm[J].Electronics and Control,2013,20(07):1-5.
[13] Yunxia Lian, Yongsheng Fan, Hongying Yu, Zhen Yang. Application of improved D*Lite algorithm in virtual soldier path planning[J].Modern Electronic Technology,2018,41(06):23-27+33.
[14] Shuaijun Wang, Likun Hu,Yifei Wang. Path planning of indoor mobile robot based on improved D* algorithm[J]. Computer Engineering and Design,2020,41(04):1118-1124.
[15] Hu Pan. Research on Autonomous Path Planning Algorithm of Mobile Robot in Complex Environment [D]. Dalian Maritime University,2018.

KAILI XIE received his Bachelor's degree in Software Engineering from Harbin University of Technology, Harbin, China. Since 2018, he is a M.S. candidate in Information System in Space Engineering University, Beijing, P.R. China. His research focuses on path planning.

JIE QIANG received the bachelor's degree in electronic engineering from Shanghai Jiao Tong University , Shanghai, China, in 2014, and master 's degree in computer science from Space Engineering University, Beijing, China, in 2017, where he is currently pursuing the Ph.D. degree in information system. His research interests include deep learning and reinforcement in network security.

HAITAO YANG received his Bachelor's and Master's degree in Communication Engineering and Ph.D. degree in Information System from Space Engineering University, Beijing, P.R. China. He is an associate professor of Communication & Information System. His research focuses on image processing and decision making. He published his book Design of Complex Information Network Performance.