Unmanned Ship Path Planning Based on RRT

Xinjia Chen[1], Yanxia Liu[1], Xiaobin Hong[2](✉), Xinyong Wei[2], and Yesheng Huang[2]

[1] School of Software Engineering, South China University of Technology, Guangzhou, China
[2] School of Mechanical and Automotive Engineering, South China University of Technology, Guangzhou, China
mexbhong@scut.edu.cn

Abstract. Path planning is a task of primary importance for unmanned ship, but current algorithms are complex and inefficient. In this paper, we propose a Rapidly-Exploring Random Tree algorithm (RRT) for path planning of unmanned ship, which can obtain an asymptotically optimal path planning in limited time. Moreover, an extension of RRT algorithm has been proposed to overcome the actual demand of multi-waypoint path planning for unmanned ship. The feasibility and effectiveness of the proposed algorithm was proved by simulation on MATLAB™ platform.

Keywords: Unmanned ship · Path planning · Rapidly-Exploring Random Tree

1 Introduction

With the rapid development of science and technology, the field of modern unmanned ship is constantly being exploited and utilized. Unmanned ship plays an important role in autonomous navigation, especially sea patrol and cargo delivery. The rise of artificial intelligence promotes the rapid development of driverless technology, which has led to many path planning algorithms to be proposed successively [1].

Traditional path planning algorithm is the main choice for unmanned ship path planning, such as artificial potential field method [2], Dijkstra search algorithm [3] and A* algorithm [4, 5]. These algorithms have the following shortcomings: Map modeling is complex; the heuristic ideas always complicated and real-time performance is poor. Whereas, the RRT algorithm applied in the mobile robot path planning is superior: The path points in RRT algorithm are simply generated by random sampling; map information requirement is simple; RRT algorithm can maintain the consistency with the dynamic constraint during the generation of path points, and a feasible path can be generated within limited time.

In this paper, we introduce the implementation of RRT-based path planning algorithm for unmanned ship. The definition of path planning problem in the area of unmanned ship is given in the first place. We then promote the multi-waypoint path planning method based on improved RRT and analyze the performance of RRT algorithm. Finally, by simulation experiments, the effectiveness of RRT path planning algorithm for unmanned ship is verified.

2 Definition of Unmanned Ship Path Planning Problem

The main goal of this paper is to propose a method for unmanned ships to generate a valid path within limited time. We obtain sea surface information by S57 electronic chart [6]. Moreover, an improved path planning method based on RRT algorithm is proposed to solve the problem of unmanned ship multi-waypoint path planning problem [7].

The environment model for unmanned ships is constructed from S57 electronic chart, which conforms to the S57 international standard, encapsulated by the ISO8211 standard. S57 electronic chart contains the geographical location and object information of the sea area. We parse the S57 electronic chart to acquire sea surface information using the GDAL library.

For illustrative purposes, represent the map space as M, and obstacles and non-navigation areas as $M_{obstacle}$, and navigable areas as M_{free}, then $M_{obstacle} = M/M_{free}$. The definition of single-waypoint path planning for unmanned ships is to find a continuous sequence from the specified start point q_{start} to the end point q_{goal} in M_{free}. We assume that the unmanned ship must pass n spots, which represented as targets[n]. The problem of multi-waypoint path planning for unmanned ship is defined as finding a continuous sequence in M_{free} which passes all the targets.

3 Single-Waypoint Unmanned Ship Path Planning Method Based on RRT

Rapidly-exploring Random Tree (RRT) is an incremental sampling algorithm proposed by Lavalle [8], with good performance in practical applications [9] with only a few parameters.

Moreover, RRT is an efficient planning algorithm that can be applied in multidimensional space. By setting a start point as root node, RRT will generate a random extended tree stochastically, and the algorithm terminates when a leaf enters the target area or reaches the target point.

When using RRT algorithm to plan a path for unmanned ship, points on the path tree represent the positions where the unmanned ship arrives. The algorithm terminates when the unmanned ship can reach the final target point. Assume that M represents the map converted from S57 electronic chart. Random extended tree T that keeps the information of extended nodes and the edges between the points; the start point q_{start} and the end point q_{goal}; randomly created point q_{rand} in the map area; the nearest node q_{near} in the tree T to the q_{rand}; new extended node q_{new}.

The collision detection parameter t, which can be chosen according to the size of the actual obstacle, denotes that taking t points evenly between q_{near} and q_{new}, and judging whether the edge[q_{near}, q_{new}] is in the free space. Besides, in the process of tree extension, probability p defines the probability of a new extended point toward the target point. The number of point k defines the maximum number of iterations of the algorithm. Tree extension distance Δt of the extended leaf node defines the distance between the new point and the nearest node in the tree, which can be adjusted on the basis of the dynamic restraint of unmanned ship [10]. Parameters mentioned above are those who have the impact on RRT performance.

RRT Algorithm process is as follows:

```
RRT(q_start, q_goal, map)
1. T <- InitTree();
2. T <- InsertNode(q_start);
3. for j=0 to k
4.   if rand()<p
5.      q_rand= q_goal;
6.   else  q_rand=(randi(mapWidth),randi(mapLength));
7.   q_near =Nearest(T,q_rand);
8.   q_new =NewPoint(q_near q_rand, Δt);
9.   if ObstacleFree(q_near, q_new) //collision check
10.     if GoalInNewEdge(q_new,q_rand, q_goal)  or q_new == q_goal
11.        return T;
12. return T
```

Algorithm steps:

Steps 1, 2: Initialize unique start node q_{start} for random extended tree T.
Step 3: Define the maximum number of iterations.
Steps 4, 5: Create a random point q_{rand} with probability p towards the q_{goal} direction.
Step 6: Create a random point q_{rand} with probability $(1 - p)$ towards any direction.
Step 7: Search node q_{near} for q_{rand} from T.
Step 8: A newly extended node q_{new} is generated by q_{near}, q_{rand} and distance Δt.
Step 9: Check whether there is any obstacle in q_{near} to q_{new}.
Step 10: The target is on the newly created edge or reach the target point
Step 11: Successfully find the path and returns tree T.
Step 12: After k times of iterations, q_{goal} is still not reached, and error information is returned.

New Node Extension: The point q_{rand} is randomly created in the finite space M, and the nearest node q_{near} to the q_{rand} is searched in the random extended tree T. Besides, q_{new} will be created by q_{near} and q_{rand}. The q_{new} (x, y) generation formula is as follows: (Fig. 1)

$$q_{new}(x, y) = q_{near} + \Delta t^* \frac{q_{rand} - q_{near}}{\|q_{rand} - q_{near}\|} \tag{1}$$

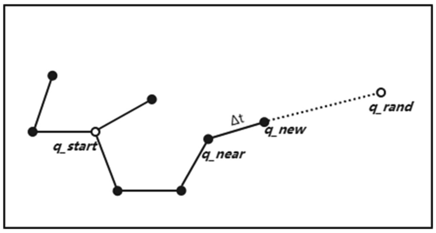

Fig. 1. New node extension

Collision Detection: To determine q_{new} is a legal extended point if there are any obstacles between q_{near} and q_{new}. Selecting some test points evenly between q_{near} and q_{new}, and then, q_{new} point is valid if all test points belong to M_{free} space (Fig. 2).

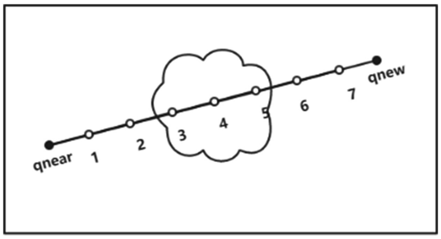

Fig. 2. Collision detection

4 Multi-waypoint Unmanned Ship Path Planning Method Based on RRT

The original RRT algorithm can quickly generate a sailing route in the case of a single waypoint. However, in the actual situation of unmanned ship, perhaps need to reach multiple preset waypoints. Current algorithm only solves the single-waypoint problem. In order to achieve multi-waypoint path planning, the target point is dynamically adjusted to waypoint, and the node information of the tree is dynamically refreshed during the growth process of RRT trees, so that the planned path passes through all the waypoints.

By setting the waypoints in a pair and consider them as root node of tree and the extension target position, makes the final path to pass all the waypoints. In this process, the extension process of the tree needs to utilize the existed node information, and the growth of the current tree is not related to tree nodes in the previous round, hence, the tree nodes need to be marked in each round to indicate whether it is an extensible node. Furthermore, to ensure that the final path includes all the waypoints, path smoothing algorithm is used for each segment of path.

The multi-waypoint RRT algorithm is shown as follows:

```
MultiRRT(targets, map)
 1. T <- InitTree();
 2. T <- InsertNode(targets[0]);
 3. for i=1 to i<targets.length()
 4.   $q_{start}$ =targets[i-1];  $q_{goal}$ =targets[i];
 5.   for j=0 to k
 6.      if rand()<p
 7.         $q_{rand}$= $q_{goal}$;
 8.      else  $q_{rand}$=(randi(mapWidth),randi(mapLength));
 9.      $q_{near}$=Nearest(T,$q_{rand}$);
10.      $q_{new}$ =NewPoint($q_{near}$,$q_{rand}$, $\Delta\tau$);
11.      if ObstacleFree($q_{near}$, $q_{new}$ )
12.         if GoalInNewEdge($q_{new}$,$q_{rand}$, $q_{goal}$)
13.            or $q_{new}$ == $q_{goal}$
14.             smooth(T,$q_{start}$,$q_{goal}$);
15.             refresh(T);
16.             break;
17.   break;
18. return T;
```

Step 3: Iterate by the number of waypoints.

Step 4: Dynamically change the start and end points in each iteration.

Step 14: After a path is generated in each cycle, invoking the path smoothing algorithm.

Step 15: Update the nodes information and mark the nodes generated in the current cycle.

Path Smoothing: There are a large number of redundant points in the path from q_{start} to q_{goal} generated by the RRT algorithm. We used a greedy approach to connect the farthest point with the start point and remove redundant points. Set the points sequence to be smoothed as path [1, n], and the smoothed path is path_smooth[1,n].

The path smooth algorithm is shown as follows:

```
Smooth(path)
1.path_smooth=path[1] //initialization
2.currentIndex = 1; //the start index to be smoothed
3.currentSmoothIndex = n;  // the end index to be smoothed
4.while currentIndex < n :
5.    while currentIndex < currentSmoothIndex :
6.        if ObstacleFree(path[currentSmoothIndex], path[currentIndex])
7.            path_smooth = [path_smooth, path[currentSmoothIndex]];
8.            currentIndex = currentSmoothIndex;
9.            break;
10.       else    currentSmoothIndex = currentSmoothIndex - 1;
11.currentSmoothIndex = n;
```

The result of the path smoothing algorithm is as follows: (Fig. 3)

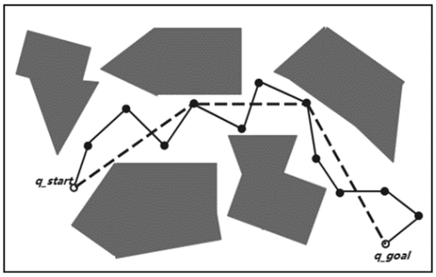

Fig. 3. Path smoothing

5 Simulation Experiments and Performance Analysis

5.1 Path Planning in Different Sea Environments Experiments

All simulation experiments in this paper running in MATLAB. Experiment 1 simulates the RRT path planning result of unmanned ship in the case of a simple sea environment. Experiment 2 simulates the path planning result under the situation of the sailing area is complex and has multiple corners. Experiment 3 simulates the result in the sea environment with multiple obstacles. RRT algorithm can still avoid obstacles accurately and quickly generate an effective path in all situations.

From Figs. 4, 5 and 6, it can be concluded that the RRT algorithm can solve a variety of sea environment situations, and can plan an asymptotically optimal sailing route within a reasonable time for different sea surface, which shows the excellent performance of the RRT algorithm for unmanned ship path planning.

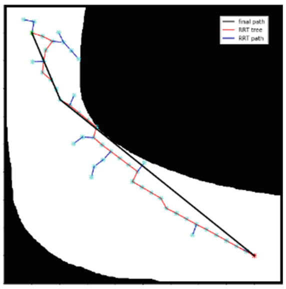

Fig. 4. Simple situation

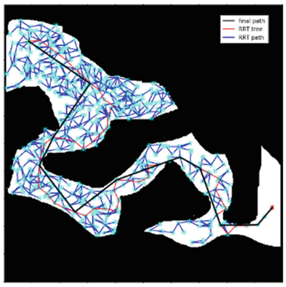

Fig. 5. Complex situation

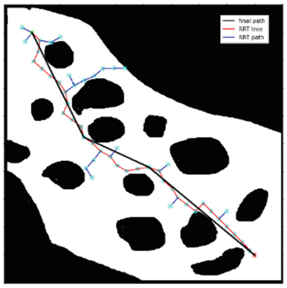

Fig. 6. Many obstacles situation

5.2 Multi-waypoint Path Planning Experiment

Experiment 4 simulates a common sea environment. The RRT algorithm calculates the result if the path only requires one waypoint. Experiment 5 is based on experiment four, presetting multiple points through which unmanned ship must pass, and using multi-waypoint RRT algorithm for path planning (Figs. 7 and 8).

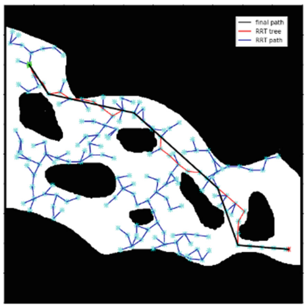

Fig. 7. One target position situation

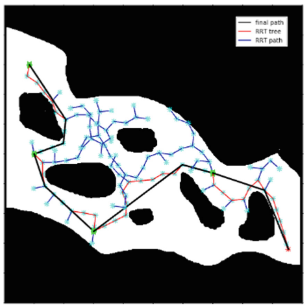

Fig. 8. Multi-waypoint situation

The simulation results show that even if the limitation of multiple waypoints is added, a path can be obtained through a multi-waypoint RRT algorithm in a roughly similar time with experiment 4, and can be accurately smoothed. In short, multi-waypoint RRT algorithm is feasible and effective.

5.3 Path Smoothing Experiment

The path smoothing algorithm obtains shorter path by removing redundant points. The comparison of result is as Fig. 9, and the black line indicates the smoothed path. Compared with the original path, the length of the path can be effectively shortened.

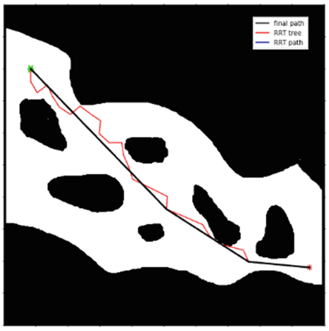

Fig. 9. Path smoothing

5.4 A* Algorithm Comparison Experiment

The A*(A-Star) algorithm in path planning is an effective algorithm for finding the optimal path. It combines the advantages of the Best-First Search and Dijkstra algorithms to improve the efficiency.

We compare the RRT algorithm with the path planning classical A* algorithm in the same operating environment and the same input matrix as M [500, 500].

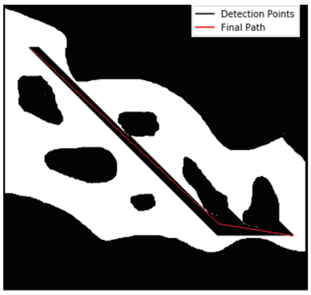

Fig. 10. A* algorithm

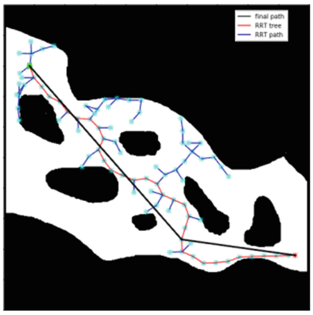

Fig. 11. RRT algorithm

The results of Experiment 6 and Experiment 7 are compared as follows:

It can be seen from the experiments, the results show in Figs. 10, 11 and Table 1, that the A* algorithm needs to detect a large number of points to find an optimal path,

Table 1. Algorithm performance analysis

Algorithm	Detection points	Time (s)	Path length
RRT	279	1.5	564
A*	8905	36	516

and the length of the final path of the RRT algorithm is slightly longer. Although the results of RRT algorithm is random, the number of created points is far less than that of the A* algorithm. Conversely, the rules for generating and detecting RRT algorithms are much simpler than other algorithms, even more, the efficiency can be increased by at least two orders of magnitude for real-time path planning of unmanned ship.

To sum up, RRT algorithm has many superiorities compared with other path planning algorithm, for instance, such as the artificial potential field method and ant colony optimization algorithms, need to set up complex heuristic rules, while RRT algorithm only extends through a simple random algorithm and can meet the dynamic constraints during the extension process.

6 Conclusion

This paper proposes the idea of applying the RRT algorithm to the field of unmanned ship path planning. And simulation experiments show that RRT can solve the problem of sea path planning in different situations. Based on the RRT algorithm, this paper proposes a multi-waypoint RRT algorithm to solve the problem of multi-waypoint path planning for unmanned ship. Finally, the path smoothing algorithm is used to shorten the path obtained by the RRT algorithm.

Acknowledgement. Supported by Science and Technology Project of Guangdong Province, China (Granted No. 2017B010118002).

References

1. Zhu, D., Tian, C., Sun, B., et al.: Complete coverage path planning of autonomous underwater vehicle based on GBNN algorithm. J. Intell. Rob. Syst., 1–13 (2018)
2. Zhang Y.: Research on USV self-avoidance navigation control system based on artificial potential field. Hainan University (2017)
3. Zhuang, J., Wan, L., Liao, Y., et al.: Global path planning of unmanned watercraft based on electronic charts. Comput. Sci. **38**(9), 211–214 (2011)
4. Chen, S., Liu, C., Huang, Z., et al.: AUV global path planning based on sparse A* algorithm. Torpedo Technol. **20**(4), 271–275 (2012)
5. Zammit, C., Kampen, E.J.V.: Comparison between A* and RRT algorithms for UAV path planning. In: Aiaa Guidance, Navigation, and Control Conference (2018)
6. IHO: IHO Transfer Standard for Digital Hydrographic Data, 3.1 edn., Publication S-57. International Hydrographic Bureau, Monaco (2000)
7. Liu, Y., Bucknall, R.: Efficient multi-task allocation and path planning for unmanned surface vehicle in support of ocean operations. Neurocomputing **275**, 1550–1566 (2018)
8. Lavalle, S.M.: Rapidly-exploring random trees: a new tool for path planning. Algorithmic Comput. Rob. New Dir., 293–308 (1998)
9. LaValle, S.M.: Planning Algorithms. Cambridge University Press, Cambridge (2006)
10. Du, Z., Wen, Y., Xiao, C., et al.: Motion planning for unmanned surface vehicle based on trajectory unit. Ocean Eng. **151**(151), 46–56 (2018)