Transactions on Computer Science and Intelligent Systems Research
ISSN: 2960-1800, eISSN: 2960-2238 | Volume 5
AIDML 2024

# Research Process of Path Planning based on RRT Algorithm and Its Improvements

Wenkang Wu *

Collage of Engineering, China Agricultural University, Beijing, China

* Corresponding Author Email: 2021307150412@cau.edu.cn

**Abstract.** At present, path-planning algorithms in the field of robotics is a problem of research value. Especially in the field of robotics, it has great significance. A good algorithm can improve the efficiency of the robot and ensure the safety of the robot in complex environments. In addition, the path planning algorithm can improve the autonomy and intelligence of the robot, help the robot to realize autonomous navigation and decision-making, and through different algorithms, the robot can also play different roles in different fields. In the past few years, many relatively perfect algorithms have been formed and used. Such as A* algorithm, Q-learning, ant colony algorithm, RRT algorithm, etc. After consulting the relevant literature, the author mainly researches the applications of RRT algorithm and its improved algorithms (RRT*, RRT*-smart, Informed RRT*). There are also some examples of combined algorithms in this article. Therefore, the author will focus on RRT algorithm and its improvements and analyze the common target of these algorithms through examples and comparisons, which is to find a shorter and more accurate path. In addition, the algorithms also need to run faster. In the end, the advantages, and disadvantages of these algorithms in different environments and fields are summarized. In the future research, scientists can further optimize the performance and applicability of these improved algorithms for specific fields or application scenarios. Resulting in better improvements to solve path planning problems, improve efficiency and reduce costs.

**Keywords:** Path-planning; RRT; RRT*; RRT*-smart; Informed RRT*.

## 1. Introduction

From the creating of the computer, automation control and robotics have had a very important impact on human beings in agriculture, transportation, machinery manufacturing, etc. From the initial motion planning problem (Move a piano from one room to another room and collision is banned), scientists began to build algorithms. With the fast development of robotic, path-planning is getting much more important applications in various fields. In the field of autonomous vehicles, Q-learning algorithm is important in the safety and efficiency of vehicle obstacle avoidance and path planning [1, 2]. Besides, in the field of robotics and automatic control, Rapidly-exploring Random Tree algorithm (RRT) is a typical path-planning method, which is widely utilized in helping robot navigate obstacles and find a better route. For instance, there are already wheelchairs on the market that can intelligently plan paths based on RRT algorithm and its improvements to better help people with disabilities [3]. Compared with other algorithms, RRT algorithm has strong adaptability to the location environment and is suitable for space of higher dimension. However, there are still some limitations in facing complex environment and dynamic obstacles [4]. Thus, exploring and improving RRT algorithm are important and have theoretical significance and application value. In short, the RRT algorithm still needs to be improved.

Rapidly-exploring Random Trees algorithm (RRT) was first pointed out in 1998. It is a search algorithm which is based on probability. In a randomly environment, the algorithm designs an optimal path from initial point to the goal point, and the searching progress is just like a tree grows branches by random sampling. Just as the author's statements, RRT algorithm is a simple and effective method for formulate route not only in two dimensions but also in three dimensions because it has few parameters, simple obstacle modeling, strong searching ability and simple structure. However,

because it is random sampling, it cannot converge to the most optimal path [5]. To work out this limitation, more new algorithms based on RRT are proposed. One of these improvements is RRT*, it can find a shorter and smoother path, which is an optimal path in the whole environment instead of partial environment by backtracking and backdating [3]. Based on RRT*, RRT*-smart is formed, which optimizes on the path found by RRT* to make it as straight as possible. Besides, RRT* -smart also introduces the smart sampling method, which is neither random sampling of RRT*, but sampling over a range [5]. Another more precise algorithm is Informed RRT* algorithm, The core of this algorithm is to use a suitable ellipse region to include a path that can avoid obstacles to reach the end point. The sampling space is included in the range of the ellipse, and the region of the ellipse is reduced again with each shortening of the path, and a good planned path is obtained in the same time as possible through continuous iterative convergence [6].

This article aims to give a comprehensive review in path planning of RRT algorithm and its optimized algorithms. The author will focus on the basic principle of RRT algorithm, RRT* algorithm, RRT*-smart, Informed RRT* and give some examples of their applications in formatting a path for robot to get to the target point and avoid the surrounding obstacles safely and efficiently. The author will also give a deep analysis in their advantages and disadvantages, as well as the range of adaptability. In addition, this article will discuss the challenges and the future directions of RRT algorithm and its improvement in solving path planning problems of complex environment and obstacle. In the future of artificial intelligence and automatic control, how to apply RRT and its improved algorithms to complex real situations (vehicle planning in logistics management, drone flight trajectory design, submersible to avoid ocean currents and reefs) and find an efficient and simple path is quite important [5, 7].

## 2. RRT Algorithm and Its Improvements

### 2.1. RRT algorithm

It has been mentioned in the introduction part that the RRT algorithm was initiated by professor LaValle in the 1990s. RRT algorithm can efficiently and simply plan paths in a multidimensional environment. RRT searching is very similar to how trees grow and their branches spread out. The following describes the principle of RRT algorithm: Consider the starting point to be the root node, and then use random sampling to construct many leaf nodes from the base node. From these leaf nodes, form a random extended tree until a leaf node in the random tree reaches the goal point. The random tree may then be used to select a path between the beginning and target points. Of course, if the contact between the leaf node and its parent node passes the obstruction during the growth process, the growth is considered illegitimate. The pseudo-code of the RRT algorithm and its annotations was shown in the Figure 1.

In the process of running the code, the last step is that go back to the random tree, starting from the target point to mark the last parent node, and then continue the loop process until the root node is marked (found). Thus, a path is formed from the initial point to the goal point and there is not hitting any obstacles [8].

Here are the parameters required by the RRT algorithm:

The configuration space itself, including: Initial and target points, Obstacles (including kinematic constraints).

Expansion step: the maximum distance that the tree expands from on parent node to its neighbors in each iteration.

Maximum number of iterations (expansions).

Decision parameters: parameters that affect the selection of the next expansion node, may encourage goal-oriented and exploration behavior.

```
Input: M, x_init, x_goal
Result: A path L from x_init to x_goal
T. init();
The pseudo-code for the above paragraph describes the initialization start and target point.
For i = 1 to n do
    x_rand ← Sample(M);   First, a random point (x_rand) is generated.
    x_near ← Near(x_rand, T);   Then find the nearest point on the tree as x_near.
    x_new ← Steer(x_rand, x_near, StepSize);   Set the step value (StepSize). Then take the point
between x_rand and x_near as x_new.
    E_i ← Edge(x_near, x_near);
    if CollisionFree(M, E_i) then
        T.addNode(x_near);
        T.addEdge(E_i);   Finally, a collision checking is needed to check if the generated
                          point will collide with an obstacle or not after it connects with x_near.
    If x_new = x_goal then
        Success()
```

**Fig. 1** The pseudo-code and annotation of the RRT algorithm (Photo credited: Original)

From the code and the parameters, comparing RRT algorithm with other algorithms, the advantages of RRT are summarized as follows: RRT algorithm has fewer parameters, its configuration is simpler, and its ability of searching is better. Besides, to make the process of path planning more efficient and more rapid, RRT algorithm can be easily combined with other algorithm. Also, it can work out the problems of high dimensional and complex environment. However, in the later running process of RRT algorithm, the rate of using nodes is low, as well as the calculation amount is too large, so it will consume more time. And because of random sampling, its path will be unstable. However, in the later process of running RRT algorithm, the rate of node utilization is low. And it will spend more time to calculate and figure out the complex environment. And because of random sampling, it may occur the condition of path instability. To sum up, the RRT algorithm is suitable for global offline and local online path planning, and can complete high-dimension and complex path planning when resources are sufficient [8]. Therefore, RRT algorithm has important research significance in solving the motion planning of unmanned vehicles. It can solve the feasible path of unmanned vehicles under kinematic constraints through some methods based on the generation of parameterized nodes in the guidance domain, and quickly cope with the changing environment, that is, update the path in real time during driving. And it also has a certain degree of universality, can cope with different traffic driving situations. It provides strong support for the wide application of autonomous vehicles [9].

In conclusion, RRT algorithm has a great impact on path planning, has a wide application prospect, and is suitable for many aspects. However, because of its random sampling principle, the shortest route cannot be accurately found, and there will be a lot of unnecessary calculation, thus affecting the algorithm running speed, making the whole path planning process inefficient. Therefore, RRT algorithm needs to be further explored, to make greater contributions to the development of the path planning field.

## 2.2. Improvements of RRT Algorithm

It is of huge challenge to further improve the RRT algorithm. Because the random of paths which were founded by the RRT algorithm, that is, the tree growth starts from the starting point and ends as soon as the end point is found, it cannot guarantee that the path found is the shortest. To further simplify the path planning process, make the calculation process of the initial RRT algorithm more efficient, and make the RRT algorithm have a wider application range, more and more improved algorithms have been proposed by scientists. According to the time order, the author introduces the principle, advantages and disadvantages, and applications of three improved algorithms RRT*, RRT*-smart and Informed RRT*.

### 2.2.1. RRT* Algorithm

Frazzoli and Karman pointed out RRT* algorithm in 2001 [10]. The RRT* algorithm is to find an optimal path. The search process of RRT* algorithm is the same as that of RRT, that is, the target

point is found in the process of tree growth is the same. However, RRT* has two major improvements in the later period. Based on RRT algorithm, RRT* algorithm makes the following improvements. The first one is that the process of reselecting the parent node in the RRT* algorithm. In RRT algorithm, target point ends up being connected to its closest point on the tree, but that may result the path is not the shortest one. Therefore, after the random tree grows to the target point, RRT* algorithm takes it as the center of the circle, specifies a radius, and chooses a better parent node within the sphere of the circle, and then search in loop the parent of the parent node until find the best path. Another improvement is rewiring, in which the original connection is removed and the new parent node is went together after a more suitable parent node has been selected.

The pseudocode for RRT* selecting parent nodes and rewiring was shown in the Figure 2.

$X_{near}$ ← nearest_neighbors(T, $x_{near}$, $r_i$) Find $X_{near}$ within the specified circle.
**for all**($x_{near}$, $X_{near}$ )**do**
  rewire_RRT*( $x_{near}$, $x_{new}$)
**for all**($x_{near}$, $X_{near}$ )**do**
  rewire_RRT*( $x_{new}$, $x_{near}$)  The rewiring process
RRT* locally rewires nodes in T using a connection radius of

$$r(i) \approx \gamma(d) \cdot \left(\frac{\log i}{i}\right)^{1/d}$$

**Input**: A new potential parent $x_{potential\_parent}$ to child node $x_{child}$
**Output**: Updated tree T
**if** (collision_free($x_{potential\_parent}$, $x_{child}$) )**then**
  c ← cost($x_{potential\_parent}$, $x_{child}$)
  **if** (cost T($x_{potential\_parent}$) + c < cost T($x_{child}$))**then**
    T.parent($x_{child}$) ← $x_{potential\_parent}$

**Fig. 2** The pseudo-code and annotation of the RRT* algorithm (Photo credited: Original).

RRT* specifically generates the path as follows:

1) Reselecting a parent node is to set a range near node xnew and find an area with r as radius c as center, and find a point within this area to replace xnew's parent node. After finding nodes, it calculates whether the total cost of replacing from starting point to neighboring nodes with xnew is lower than the original total cost in order [11].

2) The rewiring process follows the reselection of parent nodes, which aims to decline the cost between the nodes in the random tree [11].

The RRT* algorithm has following advantages. First, it has a certain optimality, because it is an improved RRT algorithm, it also considers the length of the path when finding a path, that is, it selects a shorter path, to generate an optimal or almost optimal path. Second, RRT* has a wide range of applications, which is the same as RRT algorithm for a variety of environments, including unknown, partially known, dynamic, etc. However, since the RRT* algorithm needs to consider the path's length while searching the path, the calculation is relatively large, which may influence the performance of the algorithm. Moreover, it is sensitive to the parameters: the results of the RRT* algorithm also relay on the choice of the parameters, which may result in the degradation of the algorithm performance if the parameters are not chosen properly.

RRT* algorithm also has many applications in the field of automation, which realizes more efficient and accurate path planning than RRT by optimizing node selection and path construction. In intelligent wheelchair applications, RRT* can help the wheelchair to plan the kinematic constraints and help it navigate in complex environments, to achieve accurate path trajectory control. By applying the RRT* algorithm in the trajectory planning of intelligent wheelchair, it can effectively avoid obstacles and even moving pedestrians, make the intelligent wheelchair solve the problems by using less time and promote the effectiveness. This provides a great help for the intelligent wheelchair to assist the disabled and elderly people and make the path smoother.

### 2.2.2. RRT*-smart algorithm

RRT*-smart is a further improvement of RRT* proposed in 2013 [12]. Although RRT* can converge asymptotically to the optimal solution, it converges very slowly. RRT*-smart gives path optimization and smart sampling methods based on RRT*, to improve the speed of convergence of path optimization. The core idea is that after an initial path is obtained, starting from the target point, it connects its parent node in the opposite direction, looking for the node with lower cost, and stops the connection until the collision is calculated or the cost becomes higher. Then it starts from the stopped node and connects directly to its new parent node in the reverse direction. In the process of iteration, the path tree is updated if it is less costly to find a new parent node to connect with this node. This new parent node is called Xbeacons. After finding Xbeacons, several points called Xrand will be sampled within the radius of the Rbeacons centered at that point. However, in the subsequent iterations, the RRT*-smart algorithm is not randomly sampled but added to the smart sampling method. In general, RRT*-smart is a process of optimizing the path and straightening the curve. The pseudocode and its annotation for RRT*-smart was shown in the Figure 3 and Figure 4 [12].

```
T ← InitalizeTree();   Initializing the tree
T ← InsertNode(0, Xinit, T);   Insert the first tree node, the initial node(starting point)
for i = 0 to i = N do   The loop iterates N times
  if i = n + b, n + 2b, n + 3b ... then   When beacons are available, bias sampling is
                                          performed at a constant b time interval.
                                          Where, the Biasing Ratio b is as follow.
                                          Biasing Ratio = (n / x_free) * B
    Xrand ← Sample(i, Xbeacons);   Bias sampling
  else
    Xrand ← Sample(i);   No bias sampling
    Xnearest ← Nearest(T, Xrand);   Find the node Xnearest to Xrand in the tree T
    (Xnew, Unew, Tnew) ← Steer(Xnearest, Xrand);   A new node Xnew is generated from Xnearest to Xrand
                                                   according to the control Unew when the
                                                   constraints are satisfied
  if Obstaclefree(Xnew) then   To find if Xnew can pass, perform the following steps if there
                               is no collision
    Xnear ← Near(T, Xnew, |V|);   To search some neighboring nodes Xnear in the tree T within
                                  a certain range of Xnew
```

**Fig. 3** The pseudo-code and annotation of the RRT*-smart algorithm (Photo credited: Original)

```
    Xmin ← Chooseparent(Xnear, Xnearest, Xnew);   Calculate the total cost of the path from Xinit
                                                  through Xnear to Xnew, and choose the
                                                  neighbor node Xmin with the minimum cost
                                                  as the parent of Xnew
    T ← InsertNode(Xmin , Xnew ,T);   Insert Xnew into the tree T and change Xmin to be the
                                      parent of Xnew
    T ← Rewire(T, Xnear, Xmin, Xnew);   Adjust tree T path, neighboring nodes try to take Xnew
                                        as the parent and calculate the cost, if the total cost is
                                        low, then update Xnew as the parent
  if InitalPathFound then   If the initial path leading to the target point is found
    n ← i;   Returns the current iteration number to n
    (T, directcost) ← PathOptimization(T, Xinit, Xgoal);   Compute the optimized path from Xinit
                                                           to Xgoal and return its cost by directly
                                                           connecting the nodes visible to each
                                                           other in the path
  if directcostnew < directcostold then   If the cost of optimizing the path is less than the
                                          original one
    Xbeacons ← PathOptimization(T, Xinit, Xgoal);   Generate Xbeacons
return T;
```

**Fig. 4** The pseudo-code and annotation of the RRT*-smart algorithm (Photo credited: Original)

In conclusion, RRT*-smart has the following advantages: it can make the final route shorter and smoother. Redundant inflection points are removed so that fewer path inflection points are obtained. However, under the same sampling times, the effect of smart sampling is not ideal, the convergence rate of the path is reduced due to node aggregation, so the inflection point position will be far away

from the obstacle, and the Angle at the inflection point is too large, so it is easy for robot escape when going through the path. Moreover, the optimization effect of the path is not very good, and the path quality may deviate from the optimal solution.

At present, RRT*-smart is helpful for path planning of USV in complicated and dynamic environments, and it is improved under smart sampling, which uses an elliptical sampling point range to avoid invalid sampling. In the face of dynamic obstacles, a constraint on the Angle at the node is added to satisfy that the path is smooth enough [13]. This method makes USV in the path planning efficiency, optimization, security has been greatly improved.

### 2.2.3. Informed RRT* Algorithm

Informed RRT was proposed in 2014. Informed RRT is a heuristic way of planning path for robot algorithm based on sampling method, which aims to generate shorter and more optimal paths by intellectively selecting nodes [14]. The Informed RRT* algorithm is a process-optimized algorithm that aims to optimize the sampling of RRT*. It uses an ellipse that iteratively converges for sampling, that is, the sampling points are in the ellipse rather than in the whole environment. In Informed RRT*, its ellipse has two focal points: the initial point and the goal point. Focal length is $c_{min}$, major axis is $c_{best}$ (the length of the initial path) and minor axis is b. As shown in the Figure 5, these are the parameters of the ellipse. In each later iteration, after each shorter path is traversed, the secondary path is taken as the new major axis, from which the shape of the ellipse is updated [14].

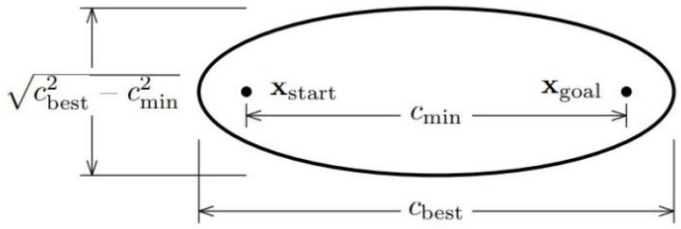


**Fig. 5** Schematic diagram of the ellipse parameter solution [15]

The pseudocode and annotation for the Informed RRT* algorithm was shown in the Figure 6.

```
if c_best < ∞ then
    c_min ← ||x_goal − x_start||_2;   Find c_min (The length of the path from start point to target point)
    x_centre ← (x_start + x_goal)/2;   Determine the center of the ellipse
    C ← RotationToWorldFrame(x_start, x_goal);   Finding the rotation matrix
    r_1 ← c_max/2;   Find the semi-major axis
    { r_i }_{i=2,...,n} ← (√(cbest^2 + cmin^2))/2);   Find the semiminor axis of the iteration in turn
    L ← diag{ r_1, r_2, ..., r_n};
    x_ball ← SampleUnitNBall;   Samples are taken from the unit circle
    x_rand ← (CLx_ball + x_centre) ∩ X;   scaled and rotated into the world coordinate system
else
    x_rand ~ u(X);
return x_rand;
```

**Fig. 6** The pseudocode and annotation for the Informed RRT* algorithm (Photo credited: Original)

Informed RRT* has the following advantages and disadvantages. Informed RRT* has a certain target-oriented, high search efficiency and wide application range. It can quickly find the goal point in the process of building the path, and through the imported information, it can select nodes more intelligently, to generate shorter and better paths faster and more efficiently. However, Informed RRT* has higher requirements on the environment to make the robot get certain environmental information more accurately. The parameters of this algorithm impact the effectiveness and smoothness terribly, such as the improper step size, the samples will lead to the degradation of the running performance of the algorithm [15].

Intelligent robots based on Informed RRT* algorithm can complete more accurate work. At present, an anti-collision path planning algorithm for head and neck surgical robots has been designed. The algorithm also incorporates a regression filtering mechanism to prevent local optima that may occur in the path planning process [16]. Informed RRT* can also be used in many other applications, such as autonomous driving, logistics distribution, and drone flight, making the process of selecting the better path and operation safely and effectively.

## 3. Discussion

In the facets of speed in generating paths, the initial RRT algorithm is relatively slow because its random sampling-based procedure means that each extension may not be toward the goal, increasing the time needed to find a valid path. However, RRT*, RRT*-Smart and Informed RRT* which are promoted from RRT significantly improve the effectiveness in path generation through introducing heuristic search and optimization strategies.

In the facets of efficiency, RRT* algorithm improves the quality of the path by introducing the idea of local optimization, that is, it can process the generated path again after each time to remove redundant nodes. RRT*-Smart further combines the idea of Smart (Sampling-based Roadmap Augmentation with Refinement Trees), and further improves the efficiency and quality of path generation through local resampling and refinement process. Informed RRT* introduces the concept of sampling region, which only samples in the region that may contain the optimal solution, thus greatly improving effectiveness of searching of the algorithm.

Since RRT* and its modified algorithms continuously optimize in the process of generating paths, the generated paths are usually shorter than the traditional RRT algorithm. This is because these algorithms can use the known information more efficiently and avoid generating unnecessarily long paths. In addition, because they are optimized after generating the path, the generated path is usually smoother, which reduces the abrupt changes and unnecessary steering during the robot motion, which is conducive to the smooth movement of the robot.

In conclusion, the promoted algorithms RRT*, RRT*-SMART and Informed RRT* outperform conventional RRT on terms of speed, efficiency, path length and path smoothness in generating paths. RRT*, RRT*-SMART, and Informed RRT* are three sample-based path planning algorithms commonly used in robot path planning and navigation. RRT improves the path quality by optimizing the structure of the tree, but it is computationally expensive. RRT-SMART improves the efficiency by introducing heuristic search and node optimization. Informed RRT* uses prior information to limit the search space and further improves the efficiency. In application, the three are widely used in unmanned driving, unmanned aerial vehicle and other fields. Promising, but the trade-off between computational efficiency and path optimization still needs to be addressed.

In summary, the four algorithms have their own advantages and disadvantages, and the appropriate algorithm should be selected according to the application scenarios and requirements.

## 4. Conclusion

This article reviews RRT based path planning algorithms, introduces the basic principle of RRT and three improved algorithms: RRT*, RRT*-smart, Informed RRT*. RRT and its improved algorithms have a profound impact in the field of path planning, which provides a basic idea for future generations to propose new improved algorithms and the integration of multiple algorithms. To solve the problems of locally optimal solution and large amount of calculation of RRT, to improve the effectiveness and accuracy of planning paths for robots to a certain environment, these improvements of RRT algorithm were created by lots of scientists. However, author found that these improved algorithms have their own advantages and disadvantages in different environments and application scenarios, and they cannot completely find the optimal route. Future research directions can further optimize the performance and practicability of these algorithms for specific fields or application

scenarios. For example, improving the characteristics and constraints of specific robots or unmanned vehicles, combining machine learning and various algorithms for promoting the accuracy and smoothness of path-planning. Thus, the effectiveness of computer and resource consumption are declined. Although RRT algorithm has been developed for more than 20 years, and the results and models are very simple, it still has great research value, which is worth scientists to continue to explore in the future, scientists can continue to optimize the algorithm, and combine other algorithms with RRT and its improvements, to apply path planning to more fields, to further improve the operational efficiency of human society.

## Reference


[1] Liu Xiaomeng. Improvement of Q-learning Reinforcement Learning Algorithm and Its Application in Unmanned Vehicles Path Planning. 2018.

[2] Hou, K., Lan, X., Zhang, Y., & Tyagi, S. K. S. Path Planning Based on A* Algorithm for Unmanned Surface Vehicle. Advances in Intelligent Systems and Computing, 2018, 783–788.

[3] Huang Hui, Sun Mengxue, Tan Xiaoyin et al. Path planning for smart wheelchairs with limited RRT sampling area [J/OL]. Mechanical and Electrical Engineering Technology. 2024:1-6.

[4] Xu Liang. Research on Robot Path Planning based on improved RRT~* and Q-learning algorithms [D]. Jilin university, 2023.

[5] Li Yikun. RRT path planning algorithms [D]. Liaoning university of science and technology, 2023.

[6] Gammell J D., Srinivasa S S, Barfoot T D. Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic[C]//2014 IEEE/RSJ International Conference on Intelligent Robots and Systems. Chicago, USA: IEEE, 2014: 2997-3004.

[7] Lan Wei. Research on path planning of underwater glider formation under the influence of ocean currents [D]. Dalian maritime university, 2023.

[8] Chen Qiulian, Jiang Huanyu, Zheng Yijun. A review of Fast expanding Random Tree Algorithm for Robot Path Planning [J]. Computer Engineering and Applications, 2019, 55(16):10-17.

[9] Feng Laichun. Research on motion planning algorithm of parameterized RRT unmanned vehicle based on guidance domain [D]. Hefei: University of Science and Technology of China, 2017.

[10] Li, Y., Wei, W., Gao, Y., Wang, D., & Fan, Z. PQ-RRT*: an improved path planning algorithm for mobile robots. Expert Systems with Applications, 2020, 113425.

[11] Luo Haitao, Sun Jiazi, Gao Pengyu et al. Based on improved RRT ~ * algorithm of intelligent wheelchair global path planning research [J]. Journal of instruments and meters, 2023, 44(10): 303-313.

[12] Nasir J, Islam F,Malik U, et al. RRT*-SMART: A Rapid Convergence Implementation of RRT*[J].International Journal of Advanced Robotic Systems, 2013, 10(7):991-996.

[13] Dong Lu, Xiong Ailing. Path Planning of unmanned Vehicle in Complex dynamic Environment Based on improved RRT~*-Smart [J]. Journal of Intelligent Science and Technology, 2012, 4(02):264-276.

[14] Gammell D J, Srinivasa S S, Barfoot D T. Informed RRT*: Optimal Incremental Path Planning Focused through an Admissible Ellipsoidal Heuristic. [J]. 2014.

[15] Jia Haoduo, Fang Lijin, Wang Huaizhen. Adaptive path planning of manipulators combining Informed-RRT* with artificial potential field. Computer Integrated Manufacturing Systems: 1-21.

[16] Li Guoqi, Hu Zhi, Chen Zhiyu et al. For head and neck surgery dual-arm robot collision path planning [J/OL]. Electronic science and technology: 2021, 1-9.