International Journal of Multiphysics
Volume 18, No. 4, 2024
ISSN: 1750-9548

Obstacle Avoidance Path Planning Research for Robotic Arm of the Power-carrying Operation Robot based on Improved RRT Algorithm

Bing Bai[1], Yanwei Wang[1]*, Jian Liang[1], Zhengchao Xu[1], Wenjian Xu[1], Liang Wang[1],

State Grid Inner Mongolia Ultra High Voltage Company, Hohhot, 010000, 1229089737@qq.com [1]

Abstract

The robotic arm of the power-carrying operation robot faces path planning challenges in complex distribution network environments. Addressing these challenges is crucial for optimizing the robot's performance and efficiency., this paper proposes an improved Rapidly-exploring Random Tree (RRT) obstacle avoidance algorithm. The algorithm improves the efficiency and accuracy of path planning by introducing a dynamic sampling function. This function allows for the dynamic adjustment of sampling points based on the distribution of obstacles. Combined with the cost function of the A* algorithm, the path is further simplified and smoothed to reduce the inflection points and optimize the motion trajectory of the robot's robotic arm. The simulation results verify the efficiency of the algorithm in reducing the path planning time and path length, in which the number of sampling points is reduced by 70.3% and the planning time is shortened by 68.3% in the 3-dimensional(3D) simulation, which proves its effectiveness in the field of power-carrying operation robot.

Keywords: Improved RRT; Path planning; Dynamic sampling; Robotic arm; Obstacle avoidance

1. Introduction

With the rapid development of the power industry and the increasing demand for efficient automated systems by power professionals, more and more power operation robots equipped with robotic arms are being deployed to the front lines of power inspection and maintenance. These robots replace power workers to efficiently complete high-precision or special tasks in harsh environments, significantly improving the operational safety of power transmission and distribution and reducing the labor intensity of power workers. The path planning problem occupies a central position in the field of robotic arm research. It involves designing an efficient and safe trajectory for the robotic arm to navigate from an initial position to a target position, ensuring that the arm can avoid various obstacles during the movement process [1-2]. In the complex environment of the distribution network, the ability of the robotic arm to quickly plan a short and smooth path is crucial for power-carrying operation robot to complete their tasks [3].

Traditional robotic arm path planning algorithms include A* (A-Star), Dijkstra, PRM (Probabilistic Roadmaps), RRT (Rapidly-exploring Random Trees), and others [4]. However, the complex power grid operation environment makes it difficult for existing traditional algorithms to efficiently complete tasks in actual operations. The sampling-based path planning algorithm RRT is simple in structure and has strong exploration capabilities in unknown environments, making it extremely suitable for application in high-dimensional spaces [5]. However, traditional RRT has issues such as poor guidance, redundant nodes, non-optimal paths, and unsmooth generated paths [6]. Many domestic and foreign scholars have proposed improvements to address these issues.

Qureshi et al. proposed the IB-RRT* algorithm for complex environments, which optimizes the optimal path in complex environments but performs poorly in narrow areas [7]. JEONG et al. proposed the Quick-RRT* algorithm, which expands the preset range of parent nodes in the path to reduce path costs [8]. Jia Haoduo and his

team effectively integrated the artificial potential field method with the Informed-RRT* algorithm, significantly improving the speed of path planning [9]. Wang Yang et al. proposed the DG-RRT algorithm, which improves the search efficiency of the RRT algorithm in complex environments through directional guidance, but did not address the applicability of higher-degree-of-freedom robotic arms [10]. Han Kang et al. proposed the Obi-RRT algorithm, which introduces intelligent algorithms to reduce high-cost path points [11]. Li Chuangye et al. introduced Sobol sequence sampling to make the distribution of sampling points more uniform, but this also led to low algorithm efficiency in complex environments [12]. In exploring the RRT algorithm to obtain the optimal path, Karaman et al. optimized by reselecting parent nodes and pruning operations, aiming to find and establish the optimal path [13]. Lai Yong et al. summarized the spatial planning algorithms of robotic arms, indicating that the introduction of B-spline in trajectory planning is an inevitable trend [14]. In the complex environment of the distribution network, The presence of numerous obstacles and the relatively limited operational space of the robotic arm pose significant challenges to performance optimization. These factors have hindered the effectiveness of many improved RRT algorithms, preventing them from meeting expectations. Typical problems involve redundant nodes and discontinuity in the generated paths.

Combining the above research results, this paper proposes an improved RRT algorithm to adapt to the complex working environment of the distribution network. Addressing the blind and random nature of traditional RRT algorithms in searching random nodes across the entire map, and the issue of generating random points in obstacle-free areas in complex maps that reduces search efficiency, we utilize a dynamic sampling function strategy based on obstacles to improve search efficiency in complex environments and reduce search time. On the other hand, by introducing the A* cost function strategy, we solve the problem of non-optimal paths in traditional algorithms and reduce path length. Finally, by deleting redundant nodes and fitting the polyline with a Cubic B-spline curve, we ensure the continuity and smoothness of the generated path to the greatest extent. Through Matlab simulation, this paper confirms that the proposed improved RRT algorithm completely outperforms the conventional RRT algorithm.

2. Robotic arm modeling and collision detection

2.1 UR10 Kinematic Modeling

Due to the practical needs of the project, this paper selected the UR10 robotic arm with 6 degrees of freedom as the research object [15]. The UR10 robot adopts a Six-Degree-of-Freedom (6-DOF) serial structure, capable of performing high-intensity automated tasks with a load capacity of up to 10 kg and an extension radius range of 1300 mm, providing a large working range. This robot not only has collaborative capabilities but also high safety, making it very suitable for the power distribution network environment.

The traditional Denavit-Hartenberg (D-H) parameter method was used to establish the kinematic modeling of the 6-DOF robotic arm. This method mainly involves determining the coordinate transformation relationship between adjacent joints of the UR10 robotic arm. The schematic diagram of the linkages using the D-H method is shown in Figure 1.

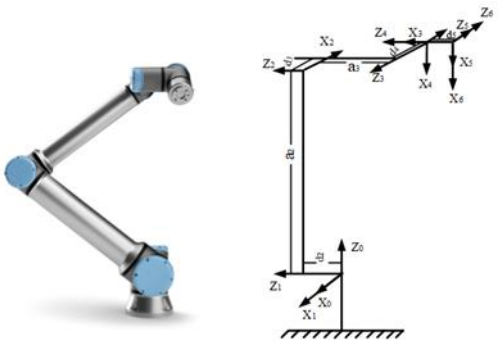


Figure 1. Schematic diagram of the relationship between the connecting rods of the DH method

Since the UR10 robotic arm only has variable joint angles θ, while the other three parameters are fixed values, the D-H model parameters are shown in Table 1.

Table 1. UR10 robotic arm model D-H parameters

Joint i	α /°	a/mm	d/mm	θ /°	Joint range
1	-90°	0	0	θ_1	-360~360
2	-90°	a_2=612.90	d_2=176.0	θ_2	-360~360
3	-90°	a_3=572.30	d_3=	θ_3	-360~360
4	-90°	0	0	θ_4	-360~360
5	-90°	0	d_5=10.96	θ_5	-360~360
6	-90°	0	0	θ_6	-360~360

Based on the coordinate systems of the robot's links and the parameter table of the D-H model, the homogeneous transformation matrix between the link coordinate systems can be derived as follows:

$$\begin{aligned} \boldsymbol{i}-_{i}^{1}T &= Rot(2,\theta_i)Trans(2,d_i)Trans(x,a_i)Rot(x,a_i) \\ &= \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & \alpha_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\sin\alpha_i & -\cos\theta_i\sin\alpha_i & \alpha_i\sin\theta \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \quad (1)$$

The end-effector position and orientation of the robot can be solved by substituting the D-H parameter given in Table 1 into Eq. (1) and multiplying them using the transformation matrices of each linkage:

$$\begin{aligned} \frac{9}{6}T &= \frac{9}{1}T\frac{1}{2}T\frac{3}{3}T\frac{4}{4}T\frac{5}{6}T \\ &= \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \quad (2)$$

Where n_x, n_y, n_z, o_x, o_y, o_z, a_x, a_y, a_z are the end-effector orientation components of the robot arm, and p_x, p_y, p_z are the end-effector position components of the robot arm.

In practical application scenarios, the end-effector position and orientation of a robotic arm is often specified, and the angle variables of each joint required to achieve that position are calculated. Subsequently, these angle variables are used to drive the robot towards the target position. This process is defined as inverse kinematic analysis of the robot [16]. Since the second, third and fourth joint axes of the UR10 robot are in parallel, the conditions for the existence of an inverse kinematics are satisfied. Therefore, the numerical solution method can be used to solve the inverse kinematics of this robot and the results are shown below:

$$\theta_1 = \arctan\left(\frac{p_y}{p_x}\right) - \arctan\frac{(d_2+d_3)^2}{\sqrt{r^2-(d_2+d_3)^2}}$$

$$\theta_2 = \arctan\frac{c_1o_x + s_1o_y + c_5c_6\dfrac{c_1a_x+s_1a_y}{s_5} + c_6\dfrac{a_z}{s_5}}{o_z - c_6\dfrac{c_1a_x+s_1a_y}{s_5} + c_5c_6\dfrac{a_z}{s_5}}$$

$$\theta_3 = \arctan\left(\begin{array}{c}\dfrac{-c_1s_2p_x - s_1s_2p_y - c_2p_z}{a_3} \\ +\dfrac{d_5\dfrac{c_1c_2a_z + s_1c_2a_y - s_2a_z}{s_5}}{a_3}\end{array}\right)$$

$$\theta_4 = \arcsin\left(\frac{-a_z}{s_5}\right) - \arctan\frac{c_1o_x + s_1o_y + c_5c_6\dfrac{c_1a_x+s_1a_y}{s_5} + c_6\dfrac{a_z}{s_5}}{o_z - c_6\dfrac{c_1a_x+s_1a_y}{s_5} + c_5c_6\dfrac{a_z}{s_5}} - \arcsin\left(\frac{-c_1s_2p_x - s_1s_2p_y - c_2p_z + d_5\dfrac{c_1c_2a_z + s_1c_2a_y - s_2a_z}{s_5}}{a_3}\right)$$

$$\theta_5 = \arccos(s_1a_x - c_1a_y)$$

$$\theta_6 = \arccos\left(\frac{c_1 n_y - s_1 n_x}{s_5}\right)$$

For solving the robot inverse kinematics, it is important to note that the inverse kinematics is not unique. In particular, when solving the inverse kinematics of the UR10 robot using a numerical solution method, up to eight different sets of solutions may be obtained. Therefore, a set of solutions must be selected based on the robot-specific constraints that best fit the current position and orientation requirements.

According to the inverse kinematics solution method for robotic arms, it is possible to convert the way the points of the robotic arm in path planning are described in the plane rectangular coordinate system to the required transformations for each link in the joint configuration space. The 6-DOF robotic arm selected in this paper is simulated in Matlab using the Robotics toolbox, as shown in Figure 2.

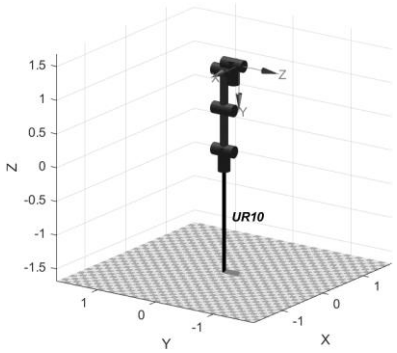


Figure 2. Schematic diagram of UR10 modeling

2.2 Collision detection

Due to the requirements of the working environment of the power distribution network, this paper uses an Bounding Box strategy for collision detection of the robotic arm, which involves the use of a cylinder to encapsulate the parts of the robotic arm and the use of a sphere to cover the obstacles in the space [17]. In order to simplify the computational complexity of collision detection,this method superimposes the radius R1 of the cylinder with the radius R2 of the sphere. Figure 3 shows the schematic diagram of the enclosing box method, where D represents the vertical distance from the center of the sphere to the axis of the cylinder. The method of robotic arm collision detection is shown in the following equation:

$$\begin{cases} D > R_1 + R_2, collision \\ D < R_1 + R_2, no\_collision \end{cases} \quad (3)$$

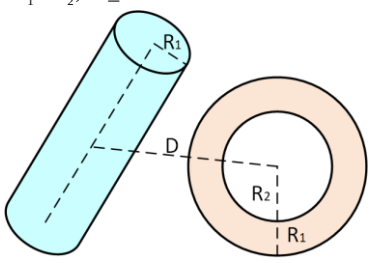


Figure 3. Collision detection model

3. Principles of Algorithms

The RRT algorithm, as a sampling-based probabilistic complete path planning method, eliminates the need for explicit modeling of the environment by randomly sampling and performing collision detection in the state space. This property allows the RRT algorithm to perform efficient searches in high-dimensional spaces in order to quickly determine a feasible path from the start point to the end point.

RRT Algorithm
RRT(Qstart, Qgoal, map) Initialize tree with Qstart as the root node

```
    While not reached Qgoal:
      Qrand = RandomPoint(map)
      Qnearest = NearestNode(tree, Qrand)
      Qnew = Steer(Qnearest, Qrand, stepSize)

      If Not ObstacleFree(Qnearest, Qnew, map):
        Continue

      Add Qnew to tree with Qnearest as parent

      If Distance(Qnew, Qgoal) <= stepSize AND ObstacleFree(Qnew, Qgoal, map):
        Add Qgoal to tree with Qnew as parent
        Return PathFromRootTo(Qgoal, tree)

    Return Failure

RandomPoint(map)
    // Generate a random point within the map boundaries
    Return Qrand

NearestNode(tree, Qrand)
    // Find the nearest node in the tree to Qrand
    Return Qnearest

Steer(Qnearest, Qrand, stepSize)
    // Move from Qnearest towards Qrand by stepSize
    Return Qnew

ObstacleFree(Qstart, Qend, map)
    // Check if the path from Qstart to Qend is free of obstacles
    Return True/False

Distance(Qpoint1, Qpoint2)
    // Calculate the distance between Qpoint1 and Qpoint2
    Return distance

PathFromRootTo(Qnode, tree)
    // Trace back from Qnode to the root to form the path
    Return path
```

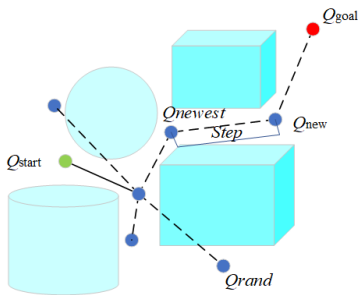


Figure 4. Schematic diagram of RRT algorithm

Let M denote the map, T denote the randomized tree, and n denote the number of iterations. However, the traditional RRT algorithm relies on its random sampling mechanism, which leads to its shortcomings in specifying the goal directionality, which in turn increases the time required to find an efficient path. As the algorithm iterates repeatedly, the likelihood of its tendency to fall into local optimal solutions increases significantly. In addition, the frequent occurrence of invalid nodes in the random tree exacerbates the path cost, and the instability of the

generated paths may cause unnecessary mechanical losses and energy wastage for robots performing precision tasks, such as tightening bolts in load distribution network environments under energized operating conditions. In view of these problems, it is necessary to improve the traditional RRT algorithm to enhance its efficiency and applicability.

4. Improved RRT Algorithm

4.1 Dynamic Sampling with Probability Distribution for Rapidly-exploring Random Tree（DS-RRT）

In order to improve the problem of irregular spreading and lack of performance of traditional RRT algorithms in complex environments, especially in distribution network scenarios with highly irregular and dense obstacle distributions, this paper proposes an obstacle-based dynamic sampling function optimization strategy. These improvements can increase the accuracy of path planning, reduce the computational complexity, and optimize the quality of paths.

In this paper, a more flexible dynamic sampling probability function is introduced, which takes into account not only the distance of the obstacles, but also the denseness of the obstacles as well as the relative position of the current exploration direction to the obstacles. The optimized probability function is:

$$P(d,\theta)=\frac{1}{1+e^{-\lambda(d-d_0+\alpha\cdot\cos(\theta))}} \qquad (4)$$

Where: d represents the distance to the nearest obstacle; θ represents the angle between the current exploration direction and the direction to the obstacle; d0 is a set distance threshold for adjusting the size of the obstacle radiation area; α is a weighting factor for adjusting the weight of the influence of the direction angle; and λ is a parameter for adjusting the sampling density.

When the sampling point is close to the obstacle (d<d0), the value of P(d,θ) is larger, which means that the probability of generating a new sampling point near the obstacle increases, thus improving the local optimization of the obstacle avoidance path. Through the cos(θ) term, this sampling function samples along the normal direction of the obstacle, i.e., it increases the probability of sample point generation when the exploration direction is the same as the normal direction of the obstacle, which helps to bypass the obstacles in the map directly. The sensitivity and directionality of the sampling strategy can be adjusted in different map environments by adjusting parameters such as λ, d0 and α to optimize the performance of path planning. The schematic diagram is shown in Figure 5.

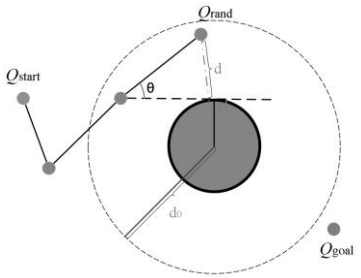


Figure 5. Schematic diagram of optimized dynamic sampling

Using this optimized dynamic sampling probability function, the RRT algorithm is able to adapt more intelligently to the distribution of obstacles in complex environments, effectively improving the sampling efficiency and path quality in the path planning process. In environments with dense or unevenly distributed obstacles, this strategy can significantly increase the generation of effective sampling points and accelerate the discovery speed of paths, while reducing unnecessary exploration through directional prioritization and directly pointing to effective paths, thus substantially improving the overall performance and application value of the algorithm.

4.2 Redundant nodes random point processing

The RRT algorithm after incorporating the dynamic sampling probability function processing effectively improves the search efficiency, although this reduces the time required for the search, the algorithm still produces

a large number of redundant random points. To further improve the efficiency, an approach incorporating the A* cost function search mechanism is used. By introducing the cost evaluation mechanism of A* to guide the sampling process, unnecessary computations were reduced and path quality was improved [18]. In this framework, the cost from the starting point to any node is defined as the forward cost of that node, i.e., as a heuristic evaluation function Q(i), while the distance from a random point to the target point is used as the backward cost H(i), and the Euclidean distance between two points is chosen as the heuristic function for the backward cost. Based on the traditional backward and forward cost functions of A*, weights are introduced, which can be adjusted according to the actual situation of the map to improve the efficiency of the algorithm. Accordingly, the integrated cost function of the node can be expressed as G(i), as shown in equation (5).

$$\begin{cases} G(i) = w_1 H(i) + w_2 Q(i) \\ Q(i) = \left| x_{\text{goal}} - x_{\text{rand}}(i) \right| \\ H(i) = \left| x_{\text{rand}}(i) - x_{\text{start}} \right| \end{cases} \quad (5)$$

w_1 and w_2 are the before and after cost weights.

With both w_1 and w_2 set to 0.5, as shown in Figure 6, a set of random nodes 10, 11 in obtained.

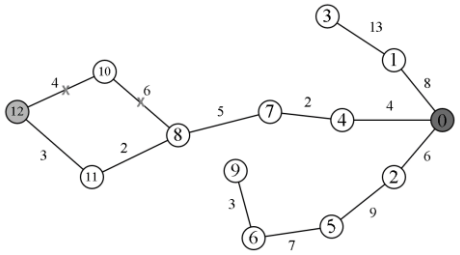


Figure 6. Schematic diagram of update node

4.3 Path Simplification & Path Smoothing

The paths obtained by the sampling strategy for dynamic obstacles and the improved RRT algorithm integrated with the A* cost function evaluation are continuous folded segments with poor path quality and more turning points, which can cause a large number of redundant nodes. The optimization of paths is particularly important due to the special characteristics of the transmission line environment, including the risk of working at height, the large distance between transmission towers, and the complex terrain and obstacles (e.g., trees, buildings, etc.) that may exist around. Among them, the smoothness of the path is directly related to the efficiency of the work and energy utilization of the power strip operation robot.

By removing the redundant nodes, the path obtained is shorter and has less curvature and fewer bends, the principle is shown in Figure 7. The path node vector Q[Q1,Q2,Q3.....Qn] obtained by RRT algorithm, collision detection for the path between nodes Qi and Qj, where i,k∈[1,2,3...n], and i>k, if there is no collision, all nodes between Qi and Qj are removed, and the simplified path is obtained by traversing all the nodes from the starting point Qstart, but there is still room for optimization of the local paths at its turning points.

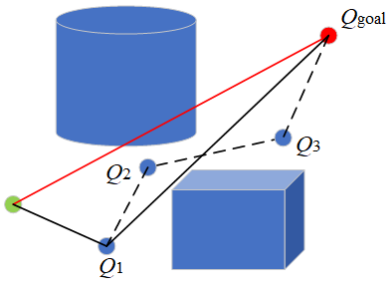


Figure 7. Path simplified schematic

Even after removing the redundant nodes, there are still redundant turns in the path. To further optimize these transitions, the inflection points of the path are finally smoothed using the Cubic B-spline curve technique.

$$P(t)=\sum_{i=0}^{3}P_iF_{i,k}(t) \qquad (6)$$

$$\begin{aligned} F_{0,3}(t)&=\frac{1}{6}(1-t)^3 \\ F_{1,3}(t)&=\frac{1}{6}(3t^3-6t^2+4) \\ F_{2,3}(t)&=\frac{1}{6}(-3t^3+3t^2+3t+1) \\ F_{3,3}(t)&=\frac{1}{6}t^3 \end{aligned} \qquad (7)$$

The $F_{i,k}(t)$ in Eq. (6) is the Cubic B-spline curve basis function, P_i is the control point of the curve, and t is the uniform node vector. The flowchart of the improved algorithm is shown in Figure 8.

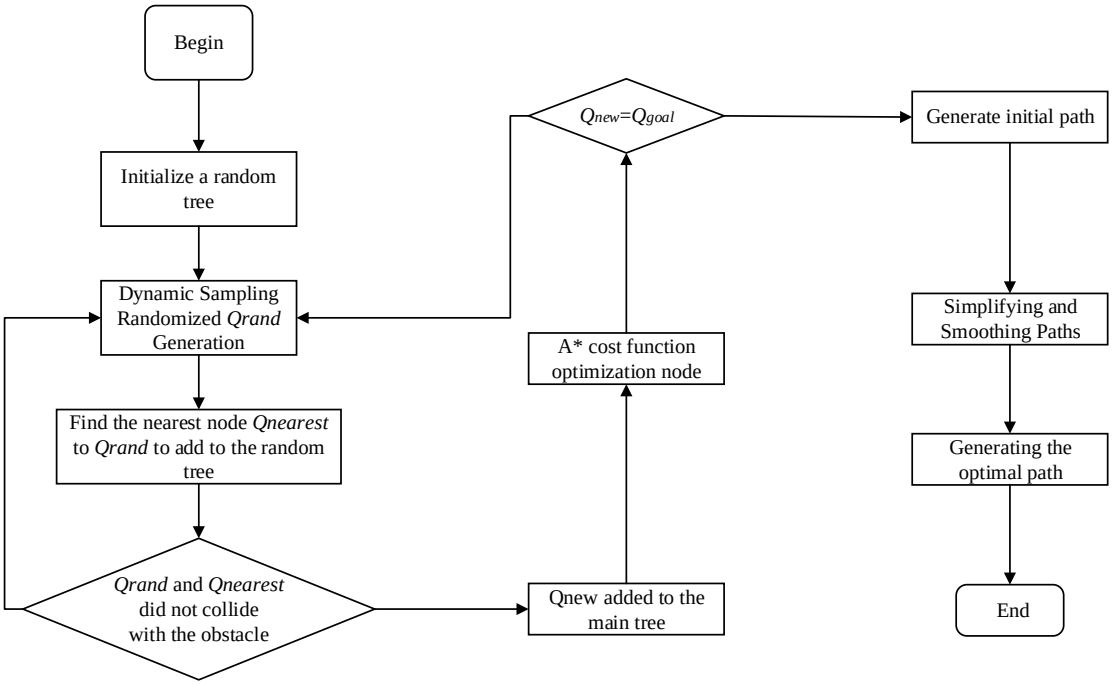


Figure 8. Flowchart of the improved RRT algorithm

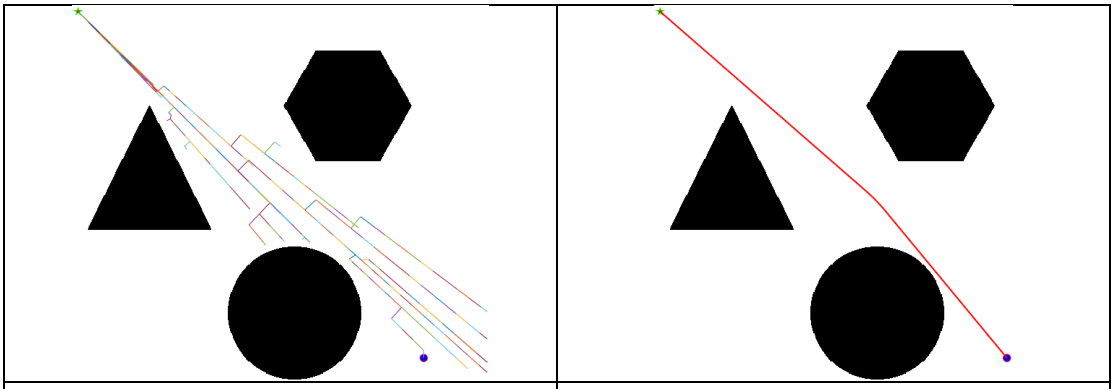
5. Simulation Experiment

In order to evaluate the practicality and efficiency of the improved algorithm proposed in this study, a series of simulation experiments are executed in this paper using Matlab software to compare and analyze the difference in path planning performance between the improved algorithm and the traditional RRT algorithm. The experiments in this paper not only cover the two-dimensional scene, in which the obstacles present various irregular shapes, but also extend to the three-dimensional space, and the obstacles are set to be spherical and cylindrical in order to increase the complexity and comprehensiveness of the test. Due to the inherent randomness of the RRT algorithm, and in order to objectively evaluate the performance of the algorithm, this paper repeats the simulation test 100 times under various conditions. The step size of the RRT algorithm is set to 20 in different test scenarios, both in 2D and 3D environments, in order to standardize the comparison criteria and facilitate data collection and analysis. The statistical results of the experiments are analyzed based on the average performance metrics of these simulation tests. All simulation experiments were done on a computer configured with Windows

11 operating system, Intel(R) Core(TM) i7-12700H processor, and 16.00 GB of RAM, using the Matlab R2022a version software.

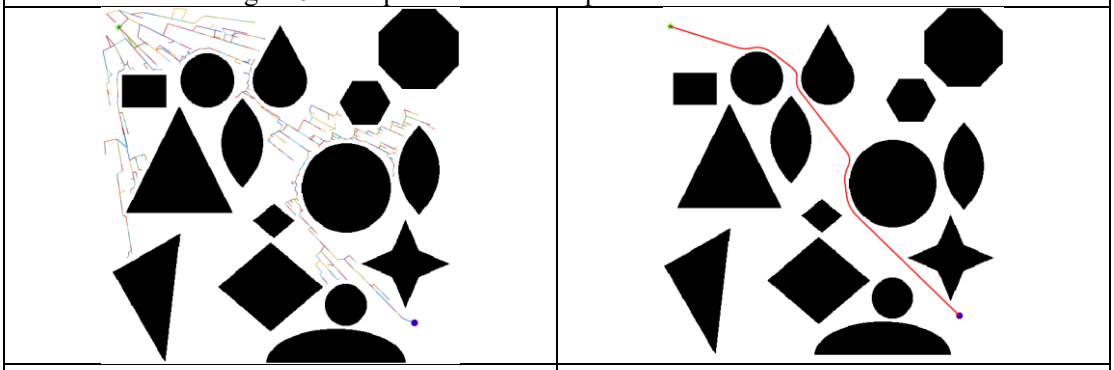
5.1 2D Simulation Experiment

Simulation results comparing the two RRT algorithms in simple and complex test scenarios show that the original RRT algorithm is not stable enough during execution and generates a large number of non-essential nodes especially in complex 2D scenarios. The modified RRT algorithm, especially when dealing with complex scenes, reduces the number of extended nodes, its directionality is significantly improved and the generated paths are smoother compared to the original RRT algorithm. Simulation of simple and complex scene scenarios are shown in Figure 9 and Figure 10, respectively.



(a)RRT (b) Improved RRT

Figure 9. Comparison of 2D simple environment simulation



(a)RRT (b) Improved RRT

Figure 10. Comparison of 2D complex environment simulation

According to the data in Table 2, the standard RRT algorithm needs to generate an average of 936 sampling points in a 2D space. In contrast, the optimized RRT algorithm needs to generate only 258 sample points on average, which means that the number of sample points is reduced by about 72%. Meanwhile, the average execution time of the standard RRT algorithm is 8.13 seconds, while the optimized version has a shorter execution time of 3.03 seconds, which is about 62% less. Thus, the optimized and improved algorithm shows a significant performance advantage.

Table 2. Comparison of the effect of 2D algorithms

atlase	algorithms	sampling point	path length /mm	Planning time /s
Simple 2D	RRT	482	712	3.64
	Improved RRT	114	660	1.96
Complex 2D	RRT	1390	818	12.62
	Improved RRT	402	682	4.10

5.2 3D Simulation Experiment

To conduct simulation experiments in 3D space, we set the starting point as (0,0,0) and the end point as (700,800,1000). Through the comparison experiments in Figure 11 and Table 3, it can be seen that the traditional RRT algorithm produces many unnecessary branches and redundant nodes when finding the path. The improved RRT algorithm in this paper reduces the number of sampling points by 70.3% and improves the time by 68.3% compared to the traditional RRT algorithm, which is a significant improvement compared to the traditional RRT algorithm.

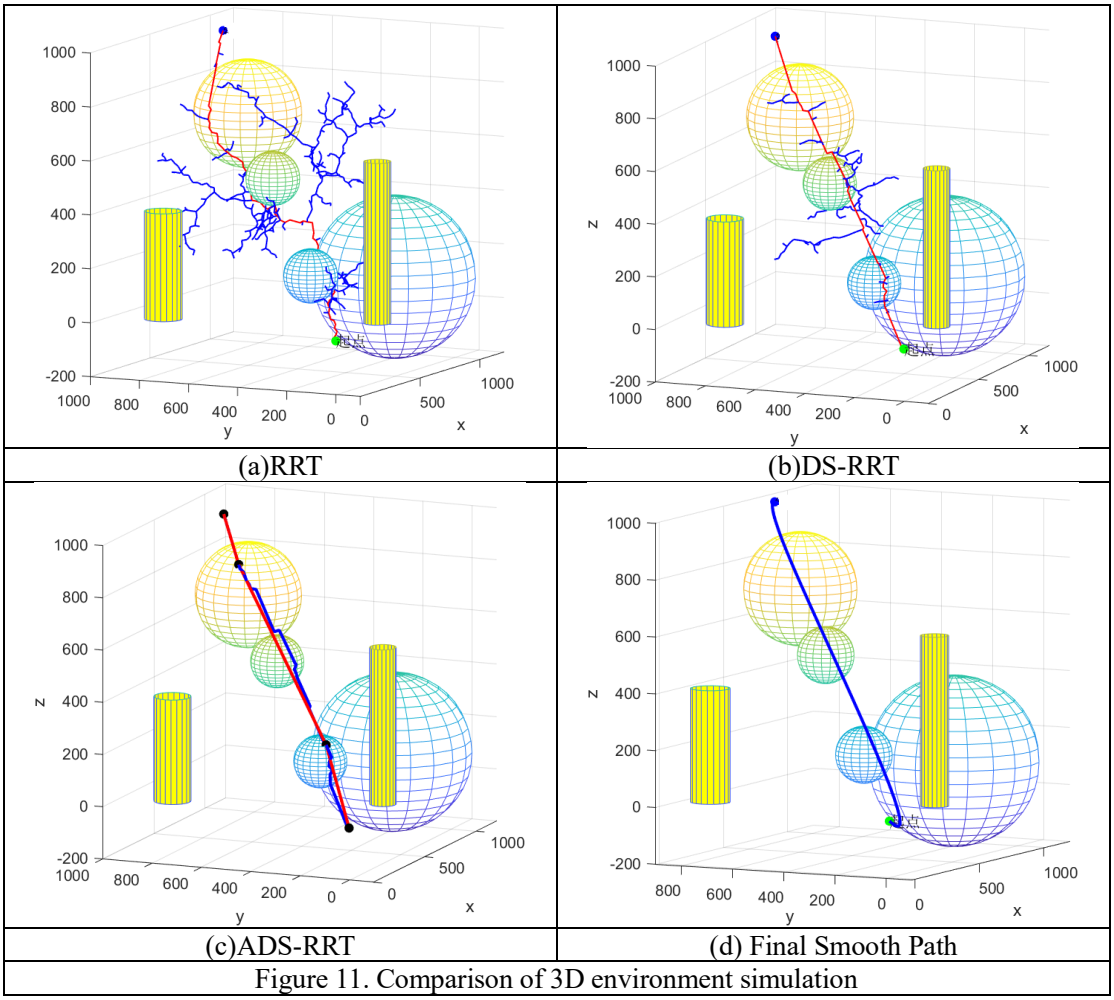


Figure 11. Comparison of 3D environment simulation

Table 3. Comparison of the effect of 3D algorithms

atlase	algorithms	sampling point	path length /mm	Planning time /s
3D	RRT	736	1880	4.84
	Improved RRT	218	1480	1.78

5.3 Robotic arm obstacle avoidance experiment

In order to verify the effectiveness of the improved RRT algorithm in this study for application in real-world environments, simulation experiments for modeling and path planning of the UR10 robotic arm were conducted in this paper using MATLAB-Robotics Toolbox. Using the proposed improved RRT path planning algorithm, a path from the start point to the end point is computed, which completely avoids the obstacles in the path. Subsequently, the simulation model of the UR10 robotic arm is validated to demonstrate the effectiveness of the improved algorithm and its practical feasibility in robotic arm path planning. In this simulation experiment, we set specific parameters to evaluate the performance of the improved RRT algorithm. The start point is set at (200,-300,200) while the end point is set at (220,350,600) and the maximum number of iterations is limited to 5000. To

simplify the experimental environment, the obstacles are assumed to be solid spheres. These parameter settings are intended to simulate the obstacle avoidance path planning of the robotic arm in 3D space. The simulation is shown in Figure 12.

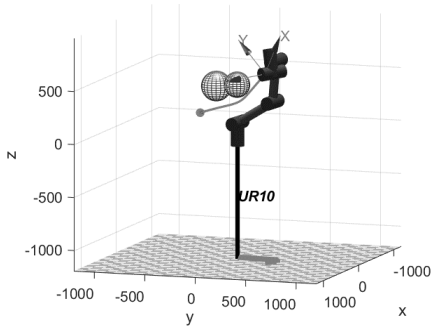


Figure 12. 3D robotic arm obstacle avoidance simulation results

As can be seen from Figure 12, due to the smooth path to reduce bending, the improved RRT algorithm plans a smooth path with smooth joint angle changes without large fluctuations, which is highly feasible in practical applications. Due to the large number of joints, the single joint position change is taken as the comparison object in the experiment, and the joint angle change comparison graph is shown in Figure 13.

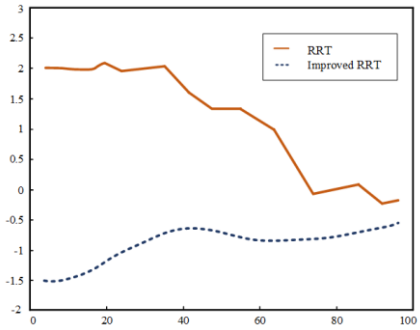


Figure 13 Comparison of the changes in the angle of the improved RRT joints

ACKNOWLEDGMENTS

State Grid Mengdong Electric Power 2023 cost-based science and technology project (526608230009)

Refrences

[1] Zhang, X.; Huang, M.; Lei, M.; Tian, H.; Chen, X.; Tian, C. Improved Rapid-Expanding-Random-Tree-Based Trajectory Planning on Drill ARM of Anchor Drilling Robots. Machines 2023, 11, 858. https://doi.org/10.3390/machines11090858.

[2] Li X ,Tong Y .Path Planning of a Mobile Robot Based on the Improved RRT Algorithm[J].Applied Sciences,2023,14(1).

[3] Yu, J.; Wu, J.; Xu, J.; Wang, X.; Cui, X.; Wang, B.; Zhao, Z. A Novel Planning and Tracking Approach for Mobile Robotic Arm in Obstacle Environment. Machines 2024, 12, 19. https://doi.org/10.3390/machines12010019.

[4] Dong, Z.; Zhong, B.; He, J.; Gao, Z. Dual-Arm Obstacle Avoidance Motion Planning Based on Improved RRT Algorithm. Machines 2024, 12, 472. https://doi.org/10.3390/machines12070472.

[5] Long, H., Li, G., Zhou, F., & Chen, T. (2023). Cooperative dynamic motion planning for dual manipulator arms based on RRT* Smart-AD Algorithm. Sensors, 23(18), 7759.

[6] Xia, X., Li, T., Sang, S., Cheng, Y., Ma, H., Zhang, Q., & Yang, K. (2023). Path planning for obstacle avoidance of robot arm based on improved potential field method. Sensors, 23(7), 3754.

[7] Qureshi, Ahmed Hussain, and Yasar Ayaz. "Intelligent bidirectional rapidly-exploring random trees for optimal motion planning in complex cluttered environments." Robotics and Autonomous Systems 68 (2015): 1-11.

[8] JEONG I B , LEE S 丿,KIM J H. Quick-RRT*: triangular in-equality-based implementation of RRT* with improved initialsolution and convergence rate[J].Expert Systems with Applications,2019,123(1):82 -90.

[9] JIA H, FANG L, WANG H. Adaptive path planning of robotic arm by integrating artificial potential field and 214000Informed-RRT~(*) algorithm[J]. Computer Integrated Manufacturing Systems,1-21[2024-03-18].

[10] Wang, Yan, et al. "Path planning of a 6-DOF measuring robot with a direction guidance RRT method." Expert Systems with Applications 238 (2024): 122057.

[11] Han K, Cheng W. Robotic arm path planning based on improved RT-Connect algorithm[J]. Computer Application and Software,2022, 39(3):260-265.

[12] LI C, Dai W, YANG C et al. Robotic arm path planning based on Sobol sequence and fast extended random tree[C]//Process Control Committee of Chinese Society of Automation,Chinese Society of Automation. Abstracts collection of the 31st Chinese Process Control Conference (CPCC 2020).2020:1.DOI:10.26914/c.cnkihy.2020.030044.

[13] Karaman S, Frazzoli E. Incremental sampling-based algorithms for optimal motion planning[J]. Robotics:Science and Systems, 2011, 6: 267-274.

[14] GUO Y, Lai G. A review of research on joint space trajectory planning and optimization of industrial robots[J]. Mechanical Transmission,2020,44(02):154-165.DOI:10.16578/j.issn.1004.2539.2020.02.024.

[15] Copot, C.; Muresan, C.; Ionescu, C.-M.; Vanlanduit, S.; De Keyser, R. Calibration of UR10 Robot Controller through Simple Auto-Tuning Approach. Robotics 2018, 7, 35. https://doi.org/10.3390/robotics703003.

[16] Kelly, M.; Bryan, C.; M., N.M.; Bram, V. Real-time constraint-based planning and control of robotic manipulators for safe hu-man–robot collaboration. Robot. Comput.-Integr. Manuf. 2024, 87, 102711.

[17] Shi, W., Wang, K., Zhao, C., & Tian, M. (2022). Obstacle avoidance path planning for the dual-arm robot based on an improved RRT algorithm. Applied Sciences, 12(8), 4087.

[18] Dai, J.; Zhang, Y.; Deng, H. Bidirectional RRT*-Based Path Planning for Tight Coordination of Dual Redundant Manipulators. Machines 2023, 11, 209. https://doi.org/10.3390/machines11020209.