applied sciences

MDPI

Article

Obstacle Avoidance Path Planning for the Dual-Arm Robot Based on an Improved RRT Algorithm

Wubin Shi [1,2], Ke Wang [2,*], Chong Zhao [2] and Mengqi Tian [1,2]

[1] University of Chinese Academy of Sciences, Beijing 100049, China; shiwubin19@csu.ac.cn (W.S.); tianmengqi19@csu.ac.cn (M.T.)
[2] Key Lab of Space Utilization, Technology and Engineering Center of Space Utilization, Chinese Academy of Sciences, Beijing 100094, China; zhaochong@csu.ac.cn
* Correspondence: wangke@csu.ac.cn

Abstract: In the future of automated production processes, the manipulator must be more efficient to complete certain tasks. Compared to single-arm robots, dual-arm robots have a larger workspace and stronger load capacity. Coordinated motion planning of multi-arm robots is a problem that must be solved in the process of robot development. This paper proposes an obstacle avoidance path planning method for the dual-arm robot based on the goal probability bias and cost function in a rapidly-exploring random tree algorithm (GA_RRT). The random tree grows to the goal point with a certain probability. At the same time, the cost function is calculated when the random state is generated. The point with the lowest cost is selected as the child node. This reduces the randomness and blindness of the RRT algorithm in the expansion process. The detection algorithm of the bounding sphere is used in the process of collision detection of two arms. The main arm conducts obstacle avoidance path planning for static obstacles. The slave arm not only considers static obstacles, but also takes on the role of the main arm at each moment as a dynamic obstacle for path planning. Finally, MATLAB is used for algorithm simulation, which proves the effectiveness of the algorithm for obstacle avoidance path planning problems for the dual-arm robot.

Keywords: dual-arm robot; improved RRT algorithm; path planning; autonomous obstacle avoidance

Citation: Shi, W.; Wang, K.; Zhao, C.; Tian, M. Obstacle Avoidance Path Planning for the Dual-Arm Robot Based on an Improved RRT Algorithm. *Appl. Sci.* **2022**, *12*, 4087. https://doi.org/10.3390/app12084087

Academic Editors: Giovanni Boschetti and João Miguel da Costa Sousa

Received: 23 February 2022
Accepted: 17 April 2022
Published: 18 April 2022

1. Introduction

With the development of science and industrial automation, robot technology has been greatly developed in recent decades, and gradually applied in military, aerospace, industry, medical, service, and other fields [1,2]. Single-arm industrial robots have achieved notable development and application in China, widely replacing manual casting, welding, palletizing, and other operations [3,4]. However, many complex operational tasks require collaboration between the robotic arms. The dual-arm robot has a larger working space, stronger load capacity, and obvious advantages in heavy lifting and assembly scenarios. However, unlike a simple combination of two single-arm robots, a dual-arm robot has some overlap in its workspace. The path planning of two arms should not only consider static obstacles in space, but also consider the interference between the two arms. In the field of dual-arm robotics, how to realize obstacle avoidance motion planning is always a hot issue [5,6].

In the field of robot path planning, many path planning algorithms have been formed. The traditional methods mainly include the artificial potential field algorithm [7–9], the A* algorithm [10,11], and the RRT algorithm [12,13], etc. The methods based on computational networks mainly include neural network algorithms [14,15] and bioinspired planning algorithms [16,17]. Bioinspired planning algorithms mainly include the genetic algorithm, ant colony optimization (ACO), and so on. The genetic algorithm is an intelligent bionic algorithm based on natural selection and genetic mechanisms [18]. The ACO is an intelligent

optimization search model established with reference to the foraging method of ant populations [19,20]. However, in the joint space of a manipulator with high degrees of freedom, each movement of the arms will generate a high computational workload. Algorithms based on computational networks have problems such as excessive computational load and insufficient real-time performance in the field of dual-arm robot path planning.

Among the traditional path planning algorithms, the artificial potential field method has the characteristics of simple implementation and good real-time performance, and has been widely used. Nonetheless, once the resultant force is zero, that is, at the local potential energy minimum point, the algorithm will fall into a deadlock state and stop searching [21]. In addition, the artificial potential field algorithm needs the sampling information of the entire workspace to avoid local minima, which affects the dynamic obstacle avoidance of the dual-arm robot.

The A* algorithm is a method for solving the shortest path in a known static road network. This method is a search algorithm obtained by adding a heuristic function to the Dijkstra algorithm [22]. It evaluates the nodes and guides the search process based on the evaluation value. Latombe et al. applied a PRM (probabilistic road map) to multi-robot path planning, but the increase in degrees of freedom and obstacles reduced the speed of the method [23].

In the field of multi-axis robot planning, due to the constraints of complex environments, most search methods are based on random sampling. Among the path planning methods used in the past few decades, the RRT algorithm, based on random search strategy, is suitable for path planning in high-dimensional space, and has been widely used in the path planning of robots [24–26]. The algorithm avoids modeling the entire space by detecting collisions at sample points in the state space. Therefore, it can effectively solve path planning problems with complex constraints in high-dimensional space, and has the advantages of probabilistic completeness and perfect scalability.

RRT is a fast search algorithm proposed by LaValle [27]. However, it is a random search algorithm, and the search path may not be in the direction of the target, therefore the convergence speed is relatively slow. In order to improve the target orientation of the RRT algorithm, Chris Urmson et al. [28] proposed a P probability RRT algorithm based on goal bias strategy. Li et al. [29] proposed a variable step size trunk fast exploration random tree (VT-RRT) algorithm. By transforming the search space of random nodes in the RRT algorithm and adaptively adjusting the step size according to the target position, the search efficiency is effectively improved, and the path planning time is reduced. Jiang Hong et al. [30] proposed a new node expansion method that is biased towards the target point. The method combines the target point gravity, obstacle repulsion, and random point gravity, and adds an adaptive function related to the obstacle distance. A pruning optimization method is proposed to optimize the path length. Lei Shao et al. [31] used an ant colony algorithm to optimize RRT. Experiments show that the combination of an ant colony algorithm and RRT algorithm effectively reduces the number of nodes and the average computing time. Kun Wei [32] proposed a dynamic path planning method for robot autonomous obstacle avoidance based on an improved RRT algorithm, namely smooth RRT (S-RRT). This method takes the directional node as the goal, which greatly improves the sampling speed and efficiency of RRT.

In the field of path planning for dual-arm robots, Andreas [33] proposed a dual-arm path planning method based on closed-chain kinematics to meet the motion constraints of the manipulator. Kim [34] proposed a dimension reduction RRT method, which reduced the dimension of high-dimensional path planning space according to the task requirements of the dual-arm robot. To ensure that the RRT algorithm has higher efficiency, Li yang [35] proposed a cooperative path planning method for a dual-arm robot based on gravity adaptive step length RRT. The simulation results show that the gravity adaptive step size RRT method can effectively constrain the step size in the workspace to ensure the effectiveness of the collision detection algorithm.

In this paper, the GA_RRT path planning algorithm is proposed based on the traditional RRT algorithm by introducing the goal probability bias strategy and A* cost function. In addition, two manipulators use the bounding sphere collision detection method. The main arm considers static obstacles, and the slave arm considers static obstacles and dynamic main arm obstacles. Finally, MATLAB software is used to simulate the obstacle avoidance path planning for the dual-arm robot. It is finally proved that the algorithm is effective and superior for the path planning of dual-arm robot.

2. Model Building

In this paper, the robot platform composed of two UR5 robotic arms is taken as an example. It mainly includes one base and two 6-DOF manipulators. According to the DH modeling method, we construct the dual-arm robot model, as shown in Figure 1. The joint diagram of the main manipulator is shown in Figure 2. Using the parameters of the robotic arm and DH method, we obtain the DH parameters in Table 1. The joint diagram of the slave manipulator, and the joint diagram of the main manipulator, are mirror-symmetric with the robot body. For example, the DH parameters α and θ of the master manipulator are opposite to those of the slave manipulator.

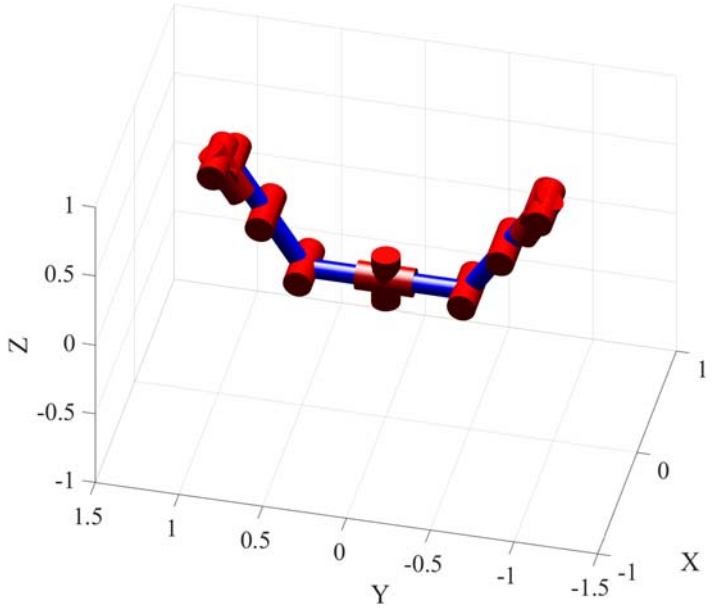


Figure 1. The model of the dual-arm robot.

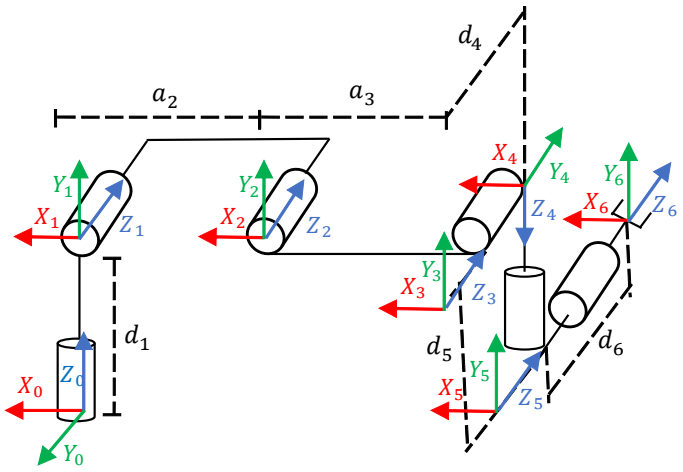


Figure 2. Coordinate system of the main manipulator joint.

Table 1. DH parameters of the main robot arm.

Joint Number	α	a	θ	d
1	90	0	θ_1	89.2
2	0	−425	θ_2	0
3	0	−392	θ_3	0
4	90	0	θ_4	109.3
5	−90	0	θ_5	94.75
6	0	0	θ_6	82.5

The homogeneous coordinate transformation matrix of a six-axis manipulator with this parameter is shown in Equation (1).

$$ {}^{i-1}_{i}T = \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i)\cos(\alpha_i) & \sin(\theta_i)\sin(\alpha_i) & \alpha_i\cos(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i) & -\cos(\theta_i)\sin(\alpha_i) & \alpha_i\sin(\theta_i) \\ 0 & \sin(\alpha_i) & \cos(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1) $$

The transformation matrix of all adjacent coordinate systems can be obtained by substituting DH parameters. The transformation matrix relative to the base coordinate system is shown in Equation (2).

$$ T = {}^{0}_{1}T^{1}_{2}T^{2}_{3}T^{3}_{4}T^{4}_{5}T^{5}_{6}T \quad (2) $$

Finally, the dual-arm manipulator needs to be unified into the base coordinate system using Equation (3).

$$ S = S_0 T \quad (3) $$

3. Algorithm Improvement

3.1. Basic RRT

RRT constructs a random tree through random sampling. In the space with obstacles, the algorithm starts to explore from the initial node, X_{init}. This process is shown in Figure 3. Firstly, the random point, X_{rand}, is generated. In all the added path nodes of the tree [X_{init}, X_1, X_2, X_3, $X_{nearest}$], the node $X_{nearest}$, which is nearest to X_{rand}, is selected. Taking $X_{nearest}$ as the root node, a fixed step R is added to the direction of X_{rand} to obtain X_{new}. The collision detection algorithm is used to detect whether the path between $X_{nearest}$ and X_{rand} is feasible. If there is no collision, X_{new} is added to the random tree, T. Alternatively, if there is a collision, X_{rand} will be regenerated. Repeat the whole process until the distance between the latest node and X_{goal} is less than the step value, and there are no obstacles between them; at this point it is considered that the algorithm has converged. When the planning algorithm converges, we start from X_{goal} and trace back along its parent node to find an effective path between the start point and the end point.

The process of the RRT algorithm is shown in Algorithm 1.

Algorithm 1: RRT algorithm

```
T = init Tree(); // Initialize the random tree
R = init R(); // Initialize the step size
T[0] = Node(X_init);
for i = 1 to N:
    X_rand = Random sampling(); // Random sampling
    X_nearest = min the distance (X_rand)Node_tree); // Find the nearest node
    X_new = extend (X_nearest, X_rand, R); // Expand towards random points
    if not obstacle(X_nearest, X_new):
        T[i] = add Node(X_new); // Add a new node to the tree
    end if
end for
return T
```

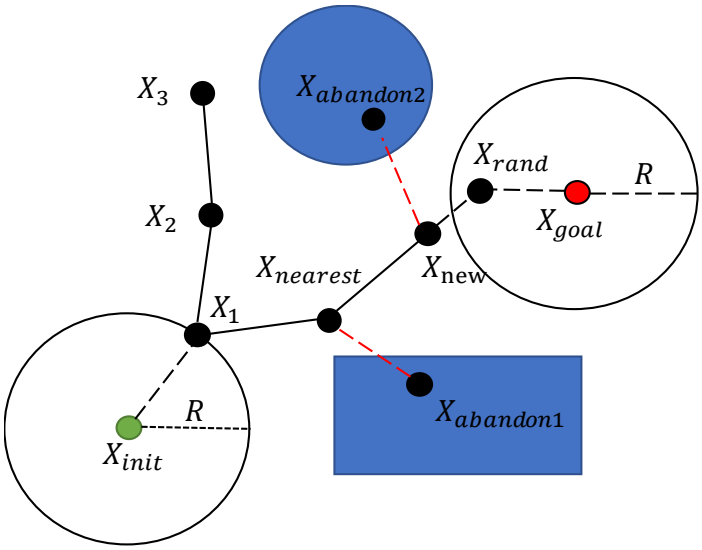


Figure 3. RRT algorithm random tree expansion process.

3.2. Path Planning of Dual-Arm Based on GA_RRT

The disadvantage of the traditional RRT algorithm is that it is a random exploration of the whole space, and does not consider the path cost. This paper provides some knowledge of goal probability bias and path cost.

Some principles of the goal probability bias and A* cost function are mainly introduced in this section. Due to the computational complexity of collision detection for dual-arm robots, collision detection is mainly performed using the method of the bounding sphere.

3.2.1. The Goal Probability Bias

The traditional RRT algorithm is a random search of space, which ensures the effectiveness of the RRT algorithm. However, excessive blind search reduces the convergence speed and consumes a large amount of computation power. Therefore, we adopt a goal probability bias strategy. This method sets a parameter, P_a, in sampling judgment. Before each extension, a random value, P_{rand}, (between point 0, 1) is randomly generated. When $0 < P_{rand} < P_a$, the random tree grows toward the target point, making the random tree expansion more objective. When $P_a < P_{rand} < 1$, X_{rand} points are generated randomly.

For the dual-arm manipulator environment, when the goal point is taken as a random point with probability P_a, we need to read the real-time position of the manipulator, and detect whether there are static obstacles and dynamic obstacles of the manipulator under the path. When there are obstacles, the cost of expansion is infinite. We abandon the random point and restart the random process.

3.2.2. A* Cost Function

Although the probability-biased RRT algorithm greatly shortens the search time, there are still many redundant random points. Therefore, this paper combines the goal probability bias RRT algorithm with the A* search algorithm. The cost function was used to search for the best sampling point, so as to reduce the level of redundant computing and optimize the path. We consider the forward and backward cost of the current node in the RRT algorithm. The distance from the initial point, X_{init}, to the selected node is called forward cost, which can also be called heuristic function $H(i)$. The distance between X_{select} and X_{goal} is called backward cost $B(i)$. The cost function of the current node is represented by $F(i)$, shown in Equation (4).

$$\begin{cases} F(i) = H(i) + B(i) \\ B(i) = \|X_{goal} - X_{rand}(i)\| \\ H(i) = \|X_{rand} - X_{init}(i)\| \end{cases} \tag{4}$$

After calculating the random node, select the random point with the smallest cost function to join the random tree list. Repeat the calculation process until extended to the target node.

We need to add two additional data groups. One is the optional random point data group, and the other is the cost function data group corresponding to the optional random point. Each time we expand the random tree, multiple random points, X_{rand}, are generated and put into the optional random point group. Next, the cost function is calculated for the optional random point group. According to the cost function data group, the minimum cost node is taken.

3.2.3. Collision Detection of Dual-Arm Robots

In order to simplify the calculation of the 6-DOF manipulator, the method of the bounding sphere is used for obstacle avoidance detection. It is the smallest sphere that surrounds three-dimensional objects. In order to reduce the complex calculation of collision detection during the movement of the arms, this method is selected in this paper. Its advantages are convenient calculation, simple structure, and it is not affected by spin. This algorithm has better security while generating partial redundant spaces. When considering whether there is a collision between the manipulator and the obstacle, or a collision between the main manipulator and the slave manipulator, it is only necessary to consider the distance and radius of the object center.

The base point of the main manipulator is denoted as O_{10}, and the center points of the six joints are denoted as $\{O_{11}, O_{12}, O_{13}, O_{14}, O_{15}, O_{16}\}$. Then, the center points of the connecting rod are denoted as $\{A_1, A_2, A_3, A_4, A_5, A_6\}$. These values are determined by the parameters of the manipulator body. Taking the joint bounding sphere as an example, the joint bounding sphere can be expressed as:

$$R = \left\{ (x, y, z) \middle| (x - o_{ijx})^2 + (y - o_{ijy})^2 + (z - o_{ijz})^2 < r^2 \right\} \tag{5}$$

The radius of the joint bounding sphere for the main manipulator is set as $\{R_{11}, R_{12}, R_{13}, R_{14}, R_{15}, R_{16}\}$, and the bounding sphere radius of the connecting rod is set as $\{R_{a1}, R_{a2}, R_{a3}, R_{a4}, R_{a5}, R_{a6}\}$. The slave arm parameters are represented by O_{2i}, B_i, R_{2i}, R_{bi}. The radius can be calculated using Equation (6)

$$R_{ij} = \frac{1}{2}\sqrt{(x_{ij\max} - x_{ij\min})^2 + (y_{ij\max} - y_{ij\min})^2 + (z_{ij\max} - z_{ij\min})^2} \tag{6}$$

where x_{ijmax}, x_{ijmin}, y_{ijmax}, y_{ijmin}, z_{ijmax}, z_{ijmax}, respectively, represent the maximum and minimum values of the object projected on the XYZ coordinate axes. Bounding sphere model of the dual-arm robot is shown in Figure 4.

The conditions for collision detection of an enclosing ball are shown in Equations (5)–(7).

1. No collision between each connecting rod and the joint in the main manipulator:

$$\begin{cases} \|O_{1l} - O_{1j}\| < (R_{1l} + R_{1j}) \\ \|O_{1i} - A_j\| < (R_{1i} + R_{aj}) \end{cases}, i \neq j, i = 1, 2, \ldots, 6, j = 1, 2, \ldots, 6 \tag{7}$$

2. No collision between each connecting rod and the joint in the slave manipulator:

$$\begin{cases} \|O_{2i} - O_{2j}\| < (R_{2i} + R_{2j}) \\ \|O_{2i} - B_j\| < (R_{2i} + R_{bj}) \end{cases}, i \neq j, i = 1, 2, \ldots, 6, j = 1, 2, \ldots, 6 \tag{8}$$

3. No collision between the links of the main manipulator and the links of the slave manipulator:

$$\begin{cases} \|O_{1i} - O_{2j}\| < (R_{1i} + R_{2j}) \\ \|O_{1i} - \mathrm{B}_j\| < (R_{1i} + R_{\mathrm{b}j}) \\ \|A_i - O_{2j}\| < (R_{ai} + R_{2j}) \\ \|A_i - \mathrm{B}_j\| < (R_{ai} + R_{\mathrm{b}j}) \end{cases}, i = 1, 2, \ldots, 6, j = 1, 2, \ldots, 6 \quad (9)$$

The static obstacle collision detection in the environment mainly calculates the distance, D, from the center of the manipulator to the center of the obstacle ball. The collision is judged by comparing the distance, D, with the sum of the radius of the enclosing ball.

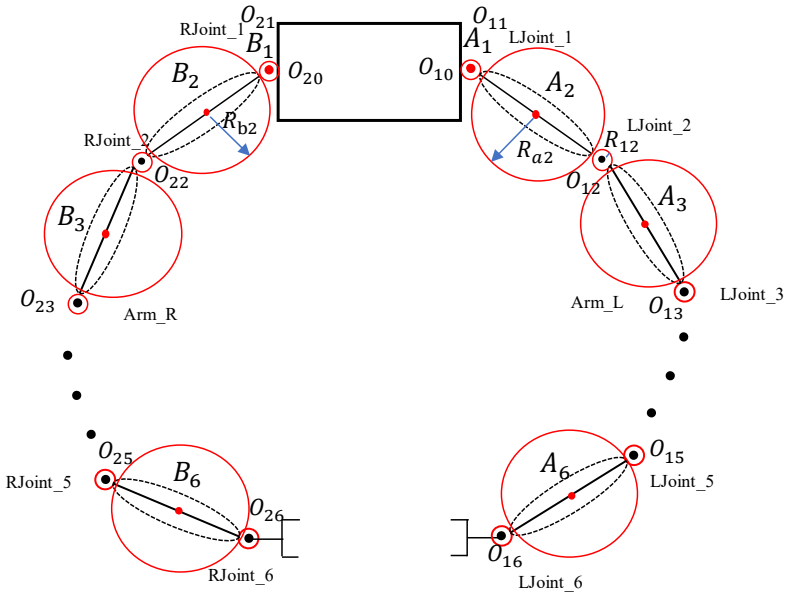


Figure 4. Bounding sphere model of the dual-arm robot.

3.2.4. Overall Process of GA_RRT

The GA_RRT algorithm combines the probabilistic bias method and A* algorithm with RRT. The process of the algorithm is shown in Algorithm 2.

Algorithm 2: GA_RRT algorithm

```
T = init Tree(); // Initialize the random tree
  R = init R(); // Initialize the step size
  T[0] = Node(X_init);
  for i = 1 to N:
      for j = 1 to M: // M random states are generated, recommended value 4
          if P_rand < P_a:
              X_direction = X_goal;// Select random points as target points
          else:
            X_direction = Random sampling(); // Random sampling
            X_nearest = min the distance X_direction, Node_tree);
            X_select = extend (X_nearest, X_direction, R); // Expand
            X_path(j) = add X_select // Add random states to cost function group
      end for
      F(j) = the distance(X_path(j), X_init) + the distance(X_path(j), X_goal); // cost function
      X_new = min F(j); // The minimum cost node is taken in tree
  if not obstacle(X_nearest, X_new):
      T[i] = add Node(X_new); // Add new nodes
  end if
  end for
  return T
```

The overall flow of GA_RRT algorithm is shown in Figure 5. First, the main flow of the algorithm is used to perform random sampling. Then, the target point is expanded as a direction point with a probability of P_a, and a random point is generated with a probability of $1 - P_a$ for expansion. After randomly generating multiple random points for expansion, the algorithm calculates the A* cost function and selects the node with the smallest cost function for the next collision detection. If it passes the detection, the node is added to the random tree. Loop this process until you are near the target point.

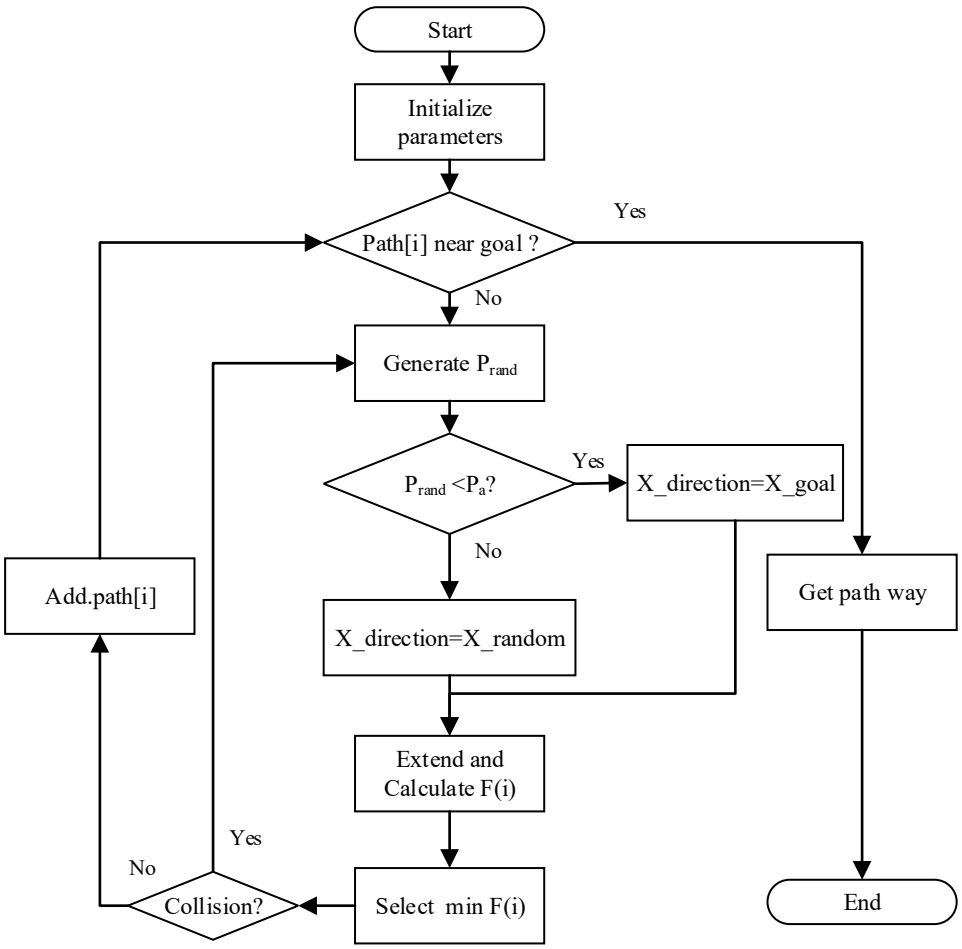


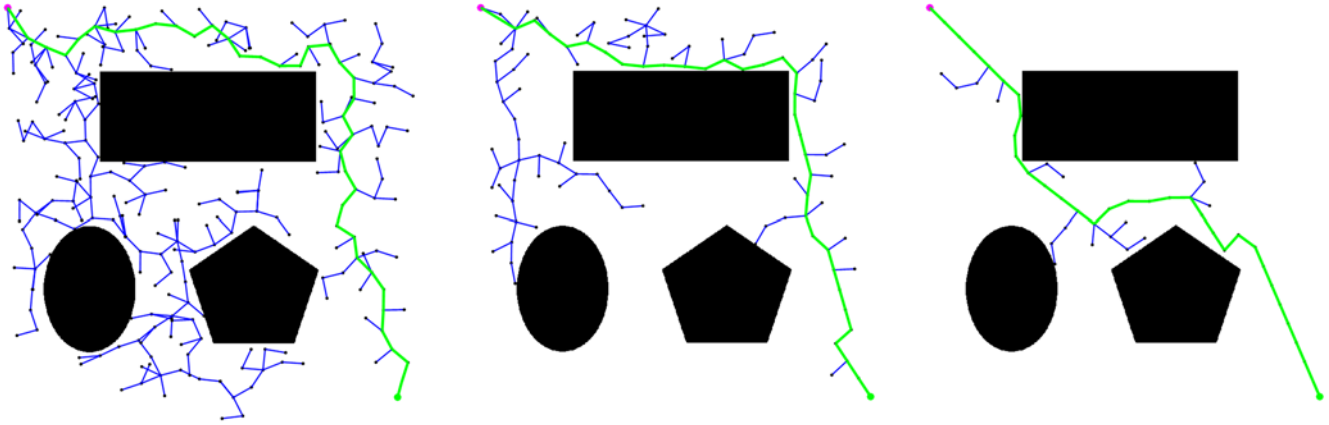
Figure 5. GA_RRT algorithm flow.

4. Simulation

4.1. Obstacle Avoidance Path Planning for Static Plane

In order to analyze the effectiveness of the GA_RRT algorithm for the path planning problem, this paper compares with the traditional RRT algorithm and the RRT algorithm based on goal bias probability (G_RRT) in two different scenarios. The hardware platform is Intel(R) Core (TM) i7-10700f CPU with 16GB of memory. The simulation experiment is built using MATLAB software.

The simulation mainly adopts two common methods to verify the validity of the path planning algorithm. Scenario A is shown in Figure 6, the initial point is 1, 1 and the target point is 750,750. As can be seen in Figure 6a, the RRT algorithm explores the entire space, while the G_RRT algorithm explores the target point with a probability of 0.5, presented in Figure 6b. This greatly reduces the number of nodes for spatial search. The GA_RRT algorithm performs node pruning by calculating the cost function while exploring the goal point. This further effectively reduces the number of tree node branches and speeds up the search efficiency.



(**a**) (**b**) (**c**)

Figure 6. The obstacle avoidance path planning result of scenario A (**a**) RRT; (**b**) G_RRT; (**c**) GA_RRT.

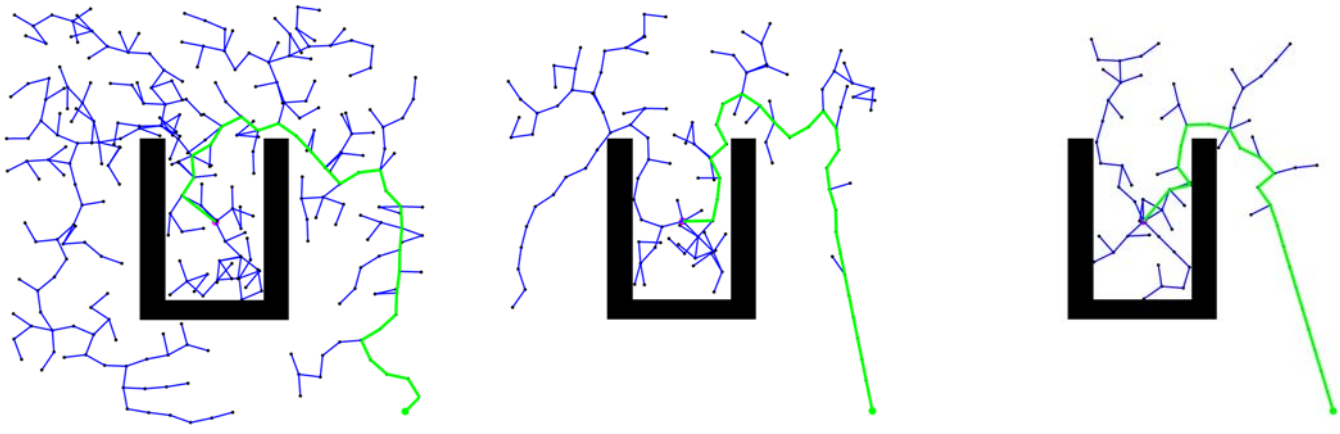
In the experiment, since the paths generated by the RRT algorithm have high randomness, the number of nodes and time of each path may be quite different. Therefore, this paper conducts ten experiments for each algorithm, and takes the average value as the final data. The path data for Scenario A are shown in Table 2.

Table 2. The average experimental data for Scenario A.

	RRT	G_RRT	GA_RRT
Total number of nodes	256	105	49
Path length	1539	1360	1373
Average time/s	8.35	4.5	3.38

It is calculated that the average number of nodes explored by the G_RRT algorithm is 58.9% less than that of the RRT algorithm. The GA_RRT algorithm is further reduced by 51.3% on the basis of the G_RRT algorithm. Due to the addition of the cost function, the path length is also optimized. Compared with the RRT algorithm, the GA_RRT path length is optimized by 10.7%. Due to the reduction in node redundancy, the time cost of the GA_RRT algorithm is reduced by 59.5% in this scenario. Compared to the G_RRT algorithm, it is reduced by 24.8%.

In scenario B, the target point is directly occluded, as shown in Figure 7. The initial point is 400,400 and the target point is 750,750.



(**a**) (**b**) (**c**)

Figure 7. The obstacle avoidance path planning result of scenario B: (**a**) RRT; (**b**) G_RRT; (**c**) GA_RRT.

The average experimental data for Scenario B is shown in Table 3.

Table 3. The average experimental data for Scenario B.

	RRT	G_RRT	GA_RRT
Total number of nodes	348	129	75
Path length	683.53	465.42	459.3
Average time/s	13.1	5.5	3.6

In scenario B, the starting point and the target point are directly blocked. Theoretically, there is no significant difference among the three algorithms in the U-shaped obstacle space. However, outside the obstacle space, the GA_RRT algorithm should have better performance. By comparing the data in Table 3, we can draw similar conclusions.

The total number of nodes explored by G_RRT is 62.9% lower than that of the RRT algorithm. The GA_RRT algorithm is further reduced by 41.8% on the basis of the G_RRT algorithm. Compared to the RRT algorithm, the GA_RRT path length is optimized by 32.8%, and the time cost is reduced by 72.5%.

4.2. Obstacle Avoidance Path Planning for Dynamic Dual-Arm Robots

Through the above path planning simulations in conventional scenarios, the effectiveness of the GA_RRT algorithm in path searching and reducing redundancy is successfully verified. Finally, in order to verify the effectiveness of the algorithm for the obstacle avoidance path planning of the dual-arm robot, this paper establishes a three-dimensional obstacle environment through the robot simulation toolbox.

The starting joint angle and target joint angle of the main arm are set to 45, −15, −15, 0, 0, 0 and 5, 45, 5, 15, −5, 0. The starting joint angle and target joint angle of the slave arm are set to 35, 30, 20, 0, 0, 0 and 0, −45, 30, −30, −15, 0, respectively. Due to the bounding sphere collision detection model, the obstacles will eventually be equivalent to a sphere. Therefore, this paper sets five spherical static obstacles of different sizes in the environment. The center and radius of each obstacle are shown in Equation (10).

$$O_{sphere0} = \begin{bmatrix} -0.55 & -0.25 & 0.15 \\ -0.5 & 0.3 & -0.5 \\ -0.45 & -0.2 & 0.15 \\ -0.3 & 0.3 & 0.15 \\ -0.4 & 0.25 & 0.15 \end{bmatrix}; \mathrm{R} = \begin{bmatrix} 0.05 & 0.04 & 0.03 & 0.05 & 0.06 \end{bmatrix} \quad (10)$$

The obstacle is placed on the path of the dual-arm robot moving towards the target point. The initial scene is shown in Figure 8a.

As shown in Figure 8b,c, if the robot arm does not plan the obstacle avoidance path, the slave robot arm will collide with the static obstacle and the master arm link during the movement. The final state of the dual-arm robot reaching the target point is shown in Figure 8d.

Keeping the surrounding environment of robot arm unchanged, the obstacle avoidance path planning algorithm of the dual-arm robot based on GA_RRT is added into the simulation.

Figure 9a shows the exploration process of the main arm joint angle. In order to visualize the six-dimensional exploration process, it is divided into the exploration process of the first three joints and the exploration process of the last three joints.

The angle exploration process is shown in Figure 9. The final result of the exploration is shown in Figure 10. From Figure 10, we can clearly observe the change process of the joint angle. The starting joint angle and target joint angle of the main arm are 45, −15, −15, 0, 0, 0 and 5, 45, 5, 15, −5, 0, respectively.

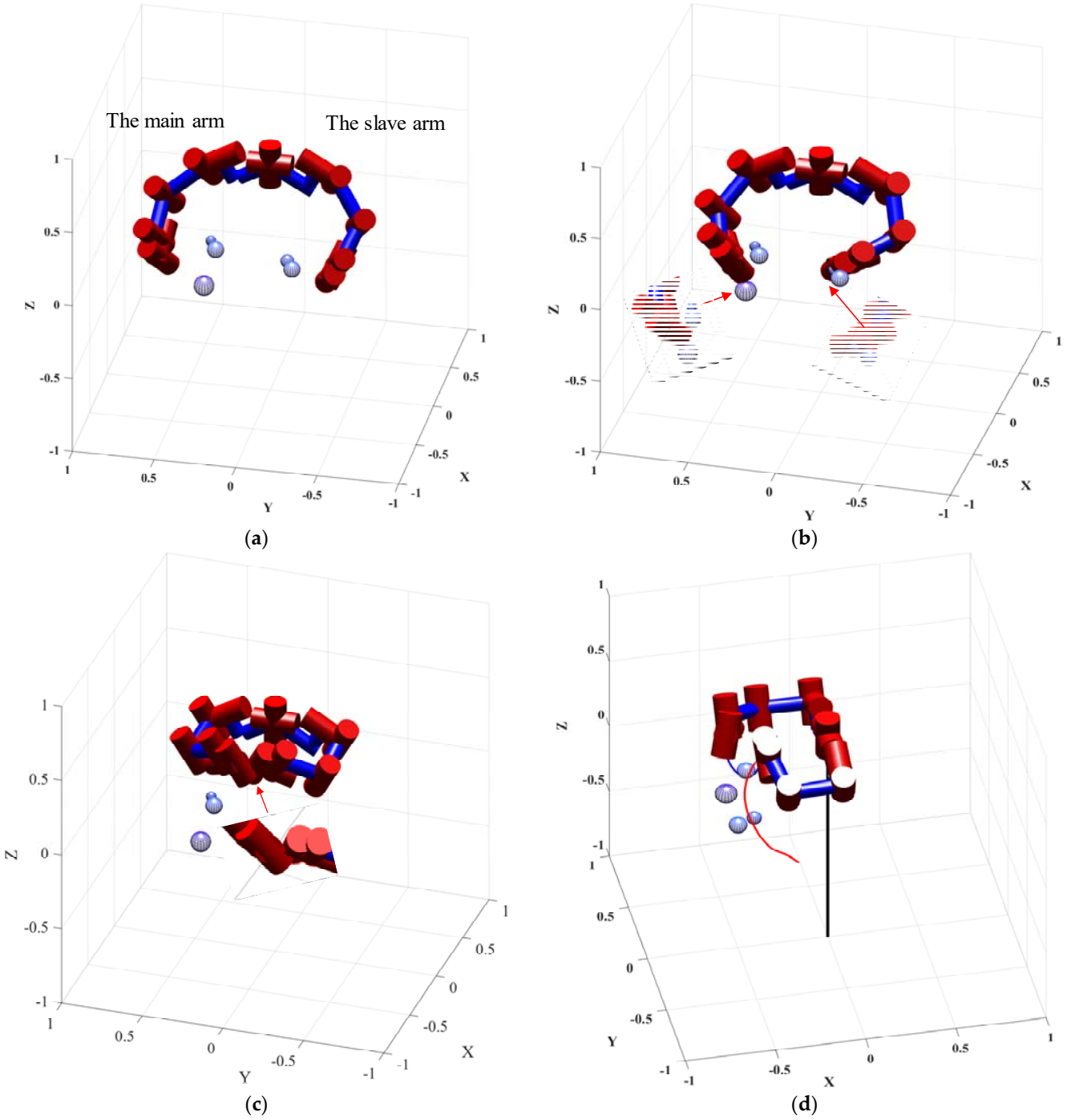


Figure 8. Motion state decomposition without obstacle avoidance path planning: (**a**) initial state; (**b**) motion state 1, collide with the environment; (**c**) motion state 2, self-collision; (**d**) side view of final state.

Similarly, Figure 11 shows the path exploration process for the slave arm. The starting joint angle and target joint angle of the slave arm are 35, 30, 20, 0, 0, 0 and 0, −45, 30, −30, −15, 0, respectively. We can draw similar conclusions from the search results in Figure 12.

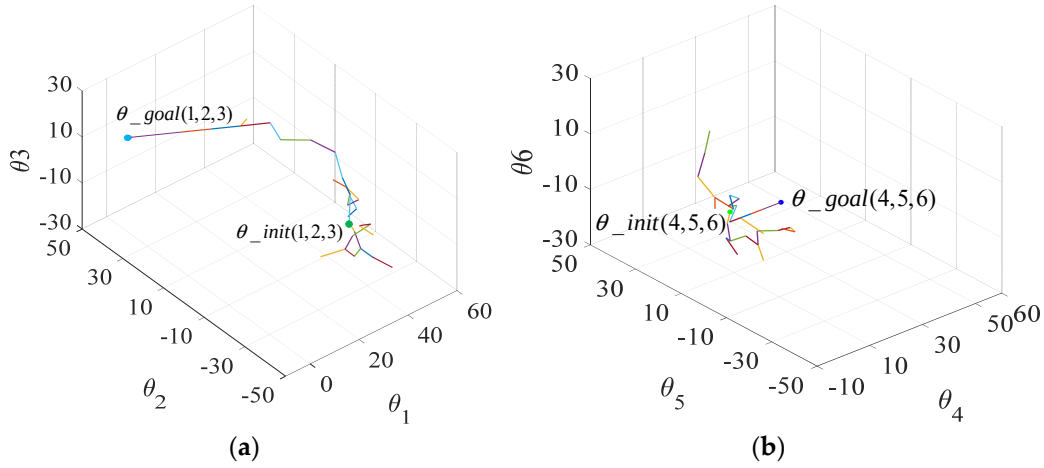


Figure 9. Exploration process of the main arm joint angle: (**a**) the first three joint angles of the main arm; (**b**) the posterior three joint angles of the main arm.

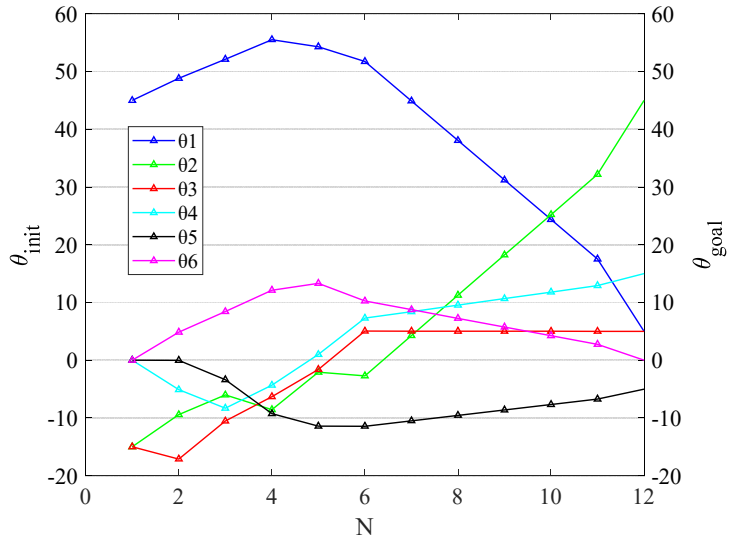


Figure 10. The final result of the path planning for the main arm.

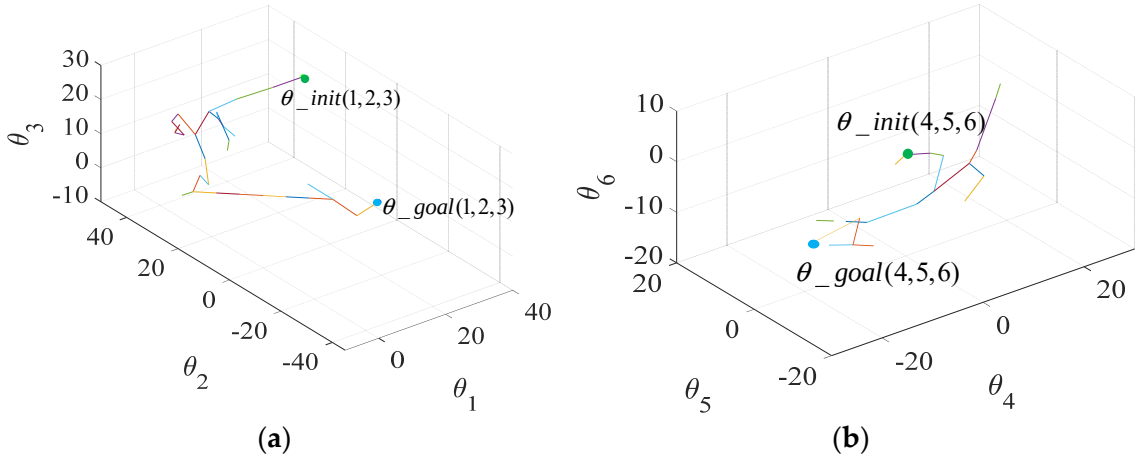


Figure 11. Exploration process of the salve arm: (**a**) the first three joint angles of the slave arm; (**b**) the posterior three joint angles of the slave arm.

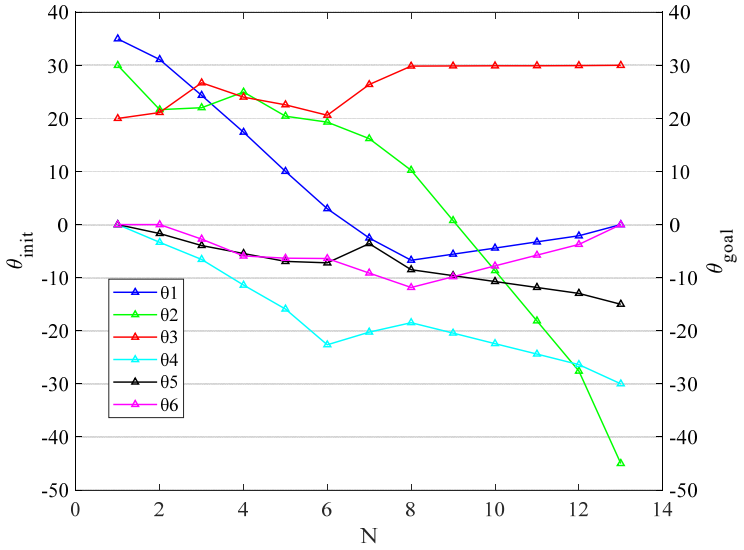


Figure 12. The final result of the path planning for the slave arm.

The simulation realizes the search process and results of a total of 12 joint angles of the main arms and slave arms. It can be seen from the search process that the joint angle search starts from the green starting point. At the beginning, because of the obstacles, the random points directly extended to the target point were not able to pass the obstacle detection. The algorithm explores the joint angle to avoid obstacles. After avoiding obstacles, all random nodes extended to the target point are quickly selected by the cost function, and the search efficiency is greatly improved.

To analyze the effectiveness of the GA_RRT algorithm for obstacle avoidance, the visualization results are presented in Figures 13 and 14.

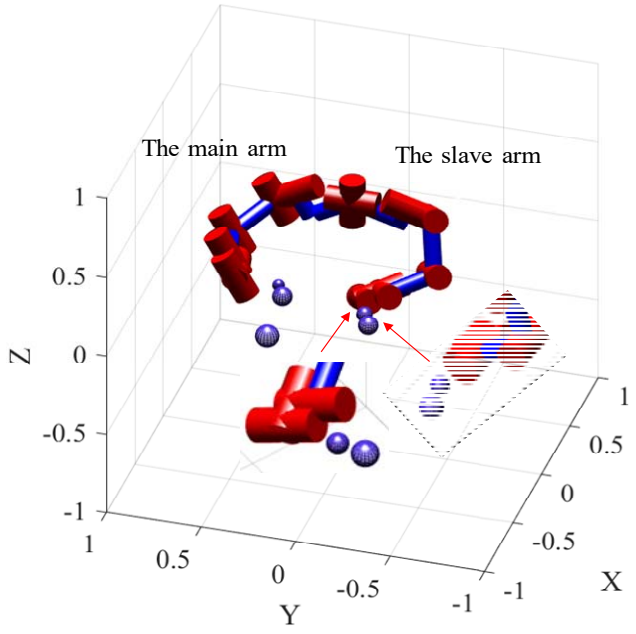


Figure 13. Static obstacle avoidance.

For static obstacles in the environment, as shown in Figure 13, the trajectory of the main arm is far away from the static obstacles. This can also be seen in Figure 14a. For the slave arm, the arm successfully avoids static obstacles, compared to Figure 8b. In addition, through the comparison of Figures 14a and 8c, it can be seen that the slave arm realizes the static obstacle avoidance and completes the dynamic obstacle avoidance of the main arm.

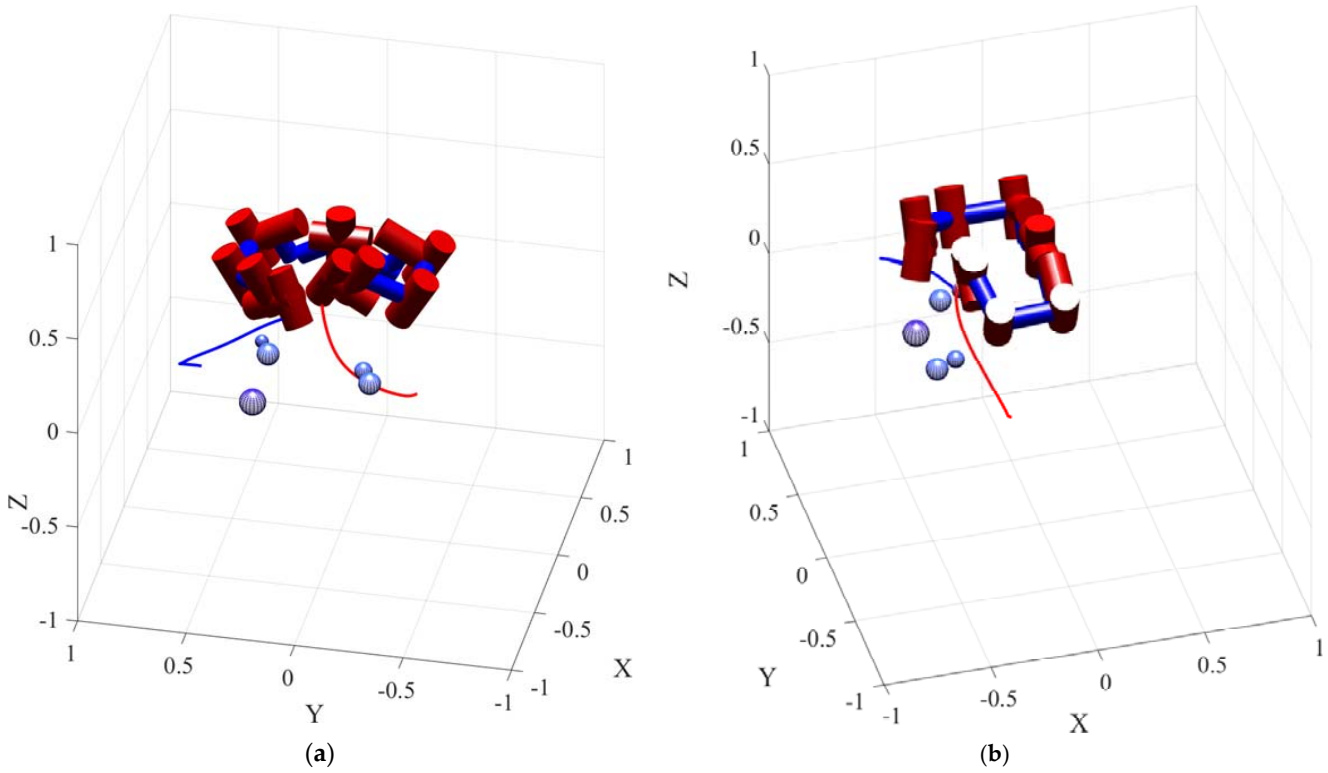


Figure 14. GA_RRT obstacle avoidance trajectory: (**a**) self-collision avoidance; (**b**) side view of final state.

After ten groups of angle planning experiments, the average planning time of the robotic arms is 2.04 s. Due to the complexity of transplanting different algorithms into the simulation of dual-arm robots, we here present a comparison by referencing the experimental data of algorithms that are similar [32]. The comparison results are shown in Table 4.

Table 4. Calculation time.

	RRT	Bi-RRT	S-RRT	GA_RRT
Average Planning Time/s	9.86	5.23	2.42	2.04

Compared to the traditional RRT algorithm, the search time of the GA_RRT algorithm is reduced by about 70% for the obstacle avoidance path planning of the dual-arm robot. Although the simulation environment is different, it can be seen that the algorithm is effective at accelerating the search efficiency.

Finally, in order to fully verify the effectiveness of the algorithm for dual-arm obstacle avoidance path planning, this paper changes the position and radius of the obstacle sphere, and records the experimental results of the GA_RRT algorithm several times. The center and radius of each obstacle in scenarios 1 and 2 are shown in Equations (11) and (12).

$$O_{sphere1} = \begin{bmatrix} -0.55 & -0.2 & 0.15 \\ -0.45 & -0.2 & 0.15 \\ -0.35 & -0.2 & 0.15 \\ -0.35 & 0.2 & 0.15 \\ -0.35 & 0.2 & -0.15 \end{bmatrix}; \mathrm{R} = [0.05, 0.05, 0.05, 0.05, 0.05] \quad (11)$$

$$O_{sphere2} = \begin{bmatrix} -0.55 & -0.2 & 0.15 \\ -0.45 & 0.2 & 0.15 \\ -0.3 & 0.3 & 0.15 \\ -0.4 & -0.25 & 0.15 \\ -0.5 & 0.3 & -0.05 \end{bmatrix}; \mathrm{R} = [0.08, 0.1, 0.06, 0.06, 0.06] \quad (12)$$

The path planning algorithm is tested ten times in each scenario, and the final average values of the tests are shown in Table 5.

Table 5. Simulation results in different obstacle environments.

	Robotic Arm	Path Length	Average Planning Time/s	Successful Times
Scenario 0	The main arm	142.5	1.73	10
	The slave arm	195.2	2.35	9
Scenario 1	The main arm	154.0	1.87	10
	The slave arm	234.3	2.90	10
Scenario 2	The main arm	153.9	1.67	10
	The slave arm	163.8	2.23	10

From the analysis of successful times, we can find that the algorithm realizes obstacle avoidance path planning for the dual-arm robot in various scenarios. Compared with the slave arm, the master arm only considers spherical obstacles, and its path planning will be faster. This can also be seen from the experimental data. The slave arm needs to detect the motion state of the main arm for dynamic obstacle avoidance. In each scenario of ten tests, path planning time will be different. However, the final average in various scenarios is about 2s. This fully demonstrates that the algorithm is effective, and its average efficiency is reliable in the obstacle avoidance scenario of the dual-arm robot.

5. Conclusions

Based on the traditional RRT algorithm, this paper introduces the goal probability bias and A* cost function algorithm. The search time and efficiency of the RRT algorithm is greatly optimized. In addition, the generated paths are also locally optimized. The effectiveness of the GA_RRT algorithm is first verified by the simulation of planar obstacle avoidance path planning. Furthermore, the algorithm is applied to obstacle avoidance path planning simulation of a dual-arm robot. In the process of obstacle avoidance, not only static obstacles are considered, but also the influence of dynamic manipulator obstacles in real time. Finally, the simulation results show that the algorithm can realize the cooperative obstacle avoidance path planning of a dual-arm robot. In terms of search efficiency, it is also superior to the traditional RRT algorithm.

As a future research task, the experimental verification process will first need to be improved. In addition, the adaptive step size will be further used to improve the search efficiency of the algorithm. The visual method is also considered to detect the dynamic obstacles in the environment, so as to realize the dual-arm path planning in unstructured environments.

Author Contributions: Conceptualization, W.S. and M.T.; methodology, W.S.; software, W.S.; validation, W.S., C.Z. and M.T.; formal analysis, K.W.; investigation, M.T.; resources, W.S.; data curation, W.S.; writing—original draft preparation, W.S. and M.T.; writing—review and editing, K.W. and C.Z.; visualization, W.S.; supervision, K.W.; project administration, C.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by China's manned space program.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Bohren, J.; Rusu, R.B.; Jones, E.G.; Marder-Eppstein, E.; Pantofaru, C.; Wise, M.; Mösenlechner, L.; Meeussen, W.; Holzer, S. Towards autonomous robotic butlers: Lessons learned with the PR2. In Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 5568–5575.
2. Matthias, B.; Kock, S.; Jerregard, H.; Kallman, M.; Lundberg, I.; Mellander, R. Safety of collaborative industrial robots: Certification possibilities for a collaborative assembly robot concept. In Proceedings of the 2011 IEEE International Symposium on Assembly and Manufacturing (ISAM), Tampere, Finland, 25–27 May 2011; pp. 1–6.
3. Connolly, C. Motoman markets co-operative and humanoid industrial robots. *Ind. Robot: Int. J.* **2009**, *36*, 417–420. [CrossRef]
4. Fuchs, M.; Borst, C.; Giordano, P.R.; Baumann, A.; Kraemer, E.; Langwald, J.; Gruber, R.; Seitz, N.; Plank, G.; Kunze, K.; et al. Rollin'Justin-Design considerations and realization of a mobile platform for a humanoid upper body. In Proceedings of the 2009 IEEE International Conference on Robotics and Automation, Kobe, Japan, 12–17 May 2009; pp. 4131–4137.
5. Wang, M. *Research on Collaborative Motion Planning and Control Methods of Dual-Arm Robots for Rescue Tasks*; University of Science and Technology of China: Hefei, China, 2015.
6. Peng, Y.C.; Carabis, D.S.; Wen, J.T. Collaborative manipulation with multiple dual arm robots under human guidance. *Int. J. Intell. Robot. Appl.* **2018**, *2*, 252–266. [CrossRef]
7. Khatib, O. Real-time obstacle avoidance for manipulators and mobile robots. In *Autonomous Robot Vehicles*; Springer: New York, NY, USA, 1986; pp. 396–404.
8. Abhishek, T.S.; Schilberg, D.; Doss, A.S.A. Obstacle Avoidance Algorithms: A Review. In Proceedings of the International Conference on Robotics, Intelligent Automation and Control Technologies (RIACT 2020), Chennai, India, 2–3 October 2020; Volume 1012, p. 012052.
9. Palmieri, G.; Scoccia, C.; Palpacelli, M.C.; Callegari, M. Motion planning and control of redundant manipulators for dynamical obstacle avoidance. *Machines* **2021**, *9*, 121. [CrossRef]
10. Duchoň, F.; Babinec, A.; Kajan, M.; Beňo, P.; Florek, M.; Fico, T.; Jurišica, L. Path planning with modified a star algorithm for a mobile robot. *Procedia Eng.* **2014**, *96*, 59–69. [CrossRef]
11. Guruji, A.K.; Agarwal, H.; Parsediya, D.K. Time-efficient A* algorithm for robot path planning. *Procedia Technol.* **2016**, *23*, 144–149. [CrossRef]
12. Han, B.; Luo, X.; Luo, Q.; Zhao, Y.; Lin, B. *Research on Obstacle Avoidance Motion Planning Technology of 6-DOF Manipulator. International Conference on Geometry and Graphics*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 604–614.
13. Wang, J.; Liu, S.; Zhang, B.; Yu, C. Manipulation Planning with Soft Constraints by Randomized Exploration of the Composite Configuration Space. *Int. J. Control. Autom. Syst.* **2021**, *19*, 1340–1351. [CrossRef]
14. Glasius, R.; Komoda, A.; Gielen, S.C.A.M. Neural network dynamics for path planning and obstacle avoidance. *Neural Netw.* **1995**, *8*, 125–133. [CrossRef]
15. Sung, I.; Choi, B.; Nielsen, P. On the training of a neural network for online path planning with offline path planning algorithms. *Int. J. Inf. Manag.* **2021**, *57*, 102142. [CrossRef]
16. Gerke, M. Genetic path planning for mobile robots. In Proceedings of the American Control Conference, San Diego, CA, USA, 2–4 June 1999; pp. 596–601.
17. Kang, H.B. *Mechanical Dual-Arms: Collision-Free Path Planning of Robot Based on Improved Ant Colony Algorithms*; Northeastern University: Shenyang, China, 2010.
18. Tuncer, A.; Yildirim, M. Dynamic path planning of mobile robots with improved genetic algorithm. *Comput. Electr. Eng.* **2012**, *38*, 1564–1572. [CrossRef]
19. Dorigo, M.; Maniezzo, V.; Colorni, A. Ant system: Optimization by a colony of cooperating agents. *IEEE Trans. Syst. Man Cybern. Part B (Cybern.)* **1996**, *26*, 29–41. [CrossRef] [PubMed]
20. Silvia, M.; Irene, L. An ant colony system for responsive dynamic vehicle routing. *Eur. J. Oper. Res.* **2015**, *245*, 704–718.
21. Zong, C.; Lu, L.; Lei, X.; Zhao, P. A path planning approach for multi-dof spatial manipulator via A* algorithm. *J. Hefei Univ. Technol. (Nat. Sci.)* **2017**, *40*, 164–168.
22. Dijkstra, E.W. A note on two problems in connexion with graphs. *Numer. Math.* **1959**, *1*, 269–271. [CrossRef]
23. Sanchez, G.; Latombe, J.C. Using a PRM planner to compare centralized and decoupled planning for multi-robot systems. In Proceedings of the IEEE International Conference on Robotics and Automation, Washington, DC, USA, 11–15 May 2002; pp. 2112–2119.
24. Karaman, S.; Frazzoli, E. Sampling-based algorithms for optimal motion planning. *Int. J. Robot. Res.* **2011**, *30*, 846–894. [CrossRef]
25. Jaillet, L.; Cortés, J.; Siméon, T. 'Sampling-based path planning on configuration-space costmaps. *IEEE Trans. Robot.* **2010**, *26*, 635–646. [CrossRef]
26. Pepy, R.; Lambert, A. Safe path planning in an uncertain-configuration space using RRT. In Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, China, 9–15 October 2006; pp. 5376–5381.
27. LaValle, S.M.; Kuffner, J.J.; James, J. Randomized Kino-dynamic Planning. *Int. J. Robot. Res.* **1999**, *15*, 378–400.
28. Urmson, C.; Simmons, R. Approaches for heuristically biasing RRT growth. In Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No. 03CH37453), Las Vegas, NV, USA, 27–31 October 2003; Volume 2, pp. 1178–1183.

29. Li, Z.; Ma, H.; Zhang, X.; Fei, Q. Path planning of the dual-arm robot based on VT-RRT algorithm. In Proceedings of the 2019 Chinese Control Conference (CCC), Guangzhou, China, 27–30 July 2019; pp. 4359–4364.
30. Jiang, H.; Jiang, X. An improved path planning algorithm based on RRT. *J. Chongqing Univ. Technol. (Nat. Sci.)* **2021**, *35*, 10–16.
31. Shao, L.; Liu, H.; Chen, C.; Du, D.; Li, J.; Liu, H. Path Planning for Mobile Robots Based on Improved RRT Algorithm. In Proceedings of the 2020 IEEE International Conference on Mechatronics and Automation (ICMA), Beijing, China, 13–16 October 2020; pp. 1240–1244.
32. Wei, K.; Ren, B. A method on dynamic path planning for robotic manipulator autonomous obstacle avoidance based on an improved RRT algorithm. *Sensors* **2018**, *18*, 571. [CrossRef] [PubMed]
33. Andreas, V.; Knut, G. An optimization-based approach to dual-arm motion planning with closed kinematics. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Madrid, Spain, 1–5 October 2018; pp. 8346–8351.
34. Kim, D.H.; Lim, S.J.; Lee, D.H.; Lee, J.; Han, C. A RRT-based motion planning of dual-arm robot for (Dis)assembly tasks. In Proceedings of the IEEE Conference on Intelligence and Safety for Robotics, Seoul, Korea, 24–26 October 2013. 6p.
35. Li, Y.; Xu, D. Cooperative path planning of dual-arm robot based on gravitational adaptive step RRT. *Robot* **2020**, *42*, 606–616.