machines

MDPI

Article

Path Planning for Manipulators of Automotive Welding Unit Based on an Improved RRT* Algorithm

Xiang Li [1], Pengxiang Wang [2,*], Yuchun Xu [3] and Jihong Yan [1,*]

[1] School of Mechatronics Engineering, Harbin Institute of Technology, Harbin 150001, China
[2] School of Energy Science and Engineering, Harbin Institute of Technology, Harbin 150001, China
[3] School of Engineering and Innovation, Aston University, Birmingham B4 7ET, UK
* Correspondence: pxwang@hit.edu.cn (P.W.); jyan@hit.edu.cn (J.Y.)

Abstract

An automotive welding unit is a modular production cell within a welding workshop that integrates industrial manipulators, welding equipment, fixtures, and control systems to perform specific welding and assembly tasks. A large number of industrial manipulators are utilized in the automotive welding unit. The capability to quickly plan a short and collision-free path in the workspace of the manipulator is of great importance for improving the manipulator's intelligence level and production efficiency. The RRT* algorithm, based on random sampling, has been widely applied in path planning for high-dimensional manipulators due to its probabilistic completeness and powerful exploration capabilities. However, the RRT* algorithm performs poorly in spaces containing narrow passages. Research on the practical application of path planning for 6-DOF manipulators is still insufficient, particularly in planning posture. To solve these two problems, an improved RRT* algorithm is proposed in this paper. New sampling and node connection strategies are designed to improve the expansion and convergence speed of the random tree in spaces containing narrow passages. A distance-constrained posture quaternion interpolation method is presented to generate smooth and continuous paths for manipulators of the automotive welding unit. Simulations and experiments are carried out to validate the proposed method, which confirms that the method can plan collision-free paths for manipulators more quickly compared to other methods.

Keywords: path planning; improved RRT* algorithm; industrial manipulator; automotive welding unit

Check for updates

Academic Editor: Raffaele Di Gregorio

Received: 10 March 2026
Revised: 6 April 2026
Accepted: 15 April 2026
Published: 17 April 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Industrial manipulators have been widely deployed across diverse manufacturing environments to collaborate with or replace human workers in performing production tasks [1–3]. However, due to the increasing complexity of production tasks and higher demands on production capacity, automation alone can no longer satisfy current manufacturing needs. Consequently, many countries have advocated for the vigorous development of smart manufacturing technologies to drive the transformation of manufacturing from automation to intelligence, thereby enhancing production efficiency and flexibility [4]. Manipulators are typically programmed through manual teaching or simulation to coordinate their movements and prevent collisions in automotive welding units [5]. The path planning for manipulators is often inefficient, leading to a significant amount of redundant movement, which reduces production efficiency.

The aim of path planning methods is to find a collision-free, continuous path under specified constraints, based on certain evaluation criteria [6]. Currently, the prevalent methods for path planning include sampling-based algorithms such as Probabilistic Road Map (PRM) [7] and Rapidly-exploring Random Tree (RRT) [8]. Graph-based search algorithms include the A* algorithm [9] and Dijkstra algorithm [10], while local planning algorithms include the Artificial Potential Field (APF) method [11]. The APF method offers certain advantages for manipulators with low degrees of freedom. However, it is challenging to determine the free space or define the potential field of a manipulator under high degrees of freedom. Artificial intelligence-based algorithms include Artificial Neural Network (ANN) [12], Genetic Algorithm (GA) [13] and Ant Colony algorithm (ACO) [14]. The core idea of these methods is to transform the path planning problem for manipulators into an optimization problem and then solve this optimization problem to obtain feasible paths. However, these algorithms converge slowly and are prone to getting stuck in local optima.

The RRT algorithm stands out among these techniques for path planning of industrial manipulators in complex environments due to its advantages of rapid search capabilities and robust performance in high-dimensional configuration spaces [15]. Furthermore, various variants of the RRT algorithm have been proposed, including RRT* [16], RRT*-Smart [17], Informed-RRT [18], etc.

Shi et al. [19] introduced a path planning method for a dual-arm robot based on target probability bias and cost functions within GA-RRT, reducing the randomness and blindness inherent in the RRT algorithm during scaling. Jiang et al. [20] employed the Improved-RRT method with hybrid constrained sampling and incorporated the APF approach into the process of node expansion. The adaptive gravitational fields, dynamic step sizes, and node constraint strategies reduced excessive exploration of the configuration space. Qi et al. [21] refined the RRT*FN algorithm by employing a heuristic sampling technique that prioritizes removing leaf nodes outside the ellipsoid. New nodes are generated through target and random point gravity, with weights dynamically adjusted using a binary search approach. Gao et al. [22] proposed a BP-RRT* method for path planning of manipulators based on an improved RRT* combined with a BP neural network to address the problem of slow path planning in multi-obstacle spaces. Meng et al. [23] proposed an RRT* method specifically designed for continuous manipulators. Experimental results have demonstrated the superiority of this W-space RRT* path planner over its C-space counterpart in obstacle avoidance and trajectory tracking. Connell and La [24] developed an enhanced RRT algorithm for environments with non-fixed obstacles. Building upon the initial path obtained by the standard RRT algorithm, predictions are made for potential collisions between the manipulator and obstacles according to their respective motion states. Wang et al. [25] proposed an improved RRT algorithm for path planning in multi-obstacle scenarios. Simulation results validated that the algorithm enables the manipulator to autonomously search for a collision-free path. Cao et al. [26] introduced an improved RRT algorithm based on GA specifically for the task of fruit picking. The initial path search method is similar to traditional RRT, but after obtaining the initial path, GA is employed to optimize it. Xie et al. [27] integrated the robotic inverse kinematics into the RRT algorithm. Following the acquisition of the end-effector pose, they calculated joint angles through inverse solutions to determine the state of each link, ensuring that neither the manipulator's end-effector nor its joints collided with obstacles.

Based on the above research findings, traditional RRT algorithms and their variants exhibit certain limitations, such as low sampling efficiency in narrow passages and slow convergence rates. Therefore, this paper proposes a novel sampling strategy for the RRT* algorithm to effectively enhance sampling density in narrow passages. For generated

paths, a new connection strategy is adopted to accelerate convergence. Furthermore, to address practical application challenges of the improved RRT* algorithm in automotive welding units, a distance-constrained posture quaternion interpolation method is introduced to constrain the end-effector's posture. The main contributions of this paper are as follows:

1. An improved RRT* algorithm is proposed by incorporating a sampling strategy based on a virtual force field, which enhances exploration efficiency and reduces redundant trajectories in complex environments.
2. A novel path optimization strategy is developed to further shorten the trajectory length and improve path quality.
3. A distance-constrained quaternion interpolation method is introduced to ensure smooth and stable orientation transitions of the manipulator's end-effector during motion.

The remainder of this paper is organized into the following sections. Section 2 presents the design and theoretical formulation of the improved RRT* algorithm and the distance-constrained quaternion interpolation method. Section 3 describes the application of the proposed methods to path planning of a manipulator, including the integration with collision detection models to generate collision-free paths. Section 4 validates the effectiveness of the proposed approach through simulation. Finally, the conclusions are given in the last section.

2. Improved RRT* Algorithm and Control Method of Manipulator's Posture

Due to the random sampling strategy of the RRT* algorithm, it can fully explore the free space and generate nodes. However, this also causes the extended tree formed by the RRT* algorithm to grow predominantly in free space, resulting in insufficient nodes generated in narrow passages. If the theoretically optimal path occurs near a narrow passage, obtaining the optimal path requires substantial time and computational resources as nodes accumulate in the growing tree, even though RRT* can perform incremental optimization. Therefore, this paper proposes a sampling strategy based on a virtual force field. Specifically, when a randomly generated node falls into the obstacle space, it is not discarded directly. Instead, a virtual attractive force is exerted on this node by nearby valid nodes in the existing tree that lie in free space. Under the influence of this virtual force, the node is iteratively adjusted and guided toward the boundary of the obstacle and eventually into a feasible free-space region. By transforming invalid samples into valid ones near obstacle boundaries, this strategy effectively increases the sampling density in narrow passages, thereby improving the probability of discovering feasible and near-optimal paths in constrained environments.

2.1. Sampling Strategy Based on Virtual Force Field

To move nodes from the obstacle space into free space, the movement direction $\mathbf{d} = (x_1, x_2, \ldots, x_D)$ for each node $\mathbf{n} = (m_1, m_2, \ldots, m_D)$ must be determined, where D is the dimension of space. The selection criterion for $\mathbf{d}$ is that nodes moving in this direction can rapidly escape the obstacle space. Nodes falling into an obstacle space receive gravitational force from nodes outside that space within a certain range, and this gravitational force decreases as the distance between nodes increases [7]. Gravity is defined as follows: for any two nodes $\mathbf{n}_1$ and $\mathbf{n}_2$ within the obstacle space and outside the space,

$$\mathbf{F}(\mathbf{n}_1, \mathbf{n}_2) = \begin{cases} \frac{A}{1+dist^2(\mathbf{n}_1,\mathbf{n}_2)} \cdot \frac{(\mathbf{n}_1-\mathbf{n}_2)}{|\mathbf{n}_1-\mathbf{n}_2|}, dist(\mathbf{n}_1, \mathbf{n}_2) \leq R \\ 0 \qquad\qquad\qquad\quad , dist(\mathbf{n}_1, \mathbf{n}_2) > R \end{cases} \tag{1}$$

where A is a preset constant, and R represents the range of gravitational influence. $dist(\mathbf{n}_1, \mathbf{n}_2)$ denotes the distance between two nodes, defined as

$$dist(\mathbf{n}_1, \mathbf{n}_2) = \left(\sum_{i=1}^{D} \left(\mathbf{n}_1^i - \mathbf{n}_2^i\right)^2\right)^{1/2} \quad (2)$$

Parameter A is the strength coefficient of the virtual repulsive force, which is used to control the speed and amplitude at which samples move from the obstacle regions to the free space. In this paper, it is set as an empirical constant, and optimal performance is achieved through experimental tuning.

The virtual gravitational force acting on each node within the obstacle space can be calculated by the above method. The direction of this virtual gravitational force then serves as the node's direction of motion. The node moves along this direction until it enters free space, as shown in Figure 1.

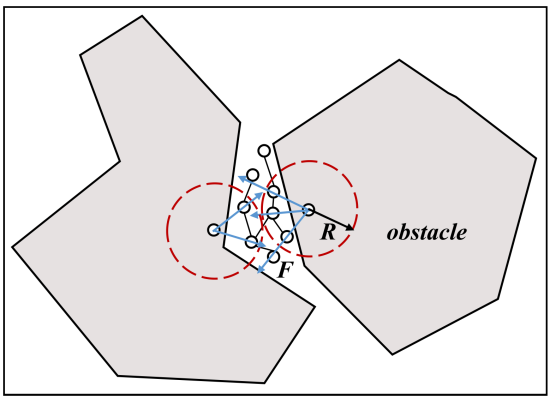


Figure 1. Gravitational force acting on nodes within obstacle space.

2.2. Connection Strategy After Obtaining the Initial Path

After obtaining the first collision-free path, relying solely on incremental optimization of the RRT* algorithm requires substantial time and computational resources to converge to the optimal path. Therefore, a novel connection strategy is adopted to accelerate the convergence speed of the improved RRT* algorithm. The whole process is as follows. The algorithm initiates from the start configuration q_{start}. The goal configuration is selected as the random node q_{rand} with a certain probability in order to reduce the randomness of expansion and accelerate convergence. Then the nearest node q_{near} is found and extended towards q_{rand} to obtain a new node q_{new}. A parent node is reselected, and the rewiring process is performed, which can optimize the path to some extent. If the new node q_{new} can pass the collision detection, it will be retained. Otherwise, it will be discarded. In this way, iterative sampling is carried out to make the path reach the goal configuration q_{end}. These nodes are connected to obtain the feasible path of the manipulator from q_{start} to q_{end}. Then perturbations are added to nodes along the path, similar to the simulated annealing algorithm. And additional nodes along the connecting lines are inserted between nodes. The improved RRT* algorithm directly connects the obtained nodes based on the triangle inequality principle (a direct connection between two nodes can yield a shorter path than indirect connections), thereby shortening the trajectory length.

The efficiency and accuracy of the improved RRT* algorithm are significantly influenced by both the search space and step size. Therefore, the search space and step size must be reasonably set to meet practical requirements. The pseudo-code of the improved RRT* algorithm is shown in Algorithm 1.

Algorithm 1. Pseudo-code of improved RRT* algorithm

Improved-RRT* ($q_{start} \rightarrow q_{end}$)

```
Tree ← Initialize_Tree ()
Tree ← Insert_Node (q_start, Tree)
For i = 1:N do
    If rand() > 0.3
        q_rand ← Sample (i)
    Else
        q_rand ← q_end
    q_nearest ← Nearest_Distance (Tree, q_rand)
    (q_new, Tree) ← Steer (q_rand, q_nearest)
    If Collision_Free (q_new)
    q_near ← Near (Tree, q_new, r)
    q_min ← Choose (q_near, q_nearest, q_new)
    Else
    q_new ← Move (Tree, q_new)
    q_near ← Near (Tree, q_new, r)
    q_min ← Choose (q_near, q_nearest, q_new)
    T ← Insert_Node (Tree, q_min, q_new)
    T ← Rewire (Tree, q_near, q_new, q_min)
    If PathFound
    (T, Cost) ← Path_Optimization (Tree, q_start, q_end)
Return T
```

We compared our improved RRT* algorithm with RRT*, RRT*-Smart, and Informed-RRT*. The test environment consisted of a 50 × 50 two-dimensional plane containing four rectangular obstacles and five circular obstacles, and narrow passages were formed between the obstacles. The same parameters were set for each algorithm as shown in Table 1. The computer is configured with an Intel i7-13700KF CPU and 32 GB of RAM. The results of each algorithm are shown in Figure 2.

Table 1. Parameters are set for each algorithm.

Parameters	Setting Value
Starting Point	[8, 5]
Target Point	[42, 20]
Step Size	4
Offset Probability	0.3
Search Radius	15
Number of Iterations	10,000

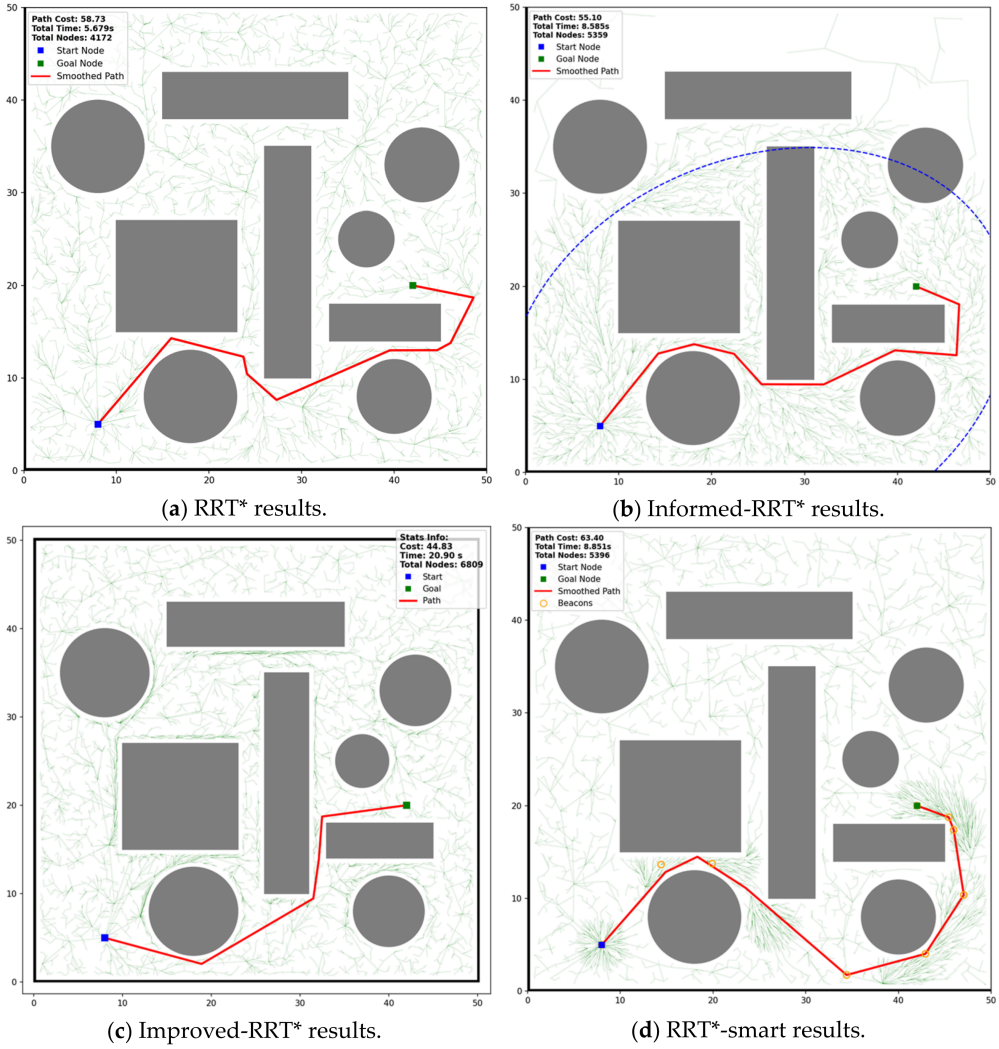
With the same number of iterations, the improved RRT* algorithm completed the search in 20.9 s and achieved the shortest path cost of 44.83 among the four algorithms. Furthermore, the improved RRT* algorithm generated more nodes in the narrow passages, with a total of 6809 nodes. This demonstrates that the improved RRT* algorithm can thoroughly explore the narrow passages and exhibits fast convergence efficiency.

2.3. Control Method of Manipulator's Posture

2.3.1. Quaternion

The improved RRT* algorithm developed in this paper generates positions in the manipulator's operational space, specifically the X, Y, and Z coordinates in Cartesian space,

without controlling the end-effector's posture. This limitation may lead to collision risks. To address this problem, a quaternion-based posture interpolation method is investigated to facilitate control of the manipulator's posture.



(**a**) RRT* results. (**b**) Informed-RRT* results.

(**c**) Improved-RRT* results. (**d**) RRT*-smart results.

Figure 2. Path quality comparison of four algorithms with 10,000 iterations in a sample run.

A quaternion is a mathematical concept created by Irish mathematician William Rowan Hamilton in 1843. The unit quaternion can be used to represent rotation in three-dimensional space. It is equivalent to the other two commonly used representations (three-dimensional orthogonal matrix and Euler angles), but avoids the gimbal lock problem in the representation of Euler angles. Given a quaternion, it can be expressed as follows:

$$\mathbf{q} = w + xi + yj + zk = [\boldsymbol{s}, \boldsymbol{v}] \tag{3}$$

The Rodrigues formula can be used to derive the rotation matrix R from a quaternion $\mathbf{q}$:

$$R = \begin{bmatrix} 1-2y^2-2z^2 & 2(xy-wz) & 2(xz+wy) \\ 2(xy+wz) & 1-2x^2-2z^2 & 2(yz-wx) \\ 2(xz-wy) & 2(yz+wx) & 1-2x^2-2y^2 \end{bmatrix} \tag{4}$$

To derive the quaternion from a rotation matrix, one may first calculate any single element from w, x, y, or z using Equation (4), then solve for the remaining elements. For instance, to calculate w, we sum the diagonal elements in R, yielding

$$r_{11} + r_{22} + r_{33} = 3 - 4x^2 - 4y^2 - 4z^2 \tag{5}$$

For a unit quaternion, the sum of squares of its components equals 1. Thus, w can be obtained as

$$w = \frac{\sqrt{1 + r_{11} + r_{22} + r_{33}}}{2} \tag{6}$$

Having computed w, the remaining components of the quaternion can be determined as follows:

$$x = \frac{r_{32} - r_{23}}{4w} \tag{7}$$

$$y = \frac{r_{13} - r_{31}}{4w} \tag{8}$$

$$z = \frac{r_{21} - r_{12}}{4w} \tag{9}$$

2.3.2. Spherical Linear Interpolation

Spherical linear interpolation (Slerp) is a linear interpolation technique for quaternions, primarily used to smoothly interpolate between two quaternions representing rotations. The Slerp formula for quaternions can be derived as follows:

$$Slerp(\mathbf{p}, \mathbf{q}, t) = \frac{\sin[(1-t)\theta]\mathbf{p} + \sin t\theta\mathbf{q}}{\sin\theta} \tag{10}$$

where $\mathbf{p}$ and $\mathbf{q}$ are two three-dimensional vectors in Cartesian space. θ denotes the angle between $\mathbf{p}$ and $\mathbf{q}$. t is a parameter.

The tree of nodes generated by the improved RRT* algorithm expands progressively outward from the root node. Each positional node generated in the tree, it requires matching the end-effector's posture. Furthermore, in order to ensure the smooth motion of the manipulator, abrupt changes in the manipulator's posture must be avoided. The execution process of the distance-constrained posture quaternion interpolation method proposed in this paper is illustrated in Figure 3. First, Slerp is performed on the manipulator's initial and final postures to obtain a series of intermediate postures. The environment dictates the number of interpolation points, which are distributed at regular intervals. Then a radius r is determined based on the Euclidean distance between the manipulator's initial and final positions. This radius is used to partition the space into several regions centered at the start point. Finally, for each node within a region, the manipulator's posture within the posture interval is matched. This approach ensures a smooth transition of the manipulator's posture from the initial to the final configuration, while maintaining positional constraints.

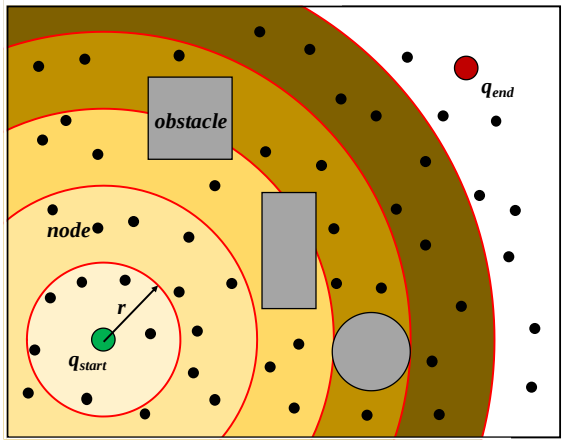


Figure 3. Diagram of distance-constrained posture quaternion interpolation.

3. Path Planning for Manipulators Based on Improved RRT* Algorithm

Robot models typically comprise link models and kinematic models. Link models describe the positional relationships between a manipulator's joints, while kinematic models describe the transformation between the end-effector's Cartesian space and joint space representations.

3.1. Coordinate System of Links

This paper focuses on the 165F and 210F industrial manipulators from the R-2000iB series developed by FANUC Corporation. When establishing the link coordinate system, the arm connected to the joint is modeled as a rigid body. Therefore, the structural model of the manipulator can be constructed through the spatial description and transformation relationships of rigid bodies. The Fanuc R-2000iB link coordinate system is shown in Figure 4. The D-H parameters of the manipulator are shown in Table 2.

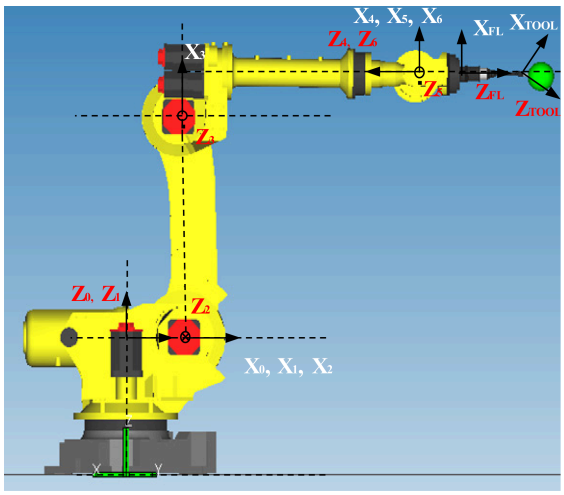


Figure 4. Link coordinate system of manipulator.

Table 2. D-H parameter table of manipulator.

Link	a_{i-1} (mm)	α_{i-1} (rad)	d_i (mm)	θ_i (rad)	θ_i Range (°)
1	0	0	0	θ_1	−180~180
2	312	−pi/2	0	θ_2 − pi/2	−60~76
3	1075	pi	0	$\theta_2 + \theta_3$	−132~230
4	225	−pi/2	−1280	θ_4	−360~360
5	0	pi/2	0	θ_5	−125~125
6	0	−pi/2	0	θ_6	−360~360
Flange	0	pi	−215 (165F)/−235 (210F)	θ_7	−360~360

3.2. Forward Kinematics of Robot

The forward kinematics of a manipulator is characterized by solving the end-effector's position and posture relative to the manipulator's base coordinate system. The joint configuration of the manipulator is represented as (θ_1, θ_2, θ_3, θ_4, θ_5, θ_6), and the end-effector's posture is expressed utilizing the RPY (Roll, Pitch, Yaw) angle. This posture is represented as a six-dimensional vector (p_x, p_y, p_z, α, β, γ). Since each rotation of the coordinate system is performed around a fixed coordinate system, the transformation matrix of the moving coordinate system relative to the fixed coordinate system can be derived by consecutively multiplying from the left:

$$R = \mathrm{Rot}(Z, \gamma)\mathrm{Rot}(Y, \beta)\mathrm{Rot}(X, \alpha) \tag{11}$$

The transformation matrix ${}^{i}_{i+1}T$ of the adjacent coordinate system T_i to T_{i+1} can be expressed as

$$ {}^{i}_{i+1}T = \mathrm{Rot}(X_i, \alpha_i)\mathrm{Trans}(a_i, 0, 0)\mathrm{Rot}(Z_i, \theta_i)\mathrm{Trans}(0, 0, d_i) \tag{12} $$

3.3. Inverse Kinematics of Robot

The inverse kinematics representation of a six-axis manipulator involves solving for the rotational angles of its six joints based on the end-effector's posture, which essentially amounts to solving a nonlinear system of equations. This paper employs an analytical method to determine the manipulator's inverse kinematic solution, a technique suitable for offline scenarios that delivers high computational accuracy.

The analytical method yields eight sets of inverse solutions for the manipulator. However, only one set of inverse solutions can represent the six joint angles of the actual manipulator. Therefore, the eight sets of inverse solutions are filtered according to the following principles:

1. Constraint principle for joint angles. The joint angles derived from the manipulator's inverse kinematics must be within the range of joint angles. If any single angle in a set of inverse solutions exceeds this range, the entire set is discarded.
2. Principle of minimal angular displacement. The angular displacements of the manipulator's six joints should be minimized. However, since the length of the link corresponding to each axis of the manipulator is different, the impact of each axis on the overall system differs. Therefore, the impact weights of six axes are introduced:

$$ \eta = [0.551, 1, 0.5, 0.1, 0.1, 0.1] \tag{13} $$

To quantify this effect, the link lengths of joints 1–3 are normalized using min–max normalization within the range of their actual physical dimensions. In this study, the link lengths of these joints fall within the interval [0.5, 1]. Therefore, the normalization range [0.5, 1] is selected to preserve the relative physical significance of each joint. The value 0.551 is obtained from this normalization process. It corresponds to the normalized length of the first joint link relative to the maximum and minimum link lengths among joints 1–3. This ensures that the weight assignment is physically meaningful rather than arbitrarily chosen.

For joints 4–6, since their coordinate origins coincide and their link lengths are relatively small, their influence on the end-effector position is limited. Therefore, a smaller uniform weight of 0.1 is assigned.

Simultaneously, an evaluation function is introduced to minimize the impact of angular displacement:

$$ F = \min\left(\sum_{i=1}^{6} \eta_i \left|\theta_i - \theta_i'\right|\right) \tag{14} $$

Equation (14) defines an evaluation function for selecting the optimal inverse kinematics solution based on the principle of minimizing joint motion variation. Specifically, the function computes the weighted sum of absolute differences between the current joint angles and candidate joint angles. By minimizing this function, the selected solution ensures the smallest overall joint movement, taking into account the different influences of each joint through the weight factors. This helps improve motion smoothness and reduces unnecessary joint displacement during trajectory execution.

3.4. Model of Collision Detection for Manipulator

An Oriented Bounding Box (OBB) is a commonly used bounding box format. An OBB selects vertices or spatial point clouds from the enclosed object and uses methods like PCA

to find the optimal orientation of its own coordinate axes. Consequently, the resulting bounding box is typically not parallel to the world coordinate axes. When an object moves arbitrarily in space, the shape of the OBB remains unchanged.

3.4.1. Cylindrical Bounding Box for Manipulators

Industrial manipulators consist of a series of linkages, typically cylindrical or slightly tapered frustums in shape. Based on the OBB concept, a cylindrical bounding box is created for the manipulator. This cylindrical bounding box is a specialized form of OBB, offering minimal redundant space and high detection accuracy. Actual external dimensions of the manipulator are captured, including link lengths and radius. During manipulator motion, changes in the centerline of the cylindrical bounding box correspond directly to joint position variations. Since the manipulator's links are rigid bodies with fixed radius, the bounding box model's radius remains constant. This allows for the unique determination of the cylindrical bounding box model whose shape remains unchanged regardless of the manipulator's position. The method for establishing the cylindrical bounding box is illustrated in Figure 5.

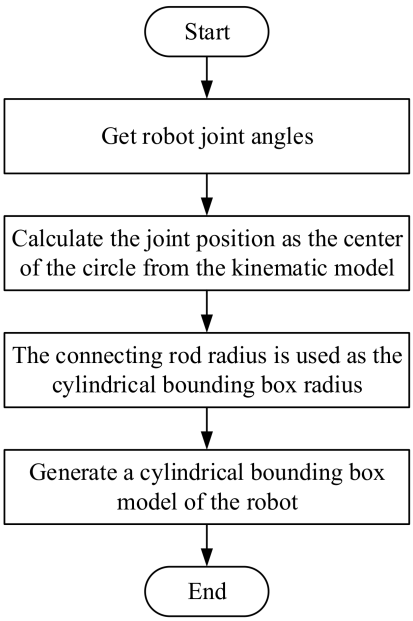


Figure 5. Construction method of cylindrical bounding box model.

To ensure the safety of the manipulator during operation, the cylindrical bounding box is not fitted precisely to the manipulator's link surfaces. A safe distance should be maintained between the manipulator and other objects during obstacle avoidance. This paper adopts a bounding box expansion strategy, which involves enlarging the envelope range of the cylindrical bounding box by appropriately expanding its radius and height, with an expansion factor λ = 1.2 selected.

3.4.2. Bounding Box Model for Obstacles

Obstacles within automotive welding units primarily include manipulators other than collision detection systems, work-in-progress body panels, and bases. Considering the relatively regular external shapes of these obstacles, the OBB bounding box model is uniformly adopted. The method for establishing the OBB bounding box is shown in Figure 6.

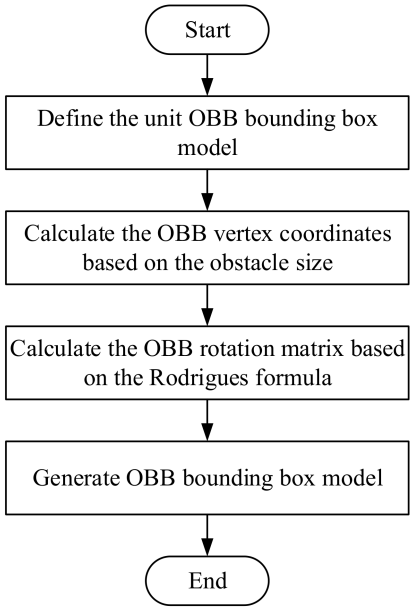


Figure 6. Construction method of OBB model for obstacles.

4. Simulation-Based Validation and Field Implementation Assessment

4.1. Results and Analysis of Simulation Experiment

The preceding sections have elucidated the theoretical research on path planning for industrial manipulators. In the practical implementation, it is necessary to consider the specific operational conditions within the industrial environment. The welding unit of a particular automotive manufacturing facility is shown in Figure 7.



Figure 7. Automotive welding unit.

Figure 8 delineates the application of the proposed method in an automotive welding unit within a specific manufacturing facility. The procedure encompasses the following steps.

1. Transformation of the manipulator's initial and target joint angles via forward kinematics to obtain the end-effector's initial and target positions, as well as the corresponding initial and target postures.
2. Generation of a set of random positions and postures through an improved RRT* algorithm coupled with quaternion interpolation for posture.
3. Computation of the manipulator's six joint angles using inverse kinematics for each generated position and posture.

4. Evaluation of the computed joint angles using a collision detection model. If a collision is detected, the random position and posture are regenerated. Conversely, if no collision is detected, the node is incorporated into the expansion tree.
5. After reaching a predefined maximum number of iterations, all feasible paths connecting the start and goal nodes in the improved RRT* tree are extracted. The optimal path is then selected by minimizing the total path length, defined as the sum of Euclidean distances between consecutive nodes.

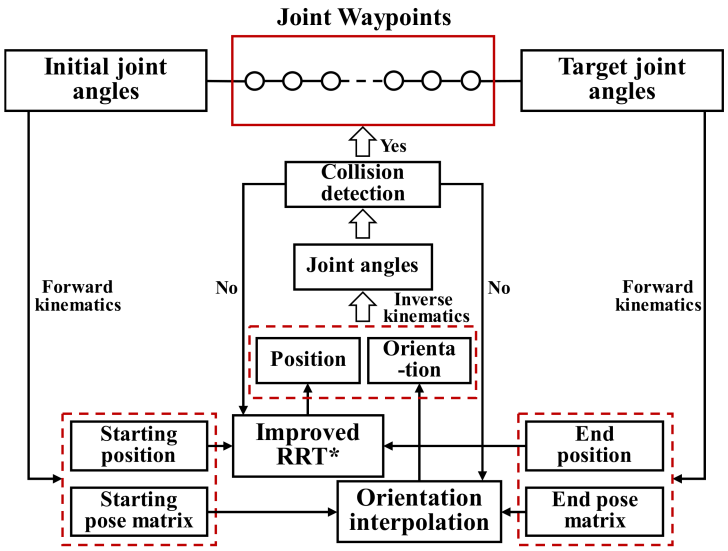


Figure 8. Improved RRT* algorithm applied to actual manipulator.

To validate the effectiveness of the proposed method for path planning in an automotive welding unit, an optimization study was conducted on the welding unit of an actual production facility. In this specific operation stage, only the No. 1 manipulator is actively involved in the task, which consists of removing the welded automotive body panel using a flip fixture and transferring it to the subsequent processing unit. The No. 2 and No. 3 manipulators remain stationary during this process and are therefore modeled as static obstacles in the environment. Consequently, path planning is performed exclusively for the No. 1 manipulator, while the configurations of the No. 2 and No. 3 manipulators are incorporated into the collision detection framework to ensure collision-free motion. The motion waypoints for the No. 1 manipulator during this operation are shown in Table 3, while Table 4 presents the corresponding joint angles. Tables 5 and 6, respectively, illustrate the pose configurations and joint angles for the No. 2 and No. 3 manipulators throughout this process.

Table 3. Waypoints of industrial manipulator No. 1.

Sequence	X (mm)	Y (mm)	Z (mm)	R (deg)	P (deg)	Y (deg)
1	−1277.56	2009.52	1075.66	−136.954	−61.674	43.133
2	380.923	1485.474	1691.494	−110.242	−28.279	−0.968
3	622.208	562.172	1993.64	−88.938	−26.21	−10.196
4	633.228	492.385	2168.98	−89.216	−25.241	−9.677
5	−269.178	−808.938	2145.324	−92.827	−22.943	−158.638
6	−1410.502	−1467.555	1433.845	−114.867	6.479	161.849

The path optimization for this section was executed for the automotive welding unit as described in this paper. The algorithm was implemented in Python 3.11, generating

optimized waypoints as presented in Table 7. The chart of the corresponding joint angles of No. 1 manipulator at the following time is shown in Figure 9.

Table 4. Joint angles of industrial manipulator No. 1.

Sequence	J1 (deg)	J2 (deg)	J3 (deg)	J4 (deg)	J5 (deg)	J6 (deg)
1	148.058	11.412	−11.893	74.083	−56.526	68.421
2	112.959	−30.709	34.651	37.814	−61.492	88.079
3	86.564	−56.862	67.144	7.472	−66.099	113.308
4	89.511	−46.077	76.056	9.717	−75.282	112.904
5	−61.873	−42.035	75.201	8.611	−77.731	110.798
6	−132.305	−6.877	34.838	−26.62	−64.682	107.033

Table 5. End-effector's positions of industrial manipulators No. 2 and No. 3.

Robot No.	X (mm)	Y (mm)	Z (mm)	R (deg)	P (deg)	Y (deg)
2	951.648	−1174	2830.98	−1.91	−74.185	−52.34
3	1417.39	−132.937	307.027	1.221	−0.118	−64.958

Table 6. Joint angles of industrial manipulators No. 2 and No. 3.

Robot No.	J1 (deg)	J2 (deg)	J3 (deg)	J4 (deg)	J5 (deg)	J6 (deg)
2	−44.07	−13.336	62.596	15.433	−34.012	−7.469
3	24.098	−20.504	2.131	0.098	−90.909	−89.052

Table 7. Path optimization algorithm outputs the joint angles of manipulator No. 1.

Sequence	J1 (deg)	J2 (deg)	J3 (deg)	J4 (deg)	J5 (deg)	J6 (deg)
1	148.058	11.412	−11.893	74.083	−56.526	68.421
2	143.788	−20.377	5.485	71.120	−57.909	71.479
3	118.141	−41.135	57.707	30.673	−73.657	128.126
4	89.511	−46.077	76.056	9.717	−75.282	112.904
5	−61.873	−42.035	75.201	8.611	−77.731	110.798
6	−132.305	−6.877	34.838	−26.62	−64.682	107.033

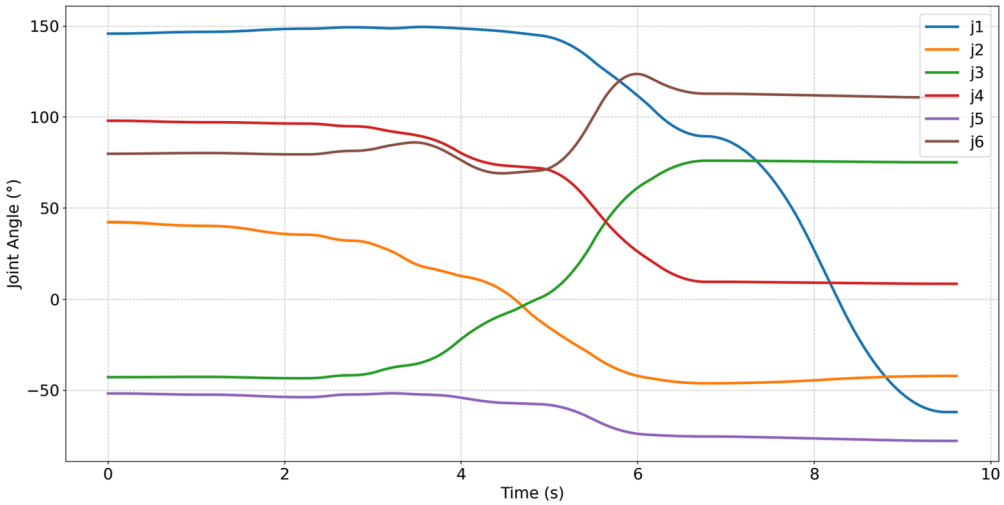
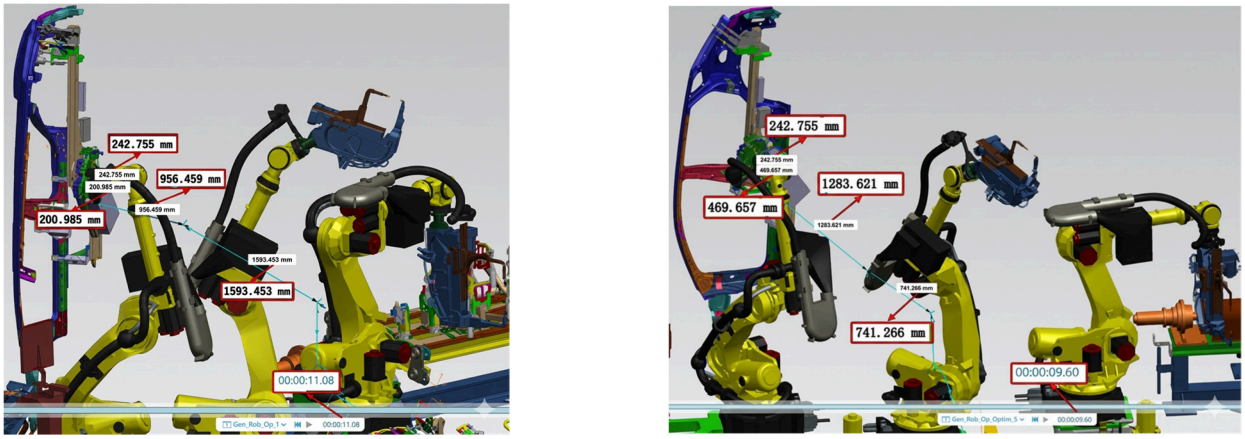


Figure 9. The optimized curve of joint angles of manipulator No. 1.

A simulation model of welding unit No. 1 was developed using Tecnomatix Process Simulate 2307 to accurately replicate the actual production environment. The optimized

algorithm output was then integrated into this simulation model. The results demonstrated the following:

1. No collisions were observed.
2. The trajectory of manipulator No. 1 was significantly optimized.
3. The cycle time for this segment was reduced from 11.08 s to 9.60 s, representing a 13.36% improvement in processing efficiency.
4. The end-effector's travel distance decreased from 2993.652 mm to 2737.299 mm. These optimizations are visually represented in Figure 10.

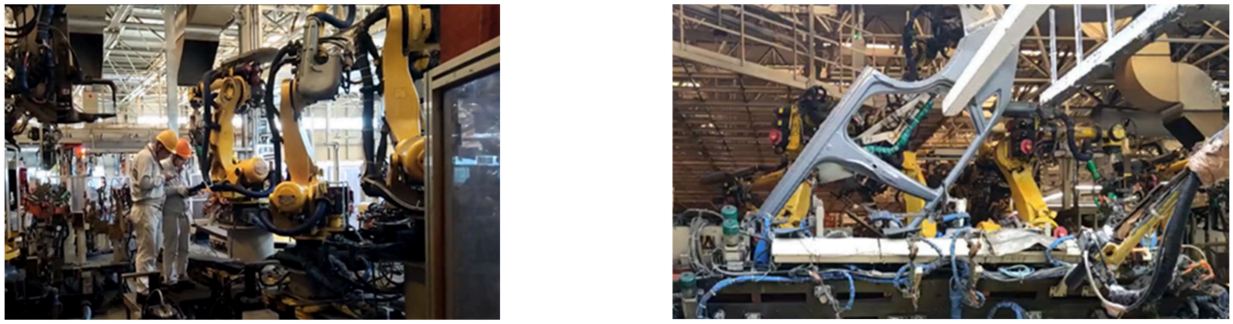


(**a**) (**b**)

Figure 10. Comparison of results before and after segment optimization. (**a**) Before optimization and (**b**) after optimization.

4.2. Field Implementation

Based on the waypoints derived from the simulation, manipulator teaching was performed in the automotive welding unit. The operational status of the manipulator was observed and recorded, as shown in Figure 11. During the on-site implementation, the manipulator successfully followed the planned trajectory without collision or interference with surrounding equipment. Compared with the original manually taught path, the optimized trajectory generated by the proposed method is smoother and more compact, with fewer redundant movements. In addition, the execution time of the operation was reduced, indicating an improvement in task efficiency. These observations demonstrate that the planned path is feasible and effective in the real environment, and the practical performance is consistent with the trends observed in the simulation results.



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

Figure 11. Field application of the improved RRT* algorithm. (**a**) Teaching process of manipulators and (**b**) motion process of manipulators.

5. Conclusions

This paper addresses the challenge of path planning for automotive welding units, offering a solution to mitigate redundant trajectories in complex production environments. The conducted research enhances the traditional RRT* algorithm through the sampling strategy based on a virtual force field and a novel path optimization strategy. The proposed method improves the manipulator's operational stability via a distance-constrained quaternion interpolation method. Through the Tecnomatix Process Simulate 2307 software, a high-fidelity simulation model was developed to accurately replicate the actual welding unit. The results demonstrate that the proposed approach not only enables effective obstacle avoidance but also achieves target positions with reduced end-effector travel distances, thereby enhancing overall production efficiency. This methodology offers significant practical value for path planning in automotive welding manipulators. Furthermore, it provides valuable theoretical and practical insights applicable to diverse industrial scenarios and analogous robotic systems.

Author Contributions: Conceptualization, J.Y. and P.W.; methodology, X.L.; software, X.L.; validation, Y.X.; investigation, P.W.; writing—original draft preparation, X.L.; writing—review and editing, J.Y.; supervision, Y.X.; funding acquisition, J.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundations of China, grant number 52275482.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: The authors declare no potential conflicts of interest with respect to the research, authorship, and/or publication of this paper.

References

1. Bartoš, M.; Bulej, V.; Bohušík, M.; Stanček, J.; Ivanov, V.; Macek, P. An overview of robot applications in automotive industry. *Transp. Res. Procedia* **2021**, *55*, 837–844. [CrossRef]
2. Dzedzickis, A.; Subačiūtė-Žemaitienė, J.; Šutinys, E.; Samukaitė-Bubnienė, U.; Bučinskas, V. Advanced applications of industrial robotics: New trends and possibilities. *Appl. Sci.* **2021**, *12*, 135. [CrossRef]
3. Goel, R.; Gupta, P. *Robotics and Industry 4.0*; Advances in Science, Technology & Innovation; Springer: Cham, Switzerland, 2020; pp. 157–169.
4. Pillai, R.; Sivathanu, B.; Mariani, M.; Rana, N.P.; Yang, B.; Dwivedi, Y.K. Adoption of AI-empowered industrial robots in auto component manufacturing companies. *Prod. Plan. Control* **2022**, *33*, 1517–1533. [CrossRef]
5. Lee, J.S.; Chua, P.C.; Chen, L.Q.; Ng, P.H.N.; Kim, Y.; Wu, Q.; Jeon, S.; Jung, J.H.; Chang, S.H.; Moon, S.K. Key enabling technologies for smart factory in automotive industry: Status and applications. *Int. J. Precis. Eng. Manuf. Smart Technol.* **2023**, *1*, 93–105. [CrossRef]
6. Wang, Z.; Chang, J.; Li, B.; Wang, C.; Liu, C. Application of improved rapidly-exploring random trees (RRT) algorithm for obstacle avoidance of snake-like manipulator. In Proceedings of the 2020 IEEE International Conference on Mechatronics and Automation (ICMA), Beijing, China, 13–16 October 2020; pp. 490–495.
7. Chen, G.; Luo, N.; Liu, D.; Zhao, Z.H.; Liang, C.C. Path planning for manipulators based on an improved probabilistic roadmap method. *Robot. Comput. Integr. Manuf.* **2021**, *72*, 102196. [CrossRef]
8. Li, B.H.; Chen, B.D. An adaptive rapidly-exploring random tree. *IEEE/CAA J. Autom. Sin.* **2021**, *9*, 283–294. [CrossRef]
9. Han, C.Y.; Li, B.Y. Mobile robot path planning based on improved A* algorithm. In Proceedings of the 2023 IEEE 11th Joint International Information Technology and Artificial Intelligence Conference (ITAIC), Chongqing, China, 8–10 December 2023; pp. 672–676.
10. Alshammrei, S.; Boubaker, S.; Kolsi, L. Improved Dijkstra algorithm for mobile robot path planning and obstacle avoidance. *Comput. Mater. Contin.* **2022**, *72*, 5939–5954. [CrossRef]
11. Batista, J.; Souza, D.; Silva, J.; Ramos, K.; Costa, L.D.R.; Braga, A. Trajectory planning using artificial potential fields with metaheuristics. *IEEE Lat. Am. Trans.* **2020**, *18*, 914–922. [CrossRef]

12. Yu, J.L.; Su, Y.C.; Liao, Y.F. The path planning of mobile robot by neural networks and hierarchical reinforcement learning. *Front. Neurorobotics* **2020**, *14*, 63. [CrossRef] [PubMed]
13. Rahmaniar, W.; Rakhmania, A.E. Mobile robot path planning in a trajectory with multiple obstacles using genetic algorithms. *J. Robot. Control (JRC)* **2022**, *3*, 1–7. [CrossRef]
14. Miao, C.W.; Chen, G.Z.; Yan, C.L.; Wu, Y.Y. Path planning optimization of indoor mobile robot based on adaptive ant colony algorithm. *Comput. Ind. Eng.* **2021**, *156*, 107230. [CrossRef]
15. Zhou, X.; Wang, X.W.; Gu, X.S. An approach for solving the three-objective arc welding robot path planning problem. *Eng. Optim.* **2023**, *55*, 650–667. [CrossRef]
16. Karaman, S.; Frazzoli, E. Sampling-based algorithms for optimal motion planning. *Int. J. Robot. Res.* **2011**, *30*, 846–894.
17. Nasir, J.; Islam, F.; Malik, U.; Ayaz, Y.; Hasan, O.; Khan, M.; Muhammad, M.S. RRT*-SMART: A rapid convergence implementation of RRT. *Int. J. Adv. Robot. Syst.* **2013**, *10*, 299. [CrossRef]
18. 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. In Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, IL, USA, 14–18 September 2014; pp. 2997–3004.
19. Shi, W.B.; Wang, K.; Zhao, C.; Tian, M.Q. Obstacle avoidance path planning for the dual-arm robot based on an improved RRT algorithm. *Appl. Sci.* **2022**, *12*, 4087. [CrossRef]
20. Jiang, L.H.; Liu, S.Y.; Cui, Y.M.; Jiang, H.X. Path planning for robotic manipulator in complex multi-obstacle environment based on improved_RRT. *IEEE/ASME Trans. Mechatron.* **2022**, *27*, 4774–4785. [CrossRef]
21. Qi, J.Y.; Yuan, Q.N.; Wang, C.; Du, X.Y.; Du, F.L.; Ren, A. Path planning and collision avoidance based on the RRT* FN framework for a robotic manipulator in various scenarios. *Complex Intell. Syst.* **2023**, *9*, 7475–7494. [CrossRef]
22. Gao, Q.Y.; Yuan, Q.N.; Sun, Y.; Xu, L.Y. Path planning algorithm of robot arm based on improved RRT* and BP neural network algorithm. *J. King Saud Univ. Comput. Inf. Sci.* **2023**, *35*, 101650. [CrossRef]
23. Meng, B.H.; Godage, I.S.; Kanj, I. RRT*-based path planning for continuum arms. *IEEE Robot. Autom. Lett.* **2022**, *7*, 6830–6837. [CrossRef]
24. Connell, D.; La, M.H. Dynamic path planning and replanning for mobile robots using RRT. In Proceedings of the 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Banff, AB, Canada, 5–8 October 2017; pp. 1429–1434.
25. Wang, Z.P.; Li, Y.S.; Zhang, H.; Liu, C.; Chen, Q.J. Sampling-based optimal motion planning with smart exploration and exploitation. *IEEE/ASME Trans. Mechatron.* **2020**, *25*, 2376–2386.
26. Cao, X.M.; Zou, X.J.; Jia, C.Y.; Chen, M.Y.; Zeng, Z.Q. RRT-based path planning for an intelligent litchi-picking manipulator. *Comput. Electron. Agric.* **2019**, *156*, 105–118. [CrossRef]
27. Xie, Y.; Zhang, Z.D.; Wu, X.D.; Shi, Z.; Chen, Y.Y.; Wu, B.X. Obstacle avoidance and path planning for multi-joint manipulator in a space robot. *IEEE Access* **2019**, *8*, 3511–3526. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.