Received 30 August 2023, accepted 19 October 2023, date of publication 26 October 2023, date of current version 3 November 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3327915

RESEARCH ARTICLE

Robot Motion Planning Based on Improved RRT Algorithm and RBF Neural Network Sliding

MENGLONG CAO, XINJIE ZHOU, AND YUNPENG JU
College of Automation and Electronic Engineering, Qingdao University of Science and Technology, Qingdao 266042, China

Corresponding author: Yunpeng Ju (03297@qust.edu.cn)

This work was supported by the Natural Science Foundation of Shandong Province under Grant ZR2020MF087.

ABSTRACT Aiming to address the limitations of the traditional rapidly-exploring random tree (RRT) algorithm in robotic arm path planning, such as high randomness, slow planning speed, non-smooth paths, and excessive corners, an improved RRT algorithm incorporating a target bias strategy and artificial potential field approach is proposed. Firstly, this algorithm employs a probabilistic sampling strategy to facilitate quicker growth of the tree structure towards the target point, thus accelerating the planning speed of the RRT algorithm. Subsequently, a repetitive greedy strategy is introduced to reduce redundant nodes in the path and enhance search efficiency. Furthermore, an artificial potential field combined with a target bias strategy is integrated to guide the RRT algorithm towards the target direction. Finally, a third-order B-spline curve optimization is introduced to ensure smoother paths. Compared to traditional RRT algorithms and RRT* algorithms, the improved RRT algorithm exhibited an increase in planning speed of 46.1% and 27.0%, 41.8% and 20.9% for two-dimensional and three-dimensional environments respectively. Path lengths were reduced by 20.9% and 10.6%, 22.0% and 8.7%, while the number of search nodes notably decreased. Considering the complexity of robotic arm operations and uncertainties in system parameters that hinder the establishment of accurate mathematical modeling, a sliding mode control based on radial basis function (RBF) neural networks is proposed. To substantiate the enduring stability of the designed controller, we harness the Lyapunov stability principle. The algorithm validation for path planning and trajectory tracking of the robotic arm are conducted in the MATLAB simulation environment. Experimental results illustrate significant improvements in terms of iteration count, planning speed, path length, and smoothness of the RRT algorithm. Moreover, the motion trajectory of the robotic arm is basically consistent with the expected trajectory.

INDEX TERMS Improved RRT algorithm, RBF neural network, sliding mode control, sports planning.

I. INTRODUCTION

The motion planning of robotic arms encompasses two main aspects: path planning and trajectory planning. Path planning involves determining the optimal motion path for a robotic arm from its starting position to a target location, while trajectory planning aims to determine the joint angles or end effector's position and orientation, enabling the robotic arm to execute precise movements [1]. As the most widely used equipment in the field of industrial robotics, robotic arms have the capability to replace humans in performing a multitude of repetitive and hazardous tasks, thereby enhancing production efficiency and workplace safety [2]. With the increasing diversity and complexity of work environments, path planning algorithms need to continuously innovate and optimize to tackle challenges in various situations. Simultaneously, trajectory planning focuses on the joint angles of the robotic arm or the position and orientation of its end effector to achieve precise trajectory tracking. Currently, motion planning extends beyond the confines of traditional industrial manufacturing. With the emergence of fields such as service robotics and medical robotics, the application scenarios of robotic arms are constantly expanding. An increasing amount of research is devoted to issues such as human-robot collaboration, planning in dynamic environments, and enhancing learning capabilities [3], [4], [5].

The associate editor coordinating the review of this manuscript and approving it for publication was Mark Kok Yew Ng.

A. PATH PLANNING SEGMENT

Currently, prevalent methods for robotic arm path planning include sampling-based search and graph-based search approaches [6]. Because objects and obstacles in two-dimensional space are usually planar in shape, collision detection is relatively simple, and basic geometric principles can be used for collision detection, making it easier to handle obstacle avoidance problems. However, in three-dimensional spaces, the complexity of state and action spaces intensifies, resulting in a sharp increase in computational complexity for graph search algorithms. These algorithms demand more time and computational resources to find feasible paths. Among sampling-based approaches, rapidly-exploring random trees (RRT) are widely used [7]. The RRT algorithm effectively finds feasible paths in complex environments by employing random sampling and tree growth. RRT demonstrates good applicability in handling high-dimensional state spaces and intricate obstacles, yet it faces issues like prolonged planning time and tendency to generate redundant nodes. Consequently, researchers have proposed various improvements to the algorithm.

Zhang et al. [8] introduced a regression mechanism to improve the RRT algorithm to prevent over searching the configuration space. Gu et al. [9] utilized RRT for global planning and incorporated target heuristic information, accelerating algorithm convergence. Yuan et al. [10] proposed an improved fast exploration random tree algorithm based on heuristic probability bias and target factors. This combination led to faster convergence and evasion of local minima. Mashayekhi et al. [11] introduced a dual-tree search algorithm that combines bidirectional and unidirectional search capabilities, offering higher success rates in pathfinding. Pohan et al. [12] presented an enhanced RRT algorithm integrated with ant colony systems, amalgamating RRT's strengths with ant colony characteristics to achieve optimal convergence at a favorable pace. Zhang et al. [13] proposed an improved RRT algorithm combining it with the Salp Swarm Algorithm (SSA) and utilizing the predatory behavior of sea squirt colonies to reduce redundant node generation and enhance search efficiency. Xu et al. [14] combined the artificial potential field algorithm with RRT to mitigate oscillations present in artificial potential field path planning methods.

Building on the discussions, this paper proposes an improved RRT path planning algorithm. Firstly, the algorithm employs a probabilistic sampling strategy, enabling the tree structure to grow towards the target point more rapidly, thus accelerating the planning speed of the RRT algorithm. Subsequently, a repeated greedy strategy is introduced to reduce redundant nodes in the path, enhancing search efficiency. Following this, an artificial potential field combined with a target bias strategy is incorporated, directing the algorithm to favor searching in the direction of the target point. Finally, a third-order B-spline curve optimization is integrated to achieve smoother paths.

B. TRAJECTORY PLANNING SEGMENT

Sliding mode control (SMC) as a robust control method, exhibits distinct advantages in robotic arm trajectory tracking [15]. As an important component of modern industrial automation systems, robotic arms need to achieve high-precision, fast response, and robust motion control. However, due to the nonlinear nature of robotic arm systems, external disturbances, and model uncertainties, conventional control methods often fall short in addressing these challenges [16]. In this context, sliding mode control has gained significant attention for its outstanding performance in robotic arm trajectory tracking. By introducing a sliding surface, sliding mode control achieves robust control over system states. The design of the sliding surface enables the controller to respond quickly and resist external disturbances, thus achieving highly accurate trajectory tracking [17]. Almakhles et al. [18] combined integral sliding mode and backstepping sliding mode to form a dual-loop control structure, ensuring robust trajectory tracking capabilities in the presence of disturbances. Xu et al. [19] designed an adaptive sliding mode fault-tolerant controller based on improved reaching law to estimate position disturbances, enhancing the system's dynamic performance. Jouila et al. [20] researched a sliding mode controller based on wavelet neural networks, employing wavelet networks to approximate uncertainties and mitigate their impact. Mehran et al. [21] proposed a sliding mode control for robotic arms based on an extended grey wolf optimizer, greatly mitigating the influence of external disturbances. Yin et al. [22] introduced an adaptive sliding mode control based on lateral deviation, effectively reducing the chattering phenomenon. An et al. [23] combined iterative learning and sliding mode control, resulting in superior trajectory tracking with smaller tracking errors and faster learning rates compared to traditional methods. Addressing high-precision trajectory tracking for robotic arms, Xian et al. [24] proposed a continuous sliding mode control (CSMC) scheme based on time-varying disturbance estimation and compensation, avoiding the chattering of traditional sliding mode control and enhancing disturbance resistance.

For the challenges posed by the complexity of a 2-DOF robotic arm's real-world operations and uncertainties in system parameters, making accurate mathematical modeling problematic, this paper introduces a sliding mode control based on RBF neural networks. The RBF neural network approximates uncertainties in the model [23], while the sliding mode controller adjusts parameters in real-time. Drawing upon the Lyapunov stability principle, the paper proves the asymptotic stability of the designed RBF-SMC controller, enabling the robotic arm system to converge to the desired trajectory within a finite time and achieve high-precision tracking performance.

The remainder of this paper is structured as follows. In Section Two, the problem of robotic arm path planning is introduced, encompassing traditional RRT algorithms and their improvements. Section Three delves into robotic arm trajectory planning, including the establishment of the robotic arm's dynamic model and controller design. In Section Four, simulation experiments validate the feasibility of the proposed algorithm. Lastly, Section Five presents the paper's conclusions, outlines its limitations, and provides an outlook for future developments.

II. ROBOTIC ARM PATH PLANNING

A. RRT ALGORITHM

The RRT algorithm was introduced by Steven M. LaValle and James J. Kuffner Jr. as a method for efficiently exploring non-convex high-dimensional spaces using a randomly constructed space filling tree [25]. The core idea behind RRT is to explore the feasible space by continuously sampling and expanding, while avoiding intersections with obstacles during the expansion process. This incremental search approach allows RRT to gradually generate paths and adapt well to high-dimensional and complex problems. Due to its stochastic nature, RRT explores the space uniformly and efficiently, reducing search bias and preventing it from getting trapped in local optima. Furthermore, RRT doesn't require a pre-built global map; it performs path planning based on local information, thus offering some level of real-time capability. As depicted in Figure 1, in the traditional RRT algorithm, the starting position is taken as the root node. A state point X_{rand} is randomly sampled from the environment. The nearest node X_{near} to X_{rand} is located in the constructed tree. The direction of the connection line between X_{near} and X_{rand} indicates the growth direction of the tree. The algorithm checks whether the connection from X_{near} to X_{rand} intersects with obstacles. If it does, the new node is discarded; otherwise, the X_{new} node is added to the tree. This process is repeated until a newly generated node's distance to the target point becomes smaller than a predefined step length. At this point, the tree growth stops, and the newly added node is connected to the target point, yielding the discovered path.

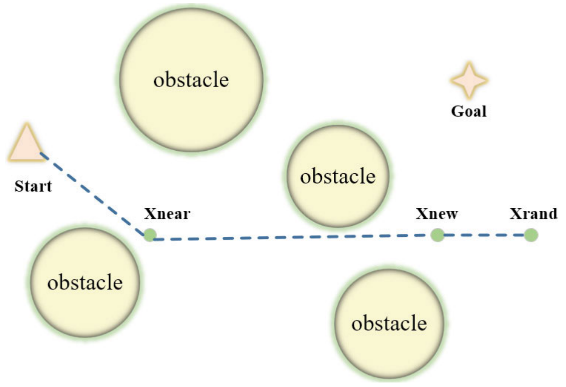


FIGURE 1. RRT algorithm spanning tree extension process.

B. IMPROVED RRT ALGORITHM

1) ALGORITHM WITH PROBABILITY SAMPLING STRATEGY

In the traditional RRT algorithm, random samples are uniformly drawn from the feasible space. However, this uniform sampling approach can have limitations. For instance, more samples might be concentrated around the robot's initial position and the open space around it, while regions farther from the goal might be less explored. This can result in slower tree growth and require more iterations to find a valid path.

To address the above issues, a probability sampling strategy is introduced. This strategy adjusts the distribution probability of sampling points in the feasible space. Depending on the location of the goal point, the sampling points' generation probability can be adjusted to favor generating points closer to the goal. This adjustment encourages the tree structure to grow more likely in the direction from the initial point towards the goal, reducing ineffective exploration in distant regions and thus improving the planning speed of the algorithm.

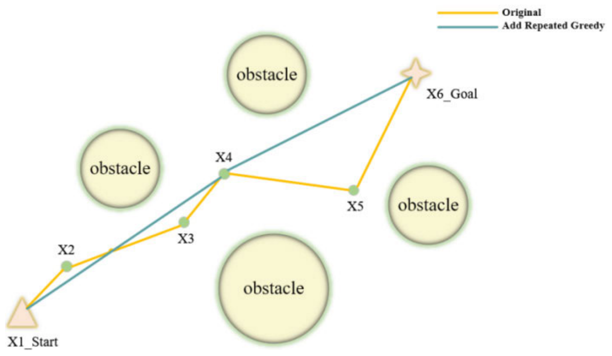


FIGURE 2. Repeated greedy strategy to Remove Redundant Nodes.

2) PATH OPTIMIZATION STRATEGY

The addition of the repeated greedy optimization strategy aims to enhance the extension process of the RRT algorithm. During node expansion, the repeated greedy strategy selects candidate nodes with the shortest distance to the goal node. However, a single application of the greedy strategy might still result in redundant nodes. By employing the repeated strategy, the number of redundant nodes is significantly reduced. As depicted in Figure 2, assuming the starting point is X_i and the goal point is X_j, a connection is established between them. If there are no obstacles between the two points, the intermediate nodes between X_i and X_j can be removed. In this case, $j=j+1$, and the process is repeated. However, if obstacles are encountered between X_i and X_j, the connection cannot be omitted. Instead, the previous goal point X_{j-1} is retained and utilized as the new parent node for further updates.

The repeated greedy strategy, while effective in selecting nodes that move toward the goal, may sometimes overlook certain factors, potentially resulting in generated paths that do not find the global optimal solution. To overcome the limitations of the greedy strategy, we introduce an artificial potential field method for further optimization. Additionally, to navigate through narrow passages and avoid local minima, we incorporate a target bias strategy into the artificial potential field. This enhances the effectiveness of path planning and is formulated as shown in Equation (1):

$$F = \alpha * F_{\text{attraction}} + \beta * F_{repulsion} \tag{1}$$

In the equation, F stands for the resultant force of the artificial potential field, with $F_{attraction}$ and $F_{repulsion}$ representing the attractive and repulsive forces of the artificial potential field respectively. The coefficients α and β are used as bias factors to regulate the respective impacts of the attractive and repulsive forces on the resultant force. By considering the attractive force towards the goal and the repulsive force from obstacles during the path planning process, the artificial potential field method can comprehensively incorporate environmental information and guide the selection of nodes. By introducing the artificial potential field method, we can enhance the intelligence of the RRT algorithm in path planning, overcome the limitations of the repeated greedy strategy, and further improve the quality of path planning.

Lastly, to achieve smoother paths, a cubic B-spline curve was employed for smoothing. The curve equation is given by Equation (2):

$$C(u) = \sum_{i=0}^{n} N_{i,k}(u) P_i \tag{2}$$

where $N_{i,k}(u)(i = 0, 1, \ldots, n)$ represents the k-th degree B-spline basis function, and $P_i(i = 0, 1, \ldots, n)$ stands for control points. The recursive formula for $N_{i,k}(u)(i = 0, 1, \ldots, n)$ is as follows:

$$N_{i,0}(u) = \begin{cases} 1 & if\ u_i \leq u \leq u_{i+1} \\ 0 & otherwise \end{cases} \tag{3}$$

$$N_{i,k}(u) = \frac{u - u_i}{u_{i+k} - u_i} N_{i,k-1}(u) + \frac{u_{i+k-1} - u}{u_{i+k-1} - u_{i+1}} N_{i+1,k-1}(u) \tag{4}$$

where k denotes the order of the spline function, and i is the index of the spline function. For the i-th segment of the curve, its representation is as follows:

$$ki(t) = \frac{1}{6}[t^3 t^2 t 1] \begin{vmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{vmatrix} \begin{bmatrix} P_i \\ P_{i+1} \\ P_{i+2} \\ P_{i+3} \end{bmatrix} \tag{5}$$

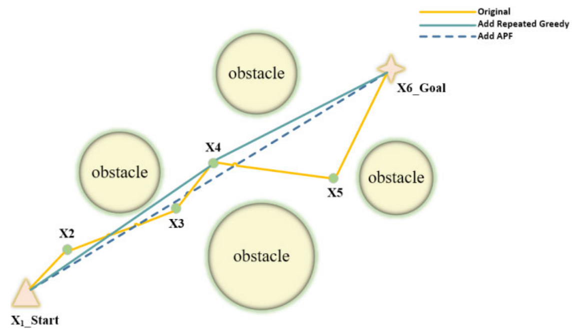


FIGURE 3. Repeated greedy strategy for removing redundant nodes after adding artificial potential fields.

The improved RRT algorithm workflow is illustrated in Figure 4.

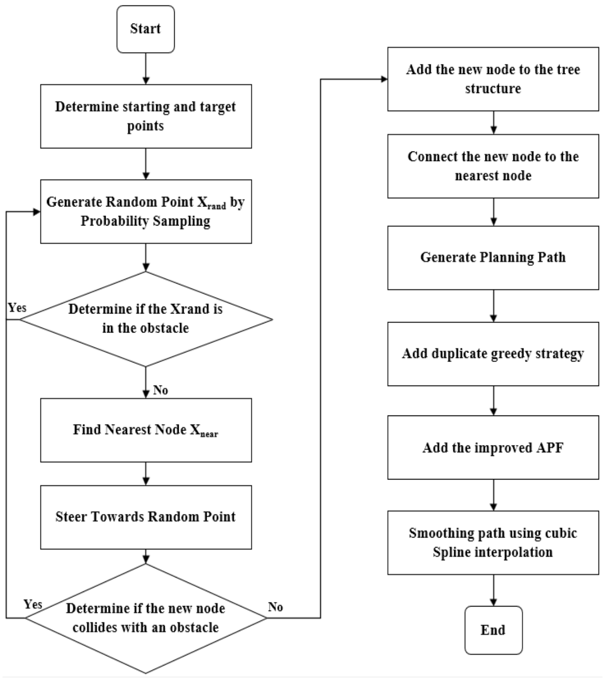

Start
Determine starting and target points
Generate Random Point X_rand by Probability Sampling
Determine if the Xrand is in the obstacle
Yes
No
Find Nearest Node X_near
Steer Towards Random Point
Determine if the new node collides with an obstacle
Yes
No
Add the new node to the tree structure
Connect the new node to the nearest node
Generate Planning Path
Add duplicate greedy strategy
Add the improved APF
Smoothing path using cubic Spline interpolation
End

FIGURE 4. Workflow diagram of the improved RRT algorithm.

III. ROBOTIC ARM TRAJECTORY PLANNING

A. ROBOTIC ARM KINEMATIC MODEL

Having an accurate kinematic model of the robotic arm is essential to implement various control methods. Common methods for modeling robotic arms include the Newton-Euler method and the Lagrange method [26]. The Newton-Euler method decomposes the robotic arm system into multiple components, treating each component as a subsystem. By applying rigid body mechanics principles, the dynamic recursive relationships between subsystems are derived, ultimately leading to the mathematical model of the entire robotic arm system. Its advantage lies in effectively handling complex mechanical structures and systems with multiple degrees of freedom, providing direct solutions for joint forces and torques. However, for intricate systems, the method involves extensive mechanical computations, making the analysis challenging.

On the other hand, the Lagrange method utilizes Lagrange mechanics principles. It calculates the system's kinetic and potential energies and substitutes them into the Lagrange equations, yielding the system's motion equations and constraint equations. This approach doesn't require considering the internal details of the system, thereby simplifying the analysis and solution processes.

When a robot has more degrees of freedom, it can perform more complex and flexible movements and operations. Multi-degree-of-freedom robots can move and rotate in different directions to adapt to various work scenarios and task requirements. For example, a robot with six degrees of freedom can achieve translation and rotation in three-dimensional space, enabling it to perform more complex and precise actions. However, as the degrees of freedom increase, the complexity of the robot's system also increases. Each degree of freedom introduces a new motion variable and constraint, requiring consideration of factors such as joint limitations, link lengths, and obstacle avoidance to ensure that the robot can safely and effectively execute the required tasks. Therefore, this article uses a two-degree-of-freedom robotic arm to study trajectory planning for robotic arms. The structure of the two-degree-of-freedom robotic arm is shown in Figure 5.

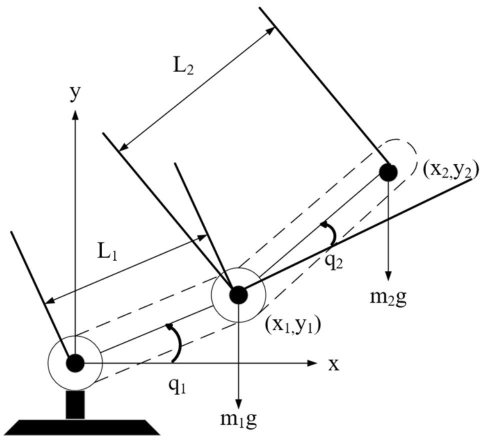


FIGURE 5. **Diagram of a two-degree-of-freedom robotic arm structure.**

Where m_1, m_2, L_1, L_2, q_1, and q_2 represent the masses, lengths, and angular displacement's of the two links. Assuming the masses of the links are concentrated at their respective endpoints, the coordinates of the endpoints of the two links are given by (x_1, y_1) and (x_2, y_2). The links are in a serial arrangement, meaning that when the first link moves, the second link also follows in rotation. The kinetic and potential energies of the first link are expressed as:

$$\begin{aligned} K_1 &= \frac{1}{2} m_1 l_1^2 \dot{q}_1^2 \\ P_1 &= m_1 g l_1 \sin(q_1) \end{aligned} \tag{6}$$

The position and velocity coordinates of the second link are given by:

$$\begin{cases} x_2 = l_1 \cos(q_1) + l_2 \cos(q_1 + q_2) \\ \dot{x}_2 = -l_1 \dot{q}_1 \sin(q_1) - l_2(\dot{q}_1 + \dot{q}_2)\sin(q_1 + q_2) \\ y_2 = l_1 \sin(q_1) + l_2 \sin(q_1 + q_2) \\ \dot{y}_2 = l_1 \dot{q}_1 \cos(q_1) + l_2(\dot{q}_1 + \dot{q}_2)\cos(q_1 + q_2) \end{cases} \tag{7}$$

Assuming $s_1 = \sin(q_1)$, $s_2 = \sin(q_2)$, $c_1 = \cos(q_1)$, $c_2 = \cos(q_2)$, $s_{12} = \sin(q_1 + q_2)$, $c_{12} = \cos(q_1 + q_2)$, the linear velocity of the second link is:

$$\begin{aligned} v_2^2 &= \dot{x}_2^2 + \dot{y}_2^2 \\ &= l_1^2 \dot{q}_1^2 + l_2^2(\dot{q}_1 + \dot{q}_2)^2 + 2 l_1 l_2 (\dot{q}_1^2 + \dot{q}_1 \dot{q}_2) c_2 \end{aligned} \tag{8}$$

This allows us to determine the kinetic energy of the second link as:

$$\begin{aligned} K_2 &= \frac{1}{2} m_2 v_2^2 = \frac{1}{2} m_2 l_1^2 \dot{q}_1^2 \\ &\quad + \frac{1}{2} m_2 l_2^2 (\dot{q}_1 + \dot{q}_2)^2 + m_2 l_1 l_2 (\dot{q}_1^2 + \dot{q}_1 \dot{q}_2) c_2 \end{aligned} \tag{9}$$

The potential energy is:

$$P_2 = m_2 g y_2 = m_2 g (l_1 s_1 + l_2 s_{12}) \tag{10}$$

From the Lagrange equation, it can be concluded that:

$$\begin{aligned} L &= K - P = (K_1 + K_2) - (P_1 + P_2) \\ &= \frac{1}{2}(m_1 + m_2) l_1^2 \dot{q}_1^2 + m_2 l_1 l_2 (\dot{q}_1^2 + \dot{q}_1 \dot{q}_2) c_2 \\ &\quad + \frac{1}{2} m_2 l_2^2 (\dot{q}_1 + \dot{q}_2)^2 - (m_1 + m_2) g l_1 s_1 - m_2 g l_2 s_{12} \end{aligned} \tag{11}$$

To derive the robotic arm's dynamic model, we can use the Euler-Lagrange equation, resulting in:

$$\begin{aligned} \frac{\partial L}{\partial q_1} &= -(m_1 + m_2) g l_1 c_1 - m_2 g l_2 c_{12} \\ \frac{\partial L}{\partial \dot{q}_1} &= (m_1 + m_2) l_1^2 \dot{q}_1^2 + m_2 l_2^2 (\dot{q}_1 + \dot{q}_2) \\ &\quad + m_2 l_1 l_2 (2\dot{q}_1 + \dot{q}_2) c_2 \\ \frac{d}{dt}\frac{\partial L}{\partial \dot{q}_1} &= (m_1 + m_2) l_1^2 \ddot{q}_1^2 + m_2 l_2^2 (\ddot{q}_1 + \ddot{q}_2) \\ &\quad + m_2 l_1 l_2 (2\ddot{q}_1 + \ddot{q}_2) c_2 - m_2 l_1 l_2 (2\dot{q}_1 + \dot{q}_2) \dot{q}_2 s_2 \\ \frac{\partial L}{\partial q_2} &= -m_2 l_1 l_2 (\dot{q}_1^2 + \dot{q}_1 \dot{q}_2) s_2 - m_2 g l_2 c_{12} \\ \frac{\partial L}{\partial \dot{q}_2} &= m_2 l_2^2 (\dot{q}_1 + \dot{q}_2) + m_2 l_1 l_2 \dot{q}_1 c_2 \\ \frac{d}{dt}\frac{\partial L}{\partial \dot{q}_2} &= m_2 l_2^2 (\ddot{q}_1 + \ddot{q}_2) + m_2 l_1 l_2 \ddot{q}_1 c_2 - m_2 l_1 l_2 \dot{q}_1 \dot{q}_2 s_2 \end{aligned} \tag{12}$$

Based on the above derivation, the control torque can be obtained as:

$$\begin{aligned} \tau_1 &= [(m_1 + m_2) l_1^2 + m_2 l_2^2 + 2 m_2 l_1 l_2 c_2] \ddot{q}_1 \\ &\quad + [m_2 l_2^2 + m_2 l_1 l_2 c_2] \ddot{q}_2 - m_2 l_1 l_2 (2\dot{q}_1 + \dot{q}_2) \dot{q}_2 s_2 \\ &\quad + (m_1 + m_2) g l_1 c_1 + m_2 g l_2 c_{12} \\ \tau_2 &= [m_2 l_2^2 + m_2 l_1 l_2 c_2] \ddot{q}_1 + m_2 l_2^2 \ddot{q}_2 \\ &\quad + m_2 l_1 l_2 \dot{q}_1^2 s_2 + m_2 g l_2 c_{12} \end{aligned} \tag{13}$$

Assuming frictional force as $F = \alpha \mathrm{sgn}(\dot{q})$ and unknown disturbance as τ_d, the resulting standard equation is:

$$M(q)\begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + C(q, \dot{q})\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} + G(q) + F(\dot{q}) + \tau_d = \tau \tag{14}$$

Assuming $p_1 = (m_1 + m_2) l_1^2$, $p_2 = m_2 l_2^2$, $p_3 = m_2 l_1 l_2$, $p_4 = (m_1 + m_2) l_1$, $p_5 = m_2 l_2$, we can obtain:

$$M(q) = \begin{bmatrix} p_1 + p_2 + 2p_3c_2 & p_2 + p_3c_2 \\ p_2 + p_3c_2 & p_2 \end{bmatrix}$$
$$C(q, \dot{q}) = \begin{bmatrix} -p_3s_2\dot{q} & -p_3(\dot{q}_1 + \dot{q}_2)s_2 \\ p_3\dot{q}_1s_2 & 0 \end{bmatrix}$$
$$G(q) = \begin{bmatrix} p_4gc_1 + p_5gc_{12} \\ p_5gc_{12} \end{bmatrix} \tag{15}$$

B. ROBOTIC ARM CONTROLLER DESIGN

The RBF neural network is commonly used to approximate nonlinear functions. It combines the concepts of traditional neural networks and radial basis functions, offering strong approximation capabilities and rapid training. This network configuration comprises three layers: input layer, hidden layer, and output layer. The prescribed algorithm for the optimal RBF network is as delineated below:

$$y = W^T h(x)$$
$$h_i = g(||x - c_i||^2/b_i^2), \quad i = 1, 2, \cdots, n \tag{16}$$

where x represents the network input signal, y is the network output signal, W stands for the weights of the neural network, and c_i and b_i are parameters of the Gaussian basis function, $h = [h_1, h_2, \ldots, h_n]^T$.

$$r = \dot{e} + \beta e \tag{17}$$

where $e(t) = q_d(t) - q(t)$, $q_d(t)$ and $q(t)$ represent the desired and actual positions respectively, $\beta = \beta^T > 0$. To approximate the unknown term f in the RBF network model, setting:

$$f(x) = W^T h(x) + \varepsilon \tag{18}$$

Hence, the output of the neural network is:

$$\hat{f}(x) = \hat{W}^T h(x)\hat{W} \tag{19}$$

where $\hat{W}$ represents the estimated value of the weights.

Taking the input of the RBF neural network as:

$$x = \begin{bmatrix} e^T & \dot{e}^T & q_d^T & \dot{q}_d^T & \ddot{q}_d^T \end{bmatrix} \tag{20}$$

where ε denotes the neural network's approximation error.

Setting $\tilde{W} = W - \hat{W}$, we have $f(x) - \hat{f}(x) = \tilde{W}^T h(x) + \varepsilon$.

Designing the control law for this paper as:

$$\tau = \hat{W}^T h(x) + K_v r - u \tag{21}$$

Setting:

$$\begin{aligned} M\dot{r} &= M(\ddot{e} + \beta\dot{e}) \\ &= M(\ddot{q}_d + \beta\dot{e}) - Cr + C(\dot{q}_d + \beta e) + G + F + \tau_d - \tau \\ &= -Cr - \tau + f + \tau_d \end{aligned} \tag{22}$$

where $f(x) = M(\ddot{q}_d + \beta\dot{e}) + C(\dot{q}_d + \beta e) + G + F$.

Substituting Equation (21) into Equation (22), we obtain:

$$\begin{aligned} M\dot{r} &= -(K_v + C)r + \tilde{W}^T h(x) + (\varepsilon + \tau_d) + u \\ &= -(K_v + C)r + \varsigma_1 \end{aligned} \tag{23}$$

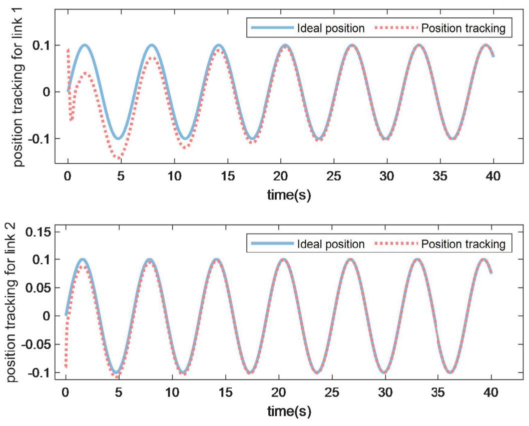


FIGURE 6. Position tracking of the two joints under RBF network control.

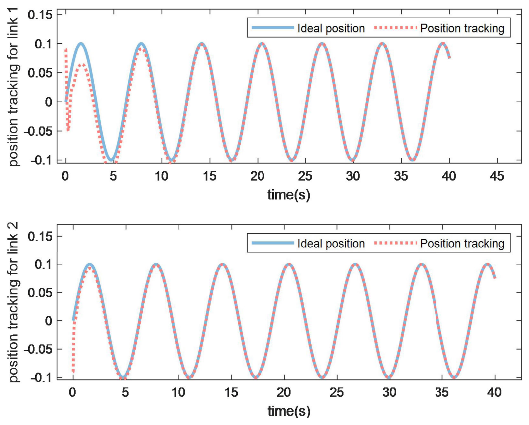


FIGURE 7. Position tracking under RBF network adaptive SMC.

u is used to overcome the robust terms of ε and τ_d, $\varsigma_1 = \tilde{W}^T h(x) + (\varepsilon + \tau_d) + u$.

Taking the robust term as $u = -(\varepsilon_N + b_d)\mathrm{sgn}(r)$ and the neural network adaptation rate as $\dot{\hat{W}} = Fhr^T$.

Let the Lyapunov function be:

$$L = \frac{1}{2}r^T Mr + \frac{1}{2}\mathrm{tr}(\tilde{W}^T F^- \tilde{W}) \tag{24}$$

By taking the derivative of equation (24), we can obtain:

$$\dot{L} = r^T M\dot{r} + \frac{1}{2}r^T\dot{M}r + \mathrm{tr}(\tilde{W}^T F^- \dot{\tilde{W}}) \tag{25}$$

Substituting Equation (23) into Equation (25), we obtain:

$$\begin{aligned} \dot{L} &= -r^T K_v r + \frac{1}{2}r^T(\dot{M} - 2C)r \\ &\quad + \mathrm{tr}\tilde{W}^T(F^- \dot{\tilde{W}} + hr^T) + r^T(\varepsilon + \tau_d + u) \\ &= -r^T K_v + r^T(\varepsilon + \tau_d + u) \end{aligned} \tag{26}$$

Because:

$$\begin{aligned} r^T(\varepsilon + \tau_d + u) &= r^T(\varepsilon + \tau_d) + r^T u = r^T(\varepsilon + \tau_d) \\ &\quad - ||r||(\varepsilon_N + b_d) \leq 0 \end{aligned}$$

We have:

$$\dot{L} \leq -r^T K_v r \leq 0$$

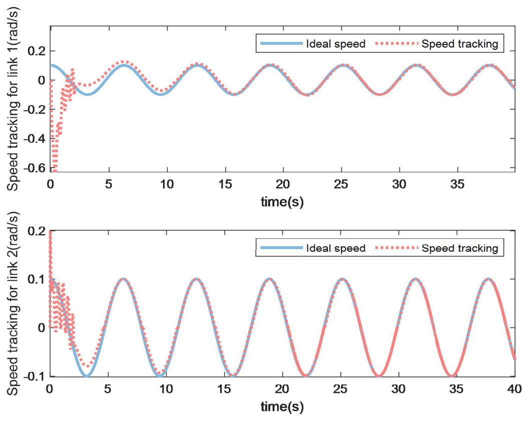


FIGURE 8. Velocity tracking of the two joints under RBF network control.

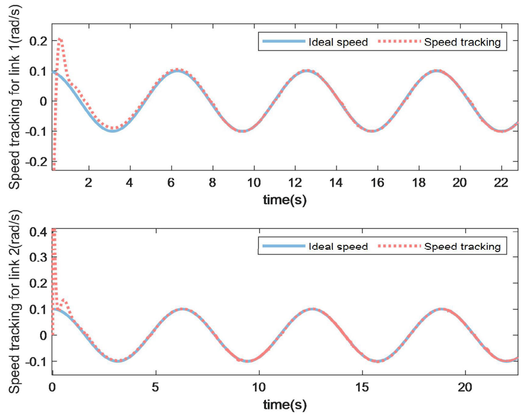


FIGURE 9. Velocity tracking under RBF network SMC.

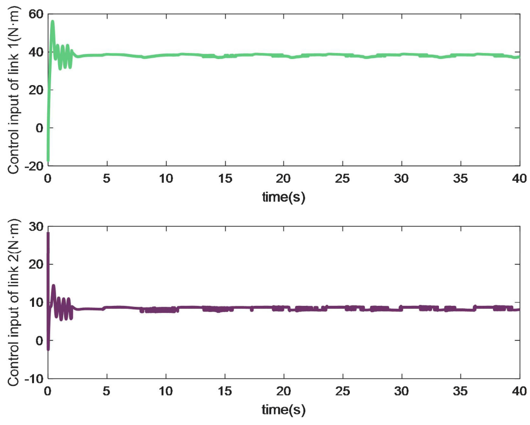


FIGURE 10. Control torque of two joints under RBF network.

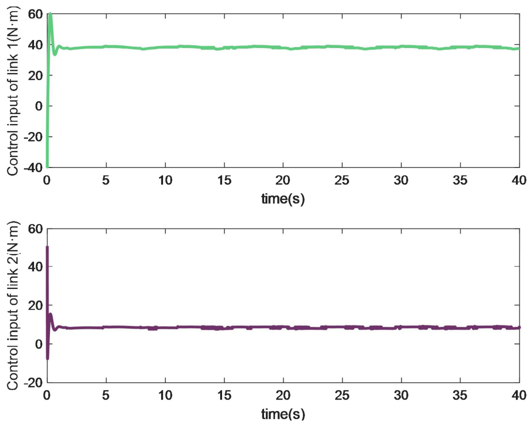


FIGURE 11. Control torque under RBF network with SMC.

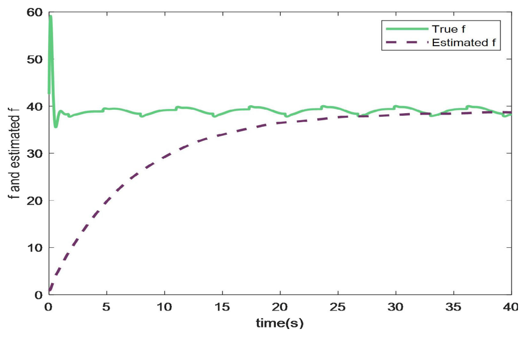


FIGURE 12. Uncertainty in RBF network.

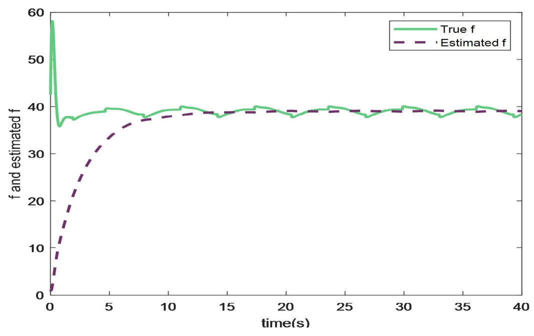


FIGURE 13. Uncertainty in RBF network with SMC.

When $\dot{L} \equiv 0$, $r \equiv 0$, applying LaSalle's invariance principle leads to the deduction that, in this scenario, the closed-loop system demonstrates asymptotic stability. When $t \to \infty$, $r \to 0$.

C. TRAJECTORY TRACKING EXPERIMENTAL ANALYSIS

The values of RBF neural network parameters directly affect the control effect of the neural network. Improper parameter selection may result in ineffective mapping by Gaussian basis functions. Assuming:

$$p = [p_1, p_2, p_3, p_4, p_5] = [2.8, 0.86, 0.75, 3.10, 0.87],$$
$$x = \begin{bmatrix} e^T & \dot{e}^T & q_d^T & \dot{q}_d^T & \ddot{q}_d^T \end{bmatrix}, b_i = 0.2, i = 1, \cdots, 7,$$

$$c = \begin{bmatrix} -1.5 & -1 & -0.5 & 0 & 0.5 & 1 & 1.5 \\ -1.5 & -1 & -0.5 & 0 & 0.5 & 1 & 1.5 \\ -1.5 & -1 & -0.5 & 0 & 0.5 & 1 & 1.5 \\ -1.5 & -1 & -0.5 & 0 & 0.5 & 1 & 1.5 \\ -1.5 & -1 & -0.5 & 0 & 0.5 & 1 & 1.5 \end{bmatrix}$$

The neural network's initial weights are initialized to zero, while the initial state of the system is [0.09, 0, −0.09, 0], with the angles of the two joints as $q_{1d} = 0.1 \sin t$, $q_{2d} = 0.1 \sin t$, $\alpha = 0.02$, the control parameters $K_v = \text{diag}[50,50]$, $F = \text{diag}[25,25]$, $\beta = \text{diag}[5,5]$. Assuming $\varepsilon_N = 0.2$, $b_d = 0.1$.

To emphasize the approximation effect of sliding mode control, only the RBF neural network is used for controlling the robotic arm, and it is compared with the method proposed

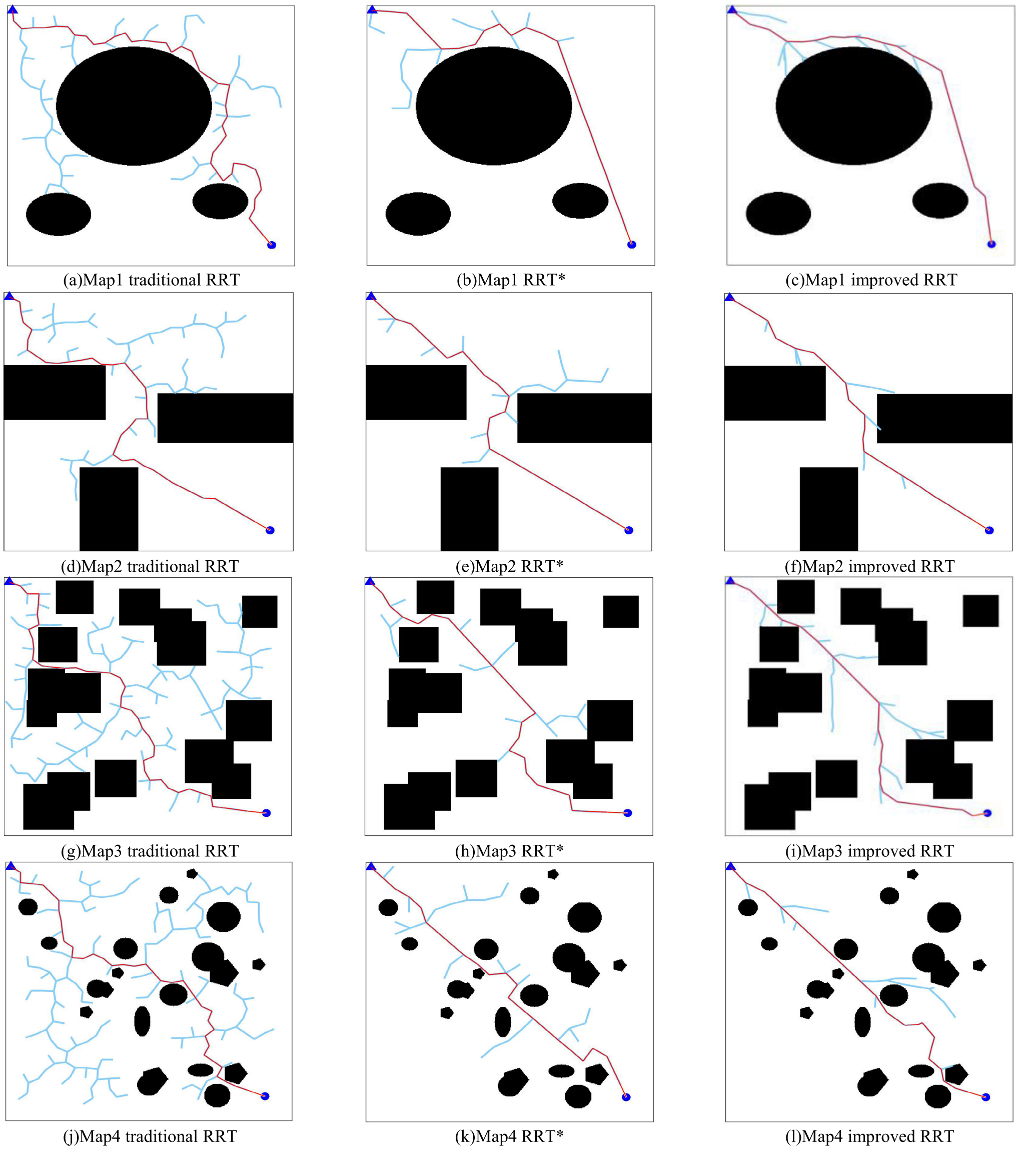


FIGURE 14. **Comparison of RRT algorithms in two-dimensional environments.**

in this paper. The tracking plots of the two joint positions of the robotic arm are shown in Figures 6 and 7.

By comparing Figures 6 and 7, it can be observed that the standalone neural network can still effectively approximate the position signals of the two joints, but it also has the drawback of slower convergence speed. In Figure 6, it took 10 seconds to approximate the ideal trajectory, while in Figure 7, the ideal control effect was achieved after 5 seconds. This indicates that the proposed control method in this paper not only ensures the approximation effect but also improves the convergence speed. The velocity tracking plots of the robotic arm are shown in Figures 8 and 9.

By comparing Figures 8 and 9, it can be observed that without the addition of sliding mode control, the two joints of the robotic arm achieve ideal velocity tracking only after 10 seconds, and there is significant oscillation present. However, with the inclusion of sliding mode control, the tracking speed improves significantly, and the oscillation phenomenon disappears, leading to a noticeable enhancement in stability and accuracy.

The output status of the robotic arm's control torque is shown in Figures 10 and 11.

The comparison between the two graphs reveals that our control approach exhibits better control performance. In comparison to conventional neural network control, our method introduces sliding mode control, which eliminates the presence of oscillations in the control system, thereby enhancing system stability and precision.

To validate the accuracy of uncertainty estimation in RBF network adaptive sliding mode control, simulation results are obtained as depicted in Figure 12 and Figure 13.

The comparison of the two graphs indicates that the RBF network adaptive sliding mode control achieves a faster and

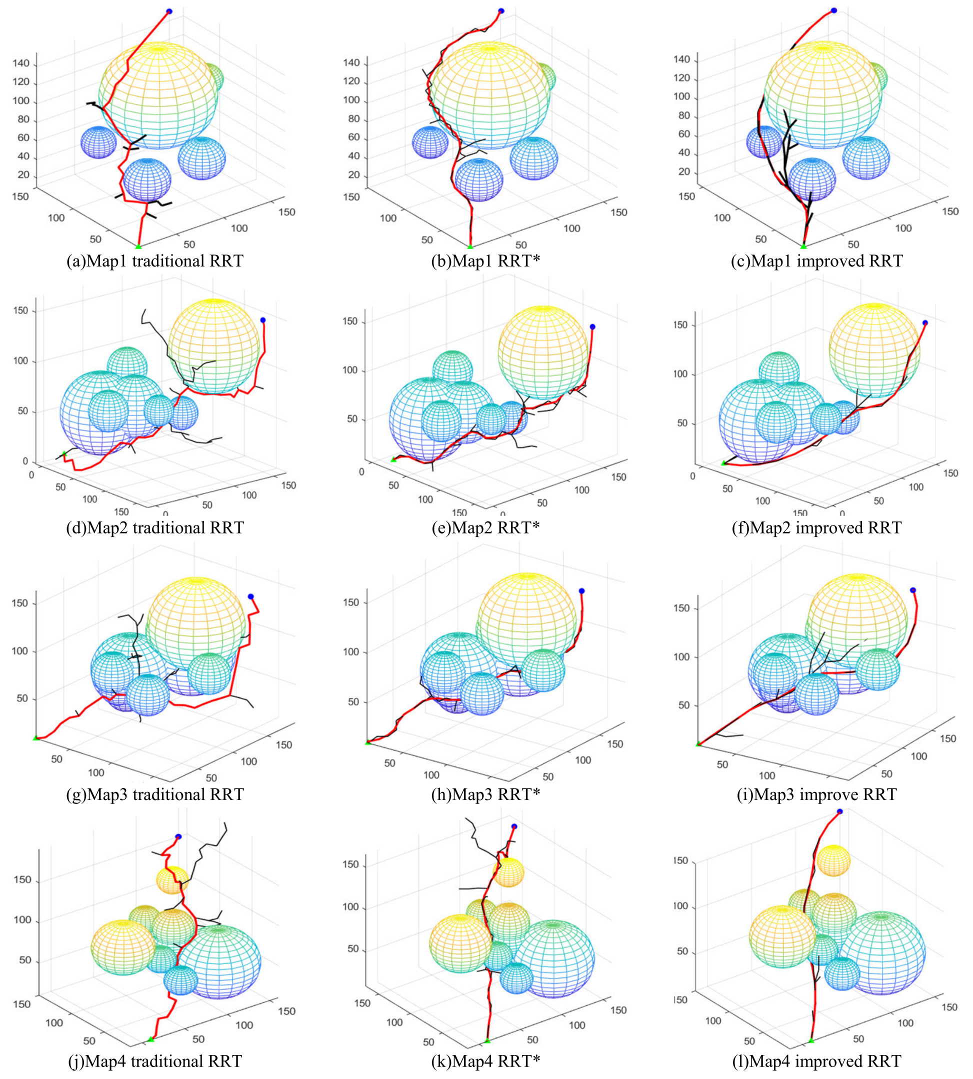


FIGURE 15. Comparison of RRT algorithms in three-dimensional environments.

more accurate approximation of uncertainties compared to standalone neural network control.

IV. SIMULATION VERIFICATION

To validate the effectiveness of the improved RRT algorithm, simulations were conducted on a computer equipped with an AMD Ryzen 7 PRO 6850HS processor and an integrated AMD graphics card using MATLAB R2022A. The proposed improved RRT algorithm was compared with the traditional RRT algorithm and the RRT* algorithm in both two-dimensional and three-dimensional environments.

A. PATH PLANNING IN TWO-DIMENSIONAL SPACE

In the context of two-dimensional simulations, the dimensions of the simulated map were set to 500×500. The algorithm was allowed a maximum of 10,000 iterations. The starting point was set at [10,10], and the destination point at [460,460]. To facilitate effective exploration by the algorithm, a step size of 20 was empirically determined through multiple experiments. To ensure the accuracy of the experiments, the algorithm was tested 30 times across different map environments, and the results were averaged. The path planning results of the improved RRT algorithm in various map scenarios are depicted in Figure 14.

Taking the average of 30 experimental runs, the algorithm comparison data is shown in table 1. In the two-dimensional map, the improved RRT algorithm proposed in this paper reduces the path search time by 46.1% compared to the traditional RRT algorithm and by 27.0% compared to the RRT* algorithm. Additionally, the path length is shortened by 20.9% and 10.6% respectively. Moreover, the number of search nodes is significantly lower than the latter two algorithms.

B. PATH PLANNING IN THREE-DIMENSIONAL SPACE

For the three-dimensional simulation environment, the map size is set to $150 \times 150 \times 150$. The maximum number of iterations is 10000, the starting point is [10,10,10], and

TABLE 1. Comparison of different RRT algorithms in two-dimensional maps.

Simulation Environment	Type	Time/s	Path Length	Number of Nodes
Map1	traditional RRT	0.895	755.1	227
	RRT*	0.654	650.2	150
	improved RRT	0.496	570.3	103
Map2	traditional RRT	1.224	800.7	163
	RRT*	0.843	730.5	105
	improved RRT	0.655	650.3	84
Map3	traditional RRT	1.950	884.5	181
	RRT*	1.510	760.1	140
	improved RRT	1.160	682.3	100
Map4	traditional RRT	1.791	820.2	177
	RRT*	1.362	750.1	110
	improved RRT	0.841	676.5	80

the destination point is [150,150,150]. To ensure better algorithmic performance, the search step length is set to 10 based on multiple experiments. In order to ensure the accuracy of the experiments, each algorithm is tested 30 times in different map environments, and the final results are averaged. The path planning results using the improved RRT algorithm in various map environments are shown in Figure 15.

TABLE 2. Comparison of different RRT algorithms in three-dimensional maps.

Simulation Environment	Type	Time/s	Path Length	Number of Nodes
Map1	traditional RRT	3.009	355.9	68
	RRT*	2.124	310.6	30
	improved RRT	1.748	282.1	22
Map2	traditional RRT	2.677	330.6	72
	RRT*	2.032	280.3	40
	improved RRT	1.442	261.2	24
Map3	traditional RRT	3.589	297.4	67
	RRT*	2.759	270.2	36
	improved RRT	2.312	256.8	23
Map4	traditional RRT	3.309	384.9	72
	RRT*	2.351	300.5	36
	improved RRT	1.864	258.7	25

Taking the average of 30 experimental runs, the comparative data for the algorithms are presented in table 2. In the three-dimensional environment, the improved RRT algorithm proposed in this paper demonstrated significant improvements compared to the traditional RRT algorithm and the RRT* algorithm. Specifically, the search time for the improved RRT algorithm was reduced by 41.8% and 20.9% when compared to the traditional RRT algorithm and the RRT* algorithm, respectively. Moreover, the path length was shortened by 22.0% and 8.7%, and the number of search nodes was notably fewer than the latter two algorithms.

The simulation results conducted in both two-dimensional and three-dimensional environments demonstrate that the improved RRT algorithm proposed in this paper exhibits significant enhancements in terms of search efficiency.

In addition, the application of machine learning in path planning can improve search efficiency, accuracy, and adaptability, enabling robots to plan paths more intelligently and adapt to different environments and task requirements.The applicability and scope of different methods are shown in Table 3.

TABLE 3. Comparison of different methods.

Method	Environmental adaptability	Applicable scope
Graph search algorithm	Moderate	simple path planning problems, sensitive to environmental changes
Machine learning	Excellent	complex path planning problems
RRT algorithm	Good	high-dimensional, continuous, non-convex complex spaces，capable of real-time path planning in dynamic environments
RRT* algorithm	Good	Applicable to the same scenarios as the RRT algorithm, but with higher path planning efficiency and quality
Improved algorithm proposed in this paper	Excellent	high-dimensional, complex path planning problems, with good adaptability to dynamic environments

V. CONCLUSION

In this study, a motion planning method for a two-degree-of-freedom robotic arm is investigated. An improved RRT algorithm is proposed to address the challenges posed by traditional RRT algorithms, such as high randomness, slow planning speed, non-smooth paths, and excessive corners. To validate the effectiveness of the algorithm, experimental simulations were conducted in both two-dimensional and three-dimensional environments, revealing the following advantages of the proposed approach:

(1) The utilization of probability sampling strategy accelerated the planning speed of the RRT algorithm, facilitating quicker growth of the tree structure toward the target point.

(2) The inclusion of repetitive greedy strategy reduced redundant nodes in the path, thereby enhancing search efficiency.

(3) The integration of artificial potential fields in conjunction with target bias strategy directed the RRT algorithm towards the direction of the target point, enhancing the effectiveness of path planning.

(4) The introduction of third-order B-spline curves for optimization led to smoother paths and improved precision in robotic arm tracking.

Compared to traditional RRT algorithms and RRT* algorithms, the improved RRT algorithm exhibited an increase in planning speed of 46.1% and 27.0%, 41.8% and 20.9% for two-dimensional and three-dimensional environments respectively. Path lengths were reduced by 20.9% and 10.6%, 22.0% and 8.7%, while the number of search nodes notably decreased. Moreover, considering the complexity of practical robotic arm operations and the uncertainty of system parameters, the paper introduced the RBF neural network to approximate uncertain terms. When combined with a sliding mode controller that adjusted parameters in real-time, the robotic arm's motion trajectory closely matched the desired trajectory.

The proposed motion planning method for the two-degree-of-freedom robotic arm provides an effective solution to the motion planning problem, as demonstrated through experimental verification. In the future, we can further explore the application of the algorithm in more complex scenarios and practical robotic systems. Additionally, we should continue refining the algorithm to enhance computational efficiency and optimize trajectory planning outcomes.

REFERENCES

[1] H. Ali, G. Xiong, H. Wu, B. Hu, Z. Shen, and H. Bai, "Multi-robot path planning and trajectory smoothing," in *Proc. IEEE 16th Int. Conf. Autom. Sci. Eng. (CASE)*, Aug. 2020, pp. 685–690.

[2] X. Cheng, J. Zhou, Z. Zhou, X. Zhao, J. Gao, and T. Qiao, "An improved RRT-connect path planning algorithm of robotic arm for automatic sampling of exhaust emission detection in Industry 4.0," *J. Ind. Inf. Integr.*, vol. 33, Jun. 2023, Art. no. 100436.

[3] Z. Jiang, S. E. Salcudean, and N. Navab, "Robotic ultrasound imaging: State-of-the-art and future perspectives," *Med. Image Anal.*, vol. 89, Oct. 2023, Art. no. 102878.

[4] C. I. Penaloza and S. Nishio, "BMI control of a third arm for multitasking," *Sci. Robot.*, vol. 3, no. 20, Jul. 2018, Art. no. eaat1228.

[5] S. N. Flesher, J. E. Downey, J. M. Weiss, C. L. Hughes, A. J. Herrera, E. C. Tyler-Kabara, M. L. Boninger, J. L. Collinger, and R. A. Gaunt, "A brain-computer interface that evokes tactile sensations improves robotic arm control," *Science*, vol. 372, no. 6544, pp. 831–836, May 2021.

[6] J. D. Gammell, T. D. Barfoot, and S. S. Srinivasa, "Informed sampling for asymptotically optimal path planning," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 966–984, Aug. 2018.

[7] B. Li and B. Chen, "An adaptive rapidly-exploring random tree," *IEEE/CAA J. Autom. Sinica*, vol. 9, no. 2, pp. 283–294, Feb. 2022.

[8] H. Zhang, Y. Wang, J. Zheng, and J. Yu, "Path planning of industrial robot based on improved RRT algorithm in complex environments," *IEEE Access*, vol. 6, pp. 53296–53306, 2018.

[9] Z. Gu, Y. Liu, W. Sun, G. Yue, and S. Sun, "UAV dynamic route planning algorithm based on RRT," *Comput. Sci.*, vol. 50, pp. 65–69, Jun. 2023.

[10] C. Yuan, W. Zhang, G. Liu, X. Pan, and X. Liu, "A heuristic rapidly-exploring random trees method for manipulator motion planning," *IEEE Access*, vol. 8, pp. 900–910, 2020.

[11] R. Mashayekhi, M. Y. I. Idris, M. H. Anisi, and I. Ahmedy, "Hybrid RRT: A semi-dual-tree RRT-based motion planner," *IEEE Access*, vol. 8, pp. 18658–18668, 2020, doi: 10.1109/ACCESS.2020.2968471.

[12] M. A. R. Pohan, B. R. Trilaksono, S. P. Santosa, and A. S. Rohman, "Path planning algorithm using the hybridization of the rapidly-exploring random tree and ant colony systems," *IEEE Access*, vol. 9, pp. 153599–153615, 2021, doi: 10.1109/access.2021.3127635.

[13] J. Zhang, Y. An, J. Cao, S. Ouyang, and L. Wang, "UAV trajectory planning for complex open storage environments based on an improved RRT algorithm," *IEEE Access*, vol. 11, pp. 23189–23204, 2023.

[14] J. Xu and K.-S. Park, "A real-time path planning algorithm for cable-driven parallel robots in dynamic environment based on artificial potential guided RRT," *Microsyst. Technol.*, vol. 26, no. 11, pp. 3533–3546, Jul. 2020.

[15] W. Jie, Z. Yudong, B. Yulong, H. H. Kim, and M. C. Lee, "Trajectory tracking control using fractional-order terminal sliding mode control with sliding perturbation observer for a 7-DOF robot manipulator," *IEEE/ASME Trans. Mechatronics*, vol. 25, no. 4, pp. 1886–1893, Aug. 2020.

[16] M. Azizkhani, I. S. Godage, and Y. Chen, "Dynamic control of soft robotic arm: A simulation study," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 3584–3591, Apr. 2022.

[17] B. Fan, Y. Zhang, Y. Chen, and L. Meng, "Intelligent vehicle lateral control based on radial basis function neural network sliding mode controller," *CAAI Trans. Intell. Technol.*, vol. 7, no. 3, pp. 455–468, Mar. 2022.

[18] D. J. Almakhles, "Robust backstepping sliding mode control for a quadrotor trajectory tracking application," *IEEE Access*, vol. 8, pp. 5515–5525, 2020.

[19] Z. Xu, Z. Liu, B. Jiang, and C. Gao, "Sliding mode fault-tolerant control of manipulator systems with actuator faults," *Control Eng. China*, vol. 30, Jul. 2023.

[20] A. Jouila and K. Nouri, "An adaptive robust nonsingular fast terminal sliding mode controller based on wavelet neural network for a 2-DOF robotic arm," *J. Franklin Inst.*, vol. 357, no. 18, pp. 13259–13282, Dec. 2020.

[21] M. Rahmani, H. Komijani, and M. H. Rahman, "New sliding mode control of 2-DOF robot manipulator based on extended grey wolf optimizer," *Int. J. Control, Autom. Syst.*, vol. 18, no. 6, pp. 1572–1580, Jan. 2020.

[22] C. Yin, S. Wang, X. Li, G. Yuan, and C. Jiang, "Trajectory tracking based on adaptive sliding mode control for agricultural tractor," *IEEE Access*, vol. 8, pp. 113021–113029, 2020.

[23] C. An, S. Jia, J. Zhou, and C. Wang, "Fast model-free learning for controlling a quadrotor UAV with designed error trajectory," *IEEE Access*, vol. 10, pp. 79669–79680, 2022.

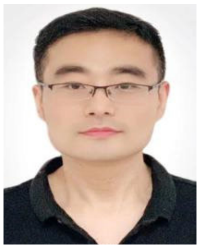
[24] J. Xian, L. Shen, J. Chen, and W. Feng, "Continuous sliding mode control of robotic manipulators based on time-varying disturbance estimation and compensation," *IEEE Access*, vol. 10, pp. 43473–43480, 2022.

[25] S. M. LaValle and J. J. Kuffner, "Rapidly-exploring random trees: Progress and prospects," *Algorithmic Comput. Robot., New Directions*, vol. 5, pp. 293–308, Jul. 2001.

[26] M. Liu, Q. Gu, B. Yang, Z. Yin, S. Liu, L. Yin, and W. Zheng, "Kinematics model optimization algorithm for six degrees of freedom parallel platform," *Appl. Sci.*, vol. 13, no. 5, p. 3082, Feb. 2023.

MENGLONG CAO was born in Shandong, China, in 1971. He received the Ph.D. degree in navigation guidance and control theory from the Harbin Institute of Technology, Harbin, China, in 2009. He is currently a Professor with the College of Automation and Electronic Engineering, Qingdao University of Science and Technology, China. He is also a Principal Member of the Institute of Autonomous Navigation and Intelligent Control, Qingdao University of Science and Technology. He is also the Vice Dean of the College of Automation and Electronic Engineering, Qingdao University of Science and Technology. His research interests include intelligent control, autonomous navigation, and information fusion.

XINJIE ZHOU was born in Shandong, China, in 1998. He is currently pursuing the master's degree in mechanical engineering with the Qingdao University of Science and Technology, Shandong. His main research interests include path planning and trajectory planning for robots.



YUNPENG JU received the M.S. degree in control theory and control engineering and the Ph.D. degree in mechanical design and theory from the Qingdao University of Science and Technology, in 2013 and 2016, respectively. He is currently a Master's Supervisor. His research interests include mechanical dynamics, robot control and navigation, and modeling and control of complex nonlinear systems.