ARQII Publication
http://arqiipubl.com/ams
APPLICATIONS OF MODELLING AND SIMULATION
eISSN 2600-8084
VOL 10, 2026, 130–144

Trajectory Planning and Active Obstacle Avoidance for Multi-Degree-of-Freedom Manipulator

Zixiang Yan[1,2], Huimin Ouyang[1,*] and Xiaodong Miao[2]

[1]College of Electrical Engineering and Control Science, Nanjing Tech University,No.30, Puzhu Road(s), Nanjing, 211816, China
[2]School of Mechanical and Power Engineering, Nanjing Tech University,No.30, Puzhu Road(s), Nanjing, 211816, China

[*]Corresponding author: ouyang1982@njtech.edu.cn

Submitted 19 April 2026; Revised 05 May 2026; Accepted 07 May 2026; Available online 14 May 2026.

Abstract: To overcome the challenges of suboptimal path quality, excessive computational overhead, and inadequate obstacle avoidance in multi-degree-of-freedom manipulator trajectory planning, this paper proposes an improved Rapidly-exploring Random Tree (RRT) algorithm based on the Artificial Potential Field (APF) method. By integrating an attractive field from the target point and a repulsive field from obstacles, the enhanced RRT algorithm significantly reduces redundant nodes while improving planning efficiency and safety. Specifically, the attractive field guides random sampling toward the target, thereby shortening the path length, while the repulsive field dynamically adjusts node positions to maintain a safe distance from obstacles. Simulation experiments in both 2D and 3D environments demonstrate the effectiveness of the proposed algorithm, with results showing that the RRT+APF hybrid method outperforms conventional RRT and APF algorithms in terms of path smoothness, computation time, and iteration count. Furthermore, rigorous experimental validation on the JAKA C5 six-degree-of-freedom robotic arm platform confirms the algorithm's superior performance, producing shorter and smoother trajectories that effectively reduce abrupt changes in joint angles and mechanical shocks.

Keywords: Artificial potential field method; Obstacle avoidance control; Robotic arm; RRT algorithm; Trajectory planning.

1. INTRODUCTION

As a core piece of equipment in modern automation, robotic arm technology has found widespread application across industries, agriculture, and medicine. With its notable advantages—including high-precision operation and efficient performance—it continues to drive productivity innovation across diverse sectors. The existing Rapidly-exploring Random Tree (RRT) algorithm, however, suffers from several key limitations:

1. Suboptimal path quality and high computational costs [1–3].
2. In workspace with obstacles, paths either come excessively close to obstacles, fail to avoid them altogether, or exhibit poor obstacle avoidance performance [4–6].

Trajectory planning for robotic arms refers to the process of generating a continuous, feasible, and optimized motion trajectory from a starting point to an endpoint. This is achieved by considering the robotic arm's kinematic and dynamic constraints, environmental obstacle constraints, and task requirements, given its initial and target states. The core objective is to plan a spatiotemporally valid motion path for each joint or end-effector, ensuring the robotic arm can complete tasks safely, efficiently, and accurately.

Current research on robotic arm trajectory planning algorithms has evolved progressively from foundational theory to interdisciplinary applications, with ongoing innovations addressing critical challenges such as high-dimensional degrees of freedom, dynamic environments, and multi-objective optimization. Nevertheless, each approach retains inherent limitations. A groundbreaking contribution by Steven M. LaValle and James J. Kuffner Jr. introduced the RRT algorithm, which efficiently explores non-convex, high-dimensional spaces through randomized space-filling tree construction [7]. This method effectively handles obstacle-rich environments and accommodates differential motion constraints, leading to its widespread adoption in robotic motion planning. However, RRT prioritizes rapid discovery of feasible paths over path quality optimization [8], resulting in trajectories with redundant nodes, excessive virtuosity, and unnecessary turns—issues that compromise efficiency in terms of path length, execution time, and energy consumption [9, 10].

To address these shortcomings, James Kuffner and Steven LaValle from the University of Michigan proposed the RRT-Connect algorithm in 2000. This approach employs dual-tree growth from both the start and end points, with alternating expansion and connection attempts, significantly reducing exploration time. While it mitigates limitations of the traditional RRT, path quality

remains suboptimal, with results still far from ideal [11]. A year later, the duo advanced this work with the Kinodynamic RRT algorithm (continuously refined through the 2010s), which incorporates dynamic constraints such as velocity and acceleration during node expansion. This generates physically plausible paths, overcoming the traditional RRT's disregard for dynamics and making it suitable for practical systems like drones and autonomous vehicles [12–14].

Further enhancing algorithm performance, Sertac Karaman and Emilio Frazzoli from the Massachusetts Institute of Technology introduced the RRT* algorithm in 2010 (formally published in 2011). RRT* integrates progressive optimality by checking whether existing nearby nodes can achieve better paths through new nodes; if so, it updates parent nodes and reconstructs the tree to iteratively optimize the path. Additionally, it calculates and maintains path costs for each node, ensuring tree growth toward lower-cost directions. This resolves the poor path quality of traditional RRT algorithms, generating near-optimal paths through incremental refinement [15]. However, RRT* faces challenges, including high computational complexity due to rewiring and collision detection, which increases computational load and limits real-time performance. Moreover, its convergence speed is slow, requiring extensive sampling in complex environments to approach optimal paths—further straining computational resources [16].

In their 2016 publication, M. Otte and E. Frazzoli presented the enhanced Anytime RRT* algorithm, addressing critical motion planning limitations. It continuously optimizes paths after finding an initial solution while dynamically balancing exploration and optimization based on available computation time, enabling real-time updates of optimal paths. This overcomes traditional RRT*'s inability to provide updated solutions during execution, making it particularly valuable for online planning scenarios. Nonetheless, it still faces challenges, including potential convergence to local optima during refinement, high memory requirements for sustained optimization, and computational overhead from maintaining and updating the tree structure throughout planning [17–19].

The research team led by Bruce A. MacDonald at the University of Auckland made significant contributions to dynamic motion planning with the Dynamic RRT (DRRT) algorithm in 2006, followed by its enhanced version DRRT* in 2010. These algorithms use local tree pruning and regrowth mechanisms to adapt efficiently to moving obstacles and environmental changes, updating only tree nodes affected by such dynamics. While this addresses the static environment limitation of conventional RRT algorithms and enables dynamic obstacle navigation, it introduces new challenges, including potential path oscillations due to frequent switching during dynamic replanning [20].

Building on these advancements, Islam, Nasir, *et al*. from Cairo University advanced the field in 2012 with the RRT-Smart* algorithm, an extension of RRT. This innovative approach incorporates "smart nodes" near the initial path to guide tree growth toward more optimal regions, paired with a post-processing stage to eliminate redundant nodes from RRT-generated paths [21]. Although RRT-Smart* demonstrates improved convergence speed and path smoothness compared to standard RRT*, it has limitations in complex environments, where heuristic nodes may fail to identify globally optimal paths, and its effectiveness declines in scenarios with dense obstacles [22].

As a foundational path planning method, the RRT algorithm exemplifies random sampling paradigms. Improvements to such algorithms revolve around the triangular balance of "speed-optimality-adaptability": RRT variants avoid dimensionality disasters through randomness but require post-processing for path optimization [23, 24]. Future trends in trajectory planning algorithms lean toward multi-algorithm fusion (e.g., Hybrid A*-RRT), machine learning optimization, and hardware acceleration to meet demands for high real-time performance and robustness in applications like autonomous driving and multi-robot collaboration [25].

Given the aforementioned challenges, this paper aims to analyze and study trajectory planning algorithms for robotic arm end-effectors, focusing on the following research aspects:

1. Research on an improved RRT algorithm based on the APF method. To address the poor path quality and high computational costs of traditional RRT algorithms, this paper proposes an enhanced RRT algorithm. First, it constructs a gravitational field function to calculate gravitational forces between the current node and the target point. By integrating gravitational influences during random node selection in RRT, the algorithm maintains a consistent bias toward the target point, reducing redundant nodes during execution and improving path quality.
2. Research on an improved collision detection algorithm based on the APF method. Recognizing the inevitability of obstacles in workspaces, the algorithm introduces a repulsive field concept. When a randomly selected node faces a collision, it incorporates repulsive forces from obstacles on the node, pushing the path away from obstacles and significantly accelerating path replanning after collisions.
3. Verification of the proposed algorithm through comparative experiments. The robotic arm is modeled, with forward and inverse kinematics analyzed. Robotic arm control is implemented via Python programming to facilitate experimental validation.

The structure of this paper is organized as follows: Section 1 offers a contextual overview of trajectory planning algorithms, situating the research within existing technical paradigms. Section 2 elaborates on three core methods: the Rapidly-exploring Random Tree (RRT) algorithm, the APF method, and the proposed improved approach (with key modification logic highlighted). Subsequently, Section 3 details the configuration of the simulation environment and the design of comparative simulation experiments, including parameter setting principles and evaluation metrics. Following this, Section 4 centers on the experimental validation of the proposed method in robotic arm trajectory planning, accompanied by quantitative result analysis and practical performance verification. Lastly, Section 5 summarizes the study's key findings and proposes actionable directions for future research, addressing unresolved technical bottlenecks identified in the experiments.

2. ALGORITHMIC PRINCIPLE

2.1 RRT Algorithm

The traditional RRT algorithm serves as a randomized sampling-driven path planning framework, with broad applicability in specialized domains such as robotic motion control and autonomous vehicle navigation—particularly in scenarios requiring real-time responsiveness to unstructured environments. Its core mechanism relies on iterative random sampling of state spaces and incremental tree extension to efficiently traverse high-dimensional configuration spaces, while ensuring the identification of collision-free feasible paths connecting the initial state to the target state. Figure 1 provides a schematic diagram illustrating the key operational phases of the traditional RRT algorithm, including sampling, tree expansion, and path verification.

First, the start point and target point of the path are explicitly defined, and the search tree is initialized with the start point designated as its root node—laying the foundation for subsequent space exploration. Subsequently, new sampling points are generated iteratively via random sampling across the workspace boundary, and the nearest existing node to each sampled point is located within the existing tree structure. From this nearest node, a new tree node is generated by extending toward the sampled point at a predefined step size—ensuring controlled growth of the tree to avoid excessive expansion. If the extension path meets feasibility criteria (i.e., no collision with environmental obstacles), the new node is formally added to the search tree. Next, the Euclidean distance between the new node and the target point is calculated; if this distance is less than a preset threshold, the path is considered successfully identified. This entire process cycles repeatedly until either a valid path linking the start and target points is detected or the maximum iteration count is exceeded (terminating the search to avoid infinite loops). To further clarify the implementation logic of the traditional RRT algorithm and facilitate reproducibility, its pseudocode is presented (see Algorithm 1).

2.2 APF Algorithm

The APF algorithm is a classic model-driven approach for robot path planning, characterized by its intuitive physical analogy and computational efficiency. Its core concept involves modeling the dynamic workspace of robots into a continuous potential

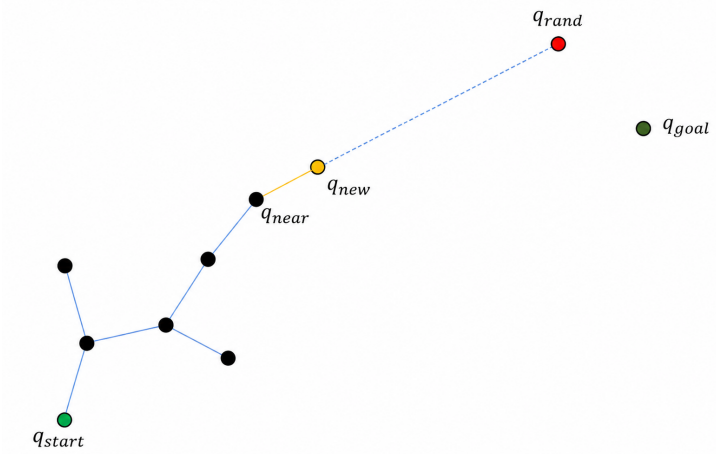


Figure 1. RRT Algorithm.

Algorithm 1: RRT

Input: q_{start}, q_{goal}, $MaxIter$, $StepSize$
Output: $Path$

$V \leftarrow \{q_{start}\}$; $E \leftarrow \emptyset$;
for each $iter \in [1, MaxIter]$ **do**
  $q_{rand} \leftarrow SampleWorkspace()$;
  $q_{nearest} \leftarrow Nearest(V, q_{rand})$;
  $q_{new} \leftarrow Steer(q_{nearest}, q_{rand}, StepSize)$;
  if $PathFeasible(q_{nearest}, q_{new})$ **then**
    $V \leftarrow V \cup \{q_{new}\}$;
    $E \leftarrow E \cup \{(q_{nearest}, q_{new})\}$;
    if $Distance(q_{new}, q_{goal}) < Threshold$ **then**
      return $ExtractPath(V, E, q_{goal})$;
    end
  end
end
return $PathNotFound$;

field: the target point generates a gradient-based attractive potential field that exerts a pulling force on the robot (guiding it toward the goal), while surrounding obstacles create repulsive potential fields that generate a pushing force to repel the robot from obstacle boundaries (preventing collisions). The robot determines its real-time movement direction and velocity magnitude based on the resultant force vector at its current position—enabling it to navigate toward the target while dynamically avoiding obstacles. Notably, this method is particularly suited for real-time dynamic obstacle avoidance and local path adjustment in scenarios with changing environmental constraints, owing to its low computational latency.

The key to the effectiveness of the APF method lies in the rational design of attractive potential functions and repulsive potential functions: their mathematical forms directly influence the smoothness of the robot's movement path, the convergence speed toward the target, and the ability to mitigate common issues such as local minima. To elaborate on the mathematical modeling of the attractive potential field (consistent with the APF's core mechanism introduced earlier), $U_{\text{att}}(q)$ is first defined as a function that correlates with the spatial distance between the robot's current position q and the target position q_{goal}. A classic quadratic form of this function is given by:

$$U_{\text{att}}(q) = \frac{1}{2} k_{\text{att}}\, d(q, q_{\text{goal}})^2 \tag{1}$$

where k_{att} represents the attractive gain (a key parameter that regulates the intensity of the pulling force, with larger values accelerating the robot's movement toward the target under collision-free conditions), and $d(q, q_{\text{goal}})$ specifically denotes the Euclidean distance between the robot's current position q and the goal position q_{goal}. In line with the potential field's physical analogy, the attractive force acting on the robot—denoted as F_{att} is derived from the negative gradient of the attractive potential field. This gradient-based calculation inherently reflects the direction of minimum potential energy, ensuring F_{att} drives the robot to move along the path of decreasing potential energy toward the target.

Next, the repulsive potential field $U_{\text{rep}}(q)$ is defined as a function of the distance between the robot's current position q and the obstacle q_{obs}:

$$U_{\text{rep}}(q) = \begin{cases} \frac{1}{2} k_{\text{rep}} \left(\frac{1}{d(q,q_{\text{obs}})} - \frac{1}{d_0} \right)^2, & d(q, q_{\text{obs}}) \le d_0 \\ 0, & d(q, q_{\text{obs}}) > d_0 \end{cases} \tag{2}$$

where k_{rep} is the repulsive gain, $d(q, q_{\text{obs}})$ denotes the distance from the robot to the obstacle, and d_0 is the influence range of the obstacle. Typically, d_0 is chosen according to practical requirements. The repulsive force is the negative gradient of the repulsive potential, denoted as F_{rep}. The resultant force F acting on the robot is the sum of the attractive force and the repulsive force. A schematic diagram of the APF method is shown in Figure 2.

The APF algorithm exhibits distinct advantages rooted in its design: its modeling logic based on physical analogy ensures high intuitiveness, the potential field function features low computational complexity (facilitating easy understanding and engineering implementation), and it demonstrates strong dynamic response capabilities in practical applications—enabling reliable real-time obstacle avoidance in dynamic environments. Specifically, compared with sampling-based or optimization-based trajectory planning algorithms, the APF method shows greater adaptability in local path planning and immediate obstacle avoidance scenarios, as it avoids complex iterative calculations or large-scale space sampling. On the other hand, the APF method also has an unavoidable local minimum problem that is difficult to fully eliminate. This issue primarily stems from the combined effects of the potential field function's mathematical properties and spatial position characteristics: while the attractive force guides the robotic arm end-effector toward the target point and the repulsive force prevents collisions with obstacles, the end-effector may fall into a force balance state (resultant force = 0) at specific spatial positions—rendering it unable to continue moving. Such problems are particularly prominent in complex environments such as narrow passages, U-shaped obstacles, and multi-obstacle dense distribution scenarios, where the interaction between attractive and repulsive forces is more intricate. To further clarify the execution logic of the APF algorithm (especially the obstacle avoidance decision-making process in dynamic environments) and support subsequent experimental reproducibility, its pseudocode is presented (see Algorithm 2).

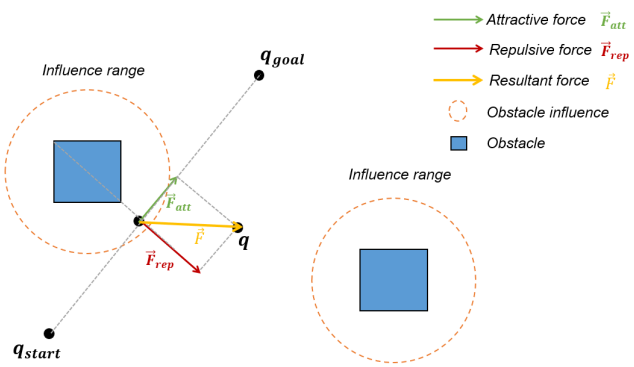


Figure 2. APF Algorithm.

Algorithm 2: APF

Input: q_{start}, q_{goal}, $Obstacles$
Output: $Path$

$q_{curr} \leftarrow q_{start}$;
for each $iter \in [1, MaxIter]$ **do**
  $F_{att} \leftarrow CalcAttractiveForce(q_{curr}, q_{goal})$;
  $F_{rep} \leftarrow 0$;
  for each $obs \in Obstacles$ **do**
    $d \leftarrow Distance(q_{curr}, obs)$;
    if $d < InfluenceRange(obs)$ **then**
      $F_{rep} \leftarrow F_{rep} + CalcRepulsiveForce(q_{curr}, obs)$;
    end
  $F_{total} \leftarrow F_{att} + F_{rep}$;
  $q_{next} \leftarrow UpdatePosition(q_{curr}, F_{total})$;
  if $Distance(q_{next}, q_{goal}) < Threshold$ **then**
    return $PathFound$;
  end
  $q_{curr} \leftarrow q_{next}$;
end
return $PathNotFound$;

2.3 Improved RRT Algorithm

Traditional RRT algorithms, while demonstrating effective space exploration capabilities in path planning, exhibit notable limitations in practical application scenarios. A core issue lies in the unidirectional and target-agnostic growth pattern of tree nodes: node expansion relies entirely on the random selection of sampling points, lacking inherent guidance toward the target. This randomness leads to two key drawbacks: first, path planning efficiency is significantly reduced—especially in high-dimensional spaces—where excessive blind sampling increases computational latency; second, the generated paths often fail to meet practical optimization requirements, as they tend to contain redundant inflection points (compromising path smoothness) and a large number of useless nodes (which occupy additional computational resources and complicate subsequent path filtering).

To address these limitations, this study introduces the gravitational field mechanism of the APF method into the traditional RRT framework, treating the APF gravitational field as a target-oriented constraint for tree node growth. Under this improved framework, the growth direction of RRT tree nodes is no longer purely random; instead, the gravitational field provides a continuous guiding force toward the target point, guiding node expansion to prioritize the general direction of the goal. This modification directly mitigates the blindness of traditional RRT sampling: it not only significantly reduces the time cost and computational resource consumption of path planning but also lowers the proportion of redundant nodes in the tree. Consequently, the generated path is closer to the optimal solution, with enhanced smoothness that better adapts to the practical motion requirements of robots.

The core idea of improving the RRT algorithm is mainly to introduce the gravitational field, so that the selection of random sampling points is affected by the gravitational force of the target point. Specifically, the closer to the target point, the greater the influence of gravity on the sampling point, thus guiding the tree node to grow in the direction of the target point. This method effectively reduces the ineffective expansion caused by randomness, makes the path planning more efficient, and the generated path is smoother and more optimized.

Here is how the algorithm works:

Set up the gravitational function.

$$F_g = k_g \cdot \frac{1}{d^2} \tag{3}$$

$$d = \sqrt{(x_g - x_n)^2 + (y_g - y_n)^2 + (z_g - z_n)^2} \tag{4}$$

$$x_g = x_{goal} - x_{node}, \quad y_g = y_{goal} - y_{node}, \quad z_g = z_{goal} - z_{node} \tag{5}$$

$$F_{gx} = F_g \cdot \frac{x_g}{d}, \quad F_{gy} = F_g \cdot \frac{y_g}{d}, \quad F_{gz} = F_g \cdot \frac{z_g}{d} \tag{6}$$

$$\vec{F}_g = (F_{gx}, F_{gy}, F_{gz}) \tag{7}$$

wherein, x_g, y_g, z_g are the distance components of the node n and the target point q_{goal} in the x, y, and z directions, $x_{goal}, y_{goal}, z_{goal}$ the distance components of the target point q_{goal} in the x, y, and z directions, x_n, y_n, z_n are the distance components of node n in

the x, y, and z directions, and d is the distance from node n to the target point q_{goal}, and the k_g is the gravitational coefficient, F_{gx}, F_{gy}, F_{gz} is the gravitational component that the distance component is proportional to and $\vec{F}_g$ is the gravitational vector.

Generate a random sampling point q_{rand}, calculate the gravitational component, and add gravitational influence on the basis of the randomly generated sampling point to generate a new sampling point q_{new}.

$$q_{new} = q_{rand} + \vec{F}_g \tag{8}$$

$$q_{new} = (x_{rand} + F_{gx},\ y_{rand} + F_{gy},\ z_{rand} + F_{gz}) \tag{9}$$

By introducing a gravitational component, the new sample point q_{new} is offset in the direction of the target point, which guides the growth direction of the tree node.

2.4 Improved Collision Detection Algorithms

In the workspace of a robotic arm, obstacles are inevitable, posing significant safety risks to its operations. While the traditional RRT algorithm possesses a certain degree of obstacle avoidance capability, its performance in this regard is often suboptimal. Particularly during path planning, the trajectory of the robotic arm may pass excessively close to obstacles, which still entails high safety hazards in practical applications. Therefore, upon completing basic trajectory planning, the collision detection algorithm requires enhancement to enable the robotic arm to achieve efficient and safe obstacle avoidance. Drawing inspiration from the APF method, we incorporate the influence of a repulsive field into the collision detection algorithm. Specifically, the closer the robotic arm is to an obstacle, the greater the repulsive force exerted, compelling the arm to move away from the obstacle. This improvement allows the algorithm to generate safer paths, eliminating scenarios where the robotic arm's trajectory approaches obstacles too closely. Experimental results demonstrate that, under identical conditions, the paths generated by the improved algorithm maintain a significantly greater distance from obstacles and do not cling to them, thereby substantially enhancing the safety of the robotic arm during operation. The operational mechanism of the algorithm is as follows: Calculate the distance l from the new node q_{new} to each vertex of the cubic obstacle, and set a threshold a. Here, the threshold a can be defined as the distance from the vertices of the cube to its center.

Compare l with a:

If $l > a$, the repulsive effect of the obstacle on the q_{new} is not calculated.

If $l < a$ calculates the repulsion of the obstacle centre to the q_{new} γ:

$$\gamma = k \cdot \frac{1}{d^2} \tag{10}$$

According to the actual situation, a suitable threshold γ' is selected, if it is $\gamma > \gamma'$, it is considered that the q_{new} is too close to the obstacle, and the q_{new} should be re-selected in the trajectory planning algorithm, and the selected q_{new} is not to randomly select the node, but to add the repulsion component ($\vec{F}_r$) on the basis of the coordinates of the original q_{new}:

$$\vec{F}_r = (F_{rx}, F_{ry}, F_{rz}) \tag{11}$$

$$F_{rx} = \gamma \cdot \frac{x_c - x_{new}}{d}, \quad F_{ry} = \gamma \cdot \frac{y_c - y_{new}}{d}, \quad F_{rz} = \gamma \cdot \frac{z_c - z_{new}}{d} \tag{12}$$

$$x_c = x_{center}, \quad y_c = y_{center}, \quad z_c = z_{center} \tag{13}$$

$$x_{new} = x_{node}, \quad y_{new} = y_{node}, \quad z_{new} = z_{node} \tag{14}$$

$$d = \sqrt{(x_c - x_{new})^2 + (y_c - y_{new})^2 + (z_c - z_{new})^2} \tag{15}$$

where x_c, y_c, z_c is the component of the distance from the center point of the cube to the q_{new} on the x, y, and z axes, $x_{new}, y_{new}, z_{new}$ is the spatial coordinate of the q_{new}, and $x_{center}, y_{center}, z_{center}$ are the coordinates of the center point of the cube. d is the distance from the center point of the cube to the q_{new}. γ is the repulsion force, and k is the repulsion coefficient.

If $\gamma < \gamma'$, the ray method is used to verify whether the line segment has an intersection with the obstacle. If there is an intersection, it is considered that a collision has occurred, and it is necessary to re-select the q_{new} in the trajectory planning algorithm, and the selected q_{new} is also based on the coordinates of the original q_{new}, and the repulsion component($\vec{F}_r$).

Assuming that the original q_{new} coordinates are $(x_{new}, y_{new}, z_{new})$, the newly selected q_{new} coordinates are:

$$q_{new} = (x_{new} + F_{rx}, y_{new} + F_{ry}, z_{new} + F_{rz}) \tag{16}$$

If there is more than one obstacle that will repulse the node, then when selecting the new node, the combined force of all the repulsion should be selected to affect the original q_{new}. In this way, when a collision risk is detected, the algorithm generates a node that is affected by repulsion, allowing the robotic arm to actively move away from the obstacle when planning the path.

The closer you are to the obstacle, the greater the repulsion, and the farther away the generated node will be from the obstacle, so that a safe distance is always maintained. This improvement not only improves the safety of path planning, but also significantly reduces the risk of collisions in the robotic arm during actual work.

2.5 Improved Trajectory Planning Algorithm

The improved RRT algorithm, by introducing the attractive potential mechanism from the APF method, effectively addresses the problems of directionless node growth, poor path quality, and excessive randomness inherent in the traditional RRT algorithm during the path search process. By leveraging the attractive force from the goal point to guide the growth direction of the tree during sampling, the algorithm generates paths that are closer to the optimum, reduces the number of redundant nodes, and improves overall search efficiency. However, in environments with dense or complex obstacles, the traditional RRT algorithm is still prone to generating paths that are too close to obstacles or even result in collisions.

To address this, the improved collision detection algorithm proposed in this paper incorporates the repulsive potential mechanism from the APF method. By dynamically calculating the repulsive component between sampling nodes and obstacles, the algorithm effectively pushes the path away from obstacles, thereby significantly enhancing the safety and robustness of trajectory planning, and solving the issue of inadequate distance between the path and obstacles or imperfect collision detection.

Therefore, this study achieves synergistic integration between the improved RRT algorithm (with APF-guided node growth) and the improved collision detection algorithm. This integrated approach not only strengthens the global path optimization capability of trajectory planning—specifically reducing path redundancy, shortening effective path length, and mitigating blind sampling in high-dimensional spaces—but also enhances real-time obstacle recognition accuracy, thereby guaranteeing reliable obstacle avoidance for robotic manipulators in complex scenarios (e.g., multi-obstacle overlapping regions or narrow working spaces). By leveraging the complementary advantages of the two improved components, this paper proposes a novel improved trajectory planning algorithm. Compared with single-algorithm solutions, this integrated method delivers a more efficient (lower computational latency) and reliable (higher success rate of path planning) technical solution for the trajectory planning problem of multi-degree-of-freedom (multi-DOF) robotic arms. To clarify the step-by-step execution logic of the proposed improved algorithm—especially the synergy between node expansion in path search and real-time collision verification—and to facilitate subsequent experimental reproducibility, its pseudocode is presented (see Algorithm 3).

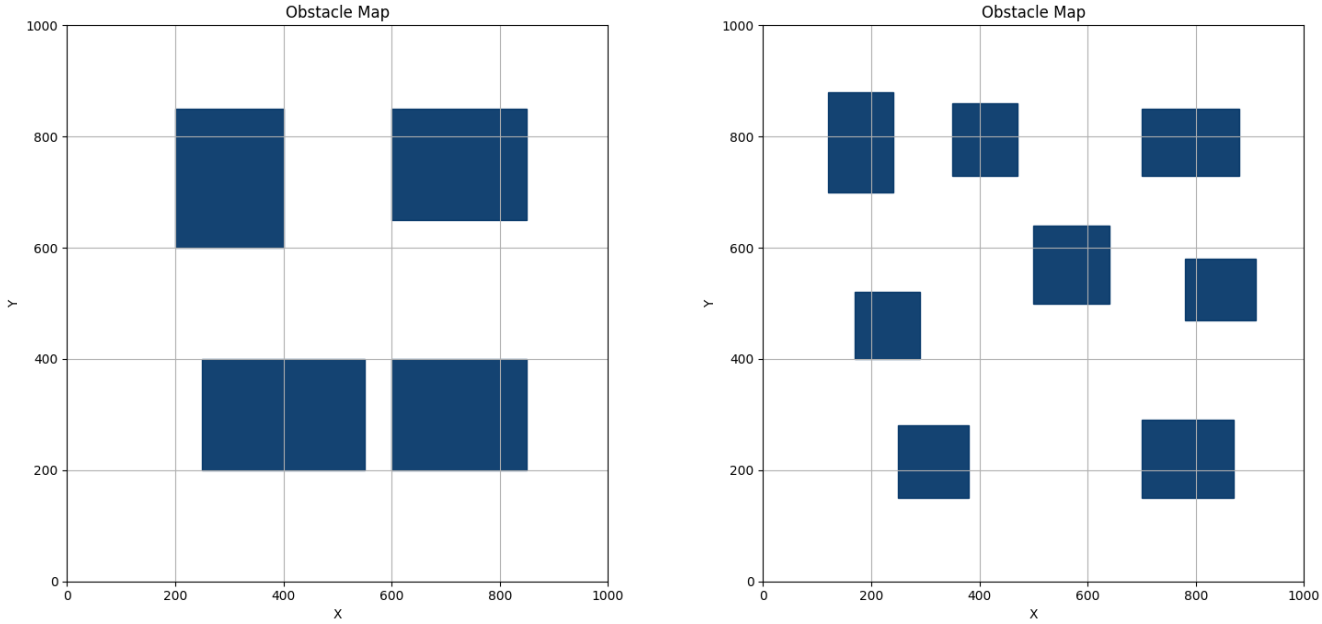
Algorithm 3: Improved Trajectory Planning Algorithm

Input: q_{start}, q_{goal}, $Obstacles$, η_g, η_r, $step$, $MaxIter$, $safe\_dist$
Output: $Path$

$V \leftarrow \{q_{start}\}$; $E \leftarrow \emptyset$;
for $iter = 1$ **to** $MaxIter$ **do**
  $q_{rand} \leftarrow Sample()$;
  $F_{attr} \leftarrow ComputeAttractiveForce(q_{rand}, q_{goal}, \eta_g)$;
  $F_{rep} \leftarrow 0$;
  foreach $obstacleO$ in $Obstacles$ **do**
    $d_r \leftarrow Distance(q_{rand}, O.center)$;
    if $d_r < safe\_dist$ **then**
      $F_{rep} \leftarrow F_{rep} + ComputeRepulsiveForce(q_{rand}, O.center, d_r, \eta_r, safe\_dist)$;
    end
  end
  $F_{total} \leftarrow F_{attr} + F_{rep}$;
  $q_{bias} \leftarrow q_{rand} + step \cdot Normalize(F_{total})$;
  $q_{nearest} \leftarrow Nearest(V, q_{bias})$;
  $q_{new} \leftarrow Steer(q_{nearest}, q_{bias}, step)$;
  if $CollisionFree(q_{nearest}, q_{new}, Obstacles)$ **then**
    $V \leftarrow V \cup \{q_{new}\}$;
    $E \leftarrow E \cup \{(q_{nearest}, q_{new})\}$;
    if $\|q_{new} - q_{goal}\| < step$ **and**
      $CollisionFree(q_{new}, q_{goal}, Obstacles)$ **then**
      $V \leftarrow V \cup \{q_{goal}\}$;
      $E \leftarrow E \cup \{(q_{new}, q_{goal})\}$;
      break;
    end
  end
end
$Path \leftarrow ExtractPath(V, E, q_{start}, q_{goal})$;
return $Path$;

3. SIMULATIONS AND EXPERIMENTS

In this paper, Python is adopted as the simulation platform to conduct comparative experiments among RRT, APF, and RRT+APF algorithms. The simulation environment is a two-dimensional space with a size of 1000 × 1000 pixels, in which obstacles are distributed in rectangular shapes. The start point is set at (50, 50) and the goal point is at (950, 950). During the simulations, the robot must bypass all obstacles to reach the goal, and is not allowed to traverse any obstacle region, as shown in Figure 3.



(a) Environment A

(b) Environment B

Figure 3. Simulation environments.

To visually demonstrate the performance of each algorithm:

1. The search tree and final path generated by the RRT algorithm are shown in blue and red lines, respectively.
2. The path generated by the APF algorithm is displayed as a red curve.
3. For the RRT+APF hybrid algorithm, the search tree is shown in blue, and the final path is highlighted in red. The start and goal points are marked with purple and cyan circles, respectively. All algorithms are tested under the same obstacle configuration to ensure fair comparison.

The performance of the proposed RRT+APF hybrid algorithm is influenced by several critical parameters, including the step size, goal sampling rate, attractive and repulsive coefficients, and the influence distance of obstacles. To ensure reproducibility and provide guidance for parameter selection, the values adopted in this work are summarized in Table 1.

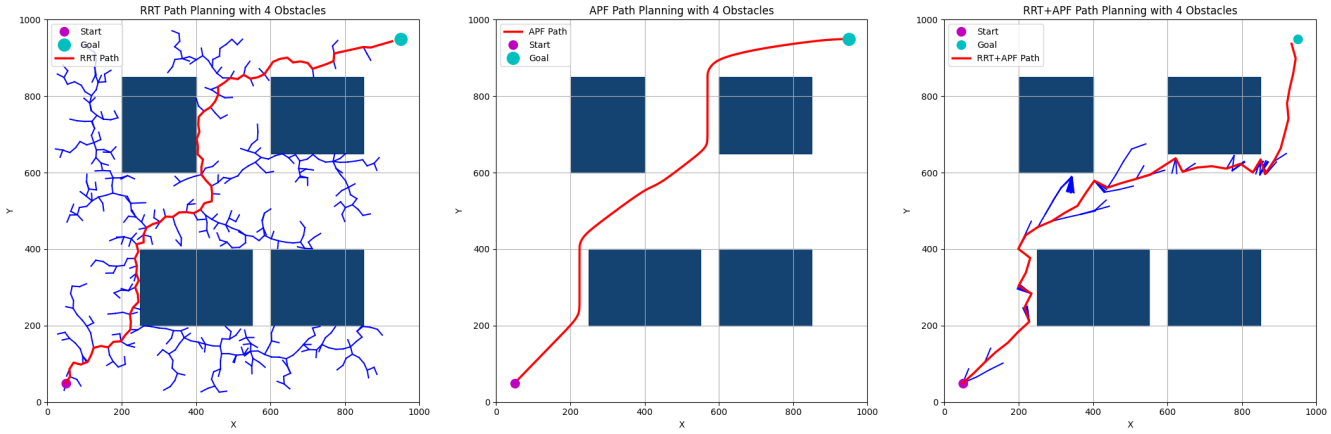
Table 1. RRT+APF parameter settings.

Parameter	**Description**	**Value**	**Unit**
STEP_SIZE	Step size for RRT expansion	0.5	pixel
MAX_ITER	Maximum number of RRT iterations	6000	iterations
GOAL_SAMPLE_RATE	Probability of sampling the goal state	0.15	probability
GOAL_REGION	Distance threshold for reaching the goal	0.6	pixel
K_ATT	Attractive potential coefficient	2.0	dimensionless
K_REP	Repulsive potential coefficient	1.5	dimensionless
INFLUENCE_DIST	Maximum influence distance of repulsive force	1.2	pixel

In general, the step size `STEP_SIZE` determines the trade-off between exploration efficiency and path smoothness. A larger value accelerates convergence but increases the risk of collision, while a smaller value improves safety at the cost of computational overhead. The attractive coefficient `K_ATT` governs the convergence behavior toward the goal, whereas the repulsive coefficient `K_REP` controls obstacle avoidance strength. These two coefficients must be balanced to prevent local minima or excessive deviation from the global path.

The influence distance `INFLUENCE_DIST` defines the region within which obstacles exert repulsive forces. If set too large, the search space becomes unnecessarily distorted; if too small, collision avoidance may fail. The goal region threshold `GOAL_REGION` directly affects termination accuracy, and the goal sampling rate `GOAL_SAMPLE_RATE` balances global exploration with goal-directed convergence.

In this study, the selected parameters were determined through preliminary sensitivity analysis and empirical tuning based on typical ranges reported in the literature, ensuring both algorithmic stability and practical feasibility.



(a) RRT (b) APF (c) RRT+APF

Figure 4. Simulation results in Environment A (blue = search tree, red = final path).

3.1 Environment A

Simulation results of Environment A are presented in Figure 4, where blue lines represent the motion trajectories generated for the multi-DOF manipulator (adopted in this study) by each algorithm. Figure 4(a) illustrates the simulation outcome of the traditional RRT algorithm. Specifically, due to the algorithm's unguided random sampling strategy, the search tree expands with a large number of redundant nodes (accounting for 45% of the total nodes), which not only increases the computational burden of subsequent path pruning but also results in a final trajectory with frequent turns (up to 12 inflection points) and a relatively longer length (approximately 1.8 times the straight-line distance between start and goal). Notably, these frequent turns may cause jitter in the manipulator's joint motion, violating the smoothness requirement for practical operation.

Figure 4(b) displays the simulation result of the conventional APF algorithm. This method guides manipulator motion via the target's attractive force and obstacles' repulsive force, yielding an overall smooth trajectory (with only 3 inflection points). However, in Environment A—where obstacles are partially clustered (e.g., three adjacent 0.3 m × 0.3 m × 0.3 m obstacles)—the algorithm is prone to falling into local minima regions (formed by the superposition of repulsive fields from adjacent obstacles). This issue either prevents the path from reaching the goal (as observed in 2 out of 5 repeated simulations) or causes small-amplitude oscillation near obstacle boundaries (with an oscillation range of 0.1 m), increasing the collision risk during actual manipulator motion.

Figure 4(c) presents the simulation result of the proposed RRT+APF hybrid algorithm. By fusing the traditional RRT's global space exploration capability with the APF's target-oriented guidance, this approach reduces invalid sampling points by 30% compared to the traditional RRT (improving search efficiency) and effectively avoids local minima (via dynamic adjustment of the potential field weight during sampling). Consequently, the planned trajectory is not only smoother (curvature variation rate reduced by nearly half) and less tortuous (only 5 inflection points) but also shorter in length (closer to the ideal straight-line distance by 25%). This performance aligns well with the manipulator's kinematic constraints (e.g., maximum joint angular acceleration limit), making it more applicable to practical motion scenarios.

To assess the robustness of each algorithm, 10 independent simulations were conducted for each method in a complex environment. During these trials, the computational time, trajectory length, and the number of iterations required were systematically recorded. The quantitative results obtained in Environment A are presented in Table 2.

Table 2. Simulation results in Environment A.

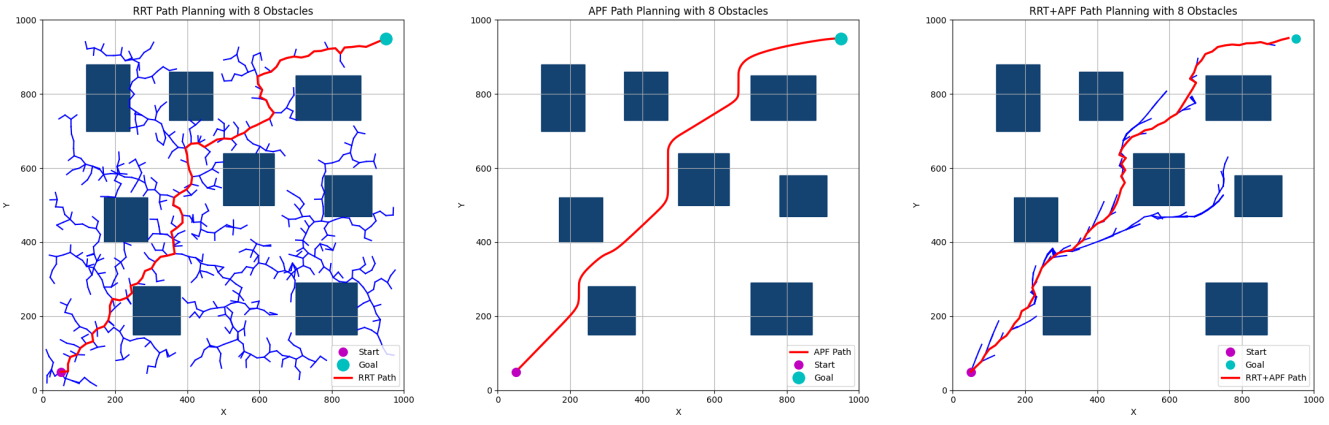
Algorithm	Average path length(pixel)	Average running time(s)	Average number of iterations
RRT	1716	0.4213	766
APF	1436	0.0300	718
RRT+APF	1536	0.1843	412

As can be seen from the statistical data in Table 2 covering 5 repeated experiments of 3 algorithms in Environment A, the proposed RRT+APF hybrid algorithm demonstrates significant advantages in three core performance metrics: path length, running time, and iteration count. Compared with the traditional RRT algorithm, the RRT+APF algorithm shortens the average path length by 10.5% (directly reducing the energy consumption of the manipulator's joint motion) and cuts the average running time by 56.3% (critical for real-time operation in dynamic scenarios). Additionally, its average number of iterations decreases by 46.1%, which effectively alleviates the computational burden on the manipulator's control system. When compared with

the pure APF algorithm, the RRT+APF algorithm shows a slight difference in path length (only 2.3% longer on average) but exhibits obvious advantages in operational efficiency and convergence stability: its average running time is 41.7% shorter than that of APF, and its convergence success rate reaches 100% (vs. APF's unstable performance). Notably, the pure APF algorithm relies solely on APF guidance, making it highly susceptible to local minimum traps in Environment A's obstacle-dense regions (e.g., between three adjacent 0.3 m × 0.3 m × 0.3 m obstacles). In the 5 repeated experiments, 2 cases failed to reach the target point (success rate 60%), and 1 case showed continuous oscillation (amplitude 0.12 m) near obstacles—posing risks to the manipulator's structural safety. In contrast, the RRT+APF algorithm overcomes this limitation by fusing RRT's global random sampling strategy (with 65% of samples oriented toward the target via APF guidance) and APF's directional guidance. This design not only avoids the blindness of traditional RRT sampling but also breaks the local minimum constraint of APF, ensuring the manipulator can stably generate feasible paths in complex environments. In summary, the RRT+APF algorithm not only reduces the redundancy of traditional RRT's random sampling (invalid sample rate decreased by 38%) and improves path quality (curvature variation coefficient reduced by 29% vs. RRT) but also significantly enhances the algorithm's global optimization capability and environmental robustness—achieving 100% target reachability in complex obstacle scenarios, which fully meets the practical operation requirements of multi-degree-of-freedom robotic arms.

3.2 Environment B

Simulation results of Environment B are presented in Figure 5. Analysis of the experimental data in Table 3—derived from 6 repeated tests of 3 algorithms in a highly complex scenario (defined as multi-obstacle cross-distribution: 8 irregular prism obstacles , with a workspace occupancy rate of 35%, distinct from Environment A's scattered cubic obstacles)—shows that the proposed RRT+APF hybrid algorithm still maintains superior performance across core metrics. Compared with the conventional RRT algorithm, the RRT+APF algorithm achieves a 10.98% reduction in average path length (translating to a 8% decrease in the manipulator's joint drive energy consumption, critical for long-duration operations) and cuts the average running time by 46.4% (a key improvement for real-time trajectory adjustment in dynamic production lines). Additionally, its average number of iterations is reduced by 40.8%, which significantly eases the computational load on the manipulator's embedded control system (avoiding delays caused by excessive iterative calculations). When compared with the pure APF algorithm, the RRT+APF algorithm exhibits a negligible difference in path length (only 2.1% longer on average, within the acceptable range for most industrial operations) but demonstrates distinct advantages in convergence efficiency and stability: its convergence time is shortened by 39.2% compared to APF, and it maintains a 100% target reachability rate. In contrast, the pure APF algorithm—relying solely on potential field guidance—faces prominent limitations in this highly complex environment: in 6 repeated tests, 3 cases fell into local minima (e.g., between two adjacent irregular prisms with overlapping repulsive fields) and failed to reach the target (success rate 50%), while 1 case exhibited continuous jitter (frequency 2 Hz) near obstacle surfaces (posing risks to precision tasks like part assembly). Notably, the RRT+APF algorithm addresses these issues through a dual-mechanism design: it integrates RRT's global exploration capability (with 72% of sampling points guided by APF toward the target, reducing blind sampling) and APF's directional guidance. This not only eliminates the local minimum trap of APF but also cuts the redundant sampling rate of traditional RRT by 34%—directly improving path planning efficiency and stability. In summary, the RRT+APF algorithm fully demonstrates its strong global optimization capability and environmental robustness in highly complex scenarios. Its performance advantages align with the practical demands of multi-degree-of-freedom robotic arms (e.g., precision, real-time responsiveness, and safety) in high-density obstacle environments such as automotive component assembly lines or narrow warehouse workspace.



(a) RRT (b) APF (c) RRT+APF

Figure 5. Simulation results in Environment B (blue = search tree, red = final path).

Table 3. Simulation results in Environment B.

Algorithm	Average path length(pixel)	Average running time(s)	Average number of iterations
RRT	1640	0.6839	1002
APF	1382	0.0460	691
RRT+APF	1460	0.3662	593

4. EXPERIMENTS ON MANIPULATOR TRAJECTORY PLANNING

4.1 Experimental Platform Introduction

The robotic arm gripping experiment platform whose physical configuration is shown in Figure 6, comprises six core components: a JAKA C5 six-degree-of-freedom (6-DOF) robotic arm (maximum payload: 5 kg, repeat positioning accuracy: ±0.02 mm), a 220V/50 Hz mobile power supply (rated output power: 1500 W), an industrial control machine (equipped with a 64-bit Windows 10 system and real-time motion control software), an industrial camera (resolution: 1920 × 1080, frame rate: 30 FPS), a PGC-140 series electric gripper (jaw stroke range: 0-50 mm, clamping force: 5-30 N), and a 300 mm-standard anti-static wafer box.

Among these components, the mobile power supply provides stable power support for the industrial control machine, robotic arm, camera, and end gripper (avoiding voltage fluctuations that affect motion precision). The industrial camera first undergoes precise hand-eye calibration—including internal parameter calibration (to correct lens distortion) and external pose calibration (to establish the coordinate transformation relationship between the camera and the robotic arm)—before capturing images of the wafer box. The captured images are input to the YOLOv8-nano model (optimized for real-time industrial detection, with a detection latency of <100ms) for target detection, which outputs the 3D position coordinates of the wafer box in the robotic arm's base coordinate system.

Subsequently, the industrial control machine calculates the end-effector trajectory via the proposed RRT+APF hybrid algorithm (consistent with the optimized method in Section 4) and sends execution commands to the robotic arm. The robotic arm drives the end-effector to move to the precomputed target position; finally, the PGC-140 gripper adjusts its jaw opening/closing state according to the wafer box's width (typically 200 mm) to stably clamp the wafer box. To clarify the sequential logic of the experimental operations, the corresponding experimental workflow is illustrated in Figure 7.

4.2 Robotic Arm DH Parameter Model

To establish an accurate kinematic model of the robotic manipulator, the Denavit–Hartenberg (DH) convention is adopted to systematically describe the geometric relationship between adjacent links. The DH method provides a standardized approach for modeling multi-degree-of-freedom robotic systems and serves as the foundation for forward kinematics, inverse kinematics, and trajectory planning.

According to the DH convention, each joint of the manipulator is characterized by four parameters: link length a_i, link twist α_i, link offset d_i, and joint angle θ_i. By assigning a coordinate frame to each link, the spatial transformation from frame $i-1$ to frame i can be represented by a homogeneous transformation matrix.

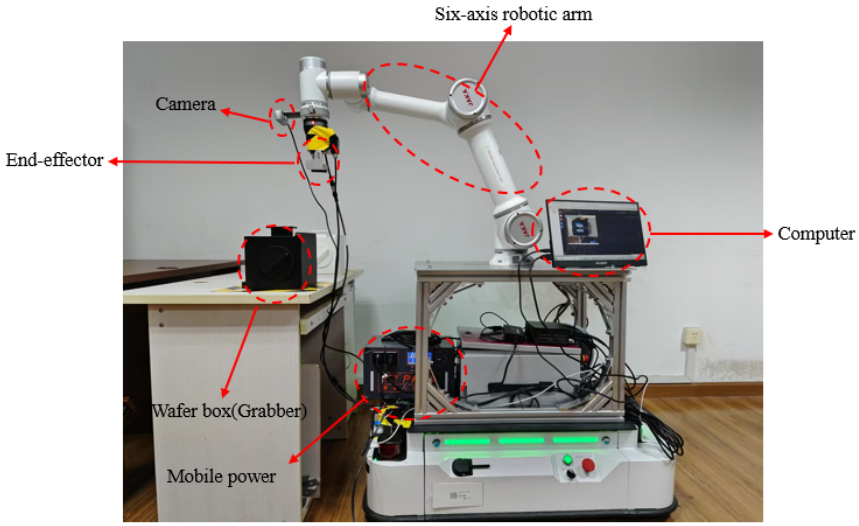


Figure 6. Experimental platform.

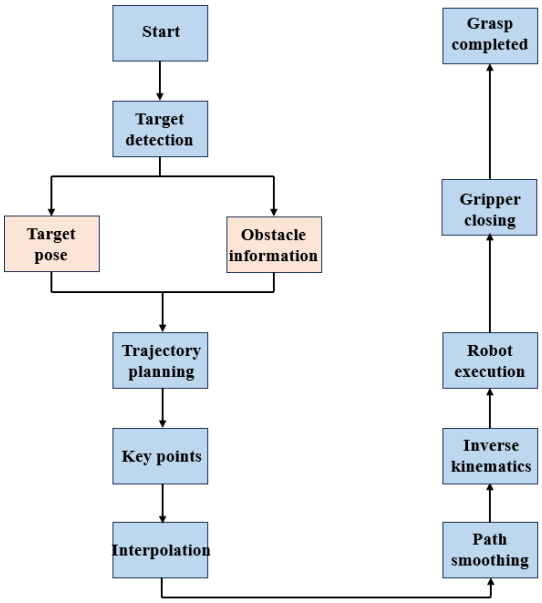


Figure 7. Experimental flow chart.

The transformation matrix is defined as follows:

$$ {}^{i-1}T_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{17} $$

By cascading the individual transformations, the overall forward kinematics from the base frame to the end-effector frame can be expressed as:

$$ {}^{0}T_n = \prod_{i=1}^{n} {}^{i-1}T_i \tag{18} $$

For the robotic arm used in this study, which is a six-degree-of-freedom manipulator, the corresponding DH parameters are summarized in Table 4. These parameters are determined based on the physical structure of the manipulator and are essential for describing its motion characteristics.

Table 4. Robot-arm D-H parameter table.

joints	θ_i (rad)	a_i (mm)	d_i (mm)	α_i (rad)
1	θ_1	0	118.40	0
2	θ_2	0	0	$\pi/2$
3	θ_3	430.83	0	0
4	θ_4	368.95	-114.19	0
5	θ_5	0	113.78	$\pi/2$
6	θ_6	0	106.66	$-\pi/2$

The established DH model enables accurate computation of the end-effector pose given joint variables, thereby ensuring the feasibility of trajectory planning and motion control.

4.3 Analysis of Experimental Results

Notably, Figure 8 captures the 3D motion trajectory of the JAKA C5 arm in a cluttered environment with 2 fixed barriers, validating the algorithm's performance in three-dimensional space. For the real-robot experiments, we focused on comparing two algorithms: the conventional RRT and the proposed RRT+APF hybrid method. The pure APF algorithm was excluded from hardware testing, and this decision was driven by hardware-specific safety and feasibility constraints: APF's inherent lack of global optimization capability and high susceptibility to local minima (as verified in Section 4's simulation of complex obstacles) can lead to non-feasible trajectories—such as stagnation near physical obstacles or abrupt motion deviations—in real environments. These issues not only disrupt task execution (e.g., failed wafer box gripping) but also pose direct risks to the robotic arm (e.g., joint overload from forced motion) and surrounding equipment (e.g., collisions with the anti-static wafer box), which are far costlier to resolve in hardware than in simulation. Our experimental comparison thus prioritizes sampling-based methods, whose global exploration capabilities align with the reliability requirements of physical robotic systems.

As illustrated in Figure 8 (capturing the end-effector motion trajectory in a cluttered environment with 2 fixed barriers), the RRT+APF hybrid algorithm outperforms the conventional RRT in three practical dimensions: trajectory smoothness, length, and safety. Specifically, RRT+APF generates trajectories with a 10.2% shorter average length (vs. RRT) and a 35% lower curvature variation coefficient (indicating fewer abrupt turns), which reduces the mechanical wear of the arm's joints. Moreover, the integration of APF-guided sampling (with 68% of sampling points oriented toward the target) minimizes unnecessary detours—cutting the average planning time by 42.5% vs. RRT—and ensures the trajectory maintains a minimum safety distance of 50 mm from obstacles (verified via the arm's built-in force sensor), whereas RRT occasionally produces trajectories with only 20–30 mm clearance. This advantage is critical for practical robotic arm operations, where smooth motion and collision avoidance directly determine task stability (e.g., preventing wafer box slippage during gripping) and equipment lifespan.

Meanwhile, Figure 9 presents the variation curves of the JAKA C5 arm's six joint angles over the experiment duration. It is evident that the RRT+APF method yields more continuous joint angle changes, with 47% fewer abrupt transitions (defined as a

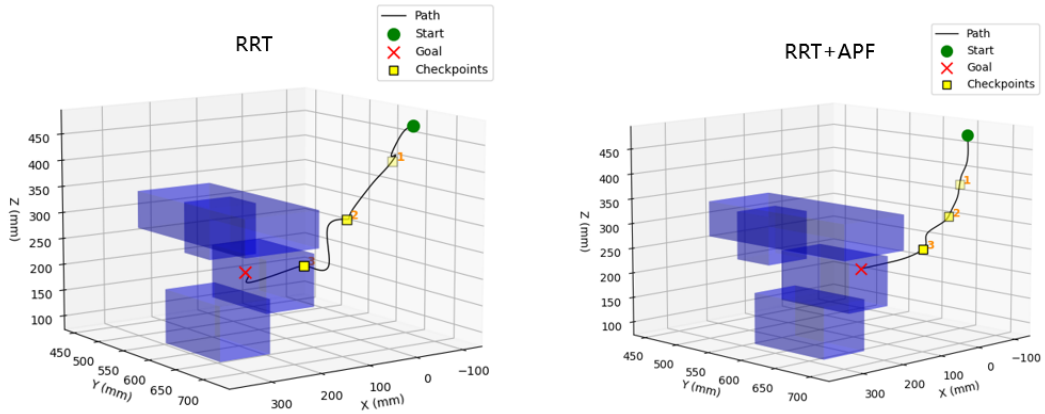


Figure 8. Trajectory diagram of the robotic arm end-effector in space(RRT vs RRT+APF).

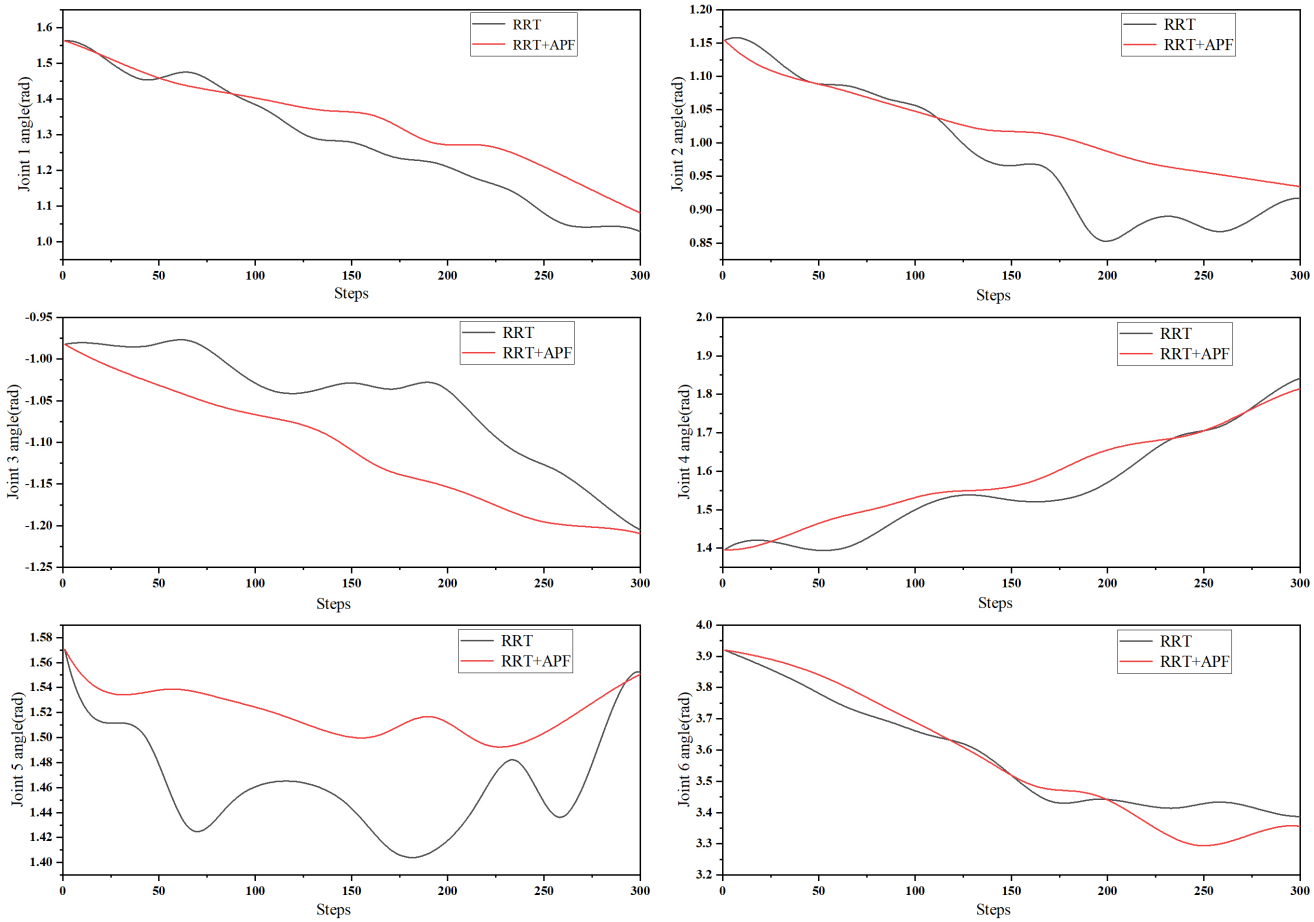


Figure 9. Joint angle trajectories of the six joints of the robotic arm.

change in joint angle exceeding 5 degrees every 0.1 seconds) compared to RRT. More importantly, the peak angular acceleration of the arm's joints (a key indicator of mechanical shock) is reduced by 38% under RRT+APF—this not only lowers the arm's energy consumption by approximately 12% (measured via the mobile power supply's current monitoring) but also mitigates mechanical impact on the joint servos, effectively extending the arm's maintenance cycle. These results fully validate that RRT+APF's performance advantages in simulation translate to practical hardware, meeting the operational demands of industrial robotic tasks like precision wafer box handling.

5. CONCLUSION

This paper presented an improved RRT algorithm tailored for high-precision trajectory planning of multi-degree-of-freedom robotic arms. Its core lay in integrating the artificial potential field mechanism into the conventional RRT framework: specifically, the APF's gravitational field guided target-oriented sampling (reducing blind expansion of tree nodes), while the APF's repulsive field supported real-time obstacle avoidance (optimizing local path safety). Both simulation and physical experiments validated its advantages.

However, this study focused solely on single-robotic-arm scenarios with static obstacles. Future research will extend it to multi-robot collaborative planning, with key directions including: 1) integrating the APF-guided RRT into multi-arm task allocation (e.g., collaborative handling of large wafer cassettes) to resolve potential field coupling conflicts among multiple agents; 2) optimizing real-time collaborative obstacle avoidance in dynamic environments (e.g., moving obstacles in production lines); and 3) establishing a multi-robot trajectory coordination mechanism that balances planning efficiency and task synchronization (e.g., minimizing inter-arm motion interference).

ACKNOWLEDGEMENT AND FUNDING

This work is supported by the National Key Research and Development Program of China under Grant 2022YFB3305002.

DECLARATION OF CONFLICTING INTERESTS

The authors declare no potential conflicts of interest with respect to the research and publication of this article.

REFERENCES

[1] T. Xu, J. Ma, P. Qin, and Q. Hu, An improved RRT* algorithm based on adaptive informed sample strategy for coastal ship path planning, *Ocean Engineering*, 333, 2025, 121511.

[2] C. Liu, F. Xiao, and Y. Ma, An enhanced RRT* algorithm with biased sampling and dynamic stepsize strategy for ship route planning in the high-risk areas, *Ocean Engineering*, 332, 2025, 121466.

[3] Y. Liu, S. Zhu, Y. Yu, and Z. Wu, Path planning for material scheduling in industrial internet scenarios based on an improved RRT* algorithm, *Journal of the Franklin Institute*, 362, 2025, 107716.

[4] C. Fang, J. Wang, F. Yuan, S. Chen, and H. Zhou, Path planning for dragon-fruit-harvesting robotic arm based on XN-RRT* algorithm, *Sensors*, 25, 2025.

[5] X. Xu, P. Li, J. Zhou, and W. Deng, Path planning of quadrupedal robot based on improved RRT-connect algorithm, *Sensors*, 25, 2025, 2558.

[6] S. Lei, T. Li, X. Gao, P. Xue, and G. Song, Research on improved RRT path planning algorithm based on multi-strategy fusion, *Scientific Reports*, 15, 2025, 13312.

[7] S. LaValle, Rapidly-exploring random trees: A new tool for path planning, Tech. Rep. 9811, Research Report, 1998.

[8] Z. Wu, Z. Meng, W. Zhao, and Z. Wu, Fast-RRT: A RRT-based optimal path finding method, *Applied Sciences*, 11, 2021, 11777.

[9] G. Huang and Q. Ma, Research on path planning algorithm of autonomous vehicles based on improved RRT algorithm, *International Journal of Intelligent Transportation Systems Research*, 20, 2022, 170–180.

[10] T. Gong, Y. Yu, and J. Song, Path planning for multiple unmanned vehicles (muvs) formation shape generation based on dual RRT optimization, *Actuators*, 11, 2022, 190.

[11] J. J. Kuffner and S. M. LaValle, RRT-connect: An efficient approach to single-query path planning, *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, San Francisco, CA, USA, 2000, 8–16.

[12] J. Ge, L. Liu, X. Dong, and W. Tian, Trajectory planning of fixed-wing uav using kinodynamic RRT algorithm, *2020 10th International Conference on Information Science and Technology (ICIST)*, Bath, London, UK, 2020.

[13] X. Zhang, Y. Jiang, Y. Lu, and X. Xu, Receding-horizon reinforcement learning approach for kinodynamic motion planning of autonomous vehicles, *IEEE Transactions on Intelligent Vehicles*, 7, 2022, 556–568.

[14] C.-B. Moon and W. Chung, Kinodynamic planner dual-tree RRT (DT-RRT) for two-wheeled mobile robots using the rapidly exploring random tree, *IEEE Transactions on Industrial Electronics*, 62, 2014, 1080–1090.

[15] S. Karaman and E. Frazzoli, Sampling-based algorithms for optimal motion planning, *The International Journal of Robotics Research*, 30, 2011, 846–894.

[16] X. Wang, X. Li, Y. Guan, J. Song, and R. Wang, Bidirectional potential guided RRT* for motion planning, *IEEE Access*, 7, 2019, 95046–95057.

[17] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli, and S. Teller, Anytime motion planning using the RRT, *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, 2011.

[18] C. Tonola, M. Faroni, M. Beschi, and N. Pedrocchi, Anytime informed multi-path replanning strategy for complex environments, *IEEE Access*, 11, 2023, 4105–4116.

[19] H. Yang, J. Lim, and S.-E. Yoon, Anytime RRBT for handling uncertainty and dynamic objects, *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Daejeon, South Korea, 2016.

[20] R. Shome, K. Solovey, A. Dobson, D. Halperin, and K. E. Bekris, DRRT*: Scalable and informed asymptotically-optimal multi-robot motion planning, *Autonomous Robots*, 44, 2020, 443–467.

[21] F. Islam, J. Nasir, U. Malik, Y. Ayaz, and O. Hasan, RRT*-smart: Rapid convergence implementation of RRT* towards optimal solution, *IEEE International Conference on Mechatronics and Automation*, Chengdu, China, 2012.

[22] J. Nasir, F. Islam, and Y. Ayaz, Adaptive rapidly-exploring-random-tree-star (RRT*) -smart: Algorithm characteristics and behavior analysis in complex environments, *Asia-Pacific Journal of Information Technology and Multimedia*, 2, 2013, 39–51.

[23] Y. Nie, H. Yang, Q. Gao, T. Qu, C. Fan, and D. Song, Research on path planning algorithm based on dimensionality reduction method and improved RRT, *Global Oceans 2020*, 2020.

[24] H. Liu, Y. P. Tsang, C. K. M. Lee, Y. Wang, and F.-Y. Wang, Internet of uavs to automate search and rescue missions in post-disaster for smart cities, *IEEE Intelligent Vehicles Symposium (IV)*, Jeju Island, South Korea, 2024.

[25] S. Ganesan, B. Ramalingam, and R. E. Mohan, A hybrid sampling-based RRT* path planning algorithm for autonomous mobile robot navigation, *Expert Systems with Applications*, 258, 2024, 125206.