Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

A Dynamic Hybrid Sampling Improved RRT* Path Planning Method of the Manipulator for Diverse Map Environments

Yuanlin Guan

guanyuanlin@qut.edu.cn

Qingdao University of Technology

Yibo Cheng

Qingdao University of Technology

Qi Cheng

Qingdao University of Technology

Xixin Yang

Qingdao University

Qiulin Chen

Qingdao University of Technology

Chunyu Yu

Qingdao University of Technology

Article

Keywords: Path Planning, DHSIRRT* Algorithm, Hybrid Sampling, Dynamic Step Size, Manipulator

Posted Date: November 20th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11465275/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

A Dynamic Hybrid Sampling Improved RRT* Path Planning Method of the Manipulator for Diverse Map Environments

Yuanlin Guan[a*] Yibo Cheng[a] Qi Cheng[a] Xixin Yang[b] Qiulin Chen[a] Chunyu Yu[a]

a. School of Mechanical and Automotive Engineering, Qingdao University of Technology, Qingdao 266520, China.

b. College of Computer Science and Technology, Qingdao University, Qingdao 266071, China.

Abstract

To address the core limitations of the original rapidly-exploring random tree (RRT) algorithm in adapting to diverse map environments, including slow convergence, poor path quality, and weak environmental adaptability, a dynamic hybrid sampling improved RRT* (DHSIRRT*) algorithm is proposed. It quantifies environmental complexity through dual metrics of obstacle density and relative channel width, and achieves continuous, smooth adjustment of the weights for uniform sampling, Gaussian sampling, and Cauchy sampling to enable adaptive adjustment of sampling weights across open, normal, dense, and maze typical maps. A two-factor dynamic step size model integrating obstacle distance and path curvature is built to maximize expansion efficiency while guaranteeing collision-free motion, and a two-step strategy of pruning and node fine-tuning improves path smoothness and reduce path length. Statistical experiments with 100 independent runs are conducted in 2D, 3D, and manipulator workstation scenarios. The results show that, compared with the original RRT, RRT*, and goal-biased RRT* algorithms, the proposed method achieves notable reductions in average iterations, planning time, and path length in dense and maze environments, while maintaining a planning success rate above 99%, which verifies the robustness and superiority of the algorithm across diverse map environments.

Keywords: Path Planning; DHSIRRT* Algorithm; Hybrid Sampling; Dynamic Step Size; Manipulator

1. Introduction

Industrial manipulators, as core equipment of intelligent manufacturing, are widely deployed in welding, assembly, material handling, spray painting, and other industrial scenarios[1,2]. The task of manipulator path planning is to find a collision-free path from an initial pose to a target pose in an obstacle-bearing workspace, while satisfying kinematic constraints such as joint limits, velocity, and acceleration[3]. As industrial automation continues to advance, manipulator working environments grow increasingly complex, imposing higher requirements on the real-time performance, optimality, and robustness of path planning algorithms.

*Corresponding author.

Email address: guanyuanlin@qut.edu.cn

Common path planning algorithms include the A* algorithm[4], Dijkstra algorithm[5], Artificial Potential Field (APF)[6], Probabilistic Roadmap (PRM)[7], and rapidly-exploring random tree algorithm (RRT)[8]. The Dijkstra and A* algorithms, both graph-search-based, have simple structures but rely on spatial discretization, making them difficult to apply to the high-dimensional configuration space of multi-degree-of-freedom manipulators[9,10]. The APF algorithm features low computational cost and good real-time performance[11], but it is prone to falling into local minima and suffers from the goal-unreachable defect. PRM is an offline sampling-based planning method with probabilistic completeness[12], but it performs poorly in narrow-passage scenarios and is susceptible to roadmap fracture. The RRT algorithm searches for paths via random sampling[13]. It has a concise framework[14], which is easy to implement, and requires no grid division of high-dimensional space, making it well suited for high-dimensional manipulator scenarios[15,16]. Nevertheless, its random sampling leads to time-consuming and uncertain planning, and it fails to provide optimality guarantees for generated paths[17]. As an optimized derivative of the basic RRT algorithm, RRT* mitigates the above-mentioned drawbacks via parent-node reselection[18] and local rewiring[19]: it reduces path cost fluctuation caused by blind sampling and achieves asymptotic optimality, and has become a mainstream algorithm for manipulator path planning, but RRT* still retains multiple inherent common drawbacks in practical application, including low sampling efficiency, fixed step size leading to poor adaptation to complex obstacle environments, and redundant nodes and unsmooth trajectories in generated paths, which restrict its overall planning performance.

To address the aforementioned limitations of the RRT* algorithm, researchers worldwide have proposed various improved variants. Targeting the sampling efficiency defect, Gammell et al.[20] developed Informed RRT*, which restricts sampling to an ellipsoidal heuristic region and discards invalid samples to accelerate convergence, but has limited adaptability to narrow-obstacle scenarios. Aiming at the fixed step size defect that restricts environmental adaptability, Cao et al.[21] proposed an improved RRT*-Connect integrating elliptical sampling, potential-field gradient adaptive step size, and piecewise Bézier smoothing, with a reconstructed cost function to balance path length and obstacle-avoidance safety. Chen et al.[22] presented a fusion algorithm based on Informed-RRT* with potential-field pre-screening, goal bias, and bidirectional segmented planning to reduce undirected global sampling, plus dynamic step size for robust obstacle avoidance in complex spaces. For the path quality defect of redundant nodes and poor smoothness, Fang et al.[23] proposed XN-RRT* with normal-distribution dynamic sampling, improved artificial potential field guidance, greedy redundant-node elimination, and cubic B-spline smoothing to further shorten paths and improve stability.

Although RRT* and its variants have achieved progress in sampling guidance[24-26] and path optimization[27], it still lack a unified adaptive mechanism for diverse map environments. To address this issue, this paper proposes DHSIRRT* algorithm with three core contributions:

(1) A dynamic-weight hybrid sampling strategy is proposed for diverse map environments. The complexity of four typical maps (open, normal, dense, and maze) is quantified by fusing global obstacle density and relative channel width. Based on the quantified complexity, a normalized Sigmoid mapping continuously adjusts the weights of uniform, Gaussian, and Cauchy sampling,

enabling an adaptive balance between exploration and exploitation in these four map environments.

(2) A two-factor dynamic step size model is developed. Step size is adjusted according to both obstacle distance and path curvature—large steps for exploration in open areas, and small steps for expansion near obstacles and in turning regions—striking a balance between planning efficiency and motion safety.

(3) A path pruning and node fine-tuning strategy is designed. Redundant nodes are removed to compress path length. Path inflection points are fine-tuned based on the interior-angle bisector principle to further improve smoothness and optimality under collision-free constraints.

Comprehensive simulations are conducted to compare the proposed method with RRT, RRT*, and goal-biased RRT*, verifying its effectiveness and superiority in manipulator path planning.

2. DHSIRRT* Algorithm

Targeting the key limitations of RRT* including a fixed sampling strategy, non-adaptive expansion step size, redundant trajectory nodes, and insufficient smoothness, this paper conducts systematic improvements to the RRT* algorithm across three dimensions: sampling strategy, expansion step size, and path optimization.

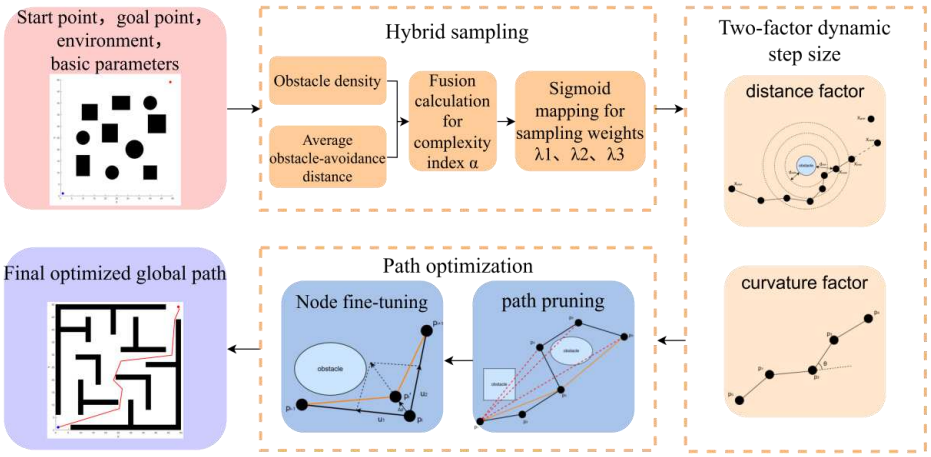


Figure 1. Overall framework of the proposed algorithm.

The overall architecture of the proposed method as shown in Fig. 1 is divided into three main modules: a hybrid sampling module, a dynamic step-size module, and a path optimization module. First, the algorithm takes the start point, goal point, environmental map, and basic parameters as inputs. Next, the environmental complexity index α is quantified based on obstacle density and average obstacle-avoidance distance, and the weights of the three sampling distributions are obtained via Sigmoid function mapping. Subsequently, node expansion is achieved using a two-factor dynamic step size integrating obstacle distance and path curvature. Finally, the resulting initial path undergoes path pruning and node fine-tuning to output a smooth, collision-free optimized global path.

2.1 Multi-Map Adaptive Hybrid Sampling

original RRT* generally employs either uniform sampling or fixed-ratio goal-biased sampling. Both strategies remain fixed throughout the planning process and fail to adapt to environmental changes. This paper proposes a hybrid sampling strategy that weights and combines three complementary probability distributions: uniform, Gaussian, and Cauchy. Through global feature quantification and smooth weight mapping, the weights of the three distributions are dynamically adjusted to ensure full global coverage (avoiding local optima) while intensifying sampling in critical regions. The fusion model is:

$$X_{\text{rand}} \sim D_{\text{hybrid}} = \lambda_1 U(C_{\min},\ C_{\max}) + \lambda_2 N(X_{\text{goal}},\ \sigma^2 \text{I}) + \lambda_3 C(\mu,\ \gamma) \tag{1}$$

where $\lambda_1 + \lambda_2 + \lambda_3 = 1$. λ_1, λ_2, λ_3 denote the weights of uniform, Gaussian, and Cauchy sampling, respectively. $\text{U}(C_{\min},\ C_{\max})$ is the uniform distribution over the configuration space. $N(X_{\text{goal}},\ \sigma^2 \text{I})$ is the multidimensional Gaussian distribution centered at the goal, and $C(\mu,\ \gamma)$ is the multidimensional independent Cauchy distribution.

The three distributions play distinct roles:

(1) Uniform sampling generates samples with equal probability across the entire configuration space C, guaranteeing coverage of unknown regions and preventing local optima.

$$x_{i1} = C_{\min} + (C_{\min} - C_{\max}) \cdot u_i,\ \ u_i \sim U(0,1) \tag{2}$$

where $C_{\min}$ and $C_{\max}$ are the lower and upper bounds of the configuration space, respectively.

(2) Gaussian sampling generates samples centered on the goal point, providing strong goal orientation to accelerate tree convergence.

$$x_{i2} = X_{\text{goal}} + \sigma \cdot z,\ \ z \sim N(0,1) \tag{3}$$

where X_{goal} is the goal point, σ is the standard deviation, and z is a standard normal sample.

(3) Cauchy sampling exploits the heavy-tailed property of the Cauchy distribution: samples can jump to distant regions with non-negligible probability, helping to traverse narrow passages and escape local traps.

$$x_{i3} = \mu + \gamma \tan[\pi(u_i - 0.5)],\ \ u_i \sim u(0,1) \tag{4}$$

where μ is the location parameter (distribution center) and γ is the scale parameter controlling distribution width.

To precisely match sampling weights to environment features, we quantify environmental complexity from two perspectives—obstacle occupancy and feasible space width, and construct a global map complexity index α that fuses the two normalized geometric features:

$$\alpha = \sqrt{\frac{\left(\frac{S_{\text{obs}}}{S_{\text{total}}}\right)^2 + \left(1 - \frac{\bar{d}}{d_{\text{ref}}}\right)^2}{2}} \tag{5}$$

where S_{obs} is the space occupied by obstacles, S_{total} is the total configuration space, and $\frac{S_{obs}}{S_{total}}$ is the global obstacle density (space congestion). $\bar{d}$ is the average obstacle-avoidance distance of free-space samples, d_{ref} is the reference distance, and $1 - \frac{\bar{d}}{d_{ref}}$ characterizes channel narrowness. $\alpha \in [0,\ \ 1]$, with larger values indicating higher planning difficulty.

To map α to the three weights λ_1, λ_2, λ_3 without the abruptness of linear mapping, a normalized Sigmoid function is used as a smooth interpolation factor. The standard Sigmoid function is:

$$\text{Sigmoid}(\alpha) = \frac{1}{1 + \text{e}^{[-k\cdot(\alpha-0.5)]}} \tag{6}$$

where *k* is the steepness coefficient. The normalized interpolation factor is:

$$f(\alpha) = \frac{\text{Sigmoid}(\alpha) - \text{Sigmoid}(0)}{\text{Sigmoid}(1) - \text{Sigmoid}(0)} \tag{7}$$

Two anchor points for extreme scenarios are defined based on the functional properties of the three sampling types. Linear interpolation between the anchors via $f(\alpha)$ yields the weights at any complexity level:

$$\begin{cases} \lambda_1 = A_1 + (B_1 - A_1)\cdot f(\alpha) \\ \lambda_2 = B_2 - (B_2 - A_2)\cdot f(\alpha) \\ \lambda_3 = A_3 + (B_3 - A_3)\cdot f(\alpha) \end{cases} \tag{8}$$

where A_1, A_2, A_3 are the lower weight bounds and B_1, B_2, B_3 are the upper weight bounds. In each sampling step, weighted sampling is performed according to the current environment weights, realizing dynamic adaptation of the sampling strategy and providing high-quality input for subsequent tree expansion.

2.2 Two-Factor Dynamic Step Size

Although the hybrid sampling strategy optimizes the algorithm at the sample distribution level, the search tree expansion still relies on a fixed step size. It is difficult to balance obstacle-avoidance safety and search convergence speed, and non-smooth paths are produced with abrupt local turns. Therefore, we propose a two-factor dynamic step-size model integrating obstacle distance and path curvature, which adjusts the expansion step size in terms of both obstacle safety margin and path turning characteristics, achieving a balance between planning efficiency and path smoothness while mitigating collision risks.

The obstacle-distance factor regulates the expansion step size based on the distance from a node to the nearest obstacle to ensure safety during expansion. The Euclidean distance from the current expansion node to the nearest obstacle is computed to evaluate the safety margin of the node's local region. A large step size is used to accelerate spatial exploration when the node is far from obstacles, while the step size is actively reduced to lower collision probability when the node is in close proximity to obstacles. The piecewise function of the obstacle-distance factor is as follows:

$$\eta_{\text{obs}} = \begin{cases} s_{\max}, & d_{\min} \geq 3d_{\text{safe}} \\ 1.0, & 2d_{\text{safe}} \leq d_{\min} < 3d_{\text{safe}} \\ \dfrac{d_{\min} - d_{\text{safe}}}{d_{\text{safe}}}, & d_{\text{safe}} \leq d_{\min} < 2d_{\text{safe}} \\ s_{\min}, & d_{\min} < d_{\text{safe}} \end{cases} \tag{9}$$

where d_{safe} denotes the preset safety distance, $s_{\max}$ represents the upper bound of step size, and $s_{\min}$ denotes the lower bound of step size. The variation behavior of the obstacle-distance factor is shown in Fig. 2.

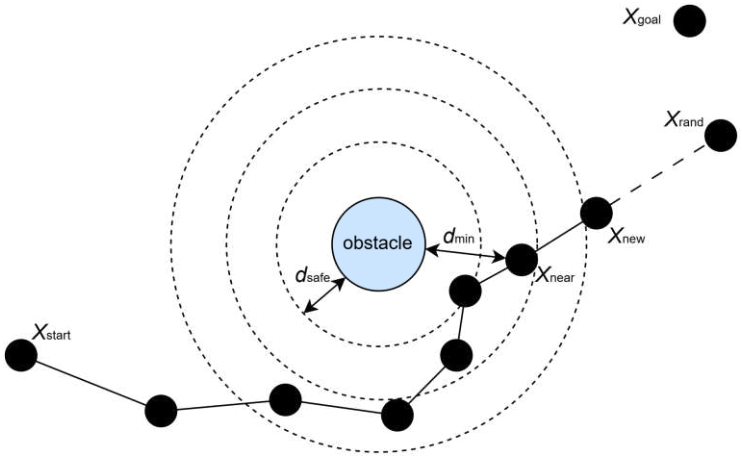


Figure 2. Obstacle distance factor.

Nevertheless, step-size adjustment relying solely on obstacle distance can only satisfy obstacle-avoidance requirements and cannot constrain the geometric profile of the path. When the search tree expands with large-angle turns, the distance-only adjustment mechanism generates zig-zag paths, degrading trajectory smoothness and likely causing manipulator motion jitter and abrupt trajectory changes. Therefore, a path-curvature factor to perform secondary optimization of the step size according to the local bending characteristics of the path is introduced. The geometric definition of the path turning angle is illustrated in Fig. 3.

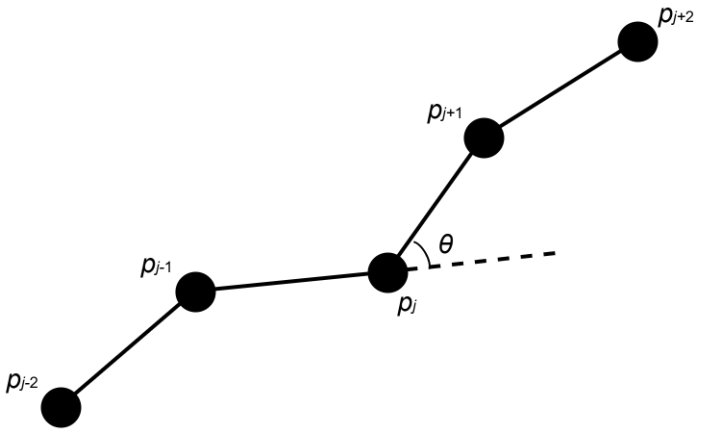


Figure 3. Path turning angle θ.

Three consecutive nodes p_{j-1}, p_j, and p_{j+1} on the random tree are taken, where p_j is the parent node for the current expansion. The path turning angle θ at p_j is calculated via the vector dot product of two adjacent path segments to quantify the local bending degree of the path.

$$\theta = \arccos\left(\frac{(p_j - p_{j-1}) \cdot (p_{j+1} - p_j)}{\|p_j - p_{j-1}\| \cdot \|p_{j+1} - p_j\|}\right) \tag{10}$$

The path turning angle ranges over $[0, \ \pi]$, where $\theta = 0$ indicates that the three points are collinear with an unchanged travel direction. Larger values of θ correspond to sharper path turns and higher local curvature.

Considering the negative correlation between turning angle and path smoothness, a curvature factor η_{curve} is constructed. Smaller turning angles yield larger factor values and longer step sizes, whereas sharper turns produce smaller factor values and shorter step sizes. The specific function is given below:

$$\eta_{\text{curve}} = \begin{cases} s_{\max} - \dfrac{3.5\theta}{\pi}, & \theta < \pi/4 \\ s_{\min}, & \theta \geq \pi/4 \end{cases} \tag{11}$$

where s_{max} and s_{min} denote the maximum and minimum step sizes, respectively. When the turning angle is less than $\pi/4$, the path undergoes a mild turn, and the curvature factor decays linearly with the increase of the turning angle, gradually reducing the step size to suppress path jitter. Otherwise, when the turning angle is greater than or equal to $\pi/4$, the path is regarded as a sharp turn. Hence, the factor is clamped to the lower bound of the step size to prevent substantial degradation of expansion efficiency caused by excessive step-size decay.

Combining the two factors described above, the final dynamic expansion step size s is expressed as:

$$s = s_{\text{base}} \cdot \sqrt{\eta_{\text{obs}} \cdot \eta_{\text{curve}}} \tag{12}$$

where s_{base} is the base step size. Upper and lower bounds are imposed on the step size to ensure it remains within a reasonable range.

Iterative expansion and node updating of the search tree are performed using the dual-factor dynamic step-size rule until the goal region is reached, yielding the final initial path.

2.3 Path Optimization

The initial path generated by RRT*, obtained by backtracking from nodes in the search tree, typically contains a large number of redundant nodes and unnecessary turns, making it difficult to directly meet the smoothness and motion feasibility requirements of path planning. Hence, this paper designs a two-step processing strategy of path pruning and node fine-tuning. Redundant intermediate nodes are removed via path pruning to compress the overall path length, and retained path waypoints undergo local geometric fine-tuning to improve path smoothness.

Path pruning eliminates redundant nodes to maximally compress path length without altering the topological feasibility of the path. The initial path obtained from RRT* search is as an ordered sequence of points $\{p_1,\ p_2,\ \ldots,\ p_M\}$, where $p_1 = x_{\text{start}}$ and $p_M = x_{\text{goal}}$. The objective of pruning is to retain as few path points as possible such that all consecutive retained points can be connected by straight collision-free lines. Therefore, a forward pruning strategy that proceeds overall from the start point toward the goal point is adopted. To efficiently locate the farthest connectable node, a reverse-order checking method from the goal point to the current node is employed at each step. p_1 is set as the starting point with the current point index $j = 1$, and the connectivity of remaining path points to p_j is tested in reverse order starting from the goal point. The first collision-free node identified is the farthest connectable node, which is added to the pruned path and updates the current point. This process is repeated until the goal point is reached, finally yielding the orange path shown in Fig. 4. The path removes a large number of collinear or detouring redundant nodes, resulting in a significant reduction in path length.

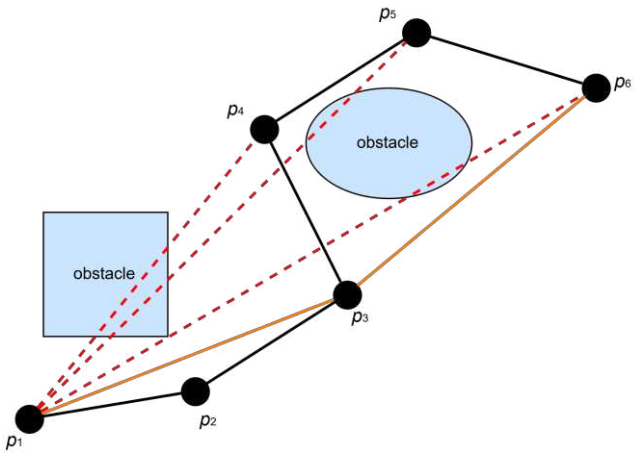


Figure 4. Path pruning.

Although the number of nodes is greatly reduced after pruning, unnatural sharp turns may still occur near obstacle corners, leaving room for further optimization of path length while path smoothness remains inadequate. Therefore, local geometric fine-tuning is performed on the nodes of the pruned path in the manner illustrated in Fig. 5.

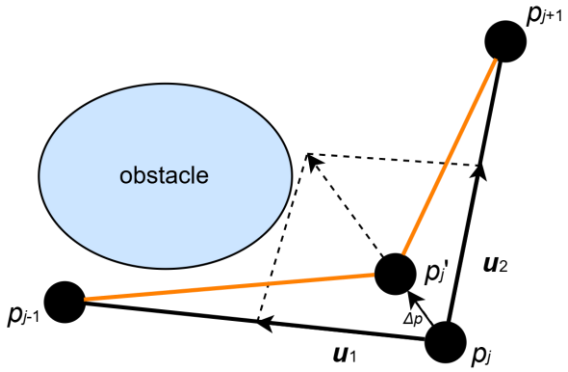


Figure 5. Node fine-tuning.

The pruned path is denoted as $\boldsymbol{P} = \{p_1,\ p_2,\ \ldots,\ p_K\}$. For each interior node $p_j(j = 2,\ \ldots,\ \mathrm{K}-1)$, the unit direction vectors u_1 and u_2 toward its preceding and succeeding adjacent nodes are computed, respectively.

$$\boldsymbol{u_1} = \frac{(p_{j-1} - p_j)}{\|p_{j-1} - p_j\|},\quad \boldsymbol{u_2} = \frac{(p_{j+1} - p_j)}{\|p_{j+1} - p_j\|} \tag{13}$$

The direction of the resultant vector $\boldsymbol{u_1} + \boldsymbol{u_2}$ approximates the interior angle bisector of the angle formed by the two adjacent path segments. The current node $\mathrm{p_i}$ is translated along this direction by a small displacement Δp, yielding a candidate new node $p_j{}' = p_j + \Delta p$.

$$\Delta p = 0.15 \cdot s_{\mathrm{base}}(\boldsymbol{u_1} + \boldsymbol{u_2}) \tag{14}$$

where s_{base} is the base step size. If both new line segments $\overline{p_{J-1}p_J{}'}$ and $\overline{p_J{}'p_{J+1}}$ remain collision-free with obstacles, the original node p_j is replaced by $p_j{}'$. The fine-tuning operation draws inflection points toward the interior of the path angle, effectively reducing turning angles, eliminating zigzag jitter, and further shortening the overall path length.

Through the two-stage optimization of path pruning and node fine-tuning, the final generated path features both a minimal number of nodes and the merits of smoothness and short length. It significantly improves the quality of the initial RRT* solution and satisfies the motion execution requirements of manipulators.

3. Experiment Simulation and Validation

To evaluate the comprehensive performance of the proposed algorithm, the calibrations of key algorithm parameters are completed, including the steepness coefficient of the Sigmoid mapping, sampling weight anchor points, and the upper and lower bounds of the dynamic step size. Comparative simulation experiments and analyses are conducted between the DHSIRRT* algorithm and the original RRT algorithm, original RRT* algorithm, and goal-biased RRT* algorithm across a variety of scenarios, to verify the effectiveness and environmental robustness of the DSHIRRT* algorithm.

3.1 Experimental Setup

The hardware platform comprises an Intel Core i7-13700H CPU, 32 GB DDR5 memory, and an NVIDIA GeForce RTX 4060 GPU. The algorithm simulation environment is Windows 11 (64-bit) with MATLAB R2024a, and the manipulator simulation environment is Ubuntu 22.04 LTS with ROS 2 Humble as the robot development framework.

3.2 Parameter Calibration

3.2.1 Sigmoid Steepness Coefficient

The steepness coefficient k of the Sigmoid mapping directly governs the smoothness of the transition from environmental complexity α to sampling weights. To determine the optimal value of the coefficient, all other algorithm parameters are fixed, and three typical values — k = 2, 4,

and 6 — are selected to plot the weight curves. The weight transition characteristics for different values are compared as illustrated in Fig. 6.

The results demonstrate that at k = 2, the weight curve is overall flat. As environmental complexity rises, the increase in Cauchy sampling weight exhibits a lag, such that the corridor exploration capability cannot be fully activated in complex scenarios. At k = 6, the weight curve shows a near-step transition around α= 0.5, meaning minor fluctuations in environmental complexity can trigger severe oscillations in the sampling distribution. At k = 4, the weight transition is smooth with a moderate gradient across the full complexity range, achieving a balance between environmental responsiveness and planning stability.

Taking into account the weight evolution characteristics presented by the curves, k = 4 is selected as the steepness coefficient of the normalized Sigmoid mapping.

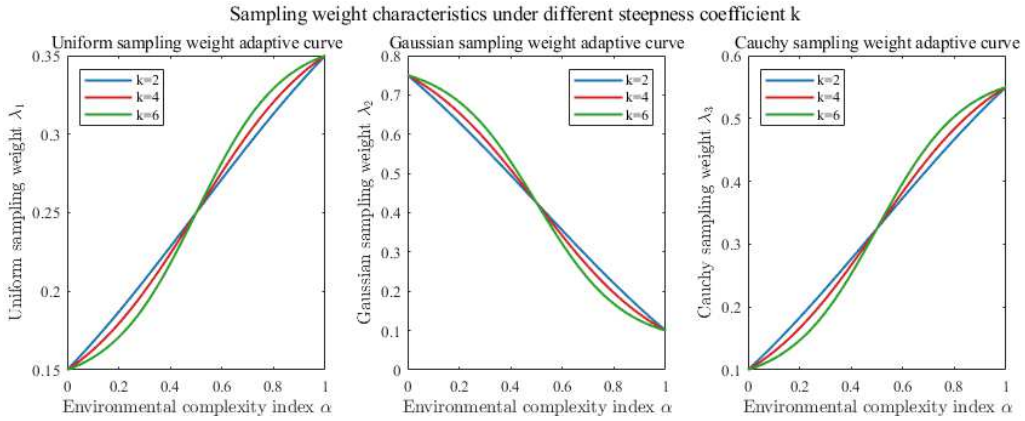
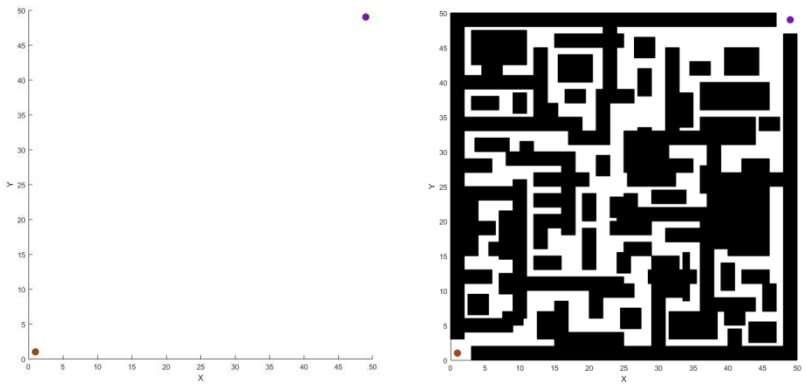


Figure 6. Sampling weight curves under different steepness coefficients (k = 2, 4, 6).

3.2.2 Weight Anchor Points

To determine reasonable anchor point values, we conduct single-variable calibration experiments aligned with the functional roles of the three sampling methods. The calibration adheres to the following constraints: the sampling weights satisfy $\lambda_1 + \lambda_2 + \lambda_3 = 1$. The lower bound of the uniform sampling weight is set to 0.1 to ensure probabilistic completeness. In each experimental group, one category of sampling weight is fixed, and the remaining parameters are adjusted across a gradient.

Theoretically, at α=1 the feasible free space is zero, where planning experiments cannot be performed. Hence, we select a fully open scenario with α=0 (Fig. 7(a)) and a high-complexity maze scenario with α=0.86 (Fig. 7(b)) to complete the calibration, in which the brown and purple points represent the start point and the goal point, respectively.



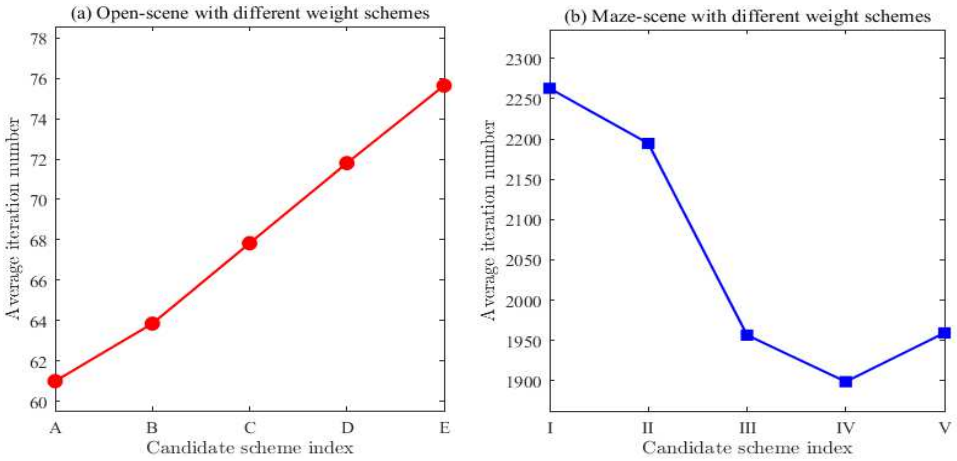
(a) Completely open scenario (α = 0) **(b) high-complexity maze scenario (α = 0.86).**

Figure 7. Calibration scenarios. Brown and purple points denote the start and goal, respectively.

For the fully open scenario, long-range jump exploration is unnecessary for the algorithm. Therefore, the Cauchy sampling weight is fixed, and only the weights of uniform sampling and Gaussian sampling are tuned. Five candidate weight ratios are designed: Scheme A (0.10, 0.80, 0.10), Scheme B (0.15, 0.75, 0.10), Scheme C (0.20, 0.70, 0.10), Scheme D (0.25, 0.65, 0.10), and Scheme E (0.30, 0.60, 0.10). The five schemes adjust the Gaussian sampling proportion in a gradient manner within the range of 0.60–0.80. The Gaussian weight is capped at 0.80 to prevent excessive goal bias from eroding spatial sampling diversity, and maintained above 0.60 to avoid degradation of goal-guidance performance.

For the high-complexity maze scenario, excessively strong goal orientation tends to trap the algorithm in dead ends. Hence, the Gaussian goal-guidance weight is fixed, while the weights of uniform sampling and Cauchy sampling are adjusted. Five candidate ratios are designed: Scheme I (0.45, 0.10, 0.45), Scheme II (0.40, 0.10, 0.50), Scheme III (0.35, 0.10, 0.55), Scheme IV (0.30, 0.10, 0.60), and Scheme V (0.25, 0.10, 0.65). The Cauchy sampling weight is tuned in a gradient across the 0.45–0.65 range. A Cauchy sampling weight above the upper bound would cause excessively high sampling dispersion and a sharp decline in effective sampling rate. Conversely, a weight below the range would result in insufficient long-range exploration capability, leaving the algorithm unable to navigate through complex obstacles.

All calibration experiments are implemented in MATLAB R2024a, with the average number of iterations and average planning time adopted as evaluation metrics. For each parameter configuration, 100 repeated trials are performed.



(a) Open scenario **(b) high-complexity maze scenario.**

Figure 8. Comparison of weight anchor schemes.

Fig. 8(a) shows the average iteration counts of the five candidate weight schemes in the open scenario, where Scheme A achieves the lowest average iteration count. Fig. 8(b) presents the average iteration counts of the five candidate weight schemes in the high-complexity scenario. The iteration count decreases rapidly from Scheme I to Scheme IV and reaches its minimum at Scheme IV. Finally, Scheme A (0.10, 0.80, 0.10) and Scheme IV (0.30, 0.10, 0.60) are selected as boundary anchor points, as listed in Table 1.

Scenario type	Complexity α	Scheme	λ_1	λ_2	λ_3
Fully open	0	A	0.10	0.80	0.10
Extremely complex	1	IV	0.30	0.10	0.60

Table 1. Anchor point parameters.

3.2.3 Dynamic Step Size Bounds

To obtain reasonable boundaries for the dynamic step size, calibration experiments are carried out using the single-variable control method. All other algorithm parameters are kept unchanged, 100 independent simulation runs are performed for each candidate parameter group, and the average number of iterations is selected as the comprehensive evaluation index.

The calibration of the upper step-size bound is conducted. The lower bound is fixed at 0.7, and five comparative experimental groups with values of 1.3, 1.4, 1.5, 1.6 and 1.7 are chosen to cover the effective operating range of the algorithm. If the upper bound is less than 1.3, the expansion step becomes overly conservative and the adjustable range of the two-factor is severely compressed. If the upper bound exceeds 1.7, the step size surpasses the safe expansion threshold and the planning performance degrades significantly.

The experimental results are shown in Fig. 9. As the coefficient increases from 1.3 to 1.5, the average number of iterations decreases continuously. Afterwards, the iteration count rises sharply

and then falls slightly. Therefore, the coefficient 1.5 achieves a balance between exploration efficiency and safety with optimal overall performance.

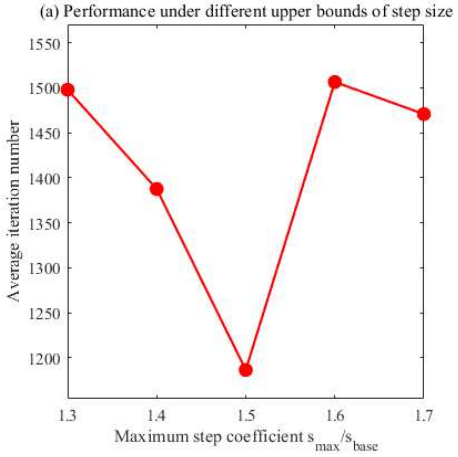


Figure 9. Algorithm performance under different step-size upper bounds.

Finally, calibration of the lower bound of step size is performed on the basis of the optimized upper bound. The upper bound is fixed at 1.5, and experimental groups with values of 0.5, 0.6, 0.7, 0.8 and 0.9 are selected. This interval also covers the valid lower operating range of the algorithm. When the lower bound exceeds 0.9, the minimum expansion step size becomes too large and the algorithm loses its fine-tuning capability. When the lower bound is less than 0.5, excessive step-size decay occurs, resulting in a substantial reduction in planning efficiency.

The experimental results are illustrated in Fig. 10. The overall curve exhibits a V-shaped profile with higher values at both ends and a trough in the middle. The minimum average iteration count is 0.7, which is thus identified as the optimal value for the lower bound.

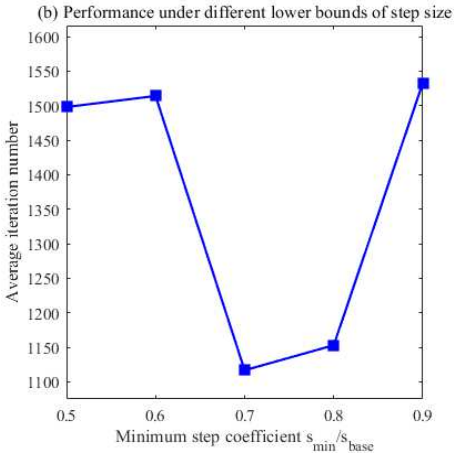


Figure 10. Algorithm performance under different step-size lower bounds.

Combined with the above calibration results, the upper bound of 1.5 and lower bound of 0.7 are finally selected for the dynamic step size in this work.

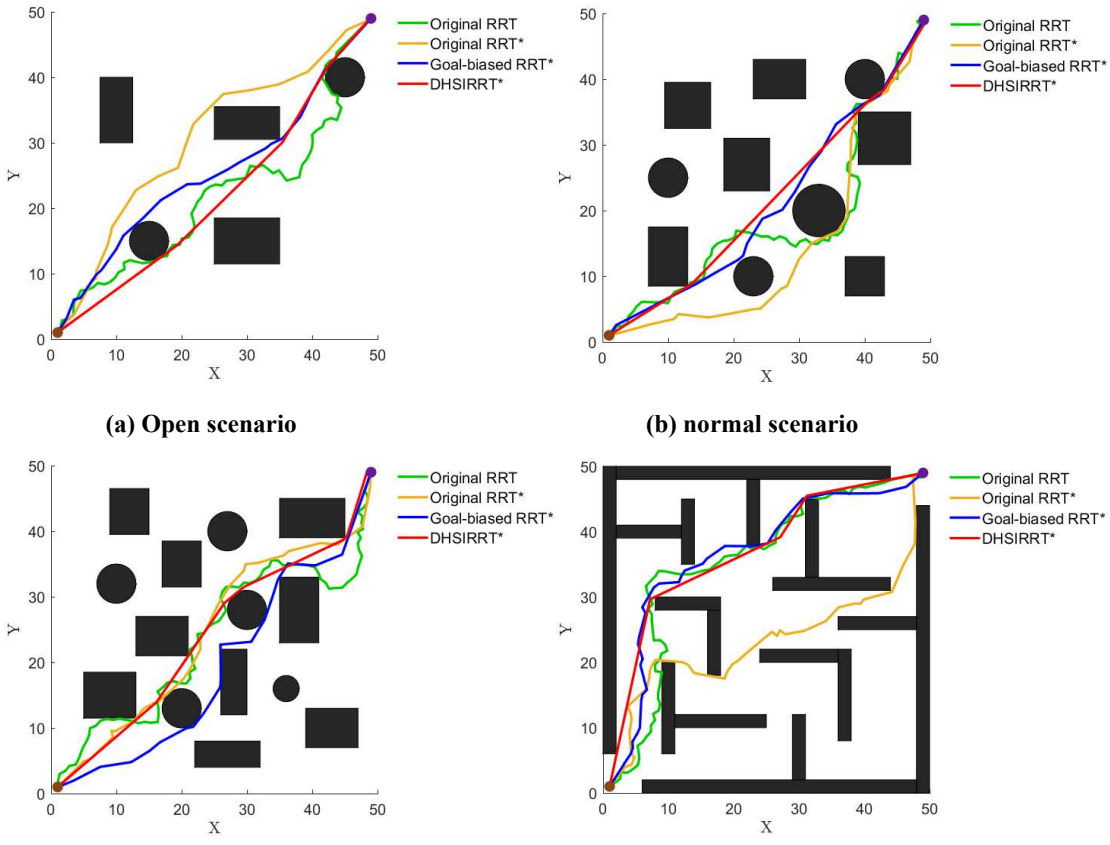
3.3 Results and Analysis

To verify the adaptability of the proposed algorithm in environments with varying complexity, four typical 2D scenarios, four 3D scenarios, and a physical manipulator workstation scenario, all covering open, normal, dense, and maze maps. All compared algorithms adopt identical basic parameters. The algorithm performance is evaluated via 100 independent repeated experiments carried out across the 2D, 3D and physical manipulator workstation scenarios of different complexity levels.

3.3.1 2D Simulation

Four 2D scenarios are designed in this work, with a map size of 50 × 50. The start point is set at (1,1) and the goal point at (49,49). The base step size for all four algorithms is set to 1, the maximum number of iterations is 5000, and the goal distance threshold is one time the base step size.

The experimental results are shown in Fig. 11. Black rectangles and circles denote obstacles, and brown and purple points represent the start and goal points, respectively. The green, orange, blue and red curve separately corresponds to the paths generated by the original RRT, the original RRT*, the goal-biased RRT* and the DHSIRRT* algorithm.

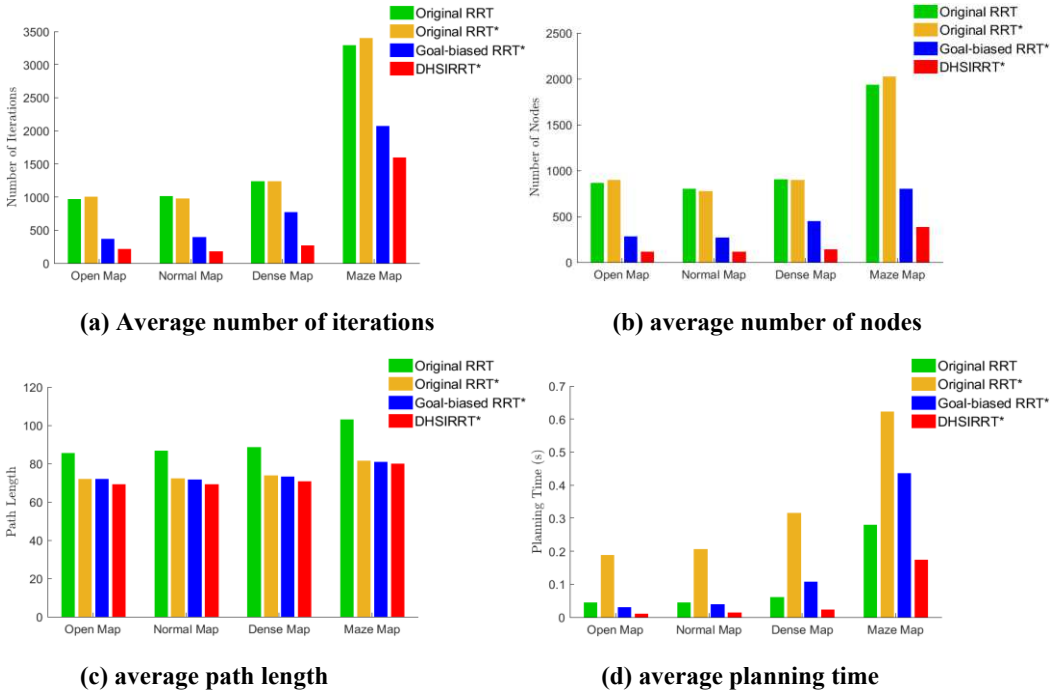


(a) Open scenario **(b) normal scenario**

(c) dense scenario **(d) maze scenario**

Figure 11. Comparison of simulation paths of the four algorithms in 2D space.

The path generated by the original RRT algorithm exhibits severe jitter and substantial detours in dense and maze environments. The original RRT* yields improved path smoothness compared with RRT, yet obvious ineffective detours remain in the maze scenario. Goal-biased RRT* produces paths closer to straight lines after introducing goal guidance, but detours still emerge in regions with dense obstacles. DHSIRRT* generates the smoothest paths that most closely approximate the straight line connecting the start and goal points across all four scenarios. Even in the maze environment, it can accurately avoid obstacles and find an approximately optimal feasible path.



(a) Average number of iterations **(b) average number of nodes**

(c) average path length **(d) average planning time**

Figure 12. Performance indicators of the four algorithms in 2D space.

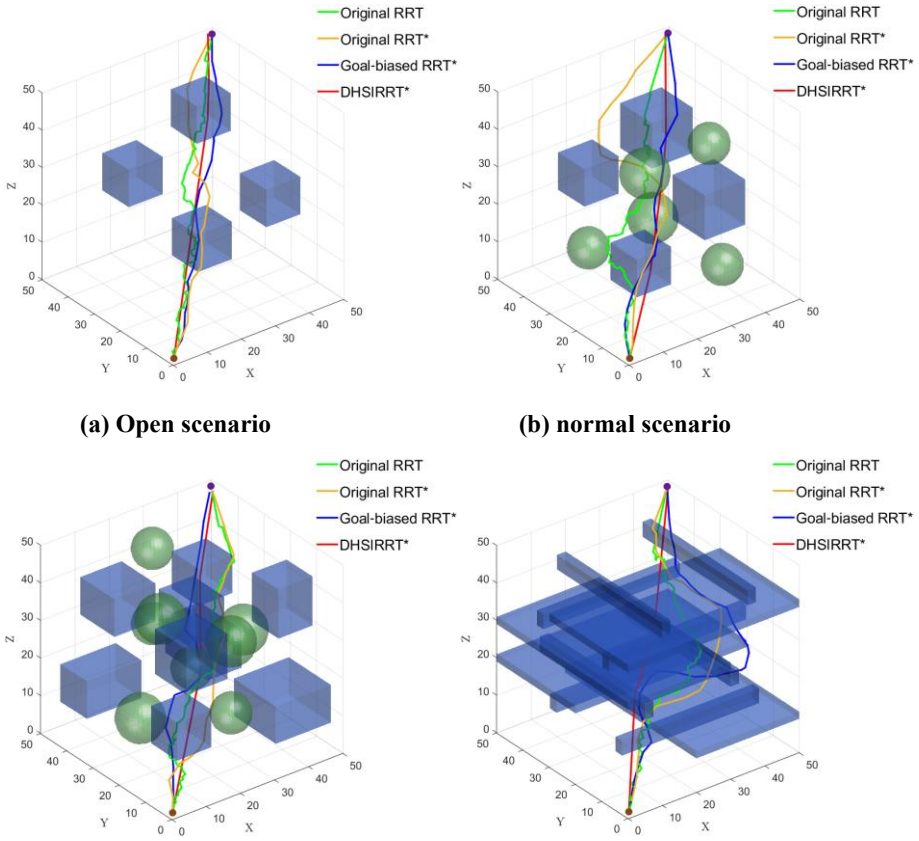
To further quantitatively compare the performance differences among various algorithms, comparative experiments are conducted in the four types of 2D simulation scenarios, where the original RRT, original RRT*, and goal-biased RRT* are adopted as benchmark algorithms for the DHSIRRT*. Figure 12 presents the comparison results of four core performance metrics: average iteration count, average node number, average path length, and average planning time. Compared with original RRT and original RRT*, the DHSIRRT* achieves substantial reductions in iteration count and node number, with maximum decreases of 81.4% and 86.6%, respectively. Relative to goal-biased RRT*, the maximum reductions in iteration count and node number reach 53.0% and 57.8%, accompanied by markedly reduced planning time.

In summary, the DHSIRRT* can simultaneously achieve higher convergence efficiency, lower computational cost and superior path quality in diverse complex 2D obstacle environments.

3.3.2 3D Simulation

To verify the performance of the proposed algorithm in high-dimensional space, four 3D industrial scenarios with a size of 50 × 50 × 50 are constructed to simulate the practical

working environment of a 6-degree-of-freedom manipulator. The start coordinate is (1,1,1) and the goal coordinate is (49,49,49). The DRRT* algorithm and other comparison algorithms are run for 100 independent trials within the same environment. The experimental visualizations and aggregated experimental data are shown in Fig. 13 and Fig. 14, respectively.

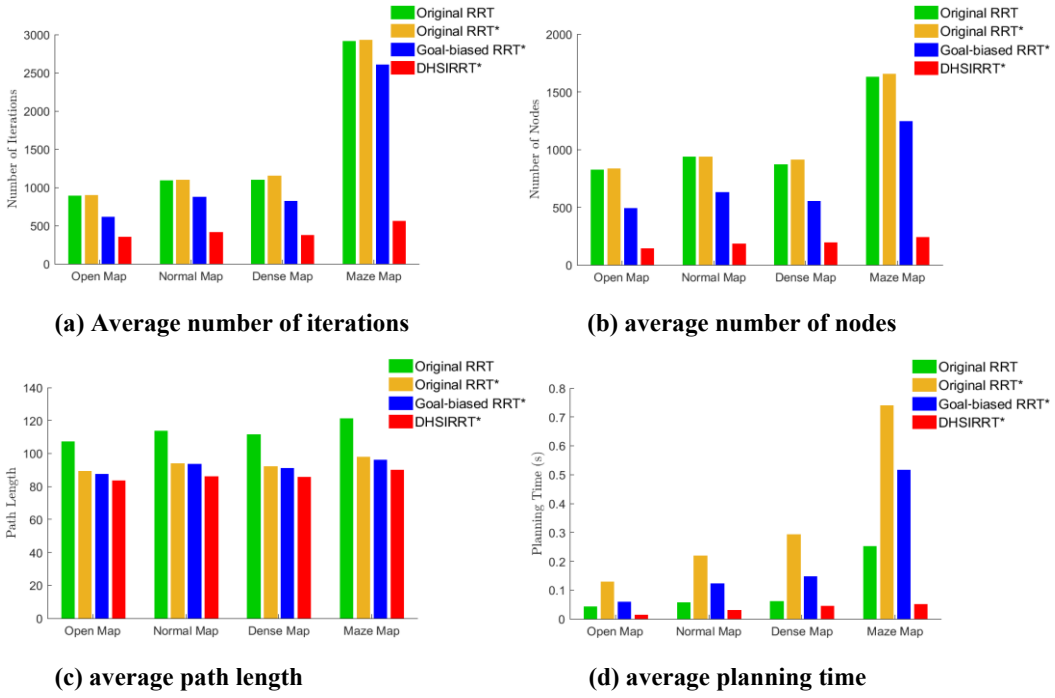


(a) Open scenario **(b) normal scenario**

(c) dense scenario **(d) maze scenario.**

Figure 13. 3D simulation results.

As the spatial dimension increases, the path tortuosity of all algorithms rises. The original RRT algorithm produces numerous irregular bends in the 3D space. Although the original RRT* optimizes path cost via the rewiring mechanism, its computational overhead increases rapidly and planning efficiency degrades noticeably. Goal-biased RRT* performs well in open scenarios but tends to be trapped by layered partitions in maze environments. Benefiting from dynamic step size and the heavy-tailed jumping property of Cauchy sampling, the DHSIRRT* algorithm can efficiently traverse layered corridors and narrow spaces, generating shorter and smoother feasible paths.



(a) Average number of iterations **(b) average number of nodes**

(c) average path length **(d) average planning time**

Figure 14. Performance indicators of the four algorithms in 3D space.

Comparative experiments are also carried out for the proposed algorithm across the four 3D simulation scenarios, with original RRT, original RRT*, and goal-biased RRT* adopted as benchmark algorithms. Figure 14 presents the comparison results of the four core performance metrics. Compared with the other benchmark algorithms, the DHSIRRT* markedly reduces the waste of computational resources caused by blind random sampling, and the performance improvement is comparable to that observed in 2D scenarios. In the 3D maze scenario, the average number of nodes of the proposed algorithm is reduced by 80.5% relative to goal-biased RRT*, demonstrating overall superior comprehensive performance over the benchmark algorithms.

Overall, the DHSIRRT* algorithm can effectively cope with complex obstacle distributions and narrow passage structures in 3D space, providing reliable technical support for path planning of 6-degree-of-freedom manipulators.

3.4 Manipulator Simulation

To verify the planning performance of the algorithm under manipulator constraints, a complete simulation verification platform is built in this section based on Ubuntu 22.04 and ROS 2 Humble. The experiments take the TCB610 six-axis industrial manipulator as the research object. The proposed improved algorithm is encapsulated as an independent planning node, and full-process visualization is realized via RViz2. A URDF link model is constructed according to the physical dimensions of the TCB610, and the motion limits of each joint are calibrated. Obstacles such as workstation cabinets and sundries are simplified using AABB bounding boxes and then published to ROS 2 topics. After finishing the planning task, the planning node generates

collision-free joint trajectories and publishes motion command topics. Meanwhile, visualization markers of the manipulator body, obstacles and planned trajectories are transmitted to RViz2.

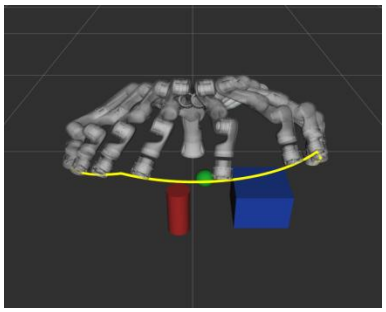
3.4.1 Experimental Setup

Four typical industrial workstation scenarios, namely open, normal, dense and maze, are configured in ascending order of environmental complexity. The initial joint angles of the manipulator and the target end-effector pose are unified across all experiments, and the maximum iteration number is set to 5000.

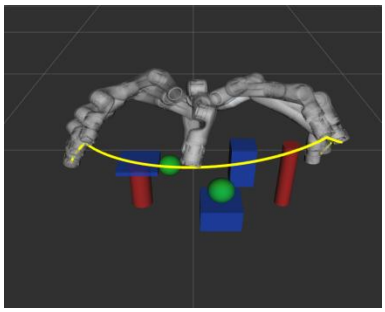
The experiments consist of two parts. The multi-environment adaptability of the improved algorithm is verified under the four scenarios. The dense scenario, which features the most complex constraints and best approximates real industrial working conditions, is selected for horizontal comparisons between the proposed algorithm and original RRT, original RRT*, and goal-biased RRT*. Each group of experiments is independently repeated 100 times, and the differences among algorithms are analyzed using three performance metrics: planning success rate, number of joint path segments, and motion planning time.

3.4.2 Multi-Scenario Performance

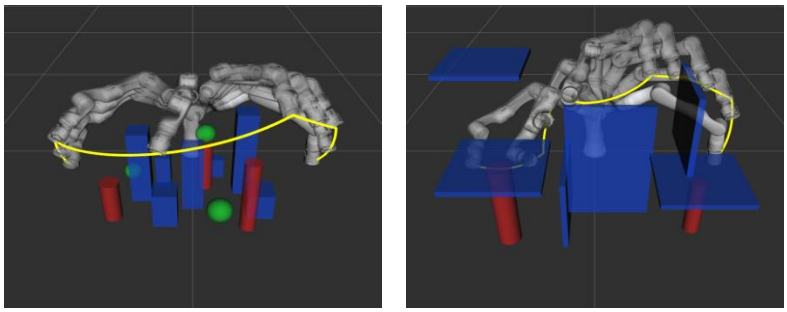
Simulation results across the four types of workstation scenarios demonstrate that the proposed algorithm can stably generate collision-free paths satisfying joint motion constraints in environments of varying complexity. In the open scenario (Fig. 15(a)) with sparse obstacles, the algorithm rapidly converges toward the target pose relying on high-weight Gaussian sampling. In the normal scenario (Fig. 15(b)), the algorithm autonomously balances global exploration and goal guidance, maintaining favorable path quality while avoiding obstacles. In the dense scenario (Fig. 15(c)), the algorithm achieves precise obstacle avoidance by increasing the Cauchy sampling weight and reducing the expansion step size. In the maze scenario (Fig. 15(d)), the algorithm leverages the heavy-tailed jumping property of Cauchy sampling to traverse narrow passages and reach the target pose.



(a) Open scenario



(b) normal scenario

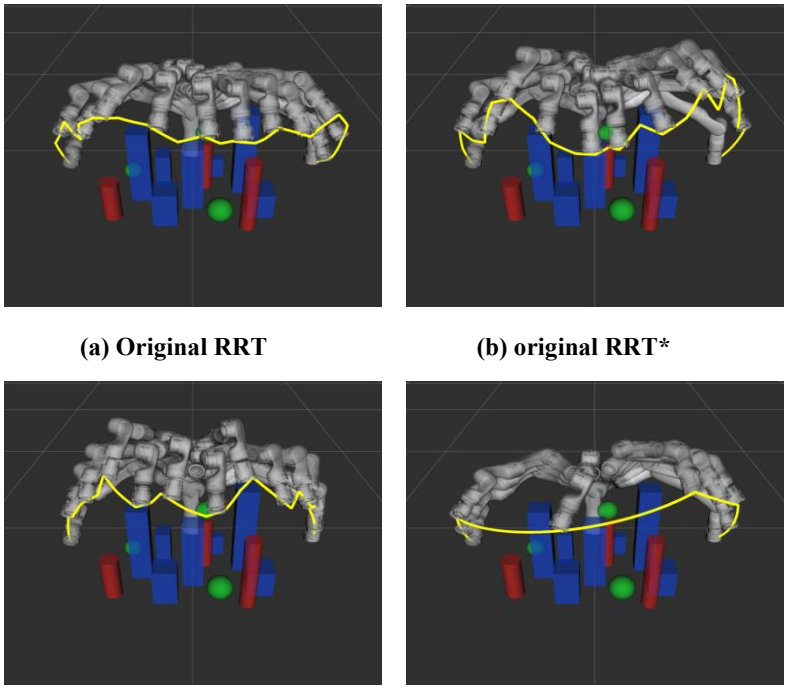


(c) dense scenario **(d) maze scenario.**

Figure 15. Simulation results of the four workstation scenarios.

3.4.3 Different Algorithms Comparison In Dense-Scenario

In dense scenarios, the feasible space is narrow and motion constraints are complex. Such conditions can fully reveal performance differences among various algorithms in terms of planning efficiency, path quality and operational reliability, and better approximate the complex working conditions of real industrial sites. Therefore, this scenario is selected for horizontal comparison of multiple algorithms. The end-effector motion trajectories of the four algorithms in this scenario are compared in Fig. 16.



(a) Original RRT **(b) original RRT***

(c) goal-biased RRT* **(d) DHSIRRT***

Figure 16. End-effector trajectories in the dense scenario.

It can be observed from the trajectory profiles that the original RRT algorithm produces end-effector trajectories with the highest tortuosity (Fig. 16(a)), featuring numerous irregular turns and redundant waypoints along with severe path jitter. The original RRT* optimizes path cost through the rewiring mechanism and improves the overall trajectory smoothness, yet obvious

detours still exist in densely obstructed areas (Fig. 16(b)). The goal-biased RRT* generates trajectories that are generally closer to the straight line between the start and goal points. However, it suffers from frequent local adjustments in cluttered obstacle environments and tends to backtrack after falling into local dead zones (Fig. 16(c)). In contrast, the trajectory generated by the proposed improved algorithm is smooth and continuous with a remarkably reduced number of turning points. The resulting path closely follows the shortest-path trend and achieves gentle obstacle avoidance without abrupt fluctuations, thereby delivering the optimal trajectory quality (Fig. 16(d)).

Metric	Original RRT	Original RRT*	Goal-biased RRT*	DHSIRRT*
Planning success rate (%)	68	56	63	99
Average path segments	33.08	28.20	22.31	5.05
Average planning time (s)	7.10	6.07	5.78	4.63

Table 2. Performance comparison in the dense scenario.

In the dense workstation scenario, 100 independent comparative experiments are conducted to validate the DHSIRRT* algorithm, with the original RRT, original RRT*, and goal-biased RRT* adopted as benchmark algorithms. The statistical results of three core performance indicators are presented in Table 2. Compared with the original RRT and RRT*, the DHSIRRT* significantly reduces computational resource consumption, achieving a maximum reduction of 84.7% in the number of joint path segments, a maximum shortening of 34.8% in motion planning time, and a maximum improvement of 43% in planning success rate. Furthermore, compared with the goal-biased RRT*, the improved algorithm reduces the number of joint path segments by 77.4% and the motion planning time by 19.9%, while increasing the planning success rate by 36%.

Overall, the DHSIRRT* algorithm can effectively cope with complex obstacle distributions and narrow working spaces in industrial workstations. It maintains excellent planning efficiency and path quality under the real motion constraints of manipulators, providing a reliable technical solution for practical path planning of 6-degree-of-freedom manipulators.

4. Conclusion

Aiming at the problems for exsiting RRT* algorithms including the lack of a unified adaptive mechanism for diverse map environments, low sampling efficiency, poor environmental adaptability under fixed step size expansion, and paths with redundant nodes and insufficient smoothness, DHSIRRT* algorithm was proposed by incorporating hybrid sampling, dynamic step sizing, and path optimization, simultaneously adapted to four typical map environments. The adaptability was validated across 2D grid maps, 3D space, and an ROS2-based manipulator simulation platform. Comparative experiments were conducted against RRT, RRT*, and goal-biased RRT*, with 100 independent runs in each scenario.

In both 2D and 3D environments, the proposed algorithm consistently outperformed goal-biased RRT* in terms of average iteration count, node number, and planning time. The maximum reductions in iteration count and node number reached 53.0% and 57.8%, respectively, and in the 3D maze scenario the average node number was further reduced by 80.5%. In the dense manipulator workstation scenario, it achieved a planning success rate of 99%, reduced the number of joint path segments by up to 84.7%, shortened the motion planning time by up to 34.8%, and improved the planning success rate by up to 43% compared with RRT and RRT*. Compared with goal-biased RRT*, it reduced the number of joint path segments by 77.4% and the planning time by 19.9%, while increasing the planning success rate from 63% to 99%. The experimental results demonstrated the effectiveness and robustness of the proposed algorithm. The DHSIRRT* algorithm can generate smooth, collision-free, and kinematically feasible paths across diverse map environments. Its superior adaptability and robustness provide a reliable foundation for the industrial deployment of the algorithm.

Future work will apply the algorithm to the physical TCB610 manipulator and integrate local path planning to further improve the dynamic obstacle avoidance capability of the algorithm.

Data availability

The data provided in this study can be obtained from the corresponding author Y.G.

References

1. Ding, Z.-J. et al. Variable sampling area RRT algorithm based on limiting joint angle for robotic arm path planning. *IEEE Access* **13**, 30665–30680 (2025).
2. Ku, X., Zhu, E. & Li, S. Obstacle avoidance path planning for robotic arms using a multi-strategy collaborative bidirectional RRT algorithm. *Sensors* **26**, 1376 (2026).
3. Josin, H. A. & Senthilnathan, R. Quicker path planning of a collaborative dual-arm robot using modified BP-RRT algorithm. Int. J. *Comput. Commun. Control* **19**, 6379 (2024).
4. Li, C., Huang, X., Ding, J., Song, K. & Lu, S. Global path planning based on a bidirectional alternating search A* algorithm for mobile robots. *Comput. Ind. Eng.* **168**, 108123 (2022).
5. Ahmad, J. & Ab Wahab, M. N. Enhancing the safety and smoothness of path planning through an integration of Dijkstra's algorithm and piecewise cubic Bezier optimization. *Expert Syst. Appl.* **289**, 128315 (2025).
6. Zhang, W., Wang, N. & Wu, W. A hybrid path planning algorithm considering AUV dynamic constraints based on improved A* algorithm and APF algorithm. *Ocean Eng.* **283**, 115333 (2023).
7. Chen, S., Yang, G., Cui, G., Yi, S. & Wu, L. Improved path planning and controller design based on PRM. *IEEE Access* **13**, 44156-44168 (2025).
8. Zhong, H., Cong, M., Wang, M., Du, Y. & Liu, D. HB-RRT: A path planning algorithm for mobile robots using Halton sequence-based rapidly-exploring random tree. *Eng. Appl. Artif. Intell*. **133**, 108362 (2024).
9. Alshammrei, S., Boubaker, S. & Kolsi, L. Improved Dijkstra algorithm for mobile robot path planning and obstacle avoidance. *Comput. Mater. Contin* **72**, 5939–5954 (2022).
10. Jin, J. et al. Path planning and fuzzy tracking method for push rake machine based on A* improvement. *High. Technol. Newsl*. **34**(09), 972–979 (2024).
11. Yang, Y. L. et al. AAPF*: a safer autonomous vehicle path planning algorithm based on the improved A* algorithm and APF algorithm. *Cluster Comput*. **27**, 11393–11406 (2024).
12. Kavraki, L. E., Švestka, P., Latombe, J.-C. & Overmars, M. H. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. Robot. Autom*. **12**, 566–580 (1996).
13. Kuffner, J. J. & LaValle, S. M. RRT-Connect: an efficient approach to single-query path planning. In *Proc. 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation* 995–1001 (IEEE, 2000).

14. Karaman, S. & Frazzoli, E. Sampling-based algorithms for optimal motion planning. *Int. J. Robot. Res*. **30**, 846–894 (2011).
15. LaValle, S. M. Rapidly-exploring random trees: a new tool for path planning. *Tech. Rep*. (Iowa State University, 1998).
16. Liang, X., Yu, F., Gao, W. & Yao, B. An improved RRT-based path planning approach with dynamic cone angle guidance for robotic manipulator obstacle avoidance. *Intell. Serv. Robo*t. **19**, 21 (2026).
17. Lei, S. et al. Research on improved RRT path planning algorithm based on multi-strategy fusion. *Sci. Rep*. **15**, 92675 (2025).
18. Dong, X. et al. FHQ-RRT: an improved path planning algorithm for mobile robots to acquire high-quality paths faster. *Sensors* **25**, 2189 (2025).
19. Wang, B., Ju, D., Xu, F. & Feng, C. Bi-RRT: an improved bidirectional RRT path planner for robot in 2D space. *IEEJ Trans. Electr. Electron. Eng*. **18,** 1639–1652 (2023).
20. Gammell, J. D., Srinivasa, S. S. & Barfoot, T. D. Informed RRT*: optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems* 2997–3004 (IEEE, 2014).
21. Cao, M., Mao, H., Tang, X., Sun, Y. & Chen, T. A novel RRT*-Connect algorithm for path planning on robotic arm collision avoidance. *Sci. Rep*. **15**, 87113 (2025).
22. Chen, Z. et al. A path-planning method considering environmental disturbance based on VPF-RRT*. *Drones* **7**, 145 (2023).
23. Fang, C. et al. Path planning for dragon-fruit-harvesting robotic arm based on XN-RRT* algorithm. *Sensors* **25,** 2773 (2025).
24. Li, H., Kang, Y. & Han, H. Dynamic informed bias RRT-Connect: improving heuristic guidance by dynamic informed bias using hybrid dual trees search. *J. Intell. Robot. Syst*. **110**, 106 (2024).
25. Huang, Y., Jiang, W. & Xu, S. A multi strategy bidirectional RRT algorithm for efficient mobile robot path planning. *Sci. Rep*. **15**, 13915 (2025).
26. Wang, Y. et al. Path planning of RRT algorithm with subregional dynamic probabilistic sampling based on artificial potential field in radiation environments. *Nucl. Eng. Technol*. **57,** 103706 (2025).
27. Wang, H., Lai, H., Du, H. & Gao, G. IBPF-RRT: an improved path planning algorithm with ultra-low number of iterations and stabilized optimal path quality. *J. King Saud Univ. Comput. Inf. Sci*. **36**, 102146 (2024).

Funding

The work was supported by the Natural Science Foundation with Innovation and Development Joint Foundation of Shandong Province (No. ZR2026LZJ004), the project of the Natural Science Foundation of Shandong Province, China (No. ZR2024ME210, ZR2026QC0763), and Shandong Province Science and Technology SMES Innovation Ability Enhancement Project (No. 2024TSGC0268).

Author contributions

Conceptualization, Y.G.; Methodology, Y.G. and Y.C.; Software, Y.C. and Q.Ch.; Validation, Y.G. and Q.Ch.; Formal analysis, X.Y.; Writing - original draft preparation, Q.C. and C.Y.; Writing - review and editing, C.Y.; All authors have read and agreed to the published version of the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to Y.G.