PATH PLANNING FOR AGRICULTURAL PICKING MANIPULATORS USING A GRID-NUMBER-GUIDED IMPROVED RRT ALGORITHM
/
基于栅格编号引导的改进 RRT 算法的农业采摘机械臂路径规划

Wei ZHAO*[1)], Yang PAN[1)], Bangbo LIU[1)], Xi XU[1)], Tianle SHI[1)], Weijian SU[1)], Xiaobiao SHANG[1)], Hao PENG[1)], Hongfu ZHANG*[1)]
College of Mechanical and Electrical Engineering, Yunnan Agricultural University, Kunming，Yunnan / China
Tel: +86 184 8716 6782; E-mail: ynau_jd@163.com
Corresponding authors: Hongfu ZHANG
DOI: https://doi.org/10.35633/inmateh-79-57

Keywords: *picking manipulator; path planning; Optimized-RRT; grid-number-guided sampling; dynamic sampling region; trajectory smoothing*

ABSTRACT
Irregular branches in agricultural picking environments require rapid collision-free path planning and smooth manipulator motion. This study developed an improved RRT algorithm guided by grid numbering, termed Optimized-RRT, based on a unified indexed-set representation of numbered primary and backup free-grid regions. Within this framework, dynamic region adjustment, target biasing, node occupancy, and local burst expansion jointly guide the search, while greedy pruning and quintic polynomial interpolation are applied to post-process the resulting path. Unlike geometric-region-based methods that generate coordinate samples within an updated sampling window, Optimized-RRT selects a valid grid number from the active set and maps it directly to the corresponding state. Compared with RRT, GB-RRT, and Informed-RRT*, the complete algorithm reduced planning time, the number of iterations, and the final number of path nodes by 40.61%–94.32%, 57.51%–88.66%, and 89.29%–94.59%, respectively, in two-dimensional scenarios, and by 88.95%–98.92%, 79.26%–94.67%, and 67.31%–90.83%, respectively, in three-dimensional scenarios. The reduction in the number of nodes resulted from the combined effects of guided search and path pruning. Manipulator simulations and a demonstration on a single experimental platform confirmed the executability of the generated trajectories under the tested conditions.*

摘要
*农业采摘环境中的不规则枝条要求机械臂快速生成无碰撞路径并实现平稳运动。本文提出基于统一索引集合表示的栅格编号引导改进 RRT 算法（Optimized-RRT），将主、备用自由栅格区域表示为可直接寻址的编号集合。在这一表示下，动态区域调整、目标偏置、节点占用和局部突发扩展协同引导搜索，贪心剪枝和五次多项式插值用于所得路径的后处理。不同于在更新后的几何区域中生成坐标样本的方法，Optimized-RRT 从当前活动集合中选择有效栅格编号，并将其直接映射为相应状态。相较于 RRT、GB-RRT*和 Informed-RRT*，完整算法流程在二维场景中将规划时间、迭代次数和最终路径节点数分别降低 40.61%-94.32%、57.51%-88.66%和 89.29%-94.59%，在三维场景中分别降低 88.95%-98.92%、79.26%-94.67%和 67.31%-90.83%。节点数的降低源于引导搜索与剪枝的共同作用。机械臂仿真和单一平台演示验证了测试条件下轨迹的可执行性。*

INTRODUCTION
Labor shortages and the need for consistent harvesting efficiency have accelerated the development of agricultural robots and autonomous fruit- and vegetable-picking systems *(Cheng et al., 2023; Chen et al., 2024)*. In orchards and greenhouses, irregular branches and canopy structures form unstructured obstacles that complicate manipulator motion *(Karim et al., 2025)*. A picking manipulator must therefore generate a collision-free path to the target pose with sufficient planning speed and motion smoothness.

Deterministic graph-search methods such as A* and D* can become computationally demanding as the state dimension and map complexity increase *(Xuan et al., 2025)*, whereas population-based optimization may converge too slowly for online planning *(Zhu and Pan, 2024)*. Sampling-based motion planners avoid exhaustive state-space enumeration and are therefore well suited to complex, high-dimensional planning problems *(Orthey et al., 2024*). In agricultural applications, improved RRT variants have been used for grape-picking manipulators *(Hu et al., 2024)*, adaptive-step planning of tea-picking arms *(Li et al., 2024)*, and RGB-D-assisted Bi-RRT planning for orchard harvesting manipulators *(Liu et al., 2024)*.

These studies establish the application relevance of RRT-based planning, while conventional RRT still remains affected by random exploration, repeated expansion, and non-smooth paths.

Region-restricted and goal-guided sampling methods are the closest prior work to the present study. RRT* established parent reselection and rewiring as a foundation for asymptotically optimal sampling-based planning *(Karaman and Frazzoli, 2011)*, while Informed-RRT* restricts post-solution sampling to an admissible ellipsoidal subset *(Gammell et al., 2014)*. *Feng et al. (2023)* combined map-dependent step size and bias probability with dynamic sampling-region updates, sampling-point optimization, and node reconnection. For manipulators, *Shang et al. (2024)* developed the RRT*-DR algorithm for obstacle-avoidance planning, and *Xiao et al. (2024)* proposed FBi-RRT with heuristic node expansion. *He et al. (2025)* further combined ellipsoidal subset sampling and goal-biased sampling with adaptive step size, node rejection, pruning, and spline smoothing. These representative methods improve efficiency by restricting the geometric sampling domain, biasing expansion toward promising states, optimizing tree connections, or combining these mechanisms.

However, geometric-region restriction and target bias do not by themselves provide a common representation for active-region membership, alternative-region switching, and repeated-cell suppression. A target-biased planner changes how often the goal is selected, but its remaining samples still require a defined sampling domain. Likewise, a dynamically updated geometric window identifies where coordinates may be generated but does not necessarily make the valid free states directly addressable. The remaining engineering need is therefore a representation that coordinates region restriction and sampling control without presenting these established strategies as individually new.

Optimized-RRT addresses this need by representing the free cells of the main and backup regions as disjoint, directly addressable sets of grid numbers. A non-goal sample is obtained by selecting a number from the active set and mapping it directly to the corresponding cell-center state; dynamic restriction is implemented by set intersection, backup-region switching preserves an alternative sampling source, and node occupancy suppresses revisits at the selected grid resolution. Target bias therefore operates within the indexed-set representation rather than replacing region management. This differs from methods that primarily update a continuous sampling window or modify the probability or direction of goal-oriented sampling.

Accordingly, the primary contribution of this study lies in the indexed-set representation and the coordinated search mechanisms that it enables, rather than in the individual application of dynamic region adjustment, target biasing, pruning, or interpolation. The proposed representation supports direct grid-number-to-state mapping generalized to *d*-dimensional spaces and is integrated with local burst expansion, greedy pruning, and quintic polynomial interpolation within a unified search-to-trajectory pipeline for agricultural picking manipulators. The evaluation included repeated 2D and 3D simulations, sensitivity and ablation analyses, manipulator simulations, and a platform-based feasibility demonstration. No new results concerning asymptotic optimality or probabilistic completeness are claimed.

MATERIALS AND METHODS

Kinematic Modeling of the Picking Manipulator

Model of the Picking Manipulator and Modified D–H Parameters

A six-axis collaborative manipulator was used. Its model and modified D–H coordinate system are shown in Fig. 1, and the corresponding parameters are listed in Table 1.

a) *b)*

Fig. 1 – Six-axis collaborative manipulator: (a) manipulator model; (b) modified D–H coordinate system

Table 1

Modified D–H parameters of the manipulator

i	α_{i-1} [°]	a_{i-1} [m]	Joint variable θ_i	d_i [m]
1	0	0	θ_1	0.187
2	90	0	θ_2	0.006
3	0	0.210	θ_3	0
4	-90	0	θ_4	0.2105
5	90	0	θ_5	0
6	-90	0	θ_6	0.1593

Based on the modified D–H convention, the homogeneous transformation matrix from frame i to frame $i-1$ is expressed as:

$$\mathbf{T}_i^{i-1} = \begin{bmatrix} c_{\theta_i} & -s_{\theta_i} & 0 & a_{i-1} \\ s_{\theta_i}c_{\alpha_{i-1}} & c_{\theta_i}c_{\alpha_{i-1}} & -s_{\alpha_{i-1}} & -d_i s_{\alpha_{i-1}} \\ s_{\theta_i}s_{\alpha_{i-1}} & c_{\theta_i}s_{\alpha_{i-1}} & c_{\alpha_{i-1}} & d_i c_{\alpha_{i-1}} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

In Eq. (1), $c_{\theta_i} = \cos\theta_i$ and $s_{\theta_i} = \sin\theta_i$; the same notation applies to α_{i-1}. The end-effector pose relative to the base frame is obtained by successively multiplying the six adjacent-frame transformation matrices:

$$\mathbf{T}_6^0 = \mathbf{T}_1^0\mathbf{T}_2^1\mathbf{T}_3^2\mathbf{T}_4^3\mathbf{T}_5^4\mathbf{T}_6^5 = \prod_{i=1}^{6}\mathbf{T}_i^{i-1} \quad (2)$$

Collision Detection Modeling

For efficient collision checking, manipulator links were approximated by cylinders and irregular obstacles by spherical envelopes *(Jiang L. et al., 2022; Shi et al., 2022; Wang H. et al., 2025)*. As shown in Fig. 2, the link radius, obstacle radius, and an additional safety margin were combined into a conservative collision radius.

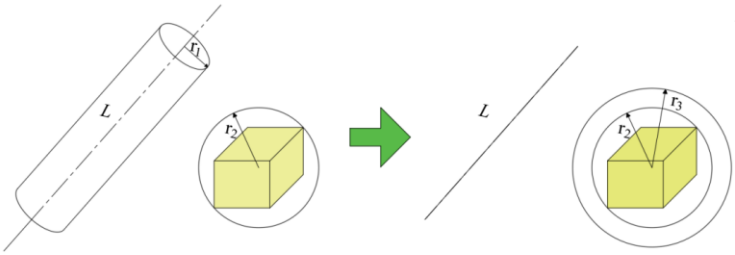


Fig. 2 – Conservative geometric simplification for collision detection

As shown in Fig. 3, each link segment was discretized into N equal subintervals. The point locations, point-to-obstacle distances, conservative collision radius, and collision criterion are defined as:

$$\begin{gathered} \mathbf{P}_i = \mathbf{P}_s + \frac{i}{N}(\mathbf{P}_e - \mathbf{P}_s), \quad i = 0,1,\ldots,N \\ d_i = \|\mathbf{P}_i - \mathbf{C}_o\|_2 = \sqrt{(x_i - x_c)^2 + (y_i - y_c)^2 + (z_i - z_c)^2} \\ r_c = r_l + r_o + \delta_s \\ \min_{0\le i\le N} d_i \le r_c \end{gathered} \quad (3)$$

In Eq. (3), $\mathbf{P}_s$ and $\mathbf{P}_e$ are the start and end points of the link segment, $\mathbf{P}_i$ is its i-th discrete point, and N is the number of equal subintervals. $\mathbf{C}_o = (x_c, y_c, z_c)^{\mathrm{T}}$ is the obstacle-center coordinate, while r_l, r_o, and δ_s denote the link-envelope radius, obstacle-envelope radius, and additional safety margin, respectively.

A collision was identified when the minimum discrete-point distance was not greater than the conservative collision radius. N was selected link-wise to keep adjacent checking points close relative to the conservative envelope. This discrete approximation improves efficiency but does not constitute certified continuous-time clearance; the safety margin represents modeling and localization uncertainty rather than a measured minimum clearance.

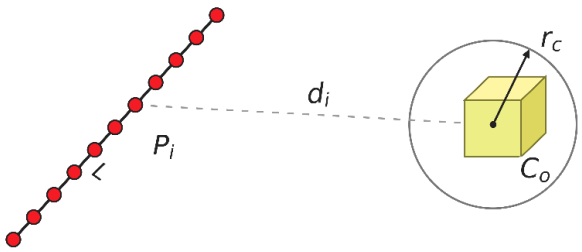


Fig. 3 – Discrete point-to-sphere collision detection along a link segment

Strategy of the Proposed Optimized-RRT Algorithm

Grid Map Construction

The planning map was discretized into fixed cells. A cell was marked as occupied when it intersected an obstacle; otherwise, it was treated as free. The complete grid set is defined as:

$$\mathcal{C} = \{c_{ij} \mid i = 1,2,\dots,n_r;\ j = 1,2,\dots,n_c\}, \qquad N_g = n_r n_c \tag{4}$$

In Eq. (4), c_{ij} denotes the cell in row i and column j, n_r and n_c are the numbers of rows and columns, and $N_g = n_r n_c$ is the total number of grid cells. The obstacle set is defined as:

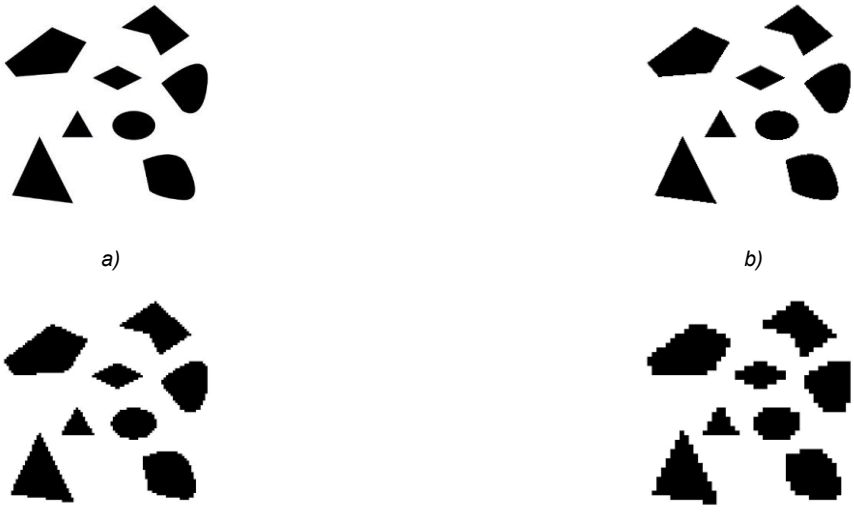
$$\mathcal{O} = \{o_k \mid k = 1,2,\dots,N_o\} \tag{5}$$

In Eq. (5), o_k denotes the k-th obstacle and N_o is the total number of obstacles. The occupancy state of each grid cell is described by the Boolean function:

$$\begin{aligned} f(c_{ij}) &= 1, \quad \text{if } \exists\, o_k \in \mathcal{O}:\ \Omega(c_{ij}) \cap \Omega(o_k) \neq \varnothing \\ f(c_{ij}) &= 0, \quad \text{otherwise} \end{aligned} \tag{6}$$

In Eq. (6), $\Omega(c_{ij})$ and $\Omega(o_k)$ denote the physical regions occupied by grid cell c_{ij} and obstacle o_k, respectively. A value of $f(c_{ij}) = 1$ identifies an obstacle cell, whereas $f(c_{ij}) = 0$ identifies a free cell.

Grid resolution controls the trade-off between obstacle-boundary fidelity and computational cost: smaller cells preserve geometry more accurately but increase storage and processing requirements. Fig. 4 compares side lengths of 2, 5, and 10 pixels. The precision retention ratio is the original obstacle area divided by the occupied grid-cell area; lower values indicate greater obstacle expansion after discretization.



a) *b)*

c) *d)*

Fig. 4 – Grid discretization at different resolutions: (a) original image; (b) 2 pixels; (c) 5 pixels; (d) 10 pixels

Table 2

Grid discretization data at different cell sizes

Grid side length [pixels]	Time [s]	Precision retention ratio
2	0.0509	0.97
5	0.0087	0.90
10	0.0027	0.77

As the grid side length increased from 2 to 10 pixels, processing time decreased from 0.0509 to 0.0027 s, while the precision retention ratio decreased from 0.97 to 0.77 (Table 2). The image-based example in Fig. 4 and Table 2 illustrates the general trade-off between grid resolution and processing efficiency. In the subsequent path-planning experiments, a grid side length of 10 mm was adopted as the baseline setting.

Grid-Number-Guided Sampling

Conventional restricted-region sampling defines a geometric domain and then generates coordinates within it; depending on the representation, samples may still fall outside the active free subset or revisit previously explored cells *(Oh et al., 2025; He et al., 2025).* Optimized-RRT instead divides the free workspace into a main sampling region and a backup sampling region (Fig. 5) and represents both regions using disjoint sets of valid grid numbers. The main set concentrates non-goal sampling toward the target, whereas the backup set preserves alternative routes when expansion in the main region becomes inefficient. Target-biased samples are introduced within this set-based region-management framework.

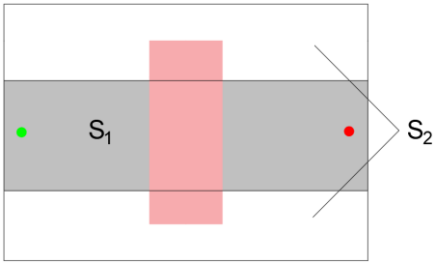


Fig. 5 – Division of the main and backup sampling regions

Each grid cell is assigned a unique number (Fig. 6). Sampling is performed by selecting a number from the active region set and mapping that number to the corresponding cell-center coordinate.

1	8	15	22	29	36	43	50	57	64
2	9	16	23	30	37	44	51	58	65
3	10	17	24	31	38	45	52	59	66
4	11	18	25	32	39	46	53	60	67
5	12	19	26	33	40	47	54	61	68
6	13	20	27	34	41	48	55	62	69
7	14	21	28	35	42	49	56	63	70

Fig. 6 – Grid numbering used for region-guided sampling

Using column-major numbering, each grid cell is assigned a unique number and the free-grid sampling set is defined as:

$$g_{ij} = (j-1)n_r + i, \qquad \mathcal{G}_f = \{g_{ij} \mid f(c_{ij}) = 0\} \tag{7}$$

In Eq. (7), $g_{ij} = (j-1)n_r + i$ is the unique number of the cell in row i and column j, and $\mathcal{G}_f$ contains only free-grid numbers. The main and backup sampling sets are defined as:

$$\begin{gathered} \mathcal{A} = \{g \in \mathcal{G}_f \mid \mathbf{x}(g) \in \Omega_{\text{main}}\} \\ \mathcal{B} = \{g \in \mathcal{G}_f \mid \mathbf{x}(g) \in \Omega_{\text{backup}}\} \\ \mathcal{A} \cup \mathcal{B} = \mathcal{G}_f, \qquad \mathcal{A} \cap \mathcal{B} = \varnothing \end{gathered} \tag{8}$$

In Eq. (8), $\mathbf{x}(g)$ maps grid number g to its cell-center coordinate, while Ω_{main} and Ω_{backup} denote the main and backup spatial regions. The two sets form a disjoint partition of $\mathcal{G}_f$, so direct number selection always produces a sample in the active free-grid region.

The following switching rule determines the active sampling region:

$$\begin{aligned} S_{k+1} &= S_2, \quad S_k = S_1, \quad \frac{N_1^{\text{succ}}}{\max\left(N_1^{\text{fail}}, 1\right)} < T_{\text{eff}} \\ S_{k+1} &= S_1, \quad S_k = S_2, \quad N_2^{\text{succ}} > \text{floor}\left(\frac{L}{10\Delta}\right) \\ S_{k+1} &= S_k \end{aligned} \tag{9}$$

In Eq. (9), S_k is the active region before the k-th switching decision. N_1^{succ} and N_1^{fail} are the numbers of successful and failed node-generation attempts in the main region S_1, respectively. T_{eff} is the region-switching threshold. N_2^{succ} is the number of valid nodes generated in the backup region S_2, L is the start-to-target distance, and Δ is the fixed expansion step.

After each switch, the counters are reset to avoid overly frequent alternation. When the valid-to-invalid expansion ratio in S_1 falls below T_{eff}, sampling is transferred to S_2; after sufficient valid expansion in S_2, the algorithm returns to S_1.

Local Burst Expansion Sampling Strategy

Local burst expansion provides two candidate samples in one iteration. The first is the center of a randomly selected grid; the second is generated within its neighboring-grid region (Fig. 7).

1	7	13	19	25	31
2	8	14	20	26	32
3	9	15	21	27	33
4	10	16	22	28	34
5	11	17	23	29	35
6	12	18	24	30	36

a)

1	7	13	19	25	31
2	8	14	20	26	32
3	9	15	21	27	33
4	10	16	22	28	34
5	11	17	23	29	35
6	12	18	24	30	36

b)

Fig. 7 – Local burst expansion: (a) primary grid selection; (b) neighboring-grid sampling region

If the primary candidate fails to generate a valid node, a second candidate is sampled from the neighboring-grid region defined by:

$$\mathcal{R}_{\text{local}} = \{(x, y) \in \Omega_{\text{free}} \mid |x - x_c| \le d,\ |y - y_c| \le d\} \tag{10}$$

In Eq. (10), (x_c, y_c) is the center of the primary grid and d is the neighborhood half-width. For a one-cell neighborhood, d equals the grid side length. The second candidate reduces iterations lost when the first candidate is invalid.

Dynamic Sampling Region Strategy

The active sampling region is dynamically narrowed as the tree approaches the target. When a newly generated node is closer to the target than all existing nodes, its grid number and the target grid number define the updated sampling interval. A narrowly updated region may stagnate near an obstacle (Fig. 8(a)). The interval is therefore expanded on both sides to preserve alternative search directions (Fig. 8(b)).

a) b)

Fig. 8 – Dynamic sampling region: (a) narrowed region; (b) expanded region

The dynamically expanded free-grid interval is defined as:

$$\begin{aligned} g_{\min} &= \max(1, \min(u, v) - f n_r) \\ g_{\max} &= \min\left(N_g, \max(u, v) + f n_r\right) \\ \mathcal{G}' &= \left\{g \in \mathcal{G}_f \mid g_{\min} \le g \le g_{\max}\right\} \end{aligned} \tag{11}$$

In Eq. (11), u and v are the grid numbers of the target and the current node closest to the target, respectively; f is the number of columns added on each side; n_r is the number of grid cells per column; and N_g is the total number of grid cells. The operators $\min(u, v)$ and $\max(u, v)$ ensure that the interval remains valid regardless of the numbering direction. The partition sets are then updated as:

$$\mathcal{A}' = \mathcal{A} \cap \mathcal{G}', \qquad \mathcal{B}' = \mathcal{B} \cap \mathcal{G}' \tag{12}$$

Equation (12) intersects the original main and backup sets with the dynamically expanded free-grid set. This preserves the original spatial partition while removing grid numbers outside the active interval.

Extension to Three-Dimensional and Joint Spaces

For a d-dimensional state $\mathbf{q} = (q_1, \ldots, q_d)$, let $q_{k,\min}$ and $q_{k,\max}$ be the bounds of dimension k, h_k its cell width, n_k the number of cells, and $i_k = \lfloor \frac{q_k - q_{k,\min}}{h_k} \rfloor$ its zero-based index. The mixed-radix grid number is $g(\mathbf{q}) = 1 + \sum_{k=1}^{d} i_k \prod_{j=1}^{k-1} n_j$; the inverse state is recovered by integer division and remainder operations followed by cell-center mapping. Equation (7) is the $d = 2$ case; $d = 3$ was used for Cartesian simulation and $d = 6$ for the manipulator configuration space, with joint bounds obtained from the URDF model.

Main and backup sets contain free grid numbers whose centers satisfy the corresponding region definitions, and a local neighborhood satisfies $|i_k - i_{k,c}| \le r_k$ in every dimension. Number/state mapping costs $O(d)$, whereas the potential grid count $\prod_{k=1}^{d} n_k$ grows exponentially with dimension, so memory and behavior depend on resolution. Node occupancy suppresses revisits at cell resolution but may discard distinct configurations within one cell. Accordingly, no probabilistic-completeness or resolution-independent scalability claim is made.

Node occupancy Strategy

Each accepted node marks its grid cell as occupied (Fig. 9). A candidate falling in an occupied cell is discarded, reducing repeated visits and redundant computation.

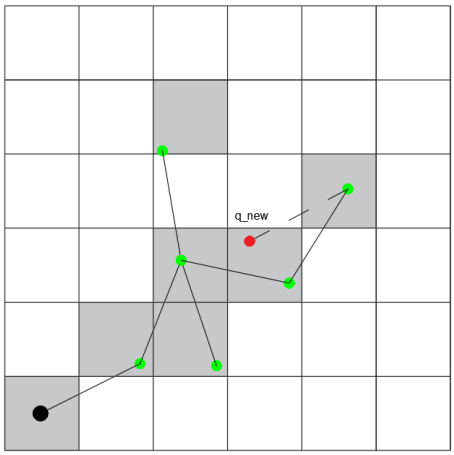


Fig. 9 – Node occupancy strategy

Target-Biased Sampling Strategy

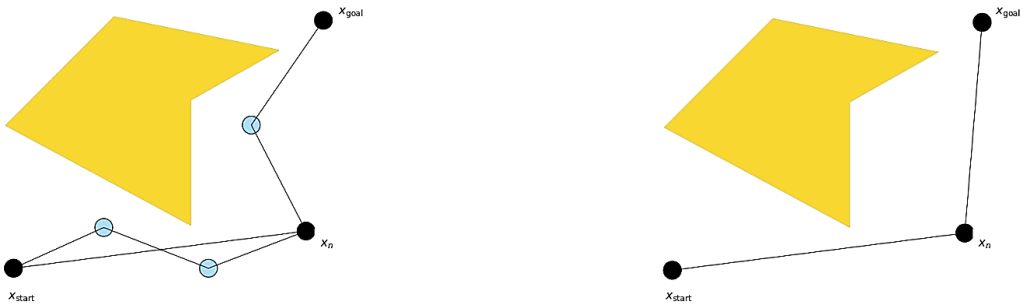
Target-biased sampling improves the directional efficiency of tree expansion. The sampling rule is defined as follows:

$$\begin{aligned} \mathbf{x}_{\text{sample}} &= \mathbf{x}_{\text{goal}}, \quad \xi \le p_g \\ \mathbf{x}_{\text{sample}} &= \mathbf{x}(g), \quad \xi > p_g, \quad g \sim U(\mathcal{G}_{\text{active}}) \\ \xi &\sim U(0,1) \end{aligned} \tag{13}$$

In Eq. (13), ξ is uniformly distributed over $[0,1]$, p_g is the target-bias probability, $\mathcal{G}_{\text{active}}$ is the currently active grid-number set, and $\mathbf{x}(g)$ maps a selected grid number to its cell-center coordinate.

Greedy Pruning Strategy

The raw path may contain redundant intermediate nodes. Greedy pruning tests direct collision-free connections from the current node to later nodes and removes all bypassed nodes (Fig. 10).



a) *b)*

Fig. 10 – Greedy pruning: (a) raw path; (b) pruned path

Quintic Polynomial Interpolation for Trajectory Smoothing

A quintic polynomial was applied independently to each joint and each consecutive pruned path segment to obtain continuous joint position, velocity, and acceleration under specified endpoint conditions *(Xu et al., 2023)*:

$$\theta_j^{(s)}(t) = a_{0,j}^{(s)} + a_{1,j}^{(s)}t + a_{2,j}^{(s)}t^2 + a_{3,j}^{(s)}t^3 + a_{4,j}^{(s)}t^4 + a_{5,j}^{(s)}t^5, \quad 0 \le t \le T_s \tag{14}$$

For segment s, the start and end positions are defined by consecutive joint-space waypoints, while the endpoint velocity and acceleration are set to zero. The polynomial coefficients are obtained from:

$$\mathbf{a}_j^{(s)} = \mathbf{M}(T_s)^{-1}\mathbf{b}_j^{(s)}$$

$$\mathbf{M}(T_s) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 1 & T_s & T_s^2 & T_s^3 & T_s^4 & T_s^5 \\ 0 & 1 & 2T_s & 3T_s^2 & 4T_s^3 & 5T_s^4 \\ 0 & 0 & 2 & 6T_s & 12T_s^2 & 20T_s^3 \end{bmatrix} \tag{15}$$

$$\mathbf{b}_j^{(s)} = \left[\theta_j^{(s)}(0) \quad \dot{\theta}_j^{(s)}(0) \quad \ddot{\theta}_j^{(s)}(0) \quad \theta_j^{(s)}(T_s) \quad \dot{\theta}_j^{(s)}(T_s) \quad \ddot{\theta}_j^{(s)}(T_s)\right]^{\mathrm{T}}$$

In Eq. (15), $\mathbf{a}_j^{(s)} = \left[a_{0,j}^{(s)}, a_{1,j}^{(s)}, \ldots, a_{5,j}^{(s)}\right]^{\mathrm{T}}$ is the coefficient vector for joint j in segment s, T_s is the segment duration, and $\mathbf{b}_j^{(s)}$ contains the prescribed endpoint position, velocity, and acceleration. In this study, the endpoint velocities and accelerations were set to zero.

Quintic interpolation was applied segment by segment after pruning to produce the final executable joint trajectory.

Workflow of Optimized-RRT

Fig. 11 summarizes the complete pipeline: initialization and d-dimensional grid numbering are followed by region-set construction, grid-number sampling, collision and occupation checks, tree extension, region updating, path backtracking, greedy pruning, and quintic interpolation.

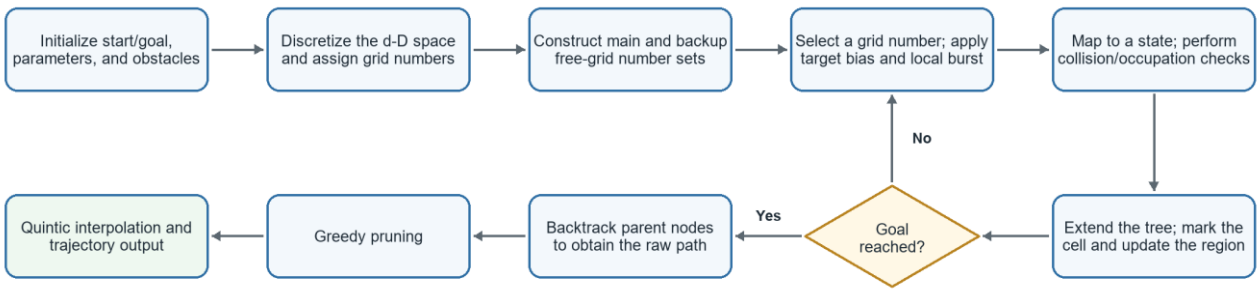


Fig. 11 – Workflow of Optimized-RRT

RESULTS

Simulation Experiments and Analysis of the Algorithm

RRT, goal-biased RRT* (GB-RRT*), Informed-RRT*, and Optimized-RRT were compared as practical time-to-feasible-path planners in 2D, 3D, and manipulator scenarios. Within each scenario, they used the same map, start/target states, obstacle model, collision criterion, computing environment, and expansion settings where applicable. Each implementation retained its own search and rewiring logic and stopped when it returned a feasible path; the comparison therefore concerns the implemented pipelines rather than asymptotic RRT* optimality.

For each algorithm and scenario, 100 independent simulations with random seeds were conducted in MATLAB 2022b on 64-bit Windows 11 Enterprise with an Intel Core i5-13500H processor. The tables report descriptive mean ± standard deviation for planning time, iterations, and path length, and the mean final node count. Greedy pruning was part of the Optimized-RRT pipeline and was not retrofitted to every baseline; consequently, final-node reductions represent the complete pipelines and cannot be attributed solely to sampling. The archived results were aggregate summaries, so no post hoc significance test or success-rate reconstruction was added, and no claim of statistical significance is made.

Two-Dimensional Simulation in a Narrow Agricultural Environment

The basic parameters of the four algorithms were kept consistent in the typical two-dimensional narrow and complex scenario. The simulation map was $1000\ \mathrm{mm} \times 1000\ \mathrm{mm}$, and each grid cell was $10\ \mathrm{mm} \times 10\ \mathrm{mm}$. The start and target points were $(35{,}35)\ \mathrm{mm}$ and $(900{,}900)\ \mathrm{mm}$, respectively, and the fixed expansion step was $\Delta = 20\ \mathrm{mm}$. The target-bias probability of both Optimized-RRT and GB-RRT* was set to $p_g = 0.05$. For Optimized-RRT, the number of expanded columns was $f = 10$, and the region-switching threshold was $T_{\mathrm{eff}} = 0.16$.

Fig. 12 shows the exploration trees and collision-free paths. Green and red circles denote the start and target, black areas denote obstacles, green lines denote the search tree, and the red line denotes the final path. In Fig. 12(d), the area between the blue boundaries is the main sampling region. Across 100 runs, Optimized-RRT reduced mean planning time by 40.61%–94.32%, iterations by 57.51%–88.66%, and final path-node count by 89.29%–94.59% relative to the three baselines (Table 3). Fig. 12 also shows a more concentrated search tree, with fewer repeated expansions near obstacles.

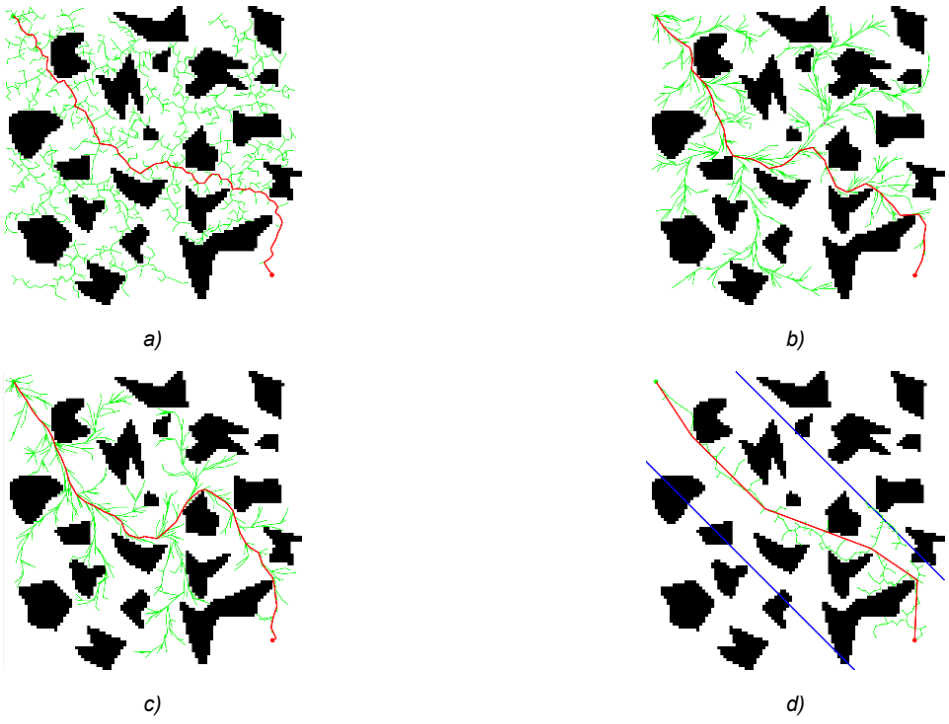


Fig. 12 – Two-dimensional planning results: (a) RRT; (b) GB-RRT*; (c) Informed-RRT*; (d) Optimized-RRT

Table 3

Performance comparison in the two-dimensional scenario (descriptive statistics)

Algorithm	Planning time [s]	Number of iterations	Path length [mm]	Final path nodes
RRT	0.097 ± 0.121	1144.1 ± 641.5	1674.990 ± 89.525	85.04
GB-RRT*	0.049 ± 0.026	662.3 ± 178.0	1448.045 ± 77.170	42.93
Informed-RRT*	0.512 ± 0.654	2480.7 ± 1127.6	1451.024 ± 74.337	43.07
Optimized-RRT	0.029 ± 0.011	281.4 ± 82.0	1291.823 ± 40.641	4.60

Three-Dimensional Simulation

The parameters of the four algorithms were kept consistent in the typical three-dimensional case. The simulation space was $500\,\mathrm{mm} \times 500\,\mathrm{mm} \times 500\,\mathrm{mm}$, and each grid cell was $10\,\mathrm{mm} \times 10\,\mathrm{mm} \times 10\,\mathrm{mm}$. The start and target points were $(10,10,10)\,\mathrm{mm}$ and $(460,460,460)\,\mathrm{mm}$, respectively, and the fixed expansion step was $\Delta = 20\,\mathrm{mm}$. For Optimized-RRT, $f = 10$ and $T_{\mathrm{eff}} = 0.16$.

In the three-dimensional scenario, Optimized-RRT reduced planning time by 88.95%–98.92%, iterations by 79.26%–94.67%, and final path-node count by 67.31%–90.83% relative to the baselines (Table 4). Figs. 13 and 14 show that Optimized-RRT generated a sparser exploration tree and a path with fewer turns. Grid-number-guided sampling concentrated expansion within the active region, while region switching preserved obstacle-bypassing alternatives and pruning simplified the final path.

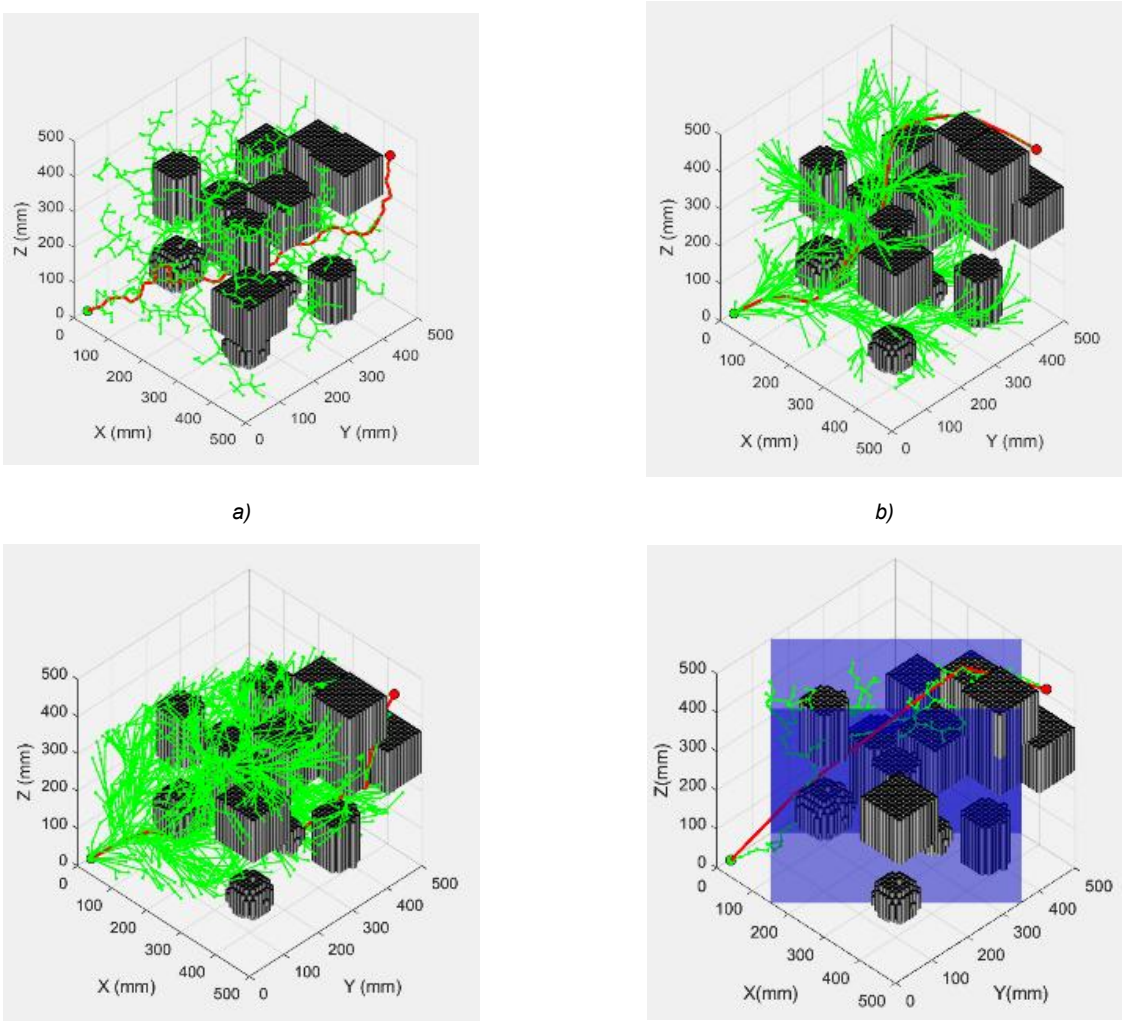


Fig. 13 – Three-dimensional exploration trees: (a) RRT; (b) GB-RRT*; (c) Informed-RRT*; (d) Optimized-RRT

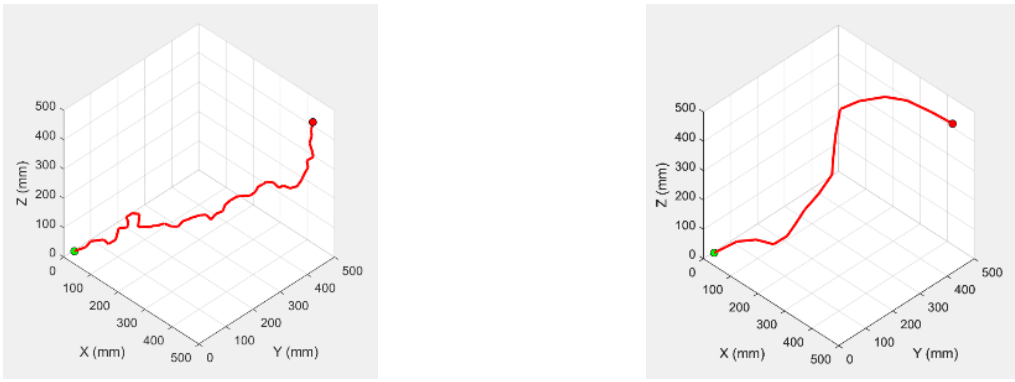


Fig. 14 – Three-dimensional final paths: (a) RRT; (b) GB-RRT*; (c) Informed-RRT*; (d) Optimized-RRT

Table 4

Performance comparison in the three-dimensional scenario (descriptive statistics)

Algorithm	Planning time [s]	Number of iterations	Path length [mm]	Final path nodes
RRT	0.662 ± 0.559	2400.9 ± 931.7	1239.371 ± 93.277	62.89
GB-RRT*	0.376 ± 0.421	1272.1 ± 650.1	932.245 ± 44.793	17.65
Informed-RRT*	3.825 ± 1.824	4948.0 ± 1383.4	947.552 ± 41.742	17.70
Optimized-RRT	0.041 ± 0.017	263.9 ± 81.1	858.918 ± 24.036	5.77

Parameter Sensitivity Analysis

A one-factor-at-a-time sensitivity analysis was conducted in the two-dimensional scenario. Grid side length was set to 5, 10, and 20 mm; expansion step Δ to 10, 20, and 30 mm; target-bias probability p_g to 0.02, 0.05, and 0.10; and region-switching threshold T_{eff} to 0.10, 0.16, and 0.25. Each level used 50 independent simulations with the other parameters fixed at their baseline values.

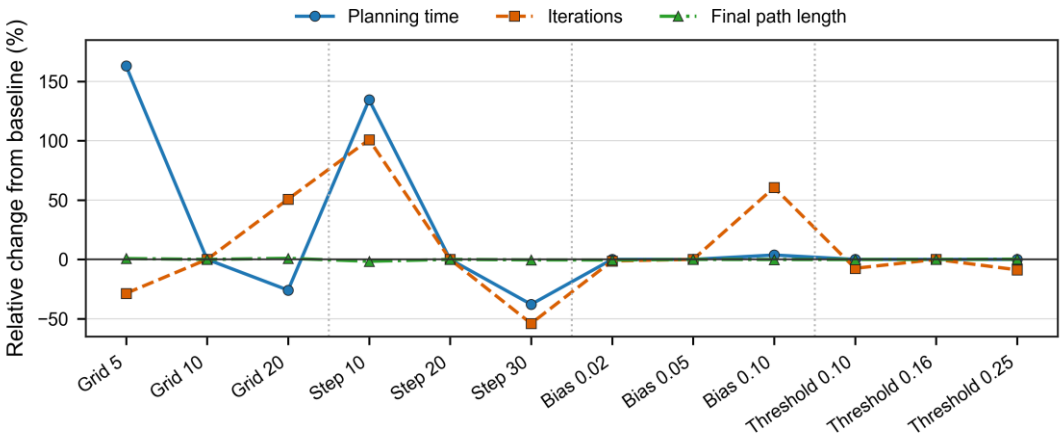


Fig. 15 – Parameter sensitivity analysis relative to the baseline settings

Fig. 15 shows that grid size and step size had the largest effects on planning time and iterations, whereas the final path length varied less. Smaller grids improved representation but increased numbering and search overhead; larger steps reduced expansion count but weakened local adjustment. Target-bias probability and the region-switching threshold had limited influence within the tested ranges. The baseline values—a 10-mm grid, $\Delta = 20\,\mathrm{mm}$, $p_g = 0.05$, and $T_{\mathrm{eff}} = 0.16$—provided a balanced combination of planning efficiency and path quality.

Ablation Experiment on Optimization Strategies

Ablation variants were created by disabling one strategy at a time: dynamic sampling-region adjustment (No-DSD), local burst expansion (No-LBES), node occupancy (No-NOP), target-biased sampling (No-GBS), or greedy pruning (No-GP). All variants used the same map, planning parameters, and random-seed sequence.

Planning time, iterations, original and final path lengths, and original and final node counts were recorded. All variants except No-GP used the same pruning procedure; No-GP retained the raw tree path.

Fig. 16 reports relative changes after each strategy was disabled; positive values indicate increases in time, iterations, or path length. The ablation results show a dimension-dependent trade-off rather than uniform benefit. Removing dynamic region adjustment reduced 2D planning time by approximately 50%, and disabling node occupancy or target bias also produced small 2D time reductions, indicating that set maintenance can outweigh guidance in a simple space. In 3D, dynamic adjustment and target bias produced clearer efficiency benefits, while local burst expansion reduced iterations in both spaces. Greedy pruning dominated final path reduction. Because Fig. 16 contains relative aggregate changes without uncertainty intervals, these effects are interpreted directionally rather than as statistically established module contributions.

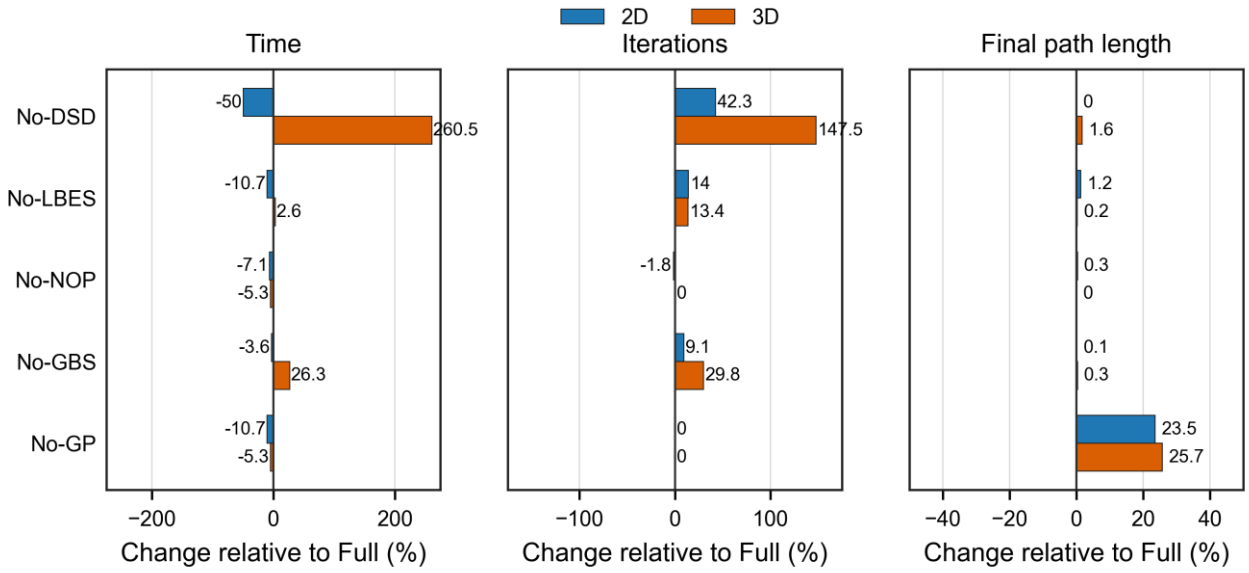
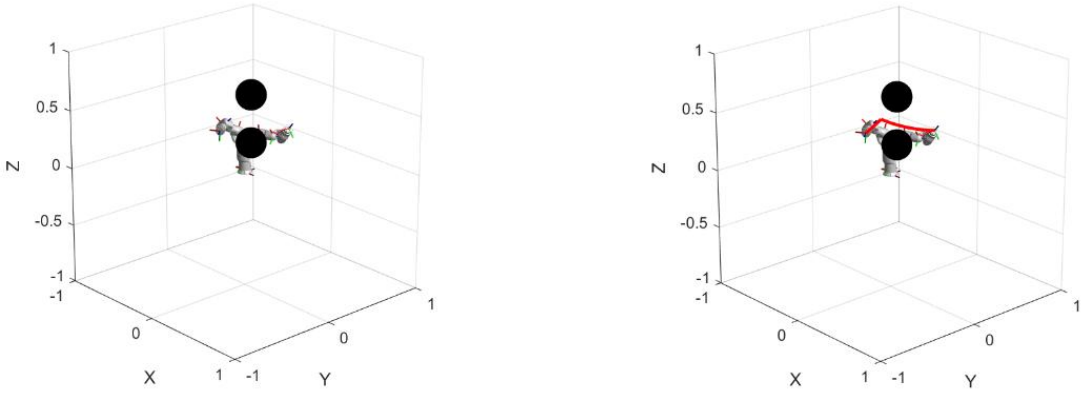


Fig. 16 – Ablation results relative to the complete Optimized-RRT

Simulation Experiment on Path Planning for the Picking Manipulator

A MATLAB 2022b manipulator simulation used the URDF model, the initial and target postures shown in Fig. 17, and two spherical obstacles centered at (0.33, −0.23, 0.40) m and (0.33, −0.23, 0.80) m (Fig. 17(a)).

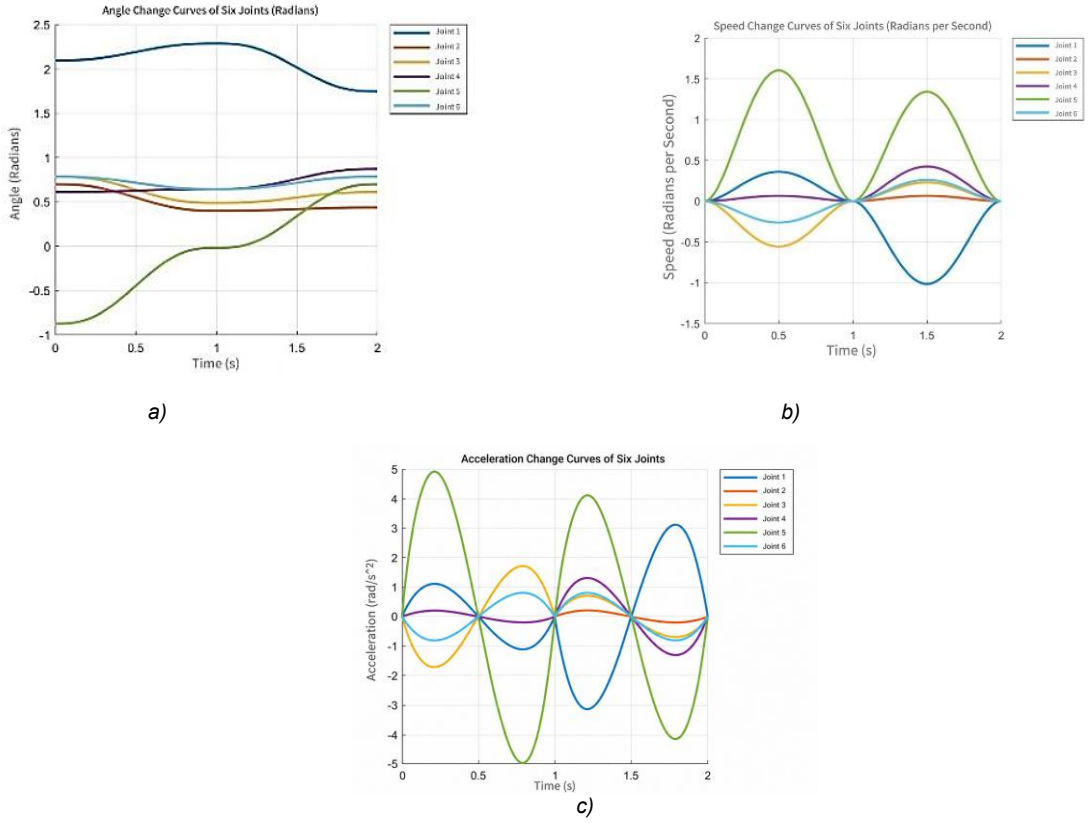
Planning was performed in the six-dimensional joint space using the URDF joint limits and joint-specific grid indices described above. The pruned path was smoothed by quintic interpolation, and the resulting end-effector trajectory is shown in Fig. 17(b).



a) b)

Fig. 17 – Manipulator planning: (a) simulation environment; (b) end-effector trajectory

Fig. 18 shows continuous joint-angle, velocity, and acceleration profiles with the prescribed zero endpoint conditions. These curves demonstrate interpolation continuity, but they were not optimized against measured hardware velocity, acceleration, or jerk limits and should not be interpreted as a time-optimality or actuator-margin assessment.



a) b)

c)

Fig. 18 – Joint trajectory profiles: (a) joint angle; (b) joint velocity; (c) joint acceleration

Real-Platform Feasibility Demonstration in an Agricultural Picking Environment

A single-case feasibility demonstration used a picking platform comprising a six-degree-of-freedom collaborative manipulator, depth camera, flexible end-effector, robot controller, and ROS upper computer; joint commands were transmitted via Gigabit Ethernet. GB-RRT* and Optimized-RRT used the same target, obstacle distribution, and initial posture. Execution time denotes total platform motion/task time rather than isolated planning computation time; joint-space path length and final node count were also recorded. In this arrangement, Optimized-RRT reduced total execution time from 23.1 to 16.5 s (28.57%), joint-space path length from 3.12 to 2.61 rad (16.35%), and final path nodes from 36 to 4 (88.89%) relative to GB-RRT* (Table 5), demonstrating executability and lower path redundancy for this case but not repeatability, statistical superiority, or robustness across greenhouse conditions.

Table 5

Single-case real-platform execution results

Algorithm / optimization rate	Total execution time [s]	Path length [rad]	Final path nodes
GB-RRT*	23.1	3.12	36
Optimized-RRT	16.5	2.61	4
Optimization rate [%]	28.57	16.35	88.89

a) b) c) d) e)

Fig. 19 – Platform sequence: (a) initial; (b–d) obstacle avoidance; (e) completed grasp

DISCUSSION AND LIMITATIONS

The indexed-set representation coordinated active-region restriction, backup-region switching, and repeated-cell suppression through a common sampling structure, but the benefits of its component strategies were dimension dependent. Grid-number guidance produced larger 3D gains while incurring 2D bookkeeping overhead. Planning time and iterations reflect sampling, extension, and switching, whereas node and path reductions depend strongly on pruning. Because baseline post-processing differed and only aggregate results were retained, comparisons are pipeline-level and establish neither module-level causality nor statistical significance. The single platform trial demonstrates feasibility, not robustness; discrete checking, uncalibrated clearance, and fixed endpoints limit safety conclusions. Future work should retain run-level data, harmonize post-processing, verify actuator and collision limits, and test varying branch density, occlusion, and localization error.

CONCLUSIONS

Optimized-RRT is built on a unified indexed-set representation of numbered main and backup free-grid regions. Within this representation, dynamic restriction, region switching, node occupancy, target bias, and local burst expansion coordinate the search, while greedy pruning and quintic interpolation post-process the path. In 2D, planning time, iterations, and final nodes decreased by 40.61%-94.32%, 57.51%-88.66%, and 89.29%-94.59%; in 3D, they decreased by 88.95%-98.92%, 79.26%-94.67%, and 67.31%-90.83%. The results support the value of the complete representation-and-processing pipeline rather than demonstrating that every component is uniformly beneficial: guided sampling improved search efficiency, whereas pruning dominated node reduction. Simulation yielded continuous trajectories, and the single platform case reduced total execution time, joint-space path length, and final nodes by 28.57%, 16.35%, and 88.89%, respectively. Repeatability, calibrated clearance, hardware-limit compliance, and robustness to branch density, occlusion, localization error, and fruit-tree geometry require further evaluation.

ACKNOWLEDGMENTS

This work was supported by the Yunnan Science and Technology Major Project under Grant No. 202602AE090051 and the China Central Fund for Guiding Development of Local Science and Technology under Grant No. 202407AB110010.

REFERENCES

[1] Chen Z., Lei X., Yuan Q., Qi Y., Ma Z., Qian S., Lyu X., (2024), Key technologies for autonomous fruit- and vegetable-picking robots: A review. *Agronomy*, vol.14, 2233. DOI:10.3390/agronomy14102233

[2] Cheng C., Fu J., Su H., Ren L., (2023), Recent advancements in agriculture robots: Benefits and challenges. *Machines*, vol.11, 48. DOI:10.3390/machines11010048

[3] Feng Y., Zhou Z., Shen Y., Wang L., (2023), Obstacle avoidance path planning based on an improved RRT algorithm (基于改进 RRT 算法的避障路径规划). *Chinese Journal of Engineering Design*, vol.30, 707–716. DOI:10.3785/j.issn.1006-754X.2024.03.141

[4] Gammell J.D., Srinivasa S.S., Barfoot T.D., (2014), Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. *Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2014)*, 2997–3004, Chicago/USA. DOI:10.1109/IROS.2014.6942976

[5] He X., Zhou Y., Liu H., Shang W., (2025), Improved RRT*-Connect manipulator path planning in a multi-obstacle narrow environment. *Sensors*, vol.25, 2364. DOI:10.3390/s25082364

[6] Hu Y., Qin J., Wang L., Chen X., Zhao Y., (2024), Path planning research on grape picking robotic arm based on improved RRT algorithm. *INMATEH-Agricultural Engineering*, vol.74, 833–844, Bucharest/Romania. DOI:10.35633/inmateh-74-73

[7] Jiang L., Liu S., Cui Y., Jiang H., (2022), Path planning for robotic manipulator in complex multi-obstacle environment based on improved RRT. *IEEE/ASME Transactions on Mechatronics*, vol.27, 4774–4785. DOI:10.1109/TMECH.2022.3165845

[8] Karaman S., Frazzoli E., (2011), Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, vol.30, 846–894. DOI:10.1177/0278364911406761

[9] Karim M.R., Ahmed S., Reza M.N., Lee K.-H., Sung J., Chung S.-O., (2025), Geometric feature characterization of apple trees from 3D LiDAR point cloud data. *Journal of Imaging*, vol.11, 5. DOI:10.3390/jimaging11010005

[10] Li X., Yang J., Wang X., Fu L., Li S., (2024), Adaptive step RRT*-based method for path planning of a tea-picking robotic arm. *Sensors*, vol.24, 7759. DOI:10.3390/s24237759

[11] Liu Z., Sampurno R.M., Abeyrathna R.M.R.D., Nakaguchi V.M., Ahamed T., (2024), Development of a collision-free path planning method for a 6-DoF orchard harvesting manipulator using RGB-D camera and Bi-RRT algorithm. *Sensors*, vol.24, 8113. DOI:10.3390/s24248113

[12] Oh T., Won Y.-J., Lee S., (2025), TA-RRT*: Adaptive sampling-based path planning using terrain analysis. *Applied Sciences*, vol.15, 2287. DOI:10.3390/app15052287

[13] Orthey A., Chamzas C., Kavraki L.E., (2024), Sampling-based motion planning: A comparative review. *Annual Review of Control, Robotics, and Autonomous Systems*, vol.7, 285–310. DOI:10.1146/annurev-control-061623-094742

[14] Shang D., Wang J., Fan H., Suo S., (2024), Obstacle avoidance path planning for manipulators based on the RRT*-DR algorithm (基于 RRT*-DR 算法的机械臂避障路径规划). *Computer Integrated Manufacturing Systems*, vol.30, no.3, 1149–1160. DOI:10.13196/j.cims.2022.0902

[15] Shi W., Wang K., Zhao C., Tian M., (2022), Obstacle avoidance path planning for the dual-arm robot based on an improved RRT algorithm. *Applied Sciences*, vol.12, 4087. DOI:10.3390/app12084087

[16] Wang H., Zhao A., Zhong Y., Zhang G., Wu F., Zou X., (2025), Enhanced visual detection and path planning for robotic arms using YOLOv10n-SSE and hybrid algorithms. *Agronomy*, vol.15, 1924. DOI:10.3390/agronomy15081924

[17] Xiao G., Zhang L., Wu T., Han Y., Ding Y., Han C., (2024), FBi-RRT: A path planning algorithm for manipulators with heuristic node expansion. *Robotica*, vol.42, 644–659. DOI:10.1017/S0263574723001674

[18] Xu J., Ren C., Chang X., (2023), Robot time-optimal trajectory planning based on quintic polynomial interpolation and improved Harris hawks algorithm. *Axioms*, vol.12, 245. DOI:10.3390/axioms12030245

[19] Xuan D.T., Hung N.T., Thang V.T., (2025), A comprehensive review of improved A* path planning algorithms and their hybrid integrations. *Automation*, vol.6, 52. DOI:10.3390/automation6040052

[20] Zhu J., Pan D., (2024), Improved genetic algorithm for solving robot path planning based on grid maps. *Mathematics*, vol.12, 4017. DOI:10.3390/math12244017