DCB-RRT*: DYNAMIC CONSTRAINED SAMPLING BASED BIDIRECTIONAL RRT* WITH IMPROVED CONVERGENCE RATE

Xining Cui**,† Caiqi Wang**,† Yi Xiong,** and Shiqian Wu**

Abstract

Rapidly-exploring random tree star (RRT*) is widely used in path planning problems because of its probabilistic completeness and asymptotic optimality. The bidirectional RRT* (B-RRT*) is proposed to speed up finding the optimal path. However, both algorithms perform blind exploration in space, which suffer from low node utilisation and poor expansion orientation. To overcome these problems, dynamic constrained sampling based on the bidirectional RRT* (DCB-RRT*) is presented. The proposed DCB-RRT* grows two random trees from the start and the end points for expansion, respectively, and dynamically adjusts the sampling area (*Dyn-Sample*) based on the number of collision detection failures, improving the effectiveness of sampling points in the initial path. In the convergence stage, a method of the dynamic angle to limit the sampling area (*Limit-Sample*) is proposed to improve the path convergence rate. The sampling point bias extension (*DCB-Extend*) is developed to increase the mutual guidance between the dual-trees and reduces the time to find the initial path. A dynamic step is also used to improve node utilisation. Numerical simulations under various environmental conditions demonstrate that DCB-RRT* has certain advantages in terms of convergence rate.

Key Words

Path planning, dynamic constrained sampling, collision detection, bias extension, dynamic step

1. Introduction

Path planning is a key problem in robotics. Given an initial position and a target position of a robot, the goal of path planning is to find a feasible path such that there is no intersection between the path and obstacles [1]–[3]. Path planning has been widely used in our daily life, such as unmanned driving [4]–[6], intelligent sorting [7]–[9], robotic surgery [10], [11], agricultural picking [12]–[14], and other fields. The optimisation criteria usually vary with different tasks, such as time, distance, energy consumption, and safety [15], among which, minimising time consumption and achieving the shortest travel distance are essential for all tasks. Extensive efforts have been dedicated to enhancing the convergence of path planning algorithms. The primary objective is to discover the shortest feasible path in the least amount of time. Effective path planning algorithms offer significant potential for enhancing automation and intelligence across diverse fields and contribute to the progress of artificial intelligence. They play a crucial role in enabling unmanned driving to navigate through increasingly complex road environments and in optimising robot picking processes, leading to greater efficiency.

Path planning algorithms are generally categorised as intelligent optimisation algorithms and traditional algorithms. Intelligent optimisation algorithms include genetic algorithm (GA) [16], [17], memetic algorithm (MA) [18], [19], particle swarm optimisation (PSO) [20], [21], and ant colony optimisation (ACO) [22], [23], *etc.*, which originate from mimicking biological behaviour. Such algorithms rely on heuristic information, and the model parameters need to be tuned several times to approach the optimal values. Traditional algorithms include artificial potential field (APF) [24], [25], A-star (A*) [26], D-star (D*) [27], probabilistic road maps (PRM) [28], and rapidly-exploring random tree (RRT) [29]. APF makes the expansion of random tree directionally. The calculation of attraction and repulsion forces is expensive in complex maps, and it is easy to fall into local optima. A* is a direct search method to find the shortest path. The map grid affects the path convergence. D* is a local planning method that plans the shortest path from the current coordinates to the target coordinates by making assumptions about the unknown part of the terrain. It is suitable for dynamic environments because of its high search efficiency. PRM is a multi-query method for constructing roadmaps using sampling, whose performance is determined by the number of sampling

** Institute of Robotics and Intelligent Systems, School of Information Science and Engineering, Wuhan University of Science and Technology, Wuhan, China 430081; e-mail: {cuixining, wangcaiqi, xiongyi, shiqian.wu}@wust.edu.cn
Corresponding author: Shiqian Wu
† These authors contributed equally to this work.

Recommended by Jiankun Wang
(DOI: 10.2316/J.2024.206-1056)

points and domain settings. RRT is a random sampling based algorithm that gradually explores a spatial region by performing collision detection on sampling points until a feasible path is found. This method has the advantages of avoiding a large number of calculations associated with modelling the space, eliminating the need for geometric partitioning of the exploration area, and featuring fast search speed, and high convergence.

Due to the randomness of RRT sampling, the final generated path is often a feasible path rather than an optimal path. Moreover, it requires a large time when facing the maps with narrow passages. Therefore, several improved RRT algorithms have been proposed in recent years [30], among which RRT* [31] is a typical one. Different from RRT, RRT* introduces a neighbourhood search process for the newly generated nodes to select low-cost parent nodes. There is also the process of rewiring to further reduce the path cost of other branches, which makes remarkable progress in solving the optimal path planning in high-dimensional space. RRT* is asymptotically optimal, *i.e.*, it can always converge to an optimal solution by giving enough running time.

Although RRT* solves the optimisation problem of RRT to some extent, the search for new parent nodes and the rewiring process also make the algorithm less efficient. Recently, researchers have made improvement to improve the efficiency of RRT*. Among them, a sampling method with goal-biased strategy was proposed in [32]. The method treats the target point as a sampling point with a certain probability, which makes the growth of the random tree with a certain direction and accordingly improve the speed of finding the initial path. It is noted that the optimal probability cannot be dynamically adjusted as the random tree grows. An algorithm called informed RRT* was proposed in [33], which builds an elliptical sampling region by using the start and goal points as focal point, and the high-quality sampling points are generated within a dynamically changing ellipse to improve the path convergence rate. In the RRT*-smart algorithm [34], the path is optimised by interconnecting the visible nodes in the initial path and which generates bias points for smart sampling. The path optimisation process is accelerated by constrained sampling. In 2013, an optimal bidirectional rapid random tree (B-RRT*) algorithm was proposed in [35]. The dual-tree expansion strategy grows two random trees from the start and end points until they meet each other. It is highlighted that B-RRT* does not change the blindness of the random tree exploring the space, the idea of bidirectional growth greatly improves the rate of convergence of path. After that, the bidirectional dual-tree idea has been prevailed in optimisation algorithms. For example, intelligent bidirectional RRT* (IB-RRT*) [36] converges quickly to the optimal path solution by introducing an intelligent sample insertion heuristic using uniform sampling heuristics. In [37], potentially guided IB-RRT* (PIB-RRT*) and potentially guided bidirectional RRT* (PB-RRT*) were proposed to overcome the blindness issue in space exploration. An algorithm called GB-RRT* by combining the principle of the RRT* with the grid searching strategy was proposed in [38]. The proposed hybridised algorithm takes advantage of the grid searching strategy to make up for the weakness of RRT* and is applicable in complex maps without relying on pre-designed road networks. From the above algorithms, the factors affecting the path convergence rate can be summarised into three points. The first point is that it does not provide a better initial solution for the convergence stage, which increases the time consumption of path convergence, such as literature [32], [33], [35], [37]. The second point is that the growth of random trees lacks guidance, which increases the time to find the initial path, thereby increasing the time of the whole process, such as literature [33]–[35], [38]. The third point is that the sampling process is unconstrained and global sampling leads to poor quality and low utilisation of the sampling points, which severely affects the rate of path convergence, such as literature [35]–[37].

In this paper, we propose a dynamic constrained sampling based bidirectional RRT* (DCB-RRT*) algorithm. The motivation of our method is to obtain high-quality sampling points by constraining the sampling area, thereby improving the convergence rate in finding the optimal solution. The algorithm is composed of three parts (*Dyn-Sample*, *Limit-Sample* and *DCB-Extend*), each of which is to prepare for improving the path convergence rate. Ablation experiments verify the enhancement of each part of the work. Numerical simulations are conducted and compared with other algorithms in various maps, and the results show that DCB-RRT* has the best performance in terms of path convergence rate. The main contributions of the paper are as follows:

1) A *Dyn-Sample* method according to the number of collision detection failures is proposed, which helps to provide a good initial solution for the convergence stage.
2) A *DCB-Extend* is proposed to increase the mutual guidance between the dual trees and improves the node utilisation rate through the dynamic step.
3) A *Limit-Sample* solution is proposed to speed up the path convergence process by improving the quality of sampling points.
4) Compared with other methods, the proposed method exhibits robustness and can better equipped to address path planning challenges in multi-type environments.

The remainder of this paper is presented below. Section 2 presents the problem definition and the B-RRT* algorithm. The proposed DCB-RRT* algorithm is described in Section 3. Section 4 explains the probabilistic completeness, asymptotic optimality, time complexity, and space complexity of the DCB-RRT* algorithm. Experimental comparisons of several maps are performed in Section 5, followed by conclusion in Section 6.

2. Background

2.1 Problem Definition

Given a configuration space $X \subset \mathbb{R}^n$, where n represents the dimension of the given space, *i.e.*, $n \in N$, $n \geq 2$. The configuration space is further classified into obstacle and

obstacle-free regions denoted by $X_{\text{obs}} \subset X$ and $X_{\text{free}} = X/X_{\text{obs}}$, respectively. $X_{\text{goal}} \subset X_{\text{free}}$ is the goal region. Let $T_a = (V_a, E_a) \subset X_{\text{free}}$ and $T_b = (V_b, E_b) \subset X_{\text{free}}$ represent two growing random trees, where V denotes the nodes and E denotes the edges connecting these nodes. $x^a_{\text{init}} \in X_{\text{free}}$ and $x^b_{\text{init}} \in X_{\text{goal}}$ represent the starting states for T_a and T_b. In this study, we only consider Euclidean space and positive Euclidean distance between any two states, *e.g.*, $x_1 \in X$ and $x_2 \in X$ is denoted by $\text{EucDis}(x_1, x_2)$. Let the path connecting any two states $x_1 \in X_{\text{free}}$ and $x_2 \in X_{\text{free}}$ be denoted by $\sigma : [0, s]$, such that $\sigma(0) = x_1$ and $\sigma(s) = x_2$, whereas s is the positive scalar length of the path. The cost function for calculating the Euclidean distance of the feasible path is $c(\cdot)$, and the set of all collision-free paths σ is denoted as $\sum_{\sigma_{\text{free}}}$.

Problem 1 (Feasible path solution). Find a path $\sigma : [0, s]$ in obstacle-free space $X_{\text{free}} \in X$ such that $\sigma(0) = x_{\text{init}} \in X_{\text{free}}$ and $\sigma(s) \in X_{\text{goal}}$. If one exists, report success. If no such path exists, report failure.

Problem 2 (Optimal path solution). Find the feasible path $c(\sigma^*) = \min\left\{c(\sigma_f), \sigma_f \in \sum_{\sigma_{\text{free}}}\right\}$, such that the cost of the path σ^* is minimum.

2.2 The B-RRT* Algorithm

The proposed algorithm is improved based on the existing B-RRT* algorithm [35], so this section describes the basic principles of the B-RRT* path planning algorithm. Its pseudocode is shown in Algorithm 1.

The detailed expansion process of B-RRT* is shown in Algorithm 1. The map space, obstacles, and parameters used are first initialised. Tree T_a is initialised by x^a_{init} as its root node, where $x^a_{\text{init}} \in X_{\text{free}}$, and tree T_b is initialised by x^b_{init} as its root node where $x^b_{\text{init}} \in X_{\text{goal}}$. The tree growth process first a node x_{rand} is randomly sampled in X_{free}, inserting the x_{new} into the selected tree T_a and rewiring it. Then, the node x_{conn} closest to x_{new} is found from the tree T_b. The function Connect tries to connect the two random trees T_a and T_b, and forming a feasible path σ_{new}. If the cost of the new path $c(\sigma_{\text{new}})$ is less than the previous best cost $c(\sigma_{\text{best}})$, σ_{best} is overwritten by σ_{new}. Finally, the trees T_a and T_b are swapped, and the random tree continues to expand until the maximum number of iterations $n_{\max}$ is reached.

To describe the process of pseudocode more clearly, some steps are explained as follows.

Nearest: It returns a node x_{nearest} that is closest to x_{rand} in the Euclidean distance.

Extend: $\text{Extend}(x_1, x_2, \text{step})$ returns a new node x_{new}. The node is generated along the direction from x_1 to x_2, and the Euclidean distance from x_1 is step.

Near: It returns a set of point sets X_{near}, which are contained in the hypersphere with a specific radius R centred on x_{new}, and the points in the point set can pass collision detection with x_{new}.

ChooseBestParent: This procedure is used to find the $x_{\min}$ in X_{near}, which provides the shortest, collision-free path σ' from the x_{init} to the x_{new}. Algorithm 2 illustrates the procedure.

Algorithm 1 B-RRT*

Input: $X, x^a_{\text{init}}, x^b_{\text{init}}, n_{\max}, R$, step
Output: $(T_a, T_b) = (V, E)$

$V \leftarrow \left\{x^a_{\text{init}}, x^b_{\text{init}}\right\}; E \leftarrow \phi; \sigma_{\text{best}} \leftarrow \infty$;
$T_a \leftarrow (x^a_{\text{init}}, E); T_b \leftarrow \left(x^b_{\text{init}}, E\right)$
for $i \leftarrow 0$ to $n_{\max}$ **do**
  $x_{\text{rand}} \leftarrow$ Sample (i);
  $x_{\text{nearest}} \leftarrow \text{Nearest}(x_{\text{rand}}, T_a)$;
  $x_{\text{new}} \leftarrow \text{Extend}(x_{\text{nearest}}, x_{\text{rand}}, \text{step})$;
  $X_{\text{near}} \leftarrow \text{Near}(x_{\text{new}}, T_a, R)$;
  $x_{\min} \leftarrow$ ChooseBestParent $(x_{\text{new}}, X_{\text{near}})$;
  if $x_{\min} \neq \phi$ **then**
    $T \leftarrow \text{Insert}(x_{\text{new}}, x_{\min}, T_a)$;
    $T \leftarrow \text{Rewire}(x_{\text{new}}, X_{\text{near}}, E)$;
  end if
  $x_{\text{conn}} \leftarrow \text{Nearest}(x_{\text{new}}, T_b)$;
  $\sigma_{\text{new}} \leftarrow \text{Connect}(x_{\text{new}}, x_{\text{conn}}, T_b)$;
  if $\sigma_{\text{new}} \neq \phi \&\& c(\sigma_{\text{new}}) < c(\sigma_{\text{best}})$ **then**
    $\sigma_{\text{best}} \leftarrow \sigma_{\text{new}}$;
  end if
  SwapTrees (T_a, T_b);
end for
return $(T_a, T_b) = (V, E)$

Algorithm 2 ChooseBestParent

Input: $x_{\text{new}}, X_{\text{near}}$
Output: $x_{\min}$

$c_{\text{best}} \leftarrow \infty$;
for each $x_{\text{near}} \in X_{\text{near}}$ **do**
  if $c(x_{\text{near}}) + \text{EucDis}(x_{\text{near}}, x_{\text{new}}) < c_{\text{best}}$ **then**
    $c_{\text{best}} = c(x_{\text{near}}) + \text{EucDis}(x_{\text{near}}, x_{\text{new}})$;
    $x_{\min} \leftarrow x_{\text{near}}$;
  end if
end for
return$x_{\min}$

Algorithm 3 Rewire

Input: $x_{\text{new}}, X_{\text{near}}, E$
Output: T

for each $x_{\text{near}} \in X_{\text{near}}$ **do**
  if $c(x_{\text{new}}) + \text{EucDis}(x_{\text{new}}, x_{\text{near}}) < c(x_{\text{near}})$ **then**
    $x_{\text{parent}} \leftarrow \text{Parent}(E, x_{\text{near}})$;
    $E \leftarrow (E \backslash \{(x_{\text{parent}}, x_{\text{near}})\} \cup \{(x_{\text{new}}, x_{\text{near}})\})$;
  end if
end for
T

Rewire: Algorithm 3 gives the process of rewiring. Here, the algorithm examines each vertex $x' \in X_{\text{near}}$ lying inside the ball region centred at x_{new}. If the cost of the path connecting x_{init} and x' through x_{new} is less than the existing cost of reaching x', then x_{new} is made the parent of x'. Instead, no changes are made to the tree and the next vertex will be checked.

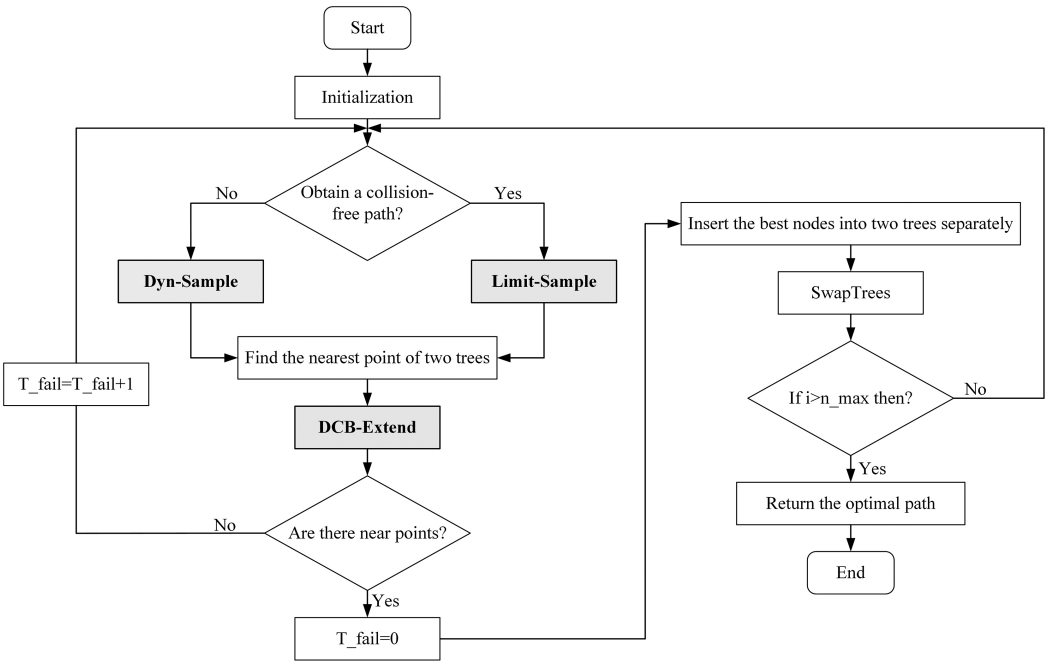

Figure 1. The flowchart of the DCB-RRT* algorithm.

Algorithm 4 Connect

Input: $x_{\text{new}}, x_{\text{conn}}, T_b$

Output: σ_{new}

$x^b_{\text{new}} \leftarrow \text{Extend}\left(x_{\text{conn}}, x_{\text{new}}, \text{step}\right)$;
$X^b_{\text{near}} \leftarrow \text{Near}\left(x^b_{\text{new}}, T_b, R\right)$
$x_{\min} \leftarrow \text{ChooseBestParent}\left(x^b_{\text{new}}, X^b_{\text{near}}\right)$;
if $x_{\min} \neq \phi$ **then**
    $E \leftarrow E \cup \left(x_{\min}, x_{new}\right)$;
    $\sigma_{\text{new}} \leftarrow \text{GeneratePath}\left(x_{\min}, x_{\text{new}}\right)$;
    return σ_{new}
end if
return NULL

Connect: Algorithm 4 outlines the implementation of Connect heuristic of B-RRT*. The purpose of this function is to generate an end-to-end feasible path σ_{new}.

GeneratePath: This procedure ends with the generation of a feasible path solution between two points, connecting x^a_{init} and x^b_{init}.

SwapTrees: The iteration ends with a swap tree, and in the next iteration, the same procedure is performed on another tree.

3. The DCB-RRT* Algorithm

This section describes the principle of the DCB-RRT* path planning algorithm, the dynamic constrained sampling method, the biased extension strategy, the dynamic step, and draws conclusions through analysis.

3.1 Proposed Algorithm

RRT* provides probabilistic completeness and asymptotic optimality by optimising the parent nodes of the sampling points and the surrounding region nodes. However, the exploration process is blind and inefficient, resulting in a very time-consuming algorithm. The dual-tree strategy balances search efficiency and feasible path length to some extent [39], but it does not solve the essential problem that exists when exploring the space. Therefore, the DCB-RRT* path planning algorithm is proposed to solve these problems, and its pseudocode is shown in Algorithm 5. Path planning can usually be divided into two stages: finding the initial path and converging to the optimal path. Its purpose is to use the least time to plan the optimal path between two points. The dynamic constrained sampling strategy proposed by DCB-RRT* in the stage of finding the initial path and the stage of converging to the optimal path, respectively, greatly improves the rate of finding the optimal path. The random tree bias extension strategy is proposed, which increases the mutual guidance of the dual-tree growth and reduces the time required for the encounter. The dynamic step is also used to improve the node utilisation.

flag: The flag of whether the initial path is found or not. If the flag is 0, the initial path has not been found yet. If the flag is 1, the path is in the path convergence stage.

Collisionfree: Given two nodes, x_1 and x_2, it checks whether the path σ from x_1 to x_2 is a feasible path, that is $\forall\tau \in [0,1]$, $\sigma(0) = x_1$, $\sigma(1) = x_2$, $\sigma(\tau) \in X_{\text{free}}$.

The flowchart of the DCB-RRT* algorithm is shown in Fig. 1. The whole process is divided into the stages of finding the initial path and converging to the optimal path. Different dynamic constrained sampling strategies, *i.e.*, *Dyn-Sample* and *Limit-Sample*, are used in the two stages, which are the key works in this study. By using the two strategies, the nearest points $\left\{x^a_{\text{nearest}}, x^b_{\text{nearest}}\right\}$ of x_{rand} are found on two random trees. Another work is to guide the bias extension of the current random tree T_a by sampling point x_{rand} and the nearest point x^b_{nearest} of the opposite random tree together, which is represented

Algorithm 5 DCB-RRT*

Input: $X, x^a_{init}, x^b_{init}, n_{\max}, T_{\text{fail}}, R, \theta_0$
Output: $(T_a, T_b) = (V, E)$

$V \leftarrow \left\{x^a_{\text{init}}, x^b_{\text{init}}\right\}; E \leftarrow \phi; \sigma_{\text{best}} \leftarrow \infty;$
$T_a \leftarrow (x^a_{\text{init}}, E); T_b \leftarrow \left(x^b_{\text{init}}, E\right); T_{\text{fail}} \leftarrow 0;$ flag $\leftarrow 0;$
for $i \leftarrow 0$ to $n_{\max}$ **do**
  if flag $== 0$ **then**
    $\theta = \theta_0 + T_{\text{fail}} \cdot \theta_0$
    $x_{\text{rand}} \leftarrow$ Dyn - Sample $\left(i, x^a_{\text{init}}, x^b_{\text{init}}, \theta\right);$
  else
    $x_{\text{rand}} \leftarrow$ Limit - Sample $\left(i, \sigma_{\text{best}}, x^a_{\text{init}}, x^b_{\text{init}}\right);$
  end if
  $\left\{x^a_{\text{nearest}}, x^b_{\text{nearest}}\right\} \leftarrow$ Nearest $(x_{\text{rand}}, T_a, T_b);$
  $x_{\text{new}} \leftarrow DCB - \text{Extend}\left(x^a_{\text{nearest}}, x^b_{\text{nearest}}, x_{\text{rand}}\right);$
  $X^a_{\text{near}} \leftarrow$ Near $(x_{\text{new}}, T_a, R);$
  if $X^a_{\text{near}} == \phi$ **then**
    $X^a_{\text{near}} \leftarrow$ Nearest&&Collisionfree $(x_{\text{new}}, T_a);$
    if $X^a_{\text{near}} == \phi$ **then**
      $T_{\text{fail}}++$
      continue
    end if
    $T_{\text{fail}} \leftarrow 0$
  end if
  $x^a_{\min} \leftarrow$ ChooseBestParent $(x_{\text{new}}, X^a_{\text{near}});$
  $T \leftarrow$ Insert $(x_{\text{new}}, x^a_{\min}, T_a);$
  $T \leftarrow$ Rewire $(x_{\text{new}}, X^a_{\text{near}}, E);$
  $X^b_{\text{near}} \leftarrow$ Near $(x_{\text{new}}, T_b, R);$
  if $X^b_{\text{near}} \neq \phi$ **then**
    $x^b_{\min} \leftarrow$ ChooseBestParent $\left(x_{\text{new}}, X^b_{\text{near}}\right);$
    $E \leftarrow E \cup \left(x^b_{\min}, x_{\text{new}}\right);$
    $\sigma_{\text{new}} \leftarrow$ GeneratePath $\left(x^b_{\min}, x_{\text{new}}\right);$
    flag $\leftarrow 1;$
  end if
  if $\sigma_{\text{new}} \neq \phi \&\& c(\sigma_{\text{new}}) < c(\sigma_{\text{best}})$ **then**
    $\sigma_{\text{best}} \leftarrow \sigma_{\text{new}};$
  end if
  SwapTrees $(T_a, T_b);$
end for
return $(T_a, T_b) = (V, E)$

by *DCB-Extend*. If there is no neighbourhood point X^a_{near} around the newly generated point x_{new} or the nearest point x^a_{nearest} that can pass collision detection, it means that the space occupied by obstacles in the sampling area is large at this time, and the sampling area should be increased by the angle constraint θ to make the random tree have a large exploration space. If there are neighbour points $\left\{X^a_{\text{near}}, X^b_{\text{near}}\right\}$ or nearest points $\left\{x^a_{\text{nearest}}, x^b_{\text{nearest}}\right\}$ of x_{new} in both dual-trees, it means that a feasible path σ can be formed between two points. In the path convergence stage, the sampling area is continuously reduced according to the angles between the nodes in the path and the starting point and the ending point to improve the convergence rate. When the iteration time is greater than the set threshold, the shortest path returned is the optimal path σ^* during this time. The *Dyn-Sample*, *Limit-Sample,* and *DCB-Extend* work of this paper are highlighted in detail below.

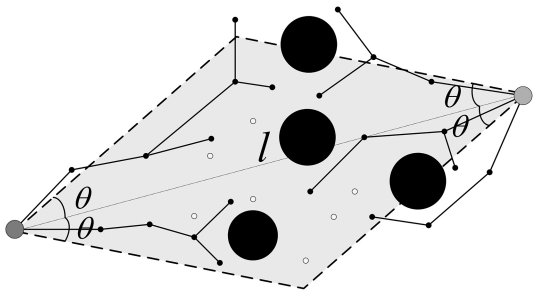

Figure 2. Schematic diagram of dynamic constraint sampling.

3.2 Dyn-Sample

Different from RRT*, B-RRT*, informed RRT*, and RRT*-smart, which are blind and inefficient, the dynamic constraint sampling strategy is proposed to dynamically adjust the sampling area based on the number of collision detection failures during sampling. If the number of collision detection failures increases cumulatively, it indicates that the obstacles around the random trees are dense so that the sampling area should be expanded to increase the scalability of the random tree. Otherwise, a small sampling area is used to reduce the exploration of irrelevant regions by the random tree. Dynamic constraint sampling schematic diagram is shown in Fig. 2, in which the starting point and the ending point are connected to form a line l, θ is the dynamic constraint sampling angle, so each sampling point should be within the area constrained by the angle (The range enclosed by dashed lines in the figure is the sampling area, where the yellow dots are sampling points and the black circles are obstacles). θ becomes large as the number of collision detection failures increases, once the collision detection is passed the value of T_{fail} is set to 0. The relationship can be presented by the following equation:

$$\theta = \theta_0 + T_{\text{fail}} \cdot \Delta \; when \begin{cases} \text{Pass collision detection } T_{\text{fail}}=0 \\ \text{Collision detection failed } T_{\text{fail}}=T_{\text{fail}}+1 \end{cases} \quad (1)$$

where θ_0 is the initial angle, Δ is the variation interval of θ between $\theta \in [\theta_0, \pi]$.

3.3 Limit-Sample

The existing algorithms suffer from the same sampling blindness in the path convergence stage, and the global scope search largely reduces the probability of finding the optimal solution. This problem can be solved using the proposed *Limit-Sample* method. As shown in Algorithm 6, considering that the initial path is close to the optimal or sub-optimal solution, the path convergence can be restricted to the region with a well-defined boundary. The difference between the proposed *Limit-Sample* and the dynamic ellipse idea of Informed RRT* is that we improve it on the basis of dual-tree instead of single-tree, and limit the sampling area by angle, so that the optimisation process is carried out in a similar rhombus space. The

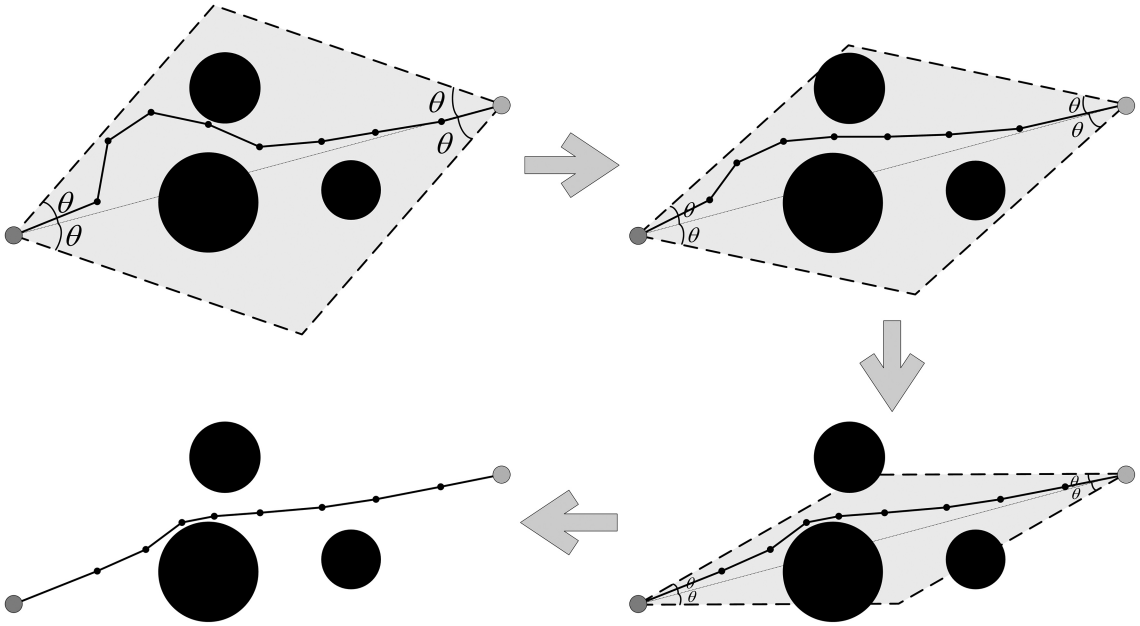

Figure 3. Path optimisation by continuously shrinking the diamond region.

Algorithm 6 Limit-Sample

Input: $\sigma_{best}, i, x^a_{init}, x^b_{\text{init}}$
Output: x_{rand}

```
if mod(i, 7) == 0 then
    x_rand = sample (i);
else
    θ_1 = max_angle (σ_best , x^a_init );
    θ_2 = max_angle (σ_best , x^b_init );
    θ_max = max (θ_1, θ_2);
    x_rand = lim − sample (i, θ_max , θ);
end if
return x_rand
```

proposed solution is more direct and effective, and has certain advantages in the convergence rate. The size of the diamond is determined by the largest angle $\theta_{\max}$ in the (θ_1, θ_2) between the farthest node in the current path and the starting and ending points, and each sampling is performed in the bounded diamond area. The process of optimising the path by continuously shrinking the diamond region is shown in Fig. 3. By gradually updating the θ, the search space is significantly reduced. To ensure the probabilistic completeness of the optimal solution in the multi-channel map, we introduce the bias ratio parameter. The bias ratio determines the number of random samplings taken within the constrained region instead of globally, and the ratio can affect the efficiency of reaching the optimal solution. In this study, the ratio is selected according to the recommendation in the paper [40].

3.4 DCB-Extend

The random trees in the DCB-RRT* algorithm are expanded from the starting point x^a_{init} and the ending point x^b_{init}, respectively. If the guidance information is not considered in the growth process, a large number of invalid nodes are generated, resulting in randomness during path planning. To improve the rate of meeting dual-trees, the bias extension strategy (*DCB-Extend*) is proposed in this study, and the growth process is shown in Algorithm 7. Here we borrow the idea of target probability bias, which is to take the target point as a certain probability of sampling point, so that the random tree growth has a directionality, which is usually used in single-tree generation model. In this work, the proposed growth strategy is implemented in the dual-tree growth model. The purpose is to make the dual-tree has mutual guidance, which improves the speed of the two trees to meet each other. This makes the current random tree not only be guided by x_{rand} when expanding from x^a_{nearest} to x_{new} but also by the co-guidance of x^b_{nearest} on another random tree, as shown in Fig. 4. The method of determining the extension direction and extension distance by vector synthesis increases the directivity between the dual-trees.

It is known that the expansion step of B-RRT* is fixed, resulting in slow local expansion. In this work, the dynamic step is adopted to speed up the growth of random trees. x_{new} is generated by the synthesis of the two vectors, where the vector v_1 is formed from x^a_{nearest} to x_{rand}. It is expanded according to the distance between them rather than a certain step, thus reducing the useless search of the blank area. The direction of another vector is from x^a_{nearest} to x^b_{nearest}, and the length of this vector indicates the distance between the two random trees. Thus, the dynamic step is expressed as below:

$$l = v_1 + v_2 \cdot \text{step} \tag{2}$$

where l is the growth step vector (x_{nearest} grows to x_{new} through the vector l), and step is the dynamic factor, the size of which depends on the spacing distance between two

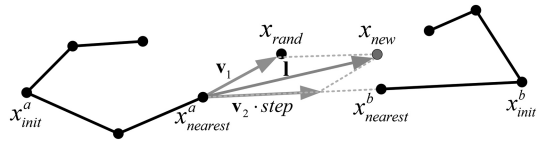

Figure 4. Schematic diagram of biased growth.

Algorithm 7 DCB-Extend

Input: $x^a_{\text{nearest}}, x^b_{\text{nearest}}, x_{\text{rand}}$
Output: x_{new}

1: $\mathbf{v}_1 = x_{\text{rand}} - x^a_{\text{nearest}}$;
2: $\mathbf{v}_2 = \left(x^a_{\text{nearest}} - x^b_{\text{nearest}}\right) / \text{norm}\left(x^a_{\text{nearest}} - x^b_{\text{nearest}}\right)$;
3: $\text{step} = \text{norm}(\mathbf{v}_1) + \left(\text{norm}\left(x^a_{\text{nearest}} - x^b_{\text{nearest}}\right) - \text{norm}(\mathbf{v}_1)\right) / \text{norm}(\mathbf{v}_1)$;
4: $x_{\text{new}} = x^a_{\text{nearest}} + \mathbf{I}$;
5: return x_{new}

random trees as shown in (3).

$$\begin{aligned} \text{step} &= \text{norm}(v_1) \\ &+ \frac{\left(\text{norm}(x^a_{\text{nearest}} - x^b_{\text{nearest}}) - \text{norm}(v_1)\right)}{\text{norm}(v_1)} \end{aligned} \tag{3}$$

Finally x_{new} can be obtained from the following equation:

$$\begin{aligned} x_{\text{new}} &= x^a_{\text{nearest}} + (v_1 + v_2 \cdot \text{step}) \\ &= x^a_{\text{nearest}} + I \end{aligned} \tag{4}$$

4. Analysis of the DCB-RRT* Algorithm

4.1 Probabilistic Completeness

In a configuration space, probabilistic completeness means that the algorithm must find a feasible path (if ones exist) from the start to the end when the number of iterations or the search time is infinite. In the process of finding the initial path, all sampling points are located in the dynamic sampling area constrained by the angle. Since all four angle constraint ranges are $\theta \in [\theta_0, \pi]$, the sampling area also fills the entire X_{free}. A sampling sequence $\varsigma_n \left\{x^a_{\text{init}}, \ldots, x^b_{\text{init}}\right\}$ consisting of n nodes is obtained by random sampling. These points are connected to form a feasible path σ_f from the x^a_{init} to the x^b_{init}, and problem 1 is finally solved. It is well known that RRT* is a probability complete algorithm. Since DCB-RRT* performs the same sampling range as the above algorithm and the bias extension strategy allows the algorithm to make two random trees meet with faster efficiency, DCB-RRT* is probability complete.

4.2 Asymptotic Optimality

Asymptotic optimality means finding the optimal path or sub-optimal path within a finite number of iterations or search time. Since both RRT* and B-RRT* introduce the *ChooseBestParent* and *Rewire* on the basis of RRT, the path structure of the extended node x_{new} and other nodes X_{near} inside the sphere of certain radius around it are optimised, resulting in a shorter current path after each iteration. Therefore, both RRT* and B-RRT* are asymptotically optimal.

For a dual-tree structure, the path of tree T_a from the x^a_{init} to its end node x_a ($x_a \in V_a$) as σ_{f_a}: $[0,1]$, where $\sigma_{f_a}(0) = x^a_{\text{init}}$, $\sigma_{f_a}(1) = x_a$ and $\forall i \in [0,1]$, $\sigma_{f_a}(i) \in X_{\text{free}}$. Likewise, the path of tree T_b from the x^b_{init} to its end node x_b ($x_b \in V_b$) as σ_{f_b}: $[0,1]$, where $\sigma_{f_b}(0) = x^b_{\text{init}}$, $\sigma_{f_b}(1) = x_b$ and $\forall i \in [0,1]$, $\sigma_{f_b}(i) \in X_{\text{free}}$. The entire path can be represented by the following equation:

$$c(\sigma_f) = c(\sigma_{f_a}) + c(\sigma_{f_b}) + EucDis(x_a, x_b) \geq c(\sigma^*) \tag{5}$$

The *ChooseBestParent* and *Rewire* strategies are also used in DCB-RRT*, so the path structure will also be optimised when the two random trees are expanded. This means that the *m-th* iteration and the *n-th* iteration ($m > n$) will satisfy the following relation:

$$\begin{cases} c(\sigma_a)_m \leq c(\sigma_a)_n \\ c(\sigma_b)_m \leq c(\sigma_b)_n \\ \text{EucDis}(x_a, x_b)_m \leq EucDis(x_a, x_b)_n \end{cases} \tag{6}$$

where $m \in N$, $n \in N$. Therefore, as the number of iterations or search time increases, the current feasible path $c(\sigma_f)$ can converge to the optimal path $c(\sigma^*)$ and problem 2 is solved eventually. It can be concluded that DCB-RRT* is asymptotic optimality.

Suppose σ to be the optimal path with the current samples. All candidate nodes that can improve the current solution are denoted as $\widehat{X} \in X_{\text{new}}$:

$$\widehat{X} = \{x \in X \,|\, f(x) \ < c(\sigma)\} \tag{7}$$

where $f(x)$ represents the path cost. The probability of path improvement from the *m-th* iteration and the *n-th* iteration ($m > n$) can be calculated as follows:

$$\begin{aligned} P(c(\sigma_m) < c(\sigma_n)) &\leq P\left(x_m \in \widehat{X}\right) \\ &\leq P(x_m \in X_{\text{new}}) = \frac{S(X_{\text{new}})}{S(X)} \end{aligned} \tag{8}$$

where the function $S(\cdot)$ represents the area of the sampling range, the $S(X_{\text{new}})$ represents the area of the selected diamond region, the $S(X)$ represents the planning domain. Equation (8) indicates that the probability of solution improvement approaches zero as the current path tends toward the optimal solution.

If sampling is limited to a defined diamond region, (8) can be reformulated as:

$$P(c(\sigma_m) < c(\sigma_n)) \leq \frac{S(X_{\text{new}})_m}{S(X_{\text{new}})_n} \tag{9}$$

$$\frac{S(X_{\text{new}})_m}{S(X_{\text{new}})_n} < \frac{S(X_{\text{new}})}{S(X)} \tag{10}$$

From (10), there are more opportunities to improve the current solution in the selected diamond region. Furthermore, the diamond area continually shrinks with the runtime of our method, which speeds up the optimisation of path planning.

Table 1
Time and Space Complexity of the Algorithms

Complexity	Algorithms					
	RRT*	Informed-RRT*	RRT*-Smart	B-RRT*	IB-RRT*	DCB-RRT*
Time complexity	$O(N*\log N)$	$O(N*\log N)$	$O(N*\log N)$	$O(N*\log N)$	$O(N*\log N)$	$O(N*\log N)$
Space complexity	$O(N)$	$O(N)$	$O(N)$	$O(N)$	$O(N)$	$O(N)$

4.3 Time Complexity

Time complexity can qualitatively describe the running time of an algorithm, which is usually expressed in big-O notation, excluding the low-order terms and leading coefficients of this function. The time complexity of both RRT and RRT* algorithms is $O(N*\log N)$ [41]. The time complexity depends on some main steps in the algorithm. The main steps of the DCB-RRT* algorithm include *Dyn-Sample* (Line 6), *Limit-Sample* (Line 8), *Nearest* (Line 10 and 14), *DCB-Extend* (Line 11), *Near* (Line 12 and 24), *Collisionfree* (Line 14), *ChooseBestParent* (Line 21 and 26), *Insert* (Line 22), and *Rewire* (Line 23). Let the time complexity of the above steps be $T_{\mathrm{Dyn-Sample}}$, $T_{\mathrm{Limit-Sample}}$, T_{Nearest}, $T_{\mathrm{DCB-Extend}}$, T_{Near}, $T_{\mathrm{Collisionfree}}$, $T_{\mathrm{ChooseBestParent}}$, T_{Insert}, and T_{Rewire}. The total time complexity of the DCB-RRT* algorithm is $T_{\mathrm{DCB-RRT*}}$.

The number of sampling points is $n_{\max}$. Since *Dyn-Sample*, *Limit-Sample*, *DCB-Extend* can be completed in linear time, their complexity can be considered $O(N)$. The time complexity of *Nearest*, *Near,* and *Collisionfree* is proved to be $O(N*\log N)$ [42]. *ChooseBestParent* is to select the best parent node $x_{\min}$ for the extended node x_{new}, *Insert* is to insert node $x_{\min}$ into the random tree, and *Rewire* is to optimise the path structure of nodes within the sphere with a certain radius. Since the number of nodes in the sphere is small relative to the total number of samples, they can be considered to have a time complexity of $O(N)$. In general, the time complexity of the DCB-RRT* algorithm can be expressed by the following equation:

$$\begin{aligned} T_{DCB-RRT*} &= T_{Dyn-Sample} + T_{Limit-Sample} + T_{\mathrm{Nearest}} \\ &\quad + T_{DCB-Extend} + T_{\mathrm{Near}} + T_{\mathrm{Collisionfree}} \\ &\quad + T_{\mathrm{ChooseBestParent}} + T_{\mathrm{Insert}} + T_{\mathrm{Rewire}} \\ &= O(N) + O(N) + O(N*\log N) \\ &\quad + O(N) + O(N*\log N) + O(N*\log N) \\ &\quad + O(N) + O(N) + O(N) \\ &= 3O(N*\log N) + 6O(N) \\ &\quad \to O(N*\log N) \end{aligned} \tag{11}$$

Therefore the time complexity of the DCB-RRT* algorithm is $O(N*\log N)$.

4.4 Space Complexity

Space complexity is a measure of the temporary storage space occupied by the algorithm during its operation, which can also be represented by big-O notation. The space complexity of the DCB-RRT* algorithm is mainly composed of the nodes and edges on the random tree. According to the total number of sampling points $n_{\max}$, the space complexity of the DCB-RRT* algorithm can be calculated by the following equation:

$$\begin{aligned} S_{DCB-RRT*} &= S(V) + S(E) \\ &= O(N) + O(N-1) \\ &= 2O(N) - 1 \to O(N) \end{aligned} \tag{12}$$

Therefore, the space complexity of the DCB-RRT* algorithm is $O(N)$, which is the same as that of the RRT* algorithm. To understand the time and space complexity of the algorithms more intuitively, the results are shown in Table 1. Through the comparison of results, it can be seen that the proposed DCB-RRT* algorithm does not increase the time and space complexity.

5. Experimental Simulations

The proposed DCB-RRT* algorithm is used to solve path planning problems in complex environments with multi-obstacles, narrow passages, and mazes, and to verify the effectiveness of the algorithm based on the characteristics of different environments. The maps used for the simulation experiments are shown in Fig. 5, where Fig. 5(a) and (b) are scattered maps with multi-obstacles, the optimal paths in Fig. 5(c) and (d) are through narrow passages, and Fig. 5(e) and (f) are maze-like maps. Each map is composed of 500×500 pixels, where the black area represents obstacles, the starting point x_{init}^{a} is a purple point and the ending point x_{init}^{b} is a green point, to test the ability to find the optimal path while avoiding obstacles.

Simulation experiments were implemented on an Intel(R) Xeon(R) CPU E5-2620 v4 @2.10GHz×2 with 64GB RAM. The Windows 10 operating system based on the x64 processor is adopted, and the simulation software is MATLAB R2022b. The performance of DCB-RRT* is compared with other five existing algorithms, where RRT* is selected as the baseline, both the informed RRT* and the RRT*-smart algorithms speed up approaching the optimal solution by selecting high-quality sampling points, and are classic improved algorithms of RRT*, B-RRT* is a milestone dual-tree strategy algorithm, and IB-RRT* algorithm improves path convergence rate by introducing a heuristic strategy, which is similar to the improved idea of the proposed algorithm in this paper. For each map, each algorithm was run 50 times independently under the same conditions to reduce chance errors. In addition, the parameter settings used in the experiment are as follows: the initial angle of dynamic constrained sampling in DCB-RRT* was $\theta_0 = 20°$, the variation interval of θ was set to

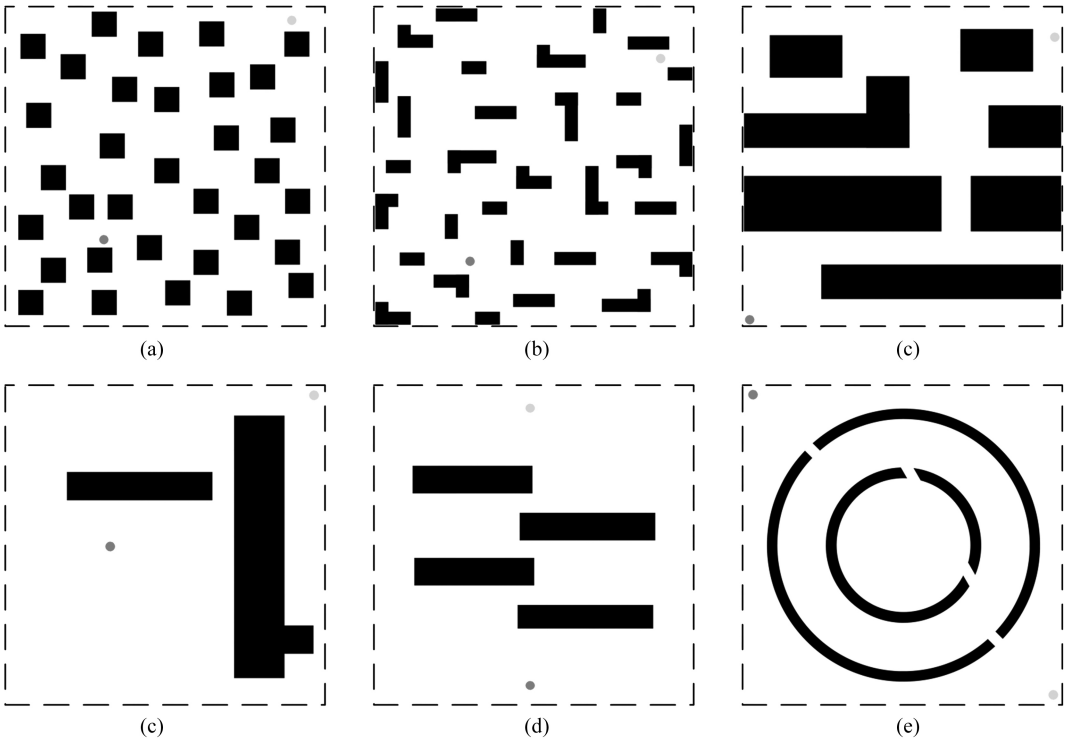

Figure 5. Maps for the simulations.

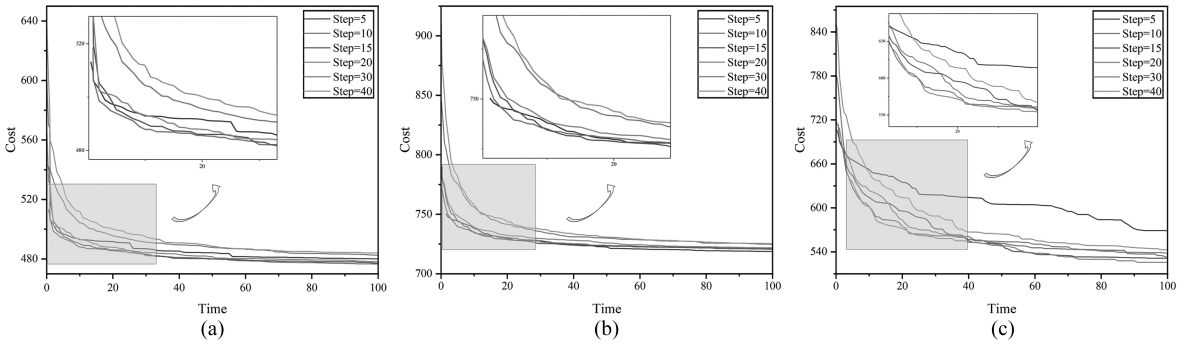

Figure 6. Convergence curves of the B-RRT* algorithm under different steps.

$\Delta = 20°$, $R = 50$ was set for all algorithms, the step of RRT* and B-RRT* was set to 10. The selection of step has a great influence on the convergence performance of the algorithms using step, such as RRT* and B-RRT*. The optimal step of the algorithm in different types of maps is also different. The proposed algorithm is improved based on B-RRT*. To select the optimal step, the B-RRT* algorithm was used to test the convergence in three types of maps (Map(a), Map(c), and Map(e)) under different steps. As shown in Fig. 6, the convergence curves show that B-RRT* exhibits the best comprehensive performance among the three types of maps when the step is 10. To compare the convergence of the six algorithms mentioned in this paper, each algorithm was executed for 100 s, returning once per second the shortest feasible path at this time. The shortest path found by RRT* within 100 s is considered as the optimal path in this paper, and the parameter $T_{1\%}$ is used to compare the convergence rate of the four algorithms. $T_{1\%}$ is the time to find a sub-optimal solution of $1.01C_{\text{optimal}}$, where C_{optimal} is the length of the optimal solution.

5.1 Ablation Experiments

To authenticate the enhancements of the three proposed aspects (*Dyn-Sample*, *Limit-Sample,* and *DCB-Extend*) in this paper, ablation experiments were conducted. The ablation experiments are the time for the algorithm to converge to the sub-optimal path. The proposed algorithm is improved based on B-RRT*, so the results of B-RRT* are used as a benchmark and verified in Map(a). In the ablation experiment, *Limit-Sample*, *Dyn-Sample,* and *DCB-Extend* are combined with B-RRT*, respectively, to verify the effectiveness of the proposed three aspects on path convergence. The results of the ablation experiment are shown in Table 2. For the convenience of marking, *Dyn-Sample* is represented by A, *Limit-Sample* is represented by B, and *DCB-Extend* is represented by C. The experimental

Table 2
The Time to Converge to Sub-Optimal Path

Time/s	Algorithms				
	B-RRT*	B-RRT*+A	B-RRT*+B	B-RRT*+C	DCB-RRT*
$T_{1\%}$	28.33	24.53	19.45	25.68	13.24

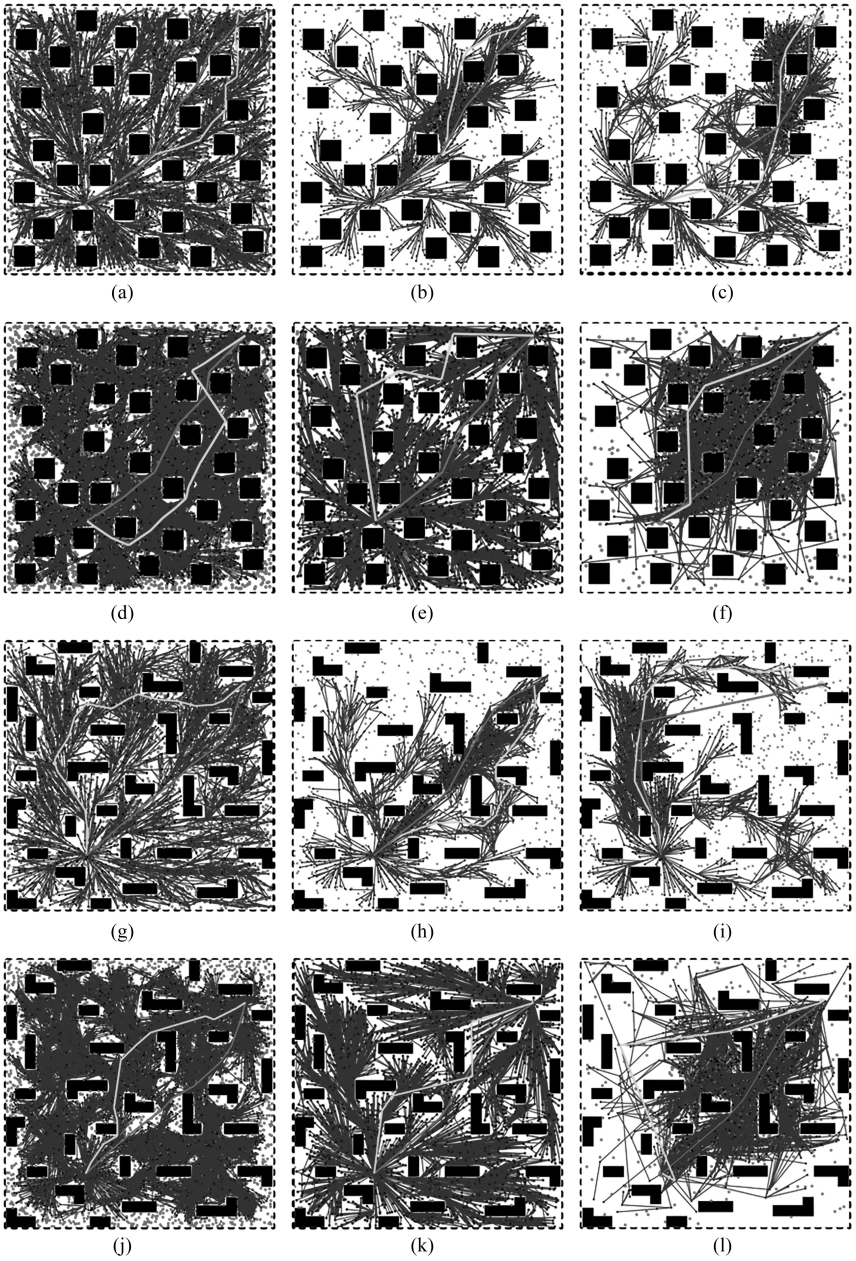

Figure 7. Representative results in multi-obstacle maps: (a) RRT*; (b) informed RRT*; (c) RRT*-smart; (d) B-RRT*; (e) IB-RRT*; (f) DCB-RRT*; (g) RRT*; (h) informed RRT*; (i) RRT*-smart; (j) B-RRT*; (k) IB-RRT*; and (l) DCB-RRT*.

results prove that the three proposed aspects have enhanced functions and can reduce the time required for the algorithm to converge to sub-optimal path.

5.2 Path Convergence Experiments

A representative set of results in the multi-obstacle map is shown in Fig. 7, where the green line is the initial path, the red line is the final path after the algorithm runs for 100 s, and the red points are random sampling points. It can be seen from the figure that since the expansion way of the random tree in RRT* and B-RRT* is a fixed step, the expansion efficiency is low and the path convergence rate is slow. Informed-RRT* improves the path convergence rate by establishing a dynamic ellipse sampling area, but the algorithm is globally randomly

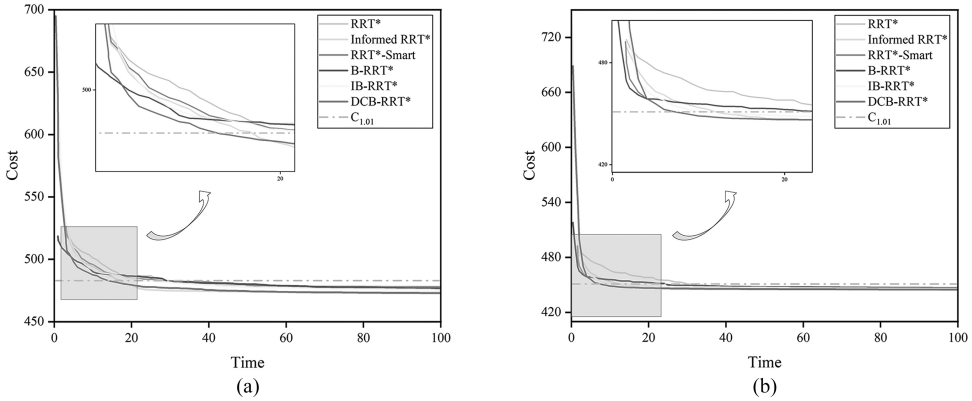

Figure 8. Path convergence curves in multi-obstacle maps.

sampling when finding the initial path with RRT*, so the initial ellipse is affected by the initial path. RRT*-smart optimises the path by generating bias points near the nodes of the initial path, and the final path generated is also greatly affected by the quality of the initial path. IB-RRT* improves search efficiency by introducing an intelligent sample insertion heuristic, but the search range is also global, so there is also the problem of sampling blindness. However, DCB-RRT* can greatly improve the quality of sampling points by dynamically constraining the sampling strategy. By forming a local convergence effect in the map, the path convergence rate is accelerated. Even in such maps with multiple channels, the algorithm has no missing probabilistic completeness, so it eventually converges to the optimal path in the case of a poor initial path. Figure 8 shows the path convergence curves of the six algorithms in two multi-obstacle maps. From the figure, it can be seen that DCB-RRT* is the fastest path convergence rate.

A representative set of results from the map with narrow passages is shown in Fig. 9. The presence of narrow passages in such maps leads to an increased probability of random tree expansion failure. The experimental results show that RRT*-smart is more advantageous in single-channel maps, because the algorithm does not need to spend more time exploring other channels to optimise the path in such maps. Compared with informed-RRT*, DCB-RRT* can make the random tree growth directional through the bias expansion strategy and can dynamically adjust the sampling area by the number of collision detection failures, which shortens the time for two trees to meet and thus reaches the path convergence stage faster. Figure 10 shows the path convergence curves of the six algorithms in narrow passage maps. It can be seen from the figure that compared with other algorithms, DCB-RRT* and RRT*-smart have certain advantages in the path convergence rate in such maps.

A representative set of results from the maze map is shown in Fig. 11. The complexity of such a map leads to the low quality of the initial paths produced by the six algorithms. The initial path causes informed-RRT* to form a larger initial ellipse area at the convergence stage, resulting in increased convergence time consumption. Similarly, the initial path generated by DCB-RRT* forms a larger angle with the starting point and ending point, resulting in a larger sampling area during the convergence stage. RRT*-smart is difficult to converge to the optimal path in such a short time due to the limitation of the sampling point method. Figure 12 shows the path convergence curves of the six algorithms in the maze maps. Despite the complexity of such maps, DCB-RRT* maintains an advantage in finding the optimal path, and the algorithm is able to find a shorter feasible path in the same running time.

The time and standard deviation of each algorithm to find the optimal path in the six maps are given in Table 3. It can be seen from the table that the path search efficiency of B-RRT*, IB-RRT*, and DCB-RRT* is higher than that of RRT* in most cases, which proves the advantage of the bidirectional strategy. Since informed-RRT* and RRT*-smart limit the sampling area in the path convergence stage, the high-quality sampling points make the path convergence rate higher than that of RRT* in multi-obstacle maps and narrow channel maps. Because the sampling points of the RRT*-smart algorithm in the convergence stage are often at the inflection points of obstacles, which makes the algorithm show great advantages in single-channel maps, RRT*-smart has the fastest convergence rate and the smallest standard deviation in map (c). However, the RRT*-smart constraint approach can make it difficult to converge in maze maps for short periods of time. The DCB-RRT* has a certain advantage in the convergence rate in different maps due to the three enhancements proposed. Among them, the proposed algorithm in the multi-obstacle map improves the average convergence rate by 61.90% over the classical B-RRT* algorithm, the proposed algorithm in the narrow passage map improves the average convergence rate by 57.96% over the classical B-RRT* algorithm, and the proposed algorithm in the maze map improves the average convergence rate by 78.44% over the classical B-RRT* algorithm. It can be seen from the standard deviation that since the convergence performance of DCB-RRT* is affected by the initial diamond size, the convergence time in maps (a) and (f) fluctuates greatly. But in general, the proposed algorithm has strong robustness, can better solve path planning problems in multiple types of complex environments, and can approach the optimal path with high search efficiency. This is of great significance to areas such as unmanned driving and intelligent sorting,

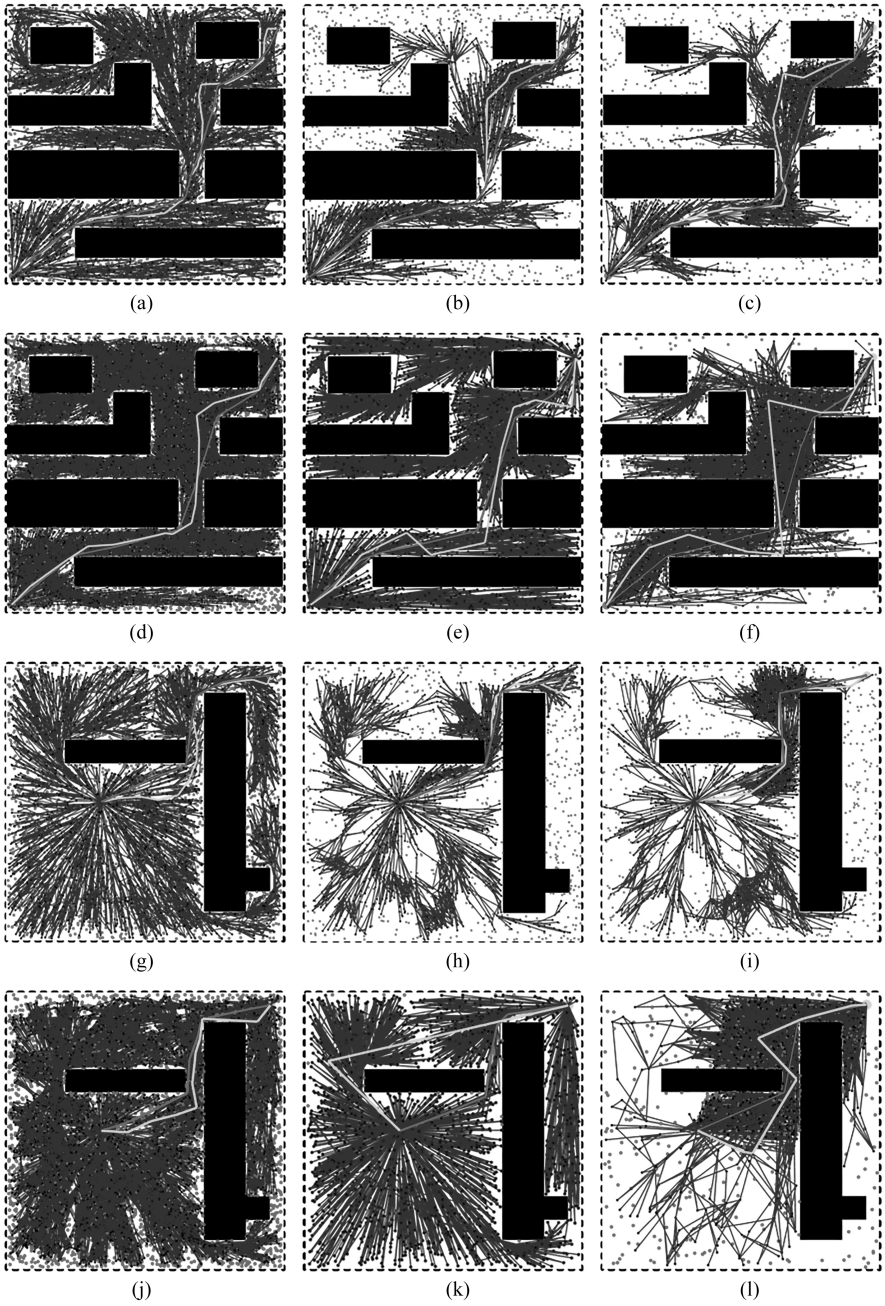

Figure 9. Representative results in narrow passages maps: (a) RRT*; (b) informed RRT*; (c) RRT*-smart; (d) B-RRT*; (e) IB-RRT*; (f) DCB-RRT*; (g) RRT*; (h) informed RRT*; (i) RRT*-smart; (j) B-RRT*; (k) IB-RRT*; and (l) DCB-RRT*.

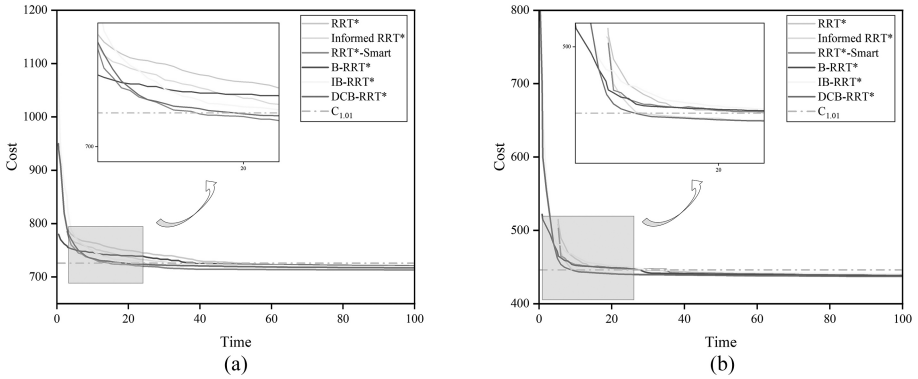

Figure 10. Path convergence curves in narrow passages maps.

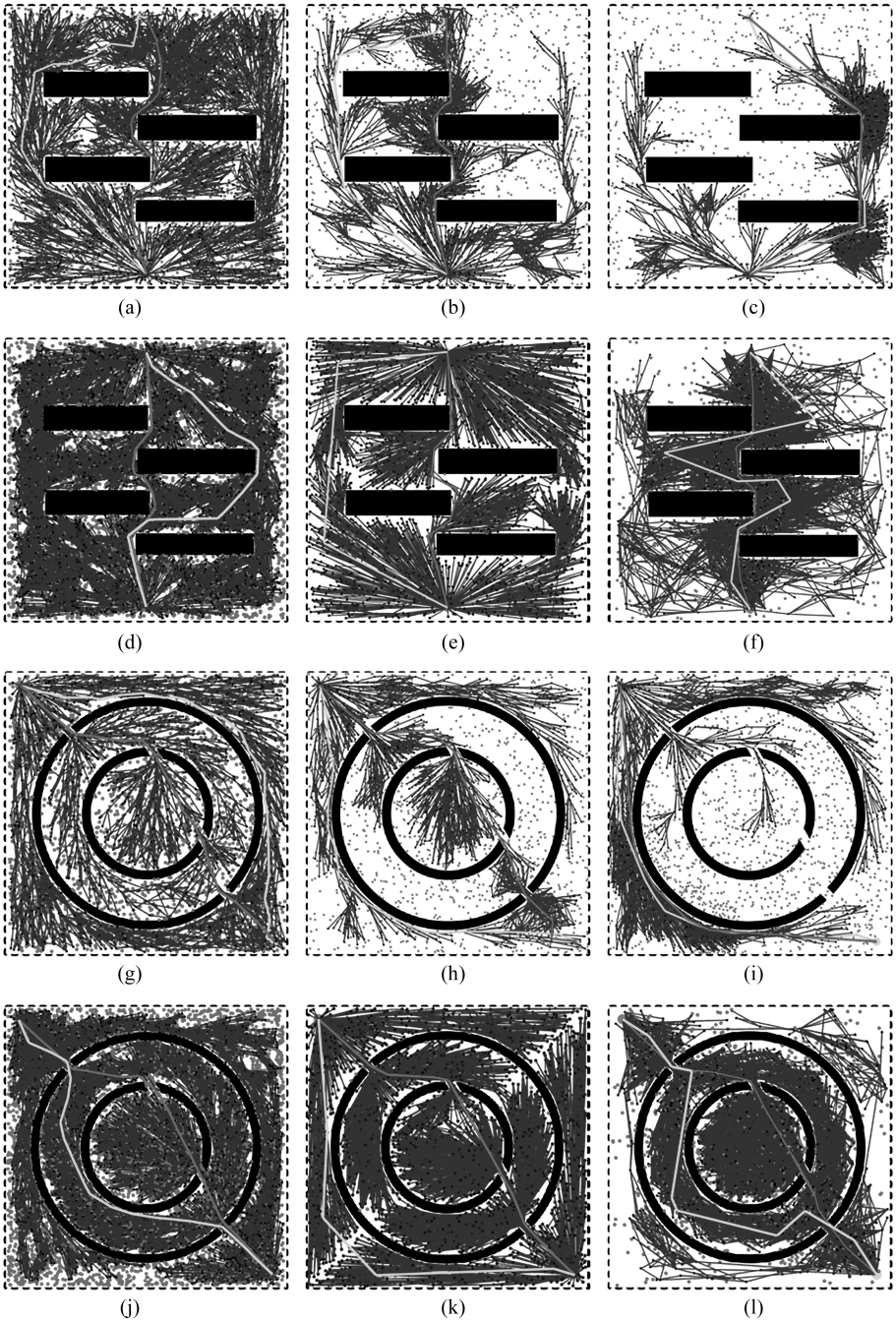

Figure 11. Representative results in maze maps : (a) RRT*; (b) informed RRT*; (c) RRT*-smart; (d) B-RRT*; (e) IB-RRT*; (f) DCB-RRT*; (g) RRT*; (h) informed RRT*; (i) RRT*-smart; (j) B-RRT*; (k) IB-RRT*; and (l) DCB-RRT*.

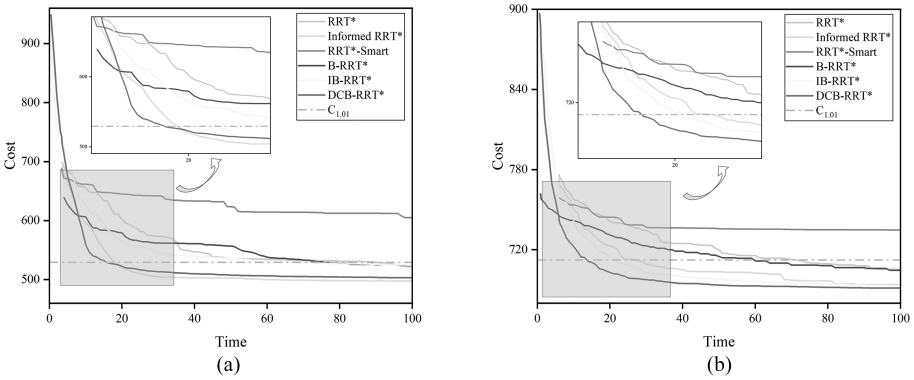

Figure 12. Path convergence curves in maze maps.

Table 3
The experimental Results on the Time of Convergence to the Optimal Path and Standard Deviation

Algorithms	RRT*		Informed-RRT*		RRT*-Smart		B-RRT*		IB-RRT*		DCB-RRT*	
Maps	Time/s											
	$T_{1\%}$	$\sigma_{1\%}$	$T_{1\%}$	$\sigma_{1\%}$	$T_{1\%}$	$\sigma_{1\%}$	$T_{1\%}$	$\sigma_{1\%}$	$T_{1\%}$	$\sigma_{1\%}$	$T_{1\%}$	$\sigma_{1\%}$
(a)	27.65	19.46	17.40	**3.93**	26.02	16.66	28.33	26.37	22.04	13.12	**13.24**	7.11
(b)	29.82	10.11	11.87	7.63	24.25	41.51	24.87	13.03	11.63	8.07	**7.33**	**2.70**
(c)	53.25	18.12	32.46	34.43	**14.72**	**3.71**	38.19	25.68	31.52	17.53	19.23	7.01
(d)	25.19	13.26	9.53	8.97	33.14	27.38	26.48	9.83	34.52	18.18	**8.93**	**4.36**
(e)	89.21	23.39	18.77	9.98	/	/	75.24	38.18	66.50	22.44	**15.41**	**7.07**
(f)	69.78	34.45	23.96	24.19	/	/	59.79	35.10	20.23	**4.00**	**13.54**	7.48

and at the same time promotes the development of artificial intelligence in various fields.

6. Conclusion

Dynamic constrained sampling based bidirectional RRT* is proposed in this paper, namely, DCB-RRT*, which has improved the convergence rate. The main work of DCB-RRT* can be divided into three points: the first point is to propose a method to dynamically adjust the sampling area according to the number of collision detection failures in the stage of finding the initial path, which improves the quality of sampling points. The second point is to propose a method of dynamic angle to limit the sampling area in the path convergence stage, which improves the path convergence rate. The third point is to propose a sampling point bias extension strategy, which increases the mutual guidance between the dual-trees and also employs dynamic steps to improve the utilisation of nodes. Compared with other classical path planning algorithms, DCB-RRT* has a more efficient convergence rate and better robustness when dealing with path planning problems in complex environments with multi-obstacles, narrow passages, and mazes. It is proved by numerical simulation experiments that it can obtain sub-optimal or optimal paths with high operational efficiency.

In the future, we will continue to research the mathematical model in *Dyn-Sample*. It would be meaningful on how to adapt the parameters according to the characteristics of the map, which would make the model perform well in different types of maps. In addition, real-time path planning in dynamic maps is a difficult and challenging task in the current research field, and we will also think about how to improve the performance of the algorithm to realise the application in dynamic maps.

Acknowledgement

This work was supported in part by the National Natural Science Foundation of China (Grant NO. 61775172) and the Hubei Key Technical Innovation Project (Grant NO. ZDCX2019000025).

References

[1] L. Zhang, Z. Liu, and X. Qin, Standards of measurement and developmental challenges in path planning for manipulator, *International Journal of Robotics and Automation*, *38*(3), 2023, 208–217.

[2] J.J. Potter, K.L. Sorensen, and W.E. Singhose, Efficient method for generating pick-and-place trajectory over obstacles, *International Journal of Robotics and Automation*, *10*(2), 2010, 6–24.

[3] L. Deng, X. Ma, J. Gu, Y. Li, Z. Xu, and Y. Wang, Artificial immune network-based multi-robot formation path planning with obstacle avoidance, *International Journal of Robotics and Automation*, *31*(3), 2016, 225–232.

[4] J. Chen, F. Ling, Y. Zhang, T. You, Y. Liu, and X. Du, Coverage path planning of heterogeneous unmanned aerial vehicles based on ant colony system, *Swarm and Evolutionary Computation*, *69*, 2022, 101005.

[5] N. Wang, Y. Zhang, C.K. Ahn, and Q. Xu, Autonomous pilot of unmanned surface vehicles: Bridging path planning and tracking, *IEEE Transactions on Vehicular Technology*, *71*(3), 2021, 2358–2374.

[6] L. Sun, Z. Fu, F. Tao, P. Si, S. Song, and C. Sun, APF–BUG-BASED intelligent path planning for autonomous vehicle with high precision in complex environment, *International Journal of Robotics and Automation*, *38*(4), 2023, 277–283.

[7] W. Wenna, D. Weili, H. Changchun, Z. Heng, F. Haibing, and Y. Yao, A digital twin for 3D path planning of large-span curved-arm gantry robot, *Robotics and Computer-Integrated Manufacturing*, *76*, 2022, 102330.

[8] T. Xue, L. Li, L. Shuang, D. Zhiping, and P. Ming, Path planning of mobile robot based on improved ant colony algorithm for logistics, *Mathematical Biosciences and Engineering*, *18*(4), 2021, 3034–3045.

[9] X. Liu, D. Jiang, B. Tao, G. Jiang, Y. Sun, J. Kong, X. Tong, G. Zhao, and B. Chen, Genetic algorithm-based trajectory optimization for digital twin robots, *Frontiers in Bioengineering and Biotechnology*, *9*, 2022, 1433.

[10] Y. Zhao, Y. Wang, J. Zhang, X. Liu, Y. Li, S. Guo, X. Yang, and S. Hong, Surgical GAN: Towards real-time path planning for passive flexible tools in endovascular surgeries, *Neurocomputing*, *500*, 2022, 567–580.

[11] K. Wu, B. Li, Y. Zhang, and X. Dai, Review of research on path planning and control methods of flexible steerable needle puncture robot, *Computer Assisted Surgery*, *27*(1), 2022, 91–112.

[12] Y. Wang, Z. He, D. Cao, L. Ma, K. Li, L. Jia, and Y. Cui, Coverage path planning for kiwifruit picking robots based on deep reinforcement learning, *Computers and Electronics in Agriculture*, *205*, 2023, 107593.

[13] G. Lin, L. Zhu, J. Li, X. Zou, and Y. Tang, Collision-free path planning for a guava-harvesting robot based on recurrent

deep reinforcement learning, *Computers and Electronics in Agriculture*, *188*, 2021, 106350.
[14] C. He, C. Deng, N. Li, and Z. Miao, Design of vision control system of tomato picking robot, *Proc. 2021 40th Chinese Control Conf. (CCC)*, Shanghai, 2021, 4267–4271.
[15] K. Karur, N. Sharma, C. Dharmatti, and J.E. Siegel, A survey of path planning algorithms for mobile robots, *Vehicles*, *3*(3), 2021, 448–468.
[16] S. Katoch, S.S. Chauhan, and V. Kumar, A review on genetic algorithm: Past, present, and future, *Multimedia Tools and Applications*, *80*, 2021, 8091–8126.
[17] Y. Liang and L. Wang, Applying genetic algorithm and ant colony optimization algorithm into marine investigation path planning model, *Soft Computing*, *24*, 2020, 8199–8210.
[18] Y. Ding, B. Xin, L. Dou, J. Chen, and B.M. Chen, A memetic algorithm for curvature-constrained path planning of messenger UAV in air-ground coordination, *IEEE Transactions on Automation Science and Engineering*, *19*(4), 2021, 3735–3749.
[19] J. Li, T. Sun, X. Huang, L. Ma, Q. Lin, J. Chen, and V.C. Leung, A memetic path planning algorithm for unmanned air/ground vehicle cooperative detection systems, *IEEE Transactions on Automation Science and Engineering*, *19*(4), 2021, 2724–2737.
[20] C. Huang, X. Zhou, X. Ran, J. Wang, H. Chen, and W. Deng, Adaptive cylinder vector particle swarm optimization with differential evolution for UAV path planning, *Engineering Applications of Artificial Intelligence*, *121*, 2023, 105942.
[21] Z. He, X. Tang, Q. Shen, C. Duan, and C. Jia, Optimisation of a six-degree-of-freedom robot trajectory based on improved multi-objective PSO algorithm, *International Journal of Robotics and Automation*, *38*(3), 2023, 218–230.
[22] Q. Luo, H. Wang, Y. Zheng, and J. He, Research on path planning of mobile robot based on improved ant colony algorithm, *Neural Computing and Applications*, *32*, 2020, 1555–1566.
[23] C. Miao, G. Chen, C. Yan, and Y. Wu, Path planning optimization of indoor mobile robot based on adaptive ant colony algorithm, *Computers and Industrial Engineering*, *156*, 2021, 107230.
[24] O. Khatib, Real-time obstacle avoidance for manipulators and mobile robots, *The International Journal of Robotics Research*, *5*(1), 1986, 90–98.
[25] Y. Huang, H. Ding, Y. Zhang, H. Wang, D. Cao, N. Xu, and C. Hu, A motion planning and tracking framework for autonomous vehicles based on artificial potential field elaborated resistance network approach, *IEEE Transactions on Industrial Electronics*, *67*(2), 2019, 1376–1386.
[26] G. Tang, C. Tang, C. Claramunt, X. Hu, and P. Zhou, Geometric A-star algorithm: An improved A-star algorithm for AGV path planning in a port environment, *IEEE Access*, *9*, 2021, 59196–59210.
[27] S. Koenig and M. Likhachev, Fast replanning for navigation in unknown terrain, *IEEE Transactions on Robotics*, *21*(3), 2005, 354–363.
[28] G. Chen, N. Luo, D. Liu, Z. Zhao, and C. Liang, Path planning for manipulators based on an improved probabilistic roadmap method, *Robotics and Computer-Integrated Manufacturing*, *72*, 2021, 102196.
[29] S.M. LaValle and J.J. Kuffner Jr., Randomized kinodynamic planning, *The International Journal of Robotics Research*, *20*(5), 2001, 378–400.
[30] L.R. Celsi and M.R. Celsi, On edge-lazy RRT collision checking in sampling-based motion planning, *International Journal of Robotics and Automation*, *36*(4), 2021, 240–245.
[31] S. Karaman, M.R. Walter, A. Perez, E. Frazzoli, and S. Teller, Anytime motion planning using the RRT, *Proc. 2011 IEEE International Conf. on Robotics and Automation*, Shanghai, 2011, 1478–1483.
[32] C. Urmson and R. Simmons, Approaches for heuristically biasing RRT growth, *Proc. 2003 IEEE/RSJ International Conf. on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, Las Vegas, NV, , 2003, 1178–1183.
[33] J.D. Gammell, S.S. Srinivasa, and T.D. Barfoot, Informed RRT: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic, *Proc. 2014 IEEE/RSJ International Conf. on Intelligent Robots and Systems*, Chicago, IL, 2014, 2997–3004.
[34] F. Islam, J. Nasir, U. Malik, Y. Ayaz, and O. Hasan, RRT*-smart: Rapid convergence implementation of RRT* towards optimal solution, *Proc. 2012 IEEE International Conf. on Mechatronics and Automation*, Chengdu, 2012, 1651–1656.
[35] M. Jordan and A. Perez, Optimal bidirectional rapidly-exploring random trees, Technical Report MIT-CSAIL-TR-2013-021, Computer Science and Artificial Intelligence Laboratory, Cambridge, 2013.
[36] A.H. Qureshi and Y. Ayaz, Intelligent bidirectional rapidly-exploring random trees for optimal motion planning in complex cluttered environments, *Robotics and Autonomous Systems*, *68*, 2015, 1–11.
[37] Z. Tahir, A.H. Qureshi, Y. Ayaz, and R. Nawaz, Potentially guided bidirectionalized RRT* for fast optimal path planning in cluttered environments, *Robotics and Autonomous Systems*, *108*, 2018, 13–27.
[38] N. Chao, Y.-k. Liu, H. Xia, A. Ayodeji, and L. Bai, Grid-based RRT* for minimum dose walking path-planning in complex radioactive environments, *Annals of Nuclear Energy*, *115*, 2018, 73–82.
[39] J.J. Kuffner and S.M. LaValle, RRT-connect: An efficient approach to single-query path planning, *Proc. 2000 ICRA. Millennium Conf.*, San Francisco, CA, 2000, 995–1001.
[40] J. Nasir, F. Islam, U. Malik, Y. Ayaz, O. Hasan, M. Khan, and M.S. Muhammad, RRT*-SMART: A rapid convergence implementation of RRT, *International Journal of Advanced Robotic Systems*, *10*(7), 2013, 299.
[41] I.-B. Jeong, S.-J. Lee, and J.-H. Kim, Quick-RRT*: Triangular inequality-based implementation of RRT* with improved initial solution and convergence rate, *Expert Systems with Applications*, *123*, 2019, 82–90.
[42] S. Karaman and E. Frazzoli, Incremental sampling-based algorithms for optimal motion planning, *Robotics Science and Systems VI*, *104*(2), 2010.

Biographies

Xining Cui received the M.S. degree from Shandong University of Science and Technology, Qindao, China, in 2021. He is currently pursuing the Ph.D. degree in Control Science and Engineering with the Wuhan University of Science and Technology, Wuhan, China. His research interests include computer vision, structured light-based sensing and robotic arm grasping.

Caiqi Wang received the B.S. degree from Wuhan University of Science and Technology, Wuhan, China. He is currently pursuing the master's degree in electronic information with Wuhan University of Science and Technology. His research interests include path planning and artificial intelligence.

Yi Xiong received the B.E. degree from Wuhan Textile University, Wuhan, China, in 2022. He is currently pursuing the M.E. degree with Wuhan University of Science and Technology, Wuhan, China. His research interests include computer vision, point cloud registration, and robotic arm manipulation.

Dr. Shiqian Wu received the B.Eng. and M.Eng. degrees from Huazhong University of Science and Technology (HUST), Wuhan, China, in 1985 and 1988, respectively, and the Ph.D. degree from Nanyang Technological University, Singapore, in 2001. He is a Professor with School of Information Science and Engineering, the Deputy Director with the Institute of Robotics & Intelligent Systems (IRIS), Wuhan University of Science and Technology (WUST), Wuhan, and the Director with Hubei Province Key Laboratory of Intelligent Information Processing and Real-Time Industrial Systems, Wuhan, China. Prior to join WUST, he was an Assistant Professor, Lecturer, and an Associate Professor with HUST from 1988 to 1997. From 2000 to 2014, he was a Research Fellow or Research Scientist with Agency for Science, Technology and Research (A-STAR), Singapore. He has co-authored two books and more than 250 scientific publications (book chapters and journal/conference papers). His research interests include Computer vision, Pattern Recognition, Robotics and artificial intelligence. He was the recipient of BEST 10% PAPER award in ICIP 2015, and the recipient of Best Paper Finalist award in ICIEA 2020.