drones

MDPI

Article

Efficient Multi-UAV Path Planning Using Locally Refinable Channel Graph in Cluttered Environments

Haoyu Tian [1], Linghao Li [1,2], Guanghui Sun [1] and Weiran Yao [1,*]

[1] School of Astronautics, Harbin Institute of Technology, Harbin 150001, China; 22b904019@stu.hit.edu.cn (H.T.); 23b936073@stu.hit.edu.cn (L.L.); guanghuisun@hit.edu.cn (G.S.)
[2] Marine Military Intelligence Innovation Institute, CSSC Systems Engineering Research Institute, Beijing 100036, China
* Correspondence: yaoweiran@hit.edu.cn

Highlights

What are the main findings?

- A novel locally refinable channel graph (LRCG) model and its extension scheme are proposed to model the search space of complex environments.
- An LRCG-based multi-UAV path planning framework is designed, which includes an LRCG hybrid conflict-based search algorithm and a constrained dynamic programming search algorithm for initial solution construction and post-processing, respectively.

What are the implications of the main findings?

- The proposed LRCG-based planning method effectively addresses the multi-UAV path planning problem in cluttered environments while ensuring that safe distance constraints between UAVs are satisfied.
- Lightweight and scalable environment modeling enables the LRCG-based planning method to achieve higher computational efficiency without sacrificing solution quality, while also exhibiting good parameter robustness.

Abstract

In multi-unmanned-aerial-vehicle (UAV) collaborative task scenarios, congestion in cluttered environments significantly hinder operational efficiency, leading to the multi-UAV path planning problem. To reduce the scale of the search space while ensuring the quality of the planned paths, this paper proposes a locally refinable channel graph (LRCG) model to describe the accessible region of the occupancy grid maps, aiming to address the issue of large search spaces adversely affecting planning efficiency. As the basic element of the LRCG, the channel edge represents the local accessible region in the form of an intersecting circle sequence, and can be expanded into multiple collision-free sub-paths in a targeted manner. An LRCG hybrid conflict-based search (LRCG-HCBS) algorithm is proposed to refine the LRCG oriented to conflicts and plan a set of initial collision-free paths. Further, this paper designs a post-processing optimization algorithm for a single path, a constrained dynamic programming search (CDPS) algorithm, which is also employed for the preprocessing refinement of the LRCG. Simulation results show that, compared with existing algorithms, the LRCG-based planning algorithm proposed in this paper demonstrates higher computational efficiency.

Keywords: multi-UAV systems; multi-UAV path planning; locally refinable channel graph; conflict-based search

Academic Editor: Octavio Garcia-Salazar

Received: 30 May 2026
Revised: 10 July 2026
Accepted: 14 July 2026
Published: 15 July 2026

1. Introduction

In obstacle-dense environments with restricted vertical maneuverability, such as indoor corridors and beneath dense forest canopies, effective multi-unmanned-aerial-vehicle (UAV) path planning serves as a crucial guarantee for efficient coordination among UAVs, mitigating congestion and collision risks, minimizing delays, and optimizing resource utilization and task execution efficiency [1,2]. However, in cluttered task environments characterized by high obstacle density, narrow corridors, large map sizes, limited free-space ratios, and high congestion levels among UAVs, the environmental map model becomes more complex, resulting in a larger search space and significantly degrading the efficiency of planning algorithms, which brings challenges to practical applications.

At present, researchers have achieved significant progress in the fields of multi-robot path planning and multi-agent path finding (MAPF), proposing a variety of planning algorithms. For multi-robot path planning problems in discrete graph-based search spaces, the solving algorithms can be divided into three categories: coupled algorithms, decoupled algorithms, and hybrid algorithms [3]. Among them, hybrid algorithms, represented by conflict-based search (CBS) [4] algorithms, are a mainstream class of approaches that decomposes the overall problem into high-level search and low-level constrained path planning for individual agents. Derived from the CBS algorithm, improved versions such as Enhanced CBS [5], Meta-Agent CBS [6], ICBS [7], and multi-objective CBS [2] have been proposed. In contrast to conflict-oriented planning algorithms such as CBS, priority-oriented planning algorithms have been proposed as bounded suboptimal planner, such as priority-based search (PBS) [8], priority inheritance with backtracking (PIBT) [9], and dueling priority-based search (D-PBS) [10]. In addition, large neighborhood search (LNS) [11], LNS2 [12] and lazy constraints addition search for MAPF (LaCAM) [13] have been proposed.

The aforementioned planning algorithms are generally based on fixed graph-based or grid-based search spaces, where an increase in the size of the search space notably affects the efficiency of the planning algorithms. For complex environments, where large-scale search spaces lead to real-time performance challenges, sampling-based search methods such as rapidly exploring random trees (RRTs) [14] and probabilistic road maps (PRMs) [15] have been introduced into the study of multi-robot path planning, replacing grid-based search methods in the low-level search. These sampling-based MRPP methods can effectively improve planning efficiency, but they also come with inherent uncertainties and require lengthy sampling and optimization processes.

Additionally, enhanced search-space models, including the generalized Voronoi diagram (GVD) [16,17], have been developed and utilized in both single-robot and multi-robot path planning scenarios. The main strategy is to adopt a small-scale skeletonized topological structure as the search space for planning or to guide subsequent path optimization. However, the issue with such methods is that they potentially omit information about accessible areas or confine paths to certain key regions of the original map [17].

To overcome the dimensional explosion and efficiency degradation that plague state-of-the-art multi-UAV path planning methods in complex scenarios, this paper proposes a novel planning framework based on locally refinable channel graph (LRCG), which enhances planning efficiency from the perspective of search-space modeling. In the LRCG, the accessible region of the environment is stored as refinable channel edges, with little loss of accessible area compared to the original grid map. Meanwhile, considering the situation where multiple UAVs move in the same local area, the channel edges can be refined into multiple sub-path edges to effectively utilize the accessible region so that the planned path is not limited to the center of the channel to avoid collisions. Simulations demonstrate

the efficiency advantage of the proposed method over existing methods and its enhanced robustness over sampling-based methods. The contributions of this study are as follows:

1. An LRCG model is established as an improved lightweight and scalable model derived from the GVD-based road map to represent the accessible region of the occupancy grid map, and its construction method, channel sub-path homotopy, and local refinement method are designed.
2. Inspired by the search idea of CBS, an LRCG hybrid conflict-based search (LRCG-HCBS) algorithm is designed on the LRCG model to generate an initial solution that satisfies the collision avoidance constraints, which includes a three-stage search algorithm and a conflict-oriented refinement algorithm.
3. Based on the basic procedure of Space-Time A*, a constrained dynamic programming search (CDPS) algorithm is proposed based on the channel model of the LRCG for post-processing optimization of the initial paths.

The remainder of this paper is organized as follows. Section 2 reviews the related research works in the field of multi-UAV path planning. Problem statement is given in Section 3, which establishes the model of multi-UAV path planning problem. Section 4 introduces the LRCG model and its construction and local refinement method. The LRCG-based planning algorithms are presented in Section 5, including the LRCG-HCBS algorithm and CDPS algorithm. Section 6 presents the results of the simulation verification. The conclusion of this paper is in Section 7.

2. Related Works

In the field of MVPP and MAPF problems, the distinction among coupled algorithms [18], decoupled algorithms [19], and hybrid algorithms [4] primarily lies in the construction of the search space. Coupled algorithms construct a unified configuration space using the states of multiple agents and employ search algorithms, integer linear programming, and other methods to explore feasible solutions. On the other hand, decoupled algorithms plan a set of optimal paths in the single-agent configuration space individually, and design different conflict resolution strategies to avoid collisions between robots, such as setting priorities and adjusting robot velocities. Hybrid algorithms, represented by conflict-based search (CBS) [4] algorithm, establish a search space at the swarm level and incorporate the conflict resolution strategies. The CBS algorithm divides multi-agent path planning into a high-level binary tree search and low-level single-agent constrained path planning and achieves inter-agent collision avoidance by constructing the constraint set. Based on the CBS algorithm, the ECBS algorithm [5] adopts a focal search method in the search procedure, which ensures that the obtained solution is within the constant factor range of the optimal solution. In addition, a CBS-TA algorithm [20] is proposed to solve the multi-robot collision-free task allocation and planning problem. Among priority-oriented planning algorithms, the priority-based search (PBS) [8] algorithm guides the path search process by adjusting the priority relationship between agents to efficiently obtain a feasible solution. Based on the PBS algorithm, the dueling priority-based search (D-PBS) [10] algorithm was proposed to solve the multiple nonholonomic robots motion planning problem in congested environments. Priority inheritance with backtracking (PIBT) [9] algorithm was proposed to iteratively solve feasible suboptimal paths based on an adaptive prioritization scheme. Large neighborhood search (LNS) [11] and LNS2 [12] iteratively resolve path conflicts by repeatedly selecting subsets of conflicting agents and replanning to obtain conflict-free paths. The lazy constraints addition search for MAPF (LaCAM) [13] algorithm adopts a two-layer search framework, searching for the configurations and constraints of agents at the high and low levels, respectively, and can be applied to problems such as multi-robot motion planning. Han et al. proposed a DDM algorithm based on two

heuristics, path diversification and optimal sub-problem solution databases [21]. A graph-attention-network-based method [22] was proposed to solve the large-scale multi-robot path planning problem. For multi-UAV cooperative path planning, Yang et al. further modeled the UAV team as a dynamic graph and combined graph neural networks with proximal policy optimization to capture inter-UAV interactions while improving scalability in high-density scenarios [23].

In the application scenarios of large-scale maps, planning methods often encounter the challenge of reduced search efficiency due to the expansive search space. This challenge is further amplified in multi-UAV missions, where waypoint allocation, path optimization, obstacle avoidance, and workload balance are often coupled in large 3D environments [24,25]. The dynamic constraints of robots render discrete point sequence path, such as those based on grid maps or graph planning, unsuitable for the task execution of the robots [26,27]. The planning problem considering the dynamic constraints can be referred to as a multi-robot motion planning (MRMP) problem.

In the study of planning methods considering dynamic constraints, the main solution idea is to obtain paths that satisfies the constraints in the path planning method of individual robots. Le et al. proposed a coordinated expansion method for the motion tree, which includes a sampling-based constrained motion planning method for an individual robot to follow the desired path and a multi-path adjustment strategy. Kinodynamic Conflict-Based Search (K-CBS) algorithm [27] and chance-constrained K-CBS algorithm [28] are proposed, in which an extension method of sampling-based methods is designed to plan paths that satisfy both collision avoidance and dynamic constraints. For discontinuity-bounded constraints, a multi-robot motion planning method, dbCBS [29], which combines the CBS and the discontinuity-bounded A* algorithm is proposed. Different from the above-mentioned planning methods, decentralized multi-robot motion planning algorithms have been proposed, such as decentralized real-time asynchronous probabilistic trajectory planning algorithm for multirobot teams (DREAM) [30] proposed by Şenbaşlar et al. For adaptive UAV data-collection tasks, Westheider et al. proposed a multi-agent deep reinforcement learning method that used counterfactual policy gradients to learn cooperative informative path planning under team-size and communication constraints [31].

To address the efficiency problem of large search space, an effective strategy is to introduce sampling-based methods to multi-robot path planning, such as rapidly exploring random trees (RRTs) [14,32], probabilistic road maps (PRMs) [3,15,33]. Shome et al. proposed the dRRT* algorithm [32], which introduced a method of constructing road maps separately for the robots, and achieved asymptotically optimal multi-robot motion planning. Solis et al. combined PRMs with CBS search and designed the CBS-MP method [3]. An iterative refinement strategy [34] was proposed to gradually optimize the initial planned paths. A sampling scheme based on staggered grids [35] was proposed, which could achieve high-quality near-optimal solutions under finite sampling. In complex urban 3D UAV environments, Liu et al. integrated task allocation, enhanced ant colony optimization, and an A*/RRT-based obstacle avoidance module to generate collision-free paths with reduced flight time and energy consumption [24].

In addition, a variety of improved search-space models have been proposed to solve planning problems efficiently. Chi et al. adopted the generalized Voronoi diagram model (GVD) as the representation of the obstacle-free region, designed a feature extraction algorithm to reduce its redundancy, and used the preliminary planned path within the GVD as the heuristic path for the subsequent sampling-based planning [16]. Aiming at the congestion risk of multi-robot path planning, Fu et al. established a virtual velocity field model based on the congestion probability field model to guide path optimization [36]. A surplus topology generation algorithm (STGA) [17] was proposed to generate a global topology

map based on grid maps, making special modeling for special areas such as narrow straight roads and crossroads, thus improving planning efficiency and security compared to the original grid map model. For scalable multi-UAV waypoint planning, Debnath et al. formulated the problem from the multi-traveling salesman perspective and proposed DECK-GA, which combines dynamic-centroid K-means clustering with a distance-efficient genetic algorithm to improve cluster stability and global route optimization [25].

3. Problem Statement

In the multi-UAV path planning problem considered in this study, the UAVs are assumed to operate within a prescribed altitude layer, so their vertical motion is restricted, and the planning problem is formulated on a two-dimensional horizontal plane. The environment is modeled as a fixed map with static obstacles, and dynamic environmental changes, sensing uncertainty, communication delay, and full flight-dynamic constraints are not explicitly considered in the present formulation. For safety assurance, a safe separation distance must be maintained between UAVs. Let $\mathcal{X} \subset \mathbb{R}^2$ be the UAV state space and $\mathcal{X}_{\mathrm{OBS}}$ the space occupied by obstacles, then the free space is expressed as $\mathcal{X}_{\mathrm{FREE}} = \mathcal{X} \backslash \mathcal{X}_{\mathrm{OBS}}$. Let the number of UAVs be M and the minimum distance between the UAVs be $d_{\min}$. The UAVs are set to have equal velocity. The multi-UAV path planning problem is an optimization problem of minimizing the total trajectory time across all UAVs, which can be expressed as:

$$\min \sum_{i=1}^{M} \mu_{\mathrm{L}}(\psi_i) \tag{1}$$

subject to

$$\psi_i(0) = \mathbf{s}_i, \qquad i = 1, 2, \ldots, M, \tag{2}$$

$$\psi_i(T_{\mathrm{F}i}) = \mathbf{g}_i, \qquad i = 1, 2, \ldots, M, \tag{3}$$

$$\psi_i(t) \in \mathcal{X}_{\mathrm{FREE}}, \qquad t \in [0, T_{\mathrm{F}i}], i = 1, 2, \ldots, M, \tag{4}$$

$$\left\| \psi_i(t) - \psi_j(t) \right\| \geq d_{\min}, \qquad i, j = 1, 2, \ldots, M, i \neq j, \tag{5}$$

where $\psi_i : [0, T_{\mathrm{F}i}] \rightarrow \mathcal{X}_{\mathrm{FREE}}$ represents the planned trajectory of UAV i, and $T_{\mathrm{F}i}$ is the time of arrival of UAV i. $\mathbf{s}_i \in \mathcal{X}_{\mathrm{FREE}}$ and $\mathbf{g}_i \in \mathcal{X}_{\mathrm{FREE}}$ are the starting point and the goal point of UAV i, respectively. $\mu_{\mathrm{L}} : \mathcal{X}_{\psi} \rightarrow \mathbb{R}^+$ is a mapping from trajectories to trajectory time, where $\mathcal{X}_{\psi}$ is the space of trajectory function ψ_i. $d_{\min}$ is the minimum distance between UAVs.

The solution for multi-UAV path planning ψ_i can generally be simplified as a finite sequence of time and position, expressed as

$$\begin{cases} \tau_i(k) = t_k, & k = 0, 1, \ldots, N_{\phi}(\phi_i), \\ \phi_i(k) = (x_k, y_k)^T, & k = 0, 1, \ldots, N_{\phi}(\phi_i), \end{cases} \tag{6}$$

subject to

$$\begin{cases} \tau_i(0) = 0, & \phi_i(0) = \mathbf{s}_i, \\ \tau_i\big(N_{\phi}(\phi_i)\big) = T_{\mathrm{F}i}, & \phi_i\big(N_{\phi}(\phi_i)\big) = \mathbf{g}_i. \end{cases} \tag{7}$$

where $N_{\phi}(\phi_i)$ is the number of path segments. The relationship between the UAV trajectory and the point sequence can be expressed as

$$\begin{cases} \psi_i(t) = (1 - \delta)\phi_i(k) + \delta\phi_i(k+1), t \in [\tau_i(k), \tau_i(k+1)), \\ \delta = (t - \tau_i(k))/(\tau_i(k+1) - \tau_i(k)), t \in [\tau_i(k), \tau_i(k+1)), \end{cases} \tag{8}$$

where $k = 0, 1, \ldots, N_{\phi}(\phi_i) - 1$.

4. Locally Refinable Channel Graph

In this study, we propose a locally refinable channel graph (LRCG) model for lightweight description of the accessible domain of occupancy grid maps (OGMs). The locally refinable channel graph model applies the generalized Voronoi diagram (GVD) as the initial skeleton and establishes the channel graph model through the channel construction algorithm. After obtaining the initial channel graph, we design a channel sub-path homotopy structure and a channel refinement method to derive multiple sub-channels from the initial channel for more efficient path planning.

4.1. LRCG Construction

The construction method of the LRCG first generates a GVD from the occupancy grid map model, constructs an initial GVD topological graph by extracting key nodes as vertices, and finally establishes the LRCG model through the channel construction method. The occupancy grid map used in this process is derived from the original map by applying an inflation operation to the obstacles, with the inflation radius set to the radius of the UAVs. The construction of the GVD can adopt an approximate GVD construction method based on the occupancy grid map [37]. The GVD of a two-dimensional occupancy grid map is defined as a set of points in $\mathcal{X}_{\mathrm{FREE}}$ that are equidistant from nearest obstacles, denoted as $\mathcal{X}_{\mathrm{GVD}}$.

The initial GVD topological graph extracted from the GVD can be denoted as

$$\begin{cases} \mathcal{G}_{\mathrm{GVD}} = \langle \mathcal{V}_{\mathrm{GVD}}, \mathcal{E}_{\mathrm{GVD}} \rangle, \\ \mathcal{V}_{\mathrm{GVD}} = \mathcal{V}_{\mathrm{JOINT}} \cup \mathcal{V}_{\mathrm{TERM}} = \{V_i\}_{i=1,2,\ldots,N_{\mathrm{GVD}}}, \\ \mathcal{E}_{\mathrm{GVD}} = \left\{ E_{\mathrm{GVD}}^{(j_1,j_2)} | j_1, j_2 \in \{1,2,\ldots,N_{\mathrm{GVD}}\}, j_1 \neq j_2 \right\}, \end{cases} \tag{9}$$

where $\mathcal{V}_{\mathrm{GVD}}$ and $\mathcal{E}_{\mathrm{GVD}}$ are the vertex set and edge set of graph $\mathcal{G}_{\mathrm{GVD}}$, N_{GVD} is the number of vertices, and $E_{\mathrm{GVD}}^{(j_1,j_2)}$ represents the edge from vertex j_1 to vertex j_2. The joint vertex set $\mathcal{V}_{\mathrm{JOINT}}$ is defined as the set of points that are equidistant from three or more obstacles and have multiple adjacent points in the $\mathcal{X}_{\mathrm{GVD}}$. The terminal vertex set $\mathcal{V}_{\mathrm{TERM}}$ is defined as the set of points that have only one adjacent point in the $\mathcal{X}_{\mathrm{GVD}}$. We denote the number of joint vertex as N_{JOINT} and the number of terminal vertex as N_{TERM}. Let $S_{\mathrm{NEIGH}}(\mathbf{x}, S)$ be the adjacent point set of point $\mathbf{x}$ in set S; the vertex set is defined as

$$\begin{cases} \mathcal{V}_{\mathrm{GVD}} = \mathcal{V}_{\mathrm{JOINT}} \cup \mathcal{V}_{\mathrm{TERM}}, \\ \mathcal{V}_{\mathrm{JOINT}} = \{\mathbf{x} | \mathbf{x} \in \mathcal{X}_{\mathrm{GVD}}, |S_{\mathrm{NEIGH}}(\mathbf{x}, \mathcal{X}_{\mathrm{GVD}})| \geq 3\}, \\ \mathcal{V}_{\mathrm{TERM}} = \{\mathbf{x} | \mathbf{x} \in \mathcal{X}_{\mathrm{GVD}}, |S_{\mathrm{NEIGH}}(\mathbf{x}, \mathcal{X}_{\mathrm{GVD}})| = 1\}. \end{cases} \tag{10}$$

The edge between two vertices in the initial GVD topological graph is represented by a sequence of path points, where the points of the path are elements in $\mathcal{X}_{\mathrm{GVD}}$. Let $N_{\mathrm{E}}(E)$ represent the number of points of path E; the edge from vertex j_1 to vertex j_2 can be described as a finite point sequence, denoted as

$$E_{\mathrm{GVD}}^{(j_1,j_2)} = \left\{ \mathbf{x}_k \middle| k = 1,2,\ldots,N_E\left(E_{\mathrm{GVD}}^{(j_1,j_2)}\right), \mathbf{x}_k \in \mathcal{X}_{\mathrm{GVD}} \right\}, \tag{11}$$

subject to $E_{\mathrm{GVD}}^{(j_1,j_2)}(1) = V_{j_1}, E_{\mathrm{GVD}}^{(j_1,j_2)}\left(N_{\mathrm{E}}\left(E_{\mathrm{GVD}}^{(j_1,j_2)}\right)\right) = V_{j_2}$.

Based on the initial GVD topological graph, the LRCG can be generated by the channel construction algorithm proposed in this paper, denoted as

$$\begin{cases} \mathcal{G}_{\mathrm{LRCG}} = \langle \mathcal{V}_{\mathrm{LRCG}}, \mathcal{E}_{\mathrm{LRCG}}, \mathrm{RSM}_{\mathrm{LRCG}} \rangle, \\ \mathcal{V}_{\mathrm{LRCG}} = \mathcal{V}_{\mathrm{GVD}}, \\ \mathcal{E}_{\mathrm{LRCG}} = \left\{ E_{\mathrm{LRCG}}^{(j_1,j_2)} \middle| j_1, j_2 \in \{1, 2, \ldots, N_{\mathrm{LRCG}}\}, j_1 \neq j_2 \right\}, \\ \mathrm{RSM}_{\mathrm{LRCG}} \in \{0,1\}^{N_{\mathrm{LRCG}} \times N_{\mathrm{LRCG}}}. \end{cases} \tag{12}$$

Herein, the vertex set $\mathcal{V}_{\mathrm{LRCG}}$ is the same as the initial GVD topology graph vertex set $\mathcal{V}_{\mathrm{GVD}}$, the number of vertices $N_{\mathrm{LRCG}} = N_{\mathrm{GVD}}$, and $\mathrm{RSM}_{\mathrm{LRCG}}$ is the edge refinement state matrix of LRCG, with 0 and 1 representing the unrefined and refined states respectively. The edge set $\mathcal{E}_{\mathrm{LRCG}}$ consists of channel edge elements constructed based on the edges of the initial GVD topological graph. We use a sequence of several intersecting circles as the model of the LRCG channel edge to simplify the description of the accessible neighborhood of a single edge, as shown in Figure 1a. The channel edge from vertex j_1 to vertex j_2 can be denoted as

$$E_{\mathrm{LRCG}}^{(j_1,j_2)} = \left\{ C(\mathbf{x}_k, r_{\mathrm{M}}(\mathbf{x}_k)) \middle| k = 1, 2, \ldots, N_{\mathrm{E}}\left(E_{\mathrm{LRCG}}^{(j_1,j_2)}\right) \right\}, \tag{13}$$

where $r_{\mathrm{M}}(\mathbf{x})$ is the Euclidean distance from point $\mathbf{x}$ to the nearest obstacle, and $C(\mathbf{x}, r)$ is a circle with $\mathbf{x}$ as center and r as radius. We define the circle with $\mathbf{x}$ as the center and $r_{\mathrm{M}}(\mathbf{x})$ as the radius as the maximum free-neighborhood circle, denoted as $C_{\mathrm{M}}(\mathbf{x}) = C(\mathbf{x}, r_{\mathrm{M}}(\mathbf{x}))$.

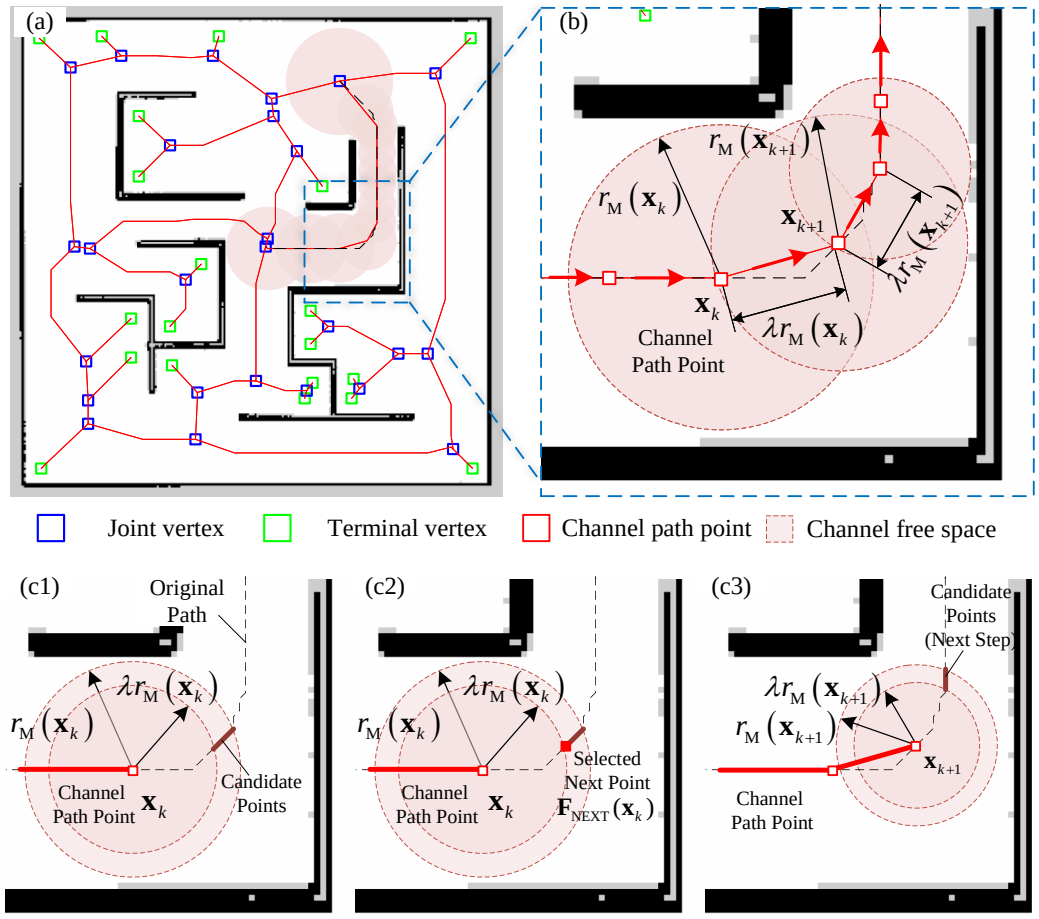

Figure 1. Channel model of the LRCG. (**a**) LRCG model of an occupancy grid map. (**b**) Channel construction and free-space model of the LRCG. (**c1–c3**) Iterative process of channel construction.

To simplify the original edge paths in the initial GVD topological graph and accelerate the path searching and optimization, we design a channel construction algorithm to construct the edges of the LRCG in (13), as shown in Figure 1b,(c1–c3). Define the simplification coefficient $\lambda \in (0, 1]$. The starting point of the original path V_{j_1} is set to the first point of the simplified path $\mathbf{x}_1$. The next point selected from the current point $\mathbf{x}_k$ on the edge of the initial GVD topological graph must satisfy the requirement that the distance between the two points is greater than $\lambda r_{\mathrm{M}}(\mathbf{x}_k)$, the next point lies within the maximum free-neighborhood circle of the current point, the maximum free-neighborhood circles of the two points intersect, and the common chord intersects the line segment connecting two points. Let the original path $E_{\mathrm{GVD}}^{(j_1,j_2)}$ be denoted as E; then, the above condition can be summarized as

$$\begin{cases} \|\mathbf{x}_k - E(n)\|_2 \geq \max\left\{\lambda r_{\mathrm{M}}(\mathbf{x}_k), \sqrt{\left|r_{\mathrm{M}}^2(\mathbf{x}_k) - r_{\mathrm{M}}^2(E(n))\right|}\right\}, \\ \|\mathbf{x}_k - E(n)\|_2 \leq r_{\mathrm{M}}(\mathbf{x}_k). \end{cases} \tag{14}$$

Herein, the simplification coefficient λ is used to control the number of points retained in the simplified path. A larger value of λ leads to stronger path simplification, thereby reducing the number of path points and improving the computational efficiency of path planning. When λ is smaller, more path points are retained, and the free space in the path neighborhood can be represented more completely. Therefore, λ mainly determines the trade-off between search-space simplification and the preservation of environmental geometric information. The next point is selected as the nearest point that satisfies the above conditions, as shown in Figure 1c2. Let $I_{\mathrm{E}}(E, \mathbf{x})$ be the point index of $\mathbf{x}$ in path E, then the next point selected is expressed as

$$\mathbf{F}_{\mathrm{NEXT}}(\mathbf{x}_k) = E\left(\min\left\{n \middle| n \text{ is subject to } (14), n > I_{\mathrm{E}}(E, \mathbf{x}_k)\right\}\right). \tag{15}$$

The above search process is executed repeatedly until the end point V_{j_2} is within the maximum free-neighborhood circle of the current point $\mathbf{x}_k$. The iterative process can be described as

$$\begin{cases} \mathbf{x}_1 = V_{j_1} \\ \mathbf{x}_{k+1} = \begin{cases} \mathbf{F}_{\mathrm{NEXT}}(\mathbf{x}_k), & \text{if} \left\|\mathbf{x}_k - V_{j_2}\right\|_2 > r_{\mathrm{M}}(\mathbf{x}_k), \\ V_{j_2}, & \text{if} \left\|\mathbf{x}_k - V_{j_2}\right\|_2 \leq r_{\mathrm{M}}(\mathbf{x}_k) \text{ and exit.} \end{cases} \end{cases} \tag{16}$$

The LRCG model generated from an occupancy grid map and its edge channel model are shown in Figure 1. For the same edge, the constructed channel edges with two vertices as starting points are different. We choose the one with the larger neighborhood free-space area as the channel edge between the two vertices.

4.2. LRCG Channel Sub-Path Homotopy

To make the LRCG represent the free space more finely, we design a channel sub-path homotopy model for edges of the LRCG, which expands the original centerline path of the channel into a class of augmented sub-paths.

According to condition (14) of the channel construction algorithm, the common chord and the line segment connecting the centers of two adjacent circles intersect. For two intersecting circles $C_1 = C(\mathbf{x}_1, r_1)$ and $C_2 = C(\mathbf{x}_2, r_2)$, the left and right endpoints of the common chord are recorded as $\boldsymbol{\varphi}_{\mathrm{L}}(C_1, C_2)$ and $\boldsymbol{\varphi}_{\mathrm{R}}(C_1, C_2)$, where we set the left and right endpoints to the left and right of the vector from $\mathbf{x}_1$ to $\mathbf{x}_2$, respectively, that is, $\overrightarrow{\mathbf{x}_1\mathbf{x}_2} \times \overrightarrow{\mathbf{x}_1\boldsymbol{\varphi}_{\mathrm{L}}(C_1, C_2)} > 0, \overrightarrow{\mathbf{x}_1\mathbf{x}_2} \times \overrightarrow{\mathbf{x}_1\boldsymbol{\varphi}_{\mathrm{R}}(C_1, C_2)} < 0$. For an original channel edge $E_{\mathrm{LRCG}}^{(J_1,J_2)}$ in the LRCG from the vertex J_1 to the vertex J_2, the number of its circles is recorded as $N = N_{\mathrm{E}}\left(E_{\mathrm{LRCG}}^{(J_1,J_2)}\right)$,

and the sequence of circles is recorded as $C_k = C(\mathbf{x}_k, r_M(\mathbf{x}_k))$, $k = 1, 2, \ldots, N$. From the circle sequence, we can get the sequence of common chords of two adjacent circles, and the sequence of their left and right endpoints is

$$\begin{cases} \mathbf{z}_{Lk} = \boldsymbol{\varphi}_L(C_k, C_{k+1}), k = 1, 2, \ldots, N-1, \\ \mathbf{z}_{Rk} = \boldsymbol{\varphi}_R(C_k, C_{k+1}), k = 1, 2, \ldots, N-1. \end{cases} \tag{17}$$

For channel edges, a channel path homotopy $H_{\mathrm{LRCG}}^{(J_1,J_2)} : [0,1] \times [0,1] \to \mathcal{X}$ is established to generate the navigable non-crossing sub-paths in the channel. The homotopy parameter $u \in [0,1]$ controls the continuous deformation from the left endpoint path $\gamma_0(s)$ to the right endpoint path $\gamma_1(s)$, and the path parameter s traces the intermediate curves. The homotopic sub-path is constructed by connecting fixed-ratio intermediate points along each chord and applying linear interpolation, denoted as

$$H_{\mathrm{LRCG}}^{(J_1,J_2)}(u, s) = \gamma_u(s) = \mathbf{M}_{k(s)}(u) + t(s)\left(\mathbf{M}_{k(s)+1}(u) - \mathbf{M}_{k(s)}(u)\right) \tag{18}$$

where $k(s)$ is the line segment index of the point on the path, $t(s)$ is the local parameter of the point on the current line segment, and $\mathbf{M}_k(u)$ is the endpoint of the path on the kth chord, denoted as

$$\mathbf{M}_k(u) = \mathbf{z}_{Lk} + u(\mathbf{z}_{Rk} - \mathbf{z}_{Lk}) \tag{19}$$

$$k(s) = \lfloor (N-2)s \rfloor + 1 \tag{20}$$

$$t(s) = (N-2)s + 1 - k(s) \tag{21}$$

Remark 1. *The sub-path in the homotopy defined by (18) is in the free space $\mathcal{X}_{\mathrm{FREE}}$, and the sub-paths do not intersect each other. Two adjacent points on any sub-path $\mathbf{M}_k(u)$, $\mathbf{M}_{k+1}(u)$ are located on two adjacent common chords respectively, and the quadrilateral formed by two adjacent common chords, denoted as $(\mathbf{z}_{Lk}\mathbf{z}_{Rk}\mathbf{z}_{R,k+1}\mathbf{z}_{L,k+1})$, is inscribed in the maximum free-neighborhood circle of a point in the LRCG channel C_{k+1}, thus belonging to the free space $\mathcal{X}_{\mathrm{FREE}}$. The line segment between the two adjacent points belongs to the quadrilateral formed by the common chords, that is, it belongs to the free space $\mathcal{X}_{\mathrm{FREE}}$. Therefore, the sub-path cluster belongs to the free space.*

4.3. LRCG Channel Refinement

The predetermined number of sub-paths is denoted as L. The channel refinement model of the LRCG is divided into two cases for discussion: the sub-path homotopy and directly adjacent vertices.

4.3.1. Channel Refinement for Sub-Path Homotopy

Considering the case where the number of original path points $N > 2$, the sub-paths of the refined channel edge are the elements of the sub-path homotopy $H_{\mathrm{LRCG}}^{(J_1,J_2)}$, denoted as $\left\{p_{\mathrm{SUB}}^{(h)} | h = 1, 2, \ldots, L\right\}$. As shown in Figure 2a, the sub-paths are constructed as point sequences corresponding to the sub-path homotopy, where the hth sub-path is

$$p_{\mathrm{SUB}}^{(h)}(k) = H_{\mathrm{LRCG}}^{(J_1,J_2)}\left(\frac{h-1}{L-1}, \frac{k-1}{N-2}\right), k = 1, 2, \ldots, N-1. \tag{22}$$

The vertex set of the refined LRCG is the union of the original vertex set and the extended vertex set, and the endpoints of the generated sub-paths are selected as the extended vertices of the LRCG. The vertex set of the refined LRCG is recorded as

$$\begin{cases} \mathcal{V}_{\mathrm{LRCG}}^{(J_1,J_2,L)} = \mathcal{V}_{\mathrm{LRCG}} \cup \mathcal{V}_{\mathrm{REFINE}}^{(J_1,J_2,L)} \\ \mathcal{V}_{\mathrm{REFINE}}^{(J_1,J_2,L)} = \left\{ p_{\mathrm{SUB}}^{(h)}(1), p_{\mathrm{SUB}}^{(h)}(N-1) \right\}_{h=1,2,\ldots,L} \end{cases}. \tag{23}$$

Herein, we stipulate that the two endpoints of the hth sub-path are numbered $I_1(h) = N_{\mathrm{LRCG}} + 2h - 1$, $I_2(h) = N_{\mathrm{LRCG}} + 2h$ in the vertex set respectively. The starting point and ending point vertices J_1 and J_2 of the original edge are defined as the parent vertices of the sub-path starting point and ending point vertices, respectively.

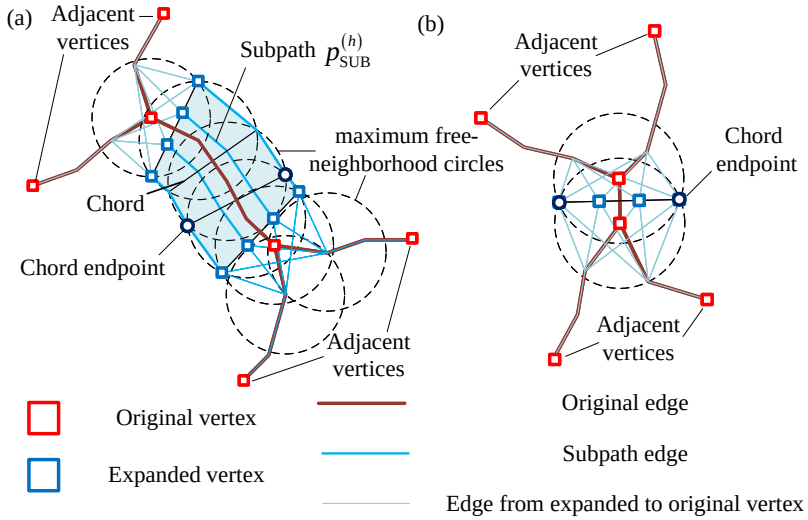

Figure 2. Channel refinement model of the LRCG. (**a**) Case where the number of original path points $N > 2$. (**b**) Case where the number of original path points $N = 2$.

The edge set of the refined LRCG is the union of the original edge set, the edge set between the extended vertices, and the edge set connecting the extended vertices with their adjacent original vertices. Between the extended vertices, the sub-path endpoint vertex is connected to the vertex corresponding to the other endpoint through the sub-path, that is, there is an edge path between the vertices corresponding to the two endpoints of the sub-path. The edge set between the extended vertices is

$$\begin{cases} \mathcal{E}_{\mathrm{SUB}}^{(J_1,J_2,L)} = \left\{ E_{\mathrm{LRCG}}^{(I_1(h),I_2(h))} \right\}_{h=1,2,\ldots,L}, \\ E_{\mathrm{LRCG}}^{(I_1(h),I_2(h))} = \left\{ C\left(p_{\mathrm{SUB}}^{(h)}(k), r_{\mathrm{M}}\left(p_{\mathrm{SUB}}^{(h)}(k) \right) \right) \right\}_{k=1,2,\ldots,N-1}. \end{cases} \tag{24}$$

The expanded vertex is connected to the adjacent vertex of its corresponding parent vertex. Based on the path from the parent vertex to the adjacent vertex, the starting point of the path is replaced from the parent vertex to the extended vertex, and the modified path is adopted as the edge from the extended vertex to the adjacent vertex, which is expressed as

$$\begin{cases} E_{\mathrm{LRCG}}^{(I_1(h),J_{\mathrm{N}})} = \{ C(p(k), r_{\mathrm{M}}(p(k))) \}_{k=1,2,\ldots,N_1}, \\ p(k) = \begin{cases} p_{\mathrm{SUB}}^{(h)}(1), & k = 1, \\ O\left(E_{\mathrm{LRCG}}^{(J_1,J_{\mathrm{N}})}(k) \right), & k = 2,\ldots,N_1. \end{cases} \\ J_{\mathrm{N}} \in S_{\mathrm{NEIGH}}\left(V_{J_1}, \mathcal{V}_{\mathrm{LRCG}} \right) \backslash \{ V_{J_2} \}, N_1 = N_{\mathrm{E}}\left(E_{\mathrm{LRCG}}^{(J_1,J_{\mathrm{N}})} \right), \end{cases} \tag{25}$$

$$\begin{cases} E_{\mathrm{LRCG}}^{(I_2(h),J_{\mathrm{N}})} = \{C(p(k), r_{\mathrm{M}}(p(k)))\}_{k=1,2,\ldots,N_2}, \\ p(k) = \begin{cases} p_{\mathrm{SUB}}^{(h)}(N-1), & k=1, \\ O\left(E_{\mathrm{LRCG}}^{(J_2,J_{\mathrm{N}})}(k)\right), & k=2,\ldots,N_2. \end{cases} \\ J_{\mathrm{N}} \in S_{\mathrm{NEIGH}}(V_{J_2}, \mathcal{V}_{\mathrm{LRCG}}) \backslash \{V_{J_1}\}, N_2 = N_{\mathrm{E}}\left(E_{\mathrm{LRCG}}^{(J_2,J_{\mathrm{N}})}\right), \end{cases} \tag{26}$$

where $S_{\mathrm{NEIGH}}(V, \mathcal{V}_{\mathrm{LRCG}})$ is the set of adjacent vertices of the vertex V in the original LRCG, and $O(C)$ represents the center position of circle C.

Remark 2. *The constructed edge paths from the extended vertex to the adjacent vertices of its parent vertex are within the free space $\mathcal{X}_{\mathrm{FREE}}$. According to condition (14), the path starting point and the second point are both within the maximum free neighborhood circle of the parent vertex V_{J_1} or V_{J_2}, the line segment between the two points is also within the free space. And the subsequent part of the constructed path is maintained, so the overall path is within the free space and is a feasible path.*

Then, the edge set connecting the extended vertices with their adjacent original vertices can be expressed as

$$\begin{cases} \mathcal{E}_{\mathrm{NEIGH}}^{(J_1,J_2,L)} = \mathcal{E}_{\mathrm{NEIGH1}}^{(J_1,J_2,L)} \cup \mathcal{E}_{\mathrm{NEIGH2}}^{(J_1,J_2,L)}, \\ \mathcal{E}_{\mathrm{NEIGH1}}^{(J_1,J_2,L)} = \left\{E_{\mathrm{LRCG}}^{(I_1(h),J_{\mathrm{N}})}\right\}_{J_{\mathrm{N}} \in S_{\mathrm{NEIGH}}(V_{J_1}, \mathcal{V}_{\mathrm{LRCG}}), h=1,2,\ldots,L}, \\ \mathcal{E}_{\mathrm{NEIGH2}}^{(J_1,J_2,L)} = \left\{E_{\mathrm{LRCG}}^{(I_2(h),J_{\mathrm{N}})}\right\}_{J_{\mathrm{N}} \in S_{\mathrm{NEIGH}}(V_{J_2}, \mathcal{V}_{\mathrm{LRCG}}), h=1,2,\ldots,L}. \end{cases} \tag{27}$$

In summary, the edge set of the refined LRCG is

$$\mathcal{E}_{\mathrm{LRCG}}^{(J_1,J_2,L)} = \mathcal{E}_{\mathrm{LRCG}} \cup \mathcal{E}_{\mathrm{NEIGH}}^{(J_1,J_2,L)} \cup \mathcal{E}_{\mathrm{SUB}}^{(J_1,J_2,L)} \tag{28}$$

4.3.2. Channel Refinement for Directly Adjacent Vertices

We further consider the special case where the number of original path points $N = 2$, that is, the two endpoints of the original path are directly connected by a straight line and the two endpoints are within the maximum free-neighborhood circle of the other point. In that case, the sub-path homotopy is reduced to a point sequence, which is the equal-division points of the common chord, as shown in Figure 2b, denoted as $\left\{v_{\mathrm{SUB}}^{(h)} | h = 1, 2, \ldots, L\right\}$, where the hth point is

$$v_{\mathrm{SUB}}^{(h)} = \frac{L-h}{L-1}\mathbf{z}_{\mathrm{L1}} + \frac{h-1}{L-1}\mathbf{z}_{\mathrm{R1}}. \tag{29}$$

Taking the above point sequence as the extended vertices, the vertex set of the refined LRCG is recorded as

$$\begin{cases} \mathcal{V}_{\mathrm{LRCG}}^{(J_1,J_2,L)} = \mathcal{V}_{\mathrm{LRCG}} \cup \mathcal{V}_{\mathrm{REFINE}}^{(J_1,J_2,L)} \\ \mathcal{V}_{\mathrm{REFINE}}^{(J_1,J_2,L)} = \left\{v_{\mathrm{SUB}}^{(h)}\right\}_{h=1,2,\ldots,L} \end{cases}. \tag{30}$$

Herein, we stipulate that the hth extended endpoint is numbered $I(h) = N_{\mathrm{LRCG}} + h$ in the vertex set.

Different from the case where sub-paths exist, the edge set of the refined LRCG consists of the original edge set and the edge set between the extended vertices and the adjacent original vertices and does not include the edges between the extended vertices. Since the extended vertices can be directly connected to both endpoints of the refined edge, the extended vertices are connected to the adjacent vertices of the two origin vertices. The

construction method of the edges is the same as the case where sub-paths exist, which is expressed as

$$\begin{cases} E_{\mathrm{LRCG}}^{(I(h),J_{\mathrm{N}})} = \{C(p(k), r_{\mathrm{M}}(p(k)))\}_{k=1,2,\ldots,N_1}, \\ p(k) = \begin{cases} v_{\mathrm{SUB}}^{(h)}, & k = 1, \\ O\left(E_{\mathrm{LRCG}}^{(J_1,J_{\mathrm{N}})}(k)\right), & k = 2,\ldots,N_1. \end{cases} \\ J_{\mathrm{N}} \in S_{\mathrm{NEIGH}}(V_{J_1}, V_{\mathrm{LRCG}}) \backslash \{V_{J_2}\}, N_1 = N_{\mathrm{E}}\left(E_{\mathrm{LRCG}}^{(J_1,J_{\mathrm{N}})}\right), \end{cases} \tag{31}$$

$$\begin{cases} E_{\mathrm{LRCG}}^{(I(h),J_{\mathrm{N}})} = \{C(p(k), r_{\mathrm{M}}(p(k)))\}_{k=1,2,\ldots,N_2}, \\ p(k) = \begin{cases} v_{\mathrm{SUB}}^{(h)}, & k = 1, \\ O\left(E_{\mathrm{LRCG}}^{(J_2,J_{\mathrm{N}})}(k)\right), & k = 2,\ldots,N_2. \end{cases} \\ J_{\mathrm{N}} \in S_{\mathrm{NEIGH}}(V_{J_2}, V_{\mathrm{LRCG}}) \backslash \{V_{J_1}\}, N_2 = N_{\mathrm{E}}\left(E_{\mathrm{LRCG}}^{(J_1,J_{\mathrm{N}})}\right). \end{cases} \tag{32}$$

Then, the edge set of the refined LRCG can be expressed as

$$\begin{cases} \mathcal{E}_{\mathrm{LRCG}}^{(J_1,J_2,L)} = \mathcal{E}_{\mathrm{LRCG}} \cup \mathcal{E}_{\mathrm{NEIGH1}}^{(J_1,J_2,L)} \cup \mathcal{E}_{\mathrm{NEIGH2}}^{(J_1,J_2,L)}, \\ \mathcal{E}_{\mathrm{NEIGH1}}^{(J_1,J_2,L)} = \left\{E_{\mathrm{LRCG}}^{(I(h),J_{\mathrm{N}})}\right\}_{J_{\mathrm{N}} \in S_{\mathrm{NEIGH}}(V_{J_1}, \mathcal{V}_{\mathrm{LRCG}}), h=1,2,\ldots,L}, \\ \mathcal{E}_{\mathrm{NEIGH2}}^{(J_1,J_2,L)} = \left\{E_{\mathrm{LRCG}}^{(I(h),J_{\mathrm{N}})}\right\}_{J_{\mathrm{N}} \in S_{\mathrm{NEIGH}}(V_{J_2}, \mathcal{V}_{\mathrm{LRCG}}), h=1,2,\ldots,L}. \end{cases} \tag{33}$$

5. LRCG-Based Multi-UAV Path Planning

To solve the multi-UAV collision-free path planning problem, this paper designed a planning framework based on the design of LRCG and its channel refinement model. The LRCG-based planning framework consists of an LRCG hybrid conflict-based search algorithm, a dynamic-programming-based channel path constraint optimization algorithm, and an LRCG shortcut-path embedding algorithm, which are used for initial solution generation, post-processing optimization, and search-space preprocessing, respectively.

5.1. LRCG Hybrid Conflict-Based Search

An LRCG hybrid conflict-based search algorithm (LRCG-HCBS) is established to plan an initial set of multi-UAV paths that meet the requirements of collision and obstacle avoidance. The core principle of the LRCG-HCBS algorithm is similar to that of the CBS algorithm, which is to maintain the constraint set of each UAV through a two-layer search framework and plan of the constrained paths of each UAV to achieve overall collision avoidance. In the two-level search framework, the high-level search adopts a binary tree search method named constraint tree (CT), and the low-level search adopts a graph-based constrained search method. The LRCG-HCBS algorithm uses the initial and refined LRCG as the search space, unlike the static graph in the CBS algorithm. The planning algorithm includes the conflict detection, constraint generation based on conflicts, and constrained path planning for the UAVs.

As a basis, the definitions of conflicts, constraints and constraint trees in the LRCG-HCBS algorithm are similar to those of the CBS algorithm:

(1) Conflict: The conflict between UAVs is represented as the conflicting parts of the trajectories of the two UAVs and the sequence of vertices of the LRCG contained in the conflicting trajectories, where the trajectory is represented in the form of a discrete time and point sequence. The conflict can be expressed as

$$\begin{cases} \chi = \langle I_1, I_2, \mu_{\mathrm{C}}, \mu_{\mathrm{VC}} \rangle, \\ \mu_{\mathrm{C}} = \left\{(t_k, \boldsymbol{\phi}_{1,k}, \boldsymbol{\phi}_{2,k})\right\}_{k=0}^{N_\psi(\mu_{\mathrm{C}})}, \\ \mu_{\mathrm{VC}} = \left\{(t_{1,k}, t_{2,k}, I_{\mathrm{V}k})\right\}_{k=0}^{N_\psi(\mu_{\mathrm{VC}})}, \end{cases} \tag{34}$$

where $I_1, I_2 \in \{1, 2, \ldots, M\}$ are the indices of the two UAVs, μ_C is the conflicting trajectory of the two UAVs, including a time series and the position sequence of the two UAVs at the corresponding time, and μ_{VC} is the conflicting vertex trajectory, including the conflicting vertex sequence and the arrival time sequence of the two UAVs.

(2) Constraint: The constraints of path planning of a UAV can be described as the trajectory and vertex trajectory that need to be avoided, which are expressed by a sequence of discrete time and points, respectively, denoted as

$$\begin{cases} \varsigma = \langle I, \psi_C, \psi_{VC} \rangle, \\ \psi_C = \left\{ (t_k, \boldsymbol{\phi}_k) \right\}_{k=0}^{N_\psi(\psi_C)}, \\ \psi_{VC} = \{ (t_k, I_{Vk}) \}_{k=0}^{N_\psi(\psi_{VC})}, \end{cases} \tag{35}$$

where ψ_C and ψ_{VC} represent the avoidance trajectory and vertex trajectory respectively, and I is the index of the UAV.

(3) Constraint Tree: The constraint tree is the binary search tree structure in the high-level search of LRCG-HCBS. The node of the constraint tree contains the list of constraints that the UAV needs to satisfy in the low-level search, the total cost of the planned paths, and records the LRCG corresponding to the node, denoted as

$$\mathbf{n} = \left\langle \mathcal{G}_{LRCG}, \{\varsigma_k\}_{k=1}^{N_\varsigma}, c \right\rangle, \tag{36}$$

where q is the node index, and N_ς is the number of constraints. We denote the constraint set of the node $\mathbf{n}$ as $\pi_C(\mathbf{n})$.

The high-level search algorithm of LRCG-HCBS adopts a best-first search manner based on the constraint tree, as shown in Algorithm 1. A node without path planning constraints is firstly defined as the root node of the constraint tree, and its corresponding LRCG is the initial unrefined LRCG. The overall search procedure can be divided into three stages: exploratory search within the initial LRCG, conflict-oriented refinement of the LRCG, and conflict-based search based on the refined LRCG, as shown in Figure 3. Initially, the unconstrained optimal path of each UAV is planned in the exploratory search, and the conflict set between UAVs is obtained. Different from the CBS algorithm on a static graph, in the LRCG-HCBS algorithm, an additional conflict-oriented LRCG refinement algorithm and a replanning process are added to generate a refined LRCG for subsequent search. In the iteration process, the cost-optimal node on the constraint tree is selected, and the low-level search method is invoked to plan the path of each UAV under the constraints of the node and judge the existence of conflicts. New constraints are constructed based on the conflicts as constraints added to the new nodes derived from the existing node. The above process of optimal node selection, processing, and expansion of new nodes is iteratively executed until the low-level planning generates a conflict-free solution.

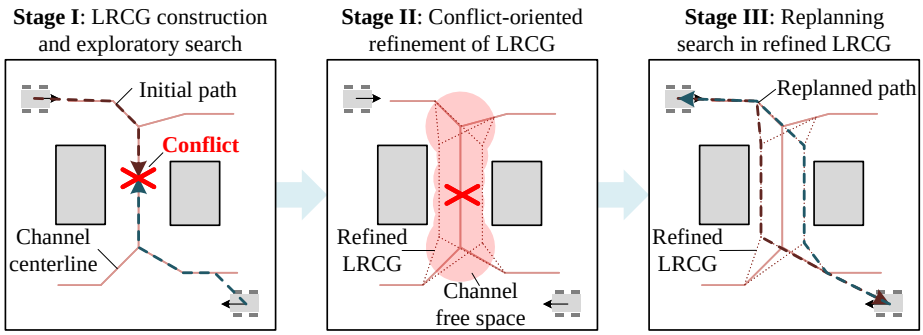

Figure 3. Three-stage search procedure of LRCG-HCBS.

Algorithm 1 High-level search of LRCG-HCBS

Require: Initial LRCG $\mathcal{G}_{\text{LRCG}}$, starting point set $\{\mathbf{s}_i\}_{i=1}^{M}$, goal point set $\{\mathbf{g}_i\}_{i=1}^{M}$, minimum distance between UAVs $d_{\min}$
Ensure: Planned paths $\{\psi_i\}_{i=1}^{M}$
1: Define the root node and add it to the constraint tree:
2: $\mathbf{n}^{(0)} \leftarrow \left\langle \mathcal{G}_{\text{LRCG}}^{(0)}, \emptyset \right\rangle, \mathcal{G}_{\text{LRCG}}^{(0)} \leftarrow \mathcal{G}_{\text{LRCG}}, \text{CT} \leftarrow \left\{ \mathbf{n}^{(0)} \right\},$
3: **while** CT $\neq \emptyset$ **do**
4: Choose the min-cost node q^* from CT
5: **for** $i = 1, 2, \ldots, M$ **do**
6: Plan the constrained path for a single UAV: $\psi_i \leftarrow \text{LowLevel}\left(\mathcal{G}_{\text{LRCG}}^{(q^*)}, \mathbf{n}^{(q^*)}, \mathbf{s}_i, \mathbf{g}_i\right),$
7: **end for**
8: Construct the conflict set between UAV paths: $\chi \leftarrow \text{ConflictDetect}\left(\{\psi_i\}_{i=1}^{M}\right)$
9: **if** $\chi = \emptyset$ **then**
10: **return** $\{\psi_i\}_{i=1}^{M}$.
11: **end if**
12: Refine the LRCG based on the conflicts: $\tilde{\mathcal{G}}_{\text{LRCG}}^{(q^*)} \leftarrow \text{ConflictOrientedRefine}\left(\mathcal{G}_{\text{LRCG}}^{(q^*)}, \chi\right).$
13: Replan the paths in updated LRCG (lines 5-11)
14: Select a conflict $\chi \in \chi$.
15: Construct the constraints ς_1, ς_2 for UAVs I_1, I_2.
16: Define two child node $\tilde{\mathbf{n}}_1, \tilde{\mathbf{n}}_2 \leftarrow \mathbf{n}^{(q^*)}$.
17: Add new constraints to the child node constraint sets:
$\pi_{\text{C}}(\tilde{\mathbf{n}}_1) = \pi_{\text{C}}\left(\mathbf{n}^{(q^*)}\right) \cup \varsigma_1, \pi_{\text{C}}(\tilde{\mathbf{n}}_2) = \pi_{\text{C}}\left(\mathbf{n}^{(q^*)}\right) \cup \varsigma_2.$
18: Add child nodes to the constraint tree: CT $\leftarrow$ CT $\cup \{\tilde{\mathbf{n}}_1, \tilde{\mathbf{n}}_2\}$.
19: **end while**

5.1.1. Low-Level Search

The low-level search algorithm uses the Space-Time A* algorithm [38] as the skeleton algorithm and the LRCG as the search space to generate UAV trajectories that meet the set dynamic obstacle avoidance constraints. In the low-level search of LRCG-HCBS, Space-Time A* is conducted on the LRCG by representing each state as an LRCG vertex with an associated arrival time. The node expansion includes waiting at the current vertex for a fixed duration and transferring to adjacent vertices. The traversal time between two adjacent vertices is calculated as the length of the corresponding path segment divided by the UAV motion velocity. During the search, feasibility is checked according to the safe separation constraint between path segments over the corresponding time interval. Therefore, unlike standard MAPF implementations that rely mainly on vertex and edge conflicts, our Space-Time A* uses distance-based conflict checking to determine whether a candidate transition is valid.

5.1.2. Conflict Detection

In the conflict detection algorithm, the planned trajectories of the two UAVs to be detected are expressed in the form of time and point sequences. The union of the time sequences of the two trajectories is used as the detection timestamp set. The positions of the two UAVs at each timestamp in the timestamp set are checked to see if the minimum distance constraint is satisfied. The earliest and latest timestamps are selected from the set of conflicting timestamps to form the conflicting time period and trajectory. In addition, in the LRCG-HCBS algorithm, the conflicting vertex trajectories, i.e., the set of vertices that the two UAVs simultaneously pass through on the conflicting trajectory, are calculated for the subsequent LRCG refinement.

5.1.3. Conflict Resolution

In the conflict resolution process, for the conflicting trajectories of the two UAVs, constraints are constructed for the two UAVs to avoid the trajectory of each other. The newly constructed constraints are added as constraints in two new child nodes derived from a node.

5.1.4. Conflict-Oriented LRCG Refinement

The main goal of the conflict-oriented LRCG refinement algorithm is to avoid conflicts by replanning in the refined graph and reduce the number of nodes in the constraint tree. Among the edges corresponding to vertex trajectories of the detected conflicts, the edges whose preceding and succeeding vertices are both initial vertices of the LRCG and have not been refined are selected as the edges to be refined. The number of conflicts on each edge of the initial LRCG is counted, which is used as the predetermined number of sub-paths for edge-channel refinement. Edges with no detected conflicts, i.e., with a conflict count of 0, are not refined. For each edge with detected conflicts, the predetermined number of sub-paths for edge-channel refinement is set equal to the number of conflicts associated with that edge and not less than 2. Finally, the edge channels of LRCG are refined according to the predetermined number of sub-paths. The overall process is shown in Algorithm 2.

Algorithm 2 Conflict-oriented refinement of LRCG-HCBS

Require: Initial LRCG $\mathcal{G}_{\mathrm{LRCG}}$, conflict set χ
Ensure: Refined LRCG $\tilde{\mathcal{G}}_{\mathrm{LRCG}}$
$\tilde{\mathcal{G}}_{\mathrm{LRCG}} \leftarrow \mathcal{G}_{\mathrm{LRCG}}$
Initialize the conflict count matrix:
$\mathrm{CCM} \leftarrow \mathbf{O}^{N_{\mathrm{LRCG}} \times N_{\mathrm{LRCG}}}$
for $\chi \in \chi$ **do**
  $\mu_{\mathrm{VC}} = \pi_{\mathrm{VC}}(\chi)$
  for $k = 1, 2, \ldots, N_{\psi}(\mu_{\mathrm{VC}}) - 1$ **do**
    Update conflict count:
    $\mathrm{CCM}\left[I_{\mathrm{V}k}, I_{\mathrm{V}(k+1)}\right] \leftarrow \mathrm{CCM}\left[I_{\mathrm{V}k}, I_{\mathrm{V}(k+1)}\right] + 1,$
    $\mathrm{CCM}\left[I_{\mathrm{V}(k+1)}, I_{\mathrm{V}k}\right] \leftarrow \mathrm{CCM}\left[I_{\mathrm{V}(k+1)}, I_{\mathrm{V}k}\right] + 1,$
  end for
end for
for $j_1 = 1, 2, \ldots, N_{\mathrm{LRCG}} - 1$ **do**
  for $j_2 = j_1 + 1, j_1 + 2, \ldots, N_{\mathrm{LRCG}}$ **do**
    if $\mathrm{CCM}[j_1, j_2] \neq 0$ **then**
      Refine conflicting and unrefined edge:
      $\tilde{\mathcal{G}}_{\mathrm{LRCG}} \leftarrow \tilde{\mathcal{G}}_{\mathrm{LRCG}}^{(j_1, j_2, \mathrm{CCM}[j_1, j_2]+1)}$
    end if
  end for
end for

5.2. Constrained Dynamic Programming Search Optimization

Based on the initial path satisfying the collision avoidance constraints obtained by the LRCG-HCBS algorithm, this paper proposes a constrained dynamic programming search algorithm (CDPS) for post-processing optimization of the paths.

During the post-processing of multi-UAV paths, a path that has not been post-processed is randomly selected as the object of post-processing, in which the paths of other UAVs are used as dynamic obstacles to be avoided. The above random selection and optimization are iteratively performed until the paths of all UAVs are optimized.

5.2.1. Multi-Layer Optional Point Graph

In the CDPS algorithm of a single path, the path optimization problem in the free space is transformed into a graph-based constraint search problem. A multi-layer optional point graph is established based on the LRCG model as the search space, where the vertex set of the graph is a multi-layer path point set in the free space, and the straight line segments between the path nodes of adjacent layers are in the free space $\mathcal{X}_{\mathrm{FREE}}$.

From the initial planned path, after filtering the repeated points in the planned path, the maximum free-neighborhood circle sequence of the path points can be obtained, which

is recorded as $p_{\mathrm{INIT}} = \{C_k = C(\mathbf{x}_k, r_{\mathrm{M}}(\mathbf{x}_k))\}_{k=1}^{N_{\mathrm{P}}(p_{\mathrm{INIT}})}$, where $N_{\mathrm{P}}(p_{\mathrm{INIT}})$ is the number of path points. An optional point set for constrained dynamic programming search is established as $\mathrm{OPS} = \{\mathrm{OPL}_1, \mathrm{OPL}_2, \ldots, \mathrm{OPL}_{N_{\mathrm{OPS}}}\}$, where N_{OPS} is the layer number of the optional point set, and OPL_k corresponds to the optional point set of the kth layer, and the number of its elements is M_k. In the multi-layer optional point set, the starting point and the goal point are the single points in the first-layer and last-layer optional point sets, respectively. When two adjacent circles in the circle sequence intersect, the equally divided point set of the common chord serve as a layer of optional point set; when the two circles are separated, the centers of the two circles serve as the single points in the single-layer optional point sets. The construction process of the multi-layer optional point set is shown in Algorithm 3.

Algorithm 3 Optional point set generation of CDPS

Require: Initial path circle sequence $p_{\mathrm{INIT}} = \{C_k\}_{k=1}^{N_{\mathrm{P}}(p_{\mathrm{INIT}})}$
Ensure: Optional point set OPS

Add a layer to OPS (starting point):
$\mathrm{OPS} \leftarrow (\mathrm{OPL}_1), \mathrm{OPL}_1 = \{\mathbf{x}_1\}, N_{\mathrm{OPS}} \leftarrow 1,$
for $k = 2, 3, \ldots, N_{\mathrm{P}}(p_{\mathrm{INIT}})$
  if C_k, C_{k-1} intersect
    Add a layer (common chord equally divided points):
    $\mathrm{OPL}_{N_{\mathrm{OPS}}} \leftarrow \left\{q_h = \frac{L-h}{L-1}\mathbf{z}_{\mathrm{L}} + \frac{h-1}{L-1}\mathbf{z}_{\mathrm{R}}\right\}_{h=1}^{L}$,
    $\mathbf{z}_{\mathrm{L}} = \boldsymbol{\varphi}_{\mathrm{L}}(C_{k-1}, C_k), \mathbf{z}_{\mathrm{R}} = \boldsymbol{\varphi}_{\mathrm{R}}(C_{k-1}, C_k),$
    $\mathrm{OPS} \leftarrow (\mathrm{OPS}, \mathrm{OPL}_{N_{\mathrm{OPS}}}), N_{\mathrm{OPS}} \leftarrow N_{\mathrm{OPS}} + 1,$
  else
    if $\mathbf{x}_{k-1} \notin \mathrm{OPS}_{N_{\mathrm{OPS}}-1}$
      Add a layer (last circle center):
      $\mathrm{OPL}_{N_{\mathrm{OPS}}} \leftarrow \{\mathbf{x}_{k-1}\},$
      $\mathrm{OPS} \leftarrow (\mathrm{OPS}, \mathrm{OPL}_{N_{\mathrm{OPS}}}), N_{\mathrm{OPS}} \leftarrow N_{\mathrm{OPS}} + 1,$
    end if
    Add a layer (current circle center):
    $\mathrm{OPL}_{N_{\mathrm{OPS}}} \leftarrow \{\mathbf{x}_k\},$
    $\mathrm{OPS} \leftarrow (\mathrm{OPS}, \mathrm{OPL}_{N_{\mathrm{OPS}}}), N_{\mathrm{OPS}} \leftarrow N_{\mathrm{OPS}} + 1,$
  end if
  if $k = N_{\mathrm{P}}(p_{\mathrm{INIT}})$
    if $\mathbf{x}_k \notin \mathrm{OPS}_{N_{\mathrm{OPS}}-1}$
      Add a layer (goal point):
      $\mathrm{OPL}_{N_{\mathrm{OPS}}} \leftarrow \{\mathbf{x}_k\},$
      $\mathrm{OPS} \leftarrow (\mathrm{OPS}, \mathrm{OPL}_{N_{\mathrm{OPS}}}), N_{\mathrm{OPS}} \leftarrow N_{\mathrm{OPS}} + 1,$
    end if
    return OPS
  end if
end for

Remark 3. *In the multi-layer optional point set, the straight line segment between any points in two adjacent layers is in the free space $\mathcal{X}_{\mathrm{FREE}}$. When a circle intersects both the front and back adjacent circles, the straight line segment connecting any two points on the two common chords is inside the circle and therefore belongs to free space, as shown in Figure 4a. When a circle is separated from the adjacent circle, the straight line segment between the centers of the two circles belongs to the edge path of the LRCG, and it is known from Remark 1 and Remark 2 that the line segment belongs to the free space, as shown in Figure 4b,c. When a circle is separated from an adjacent circle and intersects with another adjacent circle, the area between the center of the circle and the common chord of the intersecting circle belongs to the free space, as shown in Figure 4b. In summary, the straight-line connectivity between adjacent layers in the multi-layer optional points can be proved.*

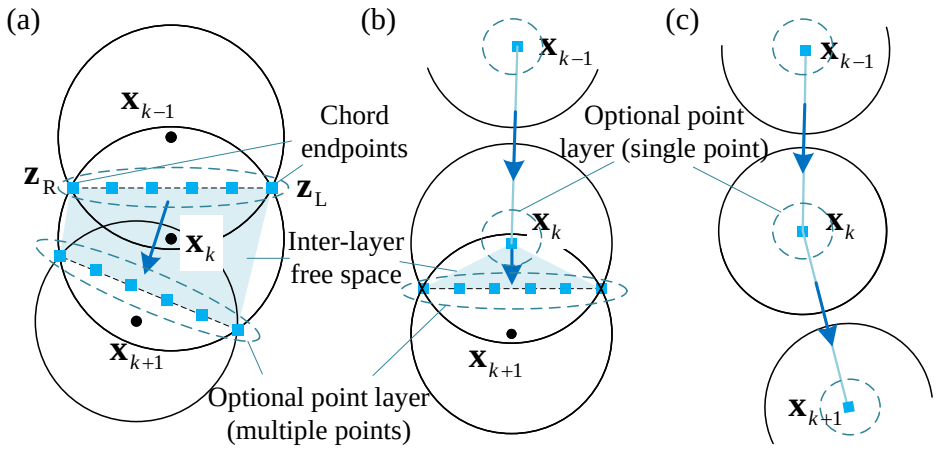

Figure 4. Multi-layer optional point set model of CDPS. (**a**) Case where a circle intersects both adjacent circles. (**b**) Case where a circle is separated from an adjacent circle and intersects with another adjacent circle. (**c**) Case where a circle is separated from both adjacent circles.

After establishing the vertex set of the multi-layer optional point graph, due to the linear connectivity of the optional points between adjacent layers, it is stipulated that there are only forward unidirectional edges between the vertices of adjacent layers.

5.2.2. Constrained Optimization

In the constrained optimization, the optimal subsequent path without collision avoidance constraints is initially planned for each optional point based on the dynamic programming method. The sequence of distance matrices between layers in the optional point set is recorded as $\left\{\mathrm{CM}_k \in \mathbb{R}^{M_k \times M_{k+1}}\right\}_{k=1}^{N_{\mathrm{OPS}}-1}$. Then, the shortest subsequent path length of the l_kth optional point in the kth layer is recorded as

$$\eta_{\mathrm{H}}(k, l_k) = \min_{\{l_p\}_{p=k+1}^{N_{\mathrm{OPS}}}} \sum_{p=k}^{N_{\mathrm{OPS}}-1} \mathrm{CM}_p\left[l_p, l_{p+1}\right], \tag{37}$$

and for the final layer, i.e., the goal point layer, $\eta_{\mathrm{H}}(N_{\mathrm{OPS}}, 1) = 0$. The index of the optional point in the next layer corresponding to the shortest subsequent path of the l_kth optional point in the kth layer is recorded as $l^*(k, l_k)$. In the dynamic programming approach, for $k = N_{\mathrm{OPS}} - 1, \ldots, 1$, the recursive process can be expressed as

$$\begin{cases} l^*(k, l_k) = \arg \min\limits_{l=1,2,\ldots,M_{k+1}} \left(\eta_{\mathrm{H}}(k+1, l) + \mathrm{CM}_k[l_k, l]\right), \\ \eta_{\mathrm{H}}(k, l_k) = \eta_{\mathrm{H}}(k+1, l^*(k, l_k)) + \mathrm{CM}_k[l_k, l^*(k, l_k)], \end{cases} \tag{38}$$

The constrained optimization algorithm based on the multi-layer optional point graph uses the Space-Time A* algorithm as the skeleton algorithm and the established graph as the search space. A key improvement in the search algorithm is the criterion for whether a node reaches the goal node, which is that both the traveled path to the node and the subsequent optimal path based on dynamic programming satisfy the constraints, rather than requiring the node to reach the goal node, as illustrated in Figure 5. The purpose of modifying the arrival criterion is to reduce the number of iterations of the constrained search, thereby improving the efficiency of the optimization.

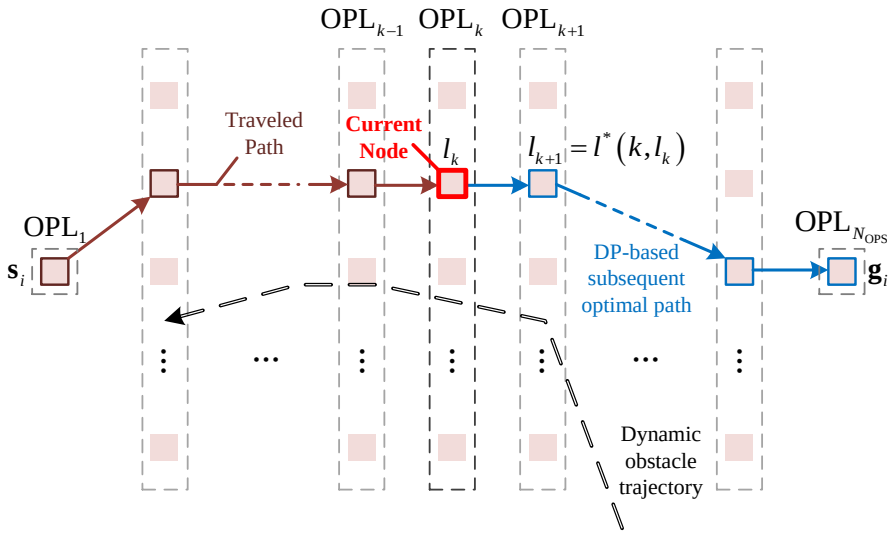

Figure 5. Constrained search and arrival criterion model of CDPS algorithm.

5.3. LRCG Shortcut-Path Embedding

Since the channel center path deviates from the boundary of the accessible region, the channel center path in the LRCG may deviate from the optimal path in the continuous space. If this deviation is not handled, it will lead to suboptimal solutions when performing path search directly on the initial LRCG. To alleviate this problem, this paper proposes a preprocessing strategy for the LRCG: first, an estimated optimal path is generated for a single UAV without collision avoidance constraints, named a shortcut path, and the key information of the shortcut path is embedded into the LRCG structure, thereby augmenting the LRCG and improving the overall quality of subsequent multi-UAV path planning.

5.3.1. Shortcut-Path Generation

The shortcut-path generation procedure is divided into LRCG-based channel path search and optimal path estimation for a single channel path. It is worth noting that due to the difference in channel width, the deviations between different channel center paths and the corresponding optimal path are also different, which means that the optimal path may not correspond to the optimal channel center path. Therefore, this paper adopts a bounded suboptimal search strategy [39], which searches for multiple different suboptimal channel center paths in addition to the optimal channel center path to avoid losing the optimal path. For the generated multiple bounded suboptimal channel center paths, we use the CDPS algorithm proposed in Section 5.2 to construct a multi-layer optional point graph and estimate the local optimal path corresponding to the channel. Unlike general CDPS, the shortcut-path generation procedure does not consider the influence of other UAVs, that is, there is no collision avoidance constraint in the search, and the shortest subsequent path from the starting point can be directly used as the estimated local optimal path. Among the estimated local optimal paths generated for multiple bounded suboptimal channel center paths, the shortest path is selected as the global shortcut path.

5.3.2. Shortcut-Path-Guided LRCG Refinement

To utilize the generated shortcut paths, a path embedding strategy is proposed to expand the initial LRCG. The expanded vertex set is the path points on the shortcut path that can be directly connected to the original LRCG vertex point. The straight connectivity depends on whether there is an obstacle between the two points, that is, whether the path point is within the maximum neighborhood circle of the original vertex point. The expanded edge set is composed of the shortcut path segmented by the expanded vertex and the straight line

segments between the shortcut-path points and the original vertices, as shown in Figure 6. After completing the above path embedding, the expanded LRCG is used as the basis for the LRCG-HCBS algorithm and the CDPS algorithm to complete the subsequent path planning.

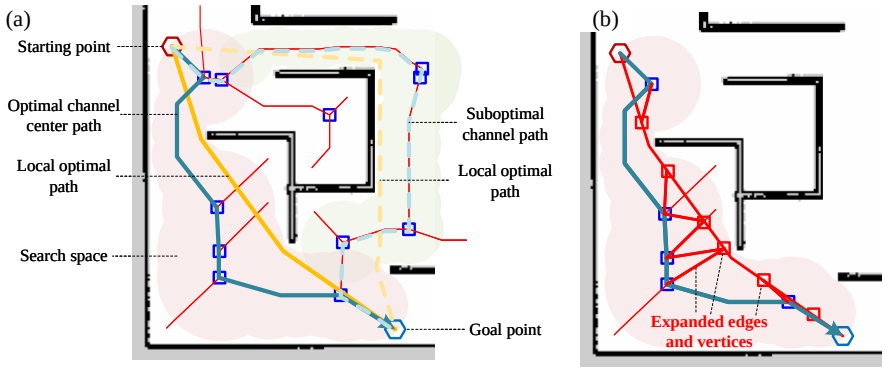

Figure 6. Shortcut path estimation and embedding. (**a**) LRCG-based bounded suboptimal path planning and local optimal path estimation. (**b**) Expansion of LRCG in shortcut-path embedding.

6. Simulations

The LRCG-HCBS path planning algorithm and CDPS post-processing optimization algorithm proposed in this paper were integrated and verified in simulations based on the Robotics Operating System (ROS) environment. The simulations were performed on an Intel i7-11800H CPU with 2.3 GHz and 32 GB of RAM.

To validate the effectiveness of the proposed method, three multi-agent pathfinding algorithms, including priority-based search (PBS), large neighborhood search (LNS), and conflict-based search (CBS), were adopted as benchmark methods, all of which employ occupancy grid maps as the path search space. Additionally, CBS-MP, a multi-robot path planning approach based on probabilistic road maps (PRMs), was included as another benchmark. The implementation details of these benchmark methods are as follows:

(1) PBS [8]: The algorithm employs a two-level priority constraint search framework. The high level searches over priority orderings among agents, and the low level invokes safe interval path planning (SIPP) to plan paths sequentially from high to low priority, with higher-priority paths treated as dynamic obstacles for lower-priority agents.

(2) LNS [11]: The algorithm starts from an initial feasible solution and iteratively applies destroy-and-repair operations, i.e., selecting a subset of agents for path replanning and replacing the current solution with an improved solution, yielding an optimized set of paths. The underlying path planning also uses the SIPP algorithm.

(3) CBS [4]: The algorithm employs a two-level spatiotemporal constraint search framework. The high level detects and resolves conflicts among agents in a constraint tree, while the low level independently plans the shortest path for each agent subject to the given constraints. The high level branches by adding spatiotemporal constraints to resolve conflicts, iterating until a conflict-free optimal solution is found.

(4) CBS-MP [3]: The algorithm differs from the above-mentioned grid-based search methods. Each agent maintains its own probabilistic road map as the search space, and a CBS-like hierarchical framework is adopted for spatiotemporal constraint handling and path search.

The proposed LRCG-based planning method and the four benchmark algorithms were all implemented in Python 3.8. In this study, the total path time was adopted as the primary metric for evaluating path quality. This choice is consistent with the objective function in Equation (1), which minimizes the cumulative travel time of all UAVs, and is also commonly used in related multi-agent path planning and multi-UAV path planning

studies to measure the overall motion efficiency of a planned solution. Compared with local geometric indicators, such as path smoothness or clearance, total path time directly reflects the global efficiency of the multi-UAV mission under the same planning objective.

In the simulations, the minimum avoidance distance with other UAVs was set to $d_{\min} = 0.5$ m. The maximum translational velocity of the UAVs was set to 1 m/s. Two occupancy grid map scenarios with different obstacle densities were built for multi-UAV path planning simulations, and their quantitative environment statistics are summarized in Table 1. In two multi-UAV path planning tasks, there were one and two pairs of UAVs (two UAVs and four UAVs) starting from two points that were symmetrical about the center point to reach each other's starting point. The number of path replanning iterations in the LNS was set to 10. In CBS-MP, N_V denotes the predestinated number of sampled vertices in individual PRM of each agent, where each vertex represents a valid configuration and a larger N_V corresponds to a denser roadmap representation. The value of N_V was set to 500.

Table 1. Quantitative statistics of the test environments.

Metrics	Scenario 1	Scenario 2
Map size	300 × 300 cells, 15 × 15 m	
Free-space ratio	0.6417	0.7514
Obstacle density	0.3583	0.2486
Corridor width range	[0.8 m, 2.8 m]	[1.4 m, 4.9 m]

The planned paths of the methods in the simulations of two planning tasks and two scenarios are illustrated in Figure 7. The comparison of the planning time and total path time corresponding to the four methods in the simulations is shown in Table 2. These comparisons demonstrate that although the path time of the LRCG-based planning algorithm is similar to that of PBS, LNS, and CBS, and slightly weaker than that of CBS-MP, the LRCG-based planning algorithm shows a significantly reduced planning time compared to the benchmark methods. The planning time of the LRCG-based planning algorithm is 13.0–31.5% of PBS, 3.5–14.5% of LNS, 9.3–30.0% of CBS, and 22.2–51.2% of CBS-MP.

Table 2. Performance comparison of the algorithms in simulations.

Metrics	Tests	Algorithms				
		PBS	LNS	CBS	CBS-MP	LRCG
Planning time (s)	Scenario 1 2-UAV	3.77	13.38	3.73	2.28	**1.12**
	Scenario 1 4-UAV	14.85	32.19	18.55	9.13	**4.67**
	Scenario 2 2-UAV	6.61	24.89	9.27	3.87	**0.86**
	Scenario 2 4-UAV	18.41	57.21	14.76	9.03	**2.65**
Total path time (s)	Scenario 1 2-UAV	41.36	41.54	41.31	**40.96**	41.22
	Scenario 1 4-UAV	81.77	84.27	**81.67**	81.68	82.56
	Scenario 2 2-UAV	41.60	41.60	41.51	**39.58**	40.39
	Scenario 2 4-UAV	83.23	83.23	83.15	**79.31**	83.12

To verify the effectiveness of the LRCG-based planning algorithm in planning a large-scale multi-UAV system, we designed a task in which eight UAVs started from the edge of the map and reached the starting positions of other UAVs on the opposite side. The planned paths are shown in Figure 8(a1–a5), and the performance comparison is shown in Table 3. The variation in the minimum distance between UAVs are shown in Figure 8(b1–b5), which meets the requirements of the minimum avoidance distance. It can be seen that the LRCG-based planning algorithm can adapt well to the needs of large-scale path planning and has better real-time performance without significantly losing path quality.

Table 3. Performance comparison of the algorithms in 8-UAV simulations.

Metrics	Algorithms				
	PBS	**LNS**	**CBS**	**CBS-MP**	**LRCG**
Planning time (s)	712.26	104.30	1310.56	108.29	**38.90**
Total path time (s)	145.76	151.29	**145.04**	150.09	153.03

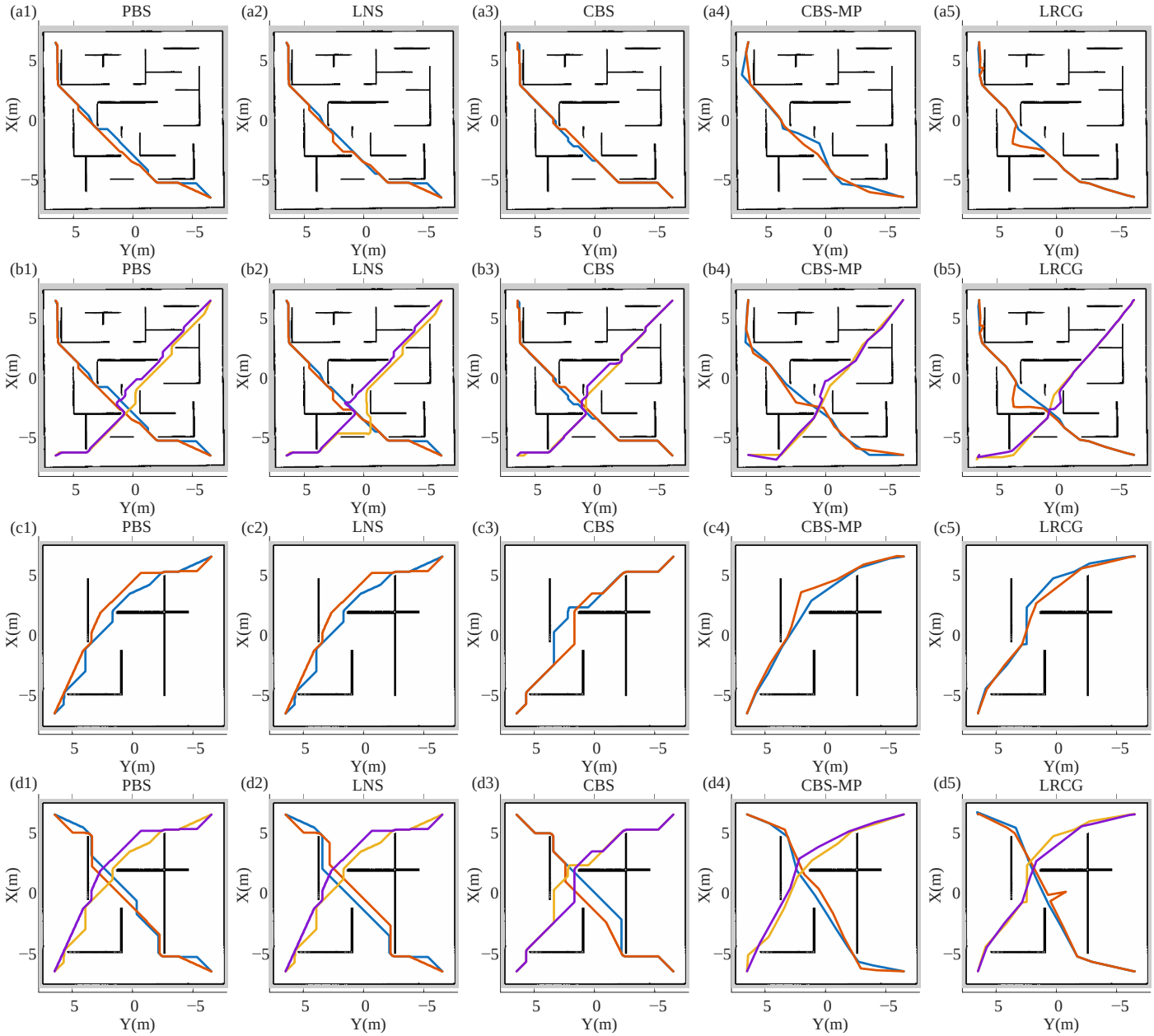

Figure 7. Simulation planned paths of multi-UAV path planning. Each column corresponds to one algorithm (PBS, LNS, CBS, CBS-MP, and LRCG), and each row corresponds to the same comparison group. (**a1**–**a5**) Planned paths of 2-UAV test in scenario 1. (**b1**–**b5**) Planned paths of 4-UAV test in scenario 1. (**c1**–**c5**) Planned paths of 2-UAV test in scenario 2. (**d1**–**d5**) Planned paths of 4-UAV test in scenario 2.

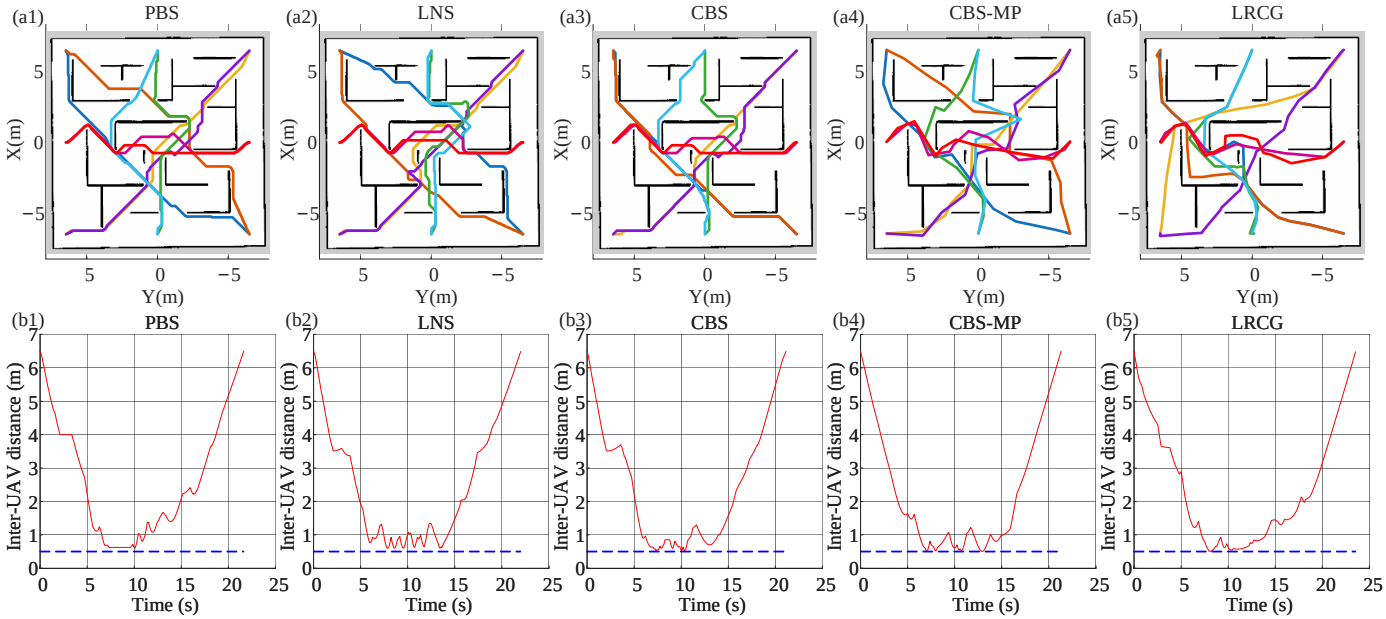

Figure 8. Simulation planned paths and inter-UAV distances of 8-UAV path planning with PBS, LNS, CBS, CBS-MP, and LRCG. Each column corresponds to one algorithm (PBS, LNS, CBS, CBS-MP, and LRCG), while the first row (**a1**–**a5**) shows the planned paths and the second row (**b1**–**b5**) shows the variation in minimum inter-UAV distances. (The blue dashed lines indicate the safety distance.)

In the simulations, we conducted 16 sets of random-point experiments for varying numbers of UAVs and compared the planning time and total path time, as illustrated in Figure 9; the circular points in the box plots indicate statistical outliers. The LRCG-based planning algorithm demonstrates a more pronounced reduction in planning time as the number of UAVs increases.

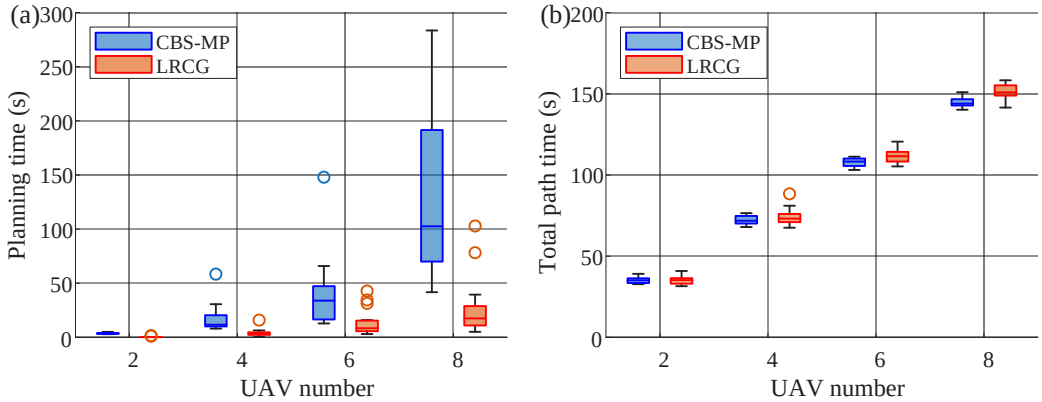

Figure 9. Box plots for the performance comparison between LRCG-based algorithm and the CBS-MP with a varying number of UAVs in simulations (16 samples). (**a**) Planning time comparison. (**b**) Total path time comparison.

Further comparative simulations were conducted between the LRCG-based planning algorithm and the CBS-MP with varying parameters. For the same set of starting points and goal points, the predestinated number of vertices of the initial PRM in the CBS-MP was changed, multiple-path planning was performed considering the randomness of CBS-MP, and the results were compared with the results of the LRCG-based planning algorithm. In the simulation, comparisons of planning time and total path time are shown in Figure 10(a1,b1,a2,b2), respectively. It can be seen that a more refined PRM of CBS-MP produces shorter planned paths, but it also leads to an increase in planning time. Compared with CBS-MP, the planning time required for obtaining a path of basically the same quality

by the LRCG-based planning algorithm is significantly shortened, demonstrating a higher planning efficiency. In contrast to the random sampling method utilized in CBS-MP, the LRCG-based planning algorithm demonstrates superior stability in both planning efficiency and performance outcomes.

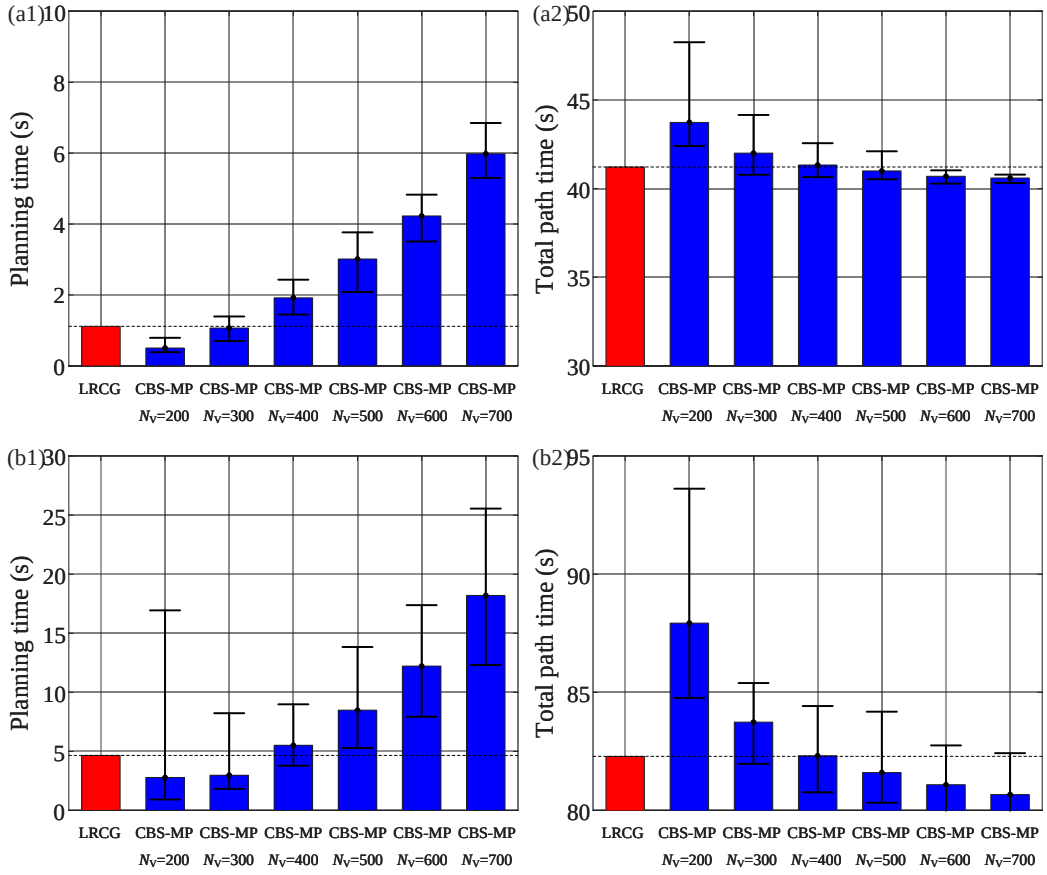

Figure 10. Performance comparison between LRCG-based planning method and the CBS-MP with varying parameters in simulations (16 samples). (**a1**,**a2**) Comparison of 2-UAV planning test. (**b1**,**b2**) Comparison of 4-UAV planning test. (The black dashed lines indicate the metrics of LRCG-based planning method.)

To verify the impact of the simplification coefficient λ on the performance of the planning algorithm, we conducted multiple multi-UAV path planning simulation experiments on the LRCG-based planning algorithm under different λ values. The corresponding planning time and total path time are shown in Figure 11a,b, respectively. The results show that, under the tested simulation settings, as λ increases, the overall planning time shows a downward trend, while the total path time is only slightly affected. This indicates that λ can be used to adjust the computational efficiency of the algorithm while maintaining solution quality in the tested scenarios. It should be noted that a suitable value of λ may depend on the map structure, obstacle distribution, and task congestion level.

An ablation study was conducted to evaluate the contribution of the LRCG-based planning framework. Two ablation variants were considered:

(1) LRCG-NSE: This variant removed the shortcut-path embedding (SE) process in the LRCG method, while retaining the subsequent LRCG-HCBS path search and CDPS-based post-processing.

(2) LRCG-NPP: This variant removed the CDPS-based post-processing (PP) process and directly used the result of LRCG-HCBS as the final solution.

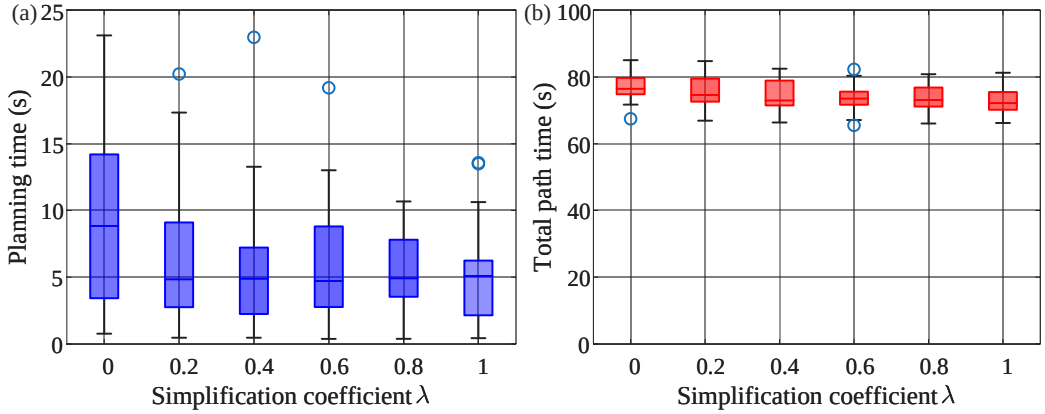

Figure 11. Box plots for the performance comparison between LRCG-based planning method with a varying simplification coefficient in simulations (16 samples). (**a**) Planning time comparison. (**b**) Total path time comparison.

Multiple random task instances were tested in the four-UAV scenario. For each ablation variant, the planning time and total path time were normalized by the corresponding result of the complete method, as shown in Figure 12. The relative total path time shows that removing either the shortcut-path embedding process or the CDPS-based post-processing process increases the total path time, indicating a degradation in solution quality. These results verify the effectiveness of shortcut-path embedding and CDPS-based post-processing in improving the planned paths.

To evaluate the influence of the random processing order in the post-processing stage of the LRCG-based planning method, we conducted repeated experiments under the same input conditions. For the two-UAV, four-UAV, and eight-UAV test scenarios, the planning process was repeated multiple times. The normalized planning time and normalized total path time are shown in Figure 13, where the results in each scenario are normalized by the corresponding mean value over repeated runs. The relative variation in planning time remains within 10%, while the variation in total path time remains within 0.5%. These results indicate that the final path quality is highly stable, demonstrating that the proposed method is insensitive to the random order used in the post-processing stage.

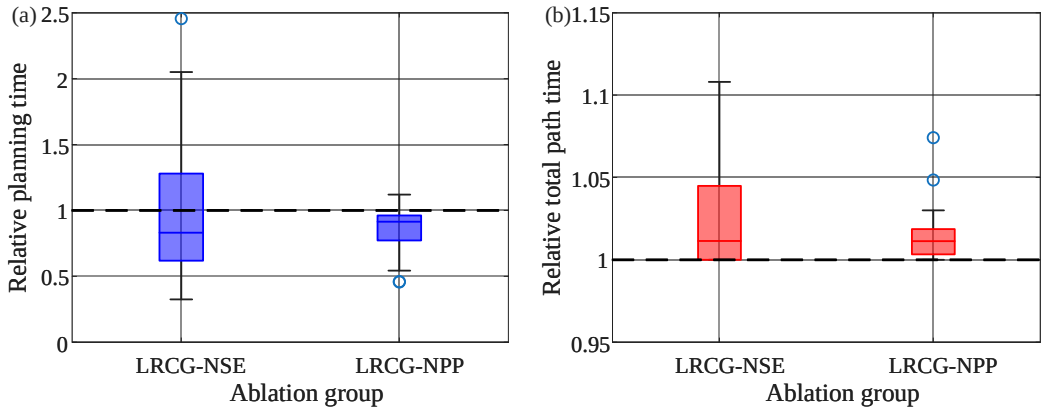

Figure 12. Box plots for the performance comparison between LRCG-based planning method and its ablative variants in simulations (16 samples). (**a**) Planning time comparison. (**b**) Total path time comparison. (The black dashed lines indicate the metrics of LRCG-based planning method.)

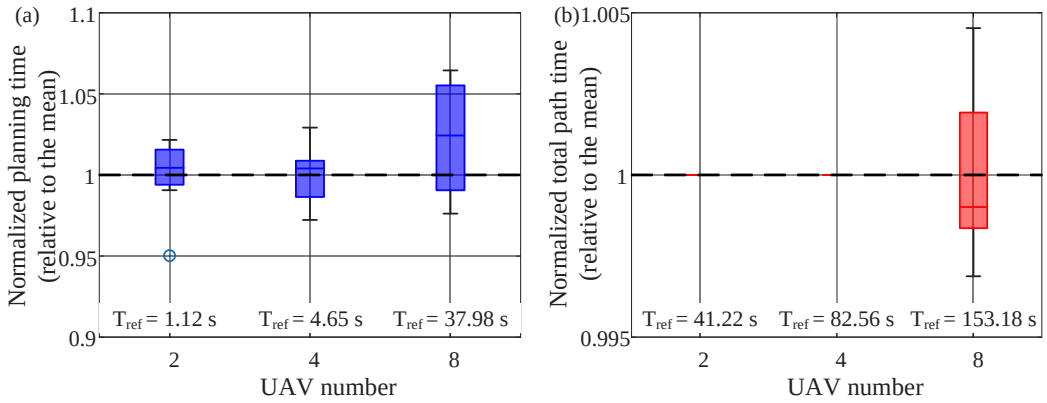

Figure 13. Box plots for the relative performance metrics in the robustness analysis simulations (16 samples). (**a**) Planning time. (**b**) Total path time. (The black dashed lines indicate the mean values.)

7. Conclusions

In this paper, we proposed the LRCG model and its channel refinement model to describe the accessible space of the environment using a lightweight data structure. Building upon the LRCG model, we developed the LRCG-HCBS algorithm and the CDPS algorithm to address multi-UAV path planning and its subsequent optimization. Simulation and experiment results demonstrated that the LRCG-based planning algorithm outperformed existing approaches in terms of path planning efficiency. For practical deployment, considering unexpected dynamic obstacles in real-world mission scenarios, local motion planning methods can be adopted to avoid risks, with the global planning path obtained by the LRCG-based planning algorithm as a reference. Future work will focus on applying the LRCG model to multi-UAV task allocation and scheduling in complex environments.

Author Contributions: Conceptualization, H.T. and W.Y.; methodology, H.T.; software, H.T.; validation, H.T. and L.L.; formal analysis, H.T.; investigation, H.T.; resources, G.S.; data curation, H.T.; writing—original draft preparation, H.T.; writing—review and editing, H.T.; visualization, W.Y.; supervision, W.Y.; project administration, G.S.; funding acquisition, G.S. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Natural Science Foundation of China under Grant U23A20346 and Grant 62473109, in part by the Key R&D Program of Heilongjiang Province under Grant 2025ZX04A01, and in part by the Sichuan Province Science and Technology Support Program under Grant 2025YFHZ0011.

Data Availability Statement: The datasets presented in this article are not readily available because the data are part of an ongoing study. Requests to access the datasets should be directed to the authors via email.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Čáp, M.; Novák, P.; Kleiner, A.; Seleckỳ, M. Prioritized Planning Algorithms for Trajectory Coordination of Multiple Mobile Robots. *IEEE Trans. Autom. Sci. Eng.* **2015**, *12*, 835–849.
2. Ren, Z.; Rathinam, S.; Choset, H. A Conflict-Based Search Framework for Multiobjective Multiagent Path Finding. *IEEE Trans. Autom. Sci. Eng.* **2023**, *20*, 1262–1274.
3. Solis, I.; Motes, J.; Sandström, R.; Amato, N.M. Representation-optimal multi-robot motion planning using conflict-based search. *IEEE Robot. Autom. Lett.* **2021**, *6*, 4608–4615.
4. Sharon, G.; Stern, R.; Felner, A.; Sturtevant, N.R. Conflict-based search for optimal multi-agent pathfinding. *Artif. Intell.* **2015**, *219*, 40–66.

5. Barer, M.; Sharon, G.; Stern, R.; Felner, A. Suboptimal variants of the conflict-based search algorithm for the multi-agent pathfinding problem. In Proceedings of the International Symposium on Combinatorial Search, Prague, Czech Republic, 15–17 August 2014; Volume 5, pp. 19–27.
6. Sharon, G.; Stern, R.; Felner, A.; Sturtevant, N. Meta-agent conflict-based search for optimal multi-agent path finding. In Proceedings of the International Symposium on Combinatorial Search, Niagara Falls, ON, Canada, 19–21 July 2012; Volume 3, pp. 97–104.
7. Boyarski, E.; Felner, A.; Stern, R.; Sharon, G.; Betzalel, O.; Tolpin, D.; Shimony, E. ICBS: The improved conflict-based search algorithm for multi-agent pathfinding. In Proceedings of the International Symposium on Combinatorial Search, Ein Gedi, Israel, 11–13 June 2015; Volume 6, pp. 223–225.
8. Ma, H.; Harabor, D.; Stuckey, P.J.; Li, J.; Koenig, S. Searching with consistent prioritization for multi-agent path finding. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, pp. 7643–7650.
9. Okumura, K.; Machida, M.; Défago, X.; Tamura, Y. Priority inheritance with backtracking for iterative multi-agent path finding. *Artif. Intell.* **2022**, *310*, 103752.
10. Zhang, X.; Xiong, G.; Wang, Y.; Teng, S.; Chen, L. D-PBS: Dueling priority-based search for multiple nonholonomic robots motion planning in congested environments. *IEEE Robot. Autom. Lett.* **2024**, *9*, 6288–6295.
11. Li, J.; Chen, Z.; Harabor, D.; Stuckey, P.J.; Koenig, S. Anytime multi-agent path finding via large neighborhood search. In *Proceedings of the International Joint Conference on Artificial Intelligence, Montreal, QC, Canada, 19–27 August 2021*; Association for the Advancement of Artificial Intelligence (AAAI): Washington, DC, USA, 2021; pp. 4127–4135.
12. Li, J.; Chen, Z.; Harabor, D.; Stuckey, P.J.; Koenig, S. MAPF-LNS2: Fast repairing for multi-agent path finding via large neighborhood search. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual, 22 February–1 March 2022; Volume 36, pp. 10256–10265.
13. Okumura, K. LaCAM: Search-based algorithm for quick multi-agent pathfinding. In Proceedings of the AAAI Conference on Artificial Intelligence, Washington, DC, USA, 7–14 February 2023; Volume 37, pp. 11655–11662.
14. Le, D.; Plaku, E. Multi-robot motion planning with dynamics via coordinated sampling-based expansion guided by multi-agent search. *IEEE Robot. Autom. Lett.* **2019**, *4*, 1868–1875.
15. Sanchez, G.; Latombe, J.C. Using a PRM planner to compare centralized and decoupled planning for multi-robot systems. In *Proceedings of the 2002 IEEE International Conference on Robotics and Automation (ICRA), Washington, DC, USA, 11–15 May 2002*; IEEE: Piscataway, NJ, USA, 2002; Volume 2, pp. 2112–2119.
16. Chi, W.; Wang, J.; Ding, Z.; Chen, G.; Sun, L. A reusable generalized voronoi diagram-based feature tree for fast robot motion planning in trapped environments. *IEEE Sensors J.* **2021**, *22*, 17615–17624.
17. Zhao, W.; Lin, R.; Dong, S.; Cheng, Y. A study of the global topological map construction algorithm based on grid map representation for multirobot. *IEEE Trans. Autom. Sci. Eng.* **2022**, *20*, 2822–2835.
18. Wagner, G.; Choset, H. Subdimensional expansion for multirobot path planning. *Artif. Intell.* **2015**, *219*, 1–24.
19. Bnaya, Z.; Felner, A. Conflict-oriented windowed hierarchical cooperative A*. In *Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 31 May–7 June 2014*; IEEE: Piscataway, NJ, USA, 2014; pp. 3743–3748.
20. Hönig, W.; Kiesel, S.; Tinka, A.; Durham, J.; Ayanian, N. Conflict-based search with optimal task assignment. In Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, Stockholm, Sweden, 10–15 July 2018.
21. Han, S.D.; Yu, J. Ddm: Fast near-optimal multi-robot path planning using diversified-path and optimal sub-problem solution database heuristics. *IEEE Robot. Autom. Lett.* **2020**, *5*, 1350–1357.
22. Li, Q.; Lin, W.; Liu, Z.; Prorok, A. Message-aware graph attention networks for large-scale multi-robot path planning. *IEEE Robot. Autom. Lett.* **2021**, *6*, 5533–5540.
23. Yang, Q.; Lu, J.; Zhang, Y.; Shao, S. Multi-UAV cooperative path planning via graph neural network and proximal policy optimization. *IEEE Access* **2025**, *13*, 193575–193588.
24. Liu, G.; Tang, Y.; Tan, Z.; Fang, M. A multi-stage optimization strategy for multi-UAV path planning in complex urban 3D environments with obstacle avoidance. *Complex Intell. Syst.* **2026**, *12*, 17.
25. Debnath, D.; Vanegas, F.; Sandino, J.; Gonzalez, F. DECK-GA: A Hybrid Clustering and Distance Efficient Genetic Algorithm for Scalable Multi-UAV Path Planning. In *Proceedings of the 2025 International Conference on Unmanned Aircraft Systems (ICUAS), Charlotte, NC, USA, 14–17 May 2025*; IEEE: Piscataway, NJ, USA, 2025; pp. 301–308.
26. Krontiris, A.; Sajid, Q.; Bekris, K.E. *Towards Using Discrete Multiagent Pathfinding to Address Continuous Problems*; Association for the Advancement of Artificial Intelligence (AAAI): Washington, DC, USA, 2012.
27. Kottinger, J.; Almagor, S.; Lahijanian, M. Conflict-based search for multi-robot motion planning with kinodynamic constraints. In *Proceedings of the 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Kyoto, Japan, 23–27 October 2022*; IEEE: Piscataway, NJ, USA, 2022; pp. 13494–13499.

28. Theurkauf, A.; Kottinger, J.; Ahmed, N.; Lahijanian, M. Chance-Constrained Multi-Robot Motion Planning Under Gaussian Uncertainties. *IEEE Robot. Autom. Lett.* **2023**, *9*, 835–842.
29. Moldagalieva, A.; Ortiz-Haro, J.; Toussaint, M.; Hönig, W. db-cbs: Discontinuity-bounded conflict-based search for multi-robot kinodynamic motion planning. In *Proceedings of the 2024 IEEE International Conference on Robotics and Automation (ICRA), Yokohama, Japan, 13–17 May 2024*; IEEE: Piscataway, NJ, USA, 2024; pp. 14569–14575.
30. Şenbaşlar, B.; Sukhatme, G.S. Dream: Decentralized real-time asynchronous probabilistic trajectory planning for collision-free multi-robot navigation in cluttered environments. *IEEE Trans. Robot.* **2024**, *41*, 573–592.
31. Westheider, J.; Rückin, J.; Popović, M. Multi-UAV adaptive path planning using deep reinforcement learning. In *Proceedings of the 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Detroit, MI, USA, 1–5 October 2023*; IEEE: Piscataway, NJ, USA, 2023; pp. 649–656.
32. Shome, R.; Solovey, K.; Dobson, A.; Halperin, D.; Bekris, K.E. dRRT*: Scalable and informed asymptotically-optimal multi-robot motion planning. *Auton. Robot.* **2020**, *44*, 443–467.
33. Kavraki, L.E.; Svestka, P.; Latombe, J.C.; Overmars, M.H. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. Robot. Autom.* **1996**, *12*, 566–580.
34. Okumura, K.; Tamura, Y.; Défago, X. Iterative refinement for real-time multi-robot path planning. In *Proceedings of the 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Prague, Czech Republic, 27 September–1 October 2021*; IEEE: Piscataway, NJ, USA, 2021; pp. 9690–9697.
35. Dayan, D.; Solovey, K.; Pavone, M.; Halperin, D. Near-optimal multi-robot motion planning with finite sampling. *IEEE Trans. Robot.* **2023**, *39*, 3422–3436.
36. Fu, J.; Yao, W.; Sun, G.; Ma, Z.; Dong, B.; Ding, J.; Wu, L. Multirobot cooperative path optimization approach for multiobjective coverage in a congestion risk environment. *IEEE Trans. Syst. Man, Cybern. Syst.* **2023**, *54*, 1816–1827.
37. Tsardoulias, E.G.; Serafi, A.T.; Panourgia, M.N.; Papazoglou, A.; Petrou, L. Construction of minimized topological graphs on occupancy grid maps based on GVD and sensor coverage information. *J. Intell. Robot. Syst.* **2014**, *75*, 457–474.
38. Silver, D. Cooperative pathfinding. In Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, Marina del Rey, CA, USA, 1–5 June 2005; Volume 1, pp. 117–122.
39. Yen, J.Y. Finding the k shortest loopless paths in a network. *Manag. Sci.* **1971**, *17*, 712–716.