Transportation Research Part C 160 (2024) 104536

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

A distributed route network planning method with congestion pricing for drone delivery services in cities

Xinyu He [b], Lishuai Li [a,b,*], Yanfang Mo [b], Jianxiang Huang [c], S. Joe Qin [d]

[a] *School of Data Science, City University of Hong Kong, Hong Kong Special Administrative Region*
[b] *Hong Kong Institute for Data Science, City University of Hong Kong, Hong Kong Special Administrative Region*
[c] *Department of Urban Planning and Design, Faculty of Architecture, The University of Hong Kong, Hong Kong Special Administrative Region*
[d] *Department of Computing and Decision Sciences, Lingnan University, Hong Kong Special Administrative Region*

ARTICLE INFO

Keywords:
Unmanned aerial vehicle
Drone delivery
Traffic management
Multi-path planning
Route network design

ABSTRACT

Unmanned aerial vehicle (UAV)-based commercial services, exemplified by drone delivery, have captured wide interest in tech companies, entrepreneurs, and policymakers. Structured route-based UAV operations have been implemented for traffic management of UAVs in support of commercial delivery services in cities. Yet, its essence, multi-path planning with constraints is not well solved in the existing literature. Centralized planning might result in inefficiencies and unfairness in the allocation of precious urban airspace to individual routes. This paper describes a novel distributed route planning method to support UAV operations in a high-density urban environment. The method allows each origin–destination (OD) pair to compete against other OD pairs for an optimized route (e.g. shortest distance), coordinated by a system-level evaluation, leading to a network design that maximizes the performance of not only the individual routes but also the entire system. The core concept is the introduction of congestion pricing, a soft constraint to coordinate the allocation of airspace. The method is tested in standard 2D scenarios and compared with other state-of-the-art methods. The results show that (1) the method is able to generate routes with short individual distances as well as occupying the least airspace by the route network; (2) in some complex scenarios, the method is able to find a solution in a short period of time while other state-of-the-art method fails. The method has also been applied to a real urban environment (Mong Kok in Hong Kong) to demonstrate its capability.

1. Introduction

Unmanned aerial vehicle (UAV)-based commercial services, exemplified by drone delivery, is a rapidly emerging industry. The number of commercial drone deliveries has increased from 34,000 in 2019 to 482,000 in 2021 and is estimated to reach 1.4 million by the end of 2022 (Sarina et al., 2022), not including the test flights to develop and prove the technology. However, efficient and safe management of the large volumes of drones operating in dense urban environments is a key challenge. There are several national-level R&D programs that are exploring different concepts of operations, data exchange requirements, and supporting frameworks to enable drone operations at scales, such as NASA/FAA unmanned aircraft system traffic management (UTM) (NASA, 2021), SESAR U-space (SESAR, 2019), Singapore uTM-UAS (Mohamed Salleh et al., 2018), and J-UTM (Ushijima, 2017). In these programs, a range of concepts of operations (ConOps) for traffic and airspace management have been theorized (Bauranov and

* Corresponding author at: School of Data Science, City University of Hong Kong, Hong Kong Special Administrative Region.
E-mail addresses: x.y.he@my.cityu.edu.hk (X. He), lishuai.li@cityu.edu.hk (L. Li), yanfang.mo@cityu.edu.hk (Y. Mo), jxhuang@hku.hk (J. Huang), joeqin@ln.edu.hk (S.J. Qin).

https://doi.org/10.1016/j.trc.2024.104536
Received 25 February 2023; Received in revised form 6 January 2024; Accepted 18 February 2024
Available online 27 February 2024
0968-090X/© 2024 Elsevier Ltd. All rights reserved.

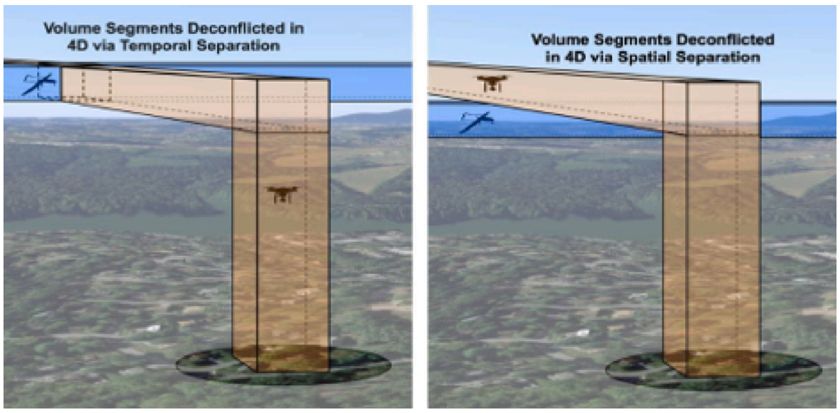

Volume Segments Deconflicted in 4D via Temporal Separation
Volume Segments Deconflicted in 4D via Spatial Separation

Fig. 1. UTM ConOps: temporal separation vs. spatial separation (NASA, 2021).

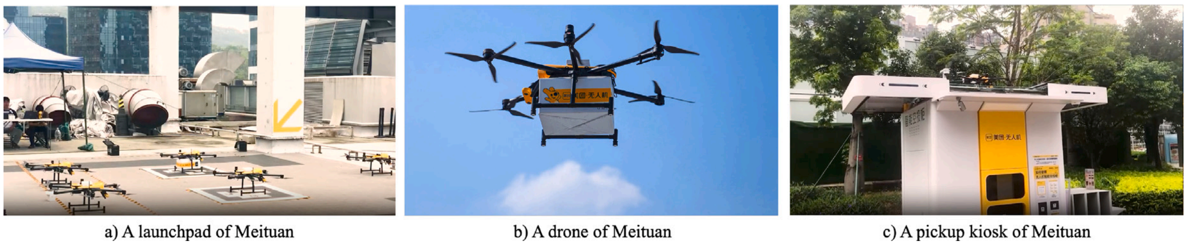


Fig. 2. Drone delivery services operated by Meituan in Shenzhen (Yang, 2023).

Rakas, 2021; NASA, 2021; SESAR, 2021), which can be broadly categorized into two groups based on whether they require airspace structure or not. Free-flight-based operations do not require any airspace structure. Each UAV flies its desired route and relies on real-time conflict detection and avoidance (Hoekstra et al., 2002; Yang and Wei, 2018). Structure-based operations use airspace structures such as layers, zones, and tubes to organize traffic flows to reduce potential conflicts. These structures reduce airspace complexity and management workload (EUROCONTROL, 2018; Jang et al., 2017; Krozel et al., 2001; Sunil et al., 2015). Additional benefits come from the capability of addressing safety, security, noise, and privacy issues related to UAV operations by designing appropriate airspace structures (EUROCONTROL, 2018). Structured route networks could use either temporal or spatial separation for traffic management, as shown in Fig. 1. Our study focuses on the spatially separated route network planning problem.

Structured route-based UAV operations, also referred to as tube-based operations, have been deployed to offer commercial drone delivery services in cities in China, such as Shenzhen and Hangzhou. MIT technology review reports the implementation and operation of drone delivery services (Fig. 2) by Meituan, a Chinese food delivery platform, in the city of Shenzhen (Yang, 2023). The company has been developing drone delivery since 2017, and for the past year and a half, it has been operating such delivery routes regularly in Shenzhen, a city known for its mature drone supply chain. The drones deliver to pickup kiosks close to residential or office buildings, rather than delivering directly to consumers' doorsteps. This model enables drones to fly predetermined routes between launchpads and kiosks, simplifying navigation in dense urban areas. Meituan made over 100,000 drone deliveries in Shenzhen in 2022, despite some impediments such as distance limits and operating hours.

An illustration of this structured route network is shown in Fig. 3. In a route network, there are multiple vertiports that drones will travel in-between. Air routes are unidirectional paths designed to connect these vertiports, more specifically, connecting the approach/departure fixes of these vertiports. The air routes are pre-defined to be conflict-free. Drones fly sequentially along an air route (inside the path) and maintain minimum inter-drone spacing (the spacing requirement between a preceding drone and a following drone). The tube around the air route is referred to as "the path" in this paper, and its width and height are set to guarantee the separation of drones in different paths. In addition, a "buffer zone" is added around the path to serve as an extra safety margin. The detailed definition and description of an air route network and its components can be found in He et al. (2022).

Currently, no existing method can directly address the challenge of air route network planning. Most existing research in Multi-Agent Path Finding (MAPF) has primarily focused on task-oriented or in-flight path planning within the context of drone delivery operations. However, there is a gap in the literature regarding infrastructure-oriented or pre-flight path planning for the design of air route networks, specifically for drone delivery applications in densely populated urban areas. The main challenge lies in achieving a balance between optimizing individual paths and optimizing the system or infrastructure as a whole. Single-path planning, which aims to minimize factors like path length, energy consumption, and ground impact for a given origin–destination pair, has been extensively studied. However, when multiple paths utilize the same airspace, conflicts and interdependencies arise, necessitating

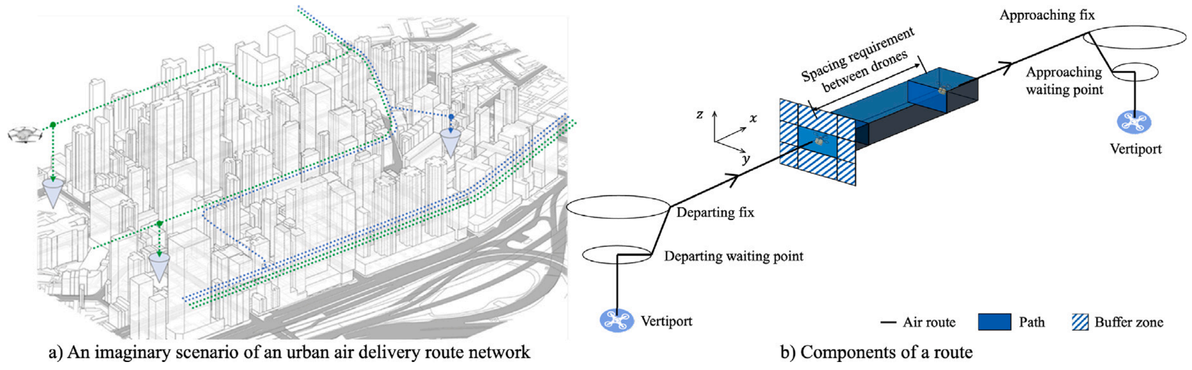

a) An imaginary scenario of an urban air delivery route network
b) Components of a route
Approaching fix
Spacing requirement between drones
Approaching waiting point
Vertiport
z
x
y
Departing fix
Departing waiting point
Vertiport
Air route
Path
Buffer zone

Fig. 3. Illustration for the structured route network.

system-level optimization. When paths are optimized individually, the allocation of scarce urban airspace is not optimized, which can lead to air traffic conflicts and congestion, potentially causing system failure. Centralized planning can achieve system-level optimization, but it may also result in inefficiencies and inequities in the allocation of valuable urban airspace to individual paths.

To address this challenge, we propose a distributed route network planning method tailored for drone delivery services in densely populated urban areas. This method incorporates the use of pickup kiosks located near residential or office buildings. It employs congestion pricing as a soft constraint, proactively mitigating traffic conflicts and coordinating airspace allocation among different routes based on system-level information. Our method adopts a distributed planning approach in which each origin–destination (OD) pair competes for an optimized route (e.g., shortest distance) following system-level coordination. This aims to maximize the performance of both individual paths and the entire system. Our approach fills a gap in current research, concentrating on infrastructure-oriented or pre-flight path planning for air route network design, and balancing the trade-off between optimizing individual paths and the system/infrastructure as a whole. This innovative focus could provide pivotal insights for future drone delivery services in urban environments.

The reminder of this paper is structured as follows. An overview of the related academic literature is provided in Section 2. The mathematical problem statement is provided in Section 3. The proposed multi-path planning technique is described in Section 4, and its theoretical properties are discussed in Section 5. In Section 6, the proposed method is tested in toy examples, standard 2D scenarios, and real-world scenarios. Finally, Section 7 concludes the paper and discusses the prospects of further work on this topic.

2. Related work

There are several types of research problems associated with drone delivery operations that have been studied in the literature, yet not all are directly related to the traffic management and route network planning problem in this paper. For example, several papers on UAV delivery investigated the vehicle routing problem (VRP), which is unrelated to traffic management and route network planning, though similar by name. The vehicle routing problem involves the coordination and allocation of drones (and sometimes trucks) to deliver small parcels from repositories to geographically distributed customers (Murray and Chu, 2015; Murray and Raj, 2020; Sacramento et al., 2019; Schermer et al., 2019). In this type of study, routes between vertiports, repositories, or customers are abstracted as a link between nodes on a graph. The detailed operations of UAV flight trajectories are not considered. Another group of studies works on the drone traffic management problem. They take temporal separations to resolve traffic conflicts among UAVs, which normally involves detecting conflicts and delaying UAVs accordingly (Zhao et al., 2018; Wu et al., 2021; Tan et al., 2019; Yang and Wei, 2021; Tang et al., 2021). The temporal solution is a tactical approach for traffic management, which is different from this study's focus — strategic planning of spatially separated air routes.

This study is about planning structured routes for drone operations, more specifically, designing multiple conflict-free paths with certain optimization objectives. Therefore, we provide a literature review focused on Multi-Agent Path Finding (MAPF) methods, summarized into two groups: ***centralized methods*** and ***distributed methods***, highlighting the state-of-the-art methods in each group and why they cannot be directly applied to solve the problem of air route network planning in this paper.

2.1. Centralized multi-agent path finding methods

A centralized approach is the most commonly used one for multi-agent path-finding problems, in which feasible routes are found by a single computation entity (Felner et al., 2017). It minimizes the summation of individual path costs and avoids conflicts among paths, yet the summation of individual path costs limits the consideration of self-interested agents as well as system-level non-additive costs, or joint-path costs, such as airspace utilization. In the centralized approach, ***optimal algorithms*** directly search the entire state space to find the best solutions, while ***suboptimal algorithms*** use heuristics to find a solution, such as priority-based searches and rule-based planning. However, either can be directly applied to solve the problem in this paper. The optimal algorithms can find an optimal solution or a feasible solution with better optimality, but they cannot scale well for real-world implementation when the search space or the number of routes to plan is large. On the other hand, the suboptimal algorithms can quickly find a feasible solution in large scenarios; however, the generated paths may be far from being optimal, and they may fail to find paths when the solution space is small.

In the ***optimal algorithms***, the state spaces are either ***agent-based*** or ***conflict-based***. In the ***agent-based*** state space, each state includes the locations of k agents at a particular time as a vector; a state is valid if its contained locations are different. For each state of k agents, there can be 4^k neighbors for the state if each agent can take actions along four dimensions (forward/backward, left/right, up/down, wait). For a state with 10 agents, there are $4^{10}(\approx 10^6)$ neighbors. Thus, searching the entire state space is computationally infeasible. Some techniques have been developed for exponential speed-up, such as independent detection and avoiding surplus nodes. Independence detection divides routes into independent groups and solves these groups separately (Standley, 2010; Wagner and Choset, 2015), two groups are independent only if there is no inter-conflict between their optimal solutions. Surplus nodes are the generated nodes that will never be expanded to find an optimal solution (Standley, 2010; Felner et al., 2012; Goldenberg et al., 2014). After implementing the exponential speedup techniques, the methods can find solutions for real-world scenarios; however, the solution quality degrades rapidly, and the computational time increases rapidly as the number of routes increases. The ***conflict-based*** state space is a constraint tree. Each node includes a set of constraints, a solution for all routes, and the cost of the solution. Each constraint includes the location and time of a conflict and the route that can solve the conflict. The solution of a node satisfies the node constraints. The tree progressively extends from the root node to the leaf node. The root node does not consider any conflicts and has an empty set of constraints and its solution is a set of individual optimal paths. A child node inherits the constraints of the parent node and adds a new constraint from existing conflicts. In each leaf, all the conflicts among the routes are solved such that the paths are conflict-free. Conflict-based search (CBS) (Sharon et al., 2015) is a state-of-the-art method that searches over such a constraint tree at two levels. The high-level search starts from the root node and ends at the leaf node which has the minimum cost. The branch-and-bound method is applied to prune the tree to speed up the process. A low-level search is invoked to find paths for each node that satisfy their constraints. Its variants (Barer et al., 2014; Cohen et al., 2019, 2016) relax some assumptions and conditions in CBS to speed up the search process at the cost of optimality.

There are also reduction-based methods for multi-path finding problems under the ***optimal algorithms*** group, such as transforming the MAPF problem into integer programming with constraints (Surynek, 2012; Yu and LaValle, 2016), and then solving it with high-performance solvers. The reduction-based methods can find feasible solutions with guaranteed optimality; however, they are very slow to find solutions for large complex 3D urban environments.

Another group of algorithms under the centralized approach uses ***suboptimal techniques***, mostly heuristics to find a solution, such as ***priority-based*** searches and ***rule-based*** planning. These heuristic methods can quickly find a feasible solution in large scenarios; however, they may fail to find paths, and the generated paths may be far from optimal. In ***priority-based*** methods, routes are planned and sorted sequentially according to pre-defined orders, or priorities. Low-priority routes detour if the airspace is occupied by high-priority routes. Priorities have a significant impact on the performance of planned routes (Warren, 1990) and should be carefully determined. Hierarchical cooperative A* (HCA*) (Silver, 2005) is a typical priority-based method. It uses an arbitrary order to plan one route for one agent at a time and stores it into a reservation table. The planned paths are impassable for later agents. In contrast, windowed HCA* (WHCA*) (Silver, 2005) employs plan-move cycles to generate routes dynamically. In each cycle, the planning phase uses an arbitrary order to plan each path but only reserves the next W steps; then, the moving phase moves each agent on the reserved paths by K ($K \leq W$) steps. Conflict-oriented WHCA* (CO-WHCA*) (Bnaya and Felner, 2014) places windows, i.e., the reservation for the next W steps, around conflicts. To determine the route order for each conflict, CO-WHCA* estimates all possible orders at each conflict and selects the best one. ***Rule-based*** methods plan a path for each route separately and follow specific movement rules to eliminate collisions. Different rules are implemented for different scenarios that do not require massive search steps. The algorithm proposed by Kornhauser et al. (1984) guarantees completeness in theory but is complex to implement. Push and Swap (Luna and Bekris, 2011) uses a "swap" macro to swap location between two adjacent dependent agents, and its variants such as Push and Rotate (De Wilde et al., 2014), Push-and-Spin (Alotaibi and Al-Rawi, 2018), etc., use different macros to suit more situations. Some rule-based methods can only solve conflicts using temporal separation, by employing strategies like modification of velocity profiles and delay in the startup time of agents (Sanchez and Latombe, 2002; Li et al., 2005; Saha and Isto, 2006).

2.2. Distributed multi-path finding methods

Distributed multi-path finding methods treat each path to be planned as a separate computation entity and employ communication and negotiation mechanisms to resolve conflicts (Lumelsky and Harinarayan, 1997). Two prominent distributed multi-path finding algorithms are the ***communication-based*** approaches and the ***no-communication-based*** ones. In the ***communication-based*** approaches, all routes cooperate to reach a consensus on resolving conflicts (Desaraju and How, 2012; Ferrera et al., 2017; Olfati-Saber et al., 2007; Purwin et al., 2008; Scerri et al., 2007; Wang and Rubenstein, 2020). For example, a communication-based method is described to navigate multiple robots to a goal in an unknown physical environment (Gilboa et al., 2006). Each robot receives messages from others, decides whether and where to move, broadcasts the movement messages to others, receives approval messages, and physically performs the move action. Such methods are also applied to solve conflicts among multiple groups of agents (Ho et al., 2022), which solve a set of bargaining problems for each pair of groups and enable them to solve partial conflicts. The communication-based approaches can improve system-level costs by iteratively updating paths. However, further research is needed to bridge the gap between theoretical methods and implementation, e.g., how to efficiently generate updated plans by each

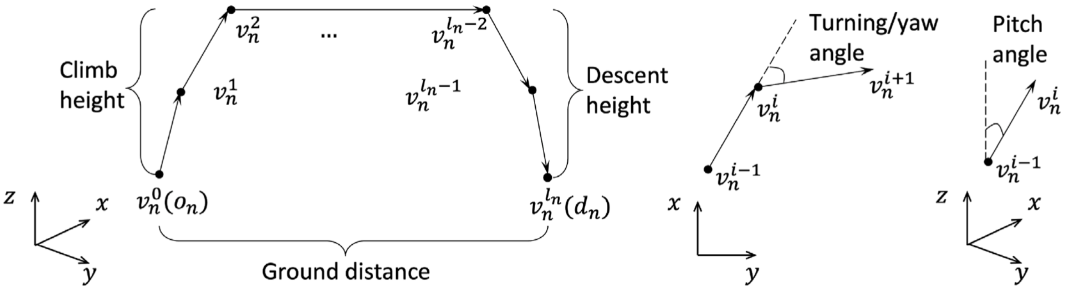


Fig. 4. Notations and components for a route $\boldsymbol{r}_n$.

path and how to design appropriate coordination strategies to optimize for a UAV route network. Therefore, in this paper, we follow this approach and propose a new method to coordinate the planning of individual paths considering system-level information.

The ***no-communication-based*** approaches do not require agents to be cooperative, here worst-case scenarios are considered for conflict avoidance. In these methods, agents are unaware of others' intentions. They observe the location and speed of other agents, predict their potential trajectories, and then plan their future trajectories accordingly (Aoude et al., 2010; Desaraju et al., 2009; Hoffmann and Tomlin, 2008). This approach is designed for navigating vehicles in an unknown environment without communication capability (or willingness) among vehicles, which is significantly different from the setting of the route planning problem in this paper.

2.3. Summary

In summary, in the context of drone delivery operations, these MAPF methods have primarily been directed towards task-oriented or in-flight path planning. This focus contrasts distinctly with the needs of infrastructure-oriented or pre-flight path planning, which are crucial for the systematic design of air route networks. We take inspiration from the work of Ho et al. (2022, 2019) who developed Pre-Flight Conflict Detection and Resolution (CDR) methods. These methods generate conflict-free paths for UAVs in both centralized and distributed settings before the actual flight occurs. However, their methods primarily focus on individual flight plans requiring both submissions and executions, without specifically addressing air route design for routine drone delivery operations. Our previous work (He et al., 2022) was one of the first attempts to address the aspect of infrastructure design. However, that work employs priority-based search algorithms — manually assigning priorities to individual routes, which compromises fairness among different routes.

3. Problem formulation

In this section, we first present the multi-path route network planning problem for drone delivery services in cities that we aim to address, then we describe our approach of employing soft constraints to reformulate this planning problem, offering a novel perspective on its resolution.

3.1. Definitions

Consider a set of OD pairs $\{(o_n, d_n), n \in [N]\}$ located in an urban environment, which is partitioned into grid cells ($\mathcal{G}$) for computation. We simplify the problem of planning a route for an OD pair by connecting departing and approaching fixes of vertiports as shown in Fig. 3b. A route $\boldsymbol{r}_n$ for the OD pair (o_n, d_n) is specified by a sequence of points along which drones can travel in a sequence from the origin o_n to the destination d_n, namely $\boldsymbol{r}_n = \left(v_n^0, v_n^1, \dots, v_n^{l_n}\right)$ where the route consists of l_n segments $\left\{\left(v_n^{i-1}, v_n^i\right), i \in [l_n]\right\}$ and $o_n = v_n^0, d_n = v_n^{l_n}$, as shown in Fig. 4. A set of routes connects departing fixes and approaching fixes, as shown in Fig. 5.

The following notations and functions in Table 1 will be used for this formulation:

3.2. Formulation for route network planning problem

The objective of the route planning problem for multiple OD pairs is two-fold. First, we expect to reduce the individual route cost, which is the operation cost for each route. The operation cost is related to energy consumption. Second, we expect to minimize the system-level cost, i.e., the impact on the risk, noise, privacy, and all other disruptions caused by drone operations, and the airspace covered by the planned route network. We formulate the conflict-free route network planning problem as follows:

$$\min_{\boldsymbol{r}_{[N]}} \sum_{n \in [N]} C_o(\boldsymbol{r}_n) + \alpha_a C_a(\boldsymbol{r}_{[N]}) + \alpha_i C_i(\boldsymbol{r}_{[N]}) \tag{1a}$$

$$\text{s.t.} \quad \lambda_y \le Y\left(v_n^{i-1}, v_n^i, v_n^{i+1}\right) \le \lambda_y, \tag{1b}$$

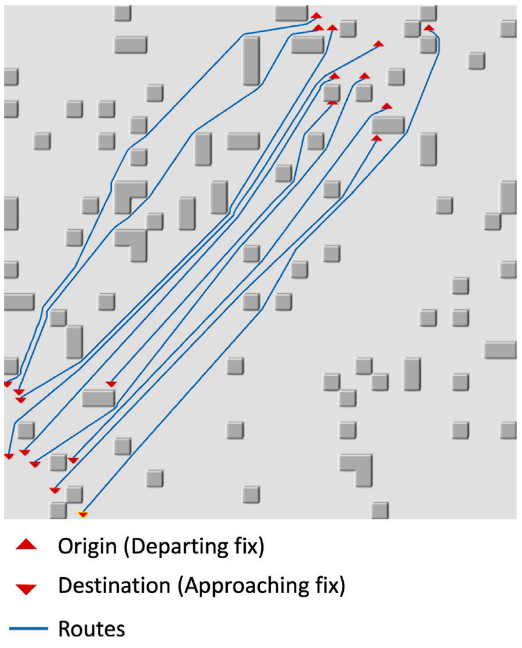


Fig. 5. An example of route network (top-down view).

$$\lambda_d \le P\left(v_n^{i-1}, v_n^i\right) \le \lambda_c, \tag{1c}$$

$$\mathcal{M}(\boldsymbol{r}_n) \cap \mathcal{B} = \emptyset, \tag{1d}$$

$$\mathcal{M}(\boldsymbol{r}_i) \cap \mathcal{M}(\boldsymbol{r}_j) = \emptyset, \forall i,j \in [N] \ \& \ i \ne j, \tag{1e}$$

Objectives in Eq. (1a) are to be minimized, where α_i and α_a are weighting factors to balance the operation cost, impact cost, and space cost.

- The first item C_o is an individual route cost, i.e., operation cost for each route. The operation cost is related to energy consumption. Specifically, the operation cost $C_o(\boldsymbol{r}_n)$ for each route $\boldsymbol{r}_n$, consists of four parts,

$$C_o(\boldsymbol{r}_n) = \sum_{i\in[l_n]} \alpha_g D_g(v_n^{i-1}, v_n^i) + \alpha_c D_c(v_n^{i-1}, v_n^i) + \alpha_d D_d(v_n^i, v_n^{i+1}) + \alpha_y \left| Y\left(v_n^{i-1}, v_n^i, v_n^{i+1}\right)\right|, \tag{2}$$

where $D_g(v_n^{i-1}, v_n^i)$ refers to the ground distance of the route segment (v_n^{i-1}, v_n^i), D_c and D_d refers to climb height and descent height. The last item on the right of the above formula relates to the turning angles of a route. $\alpha_g, \alpha_c, \alpha_d, \alpha_y$ are prespecified weighting factors. Detailed calculation of these distance items can refer to He et al. (2022).
- The second item C_a is the airspace cost. The occupied airspace is jointly determined by the routes for the N OD pairs, as clarified in He et al. (2022). Let $\boldsymbol{r}_{[N]} = \{\boldsymbol{r}_n, n \in [N]\}$, we write the airspace occupancy as a space cost function $C_a(\boldsymbol{r}_{[N]})$ for brevity. The calculation of C_a is

$$C_a(\boldsymbol{r}_{[N]}) = card\left(\mathcal{M}(\boldsymbol{r}_{[N]})\right) = card\left(\bigcup_{n\in[N]} \mathcal{M}(\boldsymbol{r}_n)\right), \tag{3}$$

which describes the total occupied airspace, or the grid cells, by the route network.
- The third item C_i is the impact cost. Impact cost $C_i(\boldsymbol{r}_{[N]})$ captures the impact on the risk, noise, privacy, and all other disruptions caused by drone operations; it usually depends on the airspace volume occupied by the routes $\mathcal{M}(\boldsymbol{r}_{[N]})$ and corresponding covered ground area. Every grid cell of the environment $a \in \mathcal{G}$ is associated with an impact cost $I(a)$. $C_i(\boldsymbol{r}_{[N]})$ can be calculated as

$$C_i(\boldsymbol{r}_{[N]}) = \sum_{a\in\bigcup_{n\in[N]}\mathcal{M}(\boldsymbol{r}_n)} I(a). \tag{4}$$

This work specifically focuses on risk as the impact cost, which can be decomposed into the sum of the risks for each route. Therefore, the third item can be simplified to the sum of the impact costs of each individual route. Specifically, the impact cost $C_i(\boldsymbol{r}_n)$ is calculated as

$$C_i(\boldsymbol{r}_{[N]}) = \sum_{n\in[N]} C_i(\boldsymbol{r}_n) = \sum_{n\in[N]}\sum_{a\in\mathcal{M}(\boldsymbol{r}_n)} I(a). \tag{5}$$

Table 1
Descriptions of notations and functions for the problem formulation.

Route network related	
$\mathcal{G}$	The set of all grid cells of the environment
$[N]$	The set $\{1, 2, \ldots, N\}$
$\mathcal{B}$	The airspace volume blocked by obstacles
I	Impact cost distribution over the airspace
$\{(o_n, d_n), n \in [N]\}$	N OD pairs
$\mathcal{R}_n$	Feasible route set for OD pair $\{(o_n, d_n)\}$ under flying capacity and obstacle constraints
$\boldsymbol{r}_{[N]} = \{\boldsymbol{r}_n, n \in [N]\}$	A set of routes for all OD pairs
$\boldsymbol{r}_{-n} = \{\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{n-1}, \boldsymbol{r}_{n+1}, \ldots, \boldsymbol{r}_N\}$	Routes in route set $\boldsymbol{r}_{[N]}$ except $\boldsymbol{r}_n$
$\mathcal{M}(\cdot)$	Occupied airspace volume function
$C_i(\cdot)$	Impact cost function
$C_a(\cdot)$	Space cost function
$C_t(\cdot)$	Congestion pricing function
$C_p(\cdot)$	Number of path occupations function
$C_b(\cdot)$	Number of buffer zone occupations function
$C_l(\cdot)$	Congestion level function
$E(\cdot)$	Gain function
Individual route-related	
$\mathcal{R}_n$	Feasible route set for OD pair $\{(o_n, d_n)\}$ under flying capacity and obstacle constraints
v_n^i	Waypoint for route $\boldsymbol{r}_n$
$\mathcal{S}$	Local search space
λ_y	Angle limit for turning
λ_c	Angle limit for climbing
λ_d	Angle limit for descending
α_g	Weighting factor for ground distance
α_c	Weighting factor for climb height distance
α_d	Weighting factor for descent height
α_y	Weighting factor for turning angle
α_i	Weighting factor for impact cost
α_a	Weighting factor for space cost
α_t	Weighting factor for congestion pricing
$Y(\cdot)$	Yaw angle function
$P(\cdot)$	Pitch angle function
$D_g(\cdot)$	Ground distance function
$D_c(\cdot)$	Climb height function
$D_d(\cdot)$	Descent height function
$C_o(\cdot)$	Operation cost function
$S(\cdot)$	Local search space function
Basic calculations	
$card(\cdot)$	Counting elements function

In Eq. (1a), we treat the N OD pairs equally. However, we can readily extend our formulation and results to the case where OD pairs have different priorities by adjusting the weighting of each individual route cost $C_o(\boldsymbol{r}_n)$.

Constraints in Eqs. (1b)–(1e) should be satisfied to generate a feasible route network $\boldsymbol{r}_{[N]}$, where Eqs. (1b)–(1d) are individual route constraints, Eq. (1e) are the constraints among routes.

Eqs. (1b) and (1c) are UAV flying capability constraints. The turning, climbing, and descending angles of each segment transition in a route should be within prescribed limits. The function Y specifies the yaw angle for a drone transiting every two successive segments, and P specifies the pitch angle for a drone transiting each segment; moreover, the virtual points v_n^{-1} and $v_n^{l_n+1}$ are set for take-off and landing points.

Eq. (1d) is the obstacle avoidance constraint. Let $\mathcal{B}$ be airspace volume blocked by obstacles. The route of the OD pair (o_n, d_n) should not intersect the blocked space. The function $\mathcal{M}$ in the equation calculates the airspace volume occupied by a route. Let $\mathcal{R}_n$ denote the set of all feasible routes for the OD pair (o_n, d_n) that satisfy the operation constraints (1b)–(1c) and the blocking constraint (1d), i.e.,

$$\mathcal{R}_n = \left\{ \boldsymbol{r}_n = \left(v_n^0, v_n^1, \ldots, v_n^{l_n}\right) \;\middle|\; o_n = v_n^0, d_n = v_n^{l_n}, \text{ and (1b)–(1d)} \right\}. \tag{6}$$

The feasible route set, or the feasible region $\mathcal{R}_n$ for one OD pair depends only on the drone flying capacity and the obstacle space and is not affected by the route sets of other OD pairs.

Eq. (1e) are the conflict-free constraints among multiple routes. The conflict-free requirement involves interactive route planning among all OD pairs and thus complicates the route planning problem. Mathematically, to ensure that the routes for the N OD pairs are spatially separated from each other, no two OD pairs have routes sharing the same airspace volume.

3.3. Congestion pricing-based formulation

The route planning problem (1) challenges us by the interaction of all OD pairs from two aspects, namely, the overall airspace utilization objective and the conflict-free multi-path requirement.

In subsequent sections, we will propose a unified framework to address these challenges, culminating in an innovative and efficient distributed route planning approach. Here, we detail the process of transforming route conflicts from hard constraints into soft constraints within this framework.

The transformation is inspired by the Lagrange multiplier method, a fundamental technique in mathematical optimization that is instrumental in identifying the local extrema of a function under equality constraints. It simplifies constrained problems by reformulating them into an unconstrained format, thereby enabling the application of derivative tests used in unconstrained optimization scenarios. Drawing inspiration from this method, we convert the rigid conflict-free constraints into more flexible congestion pricing constraints. Congestion pricing is the accumulated cost for traversing congested areas, where congestion in a specific area is determined by the number of routes in the area. Let $C_t(\boldsymbol{r}_n)$ denote the congestion pricing function. With this approach, we can reformulate Problem (1) as follows:

$$\min_{\boldsymbol{r}_{[N]}} \sum_{n\in[N]} C_o(\boldsymbol{r}_n) + \alpha_a C_a(\boldsymbol{r}_{[N]}) + \alpha_i C_i(\boldsymbol{r}_{[N]}) + \alpha_t \sum_{n\in[N]} C_t(\boldsymbol{r}_n) \tag{7a}$$

$$\text{s.t.} \quad \lambda_y \leq Y\left(v_n^{i-1}, v_n^{i}, v_n^{i+1}\right) \leq \lambda_y, \tag{7b}$$

$$\lambda_d \leq P\left(v_n^{i-1}, v_n^{i}\right) \leq \lambda_c, \tag{7c}$$

$$\mathcal{M}(\boldsymbol{r}_n) \cap \mathcal{B} = \emptyset \tag{7d}$$

In this way, the hard constraints only include individual route constraints. When $C_t(\boldsymbol{r}_n) = 0$, it indicates that route $\boldsymbol{r}_n$ is not charged for passing congested areas. If congestion pricing for all routes is zero, then no conflict exists in the route network. Detailed calculation of C_t will be elaborated in the subsequent sections.

4. Methodology for route network planning

4.1. Overview

The network planning problem in (1) is NP-hard (He et al., 2022). The optimal solution cannot be found in polynomial time. In this work, we propose a heuristic method to find a near-optimal solution rapidly in large real-world scenarios. The method is referred to as a Distributed Route Planning with Congestion pricing (DRP-CGSTN) method for drone delivery services in cities. This method takes a distributed planning approach, in which each origin–destination (OD) pair competes against other OD pairs for an optimized route (e.g. shortest distance) following system-level coordination, with the objectives that minimize the cost of the individual paths and the cost of the entire system. The core concept is congestion pricing, a soft constraint introduced to coordinate the allocation of airspace, contrasting to existing methods that consider conflicts as hard constraints. The rationale of congestion pricing is that when an area is passed by many paths if taking the shortest distance, this area is likely to have traffic conflicts or become congested; therefore, congestion pricing is imposed on paths that go through this area; as a result, these paths would move away from the congested area iteratively to reduce its own cost as well as system-level congestion reduction till no more system-level gain can be achieved.

The detailed steps of the proposed method are illustrated in Fig. 6. It comprises a pre-processing step and a network planning step. The pre-processing step generates grid cells for the graph search. The network planning step finds a feasible route network and is composed of five modules: initial path generation, congestion level evaluation, congestion reduction plans generation, plan evaluation and ranking, and path update. The *initial path generation* module generates the best single path for each route without consideration of congestion. The *congestion level evaluation* module evaluates the areas that are congested. The *congestion reduction plans generation* module runs distributedly. In this module, each route proposes a plan simultaneously, which is an updated path that reduces the occupied congested airspace. The *plan evaluation and ranking* module uses a gain function to evaluate and rank the congestion reduction plans, then the *path update* module updates the path in the best plan.

4.2. Pre-processing

This module generates a grid graph to encode the environment. The airspace of the environment is discretized into 3D cubic grid cells, $\mathcal{G}$, and a corresponding set of blocked airspace volumes/cells $\mathcal{B}$ and impact cost distribution I. The environment includes obstacles/terrains, vertiports, and impact distribution that determine the attributes of each cell. The vertiports determine whether a cell is an origin/destination vertiport or not, the obstacles/terrains determine if a cell is passable, and the impact distribution determines the impact on the risk, noise, privacy, and all other disruptions caused by UAV operations within each cell. More details are provided by He et al. (2022).

It should be noted that the grid discretization granularity can affect the optimality and efficiency of the generation of each single route, and thus, the whole route network. If grid cells are smaller, discretized optimality approaches the continuous optimality, but at the cost of more computational time because of a larger grid graph. As a result, the efficiency of generating conflict-free routes is also reduced.

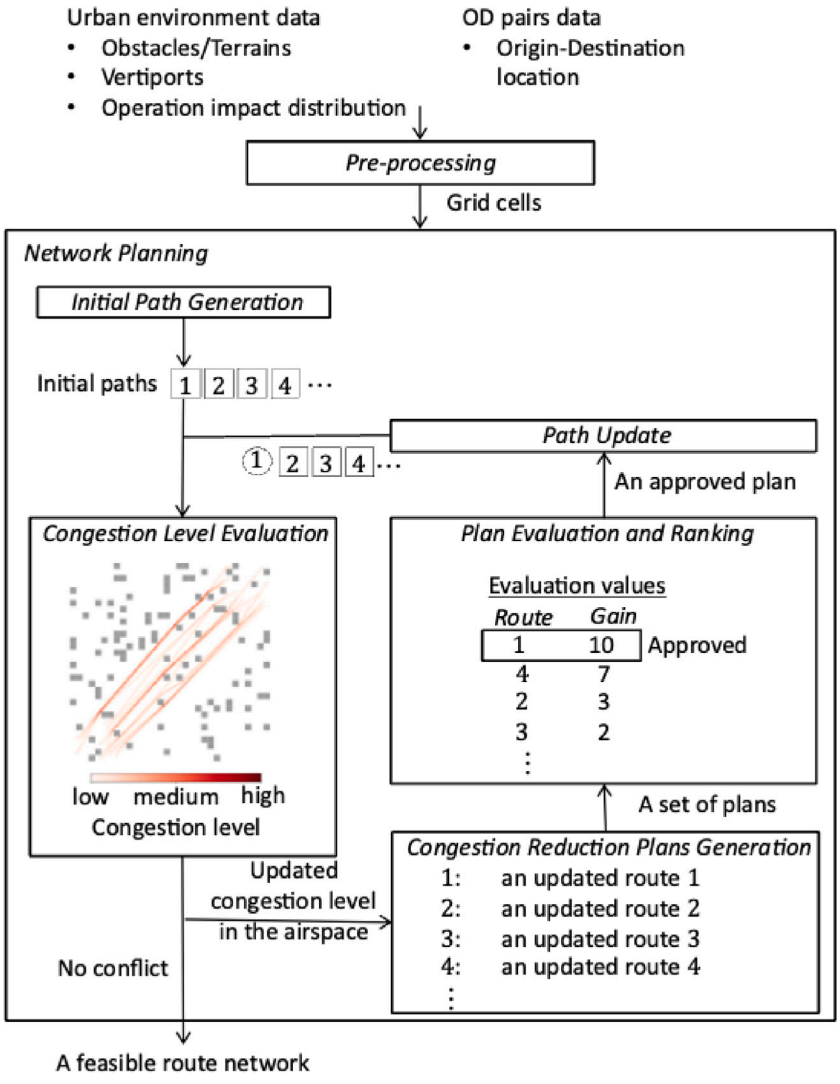


Fig. 6. Overall workflow of the proposed distributed route planning with congestion pricing method.

4.3. Network planning

4.3.1. Initial path generation

This module generates an initial route network as a warm start. It separately and distributedly generates a path for each route without consideration of conflicts among routes. Each path in the initial network is optimal if conflicts among routes are not considered. A single path-finding algorithm, Extended Theta* (He et al., 2022), is applied to generate every single path. It searches for a path from a node o_n to a node d_n that has the minimum value of the individual route cost $C_o(\boldsymbol{r}_n)$ as defined in Eq. (2). For Extended Theta*, the granularity of grid discretization can influence the generation of a single route, that is, finer grids leading to routes closer to continuous optimality but potentially reducing efficiency due to increased complexity — this aspect is separate from establishing conflict-free routes. Conflict-free routes, which concern the interaction and separation of multiple routes, remain unaffected by the granularity of the grid.

4.3.2. Congestion level evaluation

When multiple routes pass through an area, the area becomes congested and conflicted. This module evaluates the congestion level of the initial route network and each updated planning of the route network.

Congestion level evaluation is based on the number of routes in the grid cells. As illustrated in Fig. 3, each route is composed of a path and surrounding buffer zones. Thus congestion level of cells results from path occupation and buffer zone occupation. Because buffer zones can be shared by different routes, a cell has no congestion if it is occupied only by buffer zones and it has

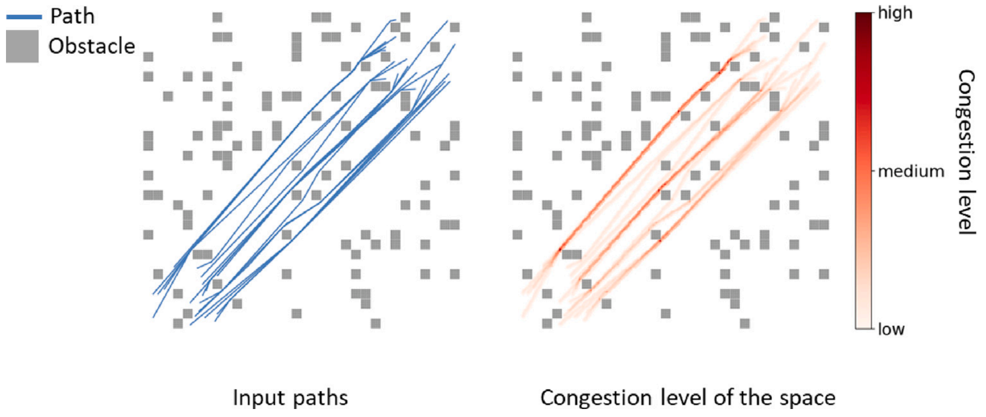


Fig. 7. Illustration for evaluating the congestion level of the airspace for a route network.

congestion only if it is occupied by paths of routes. Let $C_p(\cdot)$ and $C_b(\cdot)$ represent the number of path occupations and buffer zone occupations respectively. We can get $C_p(g)$ and $C_b(g)$ for each cell $g \in \mathcal{G}$. Let $C_l(\cdot)$ represent the congestion level, it is calculated as

$$C_l(g) = 4 * C_p(g) * (C_p(g) - 1)/2 + C_p(g) \ * \ C_b(g)$$

where $C_p(g) * (C_p(g) - 1)/2$ is the number of path-to-path conflicts, $C_p(g) * C_b(g)$ is the number of path-to-buffer conflicts, 4 is the coefficient for path-to-path conflicts because the path-to-path conflict makes airspace relatively more congested. For brevity, we can also define the congestion level of one route and a set of routes, the congestion level of a route $\boldsymbol{r}_n$ is

$$C_l(\boldsymbol{r}_n) = \sum_{g \in \mathcal{M}(\boldsymbol{r}_n)} C_l(g),$$

and the congestion level of a set of routes $\boldsymbol{r}_{[N]}$ is

$$C_l(\boldsymbol{r}_{[N]}) = \sum_{g \in \cup_{\boldsymbol{r}_n \in \boldsymbol{r}_{[N]}} \mathcal{M}(\boldsymbol{r}_n)} C_l(g),$$

and if there are only two routes $\boldsymbol{r}_i$ and $\boldsymbol{r}_j$, then

$$C_l(\boldsymbol{r}_i) = C_l(\boldsymbol{r}_j) = C_l(\{\boldsymbol{r}_i, \boldsymbol{r}_j\})$$

An illustration of the congestion level evaluation process is shown in Fig. 7.

4.3.3. *Congestion reduction plan generation*

Based on the evaluated congestion levels, routes can reduce the passage through congested areas to reduce conflicts. Therefore, this module allows each route to propose a congestion reduction plan. The core concept in this stage is congestion pricing, a soft constraint to coordinate the allocation of airspace instead of a hard constraint to resolve conflicts. The pseudocode of this algorithm is shown in Algorithm 1 in Appendix A, and the basic idea is illustrated here.

In kth iteration, this module generates a set of updated paths $\boldsymbol{r}^k_{[N]u}$ as the plans based on the original paths $\boldsymbol{r}^k_{[N]o}$. For each route, the proposed congestion reduction plan is an updated path that moves away from congested areas. Let $\boldsymbol{r}_{no}$ and $\boldsymbol{r}_{nu}$ separately represent the original path and the updated path of route $\boldsymbol{r}_n$ in the iterations, $\boldsymbol{r}_{-no}$ represent the set of routes $\boldsymbol{r}_{-no} = \boldsymbol{r}_{[N]o} \backslash \{\boldsymbol{r}_{no}\}$. The plan uses the following objective function to search for an updated path

$$f(\boldsymbol{r}_{nu}) = C_o(\boldsymbol{r}_{nu}) + \alpha_i C_i(\boldsymbol{r}_{nu}) + \alpha_a C_a(\boldsymbol{r}_{nu}|\boldsymbol{r}_{-no}) + \alpha_t C_t(\boldsymbol{r}_{nu}|\boldsymbol{r}_{-no}). \tag{8}$$

Here $C_o(\boldsymbol{r}_{nu})$ represents the operation cost in Eq. (2) of updated paths, $C_i(\boldsymbol{r}_{nu})$ represents the impact cost, $C_a(\boldsymbol{r}_{nu}|\boldsymbol{r}_{-no})$ represents the space cost of $\boldsymbol{r}_{nu}$ when other routes take $\boldsymbol{r}_{-no}$, α_t is the congestion pricing coefficient and $C_t(\boldsymbol{r}_{nu}|\boldsymbol{r}_{-no})$ is the congestion pricing of $\boldsymbol{r}_{nu}$ when other routes take $\boldsymbol{r}_{-no}$. The congestion pricing does not forbid routes to pass through congested areas; instead, the routes crossing these areas are penalized. For a drone delivery operator, the global cost is associated with the individual route cost (operation cost C_o) and system-level cost (impact cost C_i, space cost C_a and the congestion pricing C_t). For a route $\boldsymbol{r}_n$ from o_n to d_n, the congestion pricing is

$$C_t(\boldsymbol{r}_n|\boldsymbol{r}_{-n}) = C_l(\boldsymbol{r}_{[N]}) - C_l(\boldsymbol{r}_{-n}), \tag{9}$$

it is calculated as the difference between the congestion level of the route set $C_l(\boldsymbol{r}_{[N]})$ and the congestion level of the route set $C_l(\boldsymbol{r}_{-n})$. An illustration of congestion pricing is presented in Fig. 8.

The coefficient α_t reflects the importance of congestion pricing in the total costs. If α_t is small, for example, $\alpha_t \to 0$, the route can pass congested areas with negligible cost addition. If α_t is large, e.g., $\alpha_t \to \infty$, the resulting cost will be high, preventing the route from passing through congested areas.

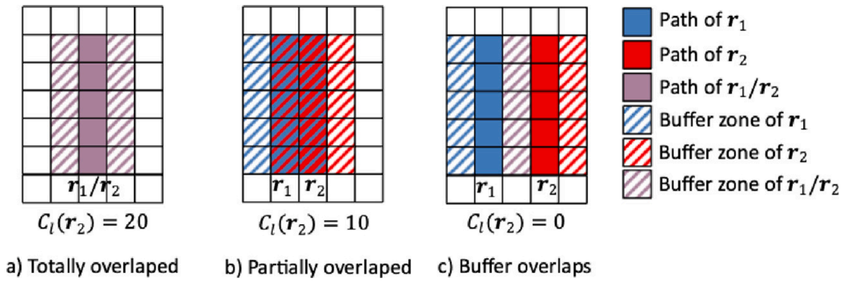


Fig. 8. Example of the congestion pricing for different plans.

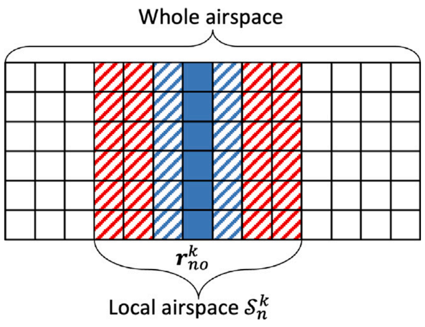


Fig. 9. Illustration for searching in a local airspace with $L_k = 3$.

To speed up generating the plans for real-world scenarios, the method searches for a path in a local airspace instead of searching in the global airspace, as shown in Fig. 9. Searching in a local space extends fewer cells and finds a near-optimal solution in less computational time. For an original path $\boldsymbol{r}_{no}^k$ in kth iteration, the search space is defined as

$$S_n^k = S(\boldsymbol{r}_{no}^k, L_k)$$

where L_k is the size of the local airspace.

4.3.4. Plan evaluation and ranking

This module evaluates and ranks congestion reduction plans proposed by the routes based on the reduction of airspace congestion and increase in path length. A plan that reduces airspace congestion by adding the least path length is more likely to be selected and executed subsequently. Elaboration of this module yields the following series of Steps.

Step 1. Plan evaluation for each route

A gain function is applied to plan evaluation. It describes how much airspace congestion is reduced and how much other cost items increase. It compares the original path $\boldsymbol{r}_{no}^k$ and the updated path $\boldsymbol{r}_{nu}^k$ in the plan.

$$E(\boldsymbol{r}_{no}^k, \boldsymbol{r}_{nu}^k) = \Delta C_t(\boldsymbol{r}_{no}^k, \boldsymbol{r}_{nu}^k) - \Delta C_o(\boldsymbol{r}_{no}^k, \boldsymbol{r}_{nu}^k) - \Delta C_a(\boldsymbol{r}_{no}^k, \boldsymbol{r}_{nu}^k) - \Delta C_i(\boldsymbol{r}_{no}^k, \boldsymbol{r}_{nu}^k) \tag{10}$$

where

$$\Delta C_t(\boldsymbol{r}_{no}^k, \boldsymbol{r}_{nu}^k) = C_l(\{\boldsymbol{r}_{no}^k \cup \boldsymbol{r}_{-no}^k\}) - C_l(\{\boldsymbol{r}_{nu}^k \cup \boldsymbol{r}_{-no}^k\}),$$

$$\Delta C_o(\boldsymbol{r}_{no}^k, \boldsymbol{r}_{nu}^k) = C_o(\boldsymbol{r}_{no}^k) - C_o(\boldsymbol{r}_{nu}^k),$$

$$\Delta C_a(\boldsymbol{r}_{no}^k, \boldsymbol{r}_{nu}^k) = C_a(\boldsymbol{r}_{no}^k | \boldsymbol{r}_{-no}^k) - C_a(\boldsymbol{r}_{nu}^k | \boldsymbol{r}_{-no}^k),$$

$$\Delta C_i(\boldsymbol{r}_{no}^k, \boldsymbol{r}_{nu}^k) = C_i(\boldsymbol{r}_{no}^k) - C_i(\boldsymbol{r}_{nu}^k).$$

ΔC_o, ΔC_a, and ΔC_i are the additional operation cost, space cost, and impact cost of the plan, and ΔC_t is the reduced airspace congestion of the plan. The gain function will be larger if it reduces more congestion with a smaller increased operation, space, and impact cost.

Step 2. Plan ranking

The ranking sorts plans from the largest gain function to the smallest. It uses centralized coordination - a roulette-wheel selection method to select the best plan. A plan with a larger gain value is more likely to be selected and executed afterward. It is done in substeps 2.1–2.3:

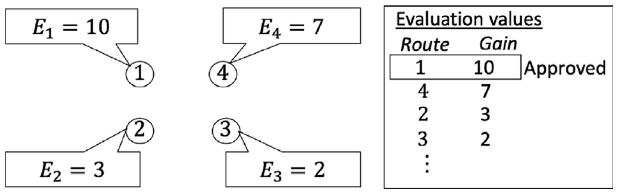


Fig. 10. Example for the evaluation and ranking process.

Substep 2.1 Determine the routes with improvements

Let I_k be the set of plans in which the updated path differs from the original path.

$$I_k = \{n \in [N] : \Delta\, C_t(\boldsymbol{r}^k_{no}, \boldsymbol{r}^k_{nu}) \neq 0 \ \text{ or } \ \Delta\, C_o(\boldsymbol{r}^k_{no}, \boldsymbol{r}^k_{nu}) \neq 0 \ \text{ or } \ \Delta\, C_a(\boldsymbol{r}^k_{no}, \boldsymbol{r}^k_{nu}) \neq 0 \ \text{ or } \ \Delta\, C_i(\boldsymbol{r}^k_{no}, \boldsymbol{r}^k_{nu}) \neq 0\}.$$

Substep 2.2 Check route feasibility and congestion pricing

If I_k is an empty set, i.e., no routes can be improved, it can be two cases. If congestion pricing equals zero, then all routes are conflict-free, thus the feasible route network can be outputted. Otherwise, the local airspace is not sufficiently large to contain a path with fewer conflicts, or the congestion pricing is not sufficiently large to move the route away from congested areas. In these cases, no plan will be selected but the congestion pricing coefficient and the neighbor space size will enlarge.

$$\alpha_t += \Delta\alpha_t$$

$$L_k += \Delta L_k;$$

Substep 2.3 Select the best plan

A roulette-wheel selection method to select the best plan. It first calculates the probability of being selected based on the softmax of the gain function:

$$P^k_n = \frac{|E^k_n|}{\sum_{n\in I_k} |E^k_n|}, \quad \forall n \in I_k.$$

Then, a random number $seed \sim Uniform(0,1)$ is generated, and the plan of route $\boldsymbol{r}_{n'}$ is selected based on the roulette wheel selection rule.

$$n' = \underset{n}{\arg\text{where}}\ seed \in \left[\sum_1^{n-1} P^k_n, \sum_1^{n} P^k_n\right].$$

An illustration of the evaluation and ranking is shown in Fig. 10.

4.3.5. Path update

This module updates the routes based on the ranking results. The route in the approved plan will update its path, whereas the path of other routes will not be changed in this iteration.

$$\boldsymbol{r}^{k+1}_{no} = \begin{cases} \boldsymbol{r}^k_{nu}, & n = n' \\ \boldsymbol{r}^k_{no}, & n \neq n' \end{cases} \tag{11}$$

After the path updating step, the method will go to the congestion level evaluation module again, followed by plan generation, plan evaluation and ranking, and path update. The iterations will continue till no conflict exists, i.e., congestion pricing equals zero for all routes.

5. Theoretical analysis

The proposed DRP-CGSTN method uses a congestion pricing to transform the conflict constraints in (1e) into soft constraints. Let λ_{ij} be the congestion pricing coefficient for $\boldsymbol{r}_i$ passing a congested airspace caused by $\boldsymbol{r}_j$, the objective function is now

$$\boldsymbol{r}_{[N]} = \underset{\boldsymbol{r}_{[N]}}{\arg\min}\, f(\boldsymbol{r}_{[N]}|\lambda) = \underset{\boldsymbol{r}_{[N]}}{\arg\min} \sum_{i\in[N]} C_o(\boldsymbol{r}_i) + \alpha_a C_a(\boldsymbol{r}_{[N]}) + \alpha_i C_i(\boldsymbol{r}_{[N]}) + \sum_{i,j\in[N], i\neq j} \lambda_{ij}\ C_l(\{\boldsymbol{r}_i, \boldsymbol{r}_j\}) \tag{12}$$

Regarding a sufficiently large λ_{ij}, the problem in Eq. (12) equals the problem in Eq. (1). If a method can find an optimal solution for Eq. (12), it can find an optimal solution for Eq. (1). DRP-CGSTN solves Problem Eq. (12) under an increasing sequence of congestion pricing coefficients. We can prove the convergence of the objective value in Eq. (12) regarding a sufficiently large λ by arguing that it is lower bounded by the optimal cost when $\lambda_{ij} = 0$, and it is non-increasing over the sequence of obtained solutions. However, it remains to show that the solution sequence will converge and that when the algorithm converges there is no conflict among routes.

Table 2
Total costs of each route by DRP-CGSTN and priority-base planning in the toy example.

	Individual optimal paths	Paths from priority-based planning	Paths from DRP-CGSTN
Cost[a] for $\boldsymbol{r}_1$	157.18	284.93	215.39
Cost for $\boldsymbol{r}_2$	147.08	147.08	206.65
Extra sum of cost for solving conflicts	–	+127.75	+117.78

[a] Cost here is the operation cost and impact cost.

Here we prove that the DRP-CGSTN method follows the optimality proposition when the space cost can be calculated based on individual routes, not allowing the buffer zones to be shared among different routes. In this case, $C_a(\boldsymbol{r}_{[N]}) = \sum_{i\in[N]} C_a(\boldsymbol{r}_i)$ and $C_i(\boldsymbol{r}_{[N]}) = \sum_{i\in[N]} C_i(\boldsymbol{r}_i)$, and the objective function Eq. (12) is now Eq. (13). The proof of Proposition 1 is presented in Appendix B.

$$\boldsymbol{r}_{[N]} = \arg\min_{\boldsymbol{r}_{[N]}} f(\boldsymbol{r}_{[N]}|\lambda) = \arg\min_{\boldsymbol{r}_{[N]}} \sum_{i\in[N]} C_o(\boldsymbol{r}_i) + \alpha_a \sum_{i\in[N]} C_a(\boldsymbol{r}_i) + \alpha_i \sum_{i\in[N]} C_i(\boldsymbol{r}_i) + \sum_{i,j\in[N],i\neq j} \lambda_{ij}\ C_l(\{\boldsymbol{r}_i,\boldsymbol{r}_j\}) \tag{13}$$

Proposition 1. *The proposed DRP-CGSTN method is able to find the optimal solution for Eq. (13) if the following assumptions are satisfied:*

A. 1 For brevity, let $\mathcal{N}(i,j) = C_l(\{\boldsymbol{r}_i,\boldsymbol{r}_j\})$ and $C(\boldsymbol{r}_n) = C_o(\boldsymbol{r}_n) + \alpha_a C_a(\boldsymbol{r}_n) + \alpha_i C_i(\boldsymbol{r}_n)$. In the congestion reduction plans generation module, each route can find the optimal solution, that is

$$\boldsymbol{r}_n = \underset{\boldsymbol{r}_n}{\text{argmin}}\ f(\boldsymbol{r}_n|(\lambda,\boldsymbol{r}_{-n})) = \underset{\boldsymbol{r}_n}{\text{argmin}}\ C(\boldsymbol{r}_n) + \sum_{l\in[N_{-n}]} \lambda_{nl}\mathcal{N}(n,l)$$

A. 2 For brevity, let $\lambda^k = (\lambda_1^k,\ldots,\lambda_{n-1}^k,\lambda_n^k,\lambda_{n+1}^k,\ldots,\lambda_N^k)$, and $\boldsymbol{r}_{-n}|\lambda^k = \arg\min_{\boldsymbol{r}_{-n}} f(\boldsymbol{r}_{[N]}|\lambda^k)$. In each iteration the congestion pricing coefficient λ is updated from λ^k to λ^{k+1} while satisfying

$$\min_{\boldsymbol{r}_n} f(\boldsymbol{r}_n|(\lambda^{k+1},\boldsymbol{r}_{-n}|\lambda^k)) - \min_{\boldsymbol{r}_n} f(\boldsymbol{r}_n|(\lambda^k,\boldsymbol{r}_{-n}|\lambda^k)) \leq \min_{\boldsymbol{r}_n} f(\boldsymbol{r}_n|(\lambda^{k+1},\boldsymbol{r}_{-n}|\lambda^{k+1})) - \min_{\boldsymbol{r}_n} f(\boldsymbol{r}_n|(\lambda^k,\boldsymbol{r}_{-n}|\lambda^{k+1})) \tag{14}$$

Based on Proposition 1, the method finds the optimal solution for Eq. (13), which is also the optimal solution for (1) when the sharing of buffer zones among different routes is not allowed. However, in the application of the proposed method in real-world scenarios, these assumptions are not always satisfied. For example, if the single path planning in congestion reduction plan generation cannot generate an optimal solution, then Assumption 1 does not hold. Also, when the updating process of the congestion pricing coefficient does not strictly satisfy Eq. (14), Assumption 2 does not stand. In summary, the proposed DRP-CGSTN is a heuristic method to solve the network planning problem as described in Eq. (1). Further theoretical analysis is needed to show the convergence and optimality of the sequence of obtained solutions over increasing λ.

6. Testing in benchmark scenarios and an real-world urban scenario

In this section, we first used an illustrative scenario to demonstrate the working of the algorithm. We then compared the proposed algorithm with other path-finding algorithms in a 2D test scenario. Subsequently, the algorithm was applied to a 3D real-world scenario in Mong Kok, HKSAR.

6.1. Illustration with a toy example

We use a simple scenario to illustrate the operation of the algorithm, as shown in Fig. 11. In this example, there is a low-impact area in the middle of the two obstacles, and the impact of passing through the low-impact areas is 1/3 that of passing through the normal areas. The objective is to find two routes, 1 and 2 to connect vertices on the left and on the right. We applied the *DRP-CGSTN* and the priority-based planning methods (He et al., 2022) to this toy example. In the priority-based planning, two routes are planned sequentially, and the sequence order is determined based on the total cost in Eq. (2).

The generated paths are shown in Fig. 11. If conflicts amongst the routes are not considered, then both follow the individual optimal paths, which pass through the middle low-impact area. For priority-based planning, $\boldsymbol{r}_2$ is planned first and $\boldsymbol{r}_1$ should resolve conflicts because this order leads to a lower cost, however, the high-priority route $\boldsymbol{r}_2$ will pass through the low-impact cost area, and the other route $\boldsymbol{r}_1$ has to fly near the upper margin. For *DRP-CGSTN*, none pass the low-impact area, but both routes cross the middle corridor. The total cost of the generated paths is shown in Table 2. As shown in the table, *DRP-CGSTN* achieves a lower cost than priority-based planning. This is because, for individual optimal paths, priority-based planning only considers $\boldsymbol{r}_1$ to solve conflicts, whereas *DRP-CGSTN* considers both routes.

6.2. Comparative tests on a standard 2D scenario

In this section, we compared our method with three prominent methods: priority-based planning (He et al., 2022), Push-and-Spin (Alotaibi and Al-Rawi, 2018), and conflict-based search (CBS) (Sharon et al., 2015). The experimental setup is shown in Fig. 12(a). This is a 2D scenario (Stern et al., 2019) of grid size 256 ∗ 256 covering an area of around 2.5 km by 2.5 km. This

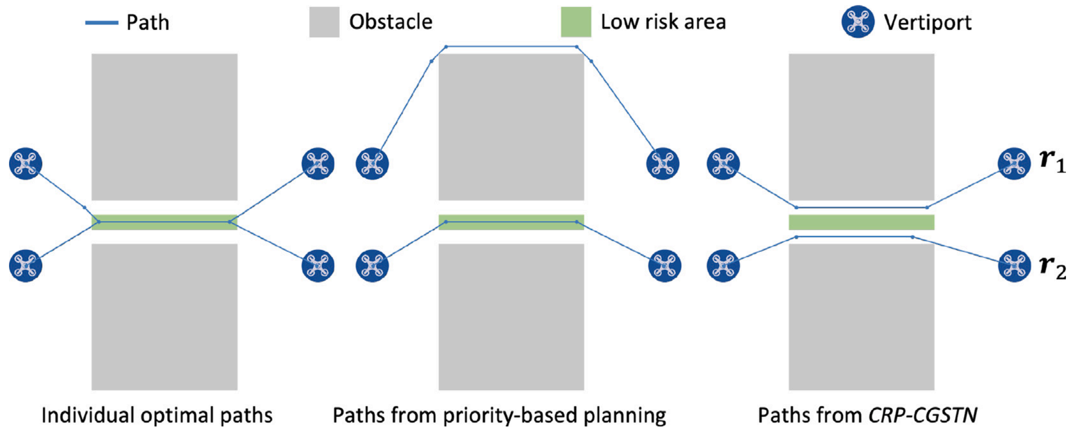


Fig. 11. Test case (top view) for priority-based planning and DRP-CGSTN.

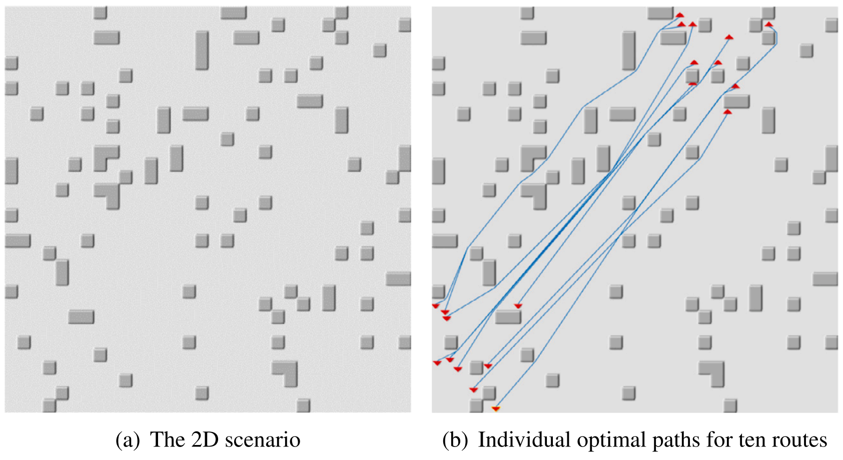


Fig. 12. The 2D scenario and the required routes.

grid comprises 65,536 cells, of which 6528 cells are inaccessible. Multiple routes were generated from the lower left corner to the upper right corner, with origins and destinations positioned in proximity to each other and near urban obstacles (residential or office buildings), to emulate a realistic drone delivery environment. All experiments were conducted using an Ubuntu server with parameters 32x Intel (R) Xeon (R) Gold 5218 CPUs and 13 GB memory. The individual optimal paths of these routes are shown in Fig. 12(b). Due to the clustered nature of these paths and the proximity of origins and destinations to obstacles and other pairs, finding networks of spatially separated paths was challenging.

Notably, both Push-and-Spin and CBS are primarily directed toward task-oriented or in-flight path planning. They were initially developed to resolve spatiotemporal conflicts among agents. In this scenario, however, we have adapted them to address spatial conflicts. A focus exclusively on spatial conflicts can lead to an increased frequency of conflicts, which may slow down the resolution process for these methods towards this problem. Furthermore, CBS does not support buffer zones overlapping to reduce space costs. Priority-based planning can accommodate all the objectives and constraints in Eq. (1a). As the order of the OD pairs (priority) affects the solution, we run priority-based planning in two strategies. In the first method, the route sequence was the same as the order of the newly added OD pairs, and in the second method, all possible route sequences were considered but the one with the minimum cost was selected as the solution. Here in the experiment, we take the distance as the cost.

The cost and computational time for priority-based planning, conflict-based search, push-and-spin and *DRP-CGSTN* method are listed in Table 3, and the generated routes are shown in Fig. 13. It is important to note that priority-based planning can output all feasible routes it identifies, though some routes may fail to be generated. For priority-based planning that runs only one sequence, the computational time was the smallest, but the cost was the largest compared with other methods. In addition, it can easily fail to find all paths as high-priority routes can block all possible paths for low-priority routes, especially OD pairs are close to others in this scenario. When all possible sequences were run, the solution had a smaller cost and more routes were found because the algorithm used the most appropriate priority (order), but the computational time increased very quickly and became unacceptably large. Push-and-spin runs relatively fast, but similar to priority-based planning with one sequence, it can quickly fail to find feasible

Table 3
Cost and computational time for the methods in the 2D scenario.

# of routes	1	2	3	4	5	6	7	8	9	10
Individual optimal routes										
Distance	275	519	773	1015	1192	1426	1662	1907	2162	2468
Priority-based planning (one sequence)										
Distance	275	523	785	INF[a]	INF[a]	INF[a]	INF[a]	INF[a]	INF[a]	INF[a]
Time (s)	0.009	0.016	0.026	–	–	–	–	–	–	–
Priority-based planning (all sequences)										
Distance	275	520	782	1029	1206	1441	1677	1925	2212	2613
Time (s)	0.009	0.026	0.13	0.75	5.3	39	280	2500	230[b]	330[b]
Push-and-Spin										
Distance	275	520	782	1029	1206	INF[a]	INF[a]	INF[a]	INF[a]	INF[a]
Time (s)	0.009	0.02	0.07	0.20	0.75	–	–	–	–	–
Conflict-based search										
Distance	275	520	775	–	–	–	–			
Time (s)	0.012	0.24	330	>3600	>3600	>3600	>3600	>3600	>3600	>3600
DRP-CGSTN										
Distance	275	520	775	1024	1202	1437	1673	1914	2167	2472
Time (s)	0.025	0.16	0.25	1.3	1.3	1.6	1.7	2.3	2.6	3.1

[a] Some routes cannot be found.

[b] The number of sequences is too large, only sample 3000 sequences here.

routes. CBS generated a route network with a small cost; however, it ran very slowly, particularly when there were many conflicts among individual optimal paths.

DRP-CGSTN found spatially separated paths within an acceptable time, and the solution had a small cost. In addition, the method treats conflicts as soft constraints and updates paths only in the neighboring airspace in each iteration. The generated routes were more aligned and still spatially separated, which improved airspace utilization. The discrepancy between DRP-CGSTN and individual optimal routes does not consistently increase with the number of routes, due to the variable adherence of the congestion pricing coefficient to Proposition 1.

6.3. Demonstration in a real-world urban scenario

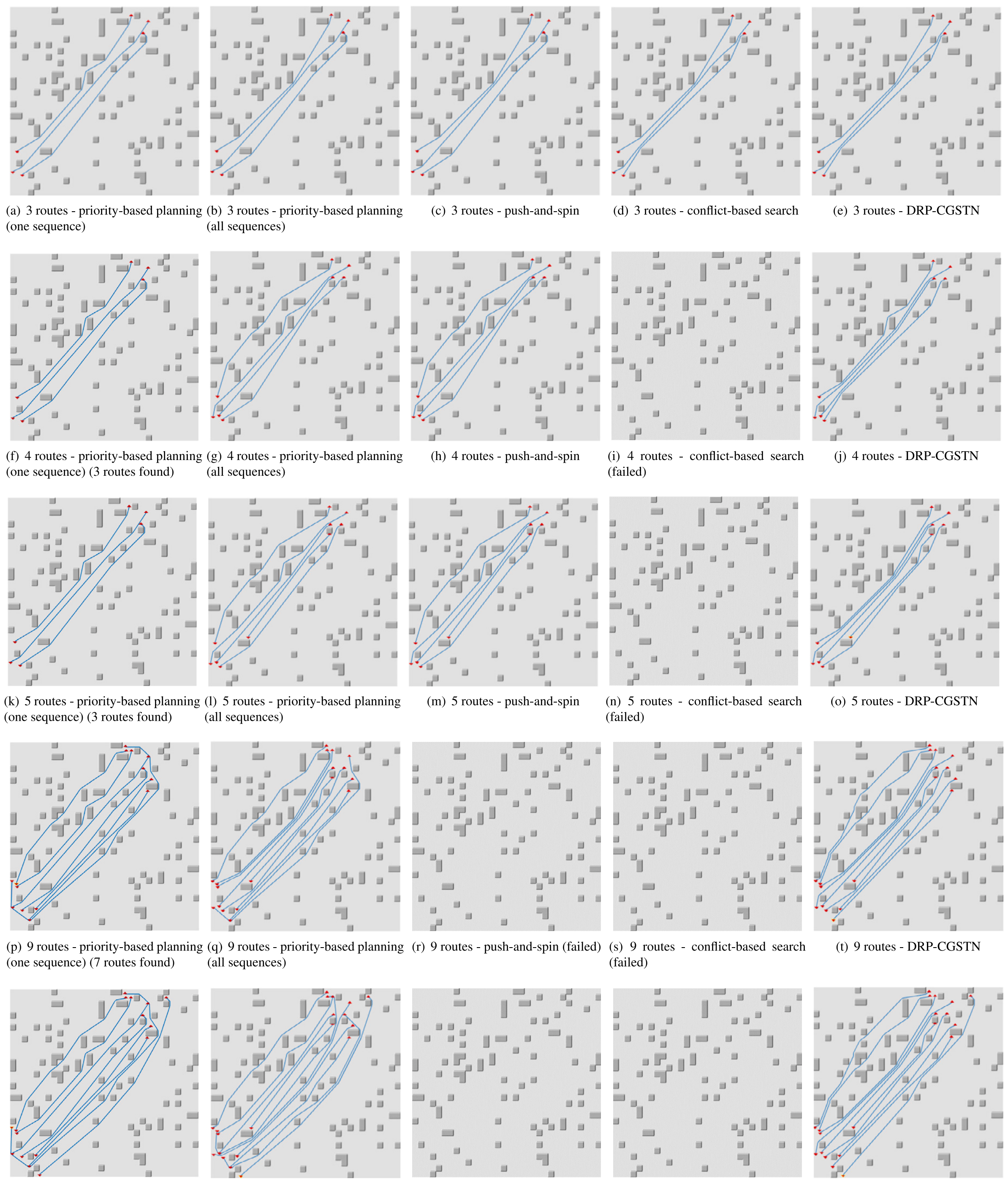
In this section, we present the set of routes generated by the proposed DRP-CGSTN method in a real-world scenario, an urban area in Mong Kok, HKSAR, with a physical size of 1840 m × 1900 m. Hong Kong has a population of more than 7.4 million that live in an area of just 1104 square kilometers, making it one of the most densely populated places in the world (Hong Kong Census and Statistics Department, 2023). Mong Kok is even denser with its extremely high population density of 130,000/km^2, and it was described as the busiest district in the world. The airspace discretization in the preprocessing step discretizes the airspace into 920 $*$ 949 $*$ 50 grid cells. The scenario is a typical urban area with tightly arranged buildings, and the time for finding one route is of the order of seconds. In the experiment, sixteen routes were generated at heights from 90 to 120 m. Under parameters $\alpha_i = 1, \alpha_a = 0.1, \Delta\alpha_t = 0.2, \Delta L_k = 10$, the distributed planning method takes 206.4 s to find a feasible solution. Two cases were considered in this study. The first case did not consider low-impact areas, whereas the second case considered them. The solutions for both cases are shown in Fig. 14, where the generated routes satisfy the separation requirements. Compared to straighter routes in Fig. 14(b), the routes in 14(c) detour over low-impact areas.

6.4. Sensitivity analysis on algorithm parameters

In this section, we analyze the impact of various algorithm parameters on overall performance. The parameters include impact cost, space cost coefficient, coefficients in the calculation of operation cost, congestion pricing coefficient, and local airspace size.

The relative value of the weight coefficient of space cost α_a and impact cost α_i in relation to other cost coefficients needs to be carefully calibrated to reflect the actual operational cost impact of airspace usage charges and impact of drone operations to people and facilities on the ground. The values can be determined by the relative monetary cost of the different categories. The weight coefficient of ground distance, climb height distance, descent height, and turning angle are determined by the energy consumption of drone operations. A higher coefficient for an item prompts the algorithm to prioritize its optimization. For a detailed analysis of these parameters, refer to He et al. (2022). For example, as the space cost coefficient increases, the number of buffer zone cells and the number of total occupied cells decrease; meanwhile, the number of path cells increases slightly, and the total impact cost increases gradually as the relative weight on risks reduces compared to weight on airspace occupancy, resulting some routes fly over high-risk areas to reduce space cost. Therefore, the relative value of the space cost coefficient in relation to other cost coefficients should be carefully calibrated so that the safety aspect is not compromised.

In this study, we focus on the two distinct parameters in our algorithm in Section 4: congestion pricing weight coefficient α_t and local airspace size L_k. We take the scenario in Section 6.3 for the testing and fix other parameters $\alpha_a = 0.1, \alpha_i = 1, \Delta\alpha_t = 0.2, \Delta L_k = 5$. We take congestion pricing weight coefficient $\Delta\alpha_t \in [0.2, 0.4, 0.6, 0.8, 1.0]$ and $\Delta L_k \in [5, 10, 15, 20, \infty]$. Here, ∞ means global search.



(a) 3 routes - priority-based planning (one sequence) (b) 3 routes - priority-based planning (all sequences) (c) 3 routes - push-and-spin (d) 3 routes - conflict-based search (e) 3 routes - DRP-CGSTN

(f) 4 routes - priority-based planning (one sequence) (3 routes found) (g) 4 routes - priority-based planning (all sequences) (h) 4 routes - push-and-spin (i) 4 routes - conflict-based search (failed) (j) 4 routes - DRP-CGSTN

(k) 5 routes - priority-based planning (one sequence) (3 routes found) (l) 5 routes - priority-based planning (all sequences) (m) 5 routes - push-and-spin (n) 5 routes - conflict-based search (failed) (o) 5 routes - DRP-CGSTN

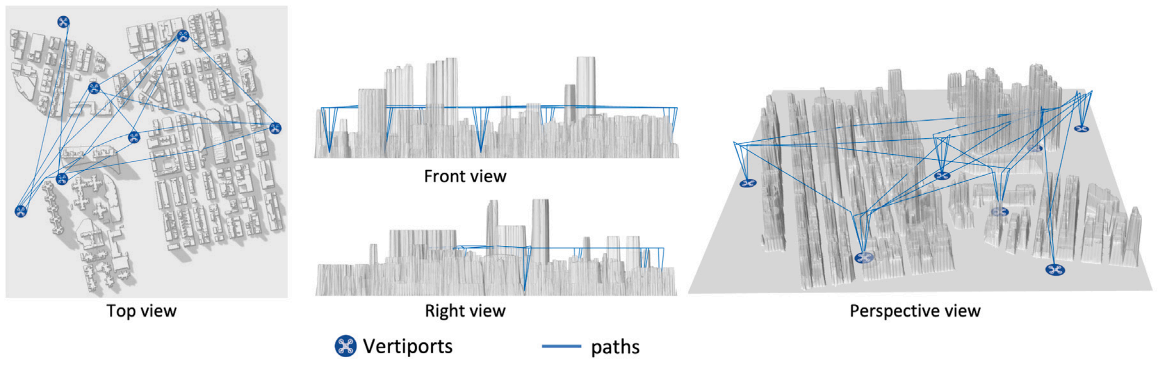
(p) 9 routes - priority-based planning (one sequence) (7 routes found) (q) 9 routes - priority-based planning (all sequences) (r) 9 routes - push-and-spin (failed) (s) 9 routes - conflict-based search (failed) (t) 9 routes - DRP-CGSTN

(u) 10 routes - priority-based planning (one sequence) (8 routes found) (v) 10 routes - priority-based planning (all sequences) (w) 10 routes - push-and-spin (failed) (x) 10 routes - conflict-based search (failed) (y) 10 routes - DRP-CGSTN

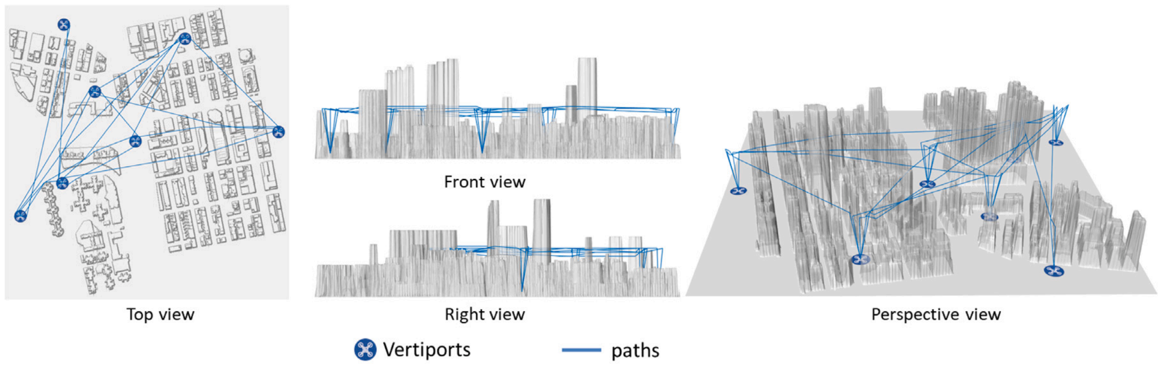
Fig. 13. Routes planned by priority-based planning, push-and-spin, conflict-based search, and DRP-CGSTN in the 2D scenario (256 $*$ 256 grid cells).

The results are shown in Table 4. The results indicate a trade-off: as the congestion pricing coefficient is increased, there is a marginal reduction in computational time, yet this leads to an increase in total costs. Conversely, expanding the local search space increases the computational time but results in a slight reduction in total costs.

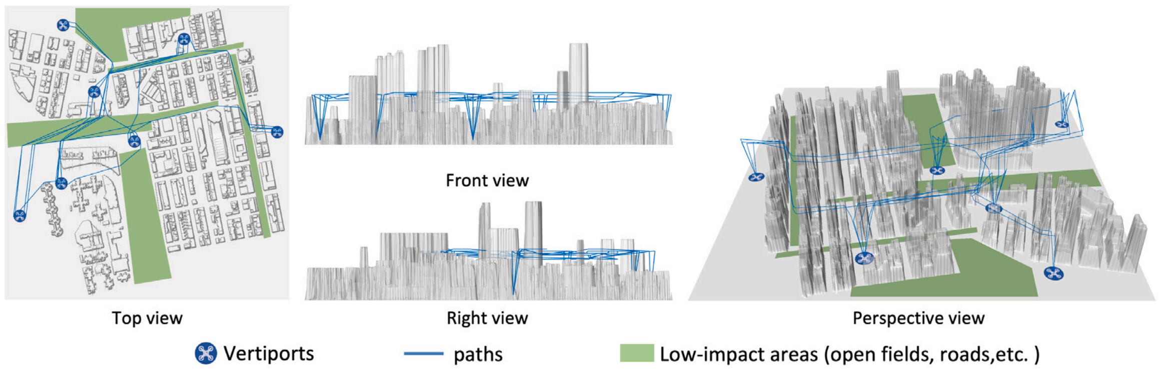
Here, a larger congestion pricing coefficient encourages the algorithm to prioritize congestion pricing optimization. This adjustment enables routes to shift towards less congested areas faster. However, this prioritization might lead to overlooking certain



(a) No low-impact area exists (individual optimal paths)



(b) No low-impact area exists (DRP-CGSTN)

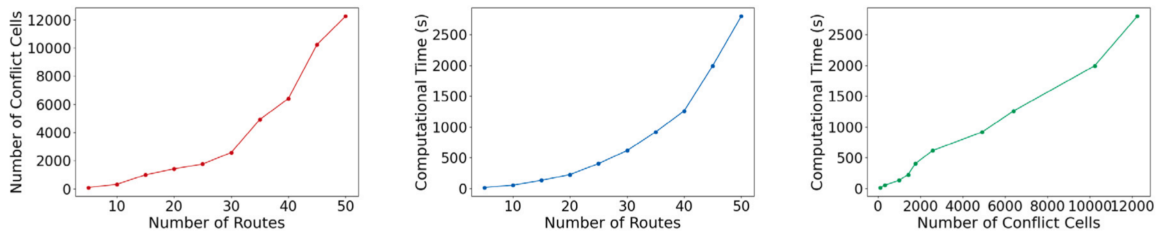


(c) Low-impact areas exist (DRP-CGSTN)

Fig. 14. Routes found by DRP-CGSTN in a real-world urban scenario (920 ∗ 949 ∗ 50 grid cells).

Table 4
Cost and computational time for the different $\Delta\alpha_t$ and ΔL_k for DRP-CGSTN.

$\Delta\alpha_t$	0.2	0.4	0.6	0.8	1.0
$Cost$	7442	7442	7443	7443	7445
$Time$ (s)	173	168	168	165	164
ΔL_k	5	10	15	20	∞
$Cost$	7442	7441	7441	7439	7439
$Time$ (s)	173	206	266	291	356



(a) Number of Conflict Cells vs. Number of Routes (b) Computational Time vs. Number of Routes (c) Computational Time vs. Number of Conflict Cells

Fig. 15. Algorithm scalability as the number of routes increases (920 ∗ 949 ∗ 50 grid cells).

solutions, resulting in higher overall costs but reduced computational time. Regarding the local search space, a smaller search space can expedite the congestion reduction plan generation. Yet this also comes with the fact that some potential solutions might be missed, leading to an increase in total cost.

6.5. Algorithm scalability and computational time

In this section, we show how the computational time of the algorithm increases as the number of routes increases. The test scenario is the urban area in Mong Kok as shown in Section 6.3. There are multiple vertiports. Multiple routes can be generated between any two vertiports if the routes use different fixes of the vertiports. The total number of routes tested ranges from 5 to 50. To ensure consistency in our experimental setup, all tests were conducted using an Ubuntu server with parameters 32x Intel (R) Xeon (R) Gold 5218 CPUs and 13 GB memory.

The results, as shown in Fig. 15, indicate a superlinear relationship between the number of routes and both the number of conflict cells and computational time. This observation implies that increases in the number of routes lead to more than proportional increases in both the number of conflict cells and the required computational resources. Furthermore, the results reveal an almost linear relationship between computational time and the number of conflict cells. This linear trend suggests that the primary factor contributing to the increases in computational time is the increment in conflict cells, which is directly correlated with the increase in the number of routes.

7. Conclusion

This study proposes the DRP-CGSTN method to find multiple paths for drone delivery. In this method, a competition mechanism is developed to allow paths to fairly seek individual optimality, not imposing priorities on any particular path; a new cost item, referred to as congestion pricing, is introduced to resolve path conflicts while balancing between individual paths' optimality and system-level optimality.

The DRP-CGSTN method can generate spatially separated structured route networks in urban areas within a reasonable time. A set of tests was carried out to demonstrate that our method can generate route networks with smaller operation, impact, and space costs within an acceptable time. In addition, these networks satisfy all requirements and can be used for commercial use.

With the implementation of the proposed method, drone delivery service providers can quickly design a drone route network and respond to changes in the service network, such as adding/removing routes based on demand and temporary changes in the environment. The competition mechanism enables multiple UAS service suppliers to share a common airspace for commercial use with other service suppliers that also operate in the airspace.

The proposed method can also be applied to any situation where multiple paths have to be generated to transport moving agents between origins and destinations. A typical scenario is warehouse logistics, in which robots move packages among shelves. The proposed method can also be extended to real-time conflict detection and resolution by each drone because of its distributed characteristics. Once a drone detects other drones in the neighborhood airspace, it can exchange its plan, negotiate with others, and update its own plan.

CRediT authorship contribution statement

Xinyu He: Formal analysis, Investigation, Methodology, Software, Writing – original draft, Writing – review & editing. **Lishuai Li:** Conceptualization, Funding acquisition, Investigation, Methodology, Project administration, Supervision, Validation, Writing – original draft, Writing – review & editing. **Yanfang Mo:** Formal analysis, Validation. **Jianxiang Huang:** Validation, Visualization. **S. Joe Qin:** Funding acquisition, Supervision, Writing – review & editing.

Acknowledgments

The work was supported by the Hong Kong Research Grants Council General Research Fund (Project No. CityU 11200823), City University of Hong Kong Strategic Research Grants (Project No. 7005569 and 7020098), and Hong Kong Innovation and Technology Commission Innovation and Technology Fund (Project No. GHP/145/20).

Appendix A

Algorithm 1: Congestion Reduction Plans Generation

```
Input  : A set of original paths $\boldsymbol{r}^k_{[N]o}$
Output: A set of congestion reduction plans, i.e., updated paths $\boldsymbol{r}^k_{[N]u}$
ReductionPlansGeneration($\boldsymbol{r}^k_{[N]o}$)
    //Parallel execution
    foreach $\boldsymbol{r}_n$ do
        $\boldsymbol{r}^k_{nu}$=ExtendedThetaStarWithCongestion($\boldsymbol{r}_n$,$\boldsymbol{r}^k_{[N]o}$);
    end
    Return a set of paths $\boldsymbol{r}^k_{[N]u}$;
ExtendedThetaStarWithCongestion($\boldsymbol{r}_n$,$\boldsymbol{r}^k_{[N]o}$)
    $open := closed := \emptyset$;  $open.Insert(s_{start})$;
    while $open \neq \emptyset$ do
        $s' := open.Pop()$;  $closed := closed \cup \{s'\}$;
        if $s' = s_{goal}$ then
            return "path found";
        end
        foreach $s \in neighbor(s')$ do
            if $s \notin closed$ and $s \notin open$ then
                $f(s_{start}, s) := \infty$;  $parent(s) := NULL$;
            end
            ComputePathCost($s'$, $s$);
        end
        return "no path found";
    end

calculateCongestionCost($s'$,$s$)
    $C_t(s', s) = C_l(\boldsymbol{r}_{[N]}) - C_l(\boldsymbol{r}_{-n})$;
    return $C_t(s', s)$;

ComputePathCost($s'$,$s$)
    $C_t(s', s) = calculateCongestionCost(s', s)$
    $f(s', s) = C_o(s', s) + \alpha_a C_a(s', s) + \alpha_i C_i(s', s) + \alpha_t C_t(s', s)$
    if $f(s_{start}, s') + f(s', s) < f(s_{start}, s)$ then
        $parent(s) := s'$
        $f(s_{start}, s) := f(s_{start}, s') + f(s', s)$
    end
```

The proposed congestion pricing function is highlighted as red in the Algorithm.

Appendix B

Proof. We used hypothesis induction to prove optimality. $k = 0$ is the initial path generation, in the 0th iteration, $\lambda_{ij} = 0$, and $\boldsymbol{r}^0_{[N]} = arg\ min_{\boldsymbol{r}_{[N]}} \sum_{n\in[N]} C(\boldsymbol{r}_n)$. This is directly proved by Assumption 1.

Assume the optimality holds for a particular k, i.e.,

$$\boldsymbol{r}^k_{[N]} = \arg\min_{\boldsymbol{r}_{[N]}} \sum_{i\in[N]} C(\boldsymbol{r}_i) + \sum_{i,j\in[N],i\neq j} \lambda^k_{ij}\mathcal{N}(i,j) \quad (15)$$

We need to show that

$$\boldsymbol{r}_{[N]}^{k+1} = \arg\min_{\boldsymbol{r}_{[N]}} \sum_{i\in[N]} C(\boldsymbol{r}_i) + \sum_{i,j\in[N],i\neq j} \lambda_{ij}^{k+1}\mathcal{N}(i,j) \tag{16}$$

Here, we show it by contradiction.

If Eq. (16) is not valid, there exists $\boldsymbol{r}'_{[N]}$ that

$$\boldsymbol{r}'_{[N]} = \arg\min_{\boldsymbol{r}_{[N]}} \sum_{i\in[N]} C(\boldsymbol{r}_i) + \sum_{i,j\in[N],i\neq j} \lambda_{ij}^{k+1}\mathcal{N}(i,j) \tag{17}$$

which means that

$$\sum_{i\in[N]} C(\boldsymbol{r}_i^{k+1}) + \sum_{i,j\in[N],i\neq j} \lambda_{ij}^{k+1}\mathcal{N}(i^{k+1},j^{k+1}) > \sum_{i\in[N]} C(\boldsymbol{r}'_i) + \sum_{i,j\in[N],i\neq j} \lambda_{ij}^{k+1}\mathcal{N}(i',j') \tag{18}$$

For *DRP-CGSTN*, it finds the paths

$$\boldsymbol{r}_n^{k+1} = \boldsymbol{r}_n^k = \arg\min_{\boldsymbol{r}_n} C(\boldsymbol{r}_n) + \sum_{l\in[N_{-n}]} \lambda_{nl}^k \mathcal{N}(n,l^k) \quad \forall n\in[N_{-m}] \tag{19}$$

$$\boldsymbol{r}_m^{k+1} = \arg\min_{\boldsymbol{r}_m} C(\boldsymbol{r}_m) + \sum_{l\in[N_{-m}]} \lambda_{ml}^{k+1}\mathcal{N}(m,l^k) \tag{20}$$

$$\lambda_{ij}^{k+1} = \lambda_{ij}^k \quad \forall i\in[N_{-m}] \tag{21}$$

Eq. (18) is now

$$\begin{gathered}\textstyle\sum_{n\in[N_{-m}]}(C(\boldsymbol{r}_n^{k+1}) - C(\boldsymbol{r}'_n)) + \sum_{i,j\in[N_{-m}],i\neq j} \lambda_{ij}^{k+1}(\mathcal{N}(i^{k+1},j^{k+1}) - \mathcal{N}(i',j')) \\ \textstyle > C(\boldsymbol{r}'_m) + \sum_{l\in[N_{-m}]} \lambda_{ml}^{k+1}\mathcal{N}(m',l') - C(\boldsymbol{r}_m^{k+1}) - \sum_{l\in[N_{-m}]} \lambda_{ml}^{k+1}\mathcal{N}(m^{k+1},l^{k+1})\end{gathered} \tag{22}$$

Then, the right-hand side of Eq. (22) is

$$right = \min_{\boldsymbol{r}_m} f(\boldsymbol{r}_m|(\lambda^{k+1},\boldsymbol{r}_{-m}|\lambda^{k+1})) - \min_{\boldsymbol{r}_m} f(\boldsymbol{r}_m|(\lambda^{k+1},\boldsymbol{r}_{-m}|\lambda^k)) \tag{23}$$

If we take

$$\boldsymbol{r}''_m = \arg\min_{\boldsymbol{r}_m} f(\boldsymbol{r}_m,\lambda^k|\boldsymbol{r}_{-m},\lambda^{k+1}) = \arg\min_{\boldsymbol{r}_m} C(\boldsymbol{r}_m) + \sum_{l\in[N_{-m}]} \lambda_{ml}^{k+1}\mathcal{N}(m,l') \tag{24}$$

the left-hand side of Eq. (22) is

$$\begin{gathered}\textstyle left = \sum_{n\in[N_{-m}]}(C(\boldsymbol{r}_n^k) - C(\boldsymbol{r}'_n)) + \sum_{i,j\in[N_{-m}],i\neq j} \lambda_{ij}^k(\mathcal{N}(i^k,j^k) - \mathcal{N}(i',j')) \\ \textstyle = [\sum_{i\in[N]} C(\boldsymbol{r}_i^k) - C(\boldsymbol{r}_m^k) - \sum_{n\in[N_{-m}]} C(\boldsymbol{r}'_n) - C(\boldsymbol{r}''_m) + C(\boldsymbol{r}''_m)] + [\sum_{i,j\in[N],i\neq j} \lambda_{ij}^k\mathcal{N}(i^k,j^k) - \sum_{l\in[N_{-m}]} \lambda_{ml}^k\mathcal{N}(m^k,l^k) \\ \textstyle - \sum_{i,j\in[N_{-m}],i\neq j} \lambda_{ij}^k\mathcal{N}(i',j') - \sum_{l\in[N_{-m}]} \lambda_{ml}^k\mathcal{N}(m'',l') + \sum_{l\in[N_{-m}]} \lambda_{ml}^k\mathcal{N}(m'',l')]\end{gathered} \tag{25}$$

It can be rewritten as

$$\begin{gathered}\textstyle left = (\sum_{i\in[N]} C(\boldsymbol{r}_i^k) + \lambda_{ij}^k\mathcal{N}(i^k,j^k)) - (\sum_{n\in[N_{-m}]} C(\boldsymbol{r}'_n) + C(\boldsymbol{r}''_m) + \sum_{i,j\in[N_{-m}],i\neq j} \lambda_{ij}^k\mathcal{N}(i',j') \\ \textstyle + \sum_{l\in[N_{-m}]} \lambda_{ml}^k\mathcal{N}(m'',l')) + (C(\boldsymbol{r}''_m) + \sum_{l\in[N_{-m}]} \lambda_{ml}^k\mathcal{N}(m'',l')) - (C(\boldsymbol{r}_m^k) + \sum_{l\in[N_{-m}]} \lambda_{ml}^k\mathcal{N}(m^k,l^k))\end{gathered} \tag{26}$$

Based on Eq. (15), we have

$$\begin{aligned}\textstyle left &< \textstyle 0 + C(\boldsymbol{r}''_m) + \sum_{l\in[N_{-m}]} \lambda_{ml}^k\mathcal{N}(m'',l') - C(\boldsymbol{r}_m^k) - \sum_{l\in[N_{-m}]} \lambda_{ml}^k\mathcal{N}(m^k,l^k) \\ &= \min_{\boldsymbol{r}_m} f(\boldsymbol{r}_m|(\lambda^k,\boldsymbol{r}_{-m}|\lambda^{k+1})) - \min_{\boldsymbol{r}_m} f(\boldsymbol{r}_m|(\lambda^k,\boldsymbol{r}_{-m}|\lambda^k))\end{aligned} \tag{27}$$

Therefore,

$$\min_{\boldsymbol{r}_m} f(\boldsymbol{r}_m|(\lambda^{k+1},\boldsymbol{r}_{-m}|\lambda^k)) - \min_{\boldsymbol{r}_m} f(\boldsymbol{r}_m|(\lambda^k,\boldsymbol{r}_{-m}|\lambda^k)) > \min_{\boldsymbol{r}_m} f(\boldsymbol{r}_m|(\lambda^{k+1},\boldsymbol{r}_{-m}|\lambda^{k+1})) - \min_{\boldsymbol{r}_m} f(\boldsymbol{r}_m|(\lambda^k,\boldsymbol{r}_{-m}|\lambda^{k+1})) \tag{28}$$

As Eq. (28) does not match with Eq. (14) in Assumption 2, Eq. (16) is valid and optimality also suits $k+1$.

References

Alotaibi, E.T.S., Al-Rawi, H., 2018. A complete multi-robot path-planning algorithm. Auton. Agents Multi-Agent Syst. 32 (5), 693–740. http://dx.doi.org/10.1007/s10458-018-9391-2.

Aoude, G.S., Luders, B.D., Levine, D.S., How, J.P., 2010. Threat-aware path planning in uncertain urban environments. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, pp. 6058–6063.

Barer, M., Sharon, G., Stern, R., Felner, A., 2014. Suboptimal variants of the conflict-based search algorithm for the multi-agent pathfinding problem. In: Seventh Annual Symposium on Combinatorial Search.

Bauranov, A., Rakas, J., 2021. Designing airspace for urban air mobility: A review of concepts and approaches. Prog. Aerosp. Sci. 125, 100726. http://dx.doi.org/10.1016/j.paerosci.2021.100726.

Bnaya, Z., Felner, A., 2014. Conflict-oriented windowed hierarchical cooperative A*. In: 2014 IEEE International Conference on Robotics and Automation. ICRA, IEEE, pp. 3743–3748.

Cohen, L., Uras, T., Kumar, T.S., Koenig, S., 2019. Optimal and bounded-suboptimal multi-agent motion planning. In: Twelfth Annual Symposium on Combinatorial Search.

Cohen, L., Uras, T., Kumar, T.S., Xu, H., Ayanian, N., Koenig, S., 2016. Improved solvers for bounded-suboptimal multi-agent path finding. In: IJCAI. pp. 3067–3074.

De Wilde, B., Ter Mors, A.W., Witteveen, C., 2014. Push and rotate: a complete multi-agent pathfinding algorithm. J. Artificial Intelligence Res. 51, 443–492. http://dx.doi.org/10.1613/jair.4447.

Desaraju, V.R., How, J.P., 2012. Decentralized path planning for multi-agent teams with complex constraints. Auton. Robots 32 (4), 385–403.

Desaraju, V., Ro, H.C., Yang, M., Tay, E., Roth, S., Del Vecchio, D., 2009. Partial order techniques for vehicle collision avoidance: Application to an autonomous roundabout test-bed. In: 2009 IEEE International Conference on Robotics and Automation. IEEE, pp. 82–87.

EUROCONTROL, 2018. Unmanned aircraft systems (UAS) ATM integration. URL https://www.eurocontrol.int/publication/unmanned-aircraft-systems-uas-atm-integration.

Felner, A., Goldenberg, M., Sharon, G., Stern, R., Beja, T., Sturtevant, N.R., Schaeffer, J., Holte, R., 2012. Partial-expansion A* with selective node generation. In: Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, 2012. Toronto, Ontario, Toronto, Ontario, Canada, pp. 471–477.

Felner, A., Stern, R., Shimony, S.E., Boyarski, E., Goldenberg, M., Sharon, G., Sturtevant, N., Wagner, G., Surynek, P., 2017. Search-based optimal solvers for the multi-agent pathfinding problem: Summary and challenges. In: Proceedings of the 10th Annual Symposium on Combinatorial Search, SoCS 2017. Vol. 2017-Janua, pp. 29–37.

Ferrera, E., Capitan, J., Castano, A.R., Marron, P.J., 2017. Decentralized safe conflict resolution for multiple robots in dense scenarios. Robot. Auton. Syst. 91, 179–193. http://dx.doi.org/10.1016/j.robot.2017.01.008.

Gilboa, A., Meisels, A., Felner, A., 2006. Distributed navigation in an unknown physical environment. In: Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems. ACM Press, New York, New York, USA, pp. 553–560. http://dx.doi.org/10.1145/1160633.1160735.

Goldenberg, M., Felner, A., Stern, R., Sharon, G., Sturtevant, N., Holte, R.C., Schaeffer, J., 2014. Enhanced partial expansion A. J. Artificial Intelligence Res. 50, 141–187. http://dx.doi.org/10.1613/jair.4171.

He, X., He, F., Li, L., Zhang, L., Xiao, G., 2022. A route network planning method for urban air delivery. Transp. Res. E 166, 102872. http://dx.doi.org/10.1016/J.TRE.2022.102872, URL http://arxiv.org/abs/2206.03085.

Ho, F., Geraldes, R., Gonçalves, A., Rigault, B., Sportich, B., Kubo, D., Cavazza, M., Prendinger, H., 2022. Decentralized multi-agent path finding for UAV traffic management. IEEE Trans. Intell. Transp. Syst. 23 (2), 997–1008. http://dx.doi.org/10.1109/TITS.2020.3019397.

Ho, F., Goncalves, A., Salta, A., Cavazza, M., Geraldes, R., Prendinger, H., 2019. Multi-agent path finding for UAV traffic management. In: International Conference on Autonomous Agents and Multiagent Systems, May 13-17, 2019, MontrÉal, Canada. Association for Computing Machinery (ACM), pp. 131–139.

Hoekstra, J.M., Van Gent, R.N., Ruigrok, R.C., 2002. Designing for safety: The ’free flight’ air traffic management concept. Reliab. Eng. Syst. Saf. 75 (2), 215–232. http://dx.doi.org/10.1016/S0951-8320(01)00096-5.

Hoffmann, G.M., Tomlin, C.J., 2008. Decentralized cooperative collision avoidance for acceleration constrained vehicles. In: 47th IEEE Conference on Decision and Control. IEEE, pp. 4357–4363.

Hong Kong Census and Statistics Department, 2023. Population estimates. URL https://www.censtatd.gov.hk/en/scode150.html.

Jang, D.S., Ippolito, C., Sankararaman, S., Stepanyan, V., 2017. Concepts of airspace structures and system analysis for UAS traffic flows for urban areas. In: AIAA Information Systems-AIAA Infotech At Aerospace, January 9-13, 2017. Grapevine, Texas, USA, http://dx.doi.org/10.2514/6.2017-0449.

Kornhauser, D., Miller, G., Spirakis, P., 1984. Coordinating pebble motion on graphs, the diameter of permutation groups, and applications. In: 25th Annual Symposium on Foundations of Computer Science, 1984.. IEEE, pp. 241–250.

Krozel, J., Peters, M., Bilimoria, K.D., Lee, C., Mitchell, J.S., 2001. System performance characteristics of centralized and decentralized air traffic separation strategies. Air Traffic Control Q. 9 (4), 311–332. http://dx.doi.org/10.2514/atcq.9.4.311.

Li, Y., Gupta, K., Payandeh, S., 2005. Motion planning of multiple agents in virtual environments using coordination graphs. In: Proceedings of the 2005 IEEE International Conference on Robotics and Automation, April 18-22, 2005. IEEE, Barcelona, Spain, Spain, pp. 378–383. http://dx.doi.org/10.1109/ROBOT.2005.1570148.

Lumelsky, V.J., Harinarayan, K.R., 1997. Decentralized motion planning for multiple mobile robots: The cocktail party model. Auton. Robots 4 (1), 121–135.

Luna, R., Bekris, K.E., 2011. Efficient and complete centralized multi-robot path planning. In: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, September 25-30, 2011. IEEE, San Francisco, CA, USA, pp. 3268–3275. http://dx.doi.org/10.1109/IROS.2011.6095085.

Mohamed Salleh, M.F.B., Wanchao, C., Wang, Z., Huang, S., Tan, D.Y., Huang, T., Low, K.H., 2018. Preliminary concept of adaptive urban airspace management for unmanned aircraft operations. In: 2018 AIAA Information Systems-AIAA Infotech@ Aerospace. p. 2260.

Murray, C.C., Chu, A.G., 2015. The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery. Transp. Res. C 54, 86–109. http://dx.doi.org/10.1016/j.trc.2015.03.005.

Murray, C.C., Raj, R., 2020. The multiple flying sidekicks traveling salesman problem: Parcel delivery with multiple drones. Transp. Res. C 110, 368–398. http://dx.doi.org/10.1016/j.trc.2019.11.003.

NASA, 2021. UTM concept of operations v2.0. URL https://www.faa.gov/uas/research_development/traffic_management/media/UTM_ConOps_v2.pdf.

Olfati-Saber, R., Fax, J.A., Murray, R.M., 2007. Consensus and cooperation in networked multi-agent systems. Proc. IEEE 95 (1), 215–233.

Purwin, O., D’Andrea, R., Lee, J.-W., 2008. Theory and implementation of path planning by negotiation for decentralized agents. Robot. Auton. Syst. 56 (5), 422–436.

Sacramento, D., Pisinger, D., Ropke, S., 2019. An adaptive large neighborhood search metaheuristic for the vehicle routing problem with drones. Transp. Res. C 102, 289–315. http://dx.doi.org/10.1016/j.trc.2019.02.018.

Saha, M., Isto, P., 2006. Multi-robot motion planning by incremental coordination. In: 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, October 9-15, 2006. IEEE, Beijing, China, pp. 5960–5963. http://dx.doi.org/10.1109/IROS.2006.282536.

Sanchez, G., Latombe, J.-C., 2002. Using a PRM planner to compare centralized and decoupled planning for multi-robot systems. In: Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292), May 11-15, 2002. Vol. 2, IEEE, Washington, DC, USA, pp. 2112–2119. http://dx.doi.org/10.1109/ROBOT.2002.1014852.

Sarina, C., Tore, J., Stephan, L., Robin, R., Leonard, T., 2022. Drone delivery: More lift than you think. URL https://www.mckinsey.com/industries/aerospace-and-defense/our-insights/future-air-mobility-blog/drone-delivery-more-lift-than-you-think.

Scerri, P., Owens, S., Yu, B., Sycara, K., 2007. A decentralized approach to space deconfliction. In: 2007 10th International Conference on Information Fusion. IEEE, pp. 1–8.

Schermer, D., Moeini, M., Wendt, O., 2019. A matheuristic for the vehicle routing problem with drones and its variants. Transp. Res. C 106, 166–204. http://dx.doi.org/10.1016/j.trc.2019.06.016.

SESAR, 2019. SESAR concept of operations for U-Space. URL https://www.sesarju.eu/node/3411.

SESAR, 2021. Sustainability, airspace optimization and urban air mobility - focus of latest very large-scale demonstrations. URL https://www.sesarju.eu/news/sustainability-capacity-and-urban-air-mobility-focus-newly-launched-sesar-demonstration-call.

Sharon, G., Stern, R., Felner, A., Sturtevant, N.R., 2015. Conflict-based search for optimal multi-agent pathfinding. Artificial Intelligence 219, 40–66. http://dx.doi.org/10.1016/j.artint.2014.11.006.

Silver, D., 2005. Cooperative pathfinding. In: Proceedings of the First Artificial Intelligence and Interactive Digital Entertainment, June 1-5, 2005. Vol. 1, Marina Del Rey, Marina del Rey, California, USA, pp. 117–122.

Standley, T., 2010. Finding optimal solutions to cooperative pathfinding problems. In: Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, July 11-15, 2010. Atlanta, Georgia, USA.

Stern, R., Sturtevant, N., Felner, A., Koenig, S., Ma, H., Walker, T., Li, J., Atzmon, D., Cohen, L., Kumar, T.K., 2019. Multi-agent pathfinding: Definitions, variants, and benchmarks. arXiv preprint arXiv:1906.08291.

Sunil, E., Hoekstra, J.M., Ellerbroek, J., Bussink, F., Nieuwenhuisen, D., Vidosavljevic, A., Kern, S., 2015. Metropolis: Relating airspace structure and capacity for extreme traffic densities. In: Proceedings of the 11th USA/Europe Air Traffic Management Research and Development Seminar (ATM2015), Lisbon (Portugal), 23-26 June, 2015. FAA/Eurocontrol.

Surynek, P., 2012. Towards optimal cooperative path planning in hard setups through satisfiability solving. In: Anthony, P., Ishizuka, M., Lukose, D. (Eds.), PRICAI 2012: Trends in Artificial Intelligence, September 3-7, 2012. Springer Berlin Heidelberg, Kuching, Malaysia, pp. 564–576. http://dx.doi.org/10.1007/978-3-642-32695-0_50.

Tan, Q., Wang, Z., Ong, Y.-S., Low, K.H., 2019. Evolutionary optimization-based mission planning for UAS traffic management (UTM). In: 2019 International Conference on Unmanned Aircraft Systems. ICUAS, IEEE, pp. 952–958.

Tang, H., Zhang, Y., Mohmoodian, V., Charkhgard, H., 2021. Automated flight planning of high-density urban air mobility. Transp. Res. C 131, 103324.

Ushijima, H., 2017. UTM project in Japan. In: Proceedings of the Global UTM Conference, Montreal, QC, Canada. Vol. 26.

Wagner, G., Choset, H., 2015. Subdimensional expansion for multirobot path planning. Artificial Intelligence 219, 1–24.

Wang, H., Rubenstein, M., 2020. Walk, stop, count, and swap: decentralized multi-agent path finding with theoretical guarantees. IEEE Robot. Autom. Lett. 5 (2), 1119–1126.

Warren, C.W., 1990. Multiple robot path coordination using artificial potential fields. In: Proceedings of the IEEE International Conference on Robotics and Automation, May 13-18, 1990. IEEE, Cincinnati, OH, USA, pp. 500–505. http://dx.doi.org/10.1109/ROBOT.1990.126028.

Wu, Y., Low, K.H., Pang, B., Tan, Q., 2021. Swarm-based 4D path planning for drone operations in urban environments. IEEE Trans. Veh. Technol. 70 (8), 7464–7479.

Yang, Z., 2023. Food delivery by drone is just part of daily life in Shenzhen. URL https://www.technologyreview.com/2023/05/23/1073500/drone-food-delivery-shenzhen-meituan/, Accessed: 2023-12-07.

Yang, X., Wei, P., 2018. Autonomous on-demand free flight operations in urban air mobility using Monte Carlo tree search. In: International Conference on Research in Air Transportation (ICRAT), June 26-29, 2018. Barcelona, Spain.

Yang, X., Wei, P., 2021. Autonomous free flight operations in urban air mobility with computational guidance and collision avoidance. IEEE Trans. Intell. Transp. Syst..

Yu, J., LaValle, S.M., 2016. Optimal multirobot path planning on graphs: Complete algorithms and effective heuristics. IEEE Trans. Robot. 32 (5), 1163–1177. http://dx.doi.org/10.1109/TRO.2016.2593448.

Zhao, Y., Zheng, Z., Liu, Y., 2018. Survey on computational-intelligence-based UAV path planning. Knowl.-Based Syst. 158, 54–64.