Delft University of Technology

A route network planning method for urban air delivery

He, Xinyu; He, Fang; Li, Lishuai; Zhang, Lei; Xiao, Gang

A route network planning method for urban air delivery

Xinyu He [a], Fang He [a,b], Lishuai Li [a,c,*], Lei Zhang [d], Gang Xiao [b]

[a] *School of Data Science, City University of Hong Kong, Hong Kong Special Administrative Region*
[b] *School of Aeronautics and Astronautics, Shanghai Jiao Tong University, Shanghai, China*
[c] *Air Transport and Operations, Faculty of Aerospace Engineering, Delft University of Technology, Delft, Netherlands*
[d] *Antwork Technology Co., Ltd, Hangzhou, China*

ARTICLE INFO

ABSTRACT

High-tech giants and start-ups are investing in drone technologies to provide urban air delivery service, which is expected to solve the last-mile problem and mitigate road traffic congestion. However, air delivery service will not scale up without proper traffic management for drones in dense urban environment. Currently, a range of Concepts of Operations (ConOps) for unmanned aircraft system traffic management (UTM) are being proposed and evaluated by researchers, operators, and regulators. Among these, the tube-based (or corridor-based) ConOps has emerged in operations in some regions of the world for drone deliveries and is expected to continue serving certain scenarios that with dense and complex airspace and requires centralized control in the future. Towards the tube-based ConOps, we develop a route network planning method to design routes (tubes) in a complex urban environment in this paper. In this method, we propose a priority structure to decouple the network planning problem, which is NP-hard, into single-path planning problems. We also introduce a novel space cost function to enable the design of dense and aligned routes in a network. The proposed method is tested on various scenarios and compared with other state-of-the-art methods. Results show that our method can generate near-optimal route networks with significant computational time-savings.

1. Introduction

Urban parcel delivery via drones belongs to the broad concept of Urban Air Mobility (UAM). It is a rapidly emerging field in research and business development, with prospects to ease urban traffic congestion, connect remote areas with great agility, lower labor costs in logistics, and ensure goods delivery in emergencies (Duvall et al., 2019; Chung, 2021; Škrinjar et al., 2019; Rajendran and Srinivas, 2020; Kellermann et al., 2020; Kitonsa and Kruglikov, 2018; Lemardelé et al., 2021). Although the number of operations of drone deliveries is not large yet, the global market of drones and electric aircraft operations is expected to increase to tens of billions of USD in the early 2030s estimated by McKinsey (Kersten et al., 2021), and around 1 trillion USD by 2040 estimated by Morgan Stanley (MorganStanley, 2021). To scale up the operations, a key challenge lies in how to manage a large volume of drone operations in a dense urban environment efficiently and safely. Therefore, there are several ongoing R&D programs to explore concepts of operation, data exchange requirements, and a supporting framework to enable drone operations at low altitudes in airspace where traditional air traffic services are not provided, 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), J-UTM (Ushijima, 2017), etc.
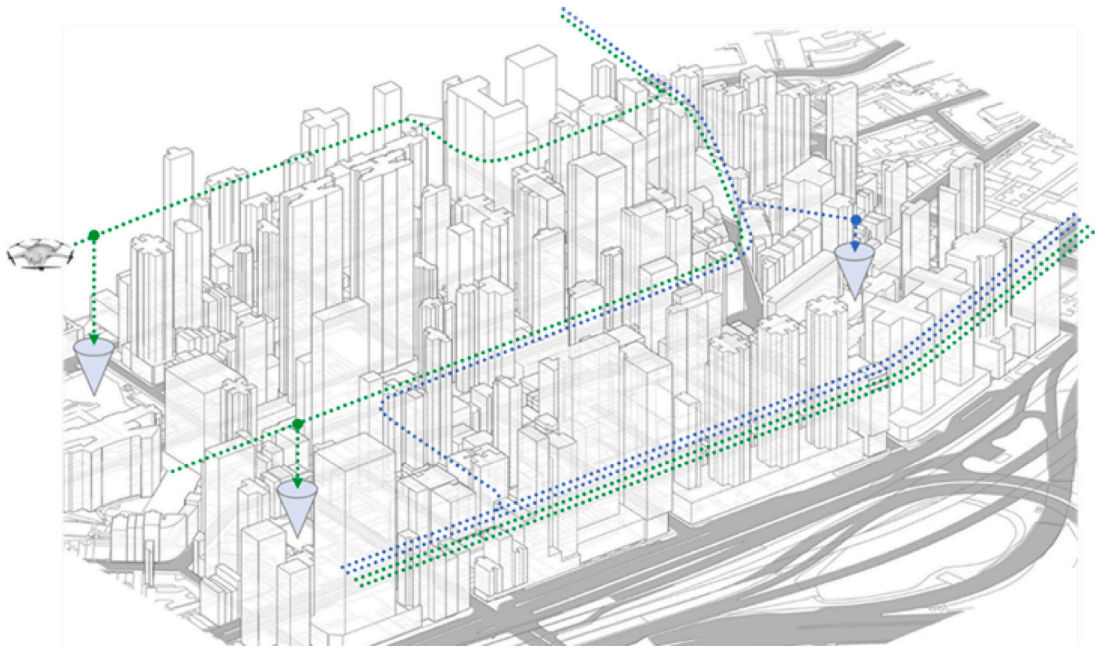

Fig. 1. An imaginary scenario of urban air delivery route network.

A range of UTM Concepts of Operations (ConOps) for traffic and airspace management are being explored and studied (SESAR, 2019; Bauranov and Rakas, 2021). Each ConOps has its own advantages and disadvantages. As stated in EUROCONTROL (2018), different ConOps may co-exist in the future and each suits certain scenarios. For example, free-flight-based operations spread the traffic over the whole airspace to reduce the number of potential conflicts (Hoekstra et al., 2002; Jardin, 2005; Krozel et al., 2001; Yang and Wei, 2018). It allows each drone to follow its optimal path, detect and avoid other flights. Structure-based operations use traffic flow constraints to reduce airspace complexity and management workload (Krozel et al., 2001; Octavian Thor and Bogdan, 2009; Sunil et al., 2015). The structures may include layers, zones, and tubes (or air corridors); they separate drones and organize traffic flows to reduce potential conflicts (Bin Mohammed Salleh et al., 2018; Jang et al., 2017; Kopardekar, 2014; Kopardekar et al., 2016).

Among these ConOps, the concept of tube-based operations has been proposed by Eurocontrol (EUROCONTROL, 2018) and NASA (Jang et al., 2017) as a kind of structure-based operations. Tubes are pre-planned routes "to cover for higher traffic demands, specifically in areas where high volume flight routes occur or there are needs to manage routing to cater for safety, security, noise, and privacy issues" (EUROCONTROL, 2018). They are also referred to as "structured routes", or "air corridors". These structured routes can follow rivers, railway lines, or other geographical areas where there is minimal impact on people on the ground. An imaginary tube-based route network scenario is shown in Fig. 1. Recently, tube-based operations have been implemented for drone deliveries in a few cities in China. For example, a start-up company, Antwork Technology, has been operating drone parcel deliveries on the tube-based inner-city network in Hangzhou, China, since 2019 when it obtained the world's first approval and business license from the Civil Aviation Administration of China (CAAC) for operating commercial drone deliveries in urban areas. We expect the tube-based operations will continue to grow for regions with dense and complex airspace and that requires centralized control in the future.

To support the tube-based operations, we develop a route network planning method to design routes (tubes) in a complex urban environment in this paper. In general, the design of a route network for drone deliveries includes (1) strategic design, (2) tactical design. The strategic design is mainly driven by business considerations, e.g. identifying areas with high demand, needing medical/emergency responses, or lacking other transportation infrastructure, to select the vertiport locations and the network types to best serve a market. This kind of network design is outside the scope of this work, which is expected to be decided before planning the routes as presented in this work. The tactical design focuses on path planning for drone operations considering operational constraints, such as air traffic management restrictions, safety requirements, public acceptance, noise ordinances, etc. The tactical design of a route network for drone operations in an urban environment is the focus of this work.

There are many existing path planning methods in the literature. However, to the best of our knowledge, none of them can be directly used to design a route network for urban air delivery at a city level due to the computational complexity involved in real-world scenarios. The problem of designing an optimal network with spatially separated routes is NP-hard (Yu and LaValle, 2013b). Path finding for one route is coupled with path finding for other routes. One route is planned with the shortest length (or the lowest cost), but other routes may need to take a significant detour to avoid any conflict, resulting in a sub-optimal solution at the system level. The computational complexity increases exponentially as the number of conflicts among routes increases. In

addition, there is an unique requirement on the design of a drone delivery network in an urban area, which is the spread of a route network needs to be minimized to reduce the impact of high volume drone operations on safety, noise, and privacy, etc. Therefore, the routes in a network need to be aligned and close to each other as much as possible, which would reduce the total size of areas been affected by drone operations and increase the utilization of limited airspace in urban areas. The alignment of traffic flows would also decrease the traffic complexity in airspace and reduce the probability of separation loss (Hoekstra et al., 2018). However, no existing studies on path planning have considered network spread and airspace utilization.

To tackle these challenges, we propose a sequential path finding method to design a tube-based route network for drone operations in an urban environment considering airspace utilization. In this method, we propose a prioritization structure to decouple the network planning problem into a set of single-path planning problems to generate paths sequentially. We set the prioritization rules to rank origin–destination (OD) pairs to ensure important routes with higher priority in planning for better system performance. To obtain better airspace utilization, we introduce a space cost function in the underlying path-finding algorithm to build a network with dense and aligned routes. The contributions of this work are three folds:

(1) A sequential route network planning method with prioritization is developed to support tube-based operations for Unmanned Aerial Vehicles (UAVs) traffic management. The proposed prioritization framework can solve the NP-hard problem by decoupling the multi path-finding problem into sequential single path-finding problems to generate results fast.
(2) A new technique, referred to as space cost function, is developed to enable the design of dense route networks with improved airspace utilization.
(3) Comparative experiments and real-world applications are carried out to benchmark with other state-of-the-art methods and show the performance of the proposed method under various settings in real-world scenarios.

The remainder of this paper is structured as follows. An overview of related academic literature is provided in Section 2. The statement of the problem is illustrated in Section 3. The proposed method for route network planning is explained in Section 4. We evaluate the proposed method in testing scenarios and a real-world scenario in Section 5. Finally, Section 6 concludes the research findings and proposes future work.

2. Literature review

On drone delivery problem, a large number of studies focuses on vehicle routing problem with drones (Shen et al., 2020) and traffic management problem for drones. The vehicle routing problems involve how to coordinate delivery trucks and drones to deliver small parcels to geographically distributed customers (Moshref-Javadi et al., 2020; Ha et al., 2018; Karak and Abdelghany, 2019; Murray and Chu, 2015; Murray and Raj, 2020; Sacramento et al., 2019; Schermer et al., 2019; Zhang et al., 2021). The traffic management problem is about how drones fly safely to finish tasks. It includes three steps. One is to plan a trajectory for each drone operation (Yang and Sukkarieh, 2008; Cekmez et al., 2014; Sonmez et al., 2015; Zhao et al., 2018; Wu et al., 2021), one is to detect conflicts when drones follow these trajectories (Tan et al., 2019; Islami et al., 2017; Kuchar and Yang, 2000), one is to resolve conflicts among drones if conflicts appear (Yang and Wei, 2020; Bertram et al., 2021; Yang and Wei, 2021; Tang et al., 2021).

In this paper, we focus on the problem of designing a network with multiple routes in an urban environment; each route connects an origin and a destination and is spatially separated from each other. Related studies can be broadly grouped in to two categories : the single-path finding (SPF) problem and the multi-path finding (MPF) problem.

2.1. Single-path finding

SPF involves moving a single agent to a destination among a set of obstacles. SPF is a well-studied area and many algorithms are proposed to find a path for the single agent.

Graph-search-based algorithms require a discretization of the state space, they search over the graph to find a path. The Dijkstra's algorithm was the pioneer (Dijkstra, 1959; Liu et al., 2020), it searches the minimum cost paths between two points in the graph by comparing all possible combinations. A* is an advancement of the Dijkstra's algorithm (Hart et al., 1968), it uses a heuristic function to estimate the cost of reaching the goal to reduce computational times. Many variants of A* were developed, like Dynamic A* (D*) (Koenig and Likhachev, 2005), Theta* (Daniel et al., 2010). For these methods, some solutions might be missed if the resolution of the discretization is large, and these methods do not guarantee to find an optimal path because of the heuristic nature, unless the heuristic is admissible, i.e., it never overestimates the cost of reaching the goal. In summary, these methods are efficient to find near-optimal paths with an appropriate resolution of the discretization in terms of the large size of the designing space.

Sampling-based algorithms do not require a discretization of the state space, they generate some possible paths by randomly adding points in the environment into a tree until some solution is found or time expires. Two prominent examples are Probabilistic Roadmap Method (PRM) (Kavraki et al., 1996) and Rapidly-Exploring Random Tree (RRT) (Lavalle, 1998). The PRM method first samples points in the environment and connect points to form a roadmap, and then it searches for a feasible path using this constructed roadmap. The RRT method grows a unidirectional search tree from the starting point until a tree branch hits the goal point. The RRT method guarantees finding a feasible solution as the number of iterations approaches infinity, however, the paths are often much longer than the optimal paths. Though its variant RRT* (Karaman and Frazzoli, 2011) is asymptotically optimal, the convergence rate is very slow. A comparison shows (Zammit and Van Kampen, 2018) that A*'s path length is more optimal and

generation time is shorter than RRT for path planning for UAVs. In summary, sampling-based methods are still efficient to find a feasible path in terms of the large size of the designing space, but the path is extremely sub-optimal.

There are also some methods like mathematical optimization-based algorithms, neural network-based algorithms, nature-inspired algorithms. The mathematical optimization-based algorithms formulate the path finding problem as binary linear programming problems (Chamseddine et al., 2012) or mixed-integer programming problems (Culligan et al., 2007), and use high-quality solvers to solve these programming problems. Neural network-based algorithms (Yang and Meng, 2000; Dezfoulian et al., 2013; Singh and Thongam, 2019) use neural networks to model complex environments. Natural inspired algorithms, like genetic algorithms (Hu and Yang, 2004), particle swarm optimization (Masehian and Sedighizadeh, 2010), ant colony optimization (Englot and Hover, 2011) are also successfully applied for path finding. However, these methods are time consuming, they are not efficient to find a path in terms of the large size of the designing space.

In summary, if the size of the designing space is large, A* and its variants are the best choices considering computational time and the optimality of the path.

2.2. Multi-path finding

MPF involves navigating the agents to their respective targets while avoiding collisions with obstacles and other agents. Such problems are NP-hard (Yu and LaValle, 2013b). There are two different Multi-path finding problems based on the different cost function to minimize, one is sum-of-cost and another is makespan (Felner et al., 2017). For sum-of-cost problems, the sum of the total time steps or total distance for all agents should be minimized. For makespan problems, the maximum of the individual path costs should be minimized. Designing tube-based route networks is similar to the sum-of-cost problems, but there is a major difference that the sum-of-cost problems only require paths to be temporal conflict-free, but tube-based route networks further require paths to be spatial conflict-free, i.e., no path can appear at the same place even at different times.

Traditional MPF algorithms fall into two settings: centralized and distributed. In a distributed setting, each agent has its computing power and it does not have full knowledge of other agents. Some work has been done for the distributed setting (Bhattacharya et al., 2010; Gilboa et al., 2006; Grady et al., 2010). By contrast, the centralized setting assumes a single central computing power which needs to find a solution for all agents. The scope of this paper is limited to the centralized setting, thus here we provide a review on MPF algorithms in a centralized setting grouped into two categories: optimal solvers and sub-optimal solvers.

2.2.1. Optimal solvers for centralized MPF

The optimal solvers can generate optimal paths and they are complete in theory. However, the dimension of the problem explode exponentially as the number of agents increases. Thus, they are often used to generate routes for a small number of agents. These solvers can be broadly classified into A*-based, reduction-based, and two-level-based methods.

A* can be well suited to solve MPF sum-of-cost problems by taking k-agent state space. Each state includes k agents as a vector, e.g., in the start state each agent is located at its start point. For each state, there are 4^k neighbors to explore if each agent can move in four cardinal directions. Even for 10 agents, there are $4^{10} \approx 10^6$ neighbors for the start state. As a result, the original A*-based method is computationally infeasible to solve real-world problems. A few techniques are developed to speed up A*. For example, independence detection divides the agents into independent groups and solves these groups separately (Standley, 2010; Wagner and Choset, 2015), here two groups are independent if the optimal solution for each group does not conflict with the other. Another technique is related to surplus nodes, which are nodes generated but never be expanded to find an optimal solution. Avoiding generating the surplus nodes makes a substantial speedup (Standley, 2010; Felner et al., 2012; Goldenberg et al., 2014). In summary, though these techniques provide exponential speedup for A*-based methods, solution quality still degrades rapidly and computational time increases fast as the agent density increase.

Reduction-based methods reduce the MPF problem to standard known problems. Examples include network flow problems with integer linear programming formulations (Yu and LaValle, 2016, 2013a), Boolean satisfiability (Surynek, 2012), and constraint satisfaction problems (Ryan, 2010). These methods are designed for the MPF makespan problems, and they are inefficient or even inapplicable for MPF sum-of-cost problems.

Two-level-based methods introduce a tree structure to solve MPF sum-of-cost problems. Each node in the tree includes a solution for all routes, At the high level, these methods search over the tree to find the optimal solution, and then the low-level search is invoked to generate a node of the tree. Typical two-level-based methods include increasing cost tree search (Sharon et al., 2013) and conflict-based search (CBS) (Sharon et al., 2015). CBS is a state-of-the-art optimal solver. It uses a constraint tree (CT) for search. In a CT, each node includes a set of constraints on agents, a solution for all agents that satisfies the constraints, and the cost of the solution. The root node of the CT contains no constraint, and the paths in the solution are individually optimal if ignoring other agents. The tree is built progressively from root nodes to leaf nodes. In each step, the high-level search assigns a conflict as a constraint for an agent to satisfy. On the leaf nodes, all conflicts are solved, so the solutions are all feasible paths. The leaf node with the minimum cost has optimal paths. In summary, these methods are efficient and find optimal paths for small problems. If there are many agents and many conflicts to solve, their computational times also increase very fast.

2.2.2. Sub-optimal solvers for centralized MPF

The sub-optimal solvers are commonly used to generate feasible routes quickly for a large number of agents for sum-of-cost problems. By decomposing the problem into several sub-problems, the computational time can be significantly reduced. But most of the time they can only find sub-optimal paths and, in some cases, completeness is sacrificed to improve time complexity. The sub-optimal solvers can be roughly classified as rule-based methods, search-based methods, and hybrid methods.

Rule-based solvers make specific agent-movement rules for different scenarios and they do not need massive search steps. They usually require special properties of the graphs to guarantee completeness. The algorithm by Kornhauser et al. (1984) guarantees the completeness for all different graphs but the implementation is complex. BIBOX (Surynek, 2009) is complete for bi-connected graphs. Push-and-Swap (Luna and Bekris, 2011) uses "swap" macro to swap locations between two adjacent dependent agents. Its variants Push-and-Rotate (De Wilde et al., 2014), Push-and-Spin (Alotaibi and Al-Rawi, 2018), etc., use more macros to handle complex situations and graphs. However, deadlocks often happen in narrow corridors and inter-agent collision-free may not be guaranteed, so the algorithms may fail to find paths even the paths exist (incompleteness). In summary, rule-based methods are efficient if there are a large number of agents, but the generated results are often far away from optimal.
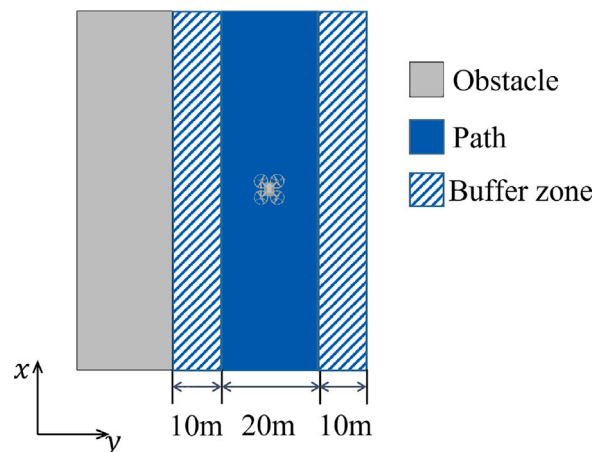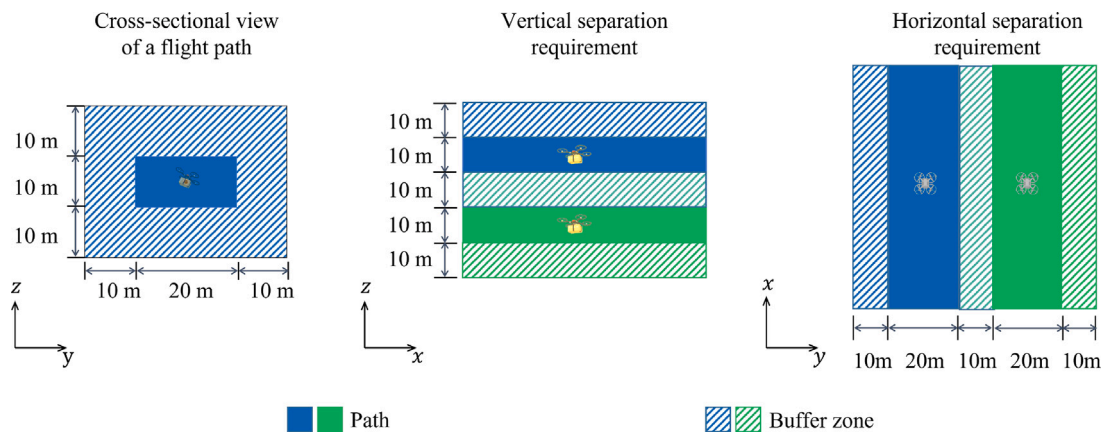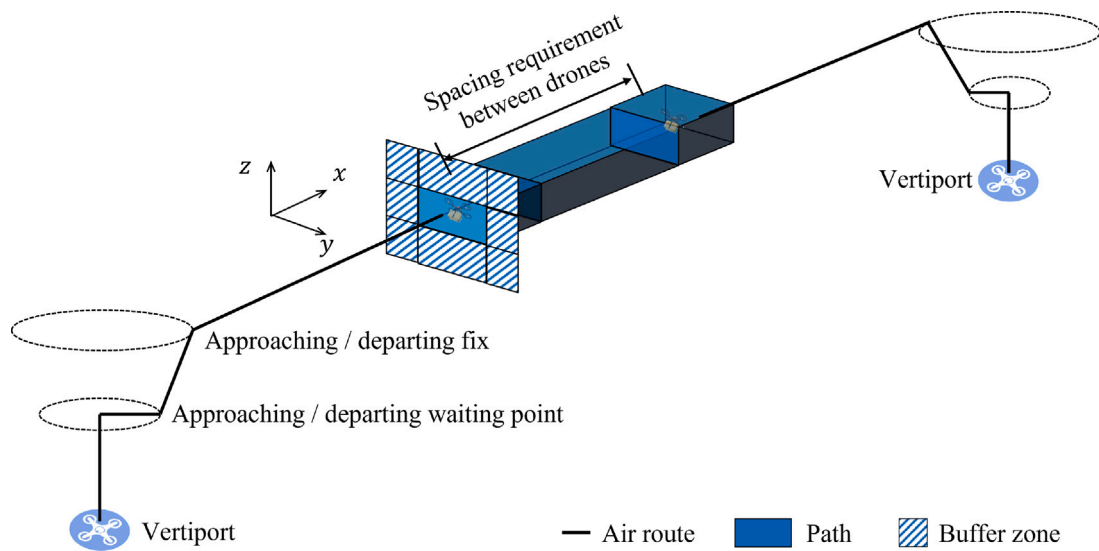
Search-based methods search over the graph to find feasible solutions. The solutions are often near-optimal and sometimes but they are not complete for many cases. Prioritized approaches are a kind of prominent search-based methods. They plan routes sequentially and treat positions of previously planned agents as moving obstacles to avoid any collision (Van Den Berg and Overmars, 2005). Hierarchical Cooperative A* (HCA*) (Silver, 2005) is a typical prioritized approach. HCA* plans one route for one agent at a time according to a predefined order, and it stores each route into a reservation table after the route is planned. The previous paths, i.e., entries in the reservation table, are impassable for later agents. Windowed HCA* (WHCA*) (Silver, 2005) runs in a similar way but it uses plan–move cycles to dynamically generate routes. In each planning phase, each agent reserves the next W steps by the order; in each moving phase, each agent follows the reserved paths by K ($K \leq W$) steps, then the next cycle starts in the current point. Conflict-oriented WHCA* (CO-WHCA*) (Bnaya and Felner, 2014) places the window, i.e., the reservation for the next W steps, only around conflicts. The choice of priorities has a great impact on the performance of planned routes (Warren, 1990) and there exist different strategies. Arbitrary order is applied in HCA* and the planning phase in each cycle in WHCA*. The decreasing order of the shortest path length for each agent is taken as the criterion in Van Den Berg and Overmars (2005). A winner-determination strategy is taken in CO-WHCA*, where all possible orders are estimated for every conflict and the best one is selected. Several search-based methods are bounded sub-optimal solvers (Barer et al., 2014; Cohen et al., 2019, 2016). Most of them are variants of conflict based search, they provide bounded sub-optimal solutions by relaxing the assumptions and conditions. In summary, prioritized approaches provide near-optimal solutions and they are still efficient when there are many agents. Bounded sub-optimal methods improve computational time compared to optimal methods, but they are still not efficient for a large number of agents.

Hybrid methods take both specific agent-movement rules and massive search steps. In the first phase, a path is planned for each agent individually with obstacle avoidance by using SPF algorithms while other agents' paths are ignored at this phase. In the next phase, the agents coordinate to ensure no inter-agent collision occur along the paths. Common coordination methods include modification of geometric paths, modification of velocity profiles, and delay in the startup time of agents (Kant and Zucker, 1986; Leroy et al., 1999; Li et al., 2005; O'Donnell and Lozano-Pérez, 1989; Peng and Akella, 2005; Saha and Isto, 2006; Sanchez and Latombe, 2002). These coordination schemes are the rules for solving conflicts. In summary, these methods can find paths fast if there are many agents, but most of them use time dimension to avoid collision and they cannot guarantee paths are spatial conflict-free.

In summary, none of the existing multi-path finding methods can readily solve the route network planning problem in this paper. The problem is NP-hard. The search space for the drone network design is large considering the size of the design space, cluttered obstacles, complexity of risk levels, and the coupled complexity of many routes to be planned. The optimal solvers are inefficient to solve the problem as it suffers the curse of dimensionality. Rule-based sub-optimal methods and hybrid sub-optimal methods are much efficient, but their solutions are often too far away from optimal. Also, they require to use the time dimension to solve conflicts. Spatial intersections may still exist in the generated network. Prioritized approaches are applicable and efficient, and their results are near-optimal. Therefore, following the prioritized approaches, we develop a set of prioritization rules and integrate them into a sequential route planning algorithm for drone route network planning in an urban environment in this paper.

3. Problem statement

This section defines the drone delivery network that we aim to design. An illustration of the network is shown in Fig. 2. Air routes are unidirectional paths established among approaching and departing fixes over Origin–Destination vertiports. Drones can fly sequentially in a path following the minimum spacing requirement. The width and height of a path are 20 m and 10 m, which is determined based on drone position uncertainties and measurement errors. There is a layer of airspace surrounding the path that serves as "buffer zones" with a width of 10 m, as shown in Fig. 3. No other path or obstacle is allowed in the buffer zone, and the vertical and horizontal separation requirements are shown in Figs. 3 and 4. However, the buffer zones of different paths can be overlapped as shown in Fig. 3.
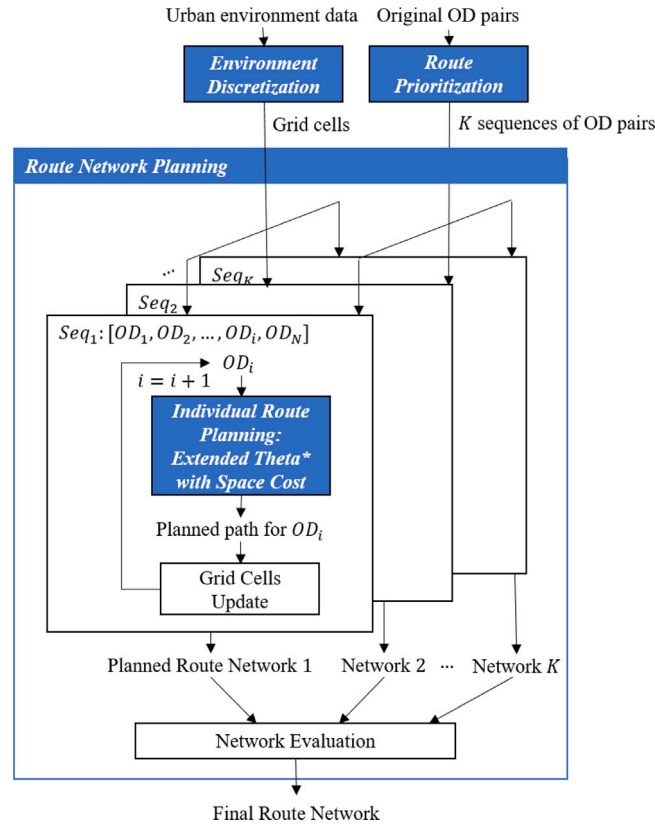
Fig. 2. Components of a route network.

Fig. 3. Dimension of a flight path and its separation requirement.

Fig. 4. Separation requirement between a path and an obstacle.

Fig. 5. Overall workflow of the proposed route network planning method.

For the design of a route network, individual paths are expected to expose minimum risks to the ground and to impose minimum energy consumption of drone operations. On a network level, the spread of the route network should be minimized. Besides, the computational time for generating a route network should also be short.

4. Methodology for route network planning

4.1. Overview

To design a route network for an urban area as described in Section 3, we develop a novel sequential route network planning method considering airspace utilization. In this method, we propose a prioritization structure to decouple the network planning problem into a set of single-path planning problems to generate paths sequentially. We set the prioritization rules to rank origin–destination (OD) pairs to ensure important routes with higher priority in planning for better system performance. To obtain better airspace utilization, we introduce a space cost function in the underlying path-finding algorithm to build a network with dense and aligned routes.

The proposed route network planning method is composed of four modules: Environment Discretization, Individual Route Planning, Route Prioritization, and Network Planning. The *Environment Discretization* module generates graphs for searching, the *Route Prioritization* module generates multiple ordered route sequences, and the *Network Planning* module generates route networks based on the graphs and the route sequences. In generating a route network, the *Individual Route Planning* module is iteratively conducted to generate a route network for each route sequence. The *Network Evaluation* module selects the route network that has a minimum cost from all generated route networks, then it checks the risk for each path in the selected route network. If all paths satisfy the risk requirement, the selected route network will be returned as the final route network; otherwise, the method fails to find a feasible route network. The overall workflow is illustrated in Fig. 5. The associated algorithm, named as *Sequential Extended Theta* with Route Prioritization and Space Cost*, is shown in Algorithm 1.
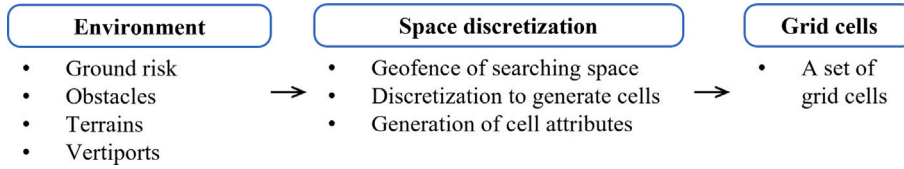
Environment	Space discretization	Grid cells
• Ground risk • Obstacles • Terrains • Vertiports	• Geofence of searching space • Discretization to generate cells • Generation of cell attributes	• A set of grid cells

Fig. 6. Flowchart of environment discretization.

Algorithm 1: Route Network Planning: Sequential Extended Theta* with Route Priority and Space Cost

---

Input : OD pairs $\{OD_i\}_{i=1}^N$, environment E

Output: A route network $\{R_i\}_{i=1}^N$

1 /* Environment Discretization */
2 Perform space discretization on E;
3 Generate grid cells to compose graph $G(V, E)$;
4 /* Route Prioritization */
5 Generate K OD pair sequences $\{Seq_k\}_{k=1}^K$ from original OD pairs $\{OD_i\}_{i=1}^N$;
6 /* Network Planning */
7 **for** Seq_k in $\{Seq_k\}_{k=1}^K$ **do**
8   **for** OD_i in Seq_k **do**
9   /*Individual planning*/
10   Run Extended Theta* with Space Cost on OD_i to get planned paths R_i^k;
11   /*Update grid cells*/
12   Set path cells for R_i^k as unreachable (impassable for obstacles and $R_{j>i}^k$);
13   Set buffer zone cells for R_i^k as reserved (impassable for obstacles and paths of $R_{j>i}^k$, passable for buffer zones of $R_{j>i}^k$);
14   **end**
15   Store the route network $\{R_i^k\}_{i=1}^N$;
16 **end**
17 /* Network Evaluation */
18 Find $\{R_i\}_{i=1}^N$ with lowest costs from K route networks $\{\{R_i^k\}_{i=1}^N\}_{k=1}^K$
19 Check risk for each path;
20 **if** *a subset of the paths do not satisfy the risk requirement* **then**
21   return "fail to generate a feasible route network"
22 **else**
23   Return the route network $\{R_i\}_{i=1}^N$
24 **end**

4.2. Environment discretization

This module aims to generate a grid graph for the network planning. It discretizes the environment into 3D cubic grid cells to compose a grid graph $G(V, E)$. The process is shown in Fig. 6.

The environment includes ground risk, obstacles, terrains, and vertiports. They are processed as different layers, as shown in Fig. 7. These layers are used to generate attribute values for each cell in the next discretization process. The risk layer specifies areas with high/low risks. It is calculated based on many factors, e.g., population density, sheltering factor, critical infrastructures, noise impact, public acceptance, etc. The calculation is outside the scope of this work. The obstacle/terrain layer includes obstacles like buildings, trees, no-fly zones, and terrains like flat ground and mountains. Drones should avoid any collision with obstacles and terrains. The vertiport layer provides taking off and landing points for drones.

The space discretization process first discretizes the environment into 3D cubic grid cells, then it determines the attributes of each cell based on environment layers. Each environment layer determines the value of an attribute of a cell. The vertiport layer determines whether a cell is an origin/destination vertiport or not, the obstacle/terrain layer determines whether a cell is passable or not, and the risk layer determines what is the ground risk level associated with each cell. These grid cells compose the grid-graph $G(V, E)$ for the following network planning.

4.3. Individual route planning: Extended Theta* with space costs

This module aims to find a conflict-free path for each OD pair that minimizes drones' energy consumption, the potential risk to ground, and airspace occupancy. The underlying algorithm for this module is referred to as *extended Theta* with space cost* in this paper. The pseudo-code of this algorithm is shown in Algorithm 2. The basic idea of this algorithm is explained below. The
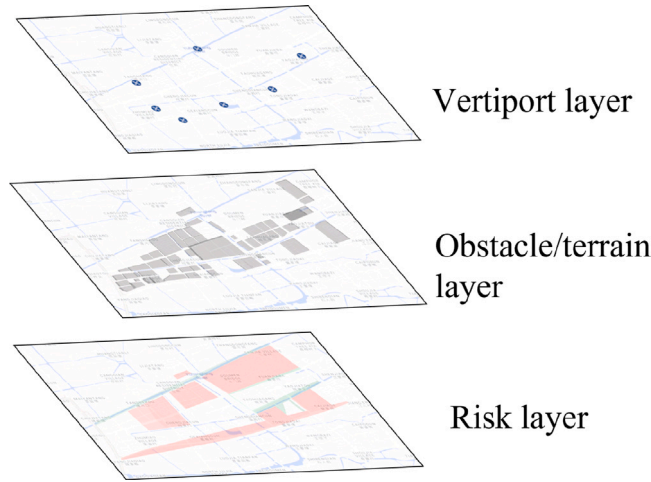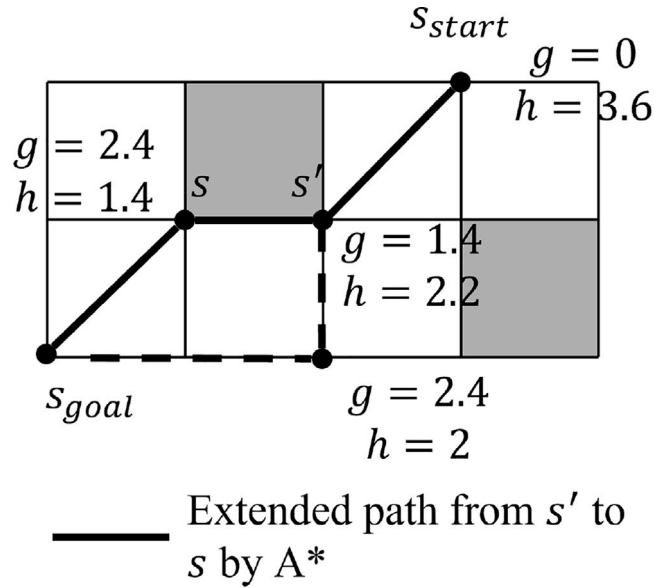
Fig. 7. Environment layers.

Fig. 8. Illustration of A* path finding.

proposed path planning algorithm, extended Theta* with space cost, is developed based on the most commonly used graph-based search algorithms, A* (Hart et al., 1968) and its variant Theta* (Daniel et al., 2010). In both A* and Theta*, the algorithm searches a path from a start node s_{start} to a destination node s_{goal} that has minimum total cost $g(s_{start}, s_{goal})$. In the path searching process, the algorithm starts from the start node s_{start}, iteratively explores a set of nodes and determine which one would be the best as the next node to generate a path in the end. Instead of searching and evaluating all nodes in the graph, a heuristic is used to guide the algorithm to extend nodes towards the destination. The heuristic is based on the cost required to extend the path all the way to the goal, which is estimated by the direct distance from the node to the goal. If a node has a higher heuristic cost, it will be less likely to be included in the path. Specifically, in the search process of a path, to extend a node s from a node s' towards the destination node, A* minimizes a cost function $f(s_{start}, s)$, which is based on the total cost of the path $g(s_{start}, s)$ and the heuristic $h(s, s_{goal})$, i.e.,

$$f(s_{start}, s) = g(s_{start}, s) + h(s, s_{goal}). \tag{1}$$

An illustration for the A* path finding is shown in Fig. 8.

In this paper, we modify the cost function g to design each route that minimizes drones' energy consumption, the potential risk to ground, and airspace occupancy, as shown in E.q. (2).

$$g(s_{start}, s_{goal}) = o(s_{start}, s_{goal}) + \omega_r r(s_{start}, s_{goal}) + \omega_p p(s_{start}, s_{goal}). \tag{2}$$

In this formula, $o(s_{start}, s_{goal})$, $r(s_{start}, s_{goal})$, and $p(s_{start}, s_{goal})$ are operational cost, risk cost, and space cost; ω_r and ω_p are weight coefficients. Aside from these cost functions, a few other operational constraints might affect the route network design, such as public acceptance, noise ordinances, air traffic restrictions, etc. These operational constraints can be considered using the cost factors in a similar way. ω_r and ω_p in E.q. (2) reflect the relative importance of risks and airspace utilization in comparison with drone flying time and energy consumption. The values can be determined by the relative monetary cost of the different categories. For example, the operational cost can be estimated by the time value of the drone delivery service and the cost of energy. The economic cost of a drone crash on ground can be estimated to gauge the risk weight value. The weight value of airspace utilization can be calculated based on the airspace usage charges or other urban airspace regulations to be established in the future.

The operational cost captures flying distance and the energy consumption for drone operations, including traversing a distance, turning, and climbing/descending. For a set of path segments $\{l_i\}_{i=1}^{n}$ that connects s_{start} and s_{goal}, the operational cost is calculated as

$$o(s_{start}, s_{goal}) = traversal(s_{start}, s_{goal}) + turning(s_{start}, s_{goal}) + climbing(s_{start}, s_{goal}) + descending(s_{start}, s_{goal}), \tag{3}$$

where

$$traversal(s_{start}, s_{goal}) = \sum_{i=1}^{n} \|l_i\|, \tag{4}$$

$$turning(s_{start}, s_{goal}) = \lambda_{turning} \sum_{i=1}^{n-1} \left| arccos(\frac{l_i \cdot l_{i+1}}{\|l_i\| \|l_{i+1}\|}) \right|, \tag{5}$$

let $\boldsymbol{n} = [0, 0, 1]^T$, then

$$climbing(s_{start}, s_{goal}) = \lambda_{climbing} \sum_{i=1}^{n} max(arcsin(\frac{l_i \cdot \boldsymbol{n}}{\|l_i\|}), 0) \|l_i\|, \tag{6}$$

$$descending(s_{start}, s_{goal}) = \lambda_{descending} \sum_{i=1}^{n} max(-arcsin(\frac{l_i \cdot \boldsymbol{n}}{\|l_i\|}), 0) \|l_i\|. \tag{7}$$

Here $\lambda_{turning}, \lambda_{climbing}, \lambda_{descending}$ are coefficients to normalize energy consumption for different drone operations.

The risk cost captures the potential risk to the ground, it reflects various risk levels by accumulating the risks involved in passing through the cells. It is calculated as

$$r(s_{start}, s_{goal}) = \lambda_r \sum_{s_{start}}^{s_{goal}} \theta_{risk}, \tag{8}$$

where θ_{risk} indicates the risk level of an area. Areas that are densely populated or with critical infrastructures have higher risk levels, with $\theta_{risk} > 1$; areas that are not populated and with less ground impact concerns have lower risk levels, with $0 < \theta_{risk} < 1$; most areas are set to have a normal risk level, with $\theta_{risk} = 1$. λ_r is a scaling factor for raw risk cost, the calculation of it is in Appendix A.

The space cost function encourages bundled paths and overlapped buffer zones to improve airspace utilization. An illustration is shown in Fig. 9. The size of buffer zones in Figs. 9(a) and 9(b) are the same and equal to the required minimum separation. In Fig. 9(a), the buffer zones of two aligned paths are not overlapped and the separation between the paths is twice the required minimum separation; in Fig. 9(b), the buffer zones are overlapped and the separation between the paths is exactly the required minimum separation. For a path between cell s_{start} and cell s_{goal}, the space cost item $p(s_{start}, s_{goal})$ measures the marginal volume of occupied airspace, i.e.,

$$p(s_{start}, s_{goal}) = \lambda_p N(s_{start}, s_{goal}), \tag{9}$$

here λ_p is a scaling factor for raw space cost, the calculation of it is in Appendix A, $N(s_{start}, s_{goal})$ is calculated by

$$N(s_{start}, s_{goal}) = N_{path}(s_{start}, s_{goal}) + N_{new\_buf}(s_{start}, s_{goal}), \tag{10}$$

where N_{path} is the increased number of path cells by adding a path between cell s_{start} and cell s_{goal} to the existing network, while N_{new_buf} is the increased number of buffer zone cells by adding a path between cell s_{start} and cell s_{goal} to the existing network. Taking Fig. 9 as an example, buffer zones in Fig. 9(a) do not overlap, so all buffer zone cells are newly introduced. For the situation in Fig. 9(b), when planning a path r_2 after path r_1 has been planned, some of r_2's buffer zone cells overlapped with the buffer zones of r_1. These buffer zone cells will not be counted again in N_{new_buf} for r_2. After the addition of the space cost function, the total cost reduces but the length of paths increases, but this is not an issue. As a drone delivery operator, the cost is associated with the direct drone operations as well as the airspace usage charges, similar to the airline operations.

To ensure the effectiveness of the space cost function for the purpose of reducing total costs, the relative value of the space cost weight coefficient ω_p in relation to other cost coefficients needs to be carefully calibrated to reflect the actual operational cost impact of airspace usage charges. For example, when the space cost weight coefficient ω_p is small, the space cost has little impact on route density. As ω_p increases, the algorithm starts to reduce large space costs at the expense of a small increase in other costs.
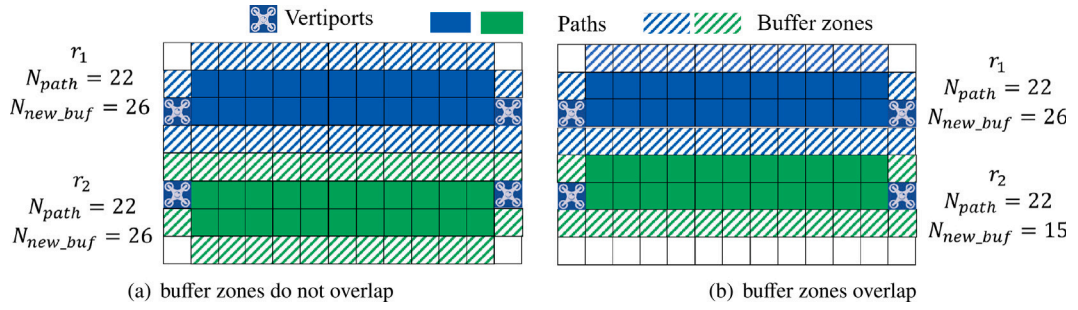
Fig. 9. An illustration for overlapping buffer zones (top view).

When ω_p is large, the algorithm generates a network where paths are heavily bundled to have more overlapped buffer zones, but the paths may pass through areas with high operational cost and risk cost.

Algorithm 2: Extended Theta* with Space Cost

```
 1 Main():
 2     open := closed := ∅
 3     g(s_start, s_start) := 0
 4     parent(s_start) := s_start
 5     open.Insert(s_start, g(s_start,
 6     s_start) + h(s_start, s_goal))
 7     while open ≠ ∅ do
 8         s' := open.Pop()
 9         [SetVertex(s')]
10         if s' = s_goal then
11             return "path found"
12         end
13         closed := closed ∪ {s'}
14         foreach s ∈ nghbr_vis(s') do
15             if s ∉ closed then
16                 if s ∉ open then
17                     g(s_start, s) := ∞
18                     parent(s) := NULL
19                 end
20                 UpdateVertex(s', s)
21             end
22         end
23     end
24     return "no path found"
25 UpdateVertex(s', s):
26     g_old := g(s_start, s)
27     ComputeCost(s', s)
28     if g(s_start, s) < g_old then
29         if s ∈ open then
30             open.Remove(s)
31         end
32         open.Insert(s, g(s_start, s) + h(s, s_goal))
33     end
```

```
34 SpaceCost(s', s):
35     /* calculate space cost
36     traveling from s' to s */
37     N(s', s) = N_path(s', s) + N_new_pro(s', s)
38     p(s', s) = λ_p N(s', s)
39     return p(s', s)
40 ComputeCost(s', s):
41     if lineofsight(parent(s'), s) then
42         /* Path 2 */
43         p(parent(s'), s) = SpaceCost(parent(s'), s)
44         g(parent(s'), s) =
                o(parent(s'), s) + ω_r r(parent(s'), s) + ω_p p(parent(s'), s)
45         if g(s_start, parent(s')) + g(parent(s'), s) < g(s_start, s) then
46             parent(s) := parent(s')
47             g(s_start, s) := g(s_start, parent(s'))
48             +g(parent(s'), s)
49         end
50     else
51         /* Path 1 */
52         p(s', s) = SpaceCost(s', s)
53         g(s', s) = o(s', s) + ω_r r(s', s) + ω_p p(s', s)
54         if g(s_start, s') + g(s', s) < g(s_start, s) then
55             parent(s) := s'
56             g(s_start, s) := g(s_start, s') + g(s', s)
57         end
58     end
```

The proposed space cost function is highlighted as red in the Algorithm.

4.4. Route prioritization and network planning

This module specifies the prioritization framework based on a set of prioritization rules to rank the importance of routes and decouple the network planning problem into sequential planning. As discussed in the introduction, the network-planning problem is an NP-hard problem. An optimal solution of a single path occupies certain airspace, which may force other paths to detour. Thus,
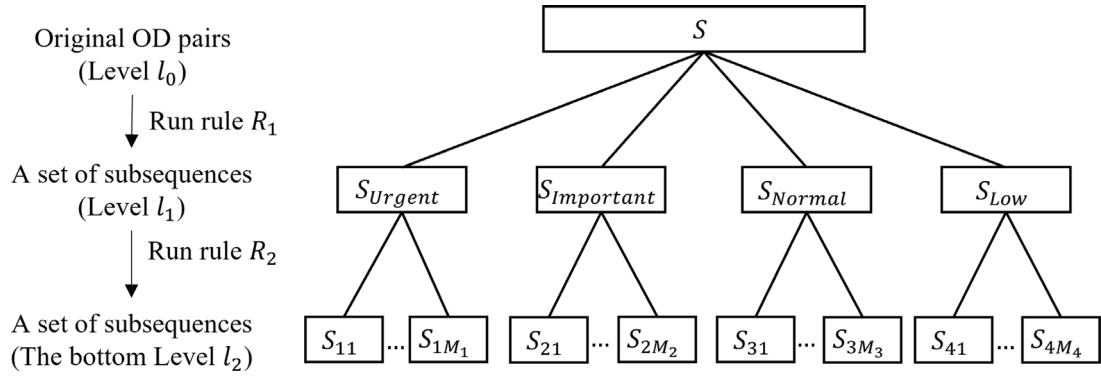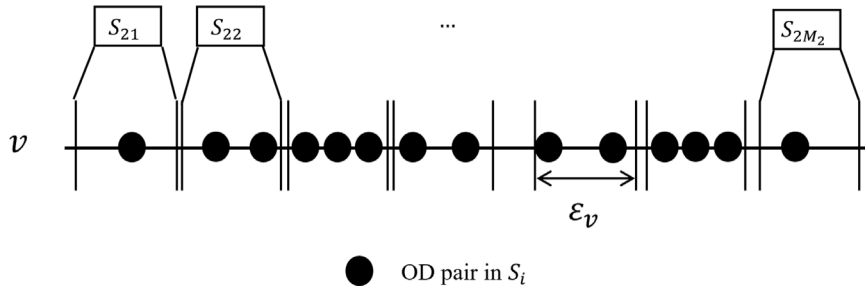
Original OD pairs
(Level l_0)

Run rule R_1

A set of subsequences
(Level l_1)

Run rule R_2

A set of subsequences
(The bottom Level l_2)

Fig. 10. Illustration of the prioritization framework.

Fig. 11. Illustration of the segmentation and shuffle in R_2.

a change of one path affects all other paths. In this paper, we use a simple but effective strategy to decouple the network planning problem into a set of single-path planning problems — plan the paths one by one from the most important to the least important.

In this module, a prioritization structure with example rules is proposed. These rules can be changed depending on specific business considerations. The prioritization structure is shown in Fig. 10. There can be multiple levels to prioritize the paths. Level l_0 has the sequence of original OD pairs and it is grouped into a series of subsequences by the most important priority rule, R_1. Then the subsequences on Level l_1 can be further grouped into subsequences by the second important priority rule, R_2. The paths in the subsequences on the bottom level are randomly shuffled to find the optimal sequence that generates the best network performance.

To give a more detailed example, suppose we have two priority rules as specified below:

$$R_1 = \{PL\}, \tag{11}$$

$$R_2 = \{v, \epsilon_v\}, \tag{12}$$

where PL denotes the urgency level of paths, detailed as $\{Urgent, Important, Normal, Low\}$; v is the expected profit value of each path and ϵ_v is segment threshold of v. More priority rules can be taken to get more subsequences.

R_1 generates four subsequences $\{S_i\}_{i=1}^4$ directly based on urgency levels in PL. These subsequences are (S_{Urgent}, $S_{Important}$, S_{Normal}, S_{Low}), each of them represents an urgency level. R_2 sorts paths within each subsequence S_i in l_1 by the expected profit value v (in descending order), and generates a new set of subsequences $\{S_{ij}\}_{j=1}^{M_i}$, where M_i is the number of subsequences in S_i. ϵ_v divides each S_i into a set of subsequences $\{S_{ij}\}_{j=1}^{M_i}$ based on the expected profit value distribution (Fig. 11), similar to generating clusters based on density in DBSCAN (Ester et al., 1996). Given a set of paths with different v, OD pairs are grouped together with similar v values. Each S_{ij} satisfies the following property:

$$|v_r - v_{r'}| \le \epsilon_v, \forall r, r' \in S_{ij}. \tag{13}$$

Paths in the subsequences on the bottom level are randomly shuffled to find the optimal sequence that generates the best network performance.

To balance between optimality and computational time, we randomly generate K sequences following the above priority structure, where K is a parameter that one can adjust. It can be adjusted in the range $[1, S_g]$,

$$K \in \mathbb{Z} : 1 \le K \le S_g. \tag{14}$$

$$S_g = \prod_{i=1,2,3,4} \prod_{j=1,2,\ldots,M_i} n(S_{ij})!, \tag{15}$$

where $n(S_{ij})$ is the number of OD pairs in sub-sequence S_{ij}, $n(S_{ij})!$ is the number of permutations in sub-sequence S_{ij}, and S_g is the number of all possible ordered arrangements of OD pairs satisfying R_1 and R_2.

The complete strategy for determining route sequence order is shown in Algorithm 3. If there are two networks that generate the same performance in terms of the total costs, the algorithm will generate both networks and their associated costs in different categories. It is up to the user to select which one to use based on the information of different cost categories and other considerations not captured in the model.

Algorithm 3: Route Prioritization

    Input   : Original OD pairs $\{OD_i\}_{i=1}^N$

    Output : Ordered OD pairs $\{OD_i'\}_{i=1}^N$

1 /* Rule 1: get $\{S_i\}_{i=1}^4$, i.e., $(S_{Urgent}, S_{Important}, S_{Normal}, S_{Low})$ */

2 $\{S_i\}_{i=1}^4 \leftarrow \emptyset$

3 **for** r *in* $\{OD_i\}_{i=1}^N$ **do**

4 | $S_l.append(r)$ if $r.PL == l$

5 **end**

6 $\{S_i\}_{i=1}^4 \leftarrow \{S_l\}$

7

8 /* Rule 2: get $\{\{S_{ij}\}_{j=1}^{M_i}\}_{i=1}^4$ */

9 **for** S_i *in* $\{S_i\}_{i=1}^4$ **do**

10 | Sort S_i in descending order of v;

11 | $\{S_{ij}\}_{j=1}^{M_i} \leftarrow$ subgroups from DBSCAN with ϵ_v in S_i;

12 | **for** S_{ij} *in* $\{S_{ij}\}_{k=1}^{M_i}$ **do**

13 | | $S_{ij} \leftarrow$ random shuffled S_{ij};

14 | **end**

15 **end**

16

17 $\{OD_i'\}_{i=1}^N \longleftarrow \{\{S_{ij}\}_{j=1}^{M_i}\}_{i=1}^4$;

18 return $\{OD_i'\}_{i=1}^N$

---

5. Testing

In this section, we first use an illustrative scenario to show how the algorithm works. Then we compare the proposed algorithm with other path finding algorithms in a test scenario. After that, the proposed algorithm is applied to a real-world scenario in Hangzhou, China. In this real-world scenario, we show a set of sensitivity analyses on algorithm parameters and provide an empirical analysis of computational time.

5.1. Illustration with a toy example

In this section, we use a toy example to show how the space cost function improves airspace utilization for the network design. We apply the *our method with space cost* algorithm and the *our method without space cost* algorithm on the toy example shown in Fig. 12. In this example, there are two symmetric obstacles and 10 vertiports (id 1–10).

In Experiment 1, both algorithms are applied to find three paths from upper vertiports to lower vertiports (id 3–8, 2–7, 4–9). The results are shown in Fig. 12. If the space cost function is not added to the algorithm, three paths (①, ②, ③) will appear on the different sides of the obstacles to achieve a shorter distance, as shown in Fig. 12(a); if the space cost function is added to the algorithm, three paths all appear in the middle of two obstacles and share the buffer zones, as shown in Fig. 12(b). After adding the space cost function into the algorithm, the generated network occupies fewer buffer zones (16.7%) at the cost of slightly more path cells (4.8%), and the total occupied airspace reduces (6.7%), as shown in Table 1.

In Experiment 2, both algorithms are applied to find five paths from upper vertiports to lower vertiports (id 3–8, 2–7, 4–9, 1–6, 5–10). The results are shown in Fig. 13. When not adding the space cost function, the method fails to find the paths ④ and ⑤ because their traversable airspaces are blocked by the buffer zones of the paths ② and ③. By contrast, when adding the space cost function, the method successfully generates the extra paths (④, ⑤).
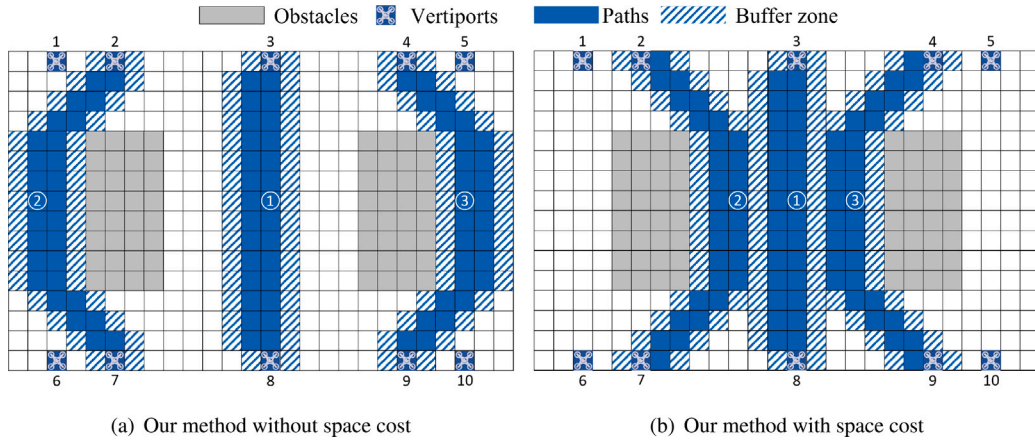
(a) Our method without space cost   (b) Our method with space cost

Fig. 12. A test case (top view) for network planning (a) without space cost and (b) with space cost.

Table 1
Summary of planning three paths in the scenario.

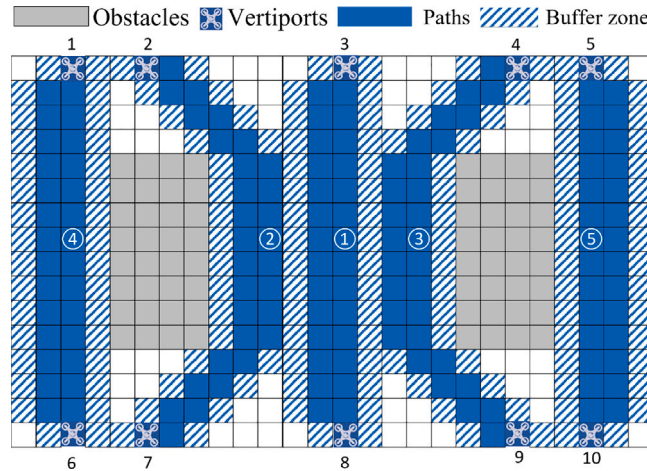|	# of buffer zone cells	# of path cells	Total occupied airspace
Our method without space cost	96	84	180
Our method with space cost	80	88	168
Improvement after adding space cost	16.7%	−4.8%	6.7%

Fig. 13. A test case (top view) of network planning with 5 paths.

5.2. Comparisons with other algorithms on a 2D scenario

In this section, we compare the proposed sequential route network planning method with two kinds of state-of-the-art MPF algorithms on a 2D standard test scenario: one is a two-level-based optimal method — conflict-based search (CBS) (Sharon et al., 2015), and one is a rule-based sub-optimal method — Push-and-Spin (Alotaibi and Al-Rawi, 2018). The scenario is shown in Fig. 14. This scenario (Sturtevant, 2012) contains 65 536 pixels/cells, and 6528 cells are inaccessible. Multiple OD pairs need to be planned in this experiment. The origins are located in the lower-left corner and the destinations are located in the upper-right corner.

Results show that the proposed method is capable to solve large scenarios within a reasonable amount of time. The comparisons are shown in Table 2. Both Push-and-Spin and the proposed method find paths quickly, while CBS cannot find paths in a reasonable time as the number of routes increases. The proposed method also shows an advantage in airspace utilization compared to CBS and Push-and-Spin. The paths generated by these algorithms are shown in Fig. 15.
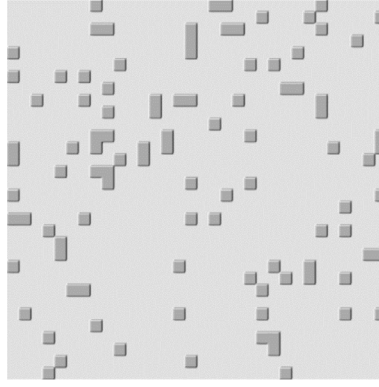
Fig. 14. The 2D standard test scenario (Sturtevant, 2012).

Table 2
Route network planning results of three methods in the 2D standard test scenario.

# of routes	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
CBS																				
Sum of distance	245	477	703	956	1203	1443	1694	1965	2263	–	–	–	–	–	–	–	–	–	–	–
# of occupied cells	1007	1923	2910	3965	4894	5918	6902	7993	9189	–	–	–	–	–	–	–	–	–	–	–
Time (s)	0.02	0.03	1.98	134.04	1201.34	2029.10	2378.24	2836.76	3165.76	–	–	–	–	–	–	–	–	–	–	–
Push-and-Spin																				
Sum of distance	245	477	707	966	1203	1466	#	#	#	#	#	#	#	#	#	#	#	#	#	#
# of occupied cells	1007	1923	2852	3904	4894	5877	#	#	#	#	#	#	#	#	#	#	#	#	#	#
Time (s)	0.02	0.03	0.04	0.04	0.05	0.06	#	#	#	#	#	#	#	#	#	#	#	#	#	#
The proposed method																				
Sum of distance	247	486	715	969	1226	1495	1734	1999	2266	2564	2893	3175	3490	3807	4087	4367	4666	4968	5316	5664
# of occupied cells	881	1491	2019	2771	3458	4121	4915	5699	6623	7400	8232	8949	9713	10 522	11 170	11 867	12 623	13 349	14 229	15 311
Time (s)	0.16	0.21	0.26	0.32	0.42	0.52	0.58	0.67	0.77	0.85	0.92	0.97	1.03	1.09	1.12	1.16	1.21	1.25	1.29	1.33

–: exceed the timeout limit (3600 s).
#: unable to find feasible solutions.

5.3. Demonstration on a real-world scenario

In this section, we use a real-world scenario to demonstrate the capability of the proposed method. The selected environment is a typical urban area in Hangzhou, China. Hangzhou has 10,711,198 residents (CITYPOPULATION, 2020) with a size of 8292.31 km^2 for urban districts. Hangzhou has logged more than 3 billion express parcels (HangzhouPostal, 2020) in 2020. Drone delivery services have been offered by Antwork Technology in Hangzhou since 2020. The size of the selected area is 5.35 km × 2.95 km (about 15.75 km^2) and the details of the scenario data are shown in Fig. 16 and Table 6. Drones are allowed to fly in the altitude range [60 m, 120 m]. A graph is extracted from the scenario data using grid size (10 m, 10 m, 10 m). The details of the graph are shown in Table 7. In the flyable altitude range, there is a total of 946 950 cells and 707 128 cells are traversable.

In this urban area, 12 routes are planned using our proposed algorithm with the following parameter settings: $\varepsilon_v = 1000, \omega_r = \omega_p = 1.0$. The planned route network is shown in Fig. 17, where paths are colored in blue. A few routes are aligned together with overlapped buffer zones, showing improved airspace utilization. Many paths are over green areas, which are preferred areas for drone operations with low risk. Only a few paths are over red areas, which have high risk, as the paths over normal regions are blocked by buildings. With an separate validity check, we confirm that the route network generated by the proposed method satisfies all operational constraints and can be used for drone delivery services.
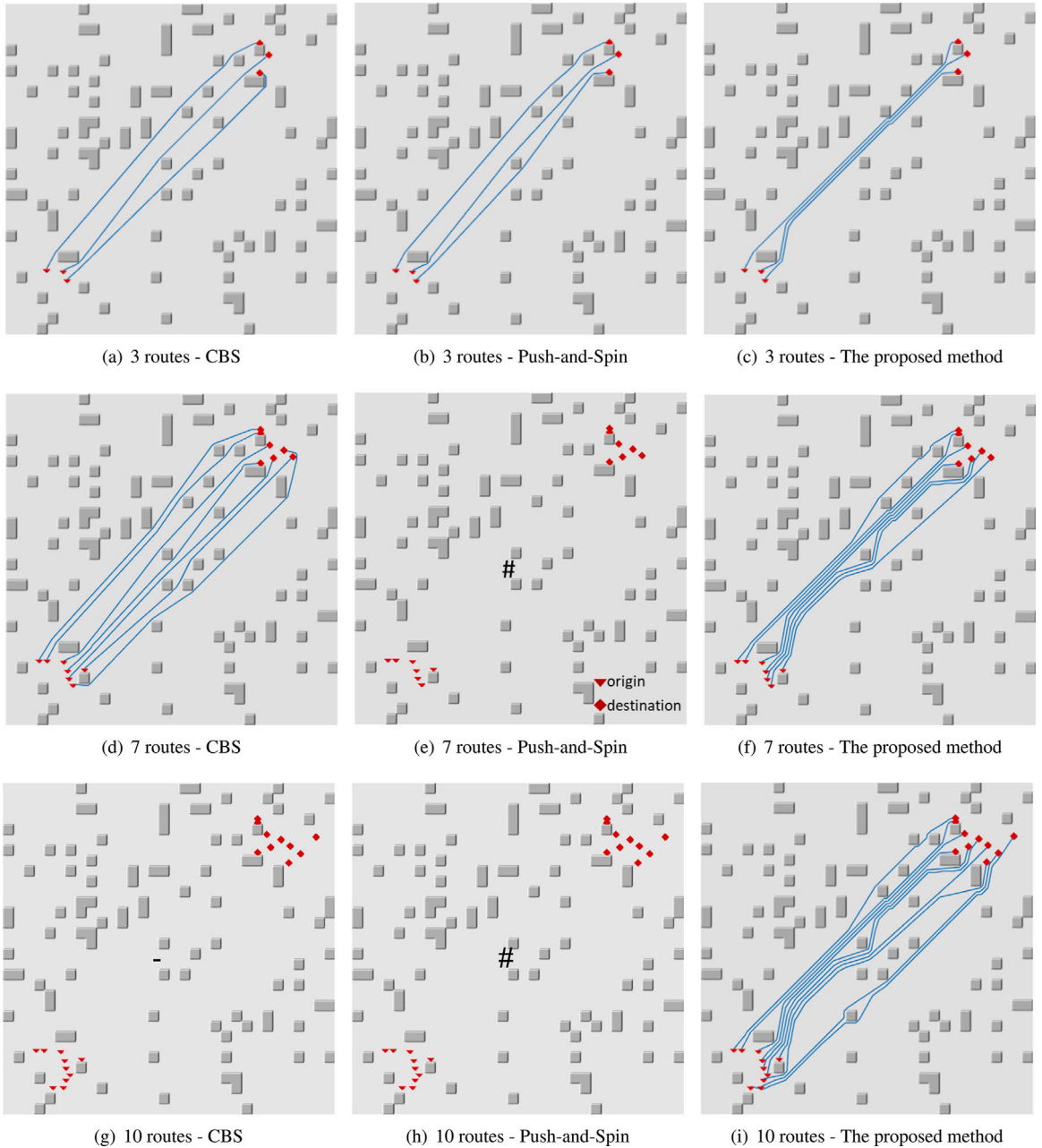
(a) 3 routes - CBS

(b) 3 routes - Push-and-Spin

(c) 3 routes - The proposed method

(d) 7 routes - CBS

(e) 7 routes - Push-and-Spin

(f) 7 routes - The proposed method

(g) 10 routes - CBS

(h) 10 routes - Push-and-Spin

(i) 10 routes - The proposed method

**Fig. 15.** Routes planned by CBS, Push-and-Spin, and the proposed method in the 2D standard test scenario.

5.4. Sensitivity analysis of algorithm parameters

### 5.4.1. Sensitivity analysis on space cost

In this section, we analyze the effectiveness of adding the space cost into the algorithm by testing different values of the space cost weight coefficient ω_p in solving the network planning problem in the real-world 3D scenario in Section 5.3.

We first compare the routes found without space cost and the ones found with space cost. The results are shown in Table 3. Compared to routes found without space cost, the number of buffer zone cells decreased from 17.26 ‰ to 13.87 ‰, and the number of total occupied cells reduced from 24.14 ‰ to 21.51 ‰. Therefore, the proposed method with space cost can generate a route network that utilizes airspace more effectively than without space cost.
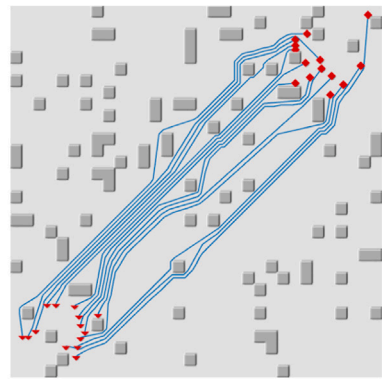
Then we test how ω_p affects the planning of a route network. The results are shown in Fig. 18. As ω_p increases, the number of buffer zone cells and the number of total occupied cells decrease; meanwhile, the number of path cells increases slightly, and the
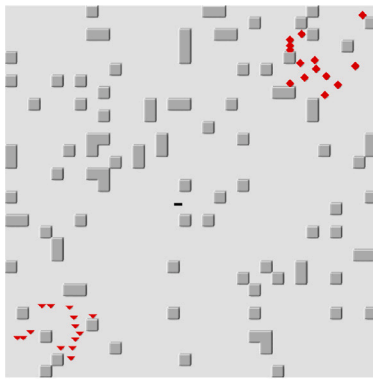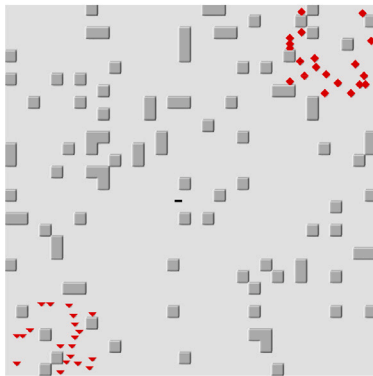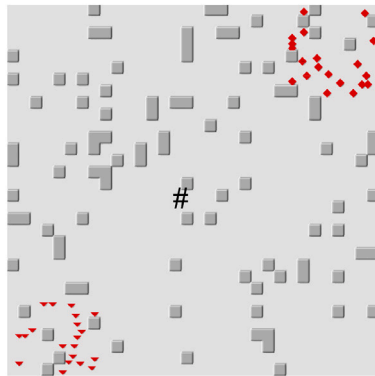
(j) 15 routes - CBS
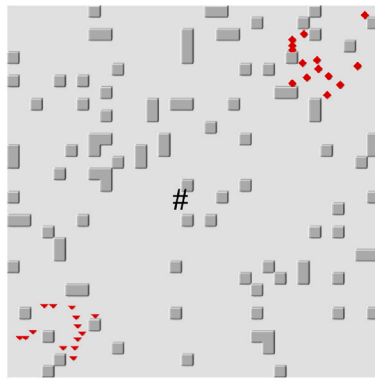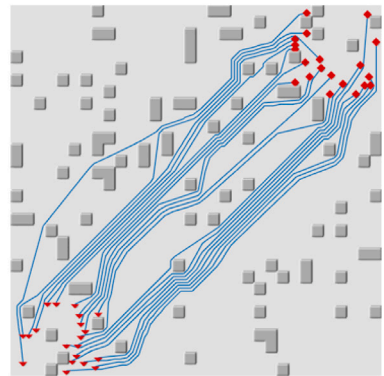
(k) 15 routes - Push-and-Spin

(l) 15 routes - The proposed method

(m) 20 routes - CBS

(n) 20 routes - Push-and-Spin

(o) 20 routes - The proposed method
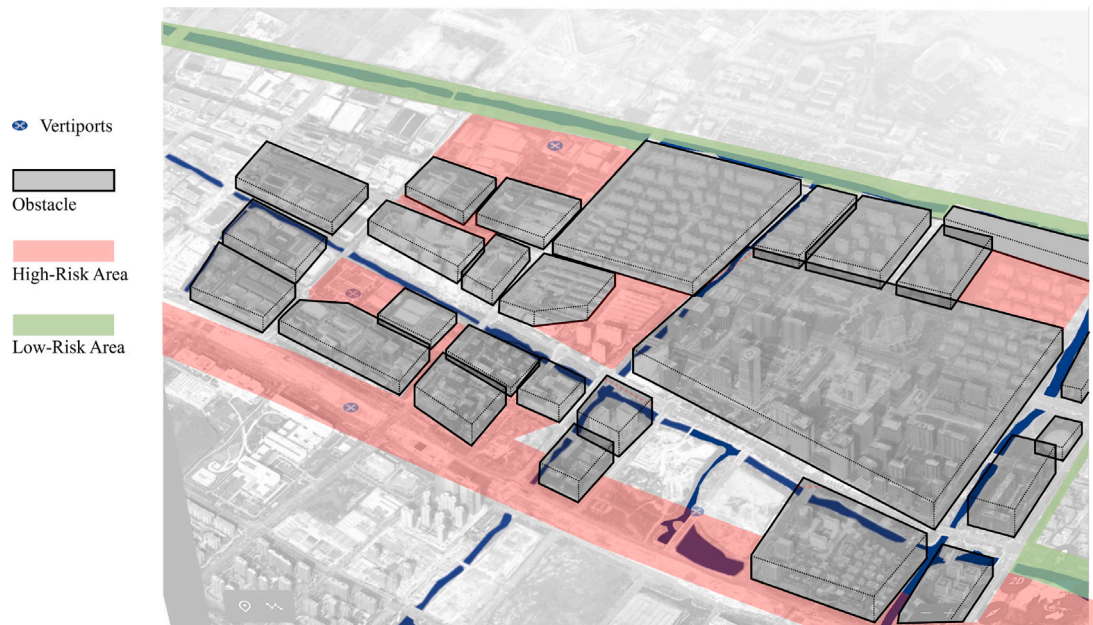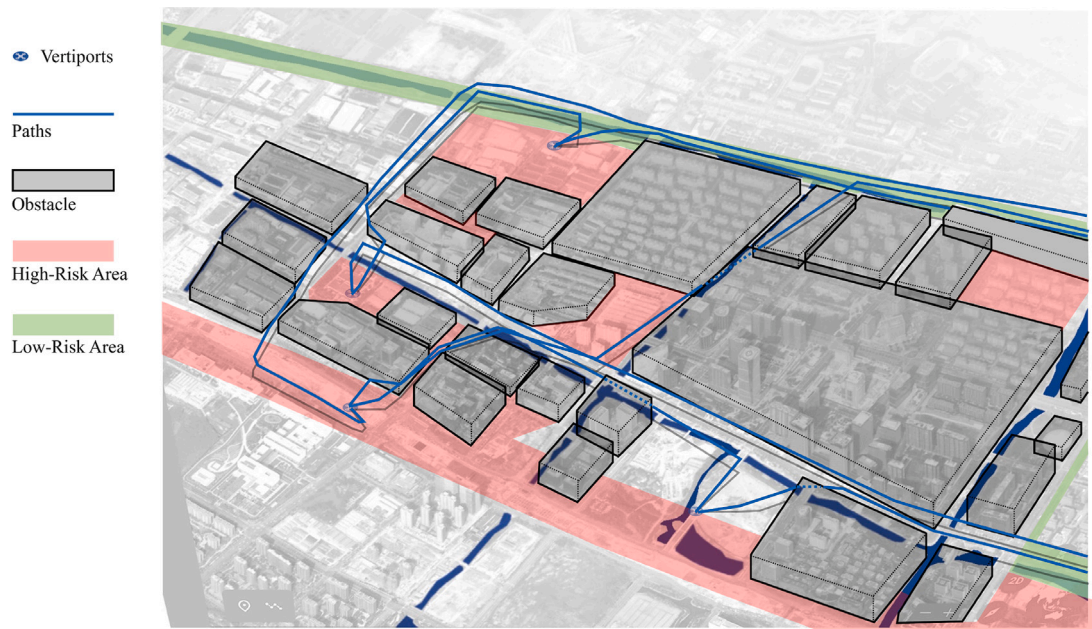
Fig. 15. (*continued*).

Fig. 16. A real-world scenario in Hangzhou, China.

(a) Perspective view of all planned paths
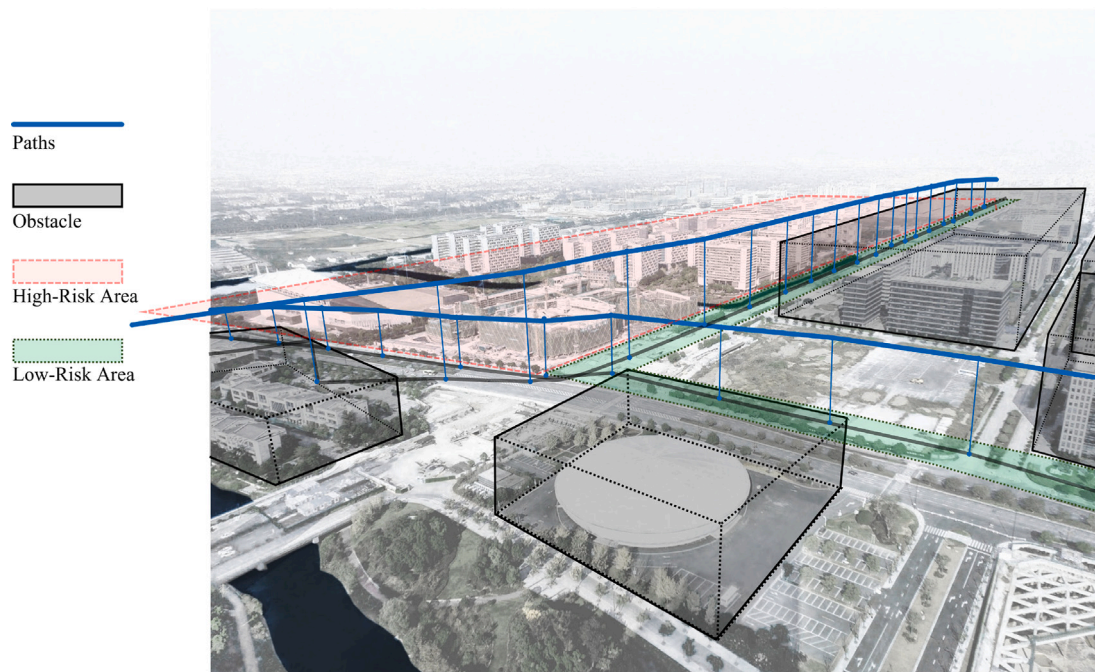
(b) A zoom-in view of the planned paths in relation to obstacles, high risk areas, and low risk areas

Fig. 17. A set of paths planned by the proposed method (3D). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

total cost of risk 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.

Table 3
Airspace occupancy for routes found without space cost vs. with space cost[a].

	Buffer zone cells	Path cells	Total occupied cells
Without space cost ($\omega_p = 0$)	17.26‰	6.88‰	24.14‰
Add space cost item ($\omega_p = 1.0$)	13.87‰	6.89‰	21.51‰

[a]Results are presented as the number of cells divides the total number of cells available for drones (707 128).

(a) Total occupied airspace (buffer zone vs. path)

(b) Total occupied airspace vs. total risks
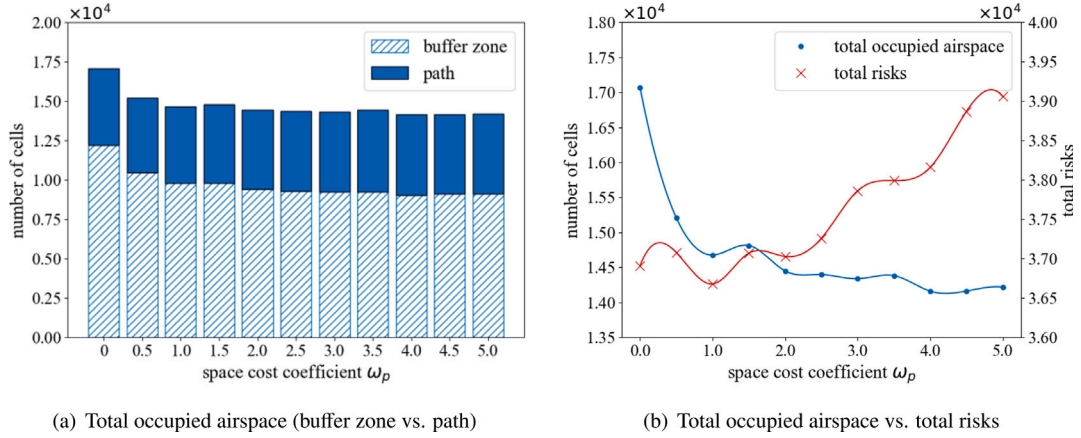
Fig. 18. Sensitivity analysis on space cost coefficient ω_p.

Table 4
Comparison between no route prioritization and with route prioritization.

Average total cost after running 10 times[a]	Urgent	Important	Normal	Low	Total
No route prioritization	16 030	4270	5572	11 429	37 301
With route prioritization[b]	15 198	4091	6091	12 331	37 711
Improvement after route prioritization	832	179	−519	−901	−410

[a]Sequences without prioritization vary, the average cost is used.
[b]Thresholds $\varepsilon_v = 1000$, the range of profit v is [−219, 9765].

5.4.2. Sensitivity analysis on route prioritization threshold

In this section, we first test the route network planning method with and without Route Prioritization module. Then we analyze how different values of the Route Prioritization threshold affect the results.

The result of comparison between no route prioritization and with route prioritization is shown in Table 4. With route prioritization, routes in Urgent and Important levels have smaller costs, but routes in Normal and Low levels have larger costs. The increase in normal and low levels is expected because route prioritization improves the quality of high-priority routes at the cost of low-priority routes. There is minimal impact on the total cost of the entire network from using route prioritization.

For the sensitivity analysis on threshold values, we take ε_v as an example to show how threshold values affect the route prioritization and the final result. Here we use 16 OD pairs, and the potential profit values of these OD pairs are derived from a normal distribution $v \sim \mathcal{N}(5000, 2000)$. We test the threshold values $\varepsilon_v \in \{100, 400, 800\}$. As shown by the results in Table 5, with the increase of ε_v, OD pairs are divided into fewer but larger subsequences, making the total number of possible ordered arrangements S_g larger, and the minimum total costs for the networks decrease. This is because with more permutation of OD sequences being tested, the results solution gets closer to the optimal one. However, it is at the cost of increased computational time.

5.5. Algorithm scalability and computational time

In this section, we first show how the computational time of the algorithm is affected by the random shuffle times K, and then we show how the computational time of the algorithm increases as the number of routes increases. The test scenario is the urban area in Hangzhou as shown in Section 5.3. All experiments are performed on a platform Intel(R) Xeon(R) Gold 5218 CPU @ 2.30 GHz.

The computational times for different K are shown in Fig. 19. As K increases, the computational time increases proportionally. This means that the total computational time is proportional to the number of OD pair sequences to run. The computational times for different number of routes to plan are shown in Fig. 20 and the result data are summarized in Table 8 in Appendix. When K is fixed, with the increase of the number of routes N, the computational time increases near linearly. The empirical results show that the proposed method is able to handle the planning of 40 routes within about 1 h for a real-world scenario.

Table 5
Sensitivity analysis on threshold of expected profit values ε_v.

OD pairs sorted by expected profit value v							
Route i	r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8
v	9481	8735	7988	7908	6957	6900	6522	5821
Route i	r_9	r_{10}	r_{11}	r_{12}	r_{13}	r_{14}	r_{15}	r_{16}
v	5800	5667	5626	5423	4793	4697	3045	−105

$\varepsilon_v = 100$
Best total costs: 35 098 — Shuffled times: $K = 10$ — Total randomness: $S_g = 32$
Subsequences: $[(r_1),(r_2),(r_3,r_4),(r_5,r_6),(r_7),(r_8,r_9),(r_{10},r_{11}),(r_{12}),(r_{13},r_{14}),(r_{15}),(r_{16})]$

$\varepsilon_v = 400$
Best total costs: 33 810 — Shuffled times: $K = 100$ — Total randomness: $S_g = 960$
Subsequences: $[(r_1),(r_2),(r_3,r_4),(r_5,r_6),(r_7),(r_8,r_9,r_{10},r_{11},r_{12}),(r_{13},r_{14}),(r_{15}),(r_{16})]$

$\varepsilon_v = 800$
Best total costs: 33 659 — Shuffled times: $K = 200$ — Total randomness: $S_g = 1440$
Subsequences: $[(r_1,r_2),(r_3,r_4),(r_5,r_6,r_7),(r_8,r_9,r_{10},r_{11},r_{12}),(r_{13},r_{14}),(r_{15}),(r_{16})]$
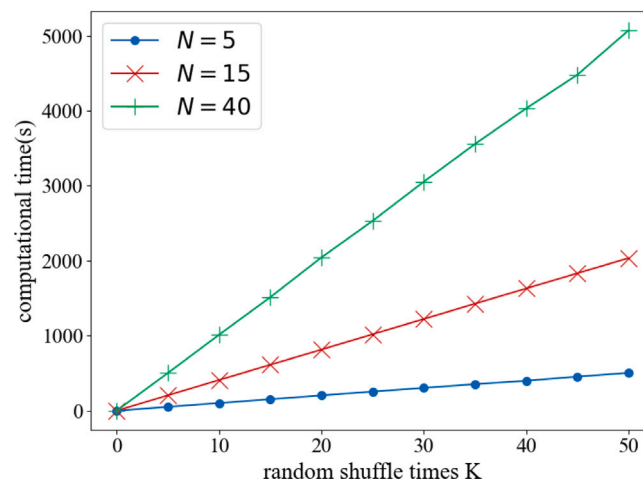
Fig. 19. Sensitivity analysis on random shuffle times K for different number of routes N.
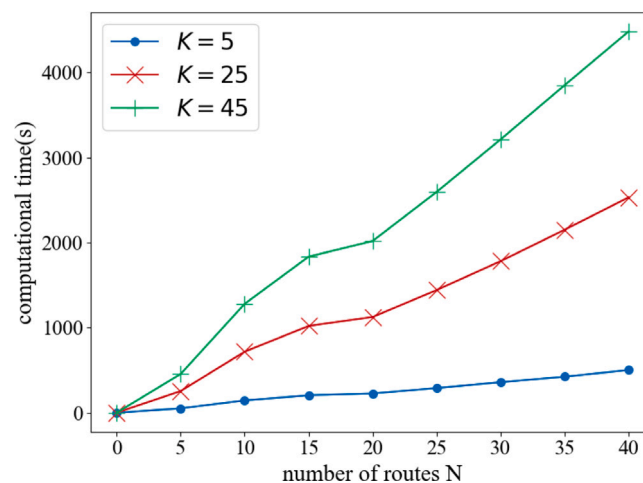
Fig. 20. Algorithm scalability as the number of routes increases.

Antwork Technology's current drone delivery network operates up to 80–100 flights per day on 50 unidirectional air routes. The length of the air routes is about 10 km on average, and the longest one is more than 30 km. These air routes are manually designed at Antwork Technology. To design a route, a field study needs to be conducted first to obtain 3D modeling of the environment. Based on the detailed 3D model, an air route is manually charted and then checked to see whether it satisfies all operational requirements.

Table 6
Description for real-world scenario data.

Types	Number	Attributes
Vertiports	7	*Id, type, radius, {latitude, longitude, altitude}*
Obstacles	55	*Id, highest/lowest altitude, points (latitudes, longitudes)*
High/low risk area	22	*Id, risk level, points (latitudes, longitudes)*

It usually takes 2 to 4 h to design one air route. Moreover, it becomes infeasible to design a network with a large number of air routes once the complexity exceeds human operators' capability.

The proposed method is expected to significantly improve the design process of the air route network by making it automatic. With the proposed method, the design of a network with 40 air routes within 1 h. Also, the scale of the network will no longer be limited by human operators' capability of dealing with the computational complexity. Using the proposed method, more routes can be designed until the airspace cannot accommodate any additional air routes.

6. Conclusion

This paper proposed a sequential route network planning method to support tube-based operations of drone deliveries. The proposed method is capable of designing a spatially separated tube-based route network in a large urban area within a reasonable time. The method is composed of four modules: Environment Discretization, Individual Route Planning, Route Prioritization, and Network Planning. The proposed prioritization structure decoupled the network planning problem into a sequential single-path-planning problem. The space cost function in the Individual Route Planning module made it possible to have the routes aligned and grouped together. A set of tests were carried out to show our method can generate route networks that satisfy all requirements within a short time and can be taken for commercial use. With the implementation of the proposed method, drone delivery service providers can quickly design a drone route network, and re-plan on a daily basis to respond to changes in their service network. With the route prioritization function, they can prioritize the design of urgent or important deliveries. The space cost function allowed higher airspace utilization and potentially led to the identification of high-volume air corridors in an urban area. From the city residents' perspective, less areas would be affected by this new type of aerial operations.

One direction of future work is to further reduce computational time while ensuring the optimality of the solution and the fairness between different routes. Distributed planning is a promising direction to explore.

Another direction to adapt the method to build tube-based route networks for other applications, where many moving agents should be transported among OD pairs. In the tube-based route network, one or multiple tubes can be planned to connect an OD pair based on the traveling demand. The moving agents following a tube can sequentially pass to arrive at the destination. Besides the drone route network, an example situation is warehouse logistics. In such a situation, robots should move packages from one place to another place, and the proposed method can be applied to plan the paths for these robots.

CRediT authorship contribution statement

Xinyu He: Methodology, Software, Investigation, Writing – original draft. **Fang He:** Methodology, Software, Investigation. **Lishuai Li:** Conceptualization, Methodology, Investigation, Validation, Writing – original draft, Writing – review & editing, Supervision, Project administration, Funding acquisition. **Lei Zhang:** Conceptualization, Data curation, Resources, Funding acquisition. **Gang Xiao:** Validation, Supervision.

Acknowledgment

Appendix A. Calculation for λ_r and λ_p

λ_r is calculated as $\lambda_r = \frac{o_0(s_{start}, s_{goal})}{r_0(s_{start}, s_{goal})}$, where $o_0(s_{start}, s_{goal})$ is the minimum operational cost and $r_0(s_{start}, s_{goal})$ is the minimum risk cost for a conflict-free path that connects s_{start} and s_{goal}.

λ_p is calculated as $\lambda_p = \frac{o_0(s_{start}, s_{goal})}{p_0(s_{start}, s_{goal})}$, where $o_0(s_{start}, s_{goal})$ is the minimum operational cost and $p_0(s_{start}, s_{goal})$ is the minimum space cost for a conflict-free path that connects s_{start} and s_{goal}.

Appendix B. Real-world scenario data specifications

See Tables 6 and 7.

Table 7
Summary for real-world scenario.

Name	Value	Example
Bounding box	X-direction	[−1780.99 m, 3566.63 m]
	Y-direction	[−1177.21 m, 1768.74 m]
	Z-direction	[0 m, 260 m]
Traversable layer	Z-direction	[60 m, 120 m]
Map size	Total area	15.75 km^2
	Grid size	(10 m, 10 m, 10 m)
	Number of grids	$535 * 295 * 6 = 946\,950$
	Number of traversable grids	707 128 (74.7%)
Cell attributes	Risk	$(0, 1) \rightarrow$ *low risk area*
		$(1, \infty) \rightarrow$ *high risk area*
		$1 \rightarrow$ *normal area*
	Reachable	*True* or *False*
	Reserved	*True* or *False*
	Buffer	*True* or *False*

Table 8
Computational time (s) for different Random shuffle Times K and number of routes N.

N	K									
	5	10	15	20	25	30	35	40	45	50
5	51.0	102.0	153.1	204.2	254.3	304.6	354.1	399.0	454.6	505.5
10	144.1	286.3	426.5	579.5	716.7	858.1	996.3	1127.1	1279.3	1430.3
15	204.8	408.5	612.2	815.2	1019.1	1222.6	1427.8	1630.3	1833.6	2035.6
20	226.2	450.7	673.7	898.3	1124.9	1347.7	1570.4	1795.0	2017.8	2242.8
25	289.3	577.2	864.0	1152.6	1441.3	1729.8	2018.4	2307.0	2594.5	2885.1
30	358.6	712.0	1067.0	1426.0	1781.0	2139.2	2498.0	2854.6	3212.1	3567.2
35	422.8	847.6	1298.6	1726.8	2150.5	2578.8	3001.5	3426.0	3849.0	4272.7
40	501.5	1011.9	1512.5	2043.4	2531.1	3054.1	3556.6	4031.7	4480.7	5074.0

Appendix C. Computational time for different random shuffle times K and number of routes N

See Table 8.

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.

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.

Bertram, J., Wei, P., Zambreno, J., 2021. Scalable fastMDP for pre-departure airspace reservation and strategic de-conflict. In: AIAA Scitech 2021 Forum. p. 0779.

Bhattacharya, S., Kumar, V., Likachev, M., 2010. Distributed optimization with pairwise constraints and its application to multi-robot path planning. In: Robotics: Science and Systems VI, Vol. 177. Robotics: Science and Systems Foundation, pp. 87–94. http://dx.doi.org/10.15607/RSS.2010.VI.023, URL http://www.roboticsproceedings.org/rss06/p23.pdf.

Bin Mohammed Salleh, M.F., Chi, W., 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: AIAA Information Systems-AIAA Infotech At Aerospace, January 8-12, 2018. Kissimmee, Florida, USA, p. 2260. http://dx.doi.org/10.2514/6.2018-2260.

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.

Cekmez, U., Ozsiginan, M., Sahingoz, O.K., 2014. A UAV path planning with parallel ACO algorithm on CUDA platform. In: 2014 International Conference on Unmanned Aircraft Systems (ICUAS). IEEE, pp. 347–354.

Chamseddine, A., Zhang, Y., Rabbath, C.A., Join, C., Theilliol, D., 2012. Flatness-based trajectory planning/replanning for a quadrotor unmanned aerial vehicle. IEEE Trans. Aerosp. Electron. Syst. 48 (4), 2832–2847. http://dx.doi.org/10.1109/TAES.2012.6324664.

Chung, S.-H., 2021. Applications of smart technologies in logistics and transport: A review. Transp. Res. E 153, 102455. http://dx.doi.org/10.1016/j.tre.2021.102455.

CITYPOPULATION, 2020. CHINA: Zhejiang. URL https://www.citypopulation.de/en/china/zhejiang/admin/.

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.

Culligan, K., Valenti, M., Kuwata, Y., How, J.P., 2007. Three-dimensional flight experiments using on-line mixed-integer linear programming trajectory optimization. In: Proceedings of the American Control Conference, July 11-13, 2007. New York, NY, USA, pp. 5322–5327. http://dx.doi.org/10.1109/ACC.2007.4283101.

Daniel, K., Nash, A., Koenig, S., Felner, A., 2010. Theta*: Any-angle path planning on grids. J. Artificial Intelligence Res. 39, 533–579. http://dx.doi.org/10.1613/jair.2994, URL https://www.jair.org/index.php/jair/article/view/10676.

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.

Dezfoulian, S.H., Wu, D., Ahmad, I.S., 2013. A generalized neural network approach to mobile robot navigation and obstacle avoidance. In: Intelligent Autonomous Systems 12. Springer, pp. 25–42.

Dijkstra, E.W., 1959. A note on two problems in connexion with graphs. Numer. Math. 1 (1), 269–271. http://dx.doi.org/10.1007/BF01386390.

Duvall, T., Green, A., Langstaff, M., Miele, K., 2019. Air-mobility solutions: What they'll need to take off. URL www.mckinsey.com/industries/capital-projects-and-infrastructure/our-insights/air-mobility-solutions-what-theyll-need-to-take-off.

Englot, B., Hover, F., 2011. Multi-goal feasible path planning using ant colony optimization. In: 2011 IEEE International Conference on Robotics and Automation. IEEE, pp. 2255–2260.

Ester, M., Kriegel, H.-P., Sander, J., Xu, X., 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, August 2-4, 1996, 96. pp. 226–231.

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, 2017-Janua. pp. 29–37.

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.

Grady, D.K., Bekris, K.E., Kavraki, L.E., 2010. Asynchronous distributed motion planning with safety guarantees under second-order dynamics. In: Algorithmic Foundations of Robotics IX, Vol. 68. Springer, Berlin, Heidelberg, pp. 53–70. http://dx.doi.org/10.1007/978-3-642-17452-0_4.

Ha, Q.M., Deville, Y., Pham, Q.D., Hà, M.H., 2018. On the min-cost traveling salesman problem with drone. Transp. Res. C 86, 597–621. http://dx.doi.org/10.1016/j.trc.2017.11.015, arXiv:1512.01503.

HangzhouPostal, 2020. Hangzhou express logistics industry develops steadily in 2020. URL https://www.hangzhou.gov.cn/art/2020/12/31/art_812262_59023763.html.

Hart, P.E., Nilsson, N.J., Raphael, B., 1968. A formal basis for the heuristic determination of minimum cost paths. IEEE Trans. Syst. Sci. Cybern. 4 (2), 100–107. http://dx.doi.org/10.1109/TSSC.1968.300136.

Hoekstra, J.M., Ellerbroek, J., Sunil, E., Maas, J., 2018. Geovectoring: reducing traffic complexity to increase the capacity of uav airspace. In: International Conference for Research in Air Transportation, Barcelona, Spain.

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.

Hu, Y., Yang, S.X., 2004. A knowledge based genetic algorithm for path planning of a mobile robot. In: IEEE International Conference on Robotics and Automation, Vol. 5. IEEE, pp. 4350–4355.

Islami, A., Chaimatanan, S., Delahaye, D., 2017. Large-scale 4D trajectory planning. In: Air Traffic Management and Systems II. Springer, pp. 27–47.

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, p. 0449. http://dx.doi.org/10.2514/6.2017-0449.

Jardin, M.R., 2005. Analytical relationships between conflict counts and air-traffic density. J. Guid. Control Dyn. 28 (6), 1150–1156. http://dx.doi.org/10.2514/1.12758.

Kant, K., Zucker, S.W., 1986. Toward efficient trajectory planning: The path-velocity decomposition. Int. J. Robot. Res 5 (3), 72–89. http://dx.doi.org/10.1177/027836498600500304.

Karak, A., Abdelghany, K., 2019. The hybrid vehicle-drone routing problem for pick-up and delivery services. Transp. Res. C 102, 427–449. http://dx.doi.org/10.1016/j.trc.2019.03.021.

Karaman, S., Frazzoli, E., 2011. Sampling-based algorithms for optimal motion planning. Int. J. Robot. Res. 30 (7), 846–894.

Kavraki, L.E., Švestka, P., Latombe, J.C., Overmars, M.H., 1996. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans. Robot. Autom. 12 (4), 566–580. http://dx.doi.org/10.1109/70.508439.

Kellermann, R., Biehle, T., Fischer, L., 2020. Drones for parcel and passenger transportation: A literature review. Transp. Res. Interdiscip. Perspect. 4, 100088.

Kersten, H., Benedikt, K., Robin, R., 2021. Advanced air mobility in 2030. URL https://www.mckinsey.com/industries/aerospace-and-defense/our-insights/advanced-air-mobility-in-2030.

Kitonsa, H., Kruglikov, S.V., 2018. Significance of drone technology for achievement of the United Nations sustainable development goals. R-Economy 4 (3), 115–120.

Koenig, S., Likhachev, M., 2005. Fast replanning for navigation in unknown terrain. IEEE Trans. Robot. 21 (3), 354–363. http://dx.doi.org/10.1109/TRO.2004.838026.

Kopardekar, P., 2014. Unmanned aerial system (UAS) traffic management (UTM): Enabling low-altitude airspace and UAS operations. URL https://ntrs.nasa.gov/api/citations/20140013436/downloads/20140013436.pdf.

Kopardekar, P., Rios, J., Prevot, T., Johnson, M., Jung, J., Robinson, J.E., 2016. Unmanned aircraft system traffic management (UTM) concept of operations. In: 16th AIAA Aviation Technology, Integration, and Operations Conference, June 13-17, 2016. Washington, D.C., USA, pp. 1–16. http://dx.doi.org/10.2514/6.2016-3292.

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.

Kuchar, J.K., Yang, L.C., 2000. A review of conflict detection and resolution modeling methods. IEEE Trans. Intell. Transp. Syst. 1 (4), 179–189.

Lavalle, S.M., 1998. Rapidly-Exploring Random Trees: A New Tool for Path Planning. Technical Report, Iowa State University, URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.35.1853.

Lemardelé, C., Estrada, M., Pagès, L., Bachofner, M., 2021. Potentialities of drones and ground autonomous delivery devices for last-mile logistics. Transp. Res. E 149, 102325. http://dx.doi.org/10.1016/j.tre.2021.102325.

Leroy, S., Laumond, J.-P., Siméon, T., 1999. Multiple path coordination for mobile robots: A geometric algorithm. In: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI), July 31-August 6, 1999, 2. pp. 1118–1123, Stockholm, Sweden.

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.

Liu, S., Jiang, H., Chen, S., Ye, J., He, R., Sun, Z., 2020. Integrating Dijkstra's algorithm into deep inverse reinforcement learning for food delivery route planning. Transp. Res. E 142, 102070. http://dx.doi.org/10.1016/j.tre.2020.102070.

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.

Masehian, E., Sedighizadeh, D., 2010. Multi-objective PSO-and NPSO-based algorithms for robot path planning. Adv. Electr. Comput. Eng. 10 (4), 69–76.

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.

MorganStanley, 2021. eVTOL/Urban air mobility TAM update: A slow take-off, but sky's the limit. URL https://assets.verticalmag.com/wp-content/uploads/2021/05/Morgan-Stanley-URBAN_20210506_0000.pdf.

Moshref-Javadi, M., Lee, S., Winkenbach, M., 2020. Design and evaluation of a multi-trip delivery model with truck and drones. Transp. Res. E 136, 101887. http://dx.doi.org/10.1016/j.tre.2020.101887.

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.

Octavian Thor, P., Bogdan, D., 2009. Free flight vs. Centralized air traffic management. Incas Bull. 3 (4), 67–75. http://dx.doi.org/10.13111/2066-8201.2011.3.4.7.

O'Donnell, P.A., Lozano-Pérez, T., 1989. Deadlock-free and collision-free coordination of two robot manipulators. In: 1989 International Conference on Robotics and Automation (ICRA), May 14-19, 1989. IEEE, Scottsdale, AZ, USA, pp. 484–489. http://dx.doi.org/10.1109/ROBOT.1989.100033.

Peng, J., Akella, S., 2005. Coordinating multiple robots with kinodynamic constraints along specified paths. Int. J. Robot. Res. 24 (4), 295–310. http://dx.doi.org/10.1177/0278364905051974.

Rajendran, S., Srinivas, S., 2020. Air taxi service for urban mobility: a critical review of recent developments, future challenges, and opportunities. Transp. Res. E 143, 102090. http://dx.doi.org/10.1016/j.tre.2020.102090.

Ryan, M., 2010. Constraint-based multi-robot path planning. In: 2010 IEEE International Conference on Robotics and Automation, May 4 - 8, 2010. IEEE, Anchorage, Alaska, USA, pp. 922–928. http://dx.doi.org/10.1109/ROBOT.2010.5509582.

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.

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.

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.

Sharon, G., Stern, R., Goldenberg, M., Felner, A., 2013. The increasing cost tree search for optimal multi-agent pathfinding. Artificial Intelligence 195, 470–495. http://dx.doi.org/10.1016/j.artint.2012.11.006.

Shen, L., Wang, Y., Liu, K., Yang, Z., Shi, X., Yang, X., Jing, K., 2020. Synergistic path planning of multi-UAVs for air pollution detection of ships in ports. Transp. Res. E 144, 102128. http://dx.doi.org/10.1016/j.tre.2020.102128.

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.

Singh, N.H., Thongam, K., 2019. Neural network-based approaches for mobile robot navigation in static and moving obstacles environments. Intell. Serv. Robot. 12 (1), 55–67.

Škrinjar, J.P., Škorput, P., Furdić, M., 2019. Application of unmanned aerial vehicles in logistic processes. Lect. Notes Netw. Syst. 42 (January), 359–366. http://dx.doi.org/10.1007/978-3-319-90893-9_43.

Sonmez, A., Kocyigit, E., Kugu, E., 2015. Optimal path planning for UAVs using genetic algorithm. In: 2015 International Conference on Unmanned Aircraft Systems (ICUAS). IEEE, pp. 50–55.

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, 24. pp. 173–178, Atlanta, Georgia, USA.

Sturtevant, N.R., 2012. Benchmarks for grid-based pathfinding. IEEE Trans. Comput. Intell. AI Games 4 (2), 144–148.

Sunil, E., Hoekstra, J., 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, June 23-26, 2015. Lisbon, Portugal.

Surynek, P., 2009. A novel approach to path planning for multiple robots in bi-connected graphs. In: 2009 IEEE International Conference on Robotics and Automation, May 12-17, 2009. IEEE, Kobe, Japan, pp. 3613–3619. http://dx.doi.org/10.1109/ROBOT.2009.5152326.

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, 26.

Van Den Berg, J.P., Overmars, M.H., 2005. Prioritized motion planning for multiple robots. In: 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, August 2-6, 2005. IEEE, Edmonton, Alta., Canada, pp. 430–435. http://dx.doi.org/10.1109/IROS.2005.1545306.

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

Warren, C.W., 1990. Multiple robot path coordination using artificial potential fields. In: Proceedings., 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, S.X., Meng, M., 2000. An efficient neural network approach to dynamic robot motion planning. Neural Netw. 13 (2), 143–148.

Yang, K., Sukkarieh, S., 2008. 3D smooth path planning for a UAV in cluttered natural environments. In: 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, pp. 794–800.

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., 2020. Scalable multi-agent computational guidance with separation assurance for autonomous urban air mobility. J. Guid. Control Dyn. 43 (8), 1473–1486.

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., 2013a. Multi-agent path planning and network flow. In: Frazzoli, E., Lozano-Perez, T., Roy, N., Rus, D. (Eds.), Algorithmic Foundations of Robotics X. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 157–173. http://dx.doi.org/10.1007/978-3-642-36279-8_10.

Yu, J., LaValle, S.M., 2013b. Structure and intractability of optimal multi-robot path planning on graphs. In: Proceedings of the 27th AAAI Conference on Artificial Intelligence, 14-18 July, 2013. pp. 1443–1449.

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.

Zammit, C., Van Kampen, E.-J., 2018. Comparison between A* and RRT algorithms for UAV path planning. In: 2018 AIAA Guidance, Navigation, and Control Conference. p. 1846.

Zhang, G., Zhu, N., Ma, S., Xia, J., 2021. Humanitarian relief network assessment using collaborative truck-and-drone system. Transp. Res. E 152, 102417. http://dx.doi.org/10.1016/j.tre.2021.102417.

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