IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 12, 2019, accepted June 24, 2019, date of publication June 27, 2019, date of current version July 18, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2925531

Low-Altitude Navigation for Multi-Rotor Drones in Urban Areas

AHMED BAHABRY[1], XIANGPENG WAN[1], (Senior Member, IEEE), HAKIM GHAZZAI[1], (Member, IEEE), HAMID MENOUAR[2], (Senior Member, IEEE), GREGG VESONDER[1], AND YEHIA MASSOUD[1], (Fellow, IEEE)

[1]School of Systems and Enterprises, Stevens Institute of Technology, Hoboken, NJ 07030, USA
[2]Qatar Mobility Innovations Center (QMIC), Qatar University, Doha, Qatar

Corresponding author: Hakim Ghazzai (hghazzai@stevens.edu)

This work was made possible, in part, by grant NPRP#9-257-1-056 from the Qatar National Research Fund (a member of The Qatar Foundation). The statements made herein are solely the responsibility of the authors.

ABSTRACT Multi-rotor drones have witnessed a drastic usage increase in several smart city applications due to their 3D mobility, flexibility, and low cost. Collectively, they can be used to accomplish different short- and long-term missions that require low-altitude motion in urban areas. Therefore, it is important to efficiently manage the operation of the fleet to leverage its use and maximize its application performances. In this paper, we propose to investigate the path routing problem for the multiple drones in urban areas, where obstacles with different heights exist. The objective is to find the best trajectories in this 3D environment while ensuring collision-free navigation. The collision is prevented by three possible alternatives: forcing the drone to statically hover, so its peer can pass first, making it fly at a different altitude, or completely changing its path. Multiple charging stations are made available to allow the drones to recharge their batteries when needed. A mixed integer linear program is first developed to model the problem and achieve optimal navigation of the fleet. Afterward, two heuristic algorithms with different conceptual constructions are designed to solve the trajectory planning problem with faster convergence speed. The selected simulation results illustrate the performance of our framework in realistic 3D maps and show that the designed heuristic approaches provide close performances to the optimal ones.

INDEX TERMS Unmanned aerial vehicles (UAVs), fleet path planning, energy management, collision avoidance, smart city.

I. INTRODUCTION

Drones, *aka* unmanned aerial vehicles (UAVs), are attracting drastic increase of attention in recent years. Their ability extends to performing a broad variety of applications, not just limited to military applications, but also in the public and civil sectors. They are supposed to touch many aspects of our daily life. Multi-rotor drones, to name a few, are employed in monitoring and surveying applications, communication and networking, agriculture, and various industrial activities [2]–[4]. They are used in monitoring industrial construction, road traffic situation, bridge inspection, making aerial videos, acting as first-responder activities in crisis regions, and helping in rescuing missions. Multinational big companies like Amazon and Uber are currently testing several new drone-based solutions for smart cities applications. For instance, Amazon is experimenting drone-based delivery systems [5] and Uber is designing an Air taxi aiming at transporting people using multi-rotor drones [6]. Indeed, drones present many advantages compared to conventional ground vehicles due to their three dimensional (3D) mobility, flexibility, and low-cost design. These features make it an appealing potential candidate for several innovative systems and applications for future smart cities.

Such drone advantages highly contribute to a wide range of applications and attract substantial investments. The PwC report [7] predicts that there are total of about $127 billion addressable future markets for commercial drone-based solutions. However, drones for smart cities and other applications confront many challenges that limit their practical use [8]. For instance, battery limitation handicaps their operation range and continuous activity, which may result in fatal consequences if the battery is drained, e.g., application failure or drone crash. Hence, it is mandatory to consider the energy

The associate editor coordinating the review of this manuscript and approving it for publication was Halil Ersin Soken.

management of these flying units [9]. Also, in urban cities, there are many high buildings that may impede the direct routes of drones especially when they are flying at low altitude. In other words, the degrees of freedom of their mobility are restricted by several permanent and non-permanent obstacles requiring consideration for optimized path selection and enhanced navigation. When managing a fleet of drones that are operating in the same urban area, it is also very important to avoid the risk of collision among them. In such crowded areas where multiple drones are undertaking different activities, avoiding such risky situations will certainly improve the potential of drone-based applications. Under this context, we propose, in this paper, to design a proactive navigation strategy for multiple drones circulating in urban areas. This is performed while considering the energy limitation and avoiding the risk of collision.

In this paper, we focus on ascertaining optimized routes for a fleet of drones flying at low altitude within the region in urban cities where buildings with different heights exist. The objective is to minimize the total time spent by the drones whose mission start from different locations to end at different destination points. The path planning is performed while ensuring that the energy level of each drone is sufficient to complete the trip. To do so, charging stations are made available around the city to let drones recharge their batteries whenever it is needed. Few work have studied the case where charging stations are placed along the way of the fixed-wing drones [10], [11]. In one of our previous work, we have proposed a charging station placement approach in urban areas for multi-rotor drones [8]. The contributions of this paper are summarized as follows:

- A generic framework is developed to determine independent 3D paths of a fleet of drones in realistic maps of urban areas. The objective is to find the fastest trajectories for all drones so that the overall time spent by the drone swarms is minimal while ensuring collision-free navigation and considering the energy constraints.
- An optimal approach based on a mixed integer linear programming problem (MILP) incorporating the different aspects of the investigated framework is efficiently developed. The MILP program i) determines the fastest routes, ii) considers the need to go to the charging stations if required, and iii) guarantees a collision free drone operation by imposing a time gap ensuring different arrival instances of the drones to any segment of the 3D map. In this way, the drones that are selecting similar segments might need to wait by statically hovering before resuming their trips. The collision is also avoided by forcing the drones to navigate at different altitudes when crossing the same path segments. Alternatively, a drone may completely select another path. The optimizer will jointly determine the best trajectories for the benefit of the whole fleet.
- Two heuristic iterative algorithms are then developed. The first one, called Ordered Fleet Iterative Algorithm (OFIA), determines a path for a drone at each iteration given the previous trajectories of its peers. Hence, if there is a risk that it will collide with the precedent selected drones, the current drone is forced to hover or change its trajectory. The second algorithm, called Step Forward Iterative Algorithm (SFIA), simultaneously determines the next path segments of all the fleet at each iteration. Hence, at each iteration, the drones take one action: cross a path segment, hover, or charge the battery. If a collision risk exists, affected drones will be forced to statically hover.

The proposed collision-free navigation solutions are proactive approaches determining the trip plan of each member of the fleet beforehand, i.e., before the motion of the fleet to ensure a safe flight for the drones. They are applied and tested on realistic maps where building and obstacle heights are taken into account, e.g., Manhattan area in New York City. The performances of all proposed solutions are evaluated and compared to each other for various system parameters including the battery capacity of the drones, their initial battery levels, and arrival gap tolerance. Results show how the fleet is efficiently managed and trajectories are obtained for different cases. Moreover, they show that the heuristic approaches achieve close total travel time to the one obtained using the optimal solution with significant computational saving.

The proposed free-collision navigation framework could be applied for various applications in smart cities. For instance, it can be used as tool to optimize the trips of drones acting as flying delivery units where each drone has a package to deliver to a pre-defined destination. Another application could be the case where drones equipped with cameras or sensors are used to automatically capture images or collect data during their trips, for example, to monitor a part of the city. Many other applications requiring determining fast trajectories in urban areas can also be executed using the proposed framework. In emergency response applications, some drones need to be quickly. Hence, the framework can assign to them higher priority levels to determine the quickest trajectories to them, while forcing lower priority level drones to hover or change their paths accordingly. Finally, the drones can be part of different applications sharing the same geographical area.

The rest of the paper is organized as follows. Section II provides a literature review. Section III presents the system model and related parameters. Section IV develops the MILP optimal formulation of the free-collision navigation problem. Section V describes the proposed heuristic approaches. Section VI presents and discusses selected simulation results. Finally, concluding remarks and future directions are drawn in Section VII.

II. RELATED WORK

Most of the previous studies investigating the employment of drones focus on their efficiency in performing different missions such as monitoring, tracking, and detecting with videos/imagery [12], multicasting [13], or data exchange

through vehicular ad hoc networks [14], [15]. They concentrate on the drone applications' outputs rather than the drone operation itself. Hence, most of these studies do not consider the energy and travel time constraints in their models. Additionally, they do not consider different flying heights of drones and/or the risks of collision in their system design. In practice, taking into account these parameters when assigning missions to drones is essential to ensure safe operation of the flying fleet.

For all applications described earlier, path planning is one of the fundamental problems that have been solved [16]. In [17], the authors proposed a search algorithm for route planning utilizing heuristic approaches. In [18], the authors took the airspace, geo-locations, flight risk, and sensing range into consideration and selected the suboptimal routes using bio-inspired algorithms. In [19], the authors provided a path planning for both UAVs and unmanned ground vehicles (UGVs) to monitor targets in urban area. In [20], the authors proposed a path planning algorithm based on evolving waypoints for a single drone. The authors of [21] proposed multi-drone-assisted search-and-reconnaissance trajectory planning with the objective of minimizing the total latency of the system. The drones having different speeds and starting instants aim to rapidly collect information about the area of interest. Graph-theory based algorithms inspired from the travel salesman problem (TSP) are developed [22]. Other studies have focused on more generalized 3D environment [23]. In [24], the authors proposed a modification of D* algorithm which incrementally finds path during the flight. The selected path effectively allows drones to avoid obstacles while navigating. The disadvantages is that the cost of selected path is based on a greedy search instead of global optimum solution. In [25], the authors presented a full dynamic model of a quadrotor UAV in 3D environment. The method considers simplified navigation model from point to point instead of realistic road network. Similarly, a random tree search under uncertainty 3D environment with simplified map is presented in [26]. Other approaches are based on meta-heuristic algorithms designed for similar vehicular routing problems [27]–[29]

In modern urban city applications, the missions of drones could be varied and their number within the region is expected to be high. Therefore, it is important to jointly optimize their application performances and ensure collision-free operation while taking into account important factors such as the capacity limits of the flying units. A recent work for delivery applications has been presented in [30]. The fleet of drones is managed by a platform employing different routing methods include optimal, first-in-first-out (FIFO), and Hungarian techniques. The framework ignores obstacles between the different points of interest, and, in terms of energy consumption, it guarantees that, before returning to its base, each drone must have at least 10% remaining charge.

Few studies have dealt with the collision avoidance problem for drones in their developed solutions. The authors of [31] have studied the path planning with the aim of minimizing drone travel time for data gathering application. The proposed model considers energy and fair route distances as well as collision avoidance. Column generation heuristic approach was used and the fleet of drones was assimilated as multiple traveling sales men. The proposed approach uses a simplified grid as a map and collision is avoided by imposing that two drones cannot be at the grid point at the same time. Similarly, in [32], the authors proposed to use reinforcement learning and dynamic evolutionary algorithms to manage the drone fleet in a grid. However, the energy issue has not been taken into account. In [33], a MILP has been developed to determine an energy-efficient path planning solution for multi-drone system. In this study, collision is avoided by discretizing the 3D space and verifying the occupancy of each 3D block by one drone at maximum. In [34], a collision avoidance problem for multiple fixed-wing drones in an open-space area is investigated to enable cooperative flight formation and effective mission completion. A consensus-based algorithm is proposed based on leader-follower strategy to ensure the convergence of the formation while maintaining a constant relative distance in the vertical direction. Most of the previous work investigating the collision-free navigation of drones deal with particular scenarios or simplify their models by dealing with grid or open-space maps and/or ignore the battery recharging issue. In the present study, we focus on incorporating all the challenges related to the drone navigation into a single framework generic for various drone-based applications in smart cities.

III. SYSTEM MODEL

We consider an urban area composed of multiple blocks each containing a certain number of buildings of different heights. We consider that the closed polygon surrounding a block is a set of path segments through which drones can navigate. Nevertheless, the drones can cross a block while flying at an altitude higher than the tallest building in that block, otherwise they have to stick to the path segments formed by the block as shown in Figure 1. We consider L layers identified by different height levels that drones could possibly fly over. For instance, Layer 1 could be the layer with the lowest height (e.g., 20 meter of altitude). It can be assimilated to the road network of the city. Layer 2 has a higher altitude and drones can fly over some buildings (e.g., 30 meter of altitude). Hence, diagonal path segments can be formed between the vertices of the block. In the present paper, we assume that the height of a block corresponds to the one of the highest building within that block. The use of L layers with pre-determined and fixed altitudes is made to organize the navigation of the fleet in the urban area. The layers can be pre-defined by the drones operator or the local authority in order to ensure safe and convenient navigation of the drones, e.g., the difference between the layer heights is enough for safe navigation and the minimum layer height is sufficiently high according to noise and privacy regulations. For the optimizer, the number of layers and their heights are part of the 3D map, i.e., inputs. Without loss of generality,

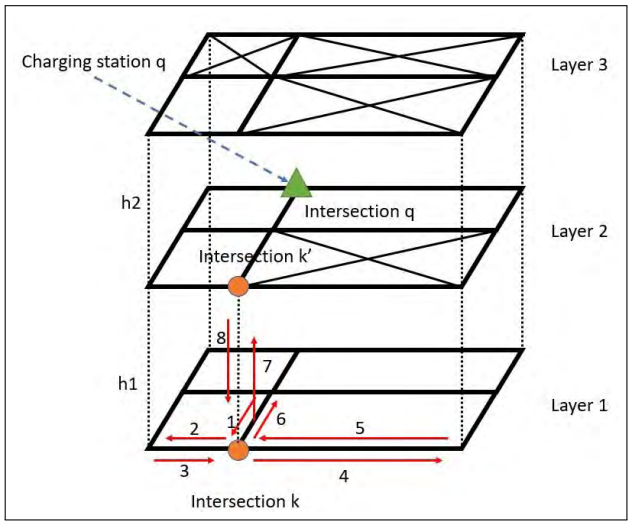


FIGURE 1. An example of three layers 3D map where the drones can navigate through. Both the horizontal (solid black line) and the vertical (dashed black line) path segments are bi-directional for drones. Eight path segments and three intersections are labeled in this example. The intersection *k* has four inputs and four outputs.

we assume that a drone vertically flies if it needs to move from a layer to another.[1]

The 3D path grid is then composed of intersections and path segments. In the paper's context, each path segment i, where $i \in \{1, \ldots, N\}$ is connecting at maximum two intersections, where N is the number of path segments in the area of interest. We denote by $\mathcal{I}$ the set including all intersections without charging stations in the area. Each intersection in $\mathcal{I}$ is indexed by k where $k \in \{1, \cdots, |\mathcal{I}|\}$, and $|\mathcal{I}|$ denotes the cardinality of the set $\mathcal{I}$. We denote by $\mathcal{Q}$ the set including all intersections with charging stations in the area. Each intersection in $\mathcal{Q}$ is indexed by q where $q \in \{1, \cdots, |\mathcal{Q}|\}$. Hence, each path segment starts at one intersection k or q and ends at another intersection k' or q'. We denote the length of a path segment i by d_i, which is calculated based on the geographical values of the two intersections. We define the geographical values of intersection u as ϕ_u and ψ_u, where $u \in \{k, q\}$, representing the longitude and latitude values, respectively. Consequently, for each intersection $k \in \mathcal{I}$, we denote its input set by $\mathcal{I}_k^{\text{In}}$ and its output set by $\mathcal{I}_k^{\text{Out}}$. In the example shown in Fig. 1, $\mathcal{I}_k^{\text{In}}$ and $\mathcal{I}_k^{\text{Out}}$ are given as follows:

$$\mathcal{I}_k^{\text{In}} = \{1, 3, 5, 8\}, \quad \text{and} \quad \mathcal{I}_k^{\text{Out}} = \{2, 4, 6, 7\}. \tag{1}$$

Similar remark can be applied for the intersections $q \in \mathcal{Q}$ but with the possibility to land or take off from the co-located charging station q. Note that we assume that at most one charging station is placed at each intersection and that a charging station can handle a sufficient number of drones simultaneously. In practice, placing charging stations at intersections is more effective than placing them in the middle of a path segment since they become more accessible to the fleet.

In this framework, we assume there are D drones flying within the area, each one of them j, where $j \in \{1, \ldots, D\}$, can take four actions at every step s. The actions can be moving up to the higher layer, moving down to the lower layer, moving in the same layer, and statically hovering at the same position. We assume that the drones can hover at every intersection, and we denote the minimum hovering time of a drone at that intersection by T^{ho} (e.g., 5 seconds). Hence, the drone may hover for multiple T^{ho}. The drone may need to hover at an intersection to avoid entering a path segment at the same time with its peer and thus, avoid any risk of collision.

The energy consumption for the first three actions depends on the speed of the drone denoted by v_j. For each drone j, we denote the power consumption at path segment i by $p_{j,i}$ and the power consumption for hovering at intersection u by $p_{j,u}$ where $u \in \{k, q\}$. It is evident that the energy consumption of vertically moving to an upper layer is greater than horizontally moving in the same layer, which is also higher than vertically moving down to a lower layer. All of them are strictly greater than the hovering power.

We denote the initial battery level and battery capacity of drone j by b_j^0 and $\bar{C}_j$, respectively. If the battery of the drone is running low, it could be reloaded in a charging station located at the intersections $q \in \mathcal{Q}$. To do so, the drone must visit one of the charging stations to reload its battery before resuming its trip. We compute the amount of energy stored by a drone after a charging period T^{ch}, e.g., 30 seconds, as $E_q^{ch} = P_q^{ch} T^{ch}$ where P_q^{ch} is the charging power of station q. Note that the stored energy of the drone j cannot exceed its battery capacity $\bar{C}_j$ when charging. Nevertheless, a drone can charge its battery for multiple T^{ch} periods. The period of time T^{ch} is the minimum time that a drone needs to spend to charge its battery.

Finally, we define the starting position of each drone j as St_j and the destination De_j, where $St_j \neq De_j$. The system model parameters are listed in Table 1.

IV. PROBLEM FORMULATION: OPTIMAL MILP-BASED SOLUTION

The objective of the proposed framework is to minimize the total travel time that the drones spend to navigate from their respective starting points to their destinations within the city, taking into account possible extra time due to imposed stops for battery charging. In general, the selected routes for some drones might not be the fastest ones from an individual perspective, but they correspond to the best routes for the whole fleet. We define the time spent by a drone j on a segment i as follows:

$$t_{j,i} = \frac{d_i}{v_j}. \tag{2}$$

To reach the destination, the drone needs to cross multiple segments that we define as the steps taken by the drone and we index them by s. Hence, we introduce the decision variable

[1]This assumption is made to simplify the notations of the framework and make the figures more tractable. The framework is still operational when diagonal navigation between layers is considered.

TABLE 1. System model parameters.

Parameters	Description
i	Path segment i connecting two intersections
k	Intersection k without charging station
q	Intersection q with charging station
u	Intersection u contains both k and q
d_i	Length of path segment i
$\mathcal{I}_k^{\text{In}}$	Set of path segments that toward intersection k
$\mathcal{I}_k^{\text{Out}}$	Set of path segments that leave intersection k
j	Drones j that flying within the area
T^{ho}	Minimum hovering time at each intersection
v_j	Average speed for drones j
$p_{j,i}$	Power consumption for drone j passing segment i
$p_{j,u}$	Power consumption for drone j hovering at intersection u
b_j^0	Initial battery level of drone j
C_j	Battery capacity of drone j
T^{ch}	Minimum battery charging period
P_q^{ch}	Charging power of station q
E_q^{ch}	Energy stored after one charging period
St_j	Start position of drone j
De_j	Destination of drone j

$x_{j,i,s}$ to indicate whether a drone j has selected the route i in step s as part of its path or not as follows:

$$x_{j,i,s} = \begin{cases} 1, & \text{if the drone } j \text{ chooses path segment } i \\ & \text{at step } s, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

Then, we introduce a decision variable indicating whether drone j went to a charging station q to reload its battery at step s or not as follows:

$$y_{j,q,s} = \begin{cases} 1, & \text{if the drone } j \text{ chooses the charging station } q \\ & \text{at step } s, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

Moreover, we add a decision variable indicating whether drone j is hovering at an intersection u where $u \in \{k, q\}$ at step s to delay its entrance to an associated segment or not as follows:

$$z_{j,u,s} = \begin{cases} 1, & \text{if the drone } j \text{ hovers at intersection } u \\ & \text{at step } s, \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Assuming that the total number of steps needed at most by drone j to reach its destination is denoted by $\mathcal{S}_j$ then, the set $\{x_{j,i,s}, y_{j,q,s}, z_{j,u,s}\}$ where $\{x_{j,i,s} = 1, y_{j,q,s} = 1,$ and $, z_{j,u,s} = 1$ for $s \in \{1, \cdots, \mathcal{S}_j\}\}$ corresponds to the entire path of the drone j including its stops at the charging stations and hovering at intersections, if any.

In addition to the binary decision variables $x_{j,i,s}$, $y_{j,q,s}$, and $z_{j,k,s}$, we introduce a continuous decision variable in order to guarantee a collision-free operation of the fleet. We define this decision variable as the arrival time of drone j at path segment i and we denote it by $T_{j,i}$. The arrival time $T_{j,i}$ at each segment i must be different from any other arrival time of another drone $T_{j',i}$ where $j' \neq j$. Hence, to avoid any collision, the difference must be greater than a certain arrival gap time denoted by $\bar{T}$ guaranteeing safe navigation over the segment

TABLE 2. Primary decision variables.

Decision Variable	Type	Description
$x_{j,i,s}$	Binary	a drone j has selected the route i in step s or not
$y_{j,q,s}$	Binary	a drone j goes to a charging station q to reload its battery at step s or not
$z_{j,u,s}$	Binary	a drone j is hovering at an intersection u at step s to delay its entrance to an associated segment or not
$T_{j,i}(s^*)$	Continuous	the arrival time of drone j at path segment i during the corresponding step s^*

path i. Assuming that the drone j arrives at segment i at step s^* then, the arrival time can be expressed as:

$$T_{j,i}(s^*) = \begin{cases} \sum_{\sigma=1}^{s^*} \Big(\sum_{i=1}^{N} x_{j,i,\sigma} t_{j,i} + \sum_{q=1}^{|\mathcal{Q}|} y_{j,q,\sigma} T^{ch} \\ + \sum_{u=1}^{|\mathcal{I}|\cup|\mathcal{Q}|} z_{j,u,\sigma} T^{ho} \Big), & \text{if drone } j \text{ arrives at } i, \\ +\infty, & \text{otherwise.} \end{cases} \tag{6}$$

Note that the framework assumes that the drones will arrive at any segment at maximum once. There is no loop in their trajectories.

A. UTILITY FUNCTION

Recall that the objective of the framework is to minimize the total time spent by all drones. Consequently, the utility function to be minimized, $\mathcal{U}$, can be expressed as follows:

$$\mathcal{U} = \sum_{j=1}^{D} T_{j,De_j}(\mathcal{S}_j). \tag{7}$$

The maximum number of steps required by all the drones to reach their destinations can be expressed as $\mathcal{S} = \max_{j \in \{1,\ldots,D\}} S_j$.

B. PROBLEM CONSTRAINTS

In order to ensure that the selected routes are reasonable and collision-free, the following constraints must be considered:

1) STEP BY STEP CONSTRAINTS

First, we need to ensure that the drone is moving in such a way it is located at one path segment, at a charging station, or hovering at an intersection u ($u \in \{k, q\}$) during each step s as long as it did not reach the destination. In the latter case, the drone stops moving. Therefore, for each drone j, the following if-else statement are added:

$$\begin{aligned} &\text{if} \quad \sum_{\sigma=1}^{s-1} x_{j,De_j,\sigma} = 0, \quad \text{then,} \\ &\sum_{i=1}^{N} x_{j,i,s} + \sum_{q=1}^{|\mathcal{Q}|} y_{j,q,s} + \sum_{u=1}^{|\mathcal{I}|\cup|\mathcal{Q}|} z_{j,u,s} = 1, \\ &\text{else} \sum_{i=1}^{N} x_{j,i,s} + \sum_{q=1}^{|\mathcal{Q}|} y_{j,q,s} + \sum_{u=1}^{|\mathcal{I}|\cup|\mathcal{Q}|} z_{j,u,s} = 0, \\ &\forall s \in \{2, \ldots, \mathcal{S}_j\}, \ \forall j. \end{aligned} \tag{8}$$

In order to convert this condition into linear constraints, we employ the big-M method as follows:

$$\left(\sum_{i=1}^{N} x_{j,i,s} + \sum_{q=1}^{|\mathcal{Q}|} y_{j,q,s} + \sum_{u=1}^{|\mathcal{I}|\cup|\mathcal{Q}|} z_{j,u,s}\right) \leq 1 + M \sum_{\sigma=1}^{s-1} x_{j,De_j,\sigma}, \tag{9a}$$

$$\left(\sum_{i=1}^{N} x_{j,i,s} + \sum_{q=1}^{|\mathcal{Q}|} y_{j,q,s} + \sum_{u=1}^{|\mathcal{I}|\cup|\mathcal{Q}|} z_{j,u,s}\right) \geq 1 - M \sum_{\sigma=1}^{s-1} x_{j,De_j,\sigma}, \tag{9b}$$

$$\left(\sum_{i=1}^{N} x_{j,i,s} + \sum_{q=1}^{|\mathcal{Q}|} y_{j,q,s} + \sum_{u=1}^{|\mathcal{I}|\cup|\mathcal{Q}|} z_{j,u,s}\right) \leq M \left(1 - \sum_{\sigma=1}^{s-1} x_{j,De_j,\sigma}\right), \tag{9c}$$

$$\left(\sum_{i=1}^{N} x_{j,i,s} + \sum_{q=1}^{|\mathcal{Q}|} y_{j,q,s} + \sum_{u=1}^{|\mathcal{I}|\cup|\mathcal{Q}|} z_{j,u,s}\right) \geq M \left(1 - \sum_{\sigma=1}^{s-1} x_{j,De_j,\sigma}\right), \quad \forall s \in \{2, \ldots, \mathcal{S}_j\},\ \forall j, \tag{9d}$$

where M is a sufficiently large positive number. Note that $\sum_{\sigma=1}^{s-1} x_{j,De_j,\sigma} \in \{0, 1\}$ as after reaching the destination, the values of $x_{j,De_j,s\geq S_j} = 0$.

2) INTERSECTION CONSTRAINTS

Second, we need to ensure that the drone moves in a sequential manner when crossing an intersection. For example, if the drone enters any intersection, then, in the next step, it must leave it, hover there, or go to the associated charging station if available. This is guaranteed by using the following condition:

$$\begin{aligned}
&\text{if } \sum_{\sigma=1}^{s} x_{j,De_j,\sigma} = 0, \quad \text{then,}\\
&\sum_{i\in\mathcal{I}_k^{\text{In}}} x_{j,i,s} + z_{j,k,s} = \sum_{i\in\mathcal{I}_k^{\text{Out}}} x_{j,i,s+1} + z_{j,k,s+1},\\
&\forall s \in \{1, \ldots, \mathcal{S}_j\},\ \forall j,\ \forall k \in \mathcal{I},\\
&\text{and } \sum_{i\in\mathcal{Q}_q^{\text{In}}} x_{j,i,s} + y_{j,q,s} + z_{j,q,s} = \sum_{i\in\mathcal{Q}_q^{\text{Out}}} x_{j,i,s+1} + y_{j,q,s+1}\\
&\qquad\qquad + z_{j,q,s+1},\\
&\forall s \in \{1, \ldots, \mathcal{S}_j\}, \quad \forall j,\ \forall q \in \mathcal{Q}.
\end{aligned} \tag{10}$$

3) COLLISION AVOIDANCE CONSTRAINTS

Third, the arrival time of drone j at path segment i should be different from the ones of other drones by the time gap $\bar{T}$ to have:

$$|T_{j,i} - T_{j',i}| \geq \bar{T}, \quad \forall i,\ \forall j,\ j' \in \{1, \ldots, D\}, j \neq j'. \tag{11}$$

Recall that the value of $T_{j,i}$ is related to the decision variable $x_{j,i,s}$, $y_{j,q,s}$, and $z_{j,k,s}$, as given in (6).

4) BATTERY LEVEL AND ENERGY CONSUMPTION CONSTRAINTS

Fourth, the battery level of each drone j should be enough to fly until reaching its destination otherwise, it needs to be recharged on the way. In other words, for each step s, the energy consumption should be less than or equal to the amount of stored energy. We express the energy stored in the battery of drone j before step s as follows:

$$\begin{cases} B_j^1 = b_j^0, \\ B_j^{s+1} = \min(B_j^s - \sum_{i=1}^{N} x_{j,i,s} p_{j,i} t_{j,i} \\ - \sum_{u=1}^{|\mathcal{I}|+|\mathcal{Q}|} z_{j,u,s} p_{j,k} T^{ho} + \sum_{q=1}^{|\mathcal{Q}|} y_{j,q,s} P_q^{ch} T^{ch}, \bar{C}_j), \quad \forall j. \end{cases} \tag{12}$$

Note that the term $\sum_{i=1}^{N} x_{j,i,s} p_{j,i} t_{j,i} + \sum_{u=1}^{|\mathcal{I}|+|\mathcal{Q}|} z_{j,u,s} p_{j,k} T^{ho}$ corresponds to the energy consumed by the drone j at step s. The total energy consumption of the drone j is expressed as follows:

$$\mathcal{E}_j = \sum_{s=1}^{\mathcal{S}_j} \sum_{i=1}^{N} x_{j,i,s} p_{j,i} t_{j,i} + \sum_{u=1}^{|\mathcal{I}|+|\mathcal{Q}|} z_{j,u,s} p_{j,u} T^{ho}. \tag{13}$$

The following condition is added to ensure that the consumed energy cannot exceed the current battery level at each step s:

$$\begin{aligned}
&\text{if } \sum_{q=1}^{|\mathcal{Q}|} y_{j,q,s} = 0,\\
&B_j^s \geq \sum_{i=1}^{N} x_{j,i,s} p_{j,i} t_{j,i} + \sum_{k=1}^{|\mathcal{I}|+|\mathcal{Q}|} z_{j,k,s} p_{j,k} T^{ho}, \quad \forall s.
\end{aligned} \tag{14}$$

Note that we also use the big-M linearization method to convert the conditions and the statements (10), (11), (12), and (14) into linear constraints.

5) INITIALIZATION CONSTRAINTS

Last, we initialize the starting location of each drone j as follows:

$$x_{j,S_{t_j},1} = 1, \text{ and } \sum_{i=1}^{N} x_{j,i,1} = 1, \quad \forall j. \tag{15}$$

C. OPTIMIZATION PROBLEM

Finally, the optimization problem is formulated as follows:

$$\begin{aligned}
&\text{(P):} \quad \underset{x_{j,i,s},\, y_{j,q,s},\, z_{j,u,s},\, T_{j,i}}{\text{minimize}} \quad \mathcal{U}\\
&\text{subject to: (8), (10), (11), (12), (14), and (15).}
\end{aligned}$$

Problem (P) is a MILP that can be solved optimally using off-the-shelf software which implements algorithms such as the branch and bound technique to determine optimal routes for

the whole fleet of drones. The MILP optimization problems are classified under the NP-hard problem which requires a very high computational time to achieve the optimal solution. For such proactive navigation problems where trajectories are determined beforehand, running time might not be a very critical challenge. Nevertheless, in many applications, it is recommended to achieve navigation solutions in reasonable and fast time, especially for large-scale problems, e.g., big map and/or large fleet size. Moreover, a drone operator may need to update its fleet trajectories very frequently, for instance, when a new drone joins the fleet. Hence, the optimization problem need to be re-executed assimilating the current locations of the already flying drones as their new starting points plus the new locations of the joining drones. This process will need to be repeated for every update in the framework such as adding new destinations points or changing of the fleet. Generally, instead of MILP, meta-heuristic and evolutionary algorithms are employed in many studies [35], [36]. However, such approaches are usually adapted to unconstrained or lightly constrained problems and may converge to different local optima. Instead, in the next section, we develop heuristic deterministic algorithms incorporating all the aspects considered in the optimal MILP-based solution.

V. HEURISTIC ALGORITHMS

In this section, we propose two heuristic approaches achieving sub-optimal solutions but converging rapidly. The performances of both approaches will be compared to the ones obtained by the MILP-based solution. The proposed heuristic approaches are designed in an iterative manner such that a decision is made at each iteration. The first algorithm, identified as OFIA, determines a full trajectory for a prioritized drone then, focuses on the next drone, while the second algorithm, SFIA, finds the simultaneous steps (selecting a path segment, hovering, or battery reloading) taken by each member of the fleet at each iteration. At convergence, both algorithms generate collision-free full trajectories for the drones with minimized total travel time while respecting the energy availability at their respective batteries.

A. ORDERED FLEET ITERATIVE ALGORITHM (OFIA)

The OFIA algorithm starts by assigning different priorities to the fleet members. In other words, the drone operator decides which drone will be given priority in determining its trajectory. For instance and in this paper, we explore two different priority assigning methods. The first one is based on the expected flying time of each drone. Hence, the priority is given to the unit that has minimum flying time to reach its destination assuming that it is solely flying in the area of interest. The second priority method is based on the expected energy consumption of the drone. In this case, priority is given to the drone that has the maximum expected energy consumption to complete its trip so that its charging time is minimized.

Once the drones are ordered according to their priority levels, we determine the fastest trajectory for the drone with highest priority in the area of interest. To do so, we employ the Dijkstra's algorithm applied to the graph formed by the roads, intersections, and layers of the 3D map [37]. Other graph-based shortest path algorithms such as Bellman-Ford algorithm [38] and Johnson's algorithm [39] can be adapted. Then, the drone with the second priority level is treated. Its path is determined based on the trajectory of the precedent drone so as to avoid collision with it. Hence, arriving at any intersection of the map must respect the time gap constraint so delays are counted in these intersections. To determine the paths of this drone when a collision risk exists. We implement a modified Dijkstra's algorithm to account for the closed intersections. The intersections are not permanently closed but they are inaccessible just for the instants where the previous drones entered them plus/minus the time gap $\bar{T}$. The procedure describing the modified Dijkstra's algorithm is given in Procedure 1. The selected path for each drone needs to consider the potential risk of collision with the previous drones. In other words, some roads should be blocked for the current drone during certain period of time. The blocked roads, when the drone leaves its starting position at time t are recorded in map_t. The drone may choose to hover and wait before entering the blocked roads or completely select another path. In Procedure 1, the drone has a total number H of hovering times that results in H different expected arrival times, the fastest path with the minimum expected arrival time is selected. Then, the drone moves one step ahead and updates its current location and t till reaching its destination.

Procedure 1 Modified Dijkstra's Algorithm Accounting for Temporary Blocked Path Segments

1: **Inputs:** Drone j, $\{St, De\}$, t, map state map_t.
2: **while** The drone does not reach De **do**
3: **for** $i = 1, \ldots, H$ **do**
4: Find the fastest route using Dijkstra's algorithm from St to De at time instant $t + \bar{T}(i-1)$ assuming an updated state of the map, $map_{t+\bar{T}(i-1)}$.
5: Record the total expected travel time obtained at the i^{th} iteration: $T_{j,De}[i]$.
6: **end for**
7: Select the fastest path among $T_{j,De}[i]$.
8: **end while**

Similarly, the trajectories of the rest of fleet are determined. Hence, extra limitations are imposed to less priority drones. During their trips, if the expected energy consumption is higher than its initial battery, then the drone heads first towards the nearest available charging station to sufficiently replenish its battery. A drone may need to visit multiple charging stations before reaching its destination, e.g., for long trips or for limited battery capacity. The modified Dijkstra's algorithm is used to find the shortest path between two points (starting point and charging station, two charging stations, or a charging station and the destination). Note that we set an energy threshold for each drone to ensure that the remaining battery is sufficient to support unexpected hovering for

Algorithm 1 OFIA

Inputs: $\{St_j, De_j\}, j \in \{1, \cdots, N\}$ where N is the fleet size and 3D map.
Sort the N drones in ascending order according to the priority strategy of the operator; $\tilde{j} = 1$ is the highest priority drone and $\tilde{N}$ is the lowest priority drone.
for $\tilde{j} = 1, \ldots, \tilde{N}$ **do**
  Set $SP_{\tilde{j}} = St_{\tilde{j}}$.
  Set the battery level of drone $\tilde{j}$ as $B_{\tilde{j}} = b^0_{\tilde{j}}$.
  while The drone $\tilde{j}$ does not reach its destination ($SP_{\tilde{j}} \neq De_{\tilde{j}}$) **do**
    Find the fastest path for the drone $\tilde{j}$ using the modified Dijkstra's algorithm between $SP_{\tilde{j}}$ and $De_{\tilde{j}}$ (See Procedure 1).
    Calculate the expected energy consumption of drone $\tilde{j}$: $\mathcal{E}_{\tilde{j}}$.
    if $B_{\tilde{j}} \geq \left(\mathcal{E}_{\tilde{j}} + E_{\text{th}}\right)$ **then**
      The drone directly heads to its destination $De_{\tilde{j}}$.
      Set $SP_{\tilde{j}} = De_{\tilde{j}}$.
    else
      The drone heads to the closest charging station first q and sufficiently charges its battery.
      Update the starting point of the drone $\tilde{j}$ by the location of the charging station ($SP_{\tilde{j}} = q$).
      Update the battery level of the drone $\tilde{j}$: $B_{\tilde{j}}$.
    end if
  end while
  Record the arrival instants of the drone $\tilde{j}$ at each intersection that the drone had passed by.
  Block those segments during the arrival instants plus/minus gap time $\bar{T}$ to prevent collision for the rest of fleet members.
end for

collision avoidance reasons. The energy threshold is denoted by E_{th}. The pseudo code of the OFIA is given in Algorithm 1. The algorithm converges when all the drones reach their destinations.

B. STEP FORWARD ITERATIVE ALGORITHM (SFIA)

The idea behind the SFIA is to determine the next step to be chosen by each member of the fleet simultaneously. In other words, at each iteration, each drone will take an action and move towards its destination. To prevent collision and battery depletion, the set of actions also include the hovering and the battery charging when needed. The SFIA avoids any specific pre-ordering of the drones hence, fairer trajectories are obtained. When simultaneously reaching the same intersection, the first-in first-out (FIFO) principle is applied to the drones. Consequently, the drone arriving first to the intersection enters the road segment first, while the second drone hovers if necessary. The algorithm starts with an initialization phase where it determines the fastest paths of each drone using the Dijkstra's algorithm assuming solo

Algorithm 2 SFIA

Inputs: $\{St_j, De_j\}, j \in \{1, \cdots, N\}$ where N is the fleet size and 3D map.
Initialization Phase:
for $j = 1, \ldots, N$ **do**
  Set $SP_j = St_j$ and $DP_j = De_j$.
  Set the battery level of drone j as $B_j = b^0_j$.
  $\text{fpath}_j = \emptyset$.
  while A full path from St_j to De_j is not obtained **do**
    Find the fastest path for drone j using Dijkstra's algorithm between $\{SP_j, DP_j\}$: path_j.
    Calculate the needed energy consumption of drone j: $\mathcal{E}_j$.
    if $B_j \geq \left(\mathcal{E}_j + E_{\text{th}}\right)$ **then**
      The drone will directly go from $\{SP_j$ to $DP_j\}$.
      Update the path of drone j: $\text{fpath}_j = \text{fpath}_j \cup \text{path}_j$.
    else
      The drone heads to the closest charging station first q ($DP_j = q$).
      Find the fastest path for drone j using Dijkstra's algorithm between $\{SP_j, DP_j\}$: path_j.
      Update the path of drone j: $\text{fpath}_j = \text{fpath}_j \cup \text{path}_j$.
      Set $SP_j = q$ and $DP_j = De_j$.
      Update the battery level of the drone after sufficiently charging: B_j.
    end if
  end while
end for
Iterative Phase:
Find all the intersections of the 3D map that will be passed by the drones based on their paths $\text{fpath}_j, \forall j \in \{1, \ldots, N\}$.
Determine the intersection $\tilde{k}$ where the earliest collision will happen and find the set of colliding drones at intersection $\tilde{k}$: $\mathcal{D}_c$. If no risk of collision is noticed then, $\tilde{k} = 0$ and $\mathcal{D}_c = \emptyset$.
while There is a collision risk ($\tilde{k} \neq 0$) **do**
  Apply FIFO principle among the elements of $\mathcal{D}_c$.
  Determine the intersection $\tilde{k}$ where the earliest collision will happen and find the set of colliding drones at intersection $\tilde{k}$: $\mathcal{D}_c$. If no risk of collision is noticed then, $\tilde{k} = 0$ and $\mathcal{D}_c = \emptyset$.
end while

trips. The corresponding needed energy consumption is then computed for each drone. If there is a risk of energy depletion, the destination of the drone is changed to the nearest charging station. Then, the iterative phase starts by taking an action for each drone based on the path determined in the initialization phase. After each step (action), the arrival instants of each drone at each intersection are recorded. In case of collision risk, the FIFO principle is applied at that intersection. The algorithm converges when all the drones reach their destinations. The pseudo code of the SFIA is given in Algorithm 2.

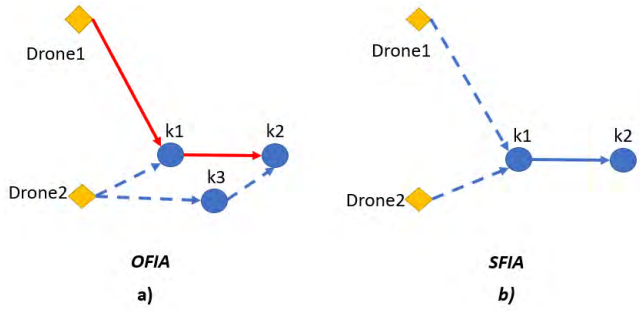


FIGURE 2. Comparison between the heuristic approaches followed by a) OFIA and b) SFIA. In OFIA, drone 2 needs to hover before entering $k1$ or go through $k3$ instead since priority is given to drone 1. In b), there is no priority among the drones. Drone 1 arriving later needs to allow drone 2 enter $k1$ safely.

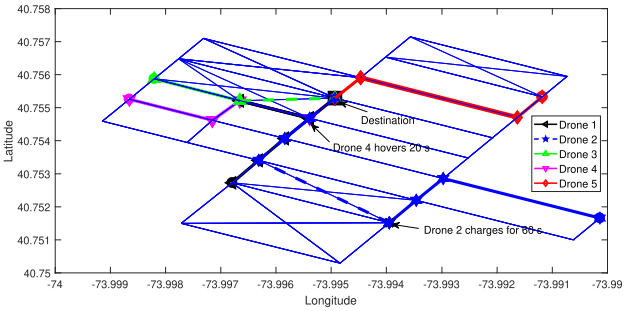


FIGURE 3. Optimal routes selected for 5 drones with different St_j (bold circles) and a common destination (bold square). The charging station is labeled by a black cross. The solid lines indicate the trajectories of the drones over layer 1 while the dashed lines indicate the trajectories of the drones over layer 2.

In Fig. 2, we highlight the differences between the OFIA and SFIA algorithms. In OFIA, the trajectory of drone 1 is already determined and the drone 2 will need to find its trajectory accordingly. Hence, two options are possible at intersection $k1$ either drone 2 will hover so it enters $k1$ after a time gap $\bar{T}$ or it will choose another path segment, e.g., move to a higher layer. In SFIA, the drone 2 arriving first to the intersection $k1$ will have the priority to enter the segment $(k1, k2)$. The drone 1 will then be forced to hover. Both algorithms have their advantages and limitations. For OFIA, assigning priority levels to drones is advantageous in case the operator wants to promote some drones or applications. However, the issue that significant delays can be caused to lowest priority drones. The SFIA determines trajectory in a much fair manner. It is useful for drones executing similar applications. However, to avoid collision, only hovering is possible unlike OFIA where changing the altitude (move to an upper or lower layer) is considered.

VI. RESULTS AND DISCUSSION

In this section, we investigate the three proposed approaches, namely the optimal MILP-based approach, OFIA, and SFIA, for different scenarios and versus different system parameters. We start by describing the simulation environment. Then, we discuss the optimal drone navigation performance for small-scale scenarios. Afterwards, we study the behavior of the fleet using the sub-optimal approaches for larger 3D maps. Finally, we compare the performance of the sub-optimal approaches to the one obtained by the MILP in terms of achieved total travel time and running time.

A. SIMULATION PARAMETERS

In our simulations, we consider different areas of the borough of Manhattan in New York city, NY, USA. The data of the off-line map containing the blocks, building heights, and road network are extracted from the collaborative project: Open Street Map.[2] We assume that two layers are made available for the navigation of the fleet unless otherwise stated. The altitudes of layers 1 and 2 are 15 and 25 meters, respectively.[3] The MILP is solved using Gurobi/YALMIP software on Matlab with an optimality gap of 0.05% and infinite CPU time limit.

We test the developed drone navigation program as well as iterative algorithms using different number of drones having common and different starting positions and destinations. We assume that there is one charging station located at the first layer of the map and we identify in the figures by a black cross. We set the maximum capacity of all drones to 10500 Joule (around 3 Wh),[4] and we assume that the initial battery of each drones is 50% of the total battery capacity unless otherwise stated. The charging power is chosen to be 40 W. We set the safety gap $\bar{T}$ as 10 seconds. We assume that the different actions taken by the drones consume different amounts of energy: Hence, we assume that the drone consumes 120 W to vertically move to an upper layer, 60 W when moving in the same layers, and 30 W when moving down to a lower layer. We also assume that the drone consumes 10 W when it is statically hovering.

B. OPTIMAL DRONE NAVIGATION

In Fig. 3, we present an example of the optimal routes of $D = 5$ drones in an area centered in Time Square, New York city, with a radius of 500 meters. Two layers are considered $L = 2$. We notice that the trip of the second drone is different from its direct shortest path between its starting position and the destination as this drone is forced to go to the charging station located at the South of the map to reload its battery for 60 seconds. This extra amount of energy is sufficient for the drone to resume its trip and reach the destination. Another remark is that this drone, directly after leaving the charging station, has chosen to move to layer 2 so it can diagonally cross the block at a higher altitude to reach the destination faster. The results also indicate that drone 4 has

[2]https://www.openstreetmap.org

[3]For clarity and tractability, we choose to provide simulation results in relatively small areas with a limited number of layers. This also allows to have a higher collision risk when a small fleet size is considered.

[4]The used battery is 25% of the Panasonic NCR 18650B Lithium Ion (http://dronesarefun.com/BatteriesForUAV.html)

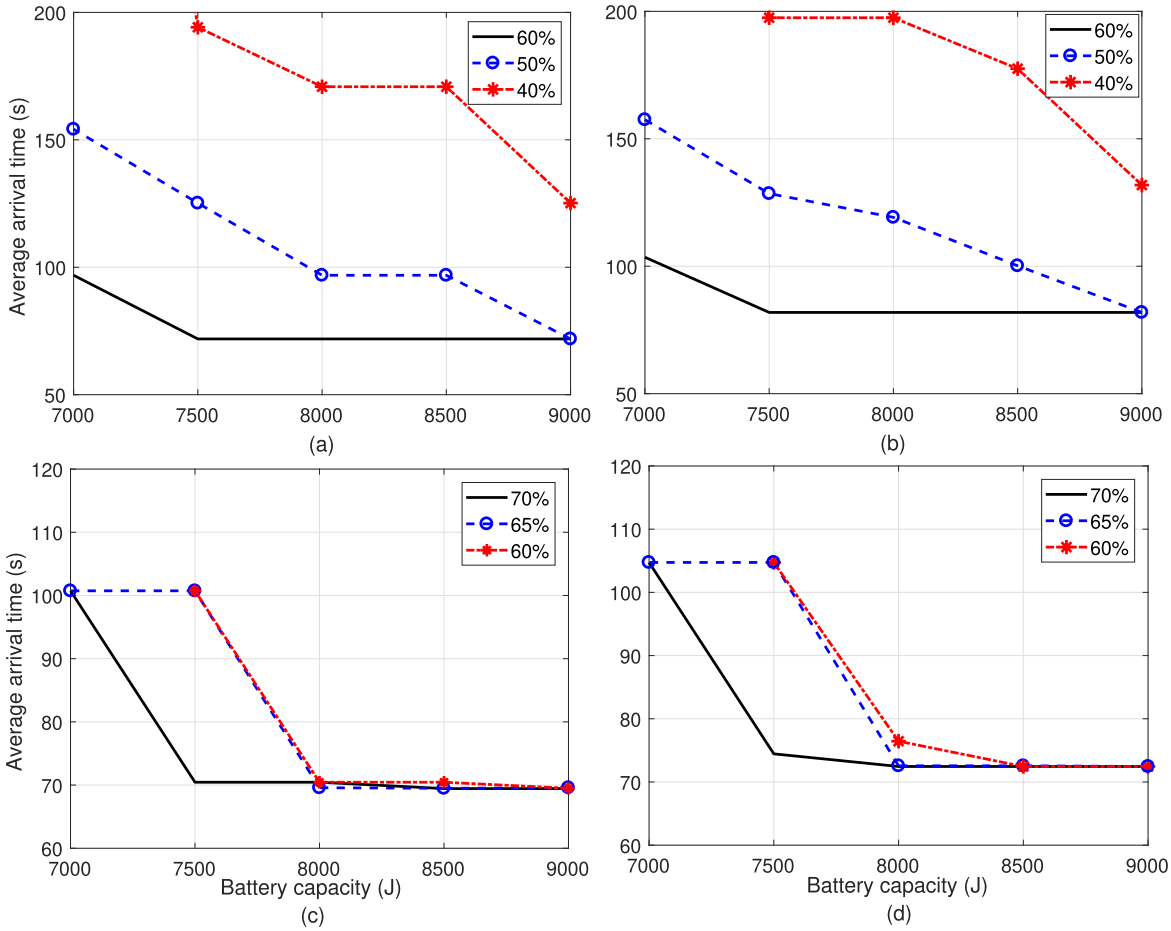


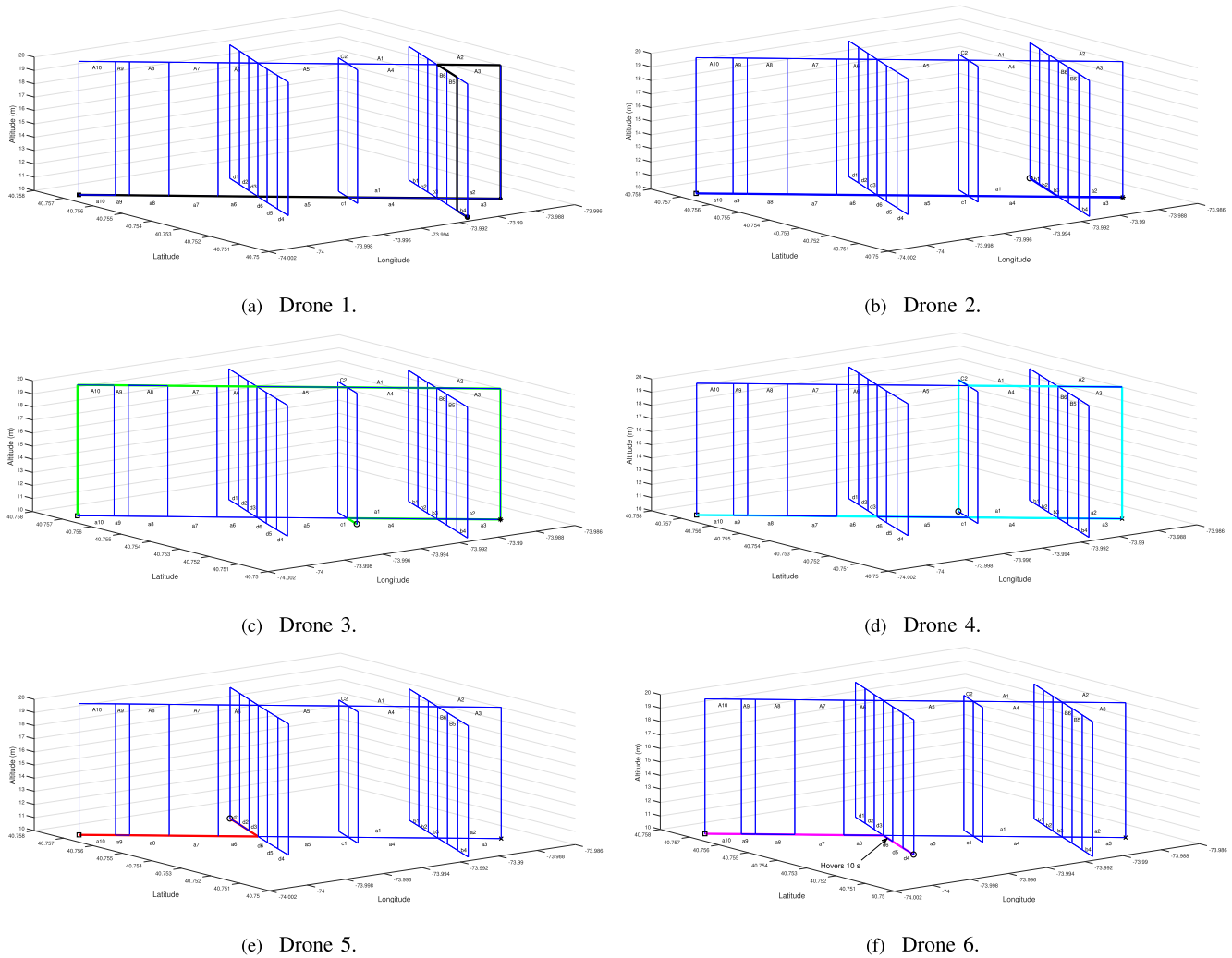
FIGURE 4. Average travel time with different fleet size D, different safety gap $\bar{T}$, different battery capacities $\bar{C} = \bar{C}_j$ and initial battery levels $b^0 = b_j^0$: (a) $D = 3$ and $\bar{T} = 10$ s, (b) $D = 3$ and $\bar{T} = 20$ s, (c) $D = 5$ and $\bar{T} = 5$ s, and (d) $D = 5$ and $\bar{T} = 10$ s.

TABLE 3. Total travel time of the fleet, the values between parenthesis are the travel times of the first drone.

	Without Hovering		With Hovering	
	$L = 1$	$L = 2$	$L = 1$	$L = 2$
$D = 1$	78.25 (78.25)	78.25 (78.25)	78.25 (78.25)	78.25 (78.25)
$D = 2$	156.05 (79.36)	150.03 (79.36)	150.02 (79.36)	150.02 (79.36)
$D = 3$	274.42 (99.25)	266.57 (89.92)	238.34 (85.32)	237.95 (87.21)
$D = 4$	392.19 (98.93)	380.06 (108.26)	336.82 (98.74)	336.14 (98.49)

chosen a longer path to avoid collision with drone 3 at layer 2. Finally, drone 4 has statically hovered for 20 seconds to avoid collision with drone 4 and drone 5 before reaching the destination.

In Table 3, we provide the arrival times of the drones using one or two layers and with or without the hovering option (e.g., multi-rotor versus fixed-wing drones). Here, for each scenario (D and L), we provide the sum of the arrival times of the D drones using a Monte Carlo simulation by averaging over 50 different samples where different starting positions are selected. We also provide the average arrival times of the first drone (between parenthesis). This allows us to identify the delay caused by the congestion in the map. By comparing the values of each column, we notice that the average travel time of drone 1 rises with the increase of the fleet size, since to avoid collision, the drone 1 needs to change its trajectory or statically hover (if enabled). Adding an extra layer with a higher altitude helps in reducing the travel time especially for the case without hovering. In that case, instead of completely changing their trajectories, the drones choose to fly at higher altitude. In the hovering case, the drone prefers to hover as it saves time and consumes less energy. The advantage of adding layers with different altitudes is clear in Table 3 where for example, with $D = 4$, the total time is reduced by 12 seconds. With hovering, the total time is reduced by 44 seconds. It should be noted that for the case without hovering when ($L = 1/D = 4$), drone 1 'has scarified' its trip for the benefit of the fleet. In many generated samples, infeasibility solutions are observed for the hovering-free scenario especially with the one layer map as it is easily to be trapped into a collision. The probability of collision is around 5% and this will trivially increase as more drones are involved. Thanks to the hovering property, the risk of collision in urban areas is significantly reduced.



(a) Drone 1. (b) Drone 2.

(c) Drone 3. (d) Drone 4.

(e) Drone 5. (f) Drone 6.

FIGURE 5. Special case: Trajectories of six drones in a small area with limited path segments and two layers. The starting point, destination, and the charging station are marked by a circle, a square, and a cross, respectively.

In Fig. 4, we analyze the impact of the battery parameters, i.e. battery capacity and initial battery levels, on the drone traveling time. The x-axis represents the capacity of drones' batteries $\bar{C} = \bar{C}_j, \forall j$, the y-axis indicates the average travel time for single drones. The initial battery levels are given in the legends. For example, 60% indicates that the initial battery level is 60% of the total battery capacity. Fig. 4(a) and Fig. 4(b) show that, for a small battery capacity and initial battery level, the arrival time significantly increases as the drones will need to recharge their batteries many times on their ways to the destination. Indeed, by increasing the initial battery level from 40% to 50%, the charging time has dropped by more than 180 seconds when $\bar{C} = 8000$ J. Increasing the values of theses parameters relaxes the problem and provides better results. In some particular cases, where the initial battery is 40% and the battery capacity is less than 7000 J, the program returns infeasibility since at least one of the drones does not have sufficient energy to reach the charging station. Increasing the battery by 10% may help in reducing the arrival time by more than 40% and by more than 65% for an initial battery positive shift of 20%. This time saving trivially depends on several other factors including the drones' starting positions and the locations of the charging stations. From the rest of the figures, we can see that reducing the time arrival gap leads to a better time saving, however, in practice, it may lead to a higher collision rate.

Fig. 5 to Fig. 6, we investigate a very special case where limited path segments are made available to the drones. We aim to visualize the optimal trajectories selected for ($D = 6$) drones in a simple 2-layer road map. We employ different colors to represent different drones. The starting positions (black circles) are intentionally chosen in a symmetric way for each pair of drone to increase the collision risk. The destinations for all the drones are co-located at the extreme left of the figure (black square), while the charging station is located at the extreme right of the figure (black cross). Hence, if its energy is not sufficient to go directly to the destination, the drone will be forced to return back to

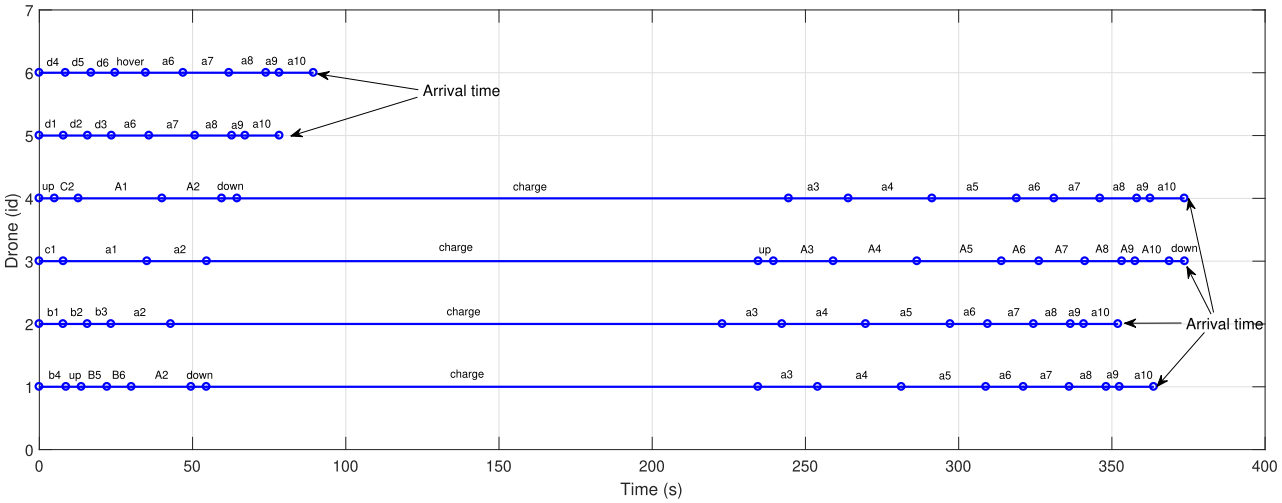


FIGURE 6. Special case: Time schedule of the whole fleet. The length of each segment (linking two circles) indicates the time needed to perform an action. The labels on each line (e.g., a1) indicate the path segments crossed by a drone. "hover" indicates the hovering action. "charge" indicates that a drone is charging its battery at the charging station. "up" and "down" indicate that a drone is moving to the upper layer or to a lower layer.

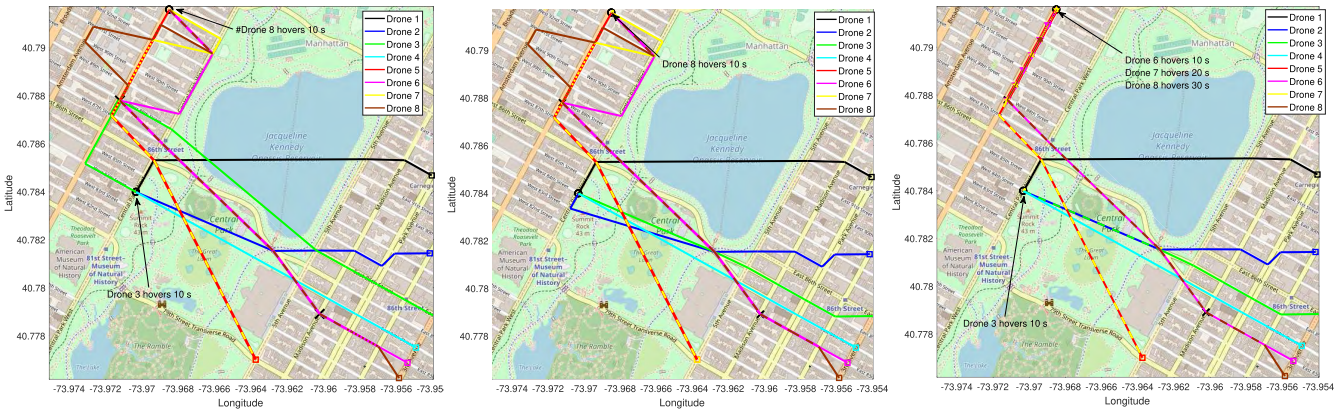
the charging station to replenish its battery. From Fig. 5a to Fig. 5b, we provide the exact trajectory of each drone in the 3D map while in Fig. 6, we provide the exact actions (time schedule) taken by each drone at each time instant. We can notice that four of the six drones went back to the charging station to recharge their batteries. All of them have almost the same charging time needed to go directly to the charging station. The first drone (black) chooses to go to the second layer then go to the charging station while the second drone (blue) chooses to go to the charging station directly over layer 1 (chose its fastest path). The reason is that both drones will collide if they choose their shortest paths. Similar remark can be noticed to drone 3 (green) and drone 4 (cyan). For drone 5 and drone 6, as they are close to the destination, there is no need that they go to recharge, so both of them head directly to the destination through the first layer. However, drone 6 decides to hover for 10 seconds at the intersection to let drone 5 goes first. Fig. 6 shows that the arrival times of the different drones are different except for drone 5 and drone 6. Their arrival times are almost the same as each one of them is coming from different path segments (layer 2 $\rightarrow$ layer 1 (down) for drone 3 and a10 for drone 4).

C. SUB-OPTIMAL DRONE NAVIGATION APPROACHES

In this section, we evaluate the performance of the proposed heuristic approaches and compare their behaviors in determining the trajectory for each member of the fleet.

In Fig. 7, we provide different selected trajectories for a fleet of drones of size 8 using the OFIA based on two different ordering schemes, i) fastest route order (Fig.7a) and ii) energy consumption order (Fig. 7b) as well as the SFIA (Fig. 7c). There are eight drones flying from two starting positions (black circles) and heading to seven different destinations (colored squares). Two charging stations exist in the region (black crosses). This 1-layer map depicts the area around central park, Upper Manhattan. The priorities of the first ordering scheme rank the drones as follows: 2, 1, 4, 3, 5, 6, 7, and 8 where drone 2 has the higher priority, while the priorities of the second ordering scheme rank the drones as follows: 3, 4, 1, 2, 5, 6, 7, and 8. If we take the case of drone 3 (green), that has the highest priority in Fig. 7b, we notice that it heads directly to its destination without hovering. But, in Fig. 7a, where it is ranked forth, the OFIA assigns to it a longer route to prevent its collision risk with drone 2. Hence, its expected energy consumption is higher, so drone 3 heads to the charging station first then flies to the destination. On the contrary, drone 2 (blue) heads to the destination directly in Fig. 7a, while to modify its trajectory in Fig. 7b. Similar remarks can be noticed for the other drones. In Fig. 7c, we notice that, with the SFIA, there is no presumed order. So, to avoid collision, drone 5 moves first, then drone 6 hovers 10 seconds, and drone 7 hovers 20 seconds, and finally drone 8 hovers 30 seconds. All of them head to the charging station first and then go to their respective destinations. In SFIA, the fastest trajectories of all drones remain unchanged but hovering is mandatory to prevent collision. It is also worth to note that many path segments are available when crossing the central park due to its the possibility to fly at low altitude between all the surrounding intersections.

In Fig. 8, we compare between the performances of the OFIA for different ordering schemes and the ones of SFIA. We explore different fleet size $D = 5$ and $D = 10$ and we run a Monte Carlo simulation to record the average arrival time achieved by each algorithm. We find out that, on average, SFIA outperforms the OFIA. The reason is that manually assigning orders to the drones may lead to less suboptimal results. Indeed, it is true that the prioritized drones are promoted but the order may negatively affect the trajectories of the rest of the fleet. Hence, there might be some situations where, with OFIA, some drones need to hover more time



(a) OFIA based on travel time. (b) OFIA based on energy consumption. (c) SFIA.

FIGURE 7. Trajectories of eight drones using the heuristic approaches in the area of upper Manhattan. Starting points and destinations are located at the west and east of the Central Park.

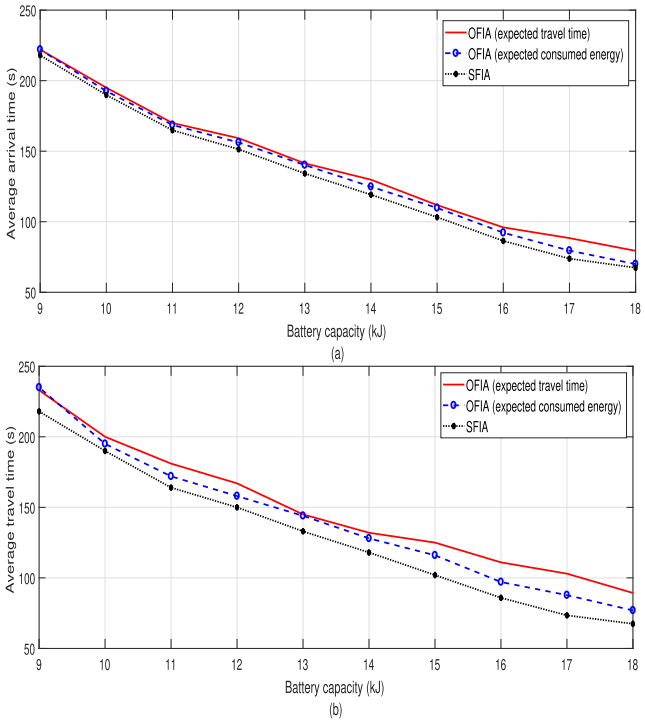


FIGURE 8. Average travel time using different heuristic approaches for $\bar{T} = 10$ s with (a) $D = 5$ and (b) $D = 10$.

compared to SFIA where drones are treated in a fair manner. Moreover, the flexibility of hovering is set to be high (the drone can hover as much as it is needed) so it is rare to find cases where the travel time of a new selected path is shorter than the old path plus hovering in intersections. Nevertheless, the idea of assigning priorities to certain drones remains beneficial in some special missions such as emergency response applications. Fig. 8 also shows that, on average, ordering the drones based on their expected energy consumption achieves slightly better results than ordering them based on their expected arrival time.

TABLE 4. Total running time in seconds of the optimal and heuristic approaches using three, five, and ten drones.

	$D = 3$	$D = 5$	$D = 10$
Optimal	552.0473	894.0574	> 1000
OFIA	1.812	2.8809	4.0247
SFIA	1.2669	1.6547	1.9706

D. COMPARISON BETWEEN OPTIMAL AND HEURISTIC APPROACHES

In Fig. 9, we explore the performances of the heuristic approaches and compare them to the ones of the MILP-based solution for different fleet size and battery properties. It is shown that heuristic approaches achieve close average arrival time compared to the MILP-based solution in most of the cases, especially for high battery capacity levels. Indeed, with higher battery properties (higher battery capacity and/or initial battery levels), more flexibility in managing the fleet is offered. Hence, the heuristic approaches have more chance to find better solutions and it is less required for them to head to the charging station. Hence, the need to go to intermediate stops for the drones negatively affect the performance of the heuristic algorithms since the modified Dijkstra algorithm (Procedure 1) for OFIA and the Dijkstra algorithm for SFIA will be executed more than once. In some cases, one of the algorithm fails to reach close performance to the ones achieved by the MILP. This is subject to several factors including the conceptual construction of each algorithm and the initial statuses of the drones, e.g., their initial locations or initial battery levels. In such scenarios, the drone operator can test both heuristic approaches and select the most effective one.

In terms of convergence speed, Table 4 evaluates the running time of all proposed approaches. It can be seen that suboptimality of the heuristic approaches is compensated by an extremely fast computational speed compared to the one of the MILP-based solution, which requires hundreds to convergence. We can conclude that heuristic approaches can

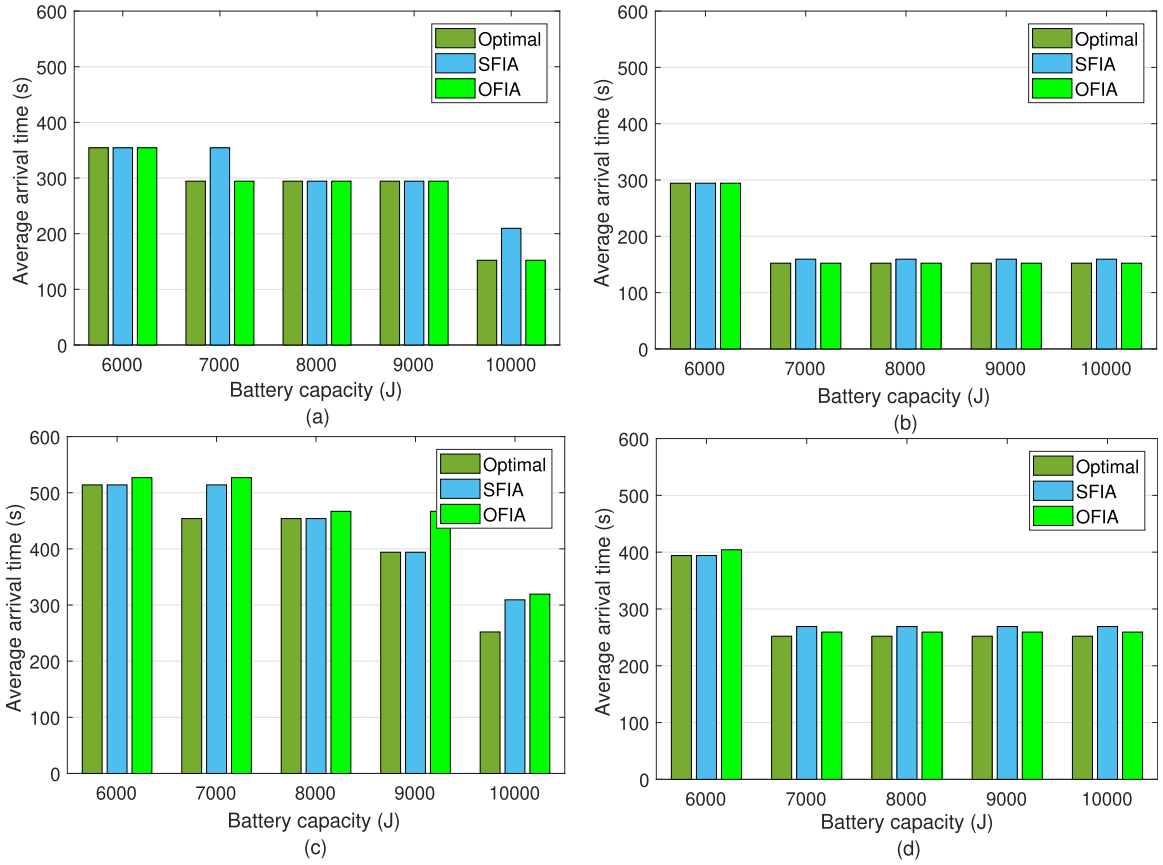


FIGURE 9. Comparison of the achieved average travel time, heuristic approaches versus optimal solution: (a) $D = 3$ and $b^0 = 0.4\bar{B}$, (b) $D = 3$ and $b^0 = 0.6\bar{B}$, (c) $D = 5$ and $b^0 = 0.4\bar{B}$, and (d) $D = 5$ and $b^0 = 0.6\bar{B}$.

be effectively employed for real-time scenarios where quick collision-free navigation can be determined for large fleet of drones. The OFIA is slower than SFIA since it is based on the modified Dijkstra's algorithm which more complex than the simple Dijkstra's algorithm executed by the SFIA. Note also the impact of increasing the drone density on both heuristic algorithms. By increasing D from three to five and then to ten, the running time of OFIA increased by around 1 and 1.2 seconds, respectively while the SFIA running time constantly increases by around 0.4 seconds only. This is because, with OFIA, increasing the drone fleet or the drone density will lead to higher collision risk and hence, the modified Dijkstra's algorithm proposed in Procedure 1 will need more iterations to converge, which delays the OFIA convergence.

VII. CONCLUSION AND FUTURE RESEARCH DIRECTIONS

In this paper, we have developed a generic framework ensuring collision-free navigation for a fleet of drones in urban areas. This is performed while taking into account the energy properties of the drones and their ability to reload their batteries in charging stations made available within the area of interest. Collision is avoided by enabling stationary hovering, altitude changing, and/or re-routing. A MILP is first formulated to determine the optimal and fastest routes to be chosen by each drone. Then, heuristic approaches with significantly faster running time are developed to find sub-optimal solutions. Our results have illustrated the obtained trajectories and time plans of the fleet members for different scenarios. They have also shown how certain drones modify their trips for the benefits of their peers. The results also explored and compared between the performance of the different proposed approaches in realistic 3D maps. The proposed framework can be used to manage large-scale of drones executing similar or different smart city applications while guaranteeing safe navigation and avoiding battery depletion.

Addressing the different challenges related to the drone navigation is primordial for enhancing the quality of experience and safety of smart city applications' users. In addition to the battery limitation and collision issues, many other open challenges can still be confronted in practice. For instance, real-time solutions, instead of proactive approaches, are needed to better enhance the navigation of drones. Hence, in case of unexpected events or missions, the drones can quickly react and determine new trajectories using centralized or decentralized schemes. Artificial intelligence can be employed to empower the flying units by autonomous and self-navigation technology that allow them make decisions on the fly. Finally, in urban areas, drones can exploit public transportation to navigate such an environment [40], [41]. Hence, navigation could be designed in accordance to the bus schedules so that drones can land over a bus and then take off when needed. This helps in saving energy and even allow

drones recharge their batteries while they are on the top of a transportation unit. In our ongoing and future work, we will focus on such solutions allowing better navigation of drones in urban areas and test them in real-world environment.

ACKNOWLEDGMENT

This paper was presented in part at the IEEE Vehicular Technology Conference (VTC'19-Spring), Kuala Lumpur, Malaysia, April–May, 2019 [1]. The statements made herein are solely the responsibility of the authors.

REFERENCES

[1] X. Wan, H. Ghazzai, Y. Massoud, and H. Menouar, "Optimal collision-free navigation for multi-rotor UAV swarms in urban areas," in *Proc. IEEE Veh. Technol. Conf. (VTC Spring)*, Kuala Lumpur, Malaysia, Apr./May 2019, pp. 1–5.

[2] F. Mohammed, A. Idries, N. Mohamed, J. Al-Jaroodi, and I. Jawhar, "UAVs for smart cities: Opportunities and challenges," in *Proc. Int. Conf. Unmanned Aircraft Syst. (ICUAS)*, Orlando, FL, USA, May 2014, pp. 267–273.

[3] (2019). *38 Ways Drones Will Impact Society: From Fighting War to Forecasting Weather, UAVs Change Everything*. [Online]. Available: https://www.cbinsights.com/research/drone-impact-society-uav/

[4] H. Menouar, I. Guvenc, K. Akkaya, A. S. Uluagac, A. Kadri, and A. Tuncer, "UAV-enabled intelligent transportation systems for the smart city: Applications and challenges," *IEEE Commun. Mag.*, vol. 55, no. 3, pp. 22–28, Mar. 2017.

[5] E. Oswald. (Mar. 2017). *Amazon's Drone Delivery Project, Prime Air*. Accessed: May 5, 2019. [Online]. Available: https://www.digitaltrends.com/cool-tech/amazon-prime-air-delivery-drones-history-progress/

[6] D. Chow. (May 2018). *Uber Just Unveiled a Prototype of Its Futuristic Air Taxi*. Accessed: May 5, 2019. [Online]. Available: https://www.nbcnews.com/mach/science/uber-just-unveiled-prototype-its-futuristic-air-taxi-ncna872771

[7] M. Mazur, A. Wisniewski, and J. McMillan, "Clarity from above: PwC global report on the commercial applications of drone technology," PricewaterhouseCoopers Int. Limited, London, U.K., Tech. Rep., May 2016. [Online]. Available: https://www.pwc.pl/pl/pdf/clarity-from-above-pwc.pdf

[8] H. Ghazzai, H. Menouar, and A. Kadri, "On the placement of UAV docking stations for future intelligent transportation systems," in *Proc. IEEE Veh. Technol. Conf. (VTC Spring)*, Sydney, NSW, Australia, Jun. 2017, pp. 1–6.

[9] H. Ghazzai, A. Kadri, M. B. Ghorbel, H. Menouar, and Y. Massoud, "A generic spatiotemporal UAV scheduling framework for multi-event applications," *IEEE Access*, vol. 7, pp. 215–229, Jan. 2019.

[10] K. Sundar and S. Rathinam, "Algorithms for routing an unmanned aerial vehicle in the presence of refueling depots," *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 1, pp. 287–294, Jan. 2014.

[11] J. Kim, B. D. Song, and J. R. Morrison, "On the scheduling of systems of UAVs and fuel service stations for long-term mission fulfillment," *J. Intell. Robotic Syst.*, vol. 70, nos. 1–4, pp. 347–359, Apr. 2013.

[12] R. Ke, Z. Li, S. Kim, J. Ash, Z. Cui, and Y. Wang, "Real-time bidirectional traffic flow parameter estimation from aerial videos," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 4, pp. 890–901, Apr. 2017.

[13] Y. Zeng, J. Xu, and R. Zhang, "Energy minimization for wireless communication with rotary-wing UAV," *IEEE Trans. Wireless Commun.*, vol. 18, no. 4, pp. 2329–2345, Apr. 2019.

[14] Y. Zhou, N. Cheng, N. Lu, and X. S. Shen, "Multi-UAV-aided networks: Aerial-ground cooperative vehicular networking architecture," *IEEE Veh. Technol. Mag.*, vol. 10, no. 4, pp. 36–44, Dec. 2015.

[15] H. Ghazzai, A. Feidi, H. Menouar, and M. L. Ammari, "An exploratory search strategy for data routing in flying ad hoc networks," in *Proc. IEEE Annu. Int. Symp. Pers. Indoor Mobile Radio Commun. (PIMRC)*, Montreal, QC, Canada, Oct. 2017, pp. 1–7.

[16] G. Gankhuyag, A. P. Shrestha, and S.-J. Yoo, "Robust and reliable predictive routing strategy for flying ad-hoc networks," *IEEE Access*, vol. 5, pp. 643–654, 2017.

[17] S.-J. Yoo, J.-H. Park, S.-H. Kim, and A. Shrestha, "Flying path optimization in UAV-assisted IoT sensor networks," *ICT Express*, vol. 2, no. 3, pp. 140–144, Sep. 2016.

[18] Q. Yang and S. Yoo, "Optimal UAV path planning: Sensing data acquisition over IoT sensor networks using multi-objective bio-inspired algorithms," *IEEE Access*, vol. 6, pp. 13671–13684, Mar. 2018.

[19] H. Yu, K. Meier, M. Argyle, and R. W. Beard, "Cooperative path planning for target tracking in urban environments using unmanned air and ground vehicles," *IEEE/ASME Trans. Mechatronics*, vol. 20, no. 2, pp. 541–552, Apr. 2015.

[20] P. Yang, K. Tang, J. A. Lozano, and X. Cao, "Path planning for single unmanned aerial vehicle by separately evolving waypoints," *IEEE Trans. Robot.*, vol. 31, no. 5, pp. 1130–1146, Oct. 2015.

[21] D. Kim, L. Xue, D. Li, Y. Zhu, W. Wang, and A. O. Tokuta, "On theoretical trajectory planning of multiple drones to minimize latency in search-and-reconnaissance operations," *IEEE Trans. Mobile Comput.*, vol. 16, no. 11, pp. 3156–3166, Nov. 2017.

[22] D. Jang, H. Chae, and H. Choi, "Optimal control-based UAV path planning with dynamically-constrained TSP with neighborhoods," in *Proc. Int. Conf. Control, Autom. Syst. (ICCAS)*, Jeju, Korea, Oct. 2017, pp. 373–378.

[23] X. Liu, C. Zhou, and M. Ding, "3D multipath planning for UAV based on network graph," *J. Syst. Eng. Electron.*, vol. 22, no. 4, pp. 640–646, Aug. 2011.

[24] S. Majumder and M. S. Prasad, "Three dimensional D* algorithm for incremental path planning in uncooperative environment," in *Proc. Int. Conf. Signal Process. Integr. Netw. (SPIN)*, Noida, India, Feb. 2016, pp. 431–435.

[25] W. Li, W. Chen, C. Wang, M. Liu, Y. Ge, and Q. Song, "A 3D path planning approach for quadrotor UAV navigation," in *Proc. IEEE Int. Conf. Inf. Autom. (ICIA)*, Yunnan, China, Aug. 2015, pp. 2481–2486.

[26] L. Yang, J. Qi, Z. Jiang, D. Song, J. Han, and J. Xiao, "Guiding attraction based random tree path planning under uncertainty: Dedicate for UAV," in *Proc. IEEE Int. Conf. Mechatronics Autom. (ICMA)*, Aug. 2014, pp. 1182–1187.

[27] A. Dixit, A. Mishra, and A. Shukla, "Vehicle routing problem with time windows using meta-heuristic algorithms: A survey," in *Harmony Search and Nature Inspired Optimization Algorithms* (Advances in Intelligent Systems and Computing), vol. 741, N. Yadav, A. Yadav, J. Bansal, K. Deep, and J. Kim, Eds. Singapore: Springer, Jan. 2019, pp. 539–546.

[28] C. Zheng, L. Li, F. Xu, F. Sun, and M. Ding, "Evolutionary route planner for unmanned air vehicles," *IEEE Trans. Robot.*, vol. 21, no. 4, pp. 609–620, Aug. 2005.

[29] K. Dorling, J. Heinrichs, G. G. Messier, and S. Magierowski, "Vehicle routing problems for drone delivery," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 47, no. 1, pp. 70–85, Jan. 2017.

[30] K. Kuru, D. Ansell, W. Khan, and H. Yetgin, "Analysis and optimization of unmanned aerial vehicle swarms in logistics: An intelligent delivery platform," *IEEE Access*, vol. 7, pp. 15804–15831, Jan. 2019.

[31] M. Garraffa, M. Bekhti, L. Létocart, N. Achir, and K. Boussetta, "Drones path planning for WSN data gathering: A column generation heuristic approach," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Barecelona, Spain, Apr. 2018, pp. 1–6.

[32] A. Majd, A. Ashraf, E. Troubitsyna, and M. Daneshtalab, "Using optimization, learning, and drone reflexes to maximize safety of swarms of drones," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Rio de Janeiro, Brazil, Jul. 2018, pp. 1–8.

[33] S. Ahmed, A. Mohamed, K. Harras, M. Kholief, and S. Mesbah, "Energy efficient path planning techniques for UAV-based systems with space discretization," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Doha, Qatar, Apr. 2016, pp. 1–6.

[34] J. Zhang, J. Yan, P. Zhang, and X. Kong, "Collision avoidance in fixed-wing UAV formation flight based on a consensus control algorithm," *IEEE Access*, vol. 6, pp. 43672–43682, Aug. 2018.

[35] M. A. Dulebenets, "Application of evolutionary computation for berth scheduling at marine container terminals: Parameter tuning versus parameter control," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 1, pp. 25–37, Jan. 2018.

[36] N. Tsiogkas and D. M. Lane, "An evolutionary algorithm for online, resource-constrained, multivehicle sensing mission planning," *IEEE Robot. Autom. Lett.*, vol. 3, no. 2, pp. 1199–1206, Jan. 2018.

[37] M. Sniedovich, "Dijkstra's algorithm revisited: The dynamic programming connexion," *Control Cybern.*, vol. 35, no. 3, pp. 599–620, 2006.

[38] M. J. Bannister and D. Eppstein, "Randomized speedup of the Bellman–Ford algorithm," in *Proc. Meeting Analytic Algorithmics Combinat. (ANALCO)*, Kyoto, Japan, Jan. 2012, pp. 41–47.

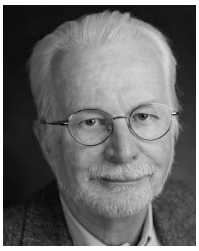
[39] D. B. Johnson, "Efficient algorithms for shortest paths in sparse networks," *J. ACM*, vol. 24, no. 1, pp. 1–13, 1977.
[40] A. Trotta, F. D. Andreagiovanni, M. D. Felice, E. Natalizio, and K. R. Chowdhury, "When UAVs ride a bus: Towards energy-efficient city-scale video surveillance," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Honolulu, HI, USA, Apr. 2018, pp. 1043–1051.
[41] N. Lasla, H. Ghazzai, H. Menouar, and Y. Massoud, "Exploiting land transport to improve the UAV's performances for longer mission coverage in smart cities," in *Proc. IEEE Veh. Technol. Conf. (VTC-Spring)*, Kuala Lumpur, Malaysia, Apr./May 2019, pp. 1–7.

AHMED BAHABRY received the degree in chemical engineering from Michigan Technological University, Houghton, MI, USA, in 2010, and the master's degree in systems engineering from the Stevens Institute of Technology, Hoboken, NJ, USA, in 2012, where he is currently pursuing the Ph.D. degree. His research interests include smart city design, unmanned aerial vehicles, and optimization.

XIANGPENG WAN (SM'19) received the bachelor's degree in electrical engineering from the Harbin Institute of Technology, Harbin, China, in 2015, and the master's degree in applied mathematics from the University of Minnesota, Minneapolis, MN, USA, in 2017. He is currently pursuing the Ph.D. degree in engineering management with the Stevens Institute of Technology, Hoboken, NJ, USA. His research interests include smart city design, big data analysis, and applied machine learning.

HAKIM GHAZZAI (S'12–M'15) received the Diplôme d'Ingénieur degree (Hons.) in telecommunication engineering and the master's degree in high-rate transmission systems from the École Supérieure des Communications de Tunis (SUP'COM), Tunis, Tunisia, in 2010 and 2011, respectively, and the Ph.D. degree in electrical engineering from KAUST, Saudi Arabia, in 2015. He is currently a Research Scientist with the Stevens Institute of Technology, Hoboken, NJ, USA. Before joining Stevens, he was a Visiting Researcher with Karlstad University, Sweden, and a Research Scientist with the Qatar Mobility Innovations Center (QMIC), Doha, Qatar, from 2015 to 2018. His general research interests include intersection of wireless networks, UAVs, the Internet of Things, intelligent transportation systems, and optimization. Since 2019, he has been on the Editorial Board of the IEEE COMMUNICATIONS LETTERS.

HAMID MENOUAR (SM'16) received the Engineering degree in computer science from the University of Sciences and Technology Houari Boumediene, Algiers, Algeria, in 2003, the D.E.A. (M.S.) degree in systems and information technologies from the University of Technology of Compiègne, Compiègne, France, in 2004, and the Ph.D. degree in computer science from Télécom ParisTech, École Nationale Supérieure des Télécommunications, Paris, France, in 2008. From 2005 to 2010, he was with Hitachi Europe, France, as a Researcher, and then as a Lead of the Cooperative Systems Team. In 2010, he moved to Qatar and joined the Qatar Mobility Innovations Center (QMIC), where he is currently a Senior R&D Expert and a Product Manager, leading different projects and initiatives in the areas of connected and automated vehicles, intelligent transport systems, unmanned aerial vehicles, the Internet of Things, smart mobility, and so on.



GREGG VESONDER received the B.A. degree in psychology from the University of Notre Dame and the M.S. and Ph.D. degrees in cognitive psychology from the University of Pittsburgh. He is currently an Industry Professor and the Program Director of software engineering with the Stevens Institute of Technology. Prior to that, he was the Executive Director of the Mobile and Pervasive Systems Research Department, AT&T Labs-Research. He has developed and managed numerous artificial intelligence projects. He has presented papers in both IJCAI and AAAI. He has published papers on empirical psychological research involving in topics, such as meta-cognition and expert novice knowledge differences supported by mathematical models. He has published several papers on simulations addressing the evolution of evolvability. He also has managed organizations involved in e-commerce (music), C++ compiler development, speech recognition and text-to-speech, and software design and analysis. He was named a Bell Labs Fellow and an AT&T Fellow for his work on artificial intelligence.

YEHIA MASSOUD (F'15) received the Ph.D. degree in electrical engineering and computer science from the Massachusetts Institute of Technology, Cambridge, MA, USA. He has held several industrial and academic positions, including a member of the Technical Staff with the Advanced Technology Group, Synopsys, Inc., Mountain View, CA, USA, a tenured Associate Professor with the Departments of Electrical and Computer Engineering and Computer Science, Rice University, Houston, TX, USA, the W. R. Bunn Head of the Department of Electrical and Computer Engineering, University of Alabama at Birmingham, Birmingham, AL, USA, and the Head of the Department of Electrical and Computer Engineering, Worcester Polytechnic Institute, Worcester, MA, USA. He is currently the Dean of the School of Systems and Enterprises, Stevens University of Science and Technology, Hoboken, NJ, USA. He leads research efforts in various areas of electrical and computer engineering, computing, and systems and software engineering. He has authored over 225 papers in peer-reviewed journals and conferences. He was an elected member of the IEEE Nanotechnology Council, from 2009 to 2011. He was selected as one of the ten MIT Alumni featured in the MIT EECS Newsletter, in 2012. He was a recipient of the Rising Star of Texas Medal, in 2007. He was also a recipient of the Synopsys Special Recognition Engineering Award, in 2000, the National Science Foundation CAREER Award, in 2005, the DAC Fellowship in 2005, several Best Paper Award nominations, and two Best Paper Awards at the IEEE International Symposium on Quality Electronic Design, in 2007, and the IEEE International Conference on Nanotechnology, in 2011. He was named a Distinguished Lecturer by the IEEE Circuits and Systems Society, from 2014 to 2015. He has served as the Theme Leader of Novel Interconnects and Architectures in the SRC Southwest Academy of Nanoelectronics, from 2006 to 2011.