Optimized landing of drones in the context of congested air traffic and limited vertiports

Zhenyu Zhou, Jun Chen and Yanchao Liu

***Abstract*—Drone fleet operators must be able to land the whole fleet in short notice. In practical operations, landing spots are usually much fewer than airborne drones. When many drones gravitate toward the limited landing spots simultaneously, congestion management becomes a challenge. This paper characterizes the fleet landing problem using mixed integer programming techniques and proposes a series of computational enhancements to reduce the solution time from hours to seconds. The solution algorithms are implemented in a software prototype for traffic management, and are thoroughly validated via extensive numerical examples and field simulations. For a fleet of 18 drones navigating at the same altitude layer within a 4-square kilometer area, all routing and trajectory computations can be completed in less than 5 seconds, and the entire fleet is able to complete landing at three pre-planned landing spots within about 3 minutes. Therefore, the models and algorithms are suitable for practical use.**

***Index Terms*—Unmanned Traffic Management, Routing, Optimization, Trajectory Planning, Mixed Integer Programming**

I. Introduction

Large-scale, organized deployment of unmanned aerial vehicles (UAVs or drones) can unlock substantial efficiency gain in transportation systems. Currently, there are mounting needs from a number of industries for the ability to transport goods and people for short distances (i.e., intra-city trips) in the low-altitude airspace, which call for a safe and efficient airspace management mechanism. Many regulatory advancements have been made in the U.S. in recent years, including a series of Congress bills [1], [2], proposed rulemakings by the Federal Aviation Administration (FAA) [3], [4], and Concept of Operations developed by FAA and the National Aeronautics and Space Administration (NASA) [5], [6].

Existing industry-led unmanned aircraft system (UAS) traffic management (UTM) developments focus more on the communication and information technology (C&IT) layer, such as networked discovery and synchronization, communication and interoperability standards among different UAS Service Suppliers (USS) [7], [8]. In contrast, the resource optimization aspects of traffic management is less discussed. It is imperative to fill this gap timely for several reasons. First, as the UAS is increasingly deployed, the airspace will become a scarce resource, and the ability to optimally plan and utilize this resource under dense heterogeneous traffic presents an essential requirement on the infrastructure. Second, the resource optimization mindset and approach should be assessed early in the UTM rulemaking process, as it would be much harder and costlier to mend inefficient mandates than making them efficient in the first time.

According to Deloitte [9], one of the biggest hurdles for urban air transportation is the cost and complexity involved in building adequate "vertiplaces" - a collective term for vertiports, vertihubs and vertistations that serve as pickup and drop-off sites for people and cargo transported by vertical takeoff and landing (VTOL) aerial vehicles - to support widespread operations within cities. It is foreseeable that in the near future the UTM system will consist of many more aircraft than landing facilities. Consequently, the problem of optimally sequencing and spacing aircraft for landing, i.e., [10], [11], will be an important operational challenge in UTM.

In NASA's concept of operations [12], [5], [13], prioritization for public safety and emergency operations is one of the five core operating principles of UTM. This means that delivery fleets must be able to vacate the airspace in short notice to make way for high-priority operations. In addition, the weather condition is more volatile at low altitudes than at higher altitudes (e.g., in the stratosphere where commercial airliners cruise), and will have more impact on small UAS. A sudden change in weather, for example, would require a whole airborne fleet to land as soon as possible. In these circumstances, the problem of efficiently landing a large number of drones at a limited number of depots (vertiports) is a realistic operational problem. We call it a *multi-port capacitated fleet landing (MCFL)* problem. Here, the term "capacitated" means limited capacity of both the airspace and the vertiports. The problem is depicted in Figure 1.

In this paper, we first formulate the MCFL problem as a mixed integer program (MIP) by applying some rigor-

Z. Zhou is with the Department of Industrial & Systems Engineering, Wayne State University, Detroit, MI, USA.

J. Chen is with the Department of Aerospace Engineering, San Diego State University, San Diego, CA, USA.

Y. Liu (corresponding author) is with the Department of Industrial & Systems Engineering, Wayne State University, Detroit, MI, USA. Email: yanchaoliu@wayne.edu

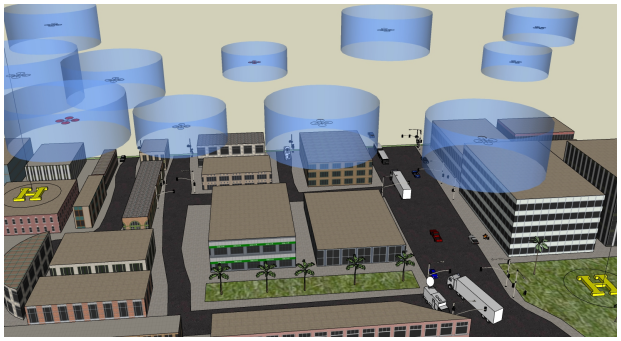

Fig. 1: Illustration of high-density urban air traffic having limited vertiport capacity.

ously derived linearization schemes for motion, proximity and collision avoidance constraints in the Euclidean space. Then, we propose a series of decomposition, parallel computing and constraint generation heuristics to significantly speed up the solution process, making it suitable for real-time use in fairly dense traffic scenarios. Lastly, a set of comprehensive computational studies is presented to showcase the effectiveness of the proposed solution methods.

Aircraft routing in congested airspace has been investigated extensively in the air traffic control (ATC) literature. Prominent methods include multicommodity network flow models [14], queuing network models [15], [16], mixed integer linear and nonlinear programming models [17], [18], semidefinite programming methods [19] and agent-based approaches [20]. While most of these methods are general enough to be applicable to UTM, in reality drone traffic presents some unique characteristics that require special treatments. For instance, as flying robots, drones have super-human response time and maneuverability and hence can tolerate smaller separation (or headway) distances; drone traffic can be omnidirectional, unfettered by fixed routes or airway corridors; multicopter and VTOL drones have unique equations of motion compared to fixed-wing aircraft typically studied in the ATC literature. To deal with these specialties, [21] proposed a progressive motion planning method based on nonlinear optimization that was able to centrally route all drones to their respective destinations in an arbitrarily dense traffic. However, in that work the airspace was assumed to be a 2D Euclidean plane with the vertical dimension omitted. For instance, the time it takes for a drone to land vertically as well as the capacity of the vertical landing corridor was ignored. In [22], a network-free mixed integer programming formulation was proposed to model dynamic drone routing problems in an Euclidean plane. In this model, although drone actions were modeled at a greater granularity to account for various business requirements in drone delivery, the UTM aspects of the problem, in particular the collision avoidance constraints, were unaddressed. The omission of important UTM considerations seems to be a common shortcoming in recent drone delivery operational models, which are mostly rooted in the literature of conventional (i.e., ground-based) vehicle routing problems (VRP), e.g., [23], [24], [25], [26]. This paper attempts to bridge the gap by explicitly modeling inter-drone separation constraints in an MIP framework. The main innovations are summarized as follows.

- We provide the first attempt at addressing the multi-port capacitated fleet landing problem from an optimization perspective, and develop a comprehensive MILP formulation by employing several innovative modeling techniques.
- Based on realistic considerations, we decompose the problem into two sequential subproblems, and develop a predictive congestion-based assignment method and a variable-fixing heuristic to solve these problems efficiently. The speedup is up to 500 folds.
- We implement the algorithms in an in-house fleet management software App and demonstrate its practical usefulness on real drone platforms using software-in-the-loop simulations.

The rest of the paper is organized as follows. In Section II, we propose the main optimization model for the fleet landing problem, and develop all-time separation bounds as well as their formulation in the mixed integer programming framework. In Section III, we decompose the decision problem into two main parts and develop a congestion-based depot assignment method to assign drones to landing depots. Section IV describes a set of variable-fixing heuristics to reduce the effective size of the routing model. Section V provides numerical experiments and result analysis, and Section VI concludes the paper with pointers for future research.

II. Main Model

The decision problem is stated as follows: given k depots (vertiports) of fixed geographic locations on the ground and given a temporal snapshot of the air traffic scene, i.e., the spatial distribution of N drones and their current status, determine the landing routes and timing for all drones. The objective is to minimize the total time to land, that is, minimize the sum of airborne time of all drones.

A. Assumptions

We make the following assumptions and considerations about the fleet landing scenario.

- The airspace is layered by altitude. A drone is restricted to fly horizontally within the altitude layer that it starts in, and can make a final vertical descent

TABLE I: Sets and Parameters in the MIP Model

Symbol	Description
$\mathcal{T}$	Set of time steps
$\mathcal{G}$	Set of sides of the approximating polygons
$\mathcal{V}$	Set of vehicles
$\mathcal{D}$	Set of depots
$\mathcal{P}$	Set of relational positions, $\mathcal{P} = \{P_1, ..., P_4\}$
$\mathcal{A}$	Set of actions for a vehicle, $\mathcal{A} = \{1 : fly, 2 : descend, 3 : ground\}$
O_v^X, O_v^Y	Coordinate of veh. v's origin
a_g, b_g, c_g	Coefficients in the g-th linear inequality
R_v	Maximum speed of veh. v
Q_d^X, Q_d^Y	Coordinate of depot d's location
Q_d^C	Capacity of depot d
h, l	Headway time and landing time, respectively
M, M^X, M^Y	Large constants of appropriate values

TABLE II: Decision Variables in the MIP Model

Symbol	Description
z_{vd}^A	$= 1$ if veh. v is assigned to depot d
$z_{v_1v_2tp}^S$	$= 1$ if v_1 and v_2 are in position p at time t
z_{vat}^{ACT}	$= 1$ if veh. v is performing action a at time t
z_{vdt}^{DEC}	$= 1$ if veh. v is descending at depot d at time t
x_{vt}, y_{vt}	Coordinate of veh. v at time t
x'_{vt}, y'_{vt}	Speed of veh. v at time t
x_v^D, y_v^D	Coordinate of veh. v's destination
x_{vt}^{DD}, y_{vt}^{DD}	Veh. v's distance to destination at time t
$x_{v_1v_2t}^{DP}, y_{v_1v_2t}^{DP}$	Distance between veh. v_1 and v_2

only when it is straight above a landing depot. Therefore, we assume that over the course of the landing process, a drone will perform three actions (or transition across three states) sequentially: 1. **fly** to a depot location, 2. **descend** vertically, 3. remain on the **ground**.

- The vertical landing corridor of a depot is capacitated, only one drone can be in the corridor at any given time.
- Each drone is surrounded by a safety disc of a radius proportional to its maximum speed and a headway time tolerance. The safety discs of any pair of drones in the same altitude layer should not overlap.

We adopt the following conventions in the rest of the paper: (1) The terms drone and vehicle are used interchangeably; (2) While the sets, parameters and variables used in the main model are summarized in notation tables, other symbols are defined inline at first use. For space limitation, proofs are omitted here but are available upon reader's request.

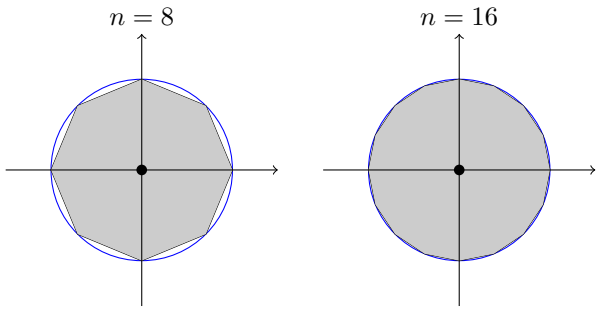

Fig. 2: Inner polyhedral approximations of a circular area.

B. Formulation

We will list out relevant constraints, and the rationale behind their formulation.

1) Objective Function: To make all drones land as quickly as possible, we will minimize the total airborne time, or equivalently, maximize the total time on the ground, that is,

$$\text{Maximize} \sum_{v\in\mathcal{V}, t\in\mathcal{T}} z_{v3t}^{ACT} \tag{1}$$

2) Equations of Motion: The location of a vehicle at time t is the result of uniform linear motion starting from the location at time $t-1$:

$$x_{vt} = x_{vt-1} + x'_{vt},\ \forall v \in \mathcal{V}, t \in \mathcal{T}, t \neq 1 \tag{2}$$

$$y_{vt} = y_{vt-1} + y'_{vt},\ \forall v \in \mathcal{V}, t \in \mathcal{T}, t \neq 1 \tag{3}$$

The velocity of a vehicle can be different in different time intervals, but within the same time interval, the velocity is a fixed vector (x'_{vt}, y'_{vt}) whose length is no more than the maximum speed of the vehicle R_v. We linearize the quadratic constraint by approximating a circular area using its n-sided inscribing polygon, illustrated in Figure 2.

$$a_g x'_{vt} + b_g y'_{vt} \leq c_g R_v,\ \forall v \in \mathcal{V}, t \in \mathcal{T}, g \in \mathcal{G} \tag{4}$$

The coefficients a_g, b_g and c_g describing the inequality of the sides $g \in \mathcal{G}$ of a unit polygon can be calculated easily, see [22] for details.

3) Drone State and Action Constraints: Each vehicle must be assigned to a depot and the assignment is invariant over time, that is, the vehicle-depot assignment, once made, will not change over the course of the landing execution process.

$$\sum_{d\in\mathcal{D}} z_{vd}^A = 1,\ \forall v \in \mathcal{V} \tag{5}$$

The destination location of a vehicle is set to the location of the depot the vehicle is assigned to.

$$x_v^D = \sum_{d\in\mathcal{D}} z_{vd}^A Q_d^X,\ \forall v \in \mathcal{V} \tag{6}$$

$$y_v^D = \sum_{d\in\mathcal{D}} z_{vd}^A Q_d^Y,\ \forall v \in \mathcal{V} \tag{7}$$

A vehicle can be in the "descend" mode ($z_{v2t}^{ACT} = 1$) only when it reaches its destination. The following set of constraints is used to enforce this relation.

$$x_{vt}^{DD} \geq x_{vt} - x_v^D,\ \forall v \in \mathcal{V}, t \in \mathcal{T} \tag{8}$$
$$x_{vt}^{DD} \geq x_v^D - x_{vt},\ \forall v \in \mathcal{V}, t \in \mathcal{T} \tag{9}$$
$$y_{vt}^{DD} \geq y_{vt} - y_v^D,\ \forall v \in \mathcal{V}, t \in \mathcal{T} \tag{10}$$
$$y_{vt}^{DD} \geq y_v^D - y_{vt},\ \forall v \in \mathcal{V}, t \in \mathcal{T} \tag{11}$$
$$x_{vt}^{DD} + y_{vt}^{DD} + (M^X + M^Y) z_{v2t}^{ACT} \leq M^X + M^Y,\ \forall\, v \in \mathcal{V}, t \in \mathcal{T} \tag{12}$$

A vehicle must be in one of the three states at any time.

$$\sum_{a\in\mathcal{A}} z_{vat}^{ACT} = 1,\ \forall v \in \mathcal{V}, t \in \mathcal{T} \tag{13}$$

A vehicle can descend at a depot only if it is assigned to this depot.

$$z_{vdt}^{DEC} \leq z_{vd}^A,\ \forall v \in \mathcal{V}, d \in \mathcal{D}, t \in \mathcal{T} \tag{14}$$

A vehicle can be descending at depot d at time t only if it is in the descend state. Conversely, when a vehicle is in the descend state, it must be actually descending at one of the depots.

$$z_{vdt}^{DEC} \leq z_{v2t}^{ACT},\ \forall v \in \mathcal{V}, d \in \mathcal{D}, t \in \mathcal{T} \tag{15}$$
$$\sum_{d\in\mathcal{D}} z_{vdt}^{DEC} = z_{v2t}^{ACT},\ \forall v \in \mathcal{V}, t \in \mathcal{T} \tag{16}$$

The number of vehicles descending at a depot at any given time cannot exceed the corridor capacity at the depot.

$$\sum_{v\in\mathcal{V}} z_{vdt}^{DEC} \leq Q_d^C,\ \forall d \in \mathcal{D}, t \in \mathcal{T} \tag{17}$$

For a multicopter drone, the vertical descending speed is much smaller than the horizontal cruise speed for safety reasons. If the descending stage takes l time periods, then it means that if a drone enters the descending state at time t, it must remain in this state in the next l periods.

$$z_{v2t}^{ACT} - z_{v2t-1}^{ACT} \leq z_{v2t'}^{ACT},\ \forall (t,t') : t < t' \leq t + l \tag{18}$$

A vehicle can be in the grounded state ($a = 3$) at time t only if it was in either the grounded state or the descending state in the previous time period $t-1$, and a vehicle that has reached the grounded state must remain in the grounded state in subsequent time periods.

$$z_{v3t}^{ACT} \leq z_{v3t-1}^{ACT} + z_{v2t-1}^{ACT},\ \forall v \in \mathcal{V}, t \in \mathcal{T} \tag{19}$$
$$z_{v3t}^{ACT} \geq z_{v3t-1}^{ACT} \tag{20}$$

These constraints ensure that the state sequence of (fly, descend, grounded) is properly maintained.

4) Inter-vehicle Separation: Collision avoidance constraints require that every pair of vehicles navigating in the same altitude layer must be at least a certain distance apart horizontally at all times. In the 2D space, this requirement leads to nonconvex quadratic constraints (Ref. [21]). Specifically, let (x_v, y_v) be the location coordinate of vehicle v at a given time, then for any two vehicles (say 1 and 2) flying at the same altitude, the collision avoidance constraint is expressed as

$$(x_1 - x_2)^2 + (y_1 - y_2)^2 \geq S \tag{21}$$

where S is the minimum separation distance. As a first step of integrating it into an MIP model, we want to replace constraint (21) by the L_1 form as follows

$$|x_1 - x_2| + |y_1 - y_2| \geq S' \tag{22}$$

As a substitute for (21), constraint (22) is desired to be both *safe* and *efficient*, whereas safe means that (22) should unconditionally imply (21) and efficient means that S' should be minimal. What should S' be?

Proposition 1. *Let (x_1, y_1) and (x_2, y_2) be the coordinates of two points in $\mathbb{R}^2$, and let S be a given constant. Then, the minimum value of S' that makes (22) $\Rightarrow$ (21) is $S' = \sqrt{2}S$.*

This proposition is a straightforward corollary of the known result (e.g., Cauchy-Schwarz inequality, or equation (2.2.5) in [27]) that if $x \in \mathbb{R}^n$, then $\|x\|_1 \leq \sqrt{n}\|x\|_2$, so we will omit the proof.

In an algebraic model, collision avoidance will be enforced at discrete time points. For instance, constraint (22) will actually take the form

$$|x_{1,t} - x_{2,t}| + |y_{1,t} - y_{2,t}| \geq S',\ \text{for each } t \in \mathcal{T} \tag{23}$$

Unfortunately, this constraint says nothing about the intervehicle distance between successive time points t and $t+1$. Could the separation requirement be violated during the interval, and by how much? Answering this question requires taking the vehicles' motion and speed into consideration. Let $\Delta_t \in \mathbb{R}^2$ be the vector that points from the coordinate of vehicle 1 to that of vehicle 2, thus the left side of equation (23) is equal to $\|\Delta_t\|_1$. Assume that the vehicles travel at a fixed velocity during the time interval and let δ_t be the relative velocity of the two vehicles during the time interval, then the length of δ_t is bounded by the maximum relative speed $R := R_1 + R_2$, where R_v denotes the maximum speed of vehicle v. In other words, we have $\|\delta_t\|_2 \leq R$, and thus $\|\delta_t\|_1 \leq \sqrt{2}R$. Equation of motion implies that $\Delta_{t+1} = \Delta_t + \delta_t$. Then the above questions translate to the inquiry "do $\|\Delta_t\|_1 \geq S'$ and $\|\Delta_{t+1}\|_1 \geq S'$ guarantee $\|\Delta_{t+\alpha}\|_1 \geq S'$ for any $\alpha \in [0,1]$?" or equivalently, what is the minimum value of $\|\Delta_t + \alpha\delta\|_1$ for $0 \leq \alpha \leq 1$,

given $\|\Delta_t\|_1 \geq S'$, $\|\Delta_{t+1}\|_1 \geq S'$ and $\|\delta_t\|_1 \leq \sqrt{2}R$? The following proposition answers this inquiry.

Proposition 2. *Given constants $\hat{S}, R \geq 0$, let $\Delta_t, \delta_t \in \mathbb{R}^n$ with $\|\delta_t\|_1 \leq \sqrt{2}R$ for $t = 0, 1, 2, \ldots$, and define $\Delta_s = \Delta_{\lfloor s \rfloor} + (s - \lfloor s \rfloor)\delta_{\lfloor s \rfloor}$, for $s \in \mathbb{R}^+$. Then the algebraic constraint $\|\Delta_t\|_1 \geq \hat{S}$, $t = 0, 1, 2, \ldots$ implies that the all-time distance $\|\Delta_s\|_1$ satisfies $\|\Delta_s\|_1 \geq max(0, \hat{S} - \sqrt{2}R/2)$ for all $s \in \mathbb{R}^+$.*

The proof is based on KKT conditions of a nonlinear optimization problem, similar to the proof of Theorem 1 in [21]. We will omit the proof here. Combining Proposition 1 and 2, we have that, in order to enforce (21) in continuous time, we need $\sqrt{2}S = \hat{S} - \sqrt{2}R/2$, which gives $\hat{S} = \sqrt{2}(S + R/2)$. Thus it is sufficient to enforce

$$|x_{1,t} - x_{2,t}| + |y_{1,t} - y_{2,t}| \geq \sqrt{2}(S + R/2), \forall t \in \mathcal{T} \quad (24)$$

In practice, the all-time separation distance S is usually specified in terms of the number of time intervals. For example, we may require that any two vehicles must be h seconds apart even in the worst case scenario, i.e., the two vehicles traveling head to head at maximum speeds. In this case, S can be written as hR. So the above constraint can be rewritten as

$$|x_{1,t} - x_{2,t}| + |y_{1,t} - y_{2,t}| \geq h'R, \forall t \in \mathcal{T} \quad (25)$$

where $h' := \sqrt{2}(h + 1/2)$.

At this point, we have a set of inequalities (25) to efficiently describe the inter-vehicle separation requirements for any pair of drones, and guarantee all-time separation. However, these inequalities contain the absolute value function which cannot be used in an MIP model. Moreover, only when both drones are in the fly mode ($a = 1$) should we enforce the separation constraint. To address these issues, we implement (25) for each pair of drones by a set of linear constraints.

$$\begin{aligned} &x_{v_1t} - x_{v_2t} + y_{v_1t} - y_{v_2t} + 3M \geq \\ &Mz^S_{v_1v_2tp_1} + Mz^{ACT}_{v_11t} + Mz^{ACT}_{v_21t} + h'(R_{v_1} + R_{v_2}), \\ &\forall\, v_1, v_2 \in \mathcal{V}, v_1 \neq v_2, t \in \mathcal{T} \end{aligned} \quad (26)$$

$$\begin{aligned} &x_{v_2t} - x_{v_1t} + y_{v_1t} - y_{v_2t} + 3M \geq \\ &Mz^S_{v_1v_2tp_2} + Mz^{ACT}_{v_11t} + Mz^{ACT}_{v_21t} + h'(R_{v_1} + R_{v_2}), \\ &\forall\, v_1, v_2 \in \mathcal{V}, v_1 \neq v_2, t \in \mathcal{T} \end{aligned} \quad (27)$$

$$\begin{aligned} &x_{v_1t} - x_{v_2t} + y_{v_2t} - y_{v_1t} + 3M \geq \\ &Mz^S_{v_1v_2tp_3} + Mz^{ACT}_{v_11t} + Mz^{ACT}_{v_21t} + h'(R_{v_1} + R_{v_2}), \\ &\forall\, v_1, v_2 \in \mathcal{V}, v_1 \neq v_2, t \in \mathcal{T} \end{aligned} \quad (28)$$

$$\begin{aligned} &x_{v_2t} - x_{v_1t} + y_{v_2t} - y_{v_1t} + 3M \geq \\ &Mz^S_{v_1v_2tp_4} + Mz^{ACT}_{v_11t} + Mz^{ACT}_{v_21t} + h'(R_{v_1} + R_{v_2}), \\ &\forall\, v_1, v_2 \in \mathcal{V}, v_1 \neq v_2, t \in \mathcal{T} \end{aligned} \quad (29)$$

$$\sum_{p \in \mathcal{P}} z^S_{v_1v_2tp} = 1,\ \forall v_1, v_2 \in \mathcal{V}, v_1 \neq v_2, t \in \mathcal{T} \quad (30)$$

In essence, equation (25) corresponds to one of the four cases given in (26) - (29), depending on the spatial relation between v_1 and v_2, and equation (30) selects one from the four possible cases.

The overall MIP model is as follows.

$$\begin{aligned} &\text{Maximize} \sum_{v \in \mathcal{V}, t \in \mathcal{T}} z^{ACT}_{v3t} \\ &\text{Subject to } (2) - (20), (26) - (30) \end{aligned}$$

Note that the model's objective function will incentivize all drones to reach the "grounded" state as soon as possible, and ensure that once the "grounded" state is reached, the state will remain "grounded" until the end of the planning time horizon $\mathcal{T}$. Admittedly, models that maximize only one of the variables are intricate in terms of constraints. In other words, the solutions have to be constructed and improved in an implicit way, forced by the objective function, integrality constraints and the dependence between the constraints. It is therefore difficult to utilize meta-heuristic methods to solve this kind of models. Systematic branch-and-bound search, as implemented in commercial MIP solvers, will be relied upon.

III. Decomposition

While the MIP model links all decisions in a unified framework which provides a gold standard in terms of solution optimality, it is difficult to solve even for commercial solvers, due to the large search space and the lack of exploitable structure in the objective function. The solution times for several MIP instances are listed in Table V under column "v" (for vanilla). Therefore the MIP model cannot be used directly in practice. More efficient methods need to be devised. To do so, we approach the solution from a human dispatcher's perspective while making reasoning and algorithmic design.

A human dispatcher would approach the problem by breaking it into a series of simpler tasks: assigning vehicles to landing depots, arbitrating the landing sequence of vehicles assigned to the same depot, and determining non-intersection trajectories.

A. Vehicle-Depot Assignment by Distance

A straightforward method for vehicle-depot assignment is to direct each drone to fly to and land at the nearest depot, i.e., assignment by distance. In solving the problem, we can first calculate the distance from a vehicle's current (starting) position to each depot and assign the vehicle to the nearest depot. The assignment allows us to fix z^A_{vd} to 1 for the assigned (v, d) pairs. This method is intuitive and can speed up the solution of the MIP model. However, a major problem with this method is that if

the fleet is not evenly distributed, assignment by distance may not be the optimal solution. Furthermore, computing time for trajectory deconfliction for highly congested traffic (as would result from suboptimal vehicle-depot assignment) is significantly higher than that for a milder situation.

B. Vehicle-Depot Assignment by Congestion

While distance is reasonably the main consideration for vehicle-depot assignment, adjustments must be made by considering different depots' expected congestion levels. To overcome the limitation of vehicle-depot assignment by only considering distance, we propose the following optimization-based adjustment methods. Here and onward, we assume that vehicles that are in the same altitude layer have the same maximum speed, i.e., R_v is the same for all $v \in \mathcal{V}$.

Depot assignment will affect the total landing time in two aspects: the congestion-free travel time and the congestion time. The essence of congestion is that the most efficient passage of a vehicle is blocked by another vehicle due to collision avoidance constraints. To approximate the actual travel time to depot d, we separate the assignment model into two parts: congestion-free travel time, denoted by T_d^D, and time incurred by congestion, denoted by T_d^C. The former is simply the time distance to the assigned destination on the ground, that is, for each depot $d \in \mathcal{D}$,

$$T_d^D = \sum_{v \in \mathcal{V}} z_{vd}^A (D_{vd}^\tau + l) \tag{31}$$

where D_{vd}^τ is the (possibly fractional) time distance between vehicle v and depot d. The latter is the number of time periods blocked by other vehicles assigned to the same depot. The objective of the assignment (z_{vd}^A) is to minimize the sum of the two parts over all drones, that is,

$$\text{Minimize} \sum_{d \in \mathcal{D}} (T_d^D + T_d^C) \tag{32}$$

The measurement bases of T_d^D and T_d^C are both time units.

In order to obtain a simple yet accurate approximation for T_d^C, we first divide the space around each depot into N sections. Each section is a ring-shaped area of width R_v centered at the depot (see Figure 5), thus corresponds to one unit of time distance. Let n be the sequence number of each section and $\mathcal{N} = \{1, 2, \ldots, N\}$ be the index set of space sections around a depot. We can then map the starting location of each vehicle in terms of which space section it is in with respect to each depot. Specifically, we define the binary parameter $Z_{vdn} = 1$ if vehicle v is in the n-th section of depot d, and 0 otherwise.

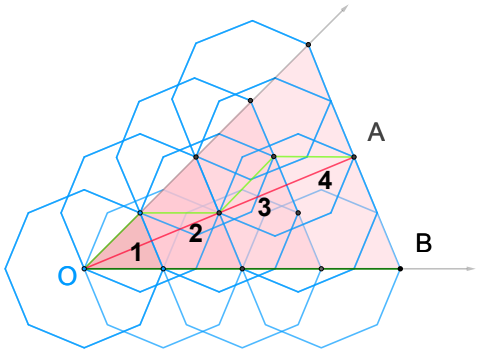

Fig. 3: Illustration of the division of space into circular sections.

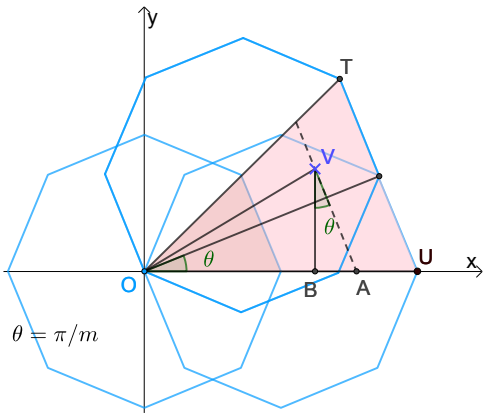

Fig. 4: Calculation of a vehicle's section number based on the approximating octagons.

How shall we determine which section a vehicle is in with respect to a given depot, i.e., the value of Z_{vdn}? Since we use an equilateral polygon to bound the velocity directions in routing, i.e., equation (4), the farthest points that a vehicle can reach in one unit of time are the vertices of the polygon. This is illustrated in Figure 3. Note that the use of 8-sided polygons (octagons) here is for illustration clarity, whereas in actual computing we adopted 16-sided polygons to better approximate the circular area. Centered at point O, space is divided into n concentric octagons by head-way time. For instance, a vehicle in the 4-th section would need four units of time to reach O. The route from A to O can be the green line or the red line, and the route from B to O can be the dark green horizontal line. Despite the differences in travel distance, the three routes would all take four time periods. We derive the method of calculating the section number by use of Figure 4. In the figure, m is the number of sides of the polygon, V marks the vehicle position, and O marks the depot location at $(0, 0)$. Let D_{vd}^τ denote the fractional time distance from vehicle v

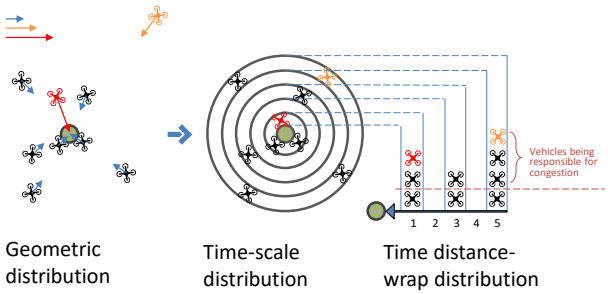

Fig. 5: Mapping 2D vehicle locations into a 1D depot-centered model.

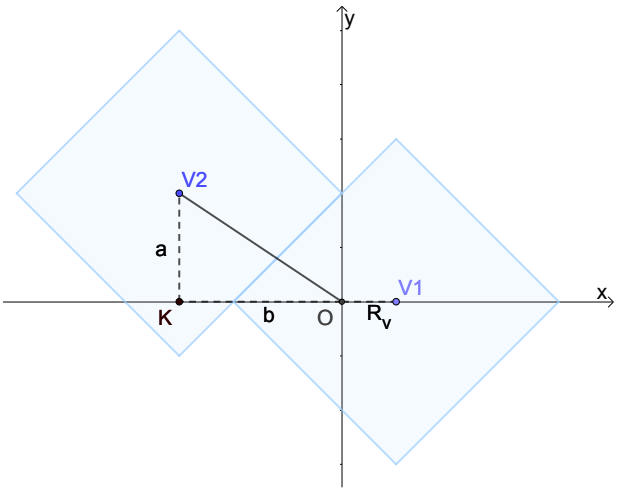

Fig. 6: Illustration for the derivation of equation (34).

to depot d, then

$$D^{\tau}_{vd} = D_{vd}(\cos\angle VOA + \sin\angle VOA \tan\theta) \quad (33)$$

where $D_{vd} = \sqrt{(O^X_v - Q^X_d)^2 + (O^Y_v - Q^Y_d)^2}/R_v$, $\angle VOA = |\text{mod}(\arctan(O^Y_v/O^X_v), 2\theta|$, and $\theta = \pi/m$. The number n of the section where V is located is given by $n = D^T_{vd} := \lceil D^{\tau}_{vd} \rceil$, where D^T_{vd} is defined to be the integer time distance from vehicle v to depot d. The quantity D^T_{vd} will be useful for generating variable fixing constraints in the computational enhancement stage. In Figure 4, $OV = D_{vd}$, $OA = D^{\tau}_{vd}$, $OU = D^T_{vd}$.

Given any two vehicles ready to land in succession at a depot d, we define the *terminal delay time* s as the shortest possible time gap between the landing start times of the two vehicles. It is the bottleneck of the whole landing process at the depot, and is affected by two factors: the landing time l of the first vehicle and the time buffer between the two vehicles due to separation constraints. We will derive an analytical formula using a typical scenario as illustrated in Figure 6. Two vehicles, $V1$ and $V2$, are ready to land at depot O one after another. $V1$ is one time distance (geometric distance is the maximum speed R_v, which is assumed to be the same value for all v) away from O so it lands first. $V1$ will start to land in one unit of time, and $V2$ will start to land in $D^T_{v2,d}$ units of time. Let s' be the time it takes for $V2$ to fly to O after $V1$ starts to land, then $D^T_{v2,d} = s' + 1$. Tight separation constraints would make $a + b + R_v = 2hR_v$, as labeled in the figure, hence $D_{v2,d} = \sqrt{a^2 + b^2}/R_v$. Because the goal is minimizing the time to reach O, i.e., s', the best terminal-stage spatial arrangement for the two vehicles is for $a = b$, or $\angle V2OK = \pi/4$. This is repeatedly observed when the problem is solved using the MIP model. Under this arrangement, we have $D^T_{v2,d} = s' + 1 = \lceil \sqrt{2}(2h-1)/2 \rceil$, and thus $s' = \lceil \sqrt{2}(2h-1)/2 \rceil - 1$. Finally, the terminal delay time s is the greater one of l and s', that is,

$$s = \max\left(l, \left\lceil \sqrt{2}(2h-1)/2 \right\rceil - 1\right) \quad (34)$$

It is worth noting that when $l \gg h$ the corridor capacity will be the dominating bottleneck, in which case the airspace congestion plays a minor role, e.g., vehicles will have more "free" time to move around and line up. In this case, the optimal vehicle-depot assignment would approach a balance by number, that is, each depot receives the same number of vehicles. On the other hand, when $h \gg l$ the problem will approach a pure motion planning problem as studied in [21], and depot assignment can be more suitably investigated by other means of congestion prediction. In this paper, we consider scenarios where l and h are comparable, e.g., experiments are conducted for cases with $l = h$, with the headway time h taking values 1, 2 and 3. These settings can cover most of the practical and difficult cases.

Applying these calculations to each vehicle-depot combination, we can map the 2D geometric distribution of vehicles into a 1D model, as illustrated in Figure 5. In the figure, the depot corridor capacity is one, that is, only one vehicle can be landing at a time. If there is at most one vehicle in each section there will not be any congestion, i.e., $T^C_d = 0$, because all vehicles can advance to the depot synchronously. When some of the sections have more than one vehicle, congestion will ensue. There are two causes for vehicles to experience congestion when assigned to a depot d: congestion due to vehicles in the same sections (denoted by S_d) and congestion due to vehicles in preceding sections (denoted by P_d). Then we have

$$T^C_d = P_d + S_d, \ \forall d \in \mathcal{D} \quad (35)$$

Let variable C_{dn} denote the number of vehicles assigned to depot d which are in the depot's n-th section, and variable K_{dn} denote the number of congestion-inducing vehicles assigned to depot d which are in the depot's n-th section. Under the assumption that the depot landing capacity is one (i.e., $Q^C_d = 1$), each section can have

at most one vehicle that is not congestion-inducing. For each $d \in \mathcal{D}$ and $n \in \mathcal{N}$ we have,

$$C_{dn} = \sum_{v} Z_{vdn} Z^{A}_{vd} \tag{36}$$

$$K_{dn} = \max(C_{dn} - 1, 0) \tag{37}$$

In the minimization problem as (32), we can express equation (37) as

$$K_{dn} \geq C_{dn} - 1 \tag{38}$$

$$K_{dn} \geq 0 \tag{39}$$

We can calculate the congestion due to vehicles in the same sections as the summation of the waiting time of each vehicle in the section, as illustrated in Figure 5. Then we have

$$S_d = \sum_{n \in \mathcal{N}} \frac{(K_{dn} + 1) K_{dn}}{2} s \tag{40}$$

We introduce a multiplier $\Phi_{d,n}$ to denote the units of time delays contributed by vehicles in all preceding sections with respect to section n. For instance, in Figure 5, for the first section we have $\Phi_{d,1} = 0$, and for each subsequent section, the congestion due to preceding sections depends on the number of vehicles in all of those sections. For $n > 1$, similar to equation (37), we have

$$\Phi_{d,n} \geq \Phi_{d,n-1} + C_{d,n-1} s - 1, n > 1 \tag{41}$$

$$\Phi_{d,n} \geq 0 \tag{42}$$

Therefore, we have

$$P_d = \sum_{n \in \mathcal{N}} C_{d,n} \Phi_{d,n} \tag{43}$$

Let us walk through an example in Figure 5 with $h = 2$: The traffic scene show that $C_{dn} = 3, 0, 2, 0$ and 4, respectively for $n = 1, ..., 5$ and $K_{dn} = 2, 0, 1, 0$ and 3, respectively for $n = 1, ..., 5$. Applying (41) in a minimization setting, we have $\Phi_{d,2} = 5$, $\Phi_{d,3} = 4$, $\Phi_{d,4} = 7$ and $\Phi_{d,5} = 6$. The meaning for $\Phi_{d,3} = 4$, for instance, is that each vehicle in section 3 will experience 4 units of time delays due to vehicles in preceding sections.

The vehicle-depot assignment optimization problem is therefore given by the objective (32), and constraints (5), (31), (35) - (36) and (38) - (43). It is a mix-integer nonlinear programming (MINLP) model. For its limited size, the model can be solved by a commercial off-the-shelf solver quite quickly.

IV. Computational Enhancements

Once the vehicle-depot assignment is made, the routing problem can be decomposed by depot, and the depot-specific problem can be solved in parallel. Furthermore, using the aforementioned assignment model, we can also approximate the total landing time of each vehicle. For vehicles assigned to the same depot, we set their landing sequence by distance - vehicles in inner sections land first, and for vehicles in the same section, vehicles that are closer to the depot (i.e., smaller D^{τ}_{vd}) land first. Suppose that vehicle v is assigned to depot d' and it is located in the depot's n'-th section, then the total landing time T^L_v for vehicle v can be approximated by

$$T^L_v = D^T_{vd'} + l + C^N_{vd'n'} s + \Phi_{d',n'} \tag{44}$$

where $C^N_{vd'n'}$ is the sequence number of vehicle v in section n', determined based on $D^{\tau}_{vd'}$. For instance, if there are 3 vehicles in the n'-th section of depot d' and vehicle v has the smallest $D^{\tau}_{vd'}$ value among the three, then $C^N_{vd'n'} = 1$. Note that we used $D^T_{vd'}$, the integer version of $D^{\tau}_{vd'}$, in equation (44) to match the integer time units used in the MIP model. Because the assignment model has been solved at this step, d' and n' for the given vehicle are known. Thus, we can fix several variables in the original MIP model to shrunken the search space and speed up the MIP solution time. The variable fixing heuristic for vehicle v is as follows.

$$z^{DEC}_{vd't} = 0, \ \forall t : t \leq T^L_v - l \tag{45}$$

$$z^{ACT}_{v1t} = 1, \ \forall t : t < T^L_v - l \tag{46}$$

$$z^{ACT}_{v2t} = 1, \ t = T^L_v \tag{47}$$

$$z^{ACT}_{v3t} = 1, \ \forall t : t > T^L_v \tag{48}$$

$$x^{DD}_{vt} \leq R_v(n - l + 1), \ \forall (n, t) : n > l, t + n = T^L_v \tag{49}$$

$$y^{DD}_{vt} \leq R_v(n - l + 1), \ \forall (n, t) : n > l, t + n = T^L_v \tag{50}$$

In practice, the actual total landing time may be different from T^L_v by a small margin. In all experiments, we observed that the variation is less than 2 time units. In case of MIP infeasibility due to over fixing, relaxing the fixing rule by a few time units will solve the problem while preserving the solution speed advantage.

V. Numerical Experiments

In this section, we perform three sets of experiments to demonstrate (1) the advantage and properties of the congestion-based assignment model, (2) the computational efficiency of the overall solution strategy as well as the efficiency gain by each enhancement step, and (3) the efficacy of the overall landing model and algorithm in a practical setting.

A. Computing Environment and Parameter Setting

All experiments were performed on a Dell Precision Tower 3420 with an Intel(R) Core(TM) i7-7700 CPU @ 3.60 GHz, 16.0 GB RAM and Windows 10 Enterprise Operating System. We used GAMS 30.1.0 for modeling,

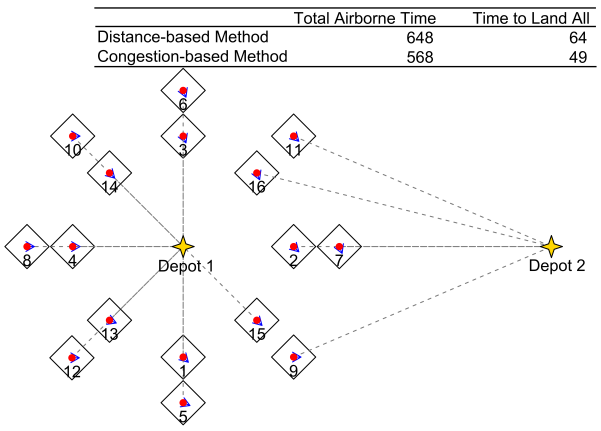

Fig. 7: Experimental scenario for comparing different assignment methods.

SCIP solver for solving the MINLP model (the vehicle-depot assignment model), and GUROBI 9 for solving the MIP model (the original model as well as different variants with the computation enhancements). Default solver options were adopted. In all experiments, we set l equal to h, used 16-sided polygons (see Figure 2) to approximate the velocity range, and used the time horizon $\mathcal{T} = \{1, \ldots, 80\}$.

B. Effects of Congestion-based Assignment Model

To demonstrate the effects of congestion on optimal vehicle-depot assignment, we created an artificial case with two depots and 16 vehicles with uneven locational distribution, as shown in Figure 7.

All vehicles' starting locations were closer to depot 1 than to depot 2, which means they would all be assigned to depot 1 if the assignment were based on distance. Such a distance-based assignment would result in a total airborne time of 648, whereas the congestion-based assignment, which agrees with the optimal assignment obtained from solving the original MIP, gives a much smaller total airborne time of 568. More detailed comparisons are summarized in Table III. We can see that the congestion-based assignment method yielded a more balanced assignment and a smaller objective value for (32). Moreover, the overall solution time using the Congestion-based method is 7.7 s, less than half of the solution time using the Distance-based method. This indicate that a balanced assignment has implicit computational benefits. This point will be demonstrated fully in the next set of computational experiments.

C. Computational Comparison of Solution Methods

In this set of experiments, vehicles are randomly scattered in a 1000×1000 area with a feasible starting

TABLE III: Comparison of Vehicle-Depot Assignment Methods

Distance-based Method

	P_d	S_d	T_d^D	$\sum_v T_v^L$	Airborne
Depot 1	136	168	334.2	648	648
Depot 2	0	0	0		

Congestion-based Method

	P_d	S_d	T_d^D	$\sum_v T_v^L$	Airborne
Depot 1	55	75	226.5	568	568
Depot 2	3	3	197.1		

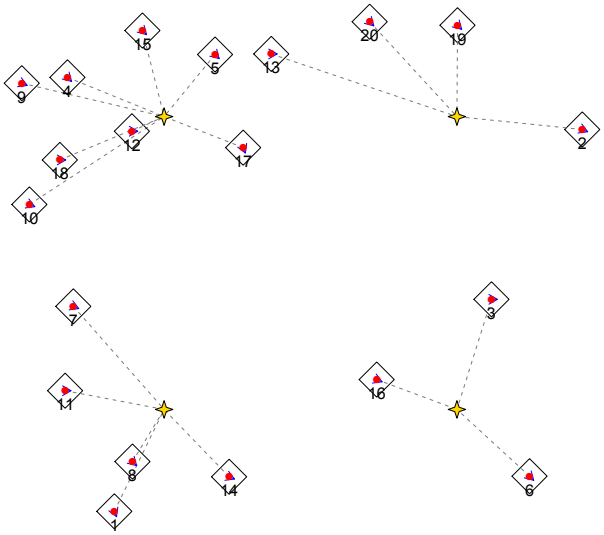

Fig. 8: A random case with $(D, V, H) = (4, 20, 3)$ and optimal depot assignment.

position, i.e., separation constraints are all satisfied in the beginning. In 2-depot scenarios, depots are located at $(250, 500)$ and $(750, 500)$. In 4-depot scenarios, depot locations are $(250, 250)$, $(250, 750)$, $(750, 250)$ and $(750, 750)$. All vehicles' maximum speeds are set to 10 per unit time. In the experiments, we test different numbers of depots (D), numbers of vehicles (V), and separation (headway) time radii (H). For simplicity, we set the vertical landing time equal to headway time in all scenarios. For each parameter combination, 10 random cases are generated. A case with $(D, V, H) = (4, 20, 3)$ and the optimal vehicle-depot assignment is illustrated in Figure 8. In this particular case, V13 is closer to depot 3 while the optimal assignment is assigning it to depot 4, corroborating the deficiency of distance-based assignment method.

We experimented five different solution procedures: (1) Solve the original MIP (vanilla); (2) Pre-assign vehicles to depots by the congestion-based assignment method, and then solve the MIP (vp); (3) Pre-assign based on congestion, then decompose the MIP by depot

TABLE IV: Average Optimality Gaps by Case and Method.

D	V	H	v	vp	vpp	vppd	vppa
2	10	1	0.0	0.0	0.0	0.0	0.0
2	10	2	0.0	0.0	0.0	0.1	0.0
2	10	3	0.0	0.0	0.0	0.7	0.0
2	20	1	0.0	0.0	0.0	0.0	0.0
2	20	2	0.0	0.0	0.0	1.4	0.0
2	20	3	2.4	2.2	2.2	7.5	2.3
4	10	1	0.0	0.1	0.1	0.1	0.1
4	10	2	0.0	0.0	0.0	0.0	0.0
4	10	3	0.0	0.1	0.1	0.1	0.1
4	20	1	0.0	0.0	0.0	0.0	0.0
4	20	2	0.0	0.0	0.0	1.1	0.0
4	20	3	0.0	0.0	0.0	2.5	0.0

TABLE V: Average Solution Time by Case and Method.

D	V	H	v	vp	vpp	vppd	vppa
2	10	1	15.7	14.6	9.3	8.4	3.0
2	10	2	38.7	32.8	11.0	10.1	2.4
2	10	3	73.3	45.1	16.6	25.1	3.3
2	20	1	498.5	176.4	35.2	33.9	4.3
2	20	2	623.6	551.9	78.6	84.6	4.2
2	20	3	4508.0	3224.6	895.8	2185.0	43.5
4	10	1	13.1	11.0	4.3	2.9	3.2
4	10	2	23.1	21.1	4.6	3.1	2.6
4	10	3	25.6	25.9	7.3	4.9	3.8
4	20	1	178.9	58.5	8.8	5.5	4.0
4	20	2	675.4	387.3	29.1	36.3	3.0
4	20	3	1762.0	838.9	53.0	111.5	3.3

and solve separately in parallel (vpp); (4) Pre-assign based on distance, then solve for different depots in parallel (vppd); (5) Pre-assign based on congestion, fix variables by (45) to (50), then solve for different depots in parallel (vppa). For each run, the solver time limit was set to 7200 seconds, and the actual solution time (if completed in time) or the optimality gaps (if time limit was reached) were recorded. Note that the optimality gap of the vanilla (v) method is simply the gap reported by the solver, and the optimality gap of all other methods are calculated by taking the relative gap between the objective value and the optimal objective value obtained by the vanilla method. In other words, the original MIP solution serves as the gold standard for the solution quality of other enhanced (but heuristic) methods.

Table IV lists the average optimality gaps. In the cases of $(D, V, H) = (2, 20, 3)$, the vanilla method reached the time limit in 4 of the 10 cases, which explains the nonzero average optimality gap. We can see that vp, vpp and vppa methods resulted in similar gaps, while the vppd method resulted in much larger gaps due to the suboptimal assignments made by the distance-based

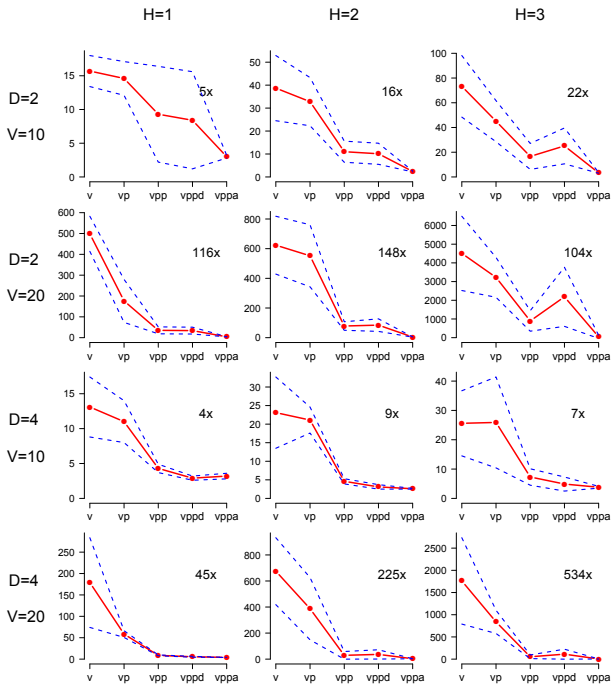

Fig. 9: Comparison of solution time of different methods.

method. Given that the cases are generated randomly, it is quite convincing that the congestion-based assignment method (the MINLP model) prevails and is generally helpful.

Table V lists the average solution time in seconds and Figure 9 exhibits the comparisons in plots. In the figure, the ratio noted in each subplot (e.g., 22x in the top right subplot) is the ratio of vppa to v in solution time, i.e., how many folds of efficiency increase was achieved by the final method vppa. The blue dashed lines outline the 95% confidence interval of nominal solution times for different case and method combinations. We can see that the solution time for vppd is generally greater than vppa, and even greater than vpp in some cases. This is because under the distance-based assignment method (vppd), some depot was overly congested thus difficult to solve. After all, in parallel computing mode, the overall solution time depends on the solution time of the hardest depot. Thus the reduction in solution time is another implicit benefit of using congestion-based assignment method. Notably, vppa exhibits robust performance in all cases with small variances in solution time. Except for extremely difficult case group of $(D, V, H) = (2, 20, 3)$, the vppa method was able to solve each of the other 110 cases in 6 seconds. Therefore, it is suitable for practical use.

D. Simulation in Real Flight Platforms

To demonstrate the practical use of the proposed modes and algorithms (vppa), we implemented the vppa method in a simulation framework built on a real fleet platform consisting of Pixhawk-based quadcopter drones with ArduPilot flight control software. Here, we configured a test flight scenario at the UAS test center located at the Alpena County Regional Airport (KAPN), Alpena, Michigan. The fleet landing procedure discussed in this paper was tested as a safety feature in the presumable context of other fleet flight tests. In an area of 4 square kilometers (shown in Figure 10, we tested a fleet of 18 software-in-the-loop (SITL) drones with three prespecified emergency landing locations (depots). To be on the safe side, we limited each depot's concurrent landing capacity (Q_d^C) to 1. Other model parameters were set as follows: the distance unit used in the model was meter (m), the time interval in the model correspondeds to 10 seconds (s) of clock time, a planning horizon of 80 time intervals (800 s) was used, the vertical landing time was $l = 1$ (10 s), the headway time h was set to 1 (10 s), the maximum speed of each drone was set to 100 (10 m/s), and the navigation commands were sent to drones every 18 s, so that the drones had abundant time to reach the intended waypoints (x_{vt} and y_{vt} coordinates) even if communication latency and motion uncertainty had been present.

The simulation was conducted as follows. First, the whole fleet was commanded to take off to 22 meters above ground level (AGL) and perform normal flight operations. Then, an airspace emergency was presumably declared which required all drones to land as soon as possible. At this point, a "brake" command was sent to all drones to brake all ongoing movements. The drones will hover still while waiting for further navigation commands. In the meantime, the fleet landing optimizer was started. It took the optimizer (vppa method) 4.7 seconds to complete all the computations, which generated step-by-step waypoint sequences for the whole fleet. Then a waypoint command was sent to each drone every 18 s to guide the landing maneuver. The process completed as intended without a problem. All drones successfully landed within 3 minutes and 14 seconds, while sufficient separation was maintained at all times. A link to the software App and a video demo will be available upon request.

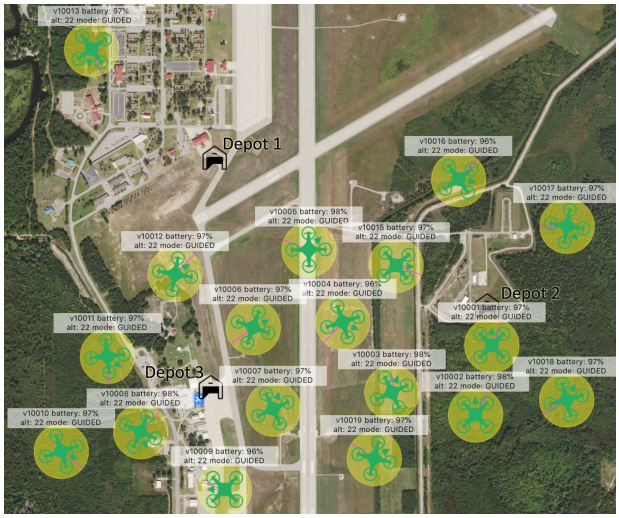

Fig. 10: Locations of 18 drones and 3 landing depots in the testing scenario.

VI. Conclusion

In this paper, we studied the new problem of efficiently landing a fleet of drones at a limited number of highly capacitated vertiports. This problem represents an important aspect of future UTM developments, as well as a unique extension to conventional vehicle routing problems. We proposed a comprehensive MIP model to describe relevant operational constraints, and conducted theoretical analysis to justify all-time separation bounds used in the linearized collision avoidance constraints. To solve realistic cases efficiently without sacrificing optimality, we proposed a congestion-based assignment model, a decomposition and parallel computing scheme, and problem-specific variable fixing strategies. These computational improvements have been thoroughly validated by a large number of numerical cases. Experimental results suggested that the proposed algorithm was able to reduce computing time by a factor of more than 500x compared to solving the original MIP using commercial solvers. The models and algorithms have also been implemented in a prototype fleet management software and has been proven useful in practice. Further research could focus on incorporating environmental uncertainty, handling external static and moving obstacles, and communicating with exogenous fleets through the UTM architecture.

Acknowledgment

The authors would like to thank three anonymous reviewers for their constructive comments that helped improve the paper. The first and third authors are supported in part by the National Science Foundation, grant number 1944068, and in part by the Michigan Translational Research & Commercialization (MTRAC) Innovation Hub for Advanced Transportation Commercialization Award, which is funded in part by the Michigan Strategic Fund and administered by the Michigan Economic Development Corporation (MEDC) Entrepreneurship & Innovation initiative.

REFERENCES

[1] U. Congress, "FAA reauthorization act of 2018," 2018. [Online]. Available: https://www.congress.gov/bill/115th-congress/house-bill/302

[2] ——, "Drone integration and zoning act of 2019," 2019. [Online]. Available: https://www.congress.gov/bill/116th-congress/senate-bill/2607

[3] Federal Aviation Administration, "Remote Identification of Unmanned Aircraft Systems," 2019. [Online]. Available: https://www.regulations.gov/docket?D=FAA-2019-1100

[4] D. Federal Aviation Administration, "Type certification of unmanned aircraft systems," in *Federal Register*, vol. 85, no. 22, February 2020.

[5] P. Kopardekar, J. Rios, T. Prevot, M. Johnson, J. Jung, and J. E. R. III, "Unmanned aircraft system traffic management (UTM) concept of operations," in *16th AIAA Aviation Technology, Integration, and Operations Conference*, June 2016.

[6] Federal Aviation Administration, "Unmanned aircraft system traffic management (UTM) concept of operations v2.0," March 2020.

[7] J. Rios, D. Mulfinger, J. Homola, and P. Venkatesan, "Nasa uas traffic management national campaign: Operations across six uas test sites," in *2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*, 2016, pp. 1–6.

[8] M. Johnson, J. Jung, J. Rios, J. Mercer, J. Homola, T. Prevot, D. Mulfinger, and P. Kopardekar, "Flight test evaluation of an unmanned aircraft system traffic management (UTM) concept for multiple beyond-visual-line-of-sight operations," in *Twelfth USA/Europe Air Traffic Management Research and Development Seminar (ATM2017)*, 2017.

[9] R. Lineberger, A. Hussain, M. Metcalfe, and V. Rutgers, "Infrastructure barriers to the elevated future of mobility," 2019. [Online]. Available: https://www2.deloitte.com/content/dam/insights/us/articles/5103_Infrastructure-barriers-to-elevated-FOM/DI_Infrastructure-barriers-to-elevated-FOM.pdf

[10] C. S. Venkatakrishnan, A. Barnett, and A. R. Odoni, "Landings at logan airport: Describing and increasing airport capacity," *Transportation Science*, vol. 27, no. 3, pp. 211–227, 1993.

[11] H. Balakrishnan and B. Chandran, *Scheduling Aircraft Landings Under Constrained Position Shifting*. AIAA, 2006.

[12] T. Prevot, J. Rios, P. Kopardekar, J. E. R. III, M. Johnson, and J. Jung, "UAS traffic management (UTM) concept of operations to safely enable low altitude flight operations," in *16th AIAA Aviation Technology, Integration, and Operations Conference*, June 13-17 2016.

[13] J. Rios, "Strategic deconfliction: System requirements," National Aeronautics and Space Administration, July 2018. [Online]. Available: https://utm.arc.nasa.gov/docs/2018-UTM-Strategic-Deconfliction-Final-Report.pdf

[14] D. Bertsimas and S. S. Patterson, "The traffic flow management rerouting problem in air traffic control: A dynamic network flow approach," *Transportation Science*, vol. 34, no. 3, pp. 239–255, 2000.

[15] Y. Wan, C. Taylor, S. Roy, C. Wanke, and Y. Zhou, "Dynamic queuing network model for flow contingency management," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 14, pp. 1380–1392, 09 2013.

[16] J. Xie and Y. Wan, *A Network Condition-Centric Flow Selection and Rerouting Strategy to Mitigate Air Traffic Congestion under Uncertainties*. AIAA, June 2017.

[17] L. Pallottino, E. M. Feron, and A. Bicchi, "Conflict resolution problems for air traffic management systems solved with mixed integer programming," *IEEE Transactions on Intelligent Transportation Systems*, vol. 3, no. 1, pp. 3–11, Mar 2002.

[18] M. Christodoulou and C. Costoulakis, "Nonlinear mixed integer programming for aircraft collision avoidance in free flight," in *Proceedings of the 12th IEEE Mediterranean Electrotechnical Conference (IEEE Cat. No.04CH37521)*, vol. 1, May 2004, pp. 327–330 Vol.1.

[19] E. Frazzoli, Z.-H. Mao, J.-H. Oh, and E. Feron, "Resolution of conflicts involving many aircraft via semidefinite programming," *Journal of Guidance, Control, and Dynamics*, vol. 24, no. 1, 2001.

[20] M. Pechoucek and D. Sislak, "Agent-Based Approach to Free-Flight Planning, Control, and Simulation," *IEEE Intelligent Systems*, vol. 24, no. 1, pp. 14–17, Jan. 2009. [Online]. Available: http://dx.doi.org/10.1109/MIS.2009.1

[21] Y. Liu, "A progressive motion-planning algorithm and traffic flow analysis for high-density 2d traffic," *Transportation Science*, vol. 53, no. 6, pp. 1501–1525, 2019.

[22] ——, "An optimization-driven dynamic vehicle routing algorithm for on-demand meal delivery using drones," *Comput. Oper. Res.*, vol. 111, pp. 1–20, 2019.

[23] J. F. Campbell, A. Corberán, I. Plana, and J. M. Sanchis, "Drone arc routing problems," *Networks*, vol. 72, no. 4, pp. 543–559, 12 2018.

[24] K. Dorling, J. Heinrichs, G. G. Messier, and S. Magierowski, "Vehicle routing problems for drone delivery," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 1, pp. 70–85, 2016.

[25] M. W. Ulmer and B. W. Thomas, "Same-day delivery with heterogeneous fleets of drones and vehicles," *Networks*, vol. 72, no. 4, pp. 475–505, 2018.

[26] R. G. Mbiadou Saleu, L. Deroussi, D. Feillet, N. Grangeon, and A. Quilliot, "An iterative two-step heuristic for the parallel drone scheduling traveling salesman problem," *Networks*, vol. 72, no. 4, pp. 459–474, 2018.

[27] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed. Baltimore, MD, USA: Johns Hopkins University Press, 1996.