Multidepot Drone Path Planning With Collision Avoidance

Kun Shen, *Student Member, IEEE*, Rutuja Shivgan, *Student Member, IEEE*, Jorge Medina, *Student Member, IEEE*, Ziqian Dong, *Senior Member, IEEE*, and Roberto Rojas-Cessa, *Senior Member, IEEE*

***Abstract*—Intersections of flight paths in multidrone missions are indications of a high likelihood of in-flight drone collisions. This likelihood can be proactively minimized during path planning. This article proposes two offline collision-avoidance multidrone path-planning algorithms: 1) DETACH and 2) STEER. Large drone tasks can be divided into smaller ones and carried out by multiple drones. Each drone follows a planned flight path that is optimized to efficiently perform the task. The path planning of the set of drones can then be optimized to complete the task in a short time, with minimum energy expenditure, or with maximum waypoint coverage. Here, we focus on maximizing waypoint coverage. Different from existing schemes, our proposed offline path-planning algorithms detect and remove possible in-flight collisions. They are based on a constrained nearest-neighbor search algorithm that aims to cover a large number of waypoints per flight path. DETACH and STEER perform vector intersection check for flight path analysis, but each at different stages of path planning. We evaluate the waypoint coverage of the proposed algorithms through a novel profit model and compare their performance on a work area with different waypoint densities. Our results show that STEER covers 40% more waypoints and generates 20% more profit than DETACH in high-density waypoint scenarios.**

***Index Terms*—Collision avoidance, drones, multidepot vehicle routing problem (MDVRP), optimization, path planning, unmanned aerial vehicles (UAVs).**

I. Introduction

UNMANNED aerial vehicles (UAVs) or drones have been widely adopted in aerial photography, real-time surveillance, search and rescue, delivery, precision agriculture, and other applications in recent years [1]–[3]. Due to limited battery capacity and flight range, completing a mission may require using multiple drones to coordinately execute a task that involves flying on a large geographical area [1]. In such cases, path planning is needed to determine the portion of the target area that a drone can cover. In this article, we focus on a multidrone path-planning problem where drones are tasked to collect data from a set of static waypoints dispersed over a determined area. The objective of the study is to maximize the number of covered waypoints while avoiding drone in-flight collisions by eliminating flight path intersections.

Much research has focused on finding optimal routes for drones to carry out a mission while maximizing drone-operation efficiency [1]–[7]. These vehicle routing problems are formulated as combinatorial optimization and integer programming problems, which are generalized as the traveling salesman problem (TSP) [8]. Because TSP is NP-hard, different heuristics have been proposed as efficient alternatives to solving the optimization problems that aim at minimizing the total travel distance [5], or the flight time [9], and also to avoid static obstacles [6], [7], [10], [11], as examples. Although these approaches provide efficient flight paths, they may generate intersecting paths that represent potential drone-to-drone in-flight collisions, which may be catastrophic for achieving the goal that drones are tasked with, in the first place.

In solving the multidrone path-planning problem and the potential of in-flight drone collision raise the following question: would planning a collision-free flight path be more cost effective than removing possible collisions from a collision-prone flight path? To answer this question, we focus on designing efficient collision-avoidance path-planning algorithms for a multidrone mission where drones are commissioned to cover the largest number of waypoints in a 2-D designated area.

The contributions of this article are as follows: we 1) formulate a multidepot vehicle routing problem (MDVRP) with the objective of maximizing the number of waypoints the drones visit in a determined area; 2) propose ORBIT, a nearest-neighbor-first greedy approach path-planning algorithm to solve the MDVRP; 3) propose DETACH, an algorithm that converts paths generated by ORBIT into collision-free paths by removing intersections among these paths; 4) propose STEER, an algorithm that, different from DETACH, generates collision-free paths at the path-planning phase; and 5) propose a profit model to holistically evaluate the cost effectiveness of the proposed algorithms. We evaluate the three proposed algorithms through extensive computer simulations and show that ORBIT covers the largest number of waypoints, however, at the cost of potential drone collisions due to path intersections. We also show that STEER covers 40% more waypoints than that of DETACH in an area with 500 waypoints and achieves about 20% more profit than DETACH, while both producing collision-free paths.

Manuscript received 27 August 2021; revised 30 December 2021; accepted 7 February 2022. Date of publication 16 February 2022; date of current version 24 August 2022. This work was supported by the National Science Foundation under Grant 1841558. *(Corresponding author: Ziqian Dong.)*

Kun Shen, Rutuja Shivgan, and Ziqian Dong are with the Department of Electrical and Computer Engineering, New York Institute of Technology, New York, NY 10023 USA (e-mail: kshen03@nyit.edu; rshivgan@nyit.edu; ziqian.dong@nyit.edu).

Jorge Medina and Roberto Rojas-Cessa are with the Networking Research Laboratory, Helen and John C. Hartmann Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07102 USA (e-mail: jmc237@njit.edu; rojas@njit.edu).

Digital Object Identifier 10.1109/JIOT.2022.3151791

The remainder of this article is organized as follows. Section II presents related work on collision-avoidance path planning. Section III introduces the proposed MDVRP formulation. Section IV presents the proposed ORBIT, DETACH, and STEER path-planning algorithms. Section V presents the simulation setup, performance metrics, and performance comparison among the three proposed path-planning algorithms. Section VI concludes this article.

II. Related Work

Multiple UAVs have been used in providing heterogeneous communication networks for disaster recovery [17], [18], combating COVID-19 [19], and supporting Internet of Things framework for smart-city applications [20] in recent years. Energy efficiency [21] and collision avoidance [22] are the challenges to tackle for multi-UAV path planning. While there are numerous works on energy-efficient path planning, we focus on collision-avoidance path planning in this study. Collision-avoidance path-planning algorithms can be categorized on whether they avoid static or moving obstacles. Most of the recent body of work has focused on collision-avoidance path planning for static obstacles [12]–[14], [23]–[25]. Collision avoidance for moving obstacles has also been recently explored [15], [16]. These approaches often adopt a graph-based approach where the vehicles and environments are represented as graphical shapes and the paths as line segments. These collision-avoidance path-planning approaches are formulated as constrained optimization problems with an objective to select an optimal path from all possible line segments connecting a source to a destination. The vehicles that such algorithms consider include UAVs, autonomous marine vehicles (AMVs), and mobile robots. The considered UAVs in such works are equipped with sensors, such as global positioning system (GPS), camera, or radar, to detect objects in the path [23], [24].

The graph-based path-planning algorithms [26], [27], also known as free-space algorithms, generate paths that keep the same distance from the edges of the obstacles, however, the solution may not be optimal. To improve the free-space algorithms, Li *et al.* [28] applied concave polygon convex decomposition and an artificial bee colony (ABC) algorithm to efficiently and quickly find a global optimal route. Cell decomposition is another approach that first decomposes a known region into several polygon-shaped subregions and finds a path from the selected subregions using heuristic algorithms, such as Dijkstra algorithm (DA), genetic algorithm (GA), and particle swarm optimization (PSO) [14], [29], [30]. Zhang *et al.* [12] proposed a cooperative and geometric learning algorithm (CGLA) to find collision-free paths for UAVs based on the risk information obtained from sensor data of the surrounding environment. However, the reliability of collision detection depends on the accuracy of the sensor data.

For finding the optimal path in multiple vehicle path planning, various ant colony algorithms have been considered [13], [14]. Xiong *et al.* [13] proposed a hybrid Voronoi-based ant colony optimization algorithm for multiple AMV path planning for adaptive ocean sampling. This hybrid approach provides an effective and robust solution for constrained mission time while avoiding intervehicle collision and obstacles. Tan *et al.* [14] used the ant colony system (ACS) algorithm to find a global optimum collision-free path for mobile robots.

Other collision-avoidance methods adopt a coordinated approach through a communication network among a swarm of drones. Kim *et al.* [25] used a real-time data link to track the trajectory of drones. Other approaches share the drones' position information through a communication network and use the drone location data to calculate a margin of safety distance for each drone to follow to avoid collision [15], [31]. Ahmed *et al.* [16] adopted flight paths at different altitudes to avoid collisions on intersected paths. Yang *et al.* [32] proposed a separation maintenance method to avoid collision between drone paths in both space and time. They manually set a safety distance and a safety time to avoid drone collision. Any violation of the preset safety distance and time are added as a penalty to the cost of the planned paths, but the added penalty does not guarantee a collision-free path.

In this article, we formulate the collision-avoidance multidrone path planning as a constrained MDVRP. We first propose an ORBIT algorithm to solve the MDVRP. Different from the existing solutions, which mostly rely on the intercommunication or a centralized control among drones to avoid collision, we assume that there is no intercommunication among drones and the drones are flying at the same height. We then propose two graph-based collision-avoidance path-planning algorithms: 1) DETACH and 2) STEER. We simulate the proposed algorithms and compare their performance based on the number of waypoints covered, the number of drones used, travel distance, and their cost effectiveness evaluated through our proposed profit model. Table I summarizes the different collision-avoidance path-planning approaches and compares the pros and cons of the existing works with those of our proposed path-planning algorithms.

III. Multidepot Vehicle Routing Problem With Collision-Avoidance Formulation

In this article, drones are tasked to survey a number of waypoints in a determined area. The following assumptions are made for the MDVRP with collision-avoidance formulation.

1) Each drone is assigned to one depot, from where it departs before performing the task and to where it returns after completing the task.
2) The depots are uniformly distributed in the area.
3) Each drone visits waypoints within a circular area with radius r centered at its depot.
4) The drones are homogeneous, each with a distance capacity Q.
5) A drone is used only if it covers a minimum number of L waypoints in its route.

Table II outlines the definition of terminology used in this article. The set of waypoints is denoted as V. The set of depots is denoted as D. Each depot is denoted as k, where $k \epsilon D$. Here, a node is defined as either a depot or a waypoint. (a_i, b_i) and

TABLE I
COMPARATIVE ANALYSIS OF COLLISION-AVOIDANCE PATH-PLANNING ALGORITHMS

Type of Obstacles	Type of Vehicle	Objective	Algorithm	Pros and Cons
Static Obstacles	Multiple UAVs [12]	Collision-free paths for multiple UAVs based on risk information [12].	Cooperative and Geometric Learning Algorithm (CGLA)	Pros: More effective than Voronoi Graph Search and Visibility Graph Search. Cons: Prone to collision due to inaccurate sensor data.
	Multiple autonomous marine vehicles (AMVs) [13]	Collision-free optimal trajectories for multiple AMVs to collect ocean measurements [13].	Hybrid Voronoi-based Ant Colony Optimization (V-ACO)	Pros: More effective and robust than conventional ACO, rapidly-exploring random tree star (RRT*) and Dijkstra's algorithm. Cons: Prone to collision due to inaccurate sensor data.
	Mobile Robots [14]	Global optimum collision-free path planning for mobile robots [14].	i) MAKLINK graph theory ii) Dijkstra Algorithm iii) Ant Colony System (ACS)	Pros: Faster convergence than genetic algorithm. Cons: Performance depends on the parameter selection.
	Multiple UAVs (this paper)	Collision-free paths for multiple UAVs while optimizing the number of waypoints covered.	i) ORBIT algorithm - greedy collision-prone path planning ii) DETACH algorithm - detects intersections in planned paths and removes possible collisions from paths iii) STEER algorithm - collision-avoidance path planning	Pros: Collision-avoidance path planning. Cons: DETACH and STEER DETACH and STEER have longer convergence time than ORBIT due to path intersection check.
Moving Obstacles	Multiple UAVs [15], [16]	Collision avoidance with commercial aircrafts and other moving obstacles using sampling-based path planning [15].	Sampling-based Path planning methods: i) Closed-Loop Rapidly Exploring Random Tree (Closed-Loop RRT) ii) Simplified node connection strategy iii) Use of intermediate points iv) Reachable set	Pros: Faster convergence. Cons: Prone to collision due to inaccurate sensor data and vehicle modeling errors.
		Minimizing the energy consumption while avoiding collision between UAVs using space discretization [16].	i) Optimal Path Planning (OPP) ii) Greedy Least Cost (GLC) iii) First Detect First Reserve (FDFR)	Pros: GLC and FDFR converge faster than OPP. OPP consumes least energy than GLC and FDFR. Cons: Prone to failure due to inaccurate sensor data. Complexity of OPP higher than GLC and FDFR.

TABLE II
TERMINOLOGY AND DEFINITION

Terminology	Definition
D	The set of depots
V	The set of waypoints
r	Radius of a drone's flight range
Q	Vehicle distance capacity
L	The minimum number of waypoints in a route
$\mathbf{M}$	Distance matrix
R_k	Route of Drone k
$\lvert R_k \rvert$	The number of waypoints in R_k
$\mathbf{R}$	The set of routes for drones
$\mathbf{R}_d$	The set of collision-free routes for drones
$d_{n_w,k}$	The return route distance from current waypoint n_w to depot k
$d_{cum,k}$	Cumulative route distance for drone k

(a_j, b_j) denote the coordinates of nodes i and j, respectively. The distance (Euclidean) between nodes i and j, d_{ij}, is defined in (1). The pairwise distance among all nodes form the distance matrix, $\mathbf{M}$. R_k denotes the route for the drone starting at depot k. The number of waypoints in R_k is denoted as $|R_k|$. $\mathbf{R}$ is the route matrix that stores the routes for the drones where row i of the matrix stores the route for Drone i. $\mathbf{R_d}$ denotes the set of collision-free routes for drones. $d_{n_w,k}$ denotes the distance of the return trip from the last waypoint in the route to depot k. $d_{cum,k}$ denotes the distance traveled by Drone k

$$d_{ij} = \sqrt{\left(a_i - a_j\right)^2 + \left(b_i - b_j\right)^2}. \tag{1}$$

The binary decision variable $x_{ijk} \epsilon \{0, 1\}$ in (2) denotes whether the edge between nodes i and j is selected as part of the route for depot k

$$x_{ijk} = \begin{cases} 1, & \text{if drone } k \text{ travels from } i \text{ to } j \\ 0, & \text{the edge is not selected.} \end{cases} \tag{2}$$

We formulate the multidepot drone path-planning problem as follows:

$$\max_{j} \quad \sum_{i \epsilon D \cup V} \sum_{j \epsilon V} \sum_{k \epsilon D} x_{ijk} \tag{3}$$

$$\text{Subject to:} \quad \sum_{i \epsilon D \cup V} \sum_{k \epsilon D} x_{ijk} \leq 1 \quad \forall\, j \subseteq V \tag{4}$$

$$\sum_{i \epsilon V} \sum_{j \epsilon D} x_{ijk} = 1 \quad \forall\, k \epsilon D \tag{5}$$

$$\sum_{i \epsilon D} \sum_{j \epsilon V} d_{ij} \sum_{k \epsilon D} x_{ijk} \leq r, \quad k = i \tag{6}$$

$$\sum_{i \epsilon D \cup V} \sum_{j \epsilon D \cup V} d_{ij} \sum_{k \epsilon D} x_{ijk} \leq Q_k \tag{7}$$

$$\sum_{i \epsilon D \cup V} \sum_{j \epsilon V} x_{ijk} \geq L \quad \forall k \epsilon D \tag{8}$$

$$\sum_{i \epsilon S} \sum_{j \epsilon S} x_{ijk} \leq |S| - 1 \quad \forall\, S \subseteq\ V, |S| \geq 2 \tag{9}$$

$$\left(x_{ijk} x_{pql}\right) H_{ijk,pql} = 0,\ i \neq p,\ j \neq q,\ k \neq l. \tag{10}$$

The objective is to maximize the number of waypoints visited by drones as denoted in (3). Constraint (4) ensures that a waypoint is visited at most once. Constraint (5) ensures that each drone returns to its depot. Constraint (6) ensures that the visited waypoints are within radius r from the depot. Constraint (7) ensures the drone travels within its distance capacity. Constraint (8) ensures that each drone visits at least L waypoints. Constraint (9) eliminates subtours (a subtour is a round-trip tour that returns to its depot without visiting all the allocated waypoints) for each drone. Constraint (10) ensures that a path is collision-free [16]. Here, $H_{ijk,pql}$ is a binary variable for collision check. $H_{ijk,pql} = 0$ indicates that there is no collision between edge ij for Drone k and edge pq for Drone l. $H_{ijk,pql} = 1$ indicates that there is a collision between the two edges. The formulated MDVRP is NP-hard. Therefore, we propose two heuristic collision-avoidance path-planning algorithms to solve the MDVRP.

IV. Proposed Path-Planning Algorithms

In this section, we introduce the proposed ORBIT, DETACH, and STEER path-planning algorithms.

A. Orbit Algorithm

ORBIT uses the nearest-neighbor-first selection to find the waypoints to include in a drone's route. The pseudocode of ORBIT is shown in Algorithm 1. ORBIT uses the distance matrix to find the nearest node from a depot and checks if the path length is within the distance capacity of the drone to be added to the path. This process is performed iteratively until the path length reaches a drone's distance capacity. The resulting route is stored in R_k. The waypoints included in R_k are marked as unavailable. ORBIT continues this process for the drones at other depots until no further routes can be formed. Path length has to meet the minimum L number of waypoints requirement to be set as a valid route. The resulting routes are stored in **R**.

Fig. 1 shows an example of the operation of ORBIT with two drones, D1 and D2, in an area with ten waypoints, which are indexed as W1–W10. ORBIT randomly chooses the first depot D1 and searches for the waypoints within its range. The candidates are W1, W2, W3, W9, W8, and W10, from which W1 is selected. ORBIT checks if the path length, which is the distance from D1 to W1 and the return trip from W1 to D1, is within the drone's capacity. The nearest waypoint W2 from W1 is added to the route and is followed by W3 and W9. However, because adding W8 to the route would exceed the distance capacity, D1 ends the search and the return path to D1 from W9 becomes the last route segment. The route for D1 is D1 → W1 → W2 → W3 → W9 → D1. Similarly, the route for D2 is D2 → W6 → W8 → W4 → W5 → D2. The unvisited waypoints W7 and W10 are called orphan nodes. Fig. 1 shows that there is a potential collision between W3 → W9 and W4 → W8 for D1 and D2, respectively.

Algorithm 1 ORBIT Algorithm

```
Input: D, V, Q, L, r
Output: R
Calculate distance matrix M
for k in D do
    T_k = [ ]
    Add all the unvisited waypoints in T_k within the coverage range r from k
    if |T_k| ≥ L then
        n = k
        d_cum,k = 0
        while |T_k| do
            Find distance d_min from n to the next nearest waypoint (n_w) in T_k
            d_cum,k = d_cum,k + d_min
            if d_cum,k + d_n_w,k ≤ Q then
                Add n_w in the R_k
                n = n_w
                Remove n_w from T_k
            else
                break
            end if
        end while
    end if
    if |R_k| ≥ L then
        Mark all waypoints in R_k as visited
        Add k to R_k and add R_k in R
    end if
end for
```

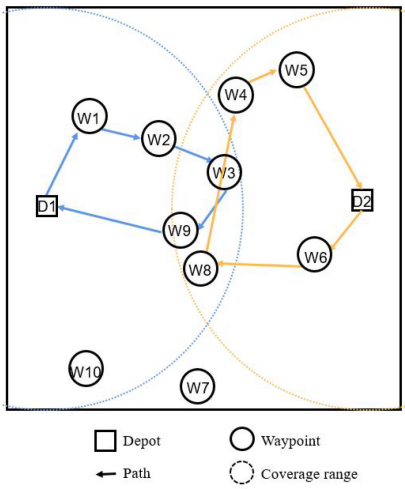

Fig. 1. ORBIT path-planning example.

These path intersections are collision hazards for drones. To avoid potential collisions, we propose DETACH and STEER for intersection-free path planning.

B. DETACH Algorithm

We propose the DETACH algorithm to remove the route intersections generated by ORBIT. DETACH is a graph-based

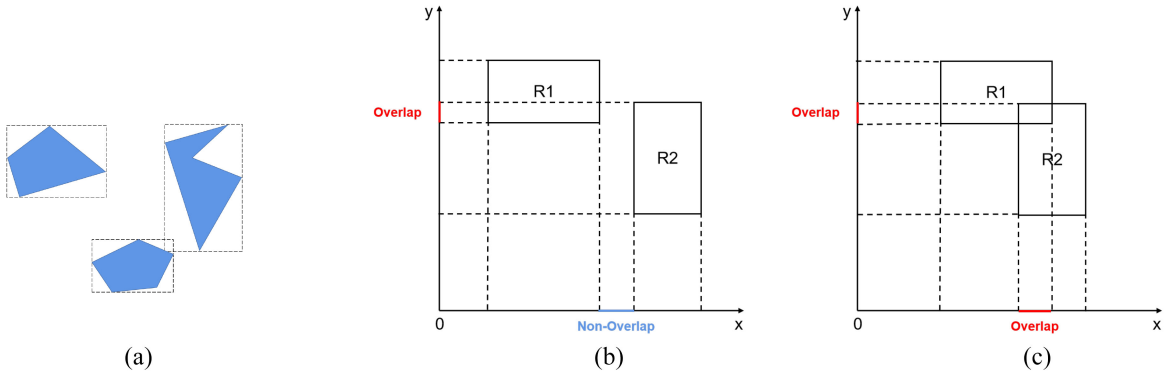

Fig. 2. Axis-aligned boundary box. (a) Boundary box. (b) Disjoint box. (c) Overlap box.

method where a route is interpreted as the edges of a polygon. To detect and avoid these collisions, DETACH uses an axis-aligned bounding box (AABB) to check for intersections between two routes. Then, it uses the 2-D vector cross product introduced in Section IV-B1 to determine intersected routes. The intersected routes are disjointed by keeping the nodes of the path that yields the least number of orphan nodes.

1) 2-D Vector Cross Product: A 2-D vector cross product produces another vector that is perpendicular to the vectors under cross production and the value of the cross product determines the relative geometric relation of vectors in a 2-D space, such as whether they are on the same line (co-linear) or intersect at an angle. As an example, for two vectors $\overrightarrow{p} = (p_x, p_y)$ and $\overrightarrow{q} = (q_x, q_y)$, the cross product is defined as

$$\overrightarrow{p} \times \overrightarrow{q} = p_x \cdot q_y - p_y \cdot p_x. \tag{11}$$

The relative geometric relation between $\overrightarrow{p}$ and $\overrightarrow{q}$ is defined as follows.

1) When $\overrightarrow{p} \times \overrightarrow{q} > 0$, $\overrightarrow{q}$ is at the counter-clockwise side of $\overrightarrow{p}$.
2) When $\overrightarrow{p} \times \overrightarrow{q} < 0$, $\overrightarrow{q}$ is at the clockwise side of $\overrightarrow{p}$.
3) When $\overrightarrow{p} \times \overrightarrow{q} = 0$, $\overrightarrow{q}$ is co-linear with $\overrightarrow{p}$.

2) Axis-Aligned Bounding Box: We use the AABB algorithm to detect route intersections. AABB is an efficient method for 2-D and 3-D object collision detection in games, simulation, and virtual reality [33]. As shown in Fig. 2(a), AABB uses the x and y axes to create a minimum enclosing rectangle to enclose each object. It uses the projections of the boundary box on the x and y axes to check for collisions between objects. No overlap between the projections of two boundary boxes indicates that the two objects do not intersect.

3) Vector Intersection Check: Vector intersection check is a two-step vector cross-product method used to find crossed vectors. The first step is to find boundary box overlap for two routes through AABB. Here, the vector cross product for three consecutively connected nodes, a, b, and c, is defined as

$$\text{cross}(a, b, c) = \overrightarrow{ab} \times \overrightarrow{bc}. \tag{12}$$

In a 2-D space, one vector divides the space into two parts. If the two points of another vector are located on the opposite sides of the space or one of the points is on the first vector and the other is in one part of the space, these two vectors

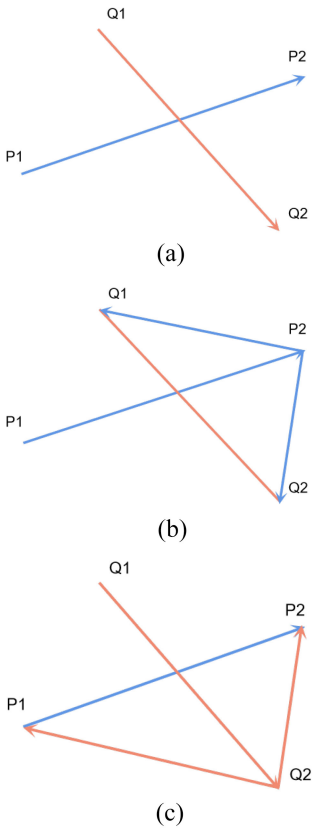

Fig. 3. Vector intersection check. (a) Two intersected vectors. (b) Condition 1. (c) Condition 2.

intersect. Fig. 3(a) shows that $P1$, $P2$, and $Q1$, $Q2$ are two pairs of consecutive nodes in two different routes and $\overrightarrow{P1P2}$ and $\overrightarrow{Q1Q2}$ are the path vectors. If the two path vectors cross, they must meet two conditions.

1) *Condition 1:* For $\overrightarrow{P1P2}$, two new vectors $\overrightarrow{P2Q1}$ and $\overrightarrow{P2Q2}$ are created as shown in Fig. 3(b). The cross products of the vectors formed with the four nodes are calculated based on (13)–(15). $C1$ is the multiplication of (13) and (14). If cross$(P1, P2, Q1)$ and cross$(P1, P2, Q2)$ have opposite signs, $Q1$ and $Q2$ are on the opposite sides of $\overrightarrow{P1P2}$. Additionally, if one of the results is 0, that means either $Q1$ or $Q2$ is on the vector $\overrightarrow{P1P2}$. Therefore, we can conclude that for $C1 \leq 0$, $Q1$

Algorithm 2 Vector Intersection Check

```
Input: Two routes: R_1 and R_2
Output: True or False
for Node i in R_1 do
    for Node j in R_2 do
        P1 = i, P2 = next i;
        Q1 = j, Q2 = next j;
        if Boundary box check for the two vector P1P2 and Q1Q2 then
            if cross(P1, P2, Q1) · cross(P1, P2, Q2) ≤ 0 and
            cross(Q1, Q2, P1) · cross(Q1, Q2, P2) ≤ 0 then
                return True
            end if
        end if
    end for
end for
return False
```

and $Q2$ are on the opposite sides or on the vector $\overrightarrow{P1P2}$; otherwise, $Q1$ and $Q2$ are on the same side of $\overrightarrow{P1P2}$

$$\text{cross}(P1, P2, Q1) = \overrightarrow{P1P2} \times \overrightarrow{P2Q1} \tag{13}$$

$$\text{cross}(P1, P2, Q2) = \overrightarrow{P1P2} \times \overrightarrow{P2Q2} \tag{14}$$

$$C1 = \text{cross}(P1, P2, Q1) \cdot \text{cross}(P1, P2, Q2) \tag{15}$$

$$C2 = \text{cross}(Q1, Q2, P1) \cdot \text{cross}(Q1, Q2, P2). \tag{16}$$

2) *Condition 2:* For $\overrightarrow{Q1Q2}$, two vectors $\overrightarrow{Q2P1}$ and $\overrightarrow{Q2P2}$ are created as shown in Fig. 3(c). Similar to Condition 1, $C2$ is the product of two cross products cross $(Q1, Q2, P1)$ and cross$(Q1, Q2, P2)$. For $C2 \leq 0$, $P1$ and $P2$ are on the opposite sides or on $\overrightarrow{Q1Q2}$; otherwise, $P1$ and $P2$ are on the same side of $\overrightarrow{Q1Q2}$.

Given these two conditions, we get $C1$ from $\overrightarrow{P1P2}$ and $C2$ from $\overrightarrow{Q1Q2}$. If both $C1$ and $C2$ are nonpositive, $\overrightarrow{P1P2}$ and $\overrightarrow{Q1Q2}$ intersect. Otherwise, they are disjoint, or collision-free. The pseudocode of vector intersection check is shown in Algorithm 2.

4) Implementation of DETACH: The pseudocode in Algorithm 3 describes DETACH. This algorithm uses the route output from ORBIT **R** as input. Fig. 4 shows two routes, Route$_A$ = [A, W1, W2, W3, W4, A] and Route$_B$ = [B, W5, W6, W7, W8, B]. Starting from the first route, DETACH conducts pairwise route intersection check in Route$_A$ and Route$_B$ using Algorithm 2 and AABB. Fig. 4(a) shows two intersections between Route$_A$ and Route$_B$. The first intersection is between edges $W3 \rightarrow W4$ and $W6 \rightarrow W7$; the second intersection is between edges $W4 \rightarrow A$ and $B \rightarrow W5$.

As shown in Fig. 4(c), the number of orphan nodes in Route$_A$ and Route$_B$ is 1 and 2, respectively. Route$_A$ removes orphan node $W4$. DETACH continues pairwise comparison for other routes until no more intersections are found. The final disjoint route is shown in Fig. 4(c).

However, there are two special cases that the fewer orphan node rule in DETACH does not cover. The first occurs when one route intersects with another through an edge without any node inside one another as shown in Fig. 5(a). Route$_A$ intersects with Route$_B$. In this case, Route$_A$ is selected to remove its orphan nodes as shown in Fig. 5(b). The second case occurs when a depot is located inside another route. As shown in Fig. 5(c), Route$_B$ = [B, W5, W6, W7, W8, B] intersects with Route$_A$ = [A, W1, W2, W3, W4, A] with Depot$_B$ inside

Algorithm 3 DETACH Algorithm

```
Obtain routes R from ORBIT
Input: D, V, R
Output: R_d
for any pair of R_i and R_j in R do
    if The projection of R_i and R_j on both axes are overlapped then
        if R_i and R_j intersect then
            Record the intersection nodes in the arrays:
            Intersection_i = Intersection_j= [ ]
            Modify the intersection node arrays and get both
            Orphan_i and Orphan_j node arrays
            if |Orphan_i| ≥ |Orphan_j| and both |Orphan_i| and
            |Orphan_j| > 0 or |Orphan_i| = 0 then
                Remove Orphan_j from R_j
            else if |Orphan_i| < |Orphan_j| and both |Orphan_i|
            and |Orphan_j| > 0 or |Orphan_j| = 0 then
                Remove Orphan_i from R_i
            end if
        end if
    end if
end for
return R_d
```

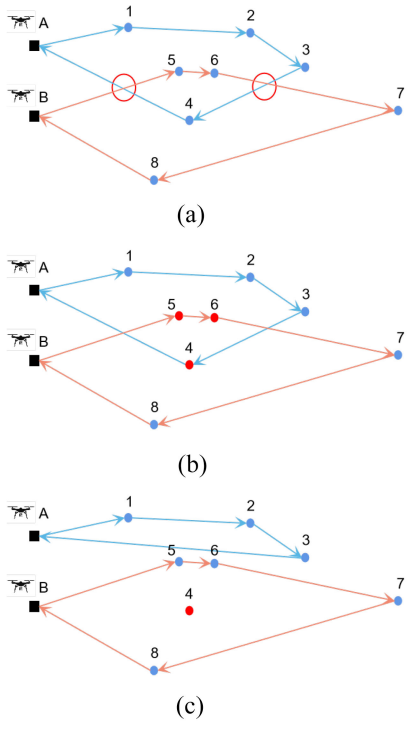

Fig. 4. DETACH example. (a) Two intersected routes. (b) Find waypoints of the intersected routes. (c) Disjoint routes.

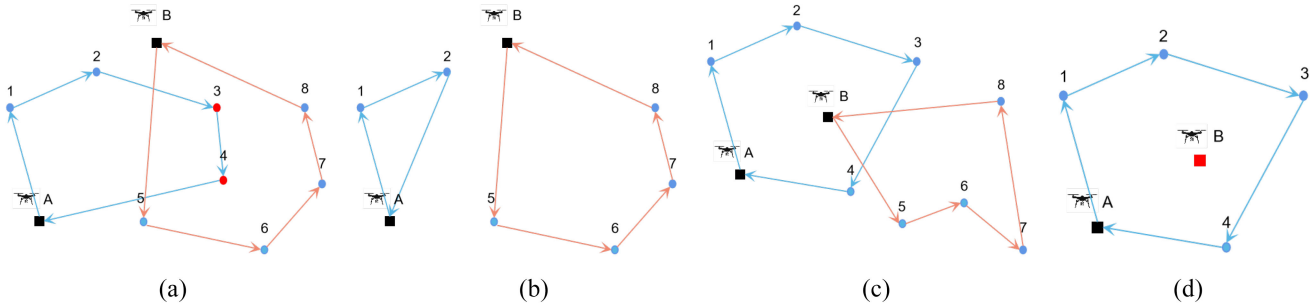

Fig. 5. Special cases for route intersection. (a) Edge crossed routes. (b) Disjoint edge crossed routes. (c) Depot inside route. (d) Eliminate route.

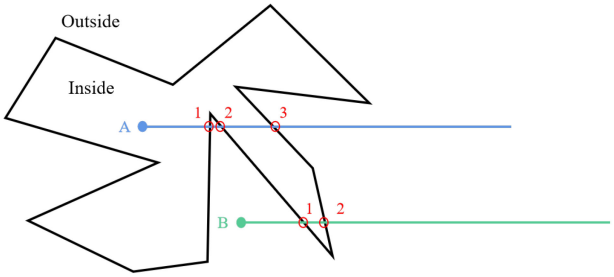

Fig. 6. Node inside a polygon check.

Route$_A$. The two orphan node arrays are Orphan$_A$ = [ ] and Orphan$_B$ = [W5, W6, W7, W8]. Based on the fewer orphan nodes rule, the new Route$_B$ = [B, B] leaves only a depot in the route. In this case, the idle depot is considered as unused and is removed.

C. STEER Algorithm

Because the intersection separation process by DETACH generates numerous orphan nodes, we propose the STEER algorithm, which generates collision-free routes at the path-planning phase. STEER performs vector intersection check during the nearest-neighbor search to avoid intersections. Moreover, STEER uses the PNPOLY algorithm to exclude nodes located inside existing routes to ensure fast convergence [34].

1) Node Inside Route Check: PNPOLY-point inclusion in polygon test is used to determine whether a point is within a polygon [34]. The detection strategy uses a horizontal line from a point in a plane toward ∞ and counts the number of intersections it generates with the edges of a polygon. If the number is odd, the point is inside the polygon; otherwise, the point is outside the polygon, as shown in Fig. 6. There are two steps in the check for whether a waypoint is inside a route, as shown in Algorithm 4.

2) Implementation of STEER: Fig. 7 shows an example of how STEER finds a collision-free route. Given the input parameters D, V, Q, L, r, and $\mathbf{M}$, STEER uses the nearest-neighbor search sequentially from the first depot to the last selected waypoint. It finds the waypoints that are in the circular area with a radius r to the depot and adds them into the neighbor set T. For node n, it finds the closest waypoint n_w from V to n in T and conducts the edge intersection check to select waypoints that generate collision-free path. Fig. 7(a) shows the route for $D1$, where R_1 is the first collision-free path obtained using the nearest-neighbor search. $D2$ selects waypoints $P6$ and $P8$ consecutively to include in the route as shown in Fig. 7(b). $P4$ is the closest waypoint to $P8$ but the edge $\overrightarrow{P8P4}$ intersects with edge $\overrightarrow{P3P9}$ in R_1. Therefore, $P8$ moves to find the next closest waypoint $P5$, which satisfies the intersection check as shown in Fig. 7(c). If the capacity and the minimum number of waypoints constraints are satisfied, the route is added to $\mathbf{R_d}$. The pseudocode of STEER is shown in Algorithm 5.

Algorithm 4 Node Inside Route

```
Input: a single node P_n = (P_x, P_y), R_k
Output: Detect if P_n is inside R_k
Define: P_e = (∞, P_y); the ray vector P_nP_e; count = 0; and j, the index of the next waypoint of node with index i in R_k
if P_n and R_k do not satisfy Axis-Aligned Bounding Box Check then
    return Pn is not inside R_k
end if
i = 0
while j ≠ 0 do
    j = (i + 1) (mod n)
    if P_nP_e intersects with edge vector P_iP_j then
        count + +
    end if
    i = j
end while
return Pn is inside R_k if count is odd
```

3) Time Complexity: In the complexity analysis of the three algorithms, we consider two parameters: 1) the number of depots N and 2) the number of waypoints M. ORBIT adopts a greedy approach and it is agnostic to route intersection; making it prone to collisions. The complexity of ORBIT is $O(NM \log M)$. DETACH uses a vector intersection check mechanism and follows a fewer-orphan nodes path rule to select the removed path in intersection removal. It has a complexity of $O(N^2M^3)$. STEER checks for intersections at the path generation stage and has a complexity of $O(NM^3 \log M)$. Therefore, ORBIT has the lowest complexity as it is collision-agnostic, while STEER is less complex than DETACH because it requires fewer intersection checks.

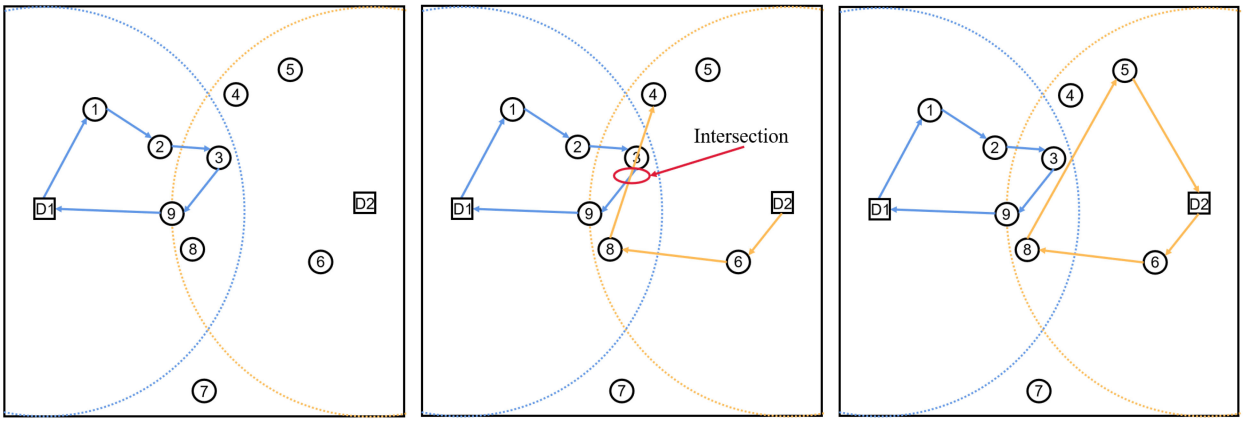

Fig. 7. Example of the STEER algorithm. (a) Route for $D1$ established. (b) Path intersection detected. (c) Two disjoint routes.

Algorithm 5 STEER Algorithm

Input: D, V, Q, L, r, $\mathbf{M}$
Output: $\mathbf{R}_d$
for k in D **do**
  T_k = [ ]
  Add all the unvisited waypoints in T_k within coverage range r from k
  if $|T_k| \geq L$ **then**
    $n = k$
    $d_{cum,k} = 0$
    Find the minimum distance d_{min} from n to waypoint (n_w) in T_k
    if $\overrightarrow{nn_w}$ is intersection free with any other edges in $\mathbf{R}_d$ and $\overrightarrow{n_wk}$ is intersection free with any other edges in $\mathbf{R}_d$ **then**
      if n_w satisfies $d_{cum,k} + d_{min} + d_{n_w,k} \leq Q$ **then**
        Add n_w to R_k; mark n_w as unavailable
        $d_{cum,k} += d_{min}$;
        $n = n_w$
      else
        Discard n_w and set it as unavailable
      end if
    else
      Discard n_w and set it as unavailable
    end if
  end if
  if $|R_k| \geq L$ **then**
    Mark all waypoints in R_k as visited
    Add k to R_k, then add R_k to $\mathbf{R}_d$
  end if
  for node i in V **do**
    if i is inside current $\mathbf{R}_d$ using Node Inside Check **then**
      Remove i from V
    end if
  end for
end for

V. Results and Analysis

In this section, we present the simulation setup and compare the results and performance of the proposed ORBIT, DETACH, and STEER algorithms.

A. Simulation Setup

We implement the algorithms in C++ and evaluate their performance through computer simulation. The flight zone is set as a 4 × 4 km^2 area with 25 depots and different waypoint densities with a range of {50, 500}, with increments of 50 waypoints. The number of depots is determined by the upper bound of the number of depots needed to cover this area. The waypoints are randomly distributed inside the area. The locations of the depots are evenly distributed in the area and are fixed for all the experiments. Each depot is home to one drone, the algorithms randomly choose the depots from the given set of 25 depots to generate routes. The simulation parameters are outlined in Table III. We set the distance capacity of the drones to that of commercial ones, or 7 km. The range of the drone is a circle with a radius of 2 km, where the center of the circle is its depot. The drones fly at the same altitude. The minimum number of visited waypoints within each route is set to at least 3% of the total number of waypoints in the area. The simulation is run 10 000 times for each scenario. Each time, the depots are randomly selected with a randomly generated set of waypoints. The three algorithms are evaluated using the same topology. We record the mean and standard deviation of the number of orphan nodes, the number of drones used, and cumulative route distance. We also propose a profit model to evaluate the cost effectiveness of the three algorithms. Fig. 8 shows an example of the generated paths of the three algorithms with 50 waypoints. The figure shows that ORBIT covers the most waypoints but with intersected routes. Fig. 8(b) and (c) shows the resulting routes of DETACH and STEER, respectively. Here, DETACH detangles the routes generated by ORBIT while STEER randomly selects a different set of depots and generates collision-free routes.

TABLE III
Simulation Setup

Parameter	Value
Area	4×4 km^2
Number of Depots/Drones	25
Capacity of drone	7 km
Radius of drone	2 km
Number of waypoints	50 − 500
Minimum path length	3% of the number of waypoints

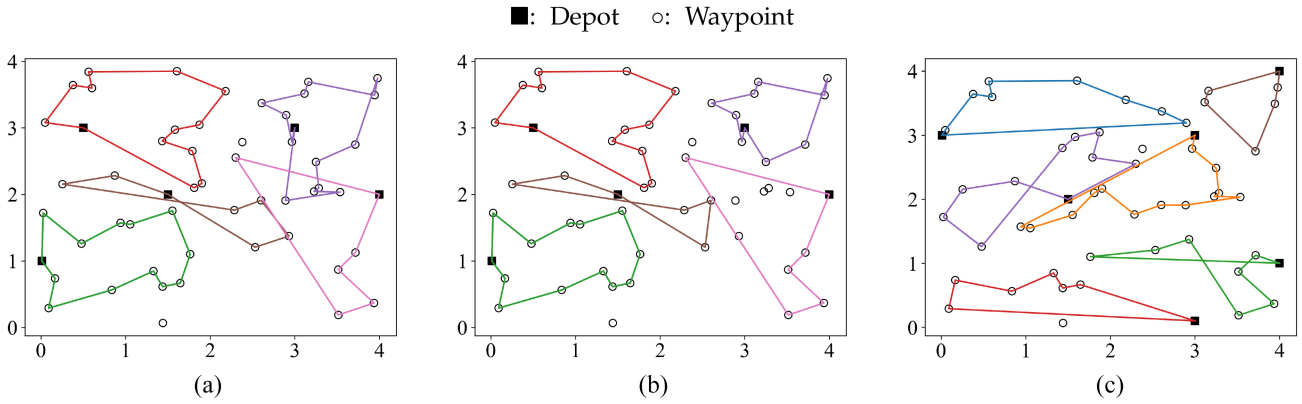

Fig. 8. Example of route generated with 50 waypoints. (a) ORBIT. (b) DETACH. (c) STEER.

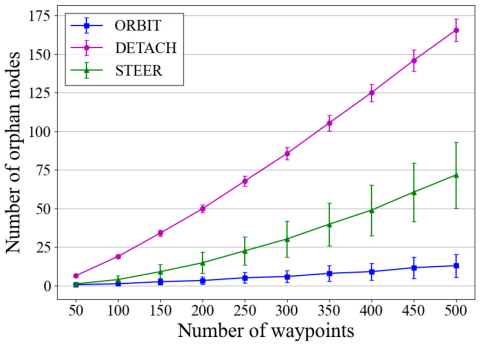

Fig. 9. Number of orphan nodes versus number of waypoints.

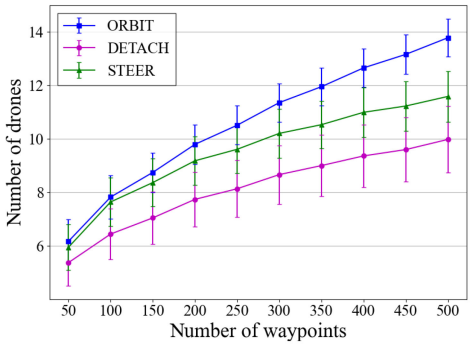

Fig. 10. Number of drones used versus number of waypoints.

B. Simulation Results

We compare the simulation results of the three algorithms in terms of the number of orphan nodes, number of drones used, and cumulative distance covered for a target area with different waypoint densities, or number of waypoints between 50 and 500. We propose a profit model to evaluate the cost effectiveness of each approach. The model considers the revenue generated from the number of covered waypoints and the cost of investment (i.e., drones), and usage as proportional to the flight distance.

Number of Orphan Nodes: The objective of our MDVRP model is to maximize the number of waypoints visited by drones, which corresponds to minimizing the number of orphan nodes. Fig. 9 shows the distribution of the number of orphan nodes the proposed algorithms leave out. The figure shows that STEER leaves fewer orphan nodes than DETACH. Although ORBIT leaves out the smallest number of orphan nodes, it is prone to collision.

As the number of waypoints (density) increases, the number of orphan nodes increases for all the algorithms, with the lowest increase rate for ORBIT and the highest for DETACH. We further compare the coverage percentage of the algorithms. For 500 waypoints, the percentage of average orphan nodes to the total number of waypoints for ORBIT, DETACH, and STEER are 2.4, 33.2, and 14.4%, respectively. ORBIT achieves this high coverage percentage at the cost of having potential drone collisions. For collision-free path planning, STEER achieves a higher waypoint coverage but at the cost of using more drones.

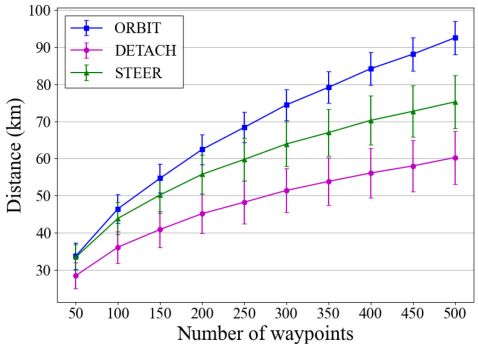

Fig. 11. Cumulative distance versus number of waypoints.

Number of Drones Used: Fig. 10 shows the average number of drones used for different number of waypoints. While ORBIT uses the largest number of drones as it covers the most waypoints. DETACH and STEER use fewer drones than ORBIT. Although STEER uses more drones than DETACH, it covers more waypoints. As the figure shows, there is a tradeoff in the number of used drones and the number of covered waypoints.

Cumulative Route Distance: Fig. 11 shows the cumulative route distance with an increasing waypoint density in the area. The figure shows that the cumulative route distance increases as the number of waypoints increases and that STEER covers more distance than DETACH. As expected, ORBIT shows the

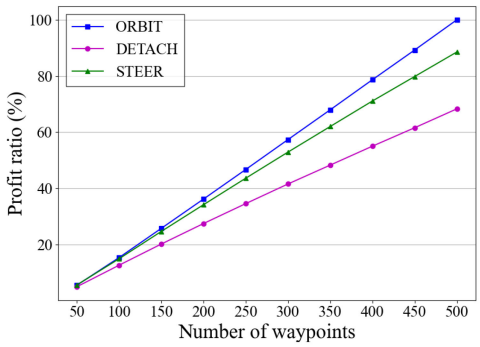

Fig. 12. Profit ratio versus number of waypoints.

maximum cumulative distance covered should collisions be neglected.

Profit Model: To make a thorough comparison where all metrics are involved, we propose a profit model considering the number of waypoints visited, the number of drones used, and the cumulative route distance. The model is defined in

$$P = N_{\text{waypoint}} \cdot \gamma - D \cdot \epsilon_{\text{distance}} - N_{\text{drone}} \cdot \epsilon_{\text{drone}}. \quad (17)$$

Here, N_{waypoint} denotes the number of visited waypoints, γ is the reward for each visited waypoint, D is the total covered distance, $\epsilon_{\text{distance}}$ is the unit cost for the distance covered by the drone, and N_{drone} is the number of drones used, each with a unit price ϵ_{drone}. We assume initial unit costs as an example, $\gamma = 50$, $\epsilon_{\text{distance}} = 5$, and $\epsilon_{\text{drone}} = 185$ to evaluate the profit of STEER and DETACH with respect to the maximum profit of ORBIT with 500 waypoints. Fig. 12 shows the profit ratio of the different algorithms to the maximum profit of ORBIT at 500 waypoints. The figure shows that for a small number of waypoints, $N_{\text{waypoint}} = 50$, the profit of the three algorithms is similar. As N_{waypoint} increases, DETACH uses fewer drones and covers the smallest distance which results in smaller costs than STEER, as shown in Figs. 10 and 11. However, STEER is more profitable than DETACH, especially as the number of waypoints increases. For 500 waypoints, STEER can achieve 20.2% more profit than DETACH.

VI. Conclusion

In this article, we formulated multidrone path planning as an MDVRP. We proposed the ORBIT algorithm to greedily find the nearest neighbor to solve the MDVRP. As ORBIT is collision-agnostic, we proposed DETACH and STEER to solve the path intersection problem to generate collision-free paths for multiple drones. DETACH adopts a two-phase approach that detects path intersections from those generated by ORBIT and detangles the path to produce collision-free routes. STEER uses a one-step approach that selects collision-free routes at the planning stage. We evaluated the three proposed algorithms through computer simulation and the results show that ORBIT covers the most waypoints; however, at the cost of potential drone collisions. Simulation results show that the number of orphan nodes of STEER is 40% of those of DETACH in an area with 500 waypoints. We also proposed a profit model to evaluate the long-term cost effectiveness of the proposed algorithms and the results showed that the profit of STEER is 20.2% higher than that of DETACH in a high waypoint density scenario using the profit of ORBIT at 500 waypoints as a baseline.

References

[1] L. Ding, D. Zhao, H. Ma, H. Wang, and L. Liu, "Energy-efficient min–max planning of heterogeneous tasks with multiple UAVs," in *Proc. IEEE 24th Int. Conf. Parallel Distrib. Syst. (ICPADS)*, Singapore, 2018, pp. 339–346.

[2] J. Modares, F. Ghanei, N. Mastronarde, and K. Dantu, "UB-ANC planner: Energy efficient coverage path planning with multiple drones," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Singapore, 2017, pp. 6182–6189.

[3] 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.

[4] R. Shivgan and Z. Dong, "Energy-efficient drone coverage path planning using genetic algorithm," in *Proc. IEEE 21st Int. Conf. High Perform. Switching Routing (HPSR)*, Newark, NJ, USA, 2020, pp. 1–6.

[5] A. Lim and F. Wang, "Multi-depot vehicle routing problem: A one-stage approach," *IEEE Trans. Autom. Sci. Eng.*, vol. 2, no. 4, pp. 397–402, Oct. 2005.

[6] X. Wang, V. Yadav, and S. N. Balakrishnan, "Cooperative UAV formation flying with obstacle/collision avoidance," *IEEE Trans. Control Syst. Technol.*, vol. 15, no. 4, pp. 672–679, Jul. 2007.

[7] H. Chen, K. Chang, and C. S. Agate, "UAV path planning with tangent-plus-Lyapunov vector field guidance and obstacle avoidance," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 49, no. 2, pp. 840–856, Apr. 2013.

[8] D. L. Applegate, R. E. Bixby, V. Chvatal, and W. J. Cook, *The Traveling Salesman Problem: A Computational Study*. Princeton, NJ, USA: Princeton Univ. Press, 2006.

[9] R. I. da Silva and M. A. Nascimento, "On best drone tour plans for data collection in wireless sensor network," in *Proc. 31st Annu. ACM Symp. Appl. Comput.*, New York, NY, USA, 2016, pp. 703–708. [Online]. Available: https://doi.org/10.1145/2851613.2851854

[10] B. M. Albaker and N. A. Rahim, "Unmanned aircraft collision detection and resolution: Concept and survey," in *Proc. 5th IEEE Conf. Ind. Electron. Appl.*, Taichung, Taiwan, 2010, pp. 248–253.

[11] H. Pham, S. A. Smolka, S. D. Stoller, D. Phan, and J. Yang, "A survey on unmanned aerial vehicle collision avoidance systems," 2015, *arXiv:1508.07723*.

[12] B. Zhang, W. Liu, Z. Mao, J. Liu, and L. Shen, "Cooperative and geometric learning algorithm (CGLA) for path planning of UAVs with limited information," *Automatica*, vol. 50, no. 3, pp. 809–820, 2014.

[13] C. Xiong, D. Chen, D. Lu, Z. Zeng, and L. Lian, "Path planning of multiple autonomous marine vehicles for adaptive sampling using Voronoi-based ant colony optimization," *Robot. Auton. Syst.*, vol. 115, pp. 90–103, May 2019.

[14] G.-Z. Tan, H. He, and A. Sloman, "Ant colony system algorithm for real-time globally optimal path planning of mobile robots," *Acta Automatica Sinica*, vol. 33, no. 3, pp. 279–285, 2007.

[15] Y. Lin and S. Saripalli, "Sampling-based path planning for UAV collision avoidance," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 11, pp. 3179–3192, Nov. 2017.

[16] 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.*, Doha, Qatar, 2016, pp. 1–6.

[17] J. Qiu, D. Grace, G. Ding, M. D. Zakaria, and Q. Wu, "Air-ground heterogeneous networks for 5G and beyond via integrating high and low altitude platforms," *IEEE Wireless Commun.*, vol. 26, no. 6, pp. 140–148, Dec. 2019.

[18] A. Saif, K. Dimyati, K. A. Noordin, S. H. Alsamhi, and A. Hawbani, "Multi-UAV and SAR collaboration model for disaster management in B5G networks," *Internet Technol. Lett.*, to be published.

[19] S. H. Alsamhi, B. Lee, M. Guizani, N. Kumar, Y. Qiao, and X. Liu, "Blockchain for decentralized multi-drone to combat COVID-19," 2021, *arXiv:2102.00969*.

[20] S. H. Alsamhi, F. Almalki, O. Ma, M. S. Ansari, and B. Lee, "Predictive estimation of optimal signal strength from drones over IoT frameworks in smart cities," *IEEE Trans. Mobile Comput.*, early access, Apr. 22, 2021, doi: 10.1109/TMC.2021.3074442.

[21] J. N. Yasin *et al.*, "Energy-efficient formation morphing for collision avoidance in a swarm of drones," *IEEE Access*, vol. 8, pp. 170681–170695, 2020.
[22] M. Kothari, I. Postlethwaite, and D.-W. Gu, "Multi-UAV path planning in obstacle rich environments using rapidly-exploring random trees," in *Proc. 48th IEEE Conf. Decis. Control (CDC) Held Jointly 28th Chin. Control Conf.*, Shanghai, China, 2009, pp. 3069–3074.
[23] L. Mejias, S. McNamara, J. Lai, and J. Ford, "Vision-based detection and tracking of aerial targets for UAV collision avoidance," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Taipei, Taiwan, 2010, pp. 87–92.
[24] Y. K. Kwag and C. H. Chung, "UAV based collision avoidance radar sensor," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, Barcelona, Spain, 2007, pp. 639–642.
[25] K.-Y. Kim, J.-W. Park, and M.-J. Tahk, "UAV collision avoidance using probabilistic method in 3-D," in *Proc. Int. Conf. Control Autom. Syst.*, Seoul, South Korea, 2007, pp. 826–829.
[26] S. Garrido, L. Moreno, M. Abderrahim, and F. Martin, "Path planning for mobile robot navigation using Voronoi diagram and fast marching," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Beijing, China, 2006, pp. 2376–2381.
[27] P. Bhattacharya and M. L. Gavrilova, "Voronoi diagram in optimal path planning," in *Proc. 4th Int. Symp. Voronoi Diagrams Sci. Eng. (ISVD)*, Glamorgan, U.K., 2007, pp. 38–47.
[28] Z. Li, Z. Zhang, H. Liu, and L. Yang, "A new path planning method based on concave polygon convex decomposition and artificial bee colony algorithm," *Int. J. Adv. Robot. Syst.*, vol. 17, no. 1, 2020, Art. no. 1729881419894787.
[29] H. I. Kang, B. Lee, and K. Kim, "Path planning algorithm using the particle swarm optimization and the improved Dijkstra algorithm," in *Proc. IEEE Pacific–Asia Workshop Comput. Intell. Ind. Appl.*, vol. 2. Wuhan, China, 2008, pp. 1002–1004.
[30] Y. Liang and L. Xu, "Global path planning for mobile robot based genetic algorithm and modified simulated annealing algorithm," in *Proc. 1st ACM/SIGEVO Summit Genet. Evol. Comput.*, 2009, pp. 303–308.
[31] J.-W. Park, H.-D. Oh, and M.-J. Tahk, "UAV collision avoidance based on geometric approach," in *Proc. SICE Annu. Conf.*, Chofu, Japan, 2008, pp. 2122–2126.
[32] L. Yang, X. Zhang, Y. Zhang, and G. Xiangmin, "Collision free 4D path planning for multiple UAVs based on spatial refined voting mechanism and PSO approach," *Chin. J. Aeronaut.*, vol. 32, no. 6, pp. 1504–1519, 2019.
[33] P. Cai *et al.*, *Collision Detection Using Axis Aligned Bounding Boxes*. Singapore: Springer, 2014, pp. 1–14. [Online]. Available: https://doi.org/10.1007/978-981-4560-32-0_1
[34] W. R. Franklin. "PNPOLY Algorithm." 2009. [Online]. Available: http://aiweb.techfak.uni-bielefeld.de/content/bworld-robot-control-software/ (accessed 2009).

Kun Shen (Student Member, IEEE) received the B.S. degree in automation science from Jiangnan University, Wuxi, China, in 2018. He is currently pursuing the M.S. degree in electrical and computer engineering with New York Institute of Technology, New York, NY, USA.

His research interests include control systems, path planning optimization, machine learning, and graphics.

Rutuja Shivgan (Student Member, IEEE) received the B.E. degree in electrical engineering from Savitribai Phule Pune University, Pune, India, in 2016, and the M.S. degree in electrical and computer engineering from New York Institute of Technology, New York, NY, USA, in 2020.

Her research interests are robotics, machine learning, and control system.

Jorge Medina (Student Member, IEEE) received the B.Sc. degree in electrical industrial engineering from La Universidad Nacional Autónoma de Honduras, Tegucigalpa, Honduras, in 2012, and the M.Sc. degree in telecommunication systems from the Blekinge Institute of Technology, Karlskrona, Sweden, in 2018. He is currently pursuing the Ph.D. degree with the Networking Research Laboratory, Helen and John C. Hartmann Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, USA.

His research interests are computer networking, computer optimization, blockchain, and machine learning.

Ziqian Dong (Senior Member, IEEE) received the M.S. and Ph.D. degrees in electrical engineering from New Jersey Institute of Technology, Newark, NJ, USA, in 2002 and 2008, respectively.

She is currently a Professor with the Department of Electrical and Computer Engineering, New York Institute of Technology, New York, NY, USA. Her research interests include architecture design and analysis of high-performance packet switches, data center networks, network security and forensics, wireless sensor networks, assistive medical devices, and data analytics and innovative sensing technology to improve sustainability and resilience of both natural and built environment.

Prof. Dong was a recipient of the 2006 and 2007 Hashimoto Fellowship for outstanding scholarship, the 2008 Hashimoto Prize for the best Ph.D. dissertation in electrical engineering at NJIT, the New Jersey Inventors Hall of Fame Graduate Student Award for her inventions in network switches, the NYIT Presidential Engagement Award in Student Engagement in Research and Scholarship, and the 2020 ASEE Curtis W. McGraw Research Award. She has served in the Technical Program Committee of IEEE GLOBECOM, ICC, HPSR, Sarnoff, GREENCOM, and as a reviewer for IEEE journals, conferences, and U.S. National Science Foundation panels. She is a Senior Member of the IEEE Communications Society, IEEE Women in Engineering, and a member of the American Society for Engineering Education (ASEE), ACM, and the Environmental Sensing, Networking and Decision-Making Technical Committee.

Roberto Rojas-Cessa (Senior Member, IEEE) received the B.S. degree from Universidad Veracruzana, Xalapa, Mexico, the M.S. degree from Research and Advanced Studies Center, Mexico City, Mexico, and the M.S. and Ph.D. degrees in computer and electrical engineering from Polytechnic University (currently, the New York University Tandon School of Engineering, Polytechnic Institute), Brooklyn, NY, USA.

He is currently a Professor with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology (NJIT), Newark, NJ, USA. He has authored the books *Advanced Internet Protocols, Services, and Applications* (Wiley, 2012) and *Interconnections for Computer Communications and Packet Networks* (CRC Press, 2017). His research interests include the wide area of networking, cyber–physical systems, energy, intelligent systems and learning, and emergency communications and systems.

Prof. Rojas-Cessa was a recipient of the Excellence in Teaching Award from the Newark College of Engineering at NJIT and the New Jersey Inventors Hall of Fame—Innovators Award in 2013, and the Excellence Progress in Research by the Department of Electrical and Computer Engineering in 2005. He serves in different capacities for IEEE conferences and specialized journals as a reviewer and editor, and as a Panelist for the U.S. National Science Foundation and the U.S. Department of Energy. He was the General Chair of IEEE Sarnoff Symposium 2011 and IEEE International Conference on High-Performance Switching and Routing 2020. In addition, he has been a Technical Program Committee Chair of the two flagship conferences of the Communications Society: International Conference on Communications and Global Communications. He was an Invited Fellow of the Japanese Society for the Advancement of Science in 2009.