Completion Time Minimization for UAV-Enabled Surveillance Over Multiple Restricted Regions

Hsiang-Chun Tsai, Y.-W. Peter Hong, *Senior Member, IEEE*, and Jang-Ping Sheu, *Fellow, IEEE*

Abstract—This work examines a UAV-enabled surveillance mission over multiple restricted regions and aims to determine the optimal UAV trajectory that minimizes the mission completion time. The UAV is prohibited from entering the restricted regions due to government regulations or adversarial concerns. However, during the surveillance of a region, the UAV can move along the region's boundary to reduce its distance to the next region once the local task is completed. To exploit this advantage, we propose a minimum completion time (MinTime) algorithm that first determines the visiting order of the regions by employing an approximate solution of the traveling salesman problem (TSP) and then optimizes the UAV trajectory over the sequence of restricted regions using dynamic programming. In the presence of obstacles, we further propose an obstacle-aware MinTime (OA-MinTime) algorithm that treats each obstacle as an additional restricted region with zero surveillance duration, allowing the UAV to avoid the obstacles in a more efficient manner. A modified TSP solution is also proposed by taking into consideration the additional distance required to circumvent the obstacles on each inter-POI path. Simulation results show that the proposed MinTime and OA-MinTime algorithms can significantly reduce the total completion time compared to conventional minimum-distance approaches.

Index Terms—UAV communications, trajectory optimization, wireless sensor networks, data gathering, dynamic programming

1 Introduction

Unmanned aerial vehicles (UAVs) have been widely adopted in many civilian and military applications, such as agriculture, disaster recovery, battlefield surveillance, etc., due to their high mobility and maneuverability. The use of UAVs as mobile base-stations [1] or relays in cellular networks [2] has also received much attention due to their deployment flexibility and the ability to avoid signal blockage and shadowing at a high altitude [3]. In internet-of-things (IoT) or wireless sensor networks (WSNs), UAVs have also been adopted as mobile sink nodes that traverse the network to gather information from the sensors [4], [5]. By treating the UAV as a mobile sensor, it can also be dispatched to perform sensing or surveillance tasks over multiple points-of-interest (POIs) in the area [6]. While most works in the literature consider cases where the POIs are directly reachable, we are interested in applications where POIs may be located within restricted regions that prohibit the UAV from entering due to government regulations or adversarial concerns. For example, in battlefield surveillance, the UAV may be prohibited from entering certain regions to avoid detection by opposing radars or attack by enemy forces. In disaster recovery, the UAV may be prohibited from entering contaminated regions or collapsed buildings and disaster sites. The scenario under consideration may also arise in everyday life where data must be gathered from devices located within buildings or regulated no-fly zones.

The use of UAVs for surveillance and data collection has been widely considered in several recent works, such as [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], focusing on the UAV trajectory design under various system requirements. For example, for surveillance applications, several works, e.g., [7], [8], [9], [10], [11], focused on minimizing the UAV's energy consumption under requirements on the detection probability or inspection criteria. For data gathering applications, UAVs' trajectory designs have been proposed by minimizing the energy or time consumption [12], [13], [14], [15] or by maximizing the amount of data that is collected [16], [17], [18]. These issues have been examined for both single-UAV and multi-UAV scenarios. However, in the works mentioned above, the POIs or ground terminals are often assumed to be directly accessible by the UAVs without any constraints on the surrounding environment. However, in many practical applications, such as battlefield surveillance or disaster recovery missions, the POIs may often be located within restricted regions that prohibit the UAV from entering. In this case, the UAV may only be allowed to travel along the boundaries of the restricted regions when performing the surveillance tasks. This imposes non-convex constraints on the UAV's trajectory that make the optimization problem difficult to solve in general.

- *Hsiang-Chun Tsai is with the Department of Computer Science, National Tsing Hua University, Hsinchu 300, Taiwan. E-mail: zx7054856@gmail.com.*
- *Y.-W. Peter Hong is with the Institute of Communications Engineering, National Tsing Hua University, Hsinchu 300, Taiwan. E-mail: ywhong@ee.nthu.edu.tw.*
- *Jang-Ping Sheu is with the Department of Computer Science and the Institute of Communications Engineering, National Tsing Hua University, Hsinchu 300, Taiwan. E-mail: sheujp@cs.nthu.edu.tw.*

Manuscript received 1 March 2022; revised 26 July 2022; accepted 15 August 2022. Date of publication 22 August 2022; date of current version 3 November 2023.
This work was supported by the Ministry of Science and Technology, Taiwan, under Grants MOST 109-2221-E-007-079-MY3, 110-2634-F-007-021, and 111-2221-E-007-042-MY3.
(Corresponding author: Y.-W. Peter Hong.)
This article has supplementary downloadable material available at https://doi.org/10.1109/TMC.2022.3200732, provided by the authors.
Digital Object Identifier no. 10.1109/TMC.2022.3200732

In this work, we consider a UAV-enabled surveillance mission over multiple POIs that are located within restricted regions, and aim to determine the optimal UAV trajectory that minimizes the total completion time of the missions. The surveillance of each POI requires the UAV to maintain at the boundary of the corresponding restricted region for a required duration of time. However, instead of staying at a fixed position during the surveillance duration, the UAV may move along the boundary of the restricted region while gathering information from the POI to reduce its distance to the next region once the local surveillance task is completed. This advantage is exploited in our work to develop an efficient trajectory design for UAV-enabled surveillance missions. The main contributions of this work can be summarized as follows:

- We examine a novel UAV trajectory design problem that takes into consideration the presence of restricted regions around the POIs that prevent the UAV from entering to perform the surveillance task, e.g., in battlefield or disaster recovery missions.
- By leveraging the surveillance time for traveling along the POI boundaries, we propose a minimum completion time (MinTime) algorithm that first determines the visiting order of the restricted regions using an approximate solution for the traveling salesman problem (TSP) and then optimizes the UAV trajectory over the boundary of the restricted regions using dynamic programming (DP).
- In the presence of obstacles, we further propose an obstacle-aware MinTime (OA-MinTime) algorithm that treats the obstacles as additional restricted regions with required surveillance duration equal to zero. By doing so, the UAV trajectory can be optimized to bypass the obstacles in a more time-efficient manner. The TSP solution can also be updated by taking into consideration the travel distance required to circumvent obstacles on the inter-POI path.
- Simulation results show that the MinTime algorithm is able to reduce the completion time by approximately 10% compared to conventional minimum distance algorithms while the OA-MinTime algorithm further improves by approximately 4% in the presence of obstacles. In certain special cases, where the field of interest is partitioned by large obstacles, the obstacle-aware solution is able to improve by more than 30% compared to the basic MinTime algorithm.

To the best of our knowledge, this is the first work that examines data gathering over POIs that are within restricted regions. While existing works, e.g., [19], [20], have considered restricted regions or flight obstacles that obstruct the flight trajectory in between POIs, they did not consider the case where POIs are within restricted regions and thus focused only on finding alternative paths to circumvent obstacles as UAVs travel from one POI to the other. In our case, the UAV must fly along the boundary of the restricted region for a required amount of time to capture the information rather than just avoid the obstacle on the path to the next destination.

The remainder of this paper is organized as follows. Section 2 reviews several related works on UAV trajectory designs for surveillance and data collection. Section 3 presents the system model and problem formulation. Sections 4 and 5 describe the proposed MinTime and OA-MinTime algorithms. Moreover, Section 6 provides the simulation results, and Section 7 concludes the paper.

2 Related Work

UAV trajectory design for surveillance and data collection was studied in several recent works, focusing mostly on maximizing the quality or amount of information that is gathered or on minimizing the energy and time consumption under task completion requirements. Specifically, for surveillance applications, [7] proposed a single-UAV trajectory optimization scheme for maritime radar wide-area persistent surveillance that simultaneously minimizes fuel consumption, maximizes detection probability, and minimizes mean revisit time. [8] considered a UAV-aided image surveillance of distant targets, where the data collected by its camera is transmitted back to a ground terminal for further processing. The average power consumption of the feedback transmission by the UAV is minimized under a delay constraint. For multi-UAV networks, [9] proposed an energy-aware stochastic surveillance policy for multiple UAVs that considers stochastic inspection policies and limited battery capacities. Randomness was introduced into both the moving patterns and the inspection requirements to address security concerns of conventional deterministic operations. Moreover, [10] focused on the navigation of a team of UAVs for the image surveillance of a group of moving pedestrians or vehicles, and [11] proposed a trajectory planner for the surveillance of a certain operational area to detect the existence of illegal UAVs.

For data collection in WSNs, several recent works proposed UAV trajectory designs that take into consideration the energy consumption of both the sensors and the UAV. Specifically, [12] considered the dispatch of a single UAV for the collection of a given amount of data from a ground terminal, and proposed a trajectory design that considers the tradeoff between transmission and propulsion energy consumption. In [21], a joint design of the sensor clustering, forwarding-tree construction, and UAV trajectory was proposed with the goal of minimizing an objective function that takes into account both the energy consumption of the sensors and the UAV's travel distance. The work considered a compressive data gathering solution where sensors' observations are aggregated along the path to the clusters and are recovered at a central sink node using sparse reconstruction methods. A similar problem was also examined in [22] for agricultural monitoring applications by considering specific energy consumption models for both the UAV and the sensors. In addition, [23] examined a UAV-enabled data collection problem for massive machine-type communications (mMTC), where both the communication devices and the UAV are battery limited. The device clustering and the UAV's hovering and flying strategies were jointly determined to minimize the overall energy consumption using a proposed artificial energy map (AEM). Moreover, [16] incorporates orthogonal frequency division multiple access (OFDMA) technology to enable the UAV to collect data from multiple sensors simultaneously within its communication

range. The hovering locations of the UAV is determined by maximizing the volume of data collected under flight energy constraints. A similar problem was also considered in [17] for both full and partial data collection scenarios. Furthermore, [24] proposed a UAV trajectory design that aims to maximize the minimum residual energy of sensors after data transmission subject to data collection and UAV traveling distance constraints. The proposed solution first finds the shortest UAV path that guarantees data collection at all the sensors. This results in hovering locations that tend to be at Voronoi vertices to enable data collection from as many adjacent sensors as possible. In addition, [25] considered the joint design of the UAV trajectory and sensor activation by minimizing the mean-squared error of the reconstructed sensor observations. Similarly, [26] examined the trajectory design for data collection from distributed sensor nodes (SNs) with the goal of minimizing the estimation error of a common underlying parameter.

For multiple UAVs, [13] considered the data collection problem in IoT, and determined the optimal UAVs' placement, device-UAV association, and uplink power control to minimize the total transmit power of the IoT devices. For time-varying networks, the optimal trajectory of each UAV was also optimized by minimizing the total energy used for the mobility of the UAVs. [18] proposed a joint trajectory design and power control algorithm that maximizes the sum rate of a UAV-enabled interference channel under constraints on the UAV velocity and altitude. [27] modeled the decision making problem of a UAV-IoT wireless energy and data transmission system as a graph-based Markov decision process, and proposed a mean-field approximation algorithm to determine the best policy for each system state. Moreover, [28] proposed a two-stage maximum energy saving device association strategy, where each UAV first solves a single backpack problem locally by considering all connectable devices, and then uses a maximum profit assignment policy to resolve conflict with other UAVs. Most existing works on multiple UAVs focus on the issue of collaboration and task assignment among UAVs. As a first work on data-gathering over POIs in restricted regions, we focus on the single-UAV scenario and examine the trajectory design under flight restrictions and surveillance-time requirements. The proposed trajectory design can also be extended to the multiple-UAV scenario by splitting the surveillance region into multiple subregions, one for each UAV. However, the division of tasks and the cooperation among UAVs require further investigation that is beyond the scope of this work.

In addition to the energy consumption, many surveillance and data gathering applications are also concerned with the timeliness of the collected information. In particular, [14] considered a UAV-aided data collection problem over a set of sensors on a straight line. The UAV's trajectory (i.e., speed and hovering points), the sensors' transmission intervals, and power were determined by minimizing the UAV's total flight time from a starting point to a destination. [15] considered a multimode UAV communication platform where the UAV was used for uplink, downlink, also relay transmissions between ground users. The UAV trajectory, bandwidth and power allocation were jointly optimized by minimizing the UAV's periodic flight duration or mission completion time. [29] considered a UAV-aided data collection mission from multiple ground users, and proposed a joint design of the UAV's trajectory, altitude, speed, and data links to ground users by minimizing the total mission time. [30] considered the UAV trajectory design for two data collection missions, namely, data aggregation and field estimation. The total hovering and traveling times of the UAV were minimized under requirements on the number of observations gathered and the average reconstruction MSE in data aggregation and field estimation applications, respectively. For multiple UAVs, [31] proposed a fine-grained trajectory plan design where the detailed hovering and traveling plans of the UAVs were determined for data gathering in WSNs by minimizing the maximum time consumption of all UAVs. [32] determined both the sensor clustering and multiple UAVs' flight trajectories to minimize the data collection flight time. Trajectory designs were proposed respectively for the case where UAVs hover exactly above the visited cluster heads and the case where UAVs hover within a range of the cluster head. Moreover, [33] further investigated a multi-UAV collaborative data collection problem using a cell partitioning approach where the sensor field is divided into multiple subregions, each served by a different UAV. The data collection time of the UAV within each subregion is minimized by jointly optimizing the UAV's data collection positions, flight speed, and the devices' transmit power.

More recently, several studies also focused on optimizing the freshness of the collected data using age-of-information (AoI) as the performance indicator. For example, [34] examined a UAV-aided data collection problem for WSNs that takes into account the AoI, which is defined as the time from the instant at which the information is sensed to the instant at which the information is delivered to the data center. UAV trajectory and sensor clustering designs were proposed to minimize the maximum and the average AoI among sensors, respectively. [35] considered a similar problem for wireless powered IoT networks where the time required for energy harvesting at the IoT devices was further taken into account. [36] minimized the average AoI of the data collected from ground nodes by jointly considering the data acquisition mode selection, the sensors' energy consumption, and the age evolution of collected information.

While the above works present effective UAV trajectory designs under different scenarios, these works assume that the ground terminals or regions of interest are directly accessible by the UAVs without any constraints on the surrounding environment. That is, the UAVs are allowed to arrive arbitrarily close to the POIs for surveillance or data collection. However, in many applications, including battlefield surveillance and disaster recovery, UAVs may not be allowed to enter the area surrounding the POIs and, thus, can only gather information from the boundaries of the restricted regions. This requirement poses non-convex constraints on the trajectory optimization problem which makes it particularly challenging to solve. We address this problem by employing DP techniques in the current work. Our proposed technique is also able to take into consideration the presence of obstacles and obtain effective obstacle-aware flight trajectories. While several existing works, e.g., [19], [20], have also looked at the design of UAV flight trajectories

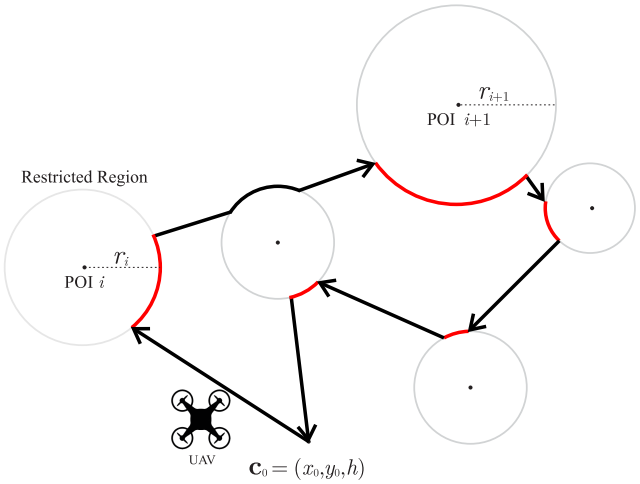


Fig. 1. Illustration of a UAV-enabled surveillance mission over multiple restricted regions.

in the presence of restricted regions or flight obstacles, most works assume that the restricted regions only obstruct the path in between POIs and, thus, focus only on finding trajectories that are able to avoid the restricted regions as the UAV travels from one location to the other. No restriction was placed on the data-gathering around the POIs. In fact, most of these works (e.g. [37], [38], [39], [40]) rely on the use of reinforcement learning to cope with the presence of obstacles. In this work, we avoid the use of black-box approaches and instead adopt DP-based techniques to address this problem analytically.

3 System Model and Problem Formulation

We consider a surveillance mission where a rotary-wing UAV is dispatched to perform surveillance over I POIs, as illustrated in Fig. 1. Each POI is located inside a restricted region that prevents the UAV from entering, e.g., due to government regulations, hostile environment, or to avoid detection by an adversary. We assume that each restricted region, say region i, takes on a circular shape with radius r_i and center coordinates given by $\mathbf{p}_i = (x_i, y_i, 0)$. Note that the shape of the restricted region may be generalized to arbitrary shapes, but is assumed to be circular here for the ease of exposition. To complete the surveillance mission at POI i, the UAV is required to stay as close as possible to the POI (i.e., at the boundary of the restricted region) for a minimum required duration of Δ_i. The order for which the UAV visits the I POIs is represented by the permutation function $\kappa : \{1, \ldots, I\} \rightarrow \{1, \ldots, I\}$, where $\kappa(i)$ represents the index of the i-th POI that is visited. The overall surveillance mission requires the UAV to start from the point $\mathbf{c}_0 = (x_0, y_0, h)$ and return back to the same point $\mathbf{c}_{I+1} = \mathbf{c}_0$, while visiting the I POIs in turn according to κ. The start and end points may be viewed as additional POIs 0 and $I+1$ with required durations $\Delta_0 = \Delta_{I+1} = 0$. The UAV is assumed to have a fixed altitude h and a maximum flight velocity $v_{\max}$. We assume that a local edge server exists at the start point (and, thus, the end point) to compute the UAV trajectory prior to dispatch. The information gathered by the UAV is then offloaded to the server once it returns back to this point when the mission is completed. We assume that the POIs and restricted regions

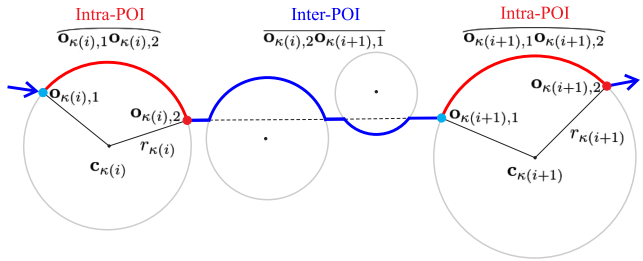


Fig. 2. Example of inter-POI and intra-POI paths.

are known a priori to enable the computation of the flight trajectory, which may be acquired from satellite images or from prior exploration (e.g., by friendly forces or previously dispatched UAVs).

Suppose that, during the visit to POI i, the UAV arrives and departs at the boundary of region i at positions $\mathbf{o}_{i,1}$ and $\mathbf{o}_{i,2}$, respectively. The path that the UAV traverses on the boundary of region i (i.e., the intra-POI path) is denoted by $\overset{\frown}{\mathbf{o}_{i,1}\mathbf{o}_{i,2}}$, as illustrated in Fig. 2, and its corresponding length is denoted by $\| \overset{\frown}{\mathbf{o}_{i,1}\mathbf{o}_{i,2}} \|$. To fulfill the minimum required surveillance duration Δ_i, the time that the UAV should stay on the boundary of region i should be at least Δ_i and, thus, is given by

$$\tau_i^{\text{intra}}(\mathbf{o}_{i,1}, \mathbf{o}_{i,2}) \triangleq \max\{\Delta_i, \| \overset{\frown}{\mathbf{o}_{i,1}\mathbf{o}_{i,2}} \|/v_{\max}\}, \tag{1}$$

where $\| \overset{\frown}{\mathbf{o}_{i,1}\mathbf{o}_{i,2}} \|/v_{\max}$ is the shortest time required for the UAV to travel the distance $\| \overset{\frown}{\mathbf{o}_{i,1}\mathbf{o}_{i,2}} \|$. For a circular restricted region centered at $\mathbf{c}_i = (x_i, y_i, h)$ (i.e., the point above POI i) with radius r_i, the intra-POI distance can be computed as

$$\| \overset{\frown}{\mathbf{o}_{i,1}\mathbf{o}_{i,2}} \| = r_i \min\{\theta_i, 2\pi - \theta_i\}, \tag{2}$$

where $\theta_i \triangleq \arccos \frac{(\mathbf{o}_{i,1} - \mathbf{c}_i)(\mathbf{o}_{i,2} - \mathbf{c}_i)}{\|\mathbf{o}_{i,1} - \mathbf{c}_i\| \|\mathbf{o}_{i,2} - \mathbf{c}_i\|}$. Notice that the minimum between θ_i and $2\pi - \theta_i$ is taken in (2) to account for the fact that the UAV may travel in either the counterclockwise or the clockwise direction depending on which yields the shortest arc to the departing point. By considering a rotary-wing UAV, the UAV may be allowed to hover at a fixed point for a sufficient amount of time or travel at a sufficiently slow velocity on the boundary of a restricted region to satisfy the minimum required surveillance time.

Moreover, let $\overline{\mathbf{o}_{\kappa(i),2}\mathbf{o}_{\kappa(i+1),1}}$ represent the inter-POI path that the UAV must traverse in between POI $\kappa(i)$ and POI $\kappa(i+1)$. If there is an unobstructed direct path between the two points $\mathbf{o}_{\kappa(i),2}$ and $\mathbf{o}_{\kappa(i+1),1}$, the distance that the UAV travels can be computed as $\|\overline{\mathbf{o}_{\kappa(i),2}\mathbf{o}_{\kappa(i+1),1}}\| = \|\mathbf{o}_{\kappa(i),2} - \mathbf{o}_{\kappa(i+1),1}\|$. However, if the direct path intersects with the restricted regions of other POIs, the UAV must travel on the boundaries of these regions to go around them, resulting in longer travel distances, as illustrated in Fig. 2. For example, suppose that the line connecting points $\mathbf{o}_{\kappa(i),2}$ and $\mathbf{o}_{\kappa(i+1),1}$ intersects with regions i_1, i_2, ..., i_M, and that $\mathbf{a}_{i_m,1}$ and $\mathbf{a}_{i_m,2}$ are the intersecting points on the boundary of region i_m. In this case, the distance that the UAV must travel from $\mathbf{o}_{\kappa(i),2}$ to $\mathbf{o}_{\kappa(i+1),1}$ can be computed as

$$\|\overline{\mathbf{o}_{\kappa(i),2}\mathbf{o}_{\kappa(i+1),1}}\| = \sum_{m=0}^{M}\Big(\|\widehat{\mathbf{a}_{i_m,1}\mathbf{a}_{i_m,2}}\| + \|\mathbf{a}_{i_m,2} - \mathbf{a}_{i_{m+1},1}\|\Big), \quad (3)$$

where $\mathbf{a}_{i_0,1} = \mathbf{a}_{i_0,2} = \mathbf{o}_{\kappa(i),2}$ and $\mathbf{a}_{i_{M+1},1} = \mathbf{o}_{\kappa(i+1),1}$. By flying at the maximum speed $v_{\max}$ over the inter-POI path, the time required for the UAV to traverse the inter-POI distance is given by

$$\tau^{\text{inter}}_{\kappa(i),\kappa(i+1)}(\mathbf{o}_{\kappa(i),2}, \mathbf{o}_{\kappa(i+1),1}) \triangleq \frac{\|\overline{\mathbf{o}_{\kappa(i),2}\mathbf{o}_{\kappa(i+1),1}}\|}{v_{\max}}. \quad (4)$$

For the start and end points (which are viewed as additional POIs 0 and $I+1$), we have $\mathbf{o}_{0,1} = \mathbf{o}_{0,2} = \mathbf{o}_{I+1,1} = \mathbf{o}_{I+1,2} = \mathbf{c}_0$ and, thus, $\tau^{\text{intra}}_0 = \tau^{\text{intra}}_{I+1} = 0$.

Given the ordering κ and the arrival and departure points $\{(\mathbf{o}_{i,1}, \mathbf{o}_{i,2})\}_{i=1}^{I}$, the completion time of the overall surveillance mission can be computed as

$$T(\kappa, \{(\mathbf{o}_{i,1}, \mathbf{o}_{i,2})\}_{i=1}^{I}) \triangleq \sum_{i=0}^{I} \tau^{\text{intra}}_{\kappa(i)}(\mathbf{o}_{\kappa(i),1}, \mathbf{o}_{\kappa(i),2}) + \tau^{\text{inter}}_{\kappa(i),\kappa(i+1)}(\mathbf{o}_{\kappa(i),2}, \mathbf{o}_{\kappa(i+1),1}), \quad (5)$$

where $\kappa(0) \triangleq 0$ and $\kappa(I+1) \triangleq I+1$ (i.e., the start and end points). The objective of this work is to find the optimal UAV trajectory that minimizes the total completion time of the overall surveillance mission. The trajectory is determined by the visiting order κ and the arrival and departure points $\{(\mathbf{o}_{i,1}, \mathbf{o}_{i,2})\}_{i=1}^{I}$. Notice that, since the set of feasible solutions for $\{(\mathbf{o}_{i,1}, \mathbf{o}_{i,2})\}_{i=1}^{I}$ consists of only the boundary points of the restricted regions and, thus, is non-convex, the optimization problem cannot be solved by standard convex optimization tools as done in [15]. In the following, we show that an efficient trajectory design can be obtained using DP. It is worthwhile to remark that, while energy consumption is a key factor in many UAV trajectory design problems, we focus on time-sensitive applications where the completion time is the primary concern, and assume that the energy sources at the UAV are sufficient to complete the surveillance tasks.

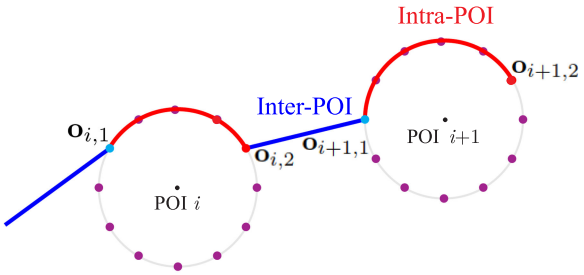
4 Minimum Completion Time Trajectory Optimization via Dynamic Programming

In this section, we describe the proposed minimum completion time (MinTime) trajectory optimization algorithm that aims to minimize the completion time of the overall surveillance mission using dynamic programming (DP). The problem involves both the choice of the visiting order κ and the optimization of the arrival and departure points $\{(\mathbf{o}_{i,1}, \mathbf{o}_{i,2})\}_{i=1}^{I}$. Notice that finding the visiting order by itself can be an NP-hard problem and, thus, joint optimization of both can be even more challenging. In this work, we adopt a two-stage approach, where the visiting order and the arrival and departure points are solved separately in consecutive order. Even though the two-stage approach cannot guarantee optimality of the solution, it is commonly adopted in the literature (e.g., in [15], [26], [29], [30]) to overcome the difficulty of joint optimization. Details of the proposed solution are described as follows.

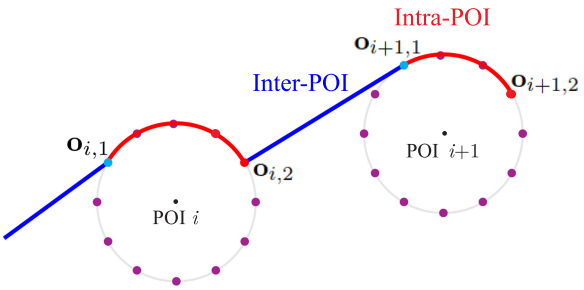
First, we examine the choice of the visiting order κ, which is analogous to a travelling salesman problem (TSP) where the path with the minimum cost, starting and ending at the same point while visiting all POIs along the way, is to be determined. Notice that our trajectory design problem is even more challenging than the traditional TSP problem in the sense that the arrival and departure points at each POI can lie anywhere on the boundary of a circle rather than at a single point. However, to reduce the complexity, we adopt a heuristic approach where the visiting order κ is first determined by solving the TSP over a graph formed by the start point $\mathbf{c}_0$ (which is also the end point $\mathbf{c}_{I+1} = \mathbf{c}_0$) and the center points of the restricted regions $\{\mathbf{c}_i\}_{i=1}^{I}$, where $\mathbf{c}_i = (x_i, y_i, h)$. To solve the TSP, we adopt an approximate algorithm described in Chapter 35 of [41], where a minimum spanning tree (MST) is first found for the complete geometric graph with nodes $\{\mathbf{c}_i\}_{i=0}^{I}$. The cost of the edge between nodes i and j is defined as the distance $\|\mathbf{c}_i - \mathbf{c}_j\|$. The MST can be obtained by adopting the standard Prim's algorithm in Chapter 23 of [41]. Suppose that the resulting MST is denoted by $\mathcal{T}$. Then, a solution to the TSP can be found by a preorder traversal of the tree $\mathcal{T}$ while skipping nodes that have already been visited previously in the traversal. This approach is known to yield an approximation ratio of 2, as shown in Chapter 35 of [41]. The desired visiting order κ is thus obtained by the TSP solution mentioned above starting with the POI that the UAV first visits after the start point $\mathbf{c}_0$. It is worthwhile to note that the proposed MinTime trajectory optimization algorithm is not specific to the TSP solution described above. Different approximate algorithms, e.g., [42], for solving the TSP can also be adopted here without affecting the operations of the proposed MinTime algorithm.

Given the visiting order κ, the parameters that remain to be determined are the arrival and departure points $\{(\mathbf{o}_{i,1}, \mathbf{o}_{i,2})\}_{i=1}^{I}$ of the POIs. To do so, we first discretize the boundary of each restricted region, say region i, into a set of discrete points, denoted by $\mathcal{O}_i$. For example, by considering a circular region centered at $\mathbf{c}_i$ with radius r_i, the boundary can be uniformly discretized into the set of J_i discrete points given by $\mathcal{O}_i \triangleq \{\mathbf{c}_i + (r_i\cos\phi_j, r_i\sin\phi_j)\}_{j=1}^{J_i}$, where $\phi_j = (j-1)2\pi/J_i$, for $j = 1, \ldots, J_i$. Increasing the number of discrete points J_i may improve the accuracy of the solution but may also increase the computational complexity of the proposed DP solution. However, we show later in our experiments that the improvement becomes limited beyond a reasonable value of J_i. By working with discrete sets of boundary points, we are able to accommodate regions of arbitrary shape since the proposed DP algorithm in the following depends only on the set of potential arrival and departure points rather than the shape of the region.

Without loss of generality, let us relabel the index of the POIs according to κ so that POIs i and $i+1$ are visited consecutively. Recall that, for any two consecutive POIs, say i and $i+1$ (after relabeling), the flight time between their departure points $\mathbf{o}_{i,2}$ and $\mathbf{o}_{i+1,2}$ is given by $\tau^{\text{inter}}_{i,i+1}(\mathbf{o}_{i,2}, \mathbf{o}_{i+1,1}) + \tau^{\text{intra}}_{i+1}(\mathbf{o}_{i+1,1}, \mathbf{o}_{i+1,2})$, which depends on the choice of the arrival point at POI $i+1$, i.e., $\mathbf{o}_{i+1,1}$, as illustrated in Fig. 3. Choosing $\mathbf{o}_{i+1,1}$ close to $\mathbf{o}_{i+1,2}$ may reduce the intra-POI flight time, but may increase the inter-POI flight time, and vice versa. Therefore, we define the cost of choosing the pair of departure points $\mathbf{o}_{i,2}$ and $\mathbf{o}_{i+1,2}$ as



(a) Example of choosing $\mathbf{o}_{i+1,1}$ closer to $\mathbf{o}_{i,2}$.



(b) Example of choosing $\mathbf{o}_{i+1,1}$ closer to $\mathbf{o}_{i+1,2}$.

Fig. 3. Illustration of the impact of choosing $\mathbf{o}_{i+1,1}$ between departure points $\mathbf{o}_{i,2}$ and $\mathbf{o}_{i+1,2}$.

$$\tau^{\min}_{i,i+1}(\mathbf{o}_{i,2},\mathbf{o}_{i+1,2}) \triangleq \min_{\mathbf{o}_{i+1,1}\in\mathcal{O}_{i+1}}\Big[\tau^{\text{inter}}_{i,i+1}(\mathbf{o}_{i,2},\mathbf{o}_{i+1,1})+\tau^{\text{intra}}_{i+1}(\mathbf{o}_{i+1,1},\mathbf{o}_{i+1,2})\Big], \quad (6)$$

which represents the minimum time required to travel between the two points $\mathbf{o}_{i,2}$ and $\mathbf{o}_{i+1,2}$ over all possible choices of $\mathbf{o}_{i+1,1}$. By the above cost measure, the problem reduces to finding the sequence of departure points $\{\mathbf{o}_{i,2}\}_{i=1}^{I}$ that minimizes the overall flight time, which can be formulated as

$$\min_{\mathbf{o}_{i,2}\in\mathcal{O}_i,\forall i}\sum_{i=0}^{I}\tau^{\min}_{i,i+1}(\mathbf{o}_{i,2},\mathbf{o}_{i+1,2}). \quad (7)$$

In fact, the problem can be solved by DP following the sequence of Bellman equations given by

$$\text{MinTime}_{i+1}(\mathbf{o}_{i+1,2}) = \min_{\mathbf{o}_{i,2}\in\mathcal{O}_i}\Big[\text{MinTime}_i(\mathbf{o}_{i,2})+\tau^{\min}_{i,i+1}(\mathbf{o}_{i,2},\mathbf{o}_{i+1,2})\Big], \quad (8)$$

for all $\mathbf{o}_{i+1,2}\in\mathcal{O}_{i+1}$ and for $i=0,\ldots,I$, where $\mathcal{O}_0=\mathcal{O}_{I+1}=\{\mathbf{c}_0\}$ and $\text{MinTime}_0(\mathbf{c}_0)=0$. Here, $\text{MinTime}_i(\mathbf{o}_{i,2})$ represents the minimum time cost that can be achieved leading up to the point $\mathbf{o}_{i,2}$ on the boundary of region i. The proposed DP-based solution is summarized in Algorithm 1.

Specifically, in Algorithm 1, the visiting order κ is first determined by the MST-based TSP solution in Step 1. The POIs are then relabeled and their inter-POI and intra-POI flight times (i.e., $\tau^{\text{inter}}_{i,i+1}(\mathbf{o}_{i,2},\mathbf{o}_{i+1,1})$, $\tau^{\text{intra}}_{i+1}(\mathbf{o}_{i+1,1},\mathbf{o}_{i+1,2})$ as well as the costs $\tau^{\min}_{i,i+1}(\mathbf{o}_{i,2},\mathbf{o}_{i+1,2})$, for all $\mathbf{o}_{i,2},\mathbf{o}_{i+1,1},\mathbf{o}_{i+1,2}$ and for $i=0,\ldots,I$) are computed in Steps 2 and 3, respectively. Then, in Steps 4 to 10, the minimum time cost $\text{MinTime}_{i+1}(\mathbf{o}_{i+1,2})$ and the corresponding points $\tilde{\mathbf{o}}_{i,2}$ and $\tilde{\mathbf{o}}_{i+1,1}$ preceding the point $\mathbf{o}_{i+1,2}$ in the corresponding path are computed, for all $\mathbf{o}_{i+1,2}\in\mathcal{O}_{i+1}$ and for $i=0,\ldots,I$. The optimal set of arriving and departing points are then found by walking backwards from POI $I+1$, and is recorded in the set $Path$. The computational complexity of Algorithm 1 is analyzed in the following.

Algorithm 1. MinTime Trajectory Optimization Algorithm

Input: POI locations $\{\mathbf{p}_i\}_{i=1}^{I}$, radii $\{r_i\}_{i=1}^{I}$, surveillance durations $\{\Delta_i\}_{i=1}^{I}$, and maximum velocity $v_{\max}$.
Output: Visiting order κ and arriving and departing points $\{(\mathbf{o}_{i,1},\mathbf{o}_{i,2})\}_{i=1}^{I}$.

1: Calculate the edge costs $\|\mathbf{c}_i-\mathbf{c}_j\|$, for all i and j, and find the corresponding visiting order κ using the MST-based TSP solution.
2: Relabel the POIs such that $\kappa(i)=i$, for $i=1,\ldots,I$.
3: Compute $\tau^{\text{inter}}_{i,i+1}(\mathbf{o}_{i,2},\mathbf{o}_{i+1,1})$ and $\tau^{\text{intra}}_{i+1}(\mathbf{o}_{i+1,1},\mathbf{o}_{i+1,2})$ (and, thus, $0,\ldots,I$, where $\mathcal{O}_0=\mathcal{O}_{I+1}=\{\mathbf{c}_0\}$.
4: Set $\text{MinTime}_0(\mathbf{c}_0)=0$.
5: **for** $i=0$ to I **do**
6: **for** $\mathbf{o}_{i+1,2}\in\mathcal{O}_{i+1}$ **do**
7: Compute $\text{MinTime}_{i+1}(\mathbf{o}_{i+1,2})=\min_{\mathbf{o}_{i,2}\in\mathcal{O}_i}[\text{MinTime}_i(\mathbf{o}_{i,2})+\tau^{\min}_{i,i+1}(\mathbf{o}_{i,2},\mathbf{o}_{i+1,2})]$ as in (8).
8: Set $\text{PrePath}_{i+1}(\mathbf{o}_{i+1,2})\leftarrow(\tilde{\mathbf{o}}_{i,2},\tilde{\mathbf{o}}_{i+1,1})$, where $\tilde{\mathbf{o}}_{i,2}=\arg\min_{\mathbf{o}_{i,2}\in\mathcal{O}_i}[\text{MinTime}_i(\mathbf{o}_{i,2})+\tau^{\min}_{i,i+1}(\mathbf{o}_{i,2},\mathbf{o}_{i+1,2})]$ and $\tilde{\mathbf{o}}_{i+1,1}=\arg\min_{\mathbf{o}_{i+1,1}\in\mathcal{O}_{i+1}}[\tau^{\text{inter}}_{i,i+1}(\tilde{\mathbf{o}}_{i,2},\mathbf{o}_{i+1,1})+\tau^{\text{intra}}_{i+1}(\mathbf{o}_{i+1,1},\mathbf{o}_{i+1,2})]$.
9: **end for**
10: **end for**
11: Set $\text{Path}=\emptyset$.
12: **for** $i=0$ to I **do**
13: Set $\text{Path}\leftarrow\text{PrePath}_{I+1-i}(\mathbf{o}_{I+1-i,2})\cup\text{Path}$.
14: **end for**
15: Return Path as the optimal set of arriving and departing points $\{(\mathbf{o}_{i,1},\mathbf{o}_{i,2})\}_{i=1}^{I}$.

Complexity Analysis. In Step 1, we calculate the costs of all edges and utilize them to compute the MST-based TSP solution. Since there is a total of $(I+1)^2$ possible edges among POIs 0 to I, the computational complexity required is $O(I^2)$. Moreover, the complexity of the MST-based TSP solution is $O(I^2\log I)$, as shown in Chapter 35 of [41]. In Step 2, we relabel the POIs in the visiting order, which requires $O(I)$ complexity. Then, in Step 3, we compute $\tau^{\text{inter}}_{i,i+1}(\mathbf{o}_{i,2},\mathbf{o}_{i+1,1})$, $\tau^{\text{intra}}_{i+1}(\mathbf{o}_{i+1,1},\mathbf{o}_{i+1,2})$, and $\tau^{\min}_{i,i+1}(\mathbf{o}_{i,2},\mathbf{o}_{i+1,2})$, for all $\mathbf{o}_{i,2},\mathbf{o}_{i+1,1}$, $\mathbf{o}_{i+1,2}$ and for $i=0,\ldots,I$. Notice that each term must be computed for at most $J^2_{\max}$ different inputs and for I possible indices. Moreover, each value of $\tau^{\text{inter}}_{i,i+1}(\mathbf{o}_{i,2},\mathbf{o}_{i+1,1})$ requires an additional search over all I possible restricted regions to determine whether or not the regions are intersected by the inter-POI path, and each value of $\tau^{\min}_{i,i+1}(\mathbf{o}_{i,2},\mathbf{o}_{i+1,2})$ requires a search over at most $J_{\max}$ possible values of $\mathbf{o}_{i+1,1}$. Therefore, the overall complexity of Step 3 is thus given by $O(I^2J^2_{\max}+IJ^2_{\max}+IJ^3_{\max})$. Furthermore, the DP in Steps 4 to 10 has complexity $O(IJ^2_{\max})$ since there are at most $J_{\max}$ possible states in each stage of DP. Finally, the backtracking of the MinTime path in Steps 11 to 14 requires only $O(I)$. Hence, the overall time complexity of the MinTime algorithm is dominated by the TSP solution in Steps 1 and 3 and, thus, is given by $O(I^2\log I+I^2J^2_{\max}+IJ^3_{\max})$.

Remark: In the case of multiple UAVs, it is possible to split the surveillance region into multiple subregions, one

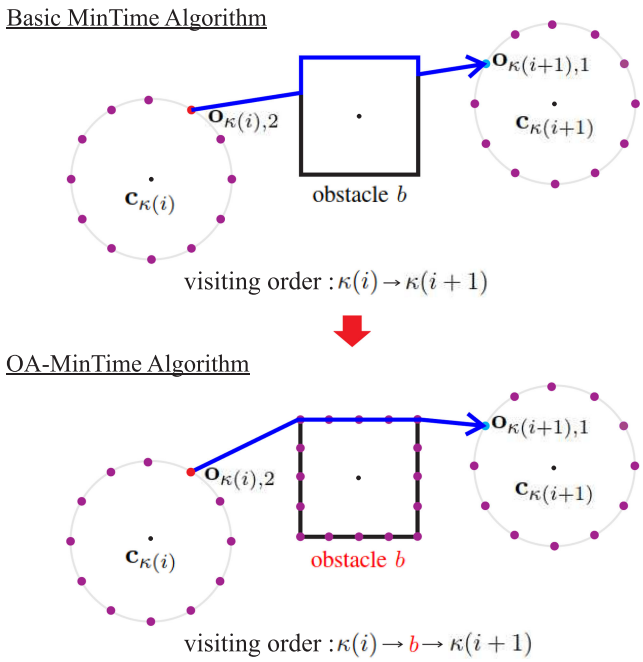
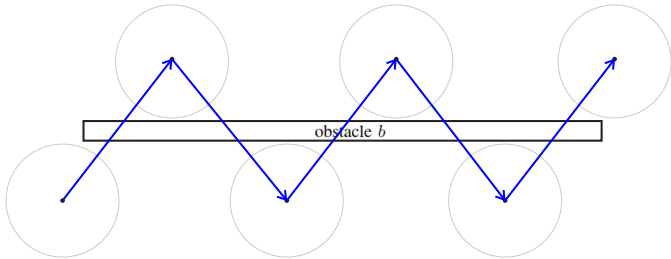
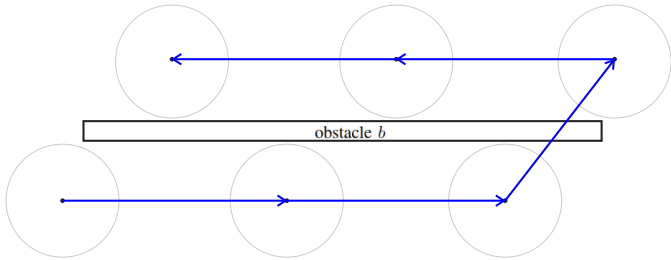


Fig. 4. An example of the OA-MinTime algorithm that treats the obstacle as an effective POI with required surveillance time equal to 0. The trajectory is optimized to circumvent the obstacle in a more time-efficient manner.



(a) The original TSP path that is obtained by taking the direct distance between the centers of POIs as the edge cost.



(b) An updated TSP path that takes into account the increased inter-POI distance due to obstacle avoidance.

Fig. 5. An example of the case where the POIs are separated by a single large obstacle.

for each UAV. In this case, each UAV can utilize the DP-based solution given above to find the minimum completion time for each subset of POIs. The task remaining is to determine the optimal partitioning of POIs that achieves the minimum overall completion time across all UAVs. However, the division of tasks and the cooperation among UAVs require further investigation that is beyond the scope of this work, but is an interesting direction for future studies.

5 Extension of the MinTime Algorithm to the Case With Obstacles

In the previous section, the MinTime algorithm was proposed by considering an ideal environment where UAVs are able to traverse the field without encountering any obstacles. However, the presence of obstacles is inevitable in practice and may impact the effectiveness of the proposed algorithm. In this section, we take into consideration the presence of flight obstacles, and modify the proposed MinTime algorithm to bypass obstacles in a more time-efficient manner. The obstacles may refer to both the restricted regions of other POIs and non-mission related buildings or no-fly zones. The key idea is to insert the obstacles (or restricted regions of other POIs) into the visiting sequence, treating them as effective POIs with surveillance durations equal to 0, and to rerun the DP procedure using the updated sequence so that the arrival and departure points at the obstacles are also optimized to reduce the total flight time. The TSP solution can also be updated to take into account the increased inter-POI distance due to the avoidance of obstacles. The resulting algorithm is referred to as the obstacle-aware MinTime (OA-MinTime) algorithm.

Suppose that the visiting order κ and the arrival and departure points $\{\mathbf{o}_{i,1}, \mathbf{o}_{i,2}\}_{i=1}^{I}$ have been determined by Algorithm 1, and that the resulting inter-POI path between $\kappa(i)$ and $\kappa(i+1)$ passes through the ordered set of obstacles $\mathcal{B}_{\kappa(i),\kappa(i+1)} = \{b_1, b_2, \ldots, b_M\}$. That is, the line connecting points $\mathbf{o}_{\kappa(i),2}$ and $\mathbf{o}_{\kappa(i+1),1}$ intersect the body of the obstacles specified by the index set $\mathcal{B}_{\kappa(i),\kappa(i+1)}$. In the basic MinTime algorithm, each obstacle is bypassed by traveling on the boundary of the obstacle from one intersecting point to the other, as illustrated in the top half of Fig. 4. Notice that this may not be the shortest flight path for the UAV between points $\mathbf{o}_{\kappa(i),2}$ and $\mathbf{o}_{\kappa(i+1),1}$. To further reduce the inter-POI travel time, we propose to treat each obstacle, say obstacle b, as an additional restricted region (or effective POI) that must also be visited, but with required surveillance duration $\Delta_b = 0$. By doing so, the UAV is able to directly travel to a point on the boundary of the obstacle that minimizes its distance towards the next POI (c.f., the bottom half of Fig. 4). By inserting the obstacles into the sequence of POIs, we obtain an updated sequence of effective POIs given by $\mathcal{B}_{\kappa(0),\kappa(1)}, \kappa(1), \mathcal{B}_{\kappa(1),\kappa(2)}, \ldots, \kappa(I), \mathcal{B}_{\kappa(I),\kappa(I+1)}$. The MinTime algorithm is then executed again on the updated sequence that is inclusive of the obstacles as additional restricted regions with the required surveillance durations set to 0. The inter-POI distance is thus reduced by avoiding the intermediate obstacles in a more efficient manner.

In addition to avoiding obstacles on the inter-POI path, we observe that the initial TSP solution that utilizes the direct distance between the center of the restricted regions as the edge cost may result in TSP paths that are extremely inefficient. For example, let us consider the special case, where the POIs are separated by a single large obstacle, as illustrated in Fig. 5. In this example, the distance between the center of the POIs on opposite sides of the obstacle are smaller than those on the same side, causing the TSP solution to choose paths that repeatedly cross the obstacle. In practice, this would require the UAV to go back and forth around the obstacle, resulting in a large overall travel distance. In this case, it would be more efficient to consider a

Algorithm 2. OA-MinTime Trajectory Optimization Algorithm

Input: POI locations $\{\mathbf{p}_i\}_{i=1}^I$, radii $\{r_i\}_{i=1}^I$, surveillance durations $\{\Delta_i\}_{i=1}^I$, maximum velocity $v_{\max}$, and obstacle locations.
Output: Visiting order κ and arrival and departure points of POIs and obstacles.
1: Calculate the modified edge costs for all i and j according to (9), and find the corresponding visiting order κ using the MST-based TSP solution
2: Execute Steps 2 to 15 of the MinTime algorithm to obtain an initial solution for the arriving and departing points $\{(\mathbf{o}_{i,1}, \mathbf{o}_{i,2})\}_{i=1}^I$.
3: Set $\mathcal{B}_{\kappa(i),\kappa(i+1)}$ as the set of obstacles that intersects with the line from $\mathbf{o}_{\kappa(i),2}$ to $\mathbf{o}_{\kappa(i+1),1}$, for $i = 0, \ldots, I$.
4: Update the visiting sequence as $\mathcal{B}_{\kappa(0),\kappa(1)}, \kappa(1), \mathcal{B}_{\kappa(1),\kappa(2)}, \ldots, \kappa(I), \mathcal{B}_{\kappa(I),\kappa(I+1)}$ and relabel the POIs and obstacles in order from 1 to I', where $I' = I + \sum_{i=0}^I |\mathcal{B}_{\kappa(i),\kappa(i+1)}|$ is the total number of effective POIs (including the actual POIs and the obstacles intersected by the inter-POI paths).
5: Execute Steps 2 to 15 of the MinTime algorithm on the sequence of effective POIs to obtain their respective arriving and departing points.

TSP path that visits POIs on the same side of the obstacle as consecutive nodes on the path. To do so, we update the TSP solution by taking into consideration the increased inter-POI distance that may be traveled by the UAV to circumvent the intermediate obstacles in between two POIs.

Specifically, suppose that the line connecting the centers of POIs i and j intersects with obstacles $b_1, b_2, \ldots, b_M$, and that $\mathbf{a}_{b_m,1}$ and $\mathbf{a}_{b_m,2}$ are the intersecting points on the boundary of obstacle b_m. With a slight abuse of notation, we denote the path from $\mathbf{a}_{b_m,1}$ to $\mathbf{a}_{b_m,2}$ on the boundary of obstacle b_m by $\mathbf{a}_{b_m,1}\widehat{\mathbf{a}_{b_m,2}}$. Then, instead of taking the distance $\|\mathbf{c}_i - \mathbf{c}_j\|$ as the cost of the edge between nodes i and j for obtaining the TSP solution, we instead consider the cost as

$$\sum_{m=0}^{M} \Big(\| \widehat{\mathbf{a}_{b_m,1}\mathbf{a}_{b_m,2}} \| + \|\mathbf{a}_{b_m,2} - \mathbf{a}_{b_{m+1},1}\| \Big), \tag{9}$$

where $\mathbf{a}_{b_0,1} = \mathbf{a}_{b_0,2} = \mathbf{c}_i$ and $\mathbf{a}_{b_{M+1},1} = \mathbf{c}_j$. Notice that the modified cost takes into account the distance required to circumvent obstacles in between POIs i and j. The TSP path is then found by following similar procedures as in Section 4 using (9) as the modified cost between edges i and j. Details of the OA-MinTime algorithm is given in Algorithm 2.

More specifically, in Algorithm 2, the edge costs required for computing the TSP solution is calculated in Step 1 using (9), and uses this information to obtain the initial visiting order κ. Then, the MinTime algorithm is executed in Step 2 to obtain the arrival and departure points of each POI. In Step 3, we check whether or not there are obstacles on the inter-POI paths of consecutive POIs, and determine the sets of obstacles that are intersected by these paths, i.e., the sets $\mathcal{B}_{\kappa(i),\kappa(i+1)}$, for $i = 0, \ldots, I$. In Step 4, we update the visiting order by inserting the sets of obstacles in between the corresponding POIs. Finally, in Step 5, the MinTime algorithm is executed again by treating the inserted obstacles as effective POIs with required surveillance durations equal to 0. Note that, while it is possible to update the visiting sequence again if additional obstacles are encountered after the MinTime path is updated in Step 5, we find that this occurs only in rare cases and the advantage of performing an additional update is limited. Furthermore, by adopting the DP-based approach on discrete points of the boundary regions, the proposed algorithm is able to handle obstacles of arbitrary shape by modifying the formula in (2) for computing the length of the intra-POI path.

Complexity Analysis. Notice that, different from Algorithm 1, the calculation of the modified edge costs in Step 1 of Algorithm 2 requires complexity of $O(I^2(I+M))$ since the cost of each edge involves the search over all possible obstacles or restricted regions that may be intersected by the inter-POI path. However, the complexity of the MST-based TSP solution is still $O(I^2 \log I)$. In Step 2 of Algorithm 2, Steps 2 to 15 of the original MinTime algorithm (i.e., Algorithm 1) is executed over the I POIs and, thus, the complexity is $O(I^2 J_{\max}^2 + I J_{\max}^3)$ as described in Section 4 (without considering the complexity of the MST-based TSP solution in Step 1 of Algorithm 1). In Step 3, we determine the sets of obstacles (or restricted regions) that are intersected by the line between the departing and arriving points, respectively, of consecutive POIs, which requires complexity of $O(I(I+M))$. In Step 4, the visiting sequence is updated and the effective POIs are related, which requires complexity of $O(I')$. Finally, the execution of the MinTime algorithm over the I' effective POIs in Step 5 requires complexity of $O(I'^2 J_{\max}^2 + I' J_{\max}^3)$. The overall complexity of the OA-MinTime algorithm is dominated by that of Steps 1 and 5 and, thus, is given by $O(I^2(I+M) + I'^2 J_{\max}^2 + I' J_{\max}^3)$.

Remark: It is worthwhile to note that, in addition to the DP-based algorithm proposed in this work, it is also possible to obtain solutions using learning-based approaches, which has attracted much attention in recent literature. Specifically, given the number of target POIs, deep learning can be utilized to produce the arrival and departure points of the POIs. However, the dimension of the neural network model increases rapidly as the number of POIs increases. To reduce the model size, one can adopt RNN-type models, such as LSTM or gated-RNN etc, where the POIs are input sequentially into the network. Alternatively, one can also adopt reinforcement learning methods where the arrival and departure points of each POI can be determined dynamically to minimize the expected discounted reward (e.g., the completion time). While the above approaches are possible, they typically require significant training overhead, way beyond the computation time of DP. The design of efficient learning algorithms for the proposed problem is interesting and nontrivial, but is beyond the scope of this work.

6 Simulation Results

In this section, we provide computer simulations to demonstrate the effectiveness of the proposed trajectory optimization algorithms. In these experiments, the POIs are placed independently according to a uniform distribution in an $1500 \times 1500\,\mathrm{m}^2$ area. Unless mentioned otherwise, we assume that the number of POIs is $I = 50$, the radii of the circular restricted regions are uniformly distributed within [50,100] meters, and the minimum required surveillance durations are also uniformly distributed within [20,50] seconds. The

maximum flight velocity is set as $v_{\max} = 5$ m/s in between POIs, and the discrete points on the circular boundaries of the regions are equally spaced by 20 meters. That is, the number of discrete points, i.e., J_i, is uniformly distributed between 16 and 31. The results are averaged over 20 realizations of the POI deployment.

The proposed MinTime algorithm is compared with three baseline algorithms, namely, the minimum distance (MinDist) algorithm by convex optimization (MinDist-CVX) [15], the MinDist algorithm by DP (MinDist-DP), and the Hover-and-Fly strategy. The MinDist-CVX algorithm is adapted from the solution proposed in [15], where the minimum distance trajectory is determined without consideration of the restricted regions around the POIs. That is, following the approach in [15], we first find an initial trajectory by solving the following optimization problem:

$$\min_{\{\mathbf{o}_i\}_{i=0}^{I}} \sum_{i=1}^{I} \|\mathbf{o}_{i-1} - \mathbf{o}_i\| \tag{10a}$$

$$\text{subject to } \|\mathbf{o}_i - \mathbf{c}_i\| < r_i, \text{ for } i = 0, \ldots, I. \tag{10b}$$

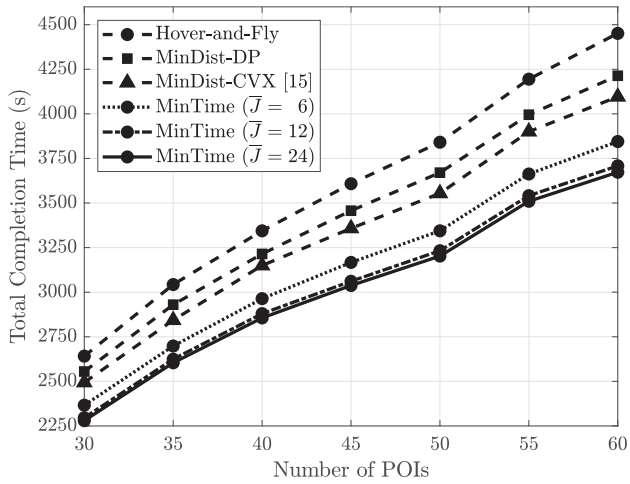
This is a convex optimization problem that can be solved by off-the-shelf solvers, such as CVX. Then, to avoid entering a restricted region, we take the two points of the region boundary that are intersecting with the initial trajectory as the arrival and departure points. The UAV travels along the boundary from the arrival point to the departure point while performing surveillance and hovers at the departure point for a certain amount of time before leaving for the next POI if the required surveillance time is not yet satisfied. On the other hand, the MinDist-DP algorithm follows the exact same procedure as that described in Algorithm 1, except for a different cost function. In particular, the cost function here is defined as the travel distance rather than the completion time. In this case, the DP solution is computed using the distance between POIs as the edge cost (rather than the travel time). By letting $d_{i+1}^{\text{intra}}(\mathbf{o}_{i+1,1}, \mathbf{o}_{i+1,2}) = \| \mathbf{o}_{i+1,1}\widehat{\mathbf{o}_{i+1,2}} \|$ and $d_{i,i+1}^{\text{inter}}(\mathbf{o}_{i,2}, \mathbf{o}_{i+1,1}) = \|\overline{\mathbf{o}_{i,2}\mathbf{o}_{i+1,1}}\|$ be the intra-POI and inter-POI distances, as defined in (2) and (3), respectively, the Bellman equations used in the MinDist-DP algorithm can be written as

$$\begin{aligned} &\text{MinDist}_{i+1}(\mathbf{o}_{i+1,2}) \\ &= \min_{\mathbf{o}_{i,2} \in \mathcal{O}_i} \Big[\text{MinDist}_i(\mathbf{o}_{i,2}) + d_{i,i+1}^{\min}(\mathbf{o}_{i,2}, \mathbf{o}_{i+1,2}) \Big], \end{aligned} \tag{11}$$

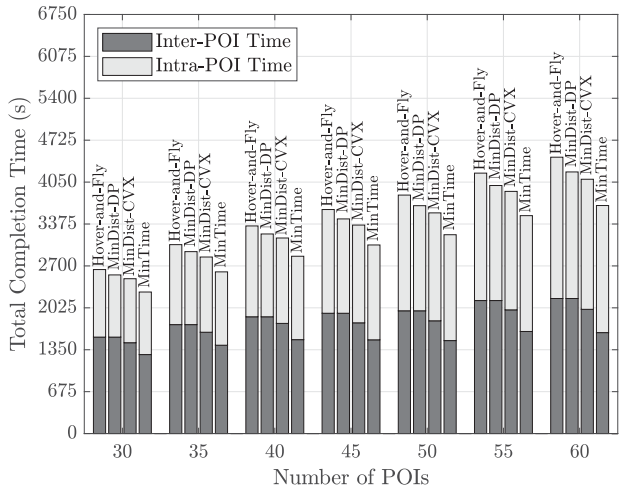
for all $\mathbf{o}_{i+1,2} \in \mathcal{O}_{i+1}$ and for $i = 0, \ldots, I$, where

$$\begin{aligned} &d_{i,i+1}^{\min}(\mathbf{o}_{i,2}, \mathbf{o}_{i+1,2}) \\ &\triangleq \min_{\mathbf{o}_{i+1,1} \in \mathcal{O}_{i+1}} \Big[d_{i,i+1}^{\text{inter}}(\mathbf{o}_{i,2}, \mathbf{o}_{i+1,1}) + d_{i+1}^{\text{intra}}(\mathbf{o}_{i+1,1}, \mathbf{o}_{i+1,2}) \Big]. \end{aligned}$$

Finally, in the Hover-and-Fly strategy, the UAV is only allowed to hover above a fixed point whenever it is performing surveillance on a POI and, once the local task is completed, it travels at maximum velocity to the next POI. The total completion time is then equal to the total required surveillance time plus the time needed to travel the minimum distance over all POIs. In this case, the UAV trajectory adopted by the Hover-and-Fly strategy will be the same as that of MinDist-DP (since the latter yields the minimum-distance trajectory over all POIs), but the total completion time will be larger for Hover-and-Fly since it does not allow the UAV to perform surveillance while flying.



(a) Total completion time.



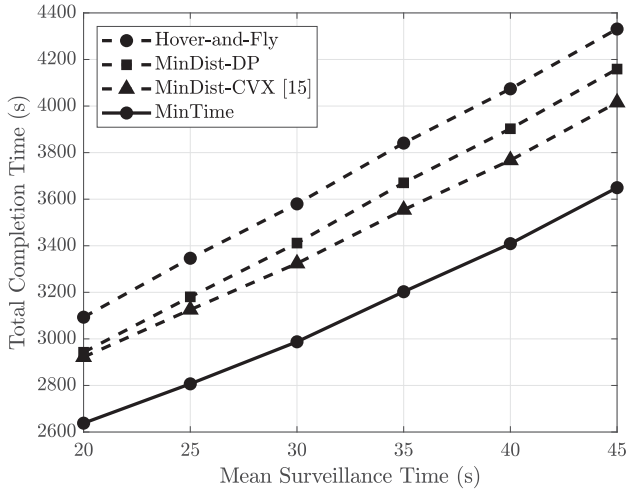
(b) Inter-POIs and Intra-POIs time.

Fig. 6. Total completion time versus the number of POIs.

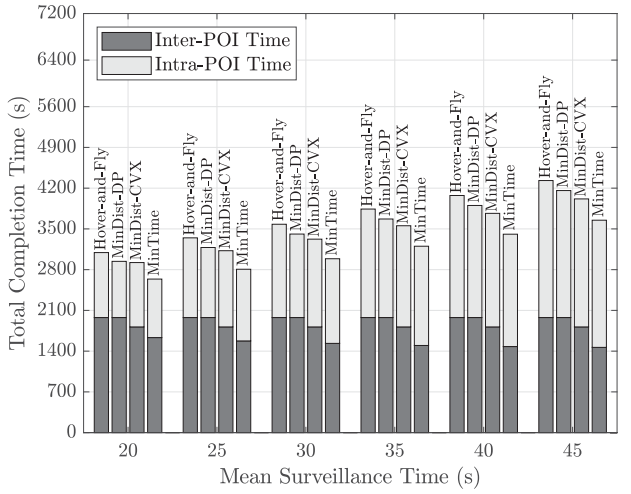
In Fig. 6a, we show the total completion time of the surveillance missions with respect to different numbers of POIs. First, we evaluate the performance of the proposed MinTime algorithm under different spacings between discrete points on the boundary. In particular, we consider cases where the discrete points on the boundaries are equally spaced by 20, 40, and 80 meters, respectively. These values correspond to average numbers of discrete points (denoted by $\bar{J} = E[J_i]$) that are approximately equal to 24, 12, and 6. We observe that little improvement can be obtained by increasing $\bar{J}$ beyond 24. Therefore, the remaining comparisons are done under this setting. Moreover, we can see that the proposed MinTime algorithm significantly outperforms the baseline MinDist-CVX, MinDist-DP, and Hover-and-Fly algorithms, especially as the number of POIs increases. The Hover-and-Fly strategy yields the largest total completion time since the UAV is not allowed to perform the surveillance task while flying. By taking into account the required surveillance time of the POIs, the MinTime algorithm tends to allow the UAV to travel farther on the boundary of the restricted regions so

TABLE 1
Computation Time for Varying Number of POIs

Methods	Number of POIs						
	30	35	40	45	50	55	60
MinDist-CVX	1.13	1.21	1.35	1.50	1.67	1.77	1.92
MinDist-DP (Hover-and-Fly)	1.35	1.81	2.20	2.78	3.19	3.80	4.53
MinTime	1.36	1.83	2.23	2.86	3.25	3.82	4.60



(a) Total completion time.



(b) Inter-POIs and Intra-POIs time.

Fig. 7. Total completion time versus surveillance time.

as to reduce the remaining distance to the next POI in the path. As a result, the total time that the UAV spends on the boundary of restricted regions (i.e., the intra-POI flight times) may increase, but the total completion time reduces due to significant reduction in the flight time between restricted regions (i.e., the inter-POI flight times). This effect can be observed in Fig. 6b where the Inter-POI and Intra-POI flight times that constitute the total completion time are shown explicitly for different values of the number of POIs.

In Table 1, we show the computation time for the experiments in Fig. 6a measured in seconds. For MinTime, we show only the case with $\bar{J} = 24$. We can see that MinDist-CVX yields the least amount of computation time whereas MinDist-DP and MinTime yield approximately the same

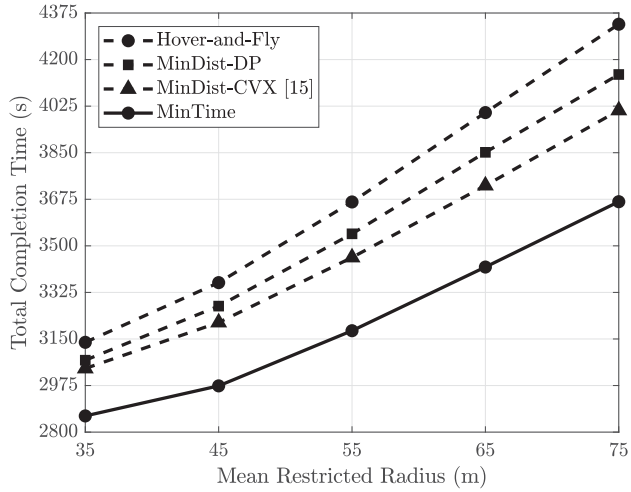
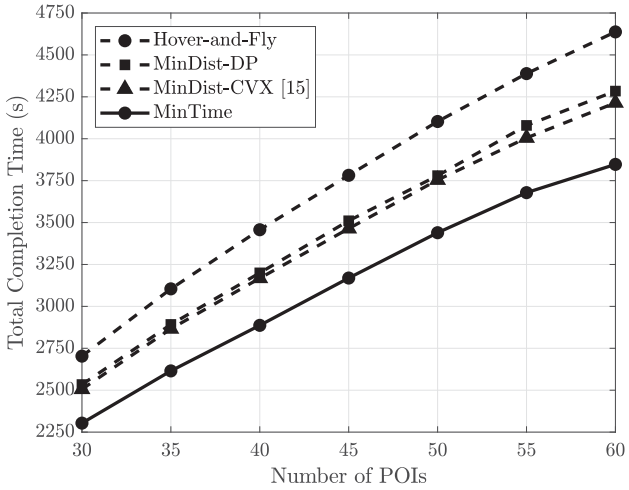


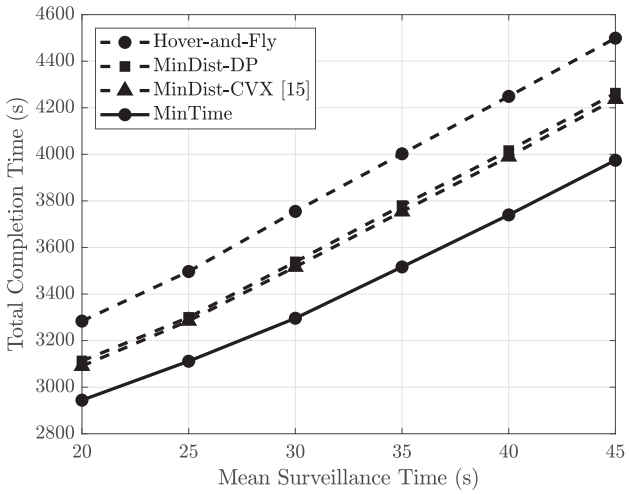
Fig. 8. Total completion time versus the mean radius of the restricted regions.

computation time since they are both based on DP. In the latter case, the computation time required for the case with 60 POIs is about 3 times that required for the case with 30 POIs. The increase is mainly due to the computation of the MST-based TSP solution and the costs between POIs. The DP process by itself increases only linearly with the number of POIs as mentioned in the complexity analysis in Section 4. Notice that both MinDist-DP and Hover-and-Fly algorithms yield the same computation time since their trajectories are obtained the same way. It is worthwhile to note that, for the case with 60 POIs, the computation time of the proposed MinTime algorithm is only 4.6 seconds, which is negligible compared to the total flight time of the UAV (which is approximated 3700 seconds). In contrast, the MinDist-CVX algorithm takes only 1.92 seconds, but the total flight time increases to over 4000 seconds. Hence, while a heuristic solution may be adopted to reduce the computation time, it may result in significant increase of the total flight time.

In Fig. 7a, we show the total completion time with respect to the mean of the required surveillance durations of the POIs. The surveillance durations are randomly chosen according to a uniform distribution within ± 15 seconds of the mean. We can see that the total completion time increases with the mean surveillance duration in all cases. However, the MinTime algorithm consistently outperforms the baseline MinDist-DP, MinDist-CVX, and Hover-and-Fly algorithms under different values of the mean surveillance duration. The gain is most significant when the mean surveillance duration is large since, in this case, the UAV is able to better utilize the surveillance time on the region boundary to travel to a location that reduces its distance to the next POI (and, thus, the inter-POI flight time). A comparison of the inter-POI and intra-POI flight times is also given in Fig. 7b. We can see that, even though the total completion time increases with the mean surveillance duration in all cases, the inter-POI time actually decreases for the proposed MinTime algorithm. This demonstrates the effectiveness of using the surveillance duration to travel towards a point on the region boundary that is closer to the next POI. This advantage is not enjoyed by the MinDist algorithms that do not take into account the surveillance duration in their designs.



(a) Total completion time versus the number of POIs.



(b) Total completion time versus the mean surveillance time.

Fig. 9. Total completion time under random polygonal restricted regions.

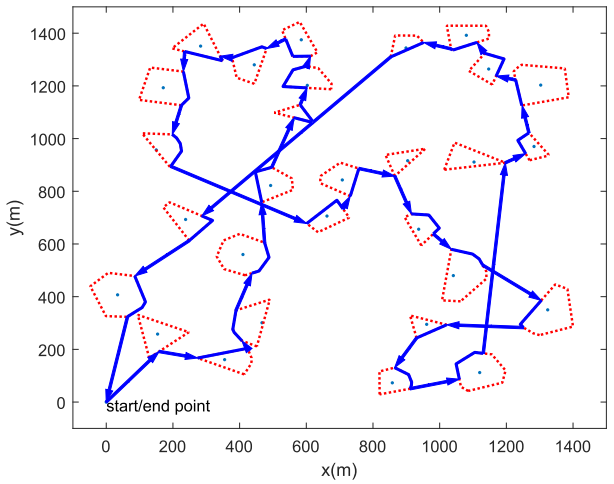


Fig. 10. A realization of the random polygon regions and the resulting UAV trajectory.

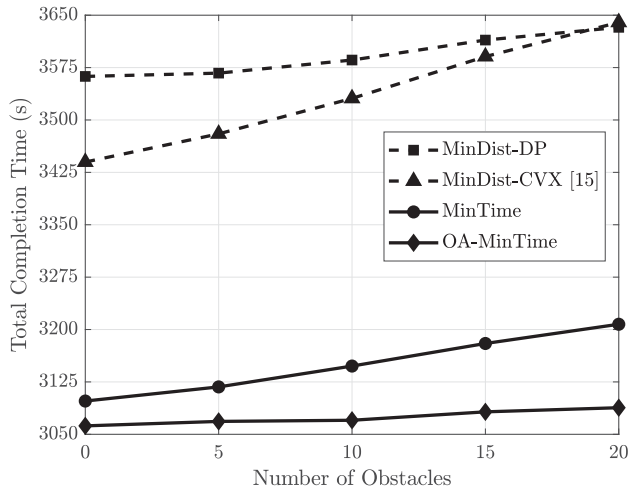


Fig. 11. Total completion time versus the number of obstacles for general case.

In Fig. 8, we show the total completion time with respect to the mean radius of the restricted regions. Notice that, in practice, a larger restricted region implies that more time is required to complete the surveillance mission. To take this effect into account, we set the mean surveillance duration as half of the average time required to travel around the restricted region once at flying speed $v_{\max}$. The surveillance durations of the POIs are i.i.d. uniformly distributed within $\pm 40\%$ seconds of the mean duration. We can see that the MinTime algorithm consistently outperforms the baseline MinDist-DP, MinDist-CVX, and Hover-and-Fly algorithms in all cases. In fact, the gain increases with the mean radius since the surveillance duration that can be exploited increases as well.

In Fig. 9, we repeat the experiments in Figs. 6a and 7a using restricted regions that take on randomly shaped polygons. This experiment demonstrates the applicability of our proposed algorithm to regions of arbitrary shape, as claimed in Section 4. The randomly shaped polygons are generated by choosing 3 to 8 points on the boundaries of the circular regions (i.e., the regions used in Figs. 6a and 7a) as the vertices of the polygons. In particular, in Figs. 9a and 9b, we show the total completion time versus the number of POIs and the mean surveillance time, respectively. We observe a similar trend as that in Figs. 6a and 7a. However, the difference between the baseline MinDist-DP and MinDist-CVX algorithms is smaller in this case since, with polygon regions, arrival and departure points are often chosen as vertices of the polygon, making the two solutions similar in many cases. The proposed MinTime algorithm still outperforms all the baseline algorithms in this case since it takes into consideration the required surveillance time in its optimization. A realization of the random polygon regions and the resulting UAV trajectory is given in Fig. 10.

In addition, we also examine the impact of flight obstacles on the proposed MinTime and OA-MinTime algorithms. Specifically, in Fig. 11, we show the total completion time with respect to the total number of obstacles in the area. The obstacles take on a rectangular shape with both the length and the width chosen independently according to a uniform distribution within $[40, 300]$ meters. We compare the proposed OA-MinTime algorithm with the basic MinTime, MinDist-DP and MinDist-CVX algorithms, which determine UAV trajectories without regard of the obstacles. The Hover-and-Fly strategy is omitted here since it always performs worse that MinDist-DP, as shown in the previous

TABLE 2
Computation Time for Varying Number of Obstacles

Methods	Number of Ostacles				
	0	5	10	15	20
MinDist-CVX	1.69	1.71	1.71	1.71	1.72
MinDist-DP (Hover-and-Fly)	2.53	2.91	3.28	3.61	3.96
MinTime	2.54	2.94	3.32	3.62	3.99
OA-MinTime	5.77	7.53	9.76	12.85	16.82

figures. We can see that the OA-MinTime algorithm improves upon the basic MinTime algorithm since the TSP solution is updated by taking into consideration the additional distance between POIs for obstacle avoidance, and the obstacles are avoided more efficiently in between POIs by modifying the DP to treat obstacles as effective POIs with zero surveillance durations. The advantage increases with the number of obstacles.

In Table 2, we show the computation time for the experiments in Fig. 11. Notice that the computation time of MinDist-CVX is similar to the case without obstacles (c.f., Table 1 with 50 POIs) since the obstacles are not taken into consideration in the trajectory design of MinDist-CVX. Hence, we can see that MinDist-CVX again yields the least amount of computation time whereas MinDist-DP and MinTime yield approximately the same computation time. However, the required computation time of the OA-MinTime algorithm is higher than that of the original MinTime algorithm since, in the former case, the obstacles are treated as effective POIs and the MinTime algorithm was used to obtain the initial solution. However, the overall computation time is still within a few seconds, which is negligible compared to the time required to complete the surveillance tasks.

In Fig. 12, we demonstrate the importance of updating the TSP path in the presence of large obstacles. We consider a special case where the large obstacles are assumed to take on long rectangular shapes, similar to that in Fig. 5. The obstacles have length that is uniformly distributed between 600 and 700 meters and width equal to 20 meters, and are placed either vertically or horizontally with equal probability. The radius of the restricted regions are fixed as 50 meters. We can see, in Fig. 12, that the total completion time increases with the number of obstacles in all cases. However, the increase is significantly less for the proposed OA-MinTime algorithm (which includes an obstacle-aware update of the TSP path). This is because the other baseline algorithms employ a TSP solution that does not take into consideration the additional distance required for obstacle-avoidance on the inter-POI paths and, thus, may choose consecutive POIs that are on opposite sides of the obstacle. This is evident from the comparison with the OA-MinTime algorithm that employs the basic TSP solution without the obstacle-aware update. This experiment shows that, in the presence of certain types of obstacles, the updated TSP is essential to obtain paths that go around the obstacles instead of back and forth across the obstacles. In fact, by doing so, the increase in the total completion time as the number of obstacles increases becomes negligible compared to other schemes.

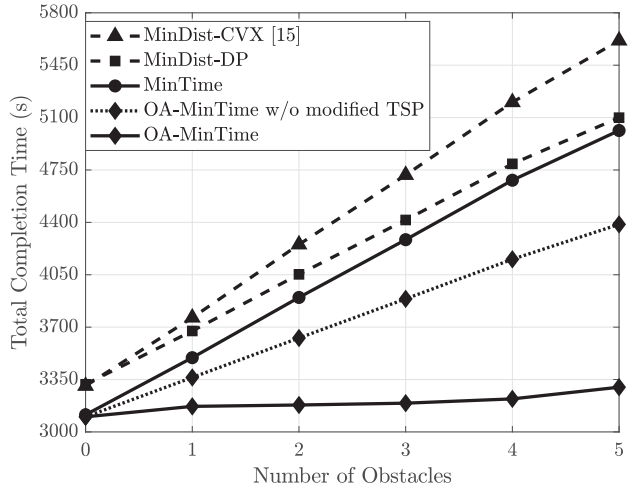


Fig. 12. Total completion time versus the number of obstacles for special case.

7 Conclusion

In this paper, we considered the trajectory optimization problem for a UAV-enabled surveillance mission, where the UAV is dispatched to collect information from multiple POIs inside restricted regions. The UAV surveillance mission is completed by visiting the boundary of every restricted region for a minimum required duration. We proposed a trajectory optimization algorithm that aims to minimize the total completion time subject to surveillance duration constraints at each POI. The proposed MinTime algorithm first determines the visiting order using an MST-based TSP solution, and then determines the arriving and departing points on the boundary of the restricted regions using DP. In the presence of obstacles, we further proposed the OA-MinTime algorithm that is able to avoid obstacles in a more time-efficient manner by treating them as additional restricted regions that must be visited, but with zero surveillance duration. In addition, we also proposed a modified TSP solution that takes into account the additional distance required to circumvent obstacles on the inter-POI paths. This was shown to be essential in special cases where POIs are separated by large obstacles. Simulation results showed that the MinTime algorithm is able to reduce the total completion time by approximately 10% compared to the baseline MinDist-CVX and MinDist-DP algorithms. In the presence of obstacles, the OA-MinTime algorithm further improves upon the basic MinTime algorithm by about 4%. In the presence of a special type of large obstacles, the OA-MinTime algorithm can further improve upon the basic MinTime algorithm by approximately 30%.

References

[1] A. Fahim and Y. Gadallah, "An optimized LTE-based technique for drone base station dynamic 3D placement and resource allocation in delay-sensitive M2M networks," *IEEE Trans. Mobile Comput.*, to be published, doi: 10.1109/TMC.2021.3089329.
[2] E. Arribas, V. Mancuso, and V. Cholvi, "Coverage optimization with a dynamic network of drone relays," *IEEE Trans. Mobile Comput.*, vol. 19, no. 10, pp. 2278–2298, Oct. 2020.
[3] M. Mozaffari, W. Saad, M. Bennis, Y.-H. Nam, and M. Debbah, "A tutorial on UAVs for wireless networks: Applications, challenges, and open problems," *IEEE Commun. Surv. Tut.*, vol. 21, no. 3, pp. 2334–2360, Jul.–Sep. 2019.

[4] C. You and R. Zhang, "3D trajectory optimization in Rician fading for UAV-enabled data harvesting," *IEEE Trans. Wireless Commun.*, vol. 18, no. 6, pp. 3192–3207, Jun. 2019.
[5] X. Chen, Z. Feng, Z. Wei, F. Gao, and X. Yuan, "Performance of joint sensing-communication cooperative sensing UAV network," *IEEE Trans. Veh. Technol*, vol. 69, no. 12, pp. 15 545–15 556, Dec. 2020.
[6] N. H. Motlagh, M. Bagaa, and T. Taleb, "UAV-based IoT platform: A crowd surveillance use case," *IEEE Commun. Mag.*, vol. 55, no. 2, pp. 128–134, Feb. 2017.
[7] A. Brown and D. Anderson, "Trajectory optimization for high-altitude long-endurance UAV maritime radar surveillance," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 56, no. 3, pp. 2406–2421, 2020.
[8] D. Han, W. Chen, and J. Liu, "Energy-efficient UAV communications under stochastic trajectory: A Markov decision process approach," *IEEE Trans. Green Commun. Netw.*, vol. 5, no. 1, pp. 106–118, Mar. 2021.
[9] S. Hosseinalipour, A. Rahmati, D. Y. Eun, and H. Dai, "Energy-aware stochastic UAV-assisted surveillance," *IEEE Trans. Wireless Commun.*, vol. 20, no. 5, pp. 2820–2837, May 2021.
[10] H. Huang and A. V. Savkin, "Navigating UAVs for optimal monitoring of groups of moving pedestrians or vehicles," *IEEE Trans. Veh. Technol*, vol. 70, no. 4, pp. 3891–3896, Apr. 2021.
[11] H. Teng, I. Ahmad, A. Msm, and K. Chang, "3D optimal surveillance trajectory planning for multiple UAVs by using particle swarm optimization with surveillance area priority," *IEEE Access*, vol. 8, pp. 86316–86327, 2020.
[12] D. Yang, Q. Wu, Y. Zeng, and R. Zhang, "Energy tradeoff in ground-to-UAV communication via trajectory design," *IEEE Trans. Veh. Technol*, vol. 67, no. 7, pp. 6721–6726, Jul. 2018.
[13] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Mobile unmanned aerial vehicles (UAVs) for energy-efficient internet of things communications," *IEEE Trans. Wireless Commun.*, vol. 16, no. 11, pp. 7574–7589, Nov. 2017.
[14] J. Gong, T.-H. Chang, C. Shen, and X. Chen, "Flight time minimization of UAV for data collection over wireless sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 9, pp. 1942–1954, Sep. 2018.
[15] J. Zhang, Y. Zeng, and R. Zhang, "UAV-enabled radio access network: Multi-mode communication and trajectory design," *IEEE Trans. Signal Process.*, vol. 66, no. 20, pp. 5269–5284, Oct. 2018.
[16] M. Chen, W. Liang, and J. Li, "Energy-efficient data collection maximization for UAV-assisted wireless sensor networks," in *Proc. IEEE Wireless Commun. Netw. Conf.*, 2021, pp. 1–7.
[17] Y. Li et al., "Data collection maximization in IoT-sensor networks via an energy-constrained UAV," *IEEE Trans. Mobile Comput.*, to be published, doi: 10.1109/TMC.2021.3084972.
[18] C. Shen, T.-H. Chang, J. Gong, Y. Zeng, and R. Zhang, "Multi-UAV interference coordination via joint trajectory and power control," *IEEE Trans. Signal Process.*, vol. 68, pp. 843–858, 2020.
[19] D. Xu, Y. Sun, D. W. K. Ng, and R. Schober, "Multiuser MISO UAV communications in uncertain environments with no-fly zones: Robust trajectory and resource allocation design," *IEEE Trans. Commun.*, vol. 68, no. 5, pp. 3153–3172, May 2020.
[20] Y. Gao, H. Tang, B. Li, and X. Yuan, "Joint trajectory and power design for UAV-enabled secure communications with no-fly zone constraints," *IEEE Access*, vol. 7, pp. 44459–44470, 2019.
[21] D. Ebrahimi, S. Sharafeddine, P.-H. Ho, and C. Assi, "UAV-aided projection-based compressive data gathering in wireless sensor networks," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1893–1905, Apr. 2019.
[22] C. Lin, G. Han, X. Qi, J. Du, T. Xu, and M. Martínez-García, "Energy-optimal data collection for unmanned aerial vehicle-aided industrial wireless sensor network-based agricultural monitoring system: A clustering compressed sampling approach," *IEEE Trans. Ind. Informat.*, vol. 17, no. 6, pp. 4411–4420, Jun. 2021.
[23] L. Shen, N. Wang, Z. Zhu, Y. Fan, X. Ji, and X. Mu, "UAV-enabled data collection for mMTC networks: AEM modeling and energy-efficient trajectory design," in *Proc. IEEE Int. Conf. Commun.*, 2020, pp. 1–6.
[24] J. Baek, S. I. Han, and Y. Han, "Energy-efficient UAV routing for wireless sensor networks," *IEEE Trans. Veh. Technol*, vol. 69, no. 2, pp. 1741–1750, Feb. 2020.
[25] S.-H. Yeh, Y.-S. Wang, T. D. P. Perera, Y.-W. Peter Hong, and D. N. K. Jayakody, "UAV trajectory optimization for data-gathering from backscattering sensor networks," in *Proc. IEEE Int. Conf. Commun.*, 2020, pp. 1–6.
[26] C. Zhan, Y. Zeng, and R. Zhang, "Trajectory design for distributed estimation in UAV-enabled wireless sensor network," *IEEE Trans. Veh. Technol*, vol. 67, no. 10, pp. 10155–10159, Oct. 2018.
[27] S. Lhazmir, O. A. Oualhaj, A. Kobbane, E. M. Amhoud, and J. Ben-Othman, "UAV for wireless power transfer in IoT networks: A GMDP approach," in *Proc. IEEE Int. Conf. Commun.*, 2020, pp. 1–6.
[28] Y.-C. Kuo, J.-H. Chiu, J.-P. Sheu, and Y.-W. P. Hong, "UAV deployment and IoT device association for energy-efficient data-gathering in fixed-wing multi-UAV networks," *IEEE Trans. Green Commun. Netw.*, vol. 5, no. 4, pp. 1934–1946, Dec. 2021.
[29] J. Li et al., "Joint optimization on trajectory, altitude, velocity, and link scheduling for minimum mission time in UAV-aided data collection," *IEEE Internet Things J.*, vol. 7, no. 2, pp. 1464–1475, Feb. 2020.
[30] O. M. Bushnaq, A. Celik, H. Elsawy, M.-S. Alouini, and T. Y. Al-Naffouri, "Aeronautical data aggregation and field estimation in IoT networks: Hovering and traveling time dilemma of UAVs," *IEEE Trans. Wireless Commun.*, vol. 18, no. 10, pp. 4620–4635, Oct. 2019.
[31] C. Luo, M. N. Satpute, D. Li, Y. Wang, W. Chen, and W. Wu, "Fine-grained trajectory optimization of multiple UAVs for efficient data gathering from WSNs," *IEEE/ACM Trans. Netw.*, vol. 29, no. 1, pp. 162–175, Jan. 2021.
[32] S. Alfattani, W. Jaafar, H. Yanikomeroglu, and A. Yongacoglu, "Multi-UAV data collection framework for wireless sensor networks," in *Proc. IEEE Glob. Commun. Conf.*, 2019, pp. 1–6.
[33] Y. Wang et al., "Multi-UAV collaborative data collection for IoT devices powered by battery," in *Proc. IEEE Wirel. Commun. Netw. Conf.*, 2020, pp. 1–6.
[34] J. Liu, P. Tong, X. Wang, B. Bai, and H. Dai, "UAV-aided data collection for information freshness in wireless sensor networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 4, pp. 2368–2382, Apr. 2021.
[35] H. Hu, K. Xiong, G. Qu, Q. Ni, P. Fan, and K. B. Letaief, "AoI-minimal trajectory planning and data collection in UAV-assisted wireless powered IoT networks," *IEEE Internet Things J.*, vol. 8, no. 2, pp. 1211–1223, Feb. 2021.
[36] Z. Jia, X. Qin, Z. Wang, and B. Liu, "Age-based path planning and data acquisition in UAV-assisted IoT networks," in *Proc. IEEE Int. Conf. Commun. Workshops*, 2019, pp. 1–6.
[37] J.-S. Lin, H.-T. Chiu, and R.-H. Gau, "Decentralized planning-assisted deep reinforcement learning for collision and obstacle avoidance in UAV networks," in *Proc. IEEE 93rd Veh. Technol. Conf.*, 2021, pp. 1–7.
[38] H. Bayerlein, M. Theile, M. Caccamo, and D. Gesbert, "UAV path planning for wireless data harvesting: A deep reinforcement learning approach," in *Proc. IEEE Glob. Commun. Conf.*, 2020, pp. 1–6.
[39] O. Bouhamed, H. Ghazzai, H. Besbes, and Y. Massoud, "A UAV-assisted data collection for wireless sensor networks: Autonomous navigation and scheduling," *IEEE Access*, vol. 8, pp. 110 446–110 460, 2020.
[40] C. Wang, J. Wang, Y. Shen, and X. Zhang, "Autonomous navigation of UAVs in large-scale complex environments: A deep reinforcement learning approach," *IEEE Trans. Veh. Technol*, vol. 68, no. 3, pp. 2124–2136, Mar. 2019.
[41] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd ed. Cambridge, MA, USA: MIT Press, 2009.
[42] G. Laporte, "The traveling salesman problem: An overview of exact and approximate algorithms," *Eur. J. Oper. Res.*, vol. 59, no. 2, pp. 231–247, 1992.

Hsiang-Chun Tsai received the BS degree in computer science from the National University of Kaohsiung, Taiwan, in 2019 and the MS degree in computer science from National Tsing Hua University, Taiwan, in 2021. His research interests include UAV communications and wireless sensor networks. He is currently an engineer at Synopsys Taiwan Company, Ltd.

Y.-W. Peter Hong (Senior Member, IEEE) received the BS degree in electrical engineering from National Taiwan University, Taipei, Taiwan, in 1999, and the PhD degree in electrical engineering from Cornell University, Ithaca, NY, USA, in 2005. He joined the Institute of Communications Engineering and the Department of Electrical Engineering with National Tsing Hua University (NTHU), Hsinchu, Taiwan, in 2005, where he is a full professor. His research interests include UAV communications, distributed signal processing for IoT and sensor networks, machine learning for wireless communications, and physical layer security. He has received the IEEE ComSoc Asia-Pacific Outstanding Young Researcher Award, in 2010, the Y. Z. Hsu Scientific Paper Award, in 2011, the National Science Council Wu Ta-You Memorial Award, in 2011, the Chinese Institute of Electrical Engineering (CIEE) Outstanding Young Electrical Engineer Award, in 2012, and the Ministry of Science and Technology Outstanding Research Award, in 2019. He currently serves as an Area Editor of *IEEE Transactions on Signal Processing*. He is also a Distinguished Lecturer of IEEE Communications Society (2022-2023) and the Vice Director of the IEEE ComSoc Asia-Pacific Board. In the past, he also served as associate editor of *IEEE Transactions on Signal Processing*, *IEEE Transactions on Information Forensics and Security*, and Editor of *IEEE Transactions on Communications*. He was also chair of the IEEE ComSoc Taipei Chapter, in 2017-2018, and co-chair of the Technical Affairs Committee, the Information Services Committee, and the Chapter Coordination Committee of the IEEE ComSoc Asia-Pacific Board, in 2014-2015, 2016-2019, and 2020-2021, respectively.

Jang-Ping Sheu (Fellow, IEEE) received the BS degree in computer science from Tamkang University, Taiwan, in 1981, and the MS and PhD degrees in computer science from National Tsing Hua University, Taiwan, in 1983 and 1987, respectively. He is currently a chair professor of the Department of Computer Science and director of the Joint Research Center of Delta-NTHU, National Tsing Hua University. He was an associate dean of the College of Electrical and Computer Science from 2016 to 2017, National Tsing Hua University. He was a director of the Computer and Communication Research Center from, 2009 to 2015, National Tsing Hua University. He was a director of Computer Center, National Central University, from 2003 to 2006. He was a director of the Department of Computer Science and Information Engineering, National Central University, from 1997 to 1999. His current research interests include wireless communications, mobile computing, Internet of Things, and UAV-assisted communication systems. He was an associate editor of the *IEEE Transactions on Parallel and Distributed Systems* and the *International Journal of Sensor Networks*. He is an Advisory Board Member of the *International Journal of Ad Hoc and Ubiquitous Computing* and the *International Journal of Vehicle Information and Communication Systems*. He received the Distinguished Research awards of the National Science Council of the Republic of China, in 1993-1994, 1995-1996, and 1997-1998. He received the Distinguished Engineering Professor Award from the Chinese Institute of Engineers in 2003. He received the K. -T. Li Research Breakthrough Award from the Institute of Information and Computing Machinery (IICM), in 2007. He received the Y. Z. Hsu Scientific Chair Professor Award and Pan Wen Yuan Outstanding Research Award, in 2009 and 2014, respectively. He received the Academic Award in Engineering from the Ministry of Education, in 2016. He received the Medal of Honor in Information Sciences from the IICM, in 2017. He received the TECO Award and the Chinese Institute of Electrical Engineering (CIEE) Fellow, in 2019 and 2021. Dr. Sheu is a Phi Tau Phi Society member.