

LOMORO: Long-term Monitoring of Dynamic Targets with Minimum Robotic Fleet under Resource Constraints

Mingke Lu, Shuaikang Wang and Meng Guo

Abstract—Long-term monitoring of numerous dynamic targets can be tedious for a human operator and infeasible for a single robot, e.g., to monitor wild flocks, detect intruders, search and rescue. Fleets of autonomous robots can be effective by acting collaboratively and concurrently. However, the online coordination is challenging due to the unknown behaviors of the targets and the limited perception of each robot. Existing work often deploys all robots available without minimizing the fleet size, or neglects the constraints on their resources such as battery and memory. This work proposes an online coordination scheme called LOMORO for collaborative target monitoring, path routing and resource charging. It includes three core components: (I) the modeling of multi-robot task assignment problem under the constraints on resources and monitoring intervals; (II) the resource-aware task coordination algorithm iterates between the high-level assignment of dynamic targets and the low-level multi-objective routing via the Martin's algorithm; (III) the online adaptation algorithm in case of unpredictable target behaviors and robot failures. It ensures the explicitly upper-bounded monitoring intervals for all targets and the lower-bounded resource levels for all robots, while minimizing the average number of active robots. The proposed methods are validated extensively via large-scale simulations against several baselines, under different road networks, robot velocities, charging rates and monitoring intervals.

I. INTRODUCTION

Mobile robots such as unmanned aerial vehicles (UAVs) and unmanned ground vehicles (UGVs) are becoming more capable of autonomous inspection and navigation. Via wireless communication and collaboration, a fleet of such robots can be deployed to monitor large areas that are otherwise too demanding for human operators, e.g., to explore unknown territory [1], and track moving targets [2], [3]. Particularly, active monitoring of unknown dynamic targets has attracted significant attention, see [4], [5], [6], [7]. It incorporates several challenging aspects of multi-robot coordination: (I) the behavior of each target regarding its velocity and future path is uncertain, meaning that the assignment of targets to robots should be adaptive; (II) to ensure the accuracy of monitoring, a minimum monitoring interval for each target is required. Moreover, for long-term (possibly indefinite) mission, it is inevitable that the robots are subject to resource constraints such as battery and memory. In this case, the planning of monitoring tasks and charging activities are heavily dependent and should be planned as a whole. Existing methods [2], [3], [4], [8], [9], [10], [11], [12] mostly consider the short-term monitoring task of one or several

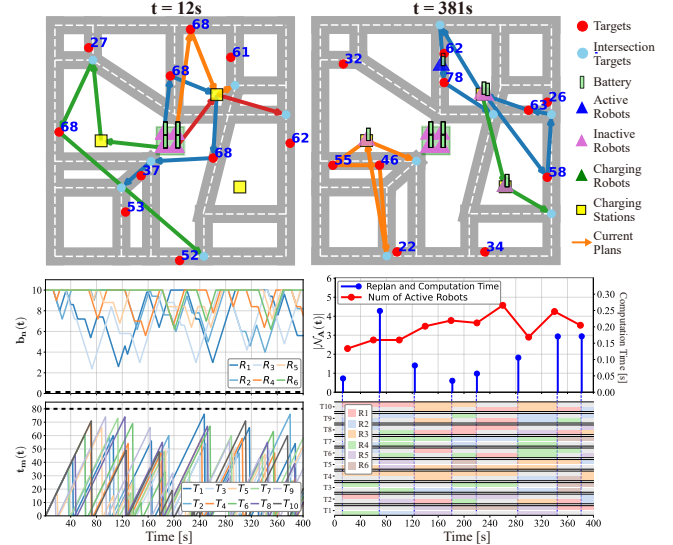


Fig. 1. Illustration of the considered scenario. **Top-Left and Top-Right:** 3–4 UAVs are actively monitoring 10 targets (in red) within a road network, with their static ($t=12s$) and online ($t=381s$) plans. **Middle Left:** Batteries of 6 robots during the online execution of 400 s. **Bottom-Left:** Intervals from the last monitoring of 10 targets. **Middle-Right:** Average number of active robots, the time when replans take place, and their computation time. **Bottom-Right:** The robots responsible for each target during any consecutive replans. Each target has 2 rows, with the lower representing the target node and the upper the intersection target node respectively.

targets with known behaviors, via a fixed fleet size of robots without resource constraints. It remains an open problem to coordinate online the minimum fleet of robots, under the strict constraints of minimum monitoring intervals over targets and the minimum resources over robots.

A. Related Work

1) *Active Information Acquisition:* Active information acquisition for sensing robots, originating in [4], [9], focuses on optimizing robot motion to maximize information gain. Early approaches employ offline search algorithms that explore the joint state-information space through forward value iteration and its reduced variant, leveraging separation principles under linear observation models. These principles have been extended to multi-robot systems: centralized, non-myopic solutions using sampling-based methods are proposed in [3], [8], while decentralized, myopic strategies are investigated in [5], [6], [7], [13]. Learning-based techniques, such as those in [2], train distributed policies to mimic optimal planners. However, existing methods predominantly assume fixed robot team sizes and targets with known dynamics or control inputs. A key unresolved challenge lies in dynamically adjusting the robot team size to monitor targets

exhibiting unknown behaviors, including velocities and road network navigation paths.

2) *Multi-robot Task Assignment*: In contrast, multi-target monitoring has also been framed as a sequential high-level assignment problem, with robots assigned to targets to minimize uncertainty. While [14] introduces a distributed Hungarian method, its scope is restricted to one-to-one assignments. Several works, including [10], [11], [12], rephrase this as a simultaneous action and target assignment problem, proposing distributed approximation algorithms such as linear programs. These methods, however, typically assume known target trajectories and rely on synchronized robot motions with predefined primitives. The complementary work of [15] maximizes the number of tracked targets using a fixed robot team via a 2-approximation greedy approach. Recent studies like [16], [17] investigate robust assignments against communication or sensing attacks but retain assumptions of synchronized motions with finite primitives. Departing from these frameworks, this work determines online the flexible assignment of robots to targets according to actual observations that can not be determined offline.

3) *Planning under Resource Constraints*: Recent advances in robot motion and task planning under resource constraints emphasize optimization-based frameworks that balance computational efficiency with energy and memory limitations. State-of-the-art approaches integrate multi-objective optimization to handle battery constraints, often leveraging dynamic programming [18] or model predictive control [19] to allocate energy budgets while ensuring task completion. Memory-aware planning algorithms [20] or hierarchical task decomposition [21], reduce computational overhead by pruning redundant states or compressing environment representations. For distributed systems, decentralized strategies in [22] use reinforcement learning to optimize local decisions under shared resource limits. Hybrid methods combining offline pre-planning with online adaptation, such as the anytime algorithms in [23], dynamically adjust plans based on real-time resource consumption. Despite progress, challenges persist in scaling these methods to highly dynamic large-scale environments and flexible fleet size.

B. Our Method

This work addresses the *long-term* monitoring task of numerous dynamic targets within a road network via a fleet of aerial robots, where each target has a strict (potentially different) monitoring interval. More importantly, the robots have limited resources (such as battery and memory) that are consumed over time and should be recharged often. The targets follow a constant-velocity model, but with unknown velocity on each road, and unknown path within the road network. The proposed method consists of three components: (I) an search-based assignment algorithm is designed that searches through a partial sequence of robots, in which the subset of targets are assigned in an incremental way. It ranks the feasible assignments based on a multi-objective measure; (II) a maximum-allowed Martin's algorithm (MAM) is proposed to determine *simultaneously* the optimal subset and

sequence of targets to monitor, and the optimal charging station to charge, via an efficient incremental label-setting procedure. (III) an online adaptation scheme is proposed to monitor the feasibility of the local plan of each vehicle online. In case of violation of the above constraints, it triggers the first component to find first an alternative assignment that is feasible (by recruiting additional robots if needed), then improves the quality as more planing time is permitted. It is proven that the constraints of resources and monitoring intervals are fulfilled at all time, while the average number of robots that actively monitors the targets is minimized. Extensions such as free membership of targets and charging stations that are dynamically moving are also demonstrated. Extensive simulations are performed over large-scale fleets and targets over complex scenes.

Main contribution of this work is three-fold: (I) the novel formulation of the minimum-fleet monitoring problem of unknown dynamic targets within road networks, under strict constraints of monitoring interval and resources; (II) the hierarchical solution that adapts the fleet size and vehicle trajectories online, according to real-time observations of the target behavior; (III) the scalable solution that allows the deployment of a few UAVs to monitor a large number of dynamic targets.

II. PROBLEM DESCRIPTION

A. System Description

Consider a workspace $\mathcal{W} \subset \mathbb{R}^2$ within which the robots and targets coexist. We define the environment as $\mathcal{P} \triangleq (\mathcal{R}, \mathcal{S})$, where $\mathcal{R} \subset \mathcal{W}$ denotes the road network and $\mathcal{S} = \{1, 2, \dots, S\}$ denotes the set of charging stations. The road network is defined as $\mathcal{R} \triangleq (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} \subset \mathcal{W}$ is the set of intersections and $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ is a set of straight roads connecting these hubs. For each charging station $s \in \mathcal{S}$, its position is $\mathbf{z}_s \in \mathcal{W}$ and the charging capacity per unit time is $\beta_s > 0$. Moreover, there is a fleet of robots $\mathcal{N} \triangleq \{1, \dots, N\}$, which is divided into two sets, i.e., $\mathcal{N} \triangleq \mathcal{N}_A(t) \cup \mathcal{N}_I(t)$, where $\mathcal{N}_A(t)$ denotes the active robots and $\mathcal{N}_I(t)$ denotes the inactive robots. The state of each robot $n \in \mathcal{N}$ is given by its position $\mathbf{x}_n(t) \in \mathcal{W}$, velocity $\mathbf{v}_n(t) \in \mathbb{R}^2$ and battery $b_n(t) > 0$. Its rate of battery consumption is defined as $\gamma_n(t) \triangleq \gamma(\|\mathbf{v}_n(t)\|), t \geq 0$, where $\gamma : \mathbb{R} \rightarrow \mathbb{R}$ is a monotonically increasing function, and $\|\cdot\|$ denotes the L_2 norm. The maximum velocity is $v_n^{\max} > 0$ and the battery capacity is $b_n^{\max} > 0$. We define the control space of the robots as $\mathcal{U}$, and the robot follows the nonlinear dynamics:

$$[\mathbf{x}_n(t+1), \mathbf{v}_n(t+1)]^T \triangleq f(\mathbf{x}_n(t), \mathbf{v}_n(t), \mathbf{u}_n(t)), \quad (1)$$

where $f(\cdot)$ is the dynamic model and $\mathbf{u}_n(t) \in \mathcal{U}$ is the control input of the robot n at time $t \geq 0$. In addition, the sensor range of robot n is $R_n > 0$, within which it can make observations and detect targets.

Given the set of targets $\mathcal{M} \triangleq \{1, \dots, M\}$, each target $m \in \mathcal{M}$ moves on roads in $\mathcal{R}$, following the constant-velocity model [4] but with different velocities, i.e., $\mathbf{y}_m(t+1) \triangleq \mathbf{y}_m(t) + h \cdot \mathbf{v}_m$, where $h > 0$ is the sampling time, $\mathbf{y}_m(t) \in \mathcal{W}$ is the target state at $t \geq 0$, and $\mathbf{v}_m \in \mathbb{R}^2$ is the constant

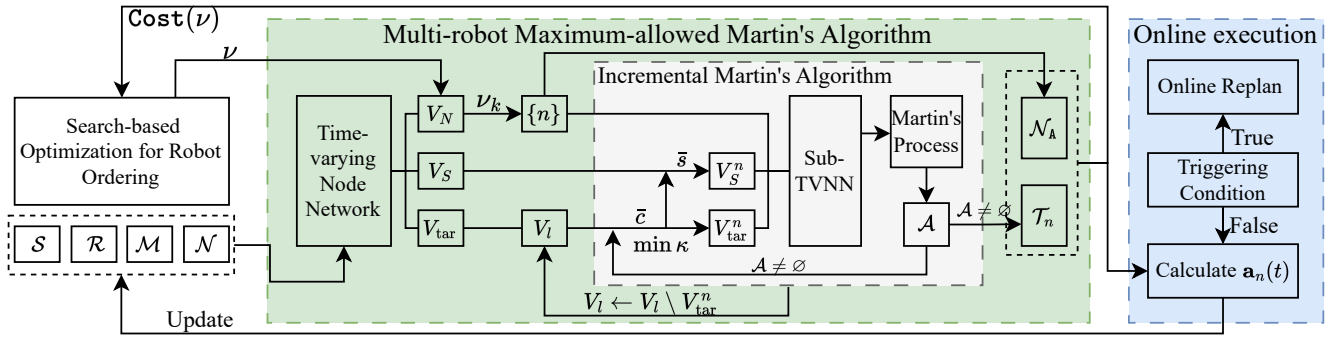


Fig. 2. Illustration of the proposed framework, which consists of three parts: Search-based optimization for robot ordering, multi-robot maximum-allowed Martin's algorithm (MAM) and online execution. The MAM consists of the time-varying node network (TVNN) and the incremental Martin's algorithm (IMA). The IMA mainly consists of incremental sub-TVNN and the Martin's Process.

velocity of target m . We assume that both the number of targets M and their initial states are known. Namely, the target may randomly switch to a different road at each hub point $e \in \mathcal{E}$, i.e., following an unknown path.

B. Local Plans and Constraints

Each robot $n \in \mathcal{N}_A(t)$ has a local plan $\mathcal{T}_n$, which is a sequence of targets and charging stations assigned to n . For each robot $n \in \mathcal{N}_A(t)$, we define $\mathbf{a}_n(t)$ as its action sequence, where $\mathbf{a}_n(t) \in \{a_{tr}(v), a_{mo}(m), a_{ch}(s)\}, v \in \mathcal{M} \cup \mathcal{S}, m \in \mathcal{M}, s \in \mathcal{S}, v, m, s \in \mathcal{T}_n$, and $T_d(\cdot)$ is the duration of the action. If $\mathbf{a}_n(t) = a_{tr}(v)$, robot n is during navigation to a target or at a charging station. If $\mathbf{a}_n(t) = a_{mo}(m)$, robot n is monitoring target m , which satisfies:

$$d(\mathbf{y}_m(t), \mathbf{x}_n(t)) \leq R_n, T_d(a_{mo}(m)) \geq T_0, \quad (2)$$

where $d(\cdot)$ is the distance function, and T_0 is the required minimum interval. If $\mathbf{a}_n(t) = a_{ch}(s)$, robot n charges at charging station s , i.e.,

$$b_n(t) \triangleq \begin{cases} \min\{b_n^{\max}, b_n(t-1) + \beta_s\}, & \text{if } \mathbf{a}_n(t) = a_{ch}(s); \\ \max\{b_n(t-1) - \gamma_n(t-1), 0\}, & \text{otherwise,} \end{cases}$$

where the battery should satisfy the constraint:

$$0 \leq b_n(t) \leq b_n^{\max}, \forall n \in \mathcal{N}, \forall t \geq 0, \quad (3)$$

which holds for all robots. Moreover, let $\mathcal{T}_n^{\text{tar}} \triangleq \mathcal{M} \cap \mathcal{T}_n$. There is a maximum number of targets any robot n can track simultaneously, due to limited computation time, i.e., $|\mathcal{T}_n^{\text{tar}}| \leq C_n$. The monitoring interval is fixed as χ_m for each target $m \in \mathcal{M}$. The time from the last monitoring to the current time is denoted by $\chi_m(t)$ that:

$$0 \leq \chi_m(t) \leq \chi_m, \forall m \in \mathcal{M}, \forall t \geq 0, \quad (4)$$

where $\chi_m(t)$ is set to zero if m is monitored, i.e., if there exists $n \in \mathcal{N}_A(t)$, such that $\mathbf{a}_n(t) = a_{mo}(m)$, $\chi_m(t)$ is set to zero. More importantly, each target should be monitored at intersections, i.e., if there exists t_0 , such that $\chi_m^{\text{inter}}(t_0) = 0, m \in \mathcal{M}$, then there exists $n \in \mathcal{N}_A(t)$, $\mathbf{a}_n(t_0) = a_{mo}(m)$, where χ_m^{inter} is the time of m to the next intersection. Last but not least, the rate of battery consumption $\gamma_n(\cdot)$ is assumed to satisfy:

$$\gamma_n(t) = \gamma(\|\mathbf{v}_n(t)\|) \leq \frac{b_n^{\max} \|\mathbf{v}_n(t)\|}{2 \cdot \sup_{\mathbf{w} \in \mathcal{W}} d_S(\mathbf{w})}, \quad (5)$$

where $d_S(\mathbf{p}) \triangleq \min_{s \in \mathcal{S}} d(\mathbf{p}, \mathbf{z}_s)$. This condition ensures that any robot can visit any position within $\mathcal{W}$ and return to one charging station with its maximum level of battery.

C. Problem Statement

The considered problem can be stated as a long-term constrained optimization problem, i.e.,

$$\min_{\mathcal{N}_A(t), \{\mathbf{u}_n(t), \mathbf{a}_n(t)\}} \lim_{T \rightarrow \infty} \frac{1}{T} \sum_{t=0}^T |\mathcal{N}_A(t)|, \quad (6)$$

where the decision variables are the set of active robots and their local plans; the objective is to minimize the average number of active robots; and the constraints are (1)-(5).

III. PROPOSED SOLUTION

The proposed solution is mainly composed of two layers: the search-based optimization for robot ordering, and the multi-robot maximum-allowed Martin's algorithm. In addition, the online execution, adaptations, and generalizations of the method are discussed. For convenience, the start time is set as $t = 0$, and the ending time of the planning horizon is $\max_{m \in \mathcal{M}} T_m$.

A. Search-based Optimization for Robot Ordering

As an essential input to the following component, the relative ordering among the robots is determined via a search-based scheme. More specifically, a search tree is constructed via iterative node selection and expansion. Each node is a partial ordering of the robots i.e., $\nu = n_1 n_2 \dots n_K$, where $\nu_k \in \mathcal{N}$ is a robot ID. The cost of a node is defined as a vector:

$$\text{Cost}(\nu) \triangleq \left(|\mathcal{N}_A(t)|, \max_{n \in \mathcal{N}_A} T_{\tau_n}, \sum_{n \in \mathcal{N}_A} \Delta b_n \right),$$

consisting of the number of active robots, the longest execution time of all active robots, and the total battery consumption. The root node is an empty sequence. **Selection:** the node with the minimum makespan is selected within the set of existing nodes. **Expansion:** once a node is selected, it is expanded by adding an additional robot to the existing sequence. Consequently, the targets are assigned to the given sequence of robots via the multi-robot maximum-allowed Martin's algorithm (MAM), as described in the sequel. Along

Algorithm 1: Multi-robot MAM

Input: $\mathcal{R}, \mathcal{S}, \mathcal{N}, \mathcal{M}$, robot sequence ν
Output: $\mathcal{N}_A(0)$ and $\mathcal{T}_n, \forall n \in \mathcal{N}_A(0)$

- 1 Construct $G = (V, E)$ using $\mathcal{R}, \mathcal{S}, \mathcal{N}, \mathcal{M}$
- 2 Initialize $\mathcal{N}_A(0) \leftarrow \emptyset$
- 3 Initialize targets-left set $V_l \leftarrow V_{\text{tar}}$
- 4 **while** V_l is not empty **do**
- 5 Robot $n \leftarrow \alpha.\text{pop}()$
- 6 $V_{\text{tar}}^n, \mathcal{T}_n \leftarrow \text{Incremental-MA}(V_l, n)$
- 7 **if** $\mathcal{T}_n \neq \emptyset$ **then**
- 8 $V_l \leftarrow V_l \setminus V_{\text{tar}}^n$
- 9 $\mathcal{N}_A(0) \leftarrow \mathcal{N}_A(0) \cup \{n\}$
- 10 **Return** $\mathcal{N}_A(0), \mathcal{T}_n, \forall n \in \mathcal{N}_A(0)$

with the sequence of targets, the node cost is also returned. This node is called feasible if all targets are assigned. Then, the procedure of selection and expansion is repeated, until the planning time elapsed or all robot sequences are exhausted in the search tree. To improve search efficiency, a branch-and-bound procedure can be applied. The lower bound on the makespan of each node is computed as the minimum makespan when all robots are active, while the upper bound is given by the one-step greedy assignment where each target is assigned to the nearest robot if feasible.

B. Multi-robot Maximum-allowed Martin's Algorithm

Given the robot sequence ν , Alg. 1 is proposed to determine the active robot set $\mathcal{N}_A(0)$ and the optimal task sequence $\mathcal{T}_n$ for each active robot. The process begins by constructing the Time-Varying Node Network (TVNN) in Line 1, followed by executing Incremental Martin's Algorithm (IMA) for each robot in sequence in Line 6, which outputs the maximum set of targets each robot can monitor along with the optimal $\mathcal{T}_n$. Finally, the minimum active robot set is obtained through a greedy approach.

1) *Time-Varying Node Network:* Robots $\mathcal{N}$, targets $\mathcal{M}$, and stations $\mathcal{S}$ are modeled as nodes in a temporal graph $G = (V, E)$, where V is the node set and $E \subset V \times V$ is the edge set. The corresponding node sets are $V_N = \{\bar{n} \mid n \in \mathcal{N}\}$, $V_M = \{\bar{m} \mid m \in \mathcal{M}\}$, and $V_S = \{\bar{s} \mid s \in \mathcal{S}\}$, with $\bar{\cdot}$ distinguishing nodes from entities. To incorporate intersection constraints from II-B and charging stations, special nodes are introduced or modified as follows:

Intersection Target Nodes. Intersection constraints introduce independent time constraints beyond (4), requiring a virtual node $\bar{m}_i$ for each target $m \in \mathcal{M}$ to form V_M^{inter} . The target node set is thus $V_{\text{tar}} = V_M \cup V_M^{\text{inter}}$. Not all targets require both $\bar{m} \in V_M$ and $\bar{m}_i \in V_M^{\text{inter}}$. If $\chi_m^{\text{inter}} < \chi_m - \chi_m(0)$, $\bar{m} \in V_M$ is unnecessary as intersection monitoring already satisfies eq. (4). If χ_m^{inter} exceeds the planning horizon, $\bar{m}_i \in V_M^{\text{inter}}$ is not needed. **Decomposition of Stations.** Upon reaching a charging station s , a robot selects its charging duration. To model this, each station node $\bar{s}$ is decomposed into a *docking node* $\bar{s}_0$ and N_s *charging nodes* $\bar{s}_i$ for $i = 1, \dots, N_s$. The notation $\bar{s} = \bar{s}_i \mid i = 0, \dots, N_s$ is

Algorithm 2: Incremental Martin's Algorithm

Input: Target left set V_l , robot n
Output: $V_{\text{tar}}^n, \mathcal{T}_n$

- 1 Initialize $V_N^n \leftarrow \{\bar{n}\}, V_{\text{tar}}^n \leftarrow \emptyset, V_S^n \leftarrow \emptyset$
- 2 Initialize $L_{\bar{n},p} \leftarrow \{\}, L_{\bar{n},t} \leftarrow \{\}$
- 3 $l_0 \leftarrow (\bar{n}, \mathbf{R}_{l_0}, \emptyset)$
- 4 $L_{\bar{n},t} \leftarrow L_{\bar{n},t} \cup \{l_0\}$
- 5 $\mathcal{T}_n \leftarrow \emptyset$
- 6 **while** V_l is not empty **do**
- 7 $\bar{c} \leftarrow \arg \min_{\bar{v} \in V_l} \kappa(\bar{v}, \bar{n})$
- 8 $\bar{s} \leftarrow \arg \min_{\bar{s} \in V_S} d(\mathbf{z}_s, \mathbf{p}_{\bar{c}}(0))$
- 9 $V_{\text{tar}}^n \leftarrow V_{\text{tar}}^n \cup \{\bar{c}\}, V_S^n \leftarrow V_S^n \cup \{\bar{s}\}$
- 10 Get E^n according to (7)
- 11 $L_{\bar{c},p} \leftarrow \{\}, L_{\bar{c},t} \leftarrow \{\}, L_{\bar{s},p} \leftarrow \{\}, L_{\bar{s},t} \leftarrow \{\}$
- 12 Add a new dimension of 1 to $\hat{\mathbf{R}}_{l_{\bar{v}}}, \forall l_{\bar{v}} \in L_p^n \cup L_t^n$
- 13 Propagate all nodes in L_p^n to $\bar{c}$ and $\bar{s}$
- 14 $\mathcal{A} \leftarrow \text{MP}(G^n, L_p^n, L_t^n)$
- 15 **if** $\mathcal{A} = \emptyset$ **then**
- 16 $V_{\text{tar}}^n \leftarrow V_{\text{tar}}^n \setminus \{\bar{c}\}, V_S^n \leftarrow V_S^n \setminus \{\bar{s}\}$
- 17 **Break**
- 18 $\mathcal{T}_n \leftarrow \mathcal{A}$
- 19 **Return** $V_{\text{tar}}^n, \mathcal{T}_n$

used to denote an inseparable unit, with V_S^{dock} and V_S^{charge} representing all docking and charging nodes, respectively. The i^{th} charging node supplies $b_n^{\text{max}} \cdot i/N_s$, providing N_s charging options and enabling automated charge planning.

Let $V = V_N \cup V_{\text{tar}} \cup V_S$, where each node $\bar{v} \in V$ has a time-varying position: $\mathbf{p}_{\bar{n}}(t) = \mathbf{x}_n(0)$ for $\bar{n} \in V_N$, $\mathbf{p}_{\bar{m}}(t) = \mathbf{y}_m(t)$ for $\bar{m} \in V_M$, $\mathbf{p}_{\bar{m}_i}(t) = \mathbf{y}_m^{\text{inter}}$ for $\bar{m}_i \in V_M^{\text{inter}}$, where $\mathbf{y}_m^{\text{inter}}$ is the position of the next intersection of m , and $\mathbf{p}_{\bar{s}}(t) = \mathbf{z}_s$ for $\bar{s} \in V_S$. The resource of interest is $\mathbf{R} \in \mathbb{R}^2$, where $R^{(1)}$ is time and $R^{(2)}$ is battery consumption. Each node $\bar{v}$ has time and battery constraints $[R_{v,\min}^{n,(i)}, R_{v,\max}^{n,(i)}]$ for $i = 1, 2$, depending on robot n . Specifically, for $\bar{v} \in V_N \cup V_S$, the time constraint is $[0, +\infty]$ and the battery constraint is $[0, b_n^{\text{max}}]$. For $\bar{m} \in V_M$, the time constraint is $[0, \chi_m - \chi_m(0)]$, and the battery constraint is $[0, b_n^{\text{max}} - \gamma_n d_S(\mathbf{p}_{\bar{m}}(t))/v_n]$. For $\bar{m}_i \in V_M^{\text{inter}}$, the time constraint is $[\chi_m^{\text{inter}}(0) - T_0, \chi_m^{\text{inter}}(0)]$, and the battery constraint is $[0, b_n^{\text{max}} - \gamma_n d_S(\mathbf{p}_{\bar{m}_i}(t))/v_n]$.

Lastly, for connectivity, we define the set of reachable nodes $E_v \subset V$ for each node $v \in V$, i.e., $E_v = \{u \in V : (v, u) \in E\}$. Each edge $(v, u) \in E$ has an associated cost vector $\mathbf{C}(v, u) \in \mathbb{R}^2$. The connectivity and edge costs are summarized as follows.

$$\begin{aligned}
C^{(1)}(\bar{v}, \bar{u}) &= d(\mathbf{p}_{\bar{v}}(t), \mathbf{p}_{\bar{u}}(t))/v_n + T_0 \cdot \mathbb{I}(\bar{v} \in V_{\text{tar}}), \\
C^{(2)}(\bar{v}, \bar{u}) &= \gamma_n C^{(1)}(\bar{v}, \bar{u}), \forall \bar{v} \in V \setminus V_S^{\text{dock}}, \bar{u} \in E_{\bar{v}}, \\
C^{(1)}(\bar{s}_0, \bar{s}_i) &= -C^{(2)}(\bar{s}_0, \bar{s}_i)/\beta_s, \\
C^{(2)}(\bar{s}_0, \bar{s}_i) &= -b_n^{\text{max}} \cdot i/N_s, \forall \bar{s}_0 \in V_S^{\text{dock}}, \bar{s}_i \in E_{\bar{s}},
\end{aligned} \tag{7}$$

where $E_n = V \setminus V_N, \forall n \in V_N, E_{\bar{v}} = V \setminus V_N \cup \{\bar{v}\}, \bar{v} \in V_{\text{tar}}; E_{\bar{s}_0} = \bar{s} \setminus \bar{s}_0, \forall \bar{s}_0 \in V_S^{\text{dock}}, E_{\bar{s}_i} = V \setminus V_N \cup V_S^{\text{dock}}, \forall \bar{s}_i \in V_S^{\text{charge}}; and the $\mathbb{I}(\cdot)$ is the indicator function.$

2) *Incremental Martin's Algorithm*: Given the target left set V_l , robot n constructs an incremental sub-TVNN $G^n = (V^n, E^n)$, where $V^n = V_N^n \cup V_{\text{tar}}^n \cup V_S^n$ with $V_{\text{tar}} \subset V_{\text{tar}}$ and $V_S \subset V_S$. In each iteration, the algorithm selects the optimal target node and its nearest charging station, adding them to V^n . Target priority is determined by $\kappa(\cdot, \bar{n})$ in Line 7, i.e.,

$$\kappa(\bar{v}, \bar{n}) = a_1 \cdot d(\mathbf{p}_{\bar{v}}(0), \mathbf{p}_{\bar{n}}(0)) / v_n + a_2 \cdot R_{\bar{v}, \max}^{n, (1)}, \quad (8)$$

where $\bar{v} \in V_{\text{tar}}^n$ and $a_1, a_2 \in \mathbb{R}^+$ are coefficients. The function $\kappa(\cdot, \bar{n})$ is defined by the travel time from $\bar{n}$ to $\bar{v}$ and the time constraint of $\bar{v}$, ensuring that nodes closer to $\bar{n}$ and with more urgent time constraints are prioritized.

The IMA follows a *label-setting* approach, utilizing labels to explore the graph and identify the Pareto front. Each label l is represented as a tuple $(\bar{v}, \mathbf{R}_l, \bar{c})$, where $\bar{v}$ is the node, $\mathbf{R}_l$ is the resource vector, and $\bar{c}$ is the predecessor node. Given the multidimensional nature of $\mathbf{R}_l$, the *dominance* rule is applied for comparison. Specifically, “ $\mathbf{a}$ dominates $\mathbf{b}$ ” is denoted as “ $\mathbf{a} <_p \mathbf{b}$ ”, meaning $\mathbf{b}$ does not yield a better outcome than $\mathbf{a}$. This relation holds if $(a_1, \dots, a_n)^T <_p (b_1, \dots, b_n)^T \Leftrightarrow \mathbf{a} \neq \mathbf{b} \wedge a_i \leq b_i, \forall i \in \{1, \dots, n\}$. Labels are managed in permanent sets $L_{v,p}$ and temporary sets $L_{v,t}$ for each node $v \in V$. The set $L_{v,p}$ contains Pareto-optimal labels at v , while $L_{v,t}$ holds intermediate labels. For convenience, we define the total permanent and temporary sets as $L_p^n = \bigcup_{\bar{v} \in V^n} L_{\bar{v},p}$ and $L_t^n = \bigcup_{\bar{v} \in V^n} L_{\bar{v},t}$, respectively.

To ensure all target nodes are visited in each plan, we introduce a binary vector $\hat{\mathbf{R}} \in \mathbb{R}^{N_{\text{tar}}}$, where $N_{\text{tar}} = |V_{\text{tar}}^n|$, to indicate the visitation status of each node in V_{tar}^n . Each target node $\bar{v} \in V_{\text{tar}}^n$ is assigned an index $i_{\bar{v}} \in \{1, 2, \dots, N_{\text{tar}}\}$, with each dimension of $\hat{\mathbf{R}}$ corresponding to a target node. A value of 1 denotes an unvisited node, while 0 indicates it has been visited. The initial resource state is set as $\hat{\mathbf{R}}_{i_0}^{\bar{n}} = \mathbf{1}_{N_{\text{tar}}}$. The cost vectors associated with node resources are given by

$$\hat{\mathbf{C}}(\bar{v}, \bar{u}) = -\mathbf{e}_{i_{\bar{u}}} \mathbb{I}(\bar{u} \in V_{\text{tar}}^n), \quad (9)$$

where $\mathbf{e}_i \in \mathbb{R}^{N_{\text{tar}}}$ is a unit vector with 1 in the i^{th} dimension and 0 elsewhere. Since revisiting target nodes is unnecessary, the resource constraint $\hat{R}^{(i)} \geq 0$ is imposed to expedite the search process. By concatenating the respective resources and costs, the final resource and cost are obtained as:

$$\tilde{\mathbf{R}} = \begin{pmatrix} \mathbf{R} \\ \hat{\mathbf{R}} \end{pmatrix}, \tilde{\mathbf{C}} = \begin{pmatrix} \mathbf{C} \\ \hat{\mathbf{C}} \end{pmatrix}. \quad (10)$$

After preparing new nodes in Lines 11-12, all permanent nodes are propagated to $\bar{c}$ and $\bar{s}$, and Martin's Process (MP), inspired by Martin's Algorithm [24], is executed to determine the optimal task sequence $\mathcal{A}$ within the current sub-network G^n . If MP fails to return a feasible solution, the maximal set of targets monitored by robot n is obtained by removing the newly added nodes, reverting to the optimal $\mathcal{T}_n$ from the previous iteration.

3) *Martin's Process*: In each iteration, the optimal l_c is selected from L_t^n , transferred into L_p^n , and propagated from $\bar{c}$ to all its viable nodes. The optimization objective is defined by the lexicographic order comparison function $<_{\text{lex}}$ (Line 2).

Algorithm 3: Martin's Process (MP)

Input: $G^n = (V^n, E^n), L_p^n, L_t^n$

Output: Multiobjective shortest path $\mathcal{A}$

```

1 while  $L_t^n$  is not empty do
2    $l_{\bar{c}} \leftarrow \arg \min_{l_{\bar{v}} \in L_t^n} <_{\text{lex}} \mathbf{R}_{l_{\bar{v}}}$ 
3    $\bar{c} \leftarrow l_{\bar{c}}.\text{Node}()$ 
4   if terminate( $l_{\bar{c}}$ ) then
5     Return  $\mathcal{A} \leftarrow \text{backtracking}(l_{\bar{c}})$ 
6    $L_{\bar{c},t} \leftarrow L_{\bar{c},t} \setminus \{l_{\bar{c}}\}$ 
7    $L_{\bar{c},p} \leftarrow L_{\bar{c},p} \cup \{l_{\bar{c}}\}$ 
8   for  $\bar{v} \in E_{\bar{c}}$  do
9     // Propagate to  $\bar{v}$ 
10     $\tilde{\mathbf{R}}_{l_{\bar{v}}} \leftarrow \tilde{\mathbf{R}}_{l_{\bar{c}}} + \tilde{\mathbf{C}}(\bar{c}, \bar{v})$ 
11    if  $R_{l_{\bar{v}}}^{(i)} \leq R_{\bar{v}, \max}^{n, (i)}, \forall i \leq 2$  and
12       $\hat{R}_{l_{\bar{v}}}^{(i)} \geq 0, \forall i \leq N_{\text{tar}}$  then
13       $R_{l_{\bar{v}}}^{(i)} \leftarrow \max\{R_{l_{\bar{v}}}^{(i)}, R_{\bar{v}, \min}^{n, (i)}\}, \forall i \leq 2$ 
14       $l_{\bar{v}} \leftarrow (\bar{v}, \tilde{\mathbf{R}}_{l_{\bar{v}}}, \bar{c})$ 
15      if  $\nexists l \in L_{\bar{v},p} \cup L_{\bar{v},t}$  s.t.  $l <_p l_{\bar{v}}$  then
16         $L_{\bar{v},t} \leftarrow L_{\bar{v},t} \setminus \{l \in L_{\bar{v},t} : l_{\bar{v}} <_p l\}$ 
17         $L_{\bar{v},t} \leftarrow L_{\bar{v},t} \cup \{l_{\bar{v}}\}$ 
18 Return  $\emptyset$ 

```

This multidimensional comparison rule prioritizes lower time consumption while also minimizing battery usage. Notably, $<_{\text{lex}}$ satisfies the properties of *dominance* and *monotonicity*, ensuring that labels are extracted in the correct order [25]. The operation $\min^{<_{\text{lex}}}$ in Line 2 refers to the minimum selection based on the $<_{\text{lex}}$ ordering. The *propagate* process begins by generating a new resource vector through cost addition in Line 9. If the new vector is valid (Line 10), it is adjusted by $R_{v, \min}^{n, (i)}$ in Line 11 and create a new label. The right boundary of the resource window is a strict constraint that cannot be exceeded, while the left boundary, if not reached, is compensated, representing waiting time for $i = 1$ and invalid excess charge for $i = 2$. If the new label is non-dominated at $\bar{v}$, dominated labels in the temporary set are removed, and the new label is inserted. When the optimal label $l_{\bar{c}}$ meets the early termination condition, i.e., $\hat{\mathbf{R}}_{l_{\bar{c}}} = \mathbf{0}_{N_{\text{tar}}}$, the optimal task sequence $\mathcal{A}$ is obtained by backtracking each label's predecessor from $l_{\bar{c}}$.

C. Online Execution

1) *Online Execution and Adaptations*: We define a Replanning Horizon T_h to ensure timely corrective actions when constraints are at risk of being violated. By analytically computing the exact time at which each target is expected to violate its constraints, constraint satisfaction can be proactively monitored. If any violation is predicted to occur within T_h , a replanning process is triggered to maintain feasibility. The duration of T_h is chosen such that the UAV can reach any drone on the field, ensuring that necessary interventions can always be executed in time.

We modify Alg. 1 and Alg. 2 in two aspects. First, in Line 3, instead of considering the entire set V_{tar} , only

targets that violated constraints in prediction, have completed monitoring, or were newly generated are included. Second, when constructing the sub-TVNN, ongoing targets from the previous allocation are prioritized and incorporated first, followed by the remaining targets in V_l . These modifications ensure recursive feasibility, as targets in V_l must satisfy $R_{v,\max}^{n,(i)} \geq T_h$ due to the triggering condition. During online execution, at each time step $t \geq 0$, the UAV follows the current plan and determines its actions accordingly. Then the robots will compute the corresponding $\mathbf{u}_n(t)$ and move according to (1), which can be computed through the NMPC in [26], and the targets move according to their kinematics and choose the next road randomly at the meantime. This process continues until a target is predicted to violate its constraints, at which point the next replanning event is triggered to ensure feasibility.

2) *Complexity Analysis*: According to [27], the complexity of IMA is $\mathcal{O}((N_{\text{tar}} + N_s |V_S^n|) N_{\text{tar}} N_{\text{label}}^2)$, where N_{label} denotes the number of non-dominated labels at the algorithm's termination. Since N_{tar} and $|V_S^n|$ are bounded by C_n , the complexity simplifies to $\mathcal{O}(N_s C_n^2 N_{\text{label}}^2)$. Thus, the complexity of MAM is $\mathcal{O}(|\mathcal{N}_A| N_s C_n^2 N_{\text{label}}^2)$.

D. Generalization

The proposed planning framework can be generalized in the following aspects: (I) *Robot Failures*. In the event of a robot failure, e.g., an unexpected UAV crash, this event is immediately treated as a triggering condition for online adaptation. Consequently, all targets assigned to the crashed UAV are added to V_l as the set of unassigned targets. A replanning is initiated to redistribute these tasks, via the same adaptation algorithm in Sec. III-C. (II) *Free Membership of Targets*. In case new targets enter the workspace or existing targets leave the workspace, the proposed framework can still be applied. Namely, any target that leaves the workspace during execution is removed from the Node Network in real time, which automatically is not considered in the assignment. Similarly, when a new target enters the scene, it is detected by at least one robot and added to the set of targets to be assigned. Then, the same procedure can be applied.

IV. NUMERICAL EXPERIMENTS

For further validation, extensive numerical simulations are conducted. The algorithm is implemented in Python3 and tested on a laptop with an Intel Core i7-1280P CPU. Simulation videos can be found in the supplementary files.

A. System Setup

As shown in Fig. 1 and Fig. 3, our algorithm performs well in both scenarios. In Fig. 1, 10 targets move within a $50 \text{ m} \times 50 \text{ m}$ road network with three charging stations. Each robot follows a linear UAV motion model at a fixed height, starting from the central drone pad with a fully charged battery, $b_n^{\max} = 10.0$. There is no limit on the number of robots at the drone pad. Robots have a maximum speed of $v_n^{\max} = 1.5 \text{ m/s}$, a battery consumption rate of $\gamma_n = 0.2 \text{ s}^{-1}$, and a charging rate of $\beta_s = 0.2 \text{ s}^{-1}$. The monitoring range

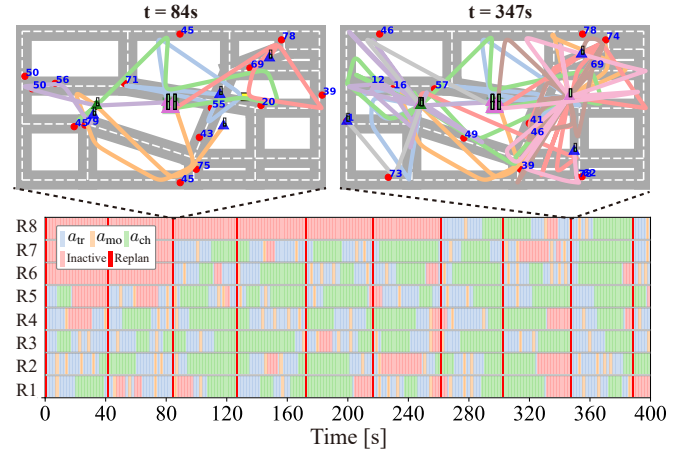


Fig. 3. **Top-Left** and **Top-Right**: Trajectories of 8 robots actively monitoring 15 targets in the considered scenario at $t=84 \text{ s}$ and $t=347 \text{ s}$ respectively. **Bottom**: The action of 8 robots at each time during the simulation of 400 s.

in (2) is $R_n = 3 \text{ m}$, and the capacity is uniformly $C_n = 6$ for all $n \in \mathcal{N}$. Targets are initially distributed randomly along roads and choose their next segments randomly at intersections, unknown to the robots. Each target moves at $\|\mathbf{v}_m\| = 0.2 \text{ m/s}$, with a maximum monitoring interval of $\chi_m = 80 \text{ s}$ and a monitoring duration of $T_0 = 2 \text{ s}$. The number of charging nodes is $N_s = 5$, and the triggering horizon is $T_h = 30 \text{ s}$. The simulation runs for 400 s. In Fig. 3, the parameters are modified as follows: the road network size is $100 \text{ m} \times 50 \text{ m}$, with two charging stations. The speed is set to 3 m/s , and the number of targets is 15.

B. Results

The final results are shown in Fig. 1 and Fig. 3. In Fig. 1, each replan employs an average of 3.7 robots, with a computational cost of 0.063 s . The average planning time per robot $n \in \mathcal{N}_A$ is 0.020 s , with 22.6 nodes in each robot's planning network. The average number of non-dominated labels at the end of each planning process is 115.4. Overall, the average replan interval is 46.13 s , and the average number of employed robots is 3.5. Each robot n charges for an average of 109.5 s , accounting for 49.6% of its active period, indicating a high degree of autonomous charging management. According to Fig. 1, in both scenarios, the batteries of all robots remain positive, and the monitoring intervals of all targets stay below χ_m throughout the simulation. In the bottom-right figure of Fig. 1, at least one robot is present in the two rows of each target during every replan, ensuring that each target remains in the task sequence of at least one robot at all times.

In the scenario depicted in Fig. 3, each replan employs an average of 5.5 robots, with an average computational cost of 0.164 s . For each replan, the average planning time per robot $n \in \mathcal{N}_A$ is 0.024 s , with an average of 16.5 nodes in each robot's planning network and 111.7 non-dominated labels at the end of planning. Overall, throughout the simulation, the average replan interval is 43.11 s , and the average number of employed robots is 6.2. Each robot n spends an average of 143.5 s charging, accounting for 47.7% of its active period.

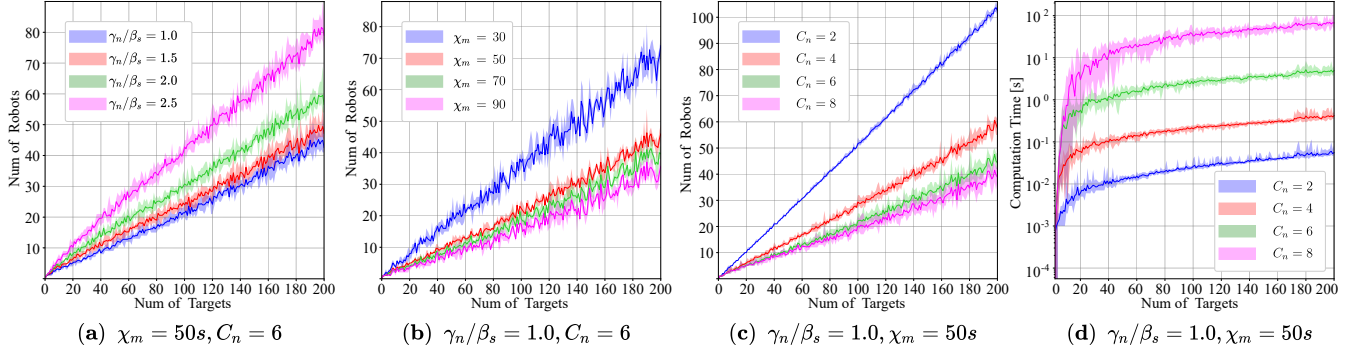


Fig. 4. Illustration of scalability analysis results. The number of active robots with an increasing number of targets, concerning γ_n/β_s (a), T_m (b) and C_n (c). The computation time with an increasing number of targets, concerning C_n (d).

TABLE I
COMPARISON ANALYSIS RESULTS

Cases	Methods	Success Rate [%]	Average $ \mathcal{N}_A $	Num of Nodes	Computation Time [s]
$M=8$	Ours	100	2.81	15.62	0.09
	NI-MAM	100	2.54	24.30	77.98
	CMA	100	2.40	5228.70	2.60
	PeT	86.7	2.90	15.36	0.10
	GCF	63.4	3.38	2.34	0.01
$M=30$	Ours	100	7.69	16.57	2.41
	NI-MAM	100	7.34	32.59	470.57
	CMA	- *	> 6	1.41e14	> 600
	PeT	78.4	7.86	16.88	2.39
	GCF	47.9	12.76	2.36	0.04
$M=100$	Ours	100	23.54	16.61	6.42
	NI-MAM	-	> 20	118.51	>600
	CMA	-	> 20	7.15e52	>600
	PeT	69.5	25.18	16.35	6.43
	GCF	33.1	41.75	2.40	0.09

*“-” means the result cannot be obtained within the limited time.

C. Comparisons

To validate the proposed framework (**Ours**), we conduct a quantitative comparison against four baselines, including two ablation studies and two common approaches: (i) Non-incremental Maximum-allowed Martin’s Algorithm (**NI-MAM**): Targets are not added incrementally for each robot but are all included in the sub-TVNN at each step. The robot returns a sequence maximizing visited targets. (ii) Centralized Martin’s Algorithm (**CMA**): All robots are treated as a single system, where Martin’s algorithm is applied in a centralized manner [28]. (iii) Periodic Triggering (**PeT**): The triggering condition is replaced by periodic triggering. (iv) Greedy Closest-First (**GCF**): Targets are assigned greedily, iteratively allocating each robot to the nearest available target until no further feasible assignments remain.

As summarized in Table I, the the success rate, average number of robots, average nodes per replan, and computation time are compared in the nominal setup of Fig. 1 with $C_n = 5$ and varying M . Although our method uses slightly more robots than (i) and (ii), the average number of nodes remains stable (always < 20), and the computation time increases more gradually (6.42 s at $M = 100$). In contrast, (i)

and (ii) show a sharp increase in both metrics as M grows, demonstrating higher scalability for our method. The PeT method exhibits similar $\mathcal{N}_A$, average nodes, and computation time as ours but has a lower success rate (below 100%) due to the absence of active replan time prediction. The GCF method has smaller average nodes and computation time due to its direct selection principle, but its success rate is lower and requires more robots.

D. Scalability Analysis

The scalability of the proposed algorithm is analyzed with respect to four aspects: the number of targets M , the ratio between battery consumption rate γ_n and charging rate β_s , the upper bound of the interval χ_m , and the capacity C_n . The results are summarized in Fig. 4, with the nominal setup shown in Fig. 1. From (a), (b), and (c), we observe that the ratio between the total number of targets and the average number of active robots remains nearly constant, which we define as *fleet efficiency*, denoted as η , and is upper-bounded by C_n . In (a), we see that as the ratio of battery consumption to charging rate increases, fleet efficiency decreases from 4.5 to 2.5, as each robot can move less and track fewer targets under higher consumption-charging ratios. In (b), when the maximum interval χ_m increases from 30 s to 90 s, fleet efficiency increases from 2.7 to 6.0, as a larger χ_m allows each robot to track more targets due to the extended tracking time. From (c), we observe that as C_n increases from 2 to 8, fleet efficiency rises from 1.9 to 5.0, as more targets can be monitored by each robot. Figure (d) examines computation time for different C_n values as the number of targets increases. Our method demonstrates strong scalability, as it takes less than 10 s to compute when $M = 200$ and $C_n = 6$. It also highlights that C_n is a crucial factor influencing computation time, with an exponential relationship between computation time and C_n .

E. Generalization

Performances under generalized scenarios are also validated as shown in Fig. 5. Namely, one robot failed at $t = 20$ s, after which the targets it was tracking are reassigned to other robots in the new replan. The adaptation takes 0.02 s and the local plans of the other robots are modified accordingly. Moreover, 6 targets enter freely into the workspace during

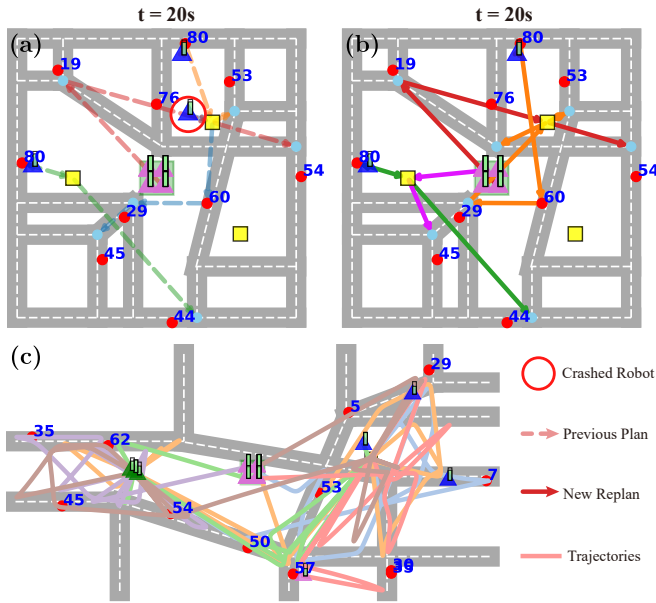


Fig. 5. Illustration of the generalization circumstances. (a): Previous plan result at $t = 20s$ when a robot accidentally crashes. (b): New replan after the robot crashed. (c): Trajectories of 6 robots monitoring 15 targets that can dynamically enter and exit the road network over a period of 400 s

online execution, with the setup otherwise identical to Fig. 3. The trajectories of the employed robots are adapted to monitor the new targets.

V. CONCLUSION

This work addresses the long-term monitoring of dynamic targets in a road network using a fleet of aerial robots with limited resources. We propose a hierarchical approach that incrementally assigns targets, optimizes monitoring sequences and charging strategies, and adapts online to real-time constraints. Our method ensures strict adherence to resource and monitoring constraints while minimizing the active fleet size. Extensive simulations demonstrate its scalability and effectiveness in deploying a small UAV fleet for large-scale target monitoring.

REFERENCES

- [1] W. Burgard, M. Moors, C. Stachniss, and F. E. Schneider, "Coordinated multi-robot exploration," *IEEE Transactions on robotics*, vol. 21, no. 3, pp. 376–386, 2005.
- [2] M. Tzes, N. Bousias, E. Chatzipantazis, and G. J. Pappas, "Graph neural networks for multi-robot active information acquisition," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 3497–3503.
- [3] S. Kalluraya, G. J. Pappas, and Y. Kantaros, "Multi-robot mission planning in dynamic semantic environments," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 1630–1637.
- [4] N. Atanasov, J. Le Ny, K. Daniilidis, and G. J. Pappas, "Information acquisition with sensing robots: Algorithms and error bounds," in *2014 IEEE International conference on robotics and automation (ICRA)*. IEEE, 2014, pp. 6447–6454.
- [5] P. Dames, M. Schwager, V. Kumar, and D. Rus, "A decentralized control policy for adaptive information gathering in hazardous environments," in *IEEE Conference on Decision and Control (CDC)*, 2012, pp. 2807–2813.
- [6] T. H. Chung, J. W. Burdick, and R. M. Murray, "A decentralized motion coordination strategy for dynamic target tracking," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2006, pp. 2416–2422.

- [7] B. Schlotfeldt, D. Thakur, N. Atanasov, V. Kumar, and G. J. Pappas, "Anytime planning for decentralized multirobot active information gathering," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1025–1032, 2018.
- [8] Y. Kantaros, B. Schlotfeldt, N. Atanasov, and G. J. Pappas, "Asymptotically optimal planning for non-myopic multi-robot information gathering," in *Robotics: Science and Systems*, 2019, pp. 22–26.
- [9] J. Le Ny and G. J. Pappas, "On trajectory optimization for active sensing in gaussian process models," in *IEEE Conference on Decision and Control (CDC)*, 2009, pp. 6286–6292.
- [10] Y. Sung, A. K. Budhiraja, R. K. Williams, and P. Tokekar, "Distributed assignment with limited communication for multi-robot multi-target tracking," *Autonomous Robots*, vol. 44, no. 1, pp. 57–73, 2020.
- [11] P. Tokekar, V. Isler, and A. Franchi, "Multi-target visual tracking with aerial robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014, pp. 3067–3072.
- [12] L. Zhou and P. Tokekar, "Sensor assignment algorithms to improve observability while tracking targets," *IEEE Transactions on Robotics*, vol. 35, no. 5, pp. 1206–1219, 2019.
- [13] J. Chen, Z. Tang, and M. Guo, "Accelerated k-serial stable coalition for dynamic capture and resource defense," *IEEE Robotics and Automation Letters*, vol. 9, no. 1, pp. 443–450, 2024.
- [14] S. Chopra, G. Notarstefano, M. Rice, and M. Egerstedt, "A distributed version of the hungarian method for multirobot assignment," *IEEE Transactions on Robotics*, vol. 33, no. 4, pp. 932–947, 2017.
- [15] P. Dames, P. Tokekar, and V. Kumar, "Detecting, localizing, and tracking an unknown number of moving targets using a team of mobile robots," *The International Journal of Robotics Research*, vol. 36, no. 13-14, pp. 1540–1553, 2017.
- [16] L. Zhou, V. Tzoumas, G. J. Pappas, and P. Tokekar, "Resilient active target tracking with multiple robots," *IEEE Robotics and Automation Letters*, vol. 4, no. 1, pp. 129–136, 2018.
- [17] L. Zhou and V. Kumar, "Robust multi-robot active target tracking against sensing and communication attacks," *IEEE Transactions on Robotics*, 2023.
- [18] C. Di Franco and G. Buttazzo, "Energy-aware coverage path planning of uavs," in *IEEE international conference on autonomous robot systems and competitions*, 2015, pp. 111–117.
- [19] A. S. Seisa, S. G. Satpute, B. Lindqvist, and G. Nikolakopoulos, "An edge-based architecture for offloading model predictive control for uavs," *Robotics*, vol. 11, no. 4, p. 80, 2022.
- [20] C. X. Guo, K. Sartipi, R. C. DuToit, G. A. Georgiou, R. Li, J. O'Leary, E. D. Nerurkar, J. A. Hesck, and S. I. Roumeliotis, "Resource-aware large-scale cooperative three-dimensional mapping using multiple mobile devices," *IEEE Transactions on Robotics*, vol. 34, no. 5, pp. 1349–1369, 2018.
- [21] J. Homberger, "A multi-agent system for the decentralized resource-constrained multi-project scheduling problem," *International Transactions in Operational Research*, vol. 14, no. 6, pp. 565–589, 2007.
- [22] M. Afrin, J. Jin, A. Rahman, A. Rahman, J. Wan, and E. Hossain, "Resource allocation and service provisioning in multi-agent cloud robotics: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 2, pp. 842–870, 2021.
- [23] A. I. Mourikis and S. I. Roumeliotis, "Optimal sensor scheduling for resource-constrained localization of mobile robot formations," *IEEE Transactions on Robotics*, vol. 22, no. 5, pp. 917–931, 2006.
- [24] E. Q. V. Martins, "On a multicriteria shortest path problem," *European Journal of Operational Research*, vol. 16, no. 2, pp. 236–245, 1984.
- [25] J. M. Paixão and J. L. Santos, "Labeling methods for the general case of the multi-objective shortest path problem—a computational study," in *Computational Intelligence and Decision Making: Trends and Applications*. Springer, 2013, pp. 489–502.
- [26] S. Wang, Y. Kantaros, and M. Guo, "Uncertainty-bounded active monitoring of unknown dynamic targets in road-networks with minimum fleet," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 4584–4590.
- [27] P. M. de las Casas, A. Sedeno-Noda, and R. Borndörfer, "An improved multiobjective shortest path algorithm," *Computers & Operations Research*, vol. 135, p. 105424, 2021.
- [28] P. Schillinger, M. Bürger, and D. V. Dimarogonas, "Multi-objective search for optimal multi-robot planning with finite ltl specifications and resource constraints," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 768–774.