

Spatio-temporal monitoring using energy-constrained, sensing robots

Buyukkocak, Ali; Di Cairano, Stefano; Vinod, Abraham P.

TR2026-144 October 01, 2026

Abstract

We consider the problem of data-driven spatiotemporal monitoring using a team of mobile robots. Specifically, we model the phenomena being monitored as unknown functions that may vary in space and time. Given limited prior information, we require an energy-constrained team to collect data and identify all critical events, i.e., locations and time instants at which the unknown function crosses a pre-specified threshold. We consider a team that consists of mobile sensors (e.g., drones) and charging stations (e.g., ground carrier vehicles). We balance the monitoring requirements with energy and dynamics constraints on the team using an anytime, optimization-based, iterative approach that updates the robot deployments based on data collected online. We also provide sufficient conditions under which our approach yields a non-trivial collection of space-time intervals that include all critical events. We demonstrate our approach using hardware experiments with drones and provide an empirical analysis of its performance and scalability in simulations.

IEEE Transactions on Automation Science and Engineering 2026

Spatio-temporal monitoring using energy-constrained, sensing robots

Ali Tefvik Buyukkocak, Stefano Di Cairano, and Abraham P. Vinod[†]

Abstract—We consider the problem of data-driven spatio-temporal monitoring using a team of mobile robots. Specifically, we model the phenomena being monitored as unknown functions that may vary in space and time. Given limited prior information, we require an energy-constrained team to collect data and identify all critical events, i.e., locations and time instants at which the unknown function crosses a pre-specified threshold. We consider a team that consists of mobile sensors (e.g., drones) and charging stations (e.g., ground carrier vehicles). We balance the monitoring requirements with energy and dynamics constraints on the team using an anytime, optimization-based, iterative approach that updates the robot deployments based on data collected online. We also provide sufficient conditions under which our approach yields a non-trivial collection of space-time intervals that include all critical events. We demonstrate our approach using hardware experiments with drones and provide an empirical analysis of its performance and scalability in simulations.

Note to Practitioners—Many field operations, such as disaster response, environmental monitoring, agriculture, and infrastructure inspection, require tracking quantities like water level, pollution, or radiation over large areas and long periods of time. Manually deploying and maintaining sensors in these settings can be costly, slow, or unsafe. This work proposes a planning approach for teams of mobile robots (for example, drones) that move through the environment, collect measurements, and visit charging stations (for example, ground vehicles) when their batteries run low. Using the data they gather, the approach estimates where and when important changes may occur (for example, a quantity crossing a safety threshold) and automatically updates robot routes and charging schedules. The updated routes ensure that these “at-risk” areas are monitored more closely without exhausting energy. Using physical and simulated experiments, we show that our approach can improve coverage and reliability compared to existing persistent but non-adaptive coverage solutions. A current limitation of our approach is the assumption of almost instantaneous recharging of the mobile sensors (for example, via battery swapping) to provide desirable guarantees for the data-driven monitoring approach. Our future work will explicitly account for realistic charging times and their impact on deployment and monitoring.

Index Terms—Planning, Scheduling and Coordination; Co-operating Robots; Environment Monitoring and Management; Optimization and Optimal Control;

I. INTRODUCTION

Monitoring using autonomous search teams has several applications including environmental monitoring, precision agriculture, search and rescue and disaster recovery [1]–[5]. Mobile sensor teams (e.g. drones) offer a promising solution due to their mobility and flexibility, and can enable autonomous

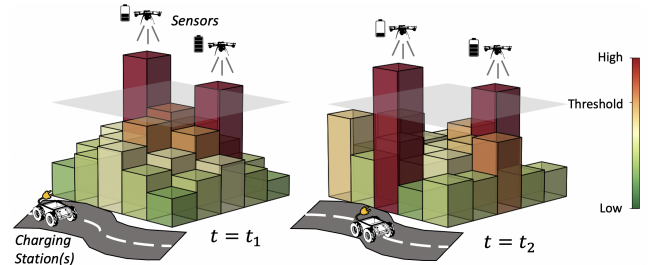


Fig. 1: Data-driven multi-agent search under spatio-temporal constraints. Measurements of time-varying cell values are used to predict critical events (i.e., threshold violations) that inform the planner on which cells to visit and when. See <https://youtu.be/Z-v2gqXjmY8> for an overview of the work.

detection and real-time response to dynamic environmental changes. However, the existing literature has paid limited attention to making data-driven team deployment decisions while respecting physical constraints, including limited energy and dynamics. This paper *proposes novel theory and algorithms for coordinated, autonomous, data-driven monitoring of spatio-temporally varying phenomena using teams of mobile robots subject to physical constraints* (see Figure 1).

Several strategies have been proposed for multi-agent search (see surveys [1]–[3]). Some of the popular approaches involve submodular maximization [6], Voronoi-based search [7], [8], active sensing/perception [9], graph-based search [10], [11], statistical learning [12]–[19], and bandit algorithms [20]–[22]. Only some of these approaches, specifically statistical learning and bandits-based approaches, adapt to data collected online, provide finite-time guarantees on the search performance, and accommodate spatio-temporal variations in the environments.

Statistical learning-based approaches [12]–[17] typically focus on estimating the spatio-temporal function modeling the environmental phenomenon using data collected by one or more mobile sensing robots. For example, [12]–[15] use Gaussian process regression to quantify estimation uncertainty and guide the sensing team toward collecting data that reduces the uncertainty. Similarly, [16] uses energy-aware ergodic search to persistently monitor the environment by deploying the team according to a time-varying target information distribution. Most of these works handle the physical constraints on the team such as dynamics and energy limitations using optimization-based planning or graph-based routing. However, they often exploit spatial correlation to facilitate extrapolation from finite data and rely on prior information, such as Gaussian process hyperparameters or statistics of the underlying spatio-temporal function, that may not always be known *a priori* and may need to be learned online [17]. On the other hand, [18], [19] consider a single unconstrained robot that

[†]Corresponding author.

Ali Tefvik Buyukkocak was with the Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA 02139 USA. He is now with the Department of Electrical and Computer Engineering at Northeastern University, Boston, MA 02115 USA (email: a.buyukkocak@northeastern.edu).

Stefano Di Cairano and Abraham P. Vinod are with Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA 02139 USA (email: {dicairano, abraham.p.vinod}@ieee.org).

visits different monitoring sites to track discrete events, with visit schedules designed from known event statistics. Our approach also focuses on detecting discrete events (threshold violations), rather than estimating the entire spatio-temporal function, but provides guarantees of exhaustiveness and non-triviality. However, our approach makes no assumption of spatial correlation or event statistics, and coordinates a team of constrained, sensing robots along with mobile charging stations to detect these events using optimization-based planning.

Bandit-based algorithms [20]–[22] focus on detecting regions of interest in the search environment that may vary spatially but are assumed to be time-invariant. The time-invariance assumption is typically needed to construct *upper confidence bounds* from the data collected online. In contrast, our approach considers environments that vary in both *space and time*, and seeks to detect all relevant discrete events using a team of constrained sensing robots.

The persistent monitoring literature has also considered the use of pre-determined paths [10], [23]–[25]. In these approaches, the path design explicitly accounts for physical constraints on the team. However, because these paths are fixed in advance, they may miss time-varying events when revisit rates are too low for small teams, or under-utilize available sensing resources for large teams.

We now briefly review the existing literature on multi-agent planning for energy-constrained robots. Existing approaches typically account for energy limitations by minimizing the travel distance [14], [15], [26] or by incorporating it as a soft constraint [27], [28]. Recent literature has shifted toward treating energy limits as hard constraints to guarantee mission safety using optimization [17], [26], [29], [30], scheduling [16], [31], [32], or via control barrier functions [33]. However, most of these approaches typically focus on persistent coverage, consider coordination of the sensors with a single charging station, and design conservative return policies. This paper also models physical limitations of the mobile sensors through a hard energy constraint: each sensor may make at most K_{charge} consecutive moves before recharging. It proposes an anytime, optimization-based framework that updates routes online for a team of mobile sensors and mobile charging stations, and explicitly balances strict energy constraints with spatiotemporal critical-event monitoring.

The main contribution of this work is a *data-driven approach for spatio-temporal monitoring that uses limited prior information to 1) correctly infer all critical events via novel temporal estimators of critical events, and 2) generates team deployments to collect data that guarantees satisfaction of temporal sensing requirements and energy and motion constraints on the team*. We demonstrate several properties for the proposed approach, including recursive feasibility, upper bounds on the time interval lengths for critical event detection, anytime guarantees, and uniqueness of the estimated intervals. We validate the approach via numerical simulations and physical experiments with drones.

We model the spatio-temporal monitoring problem using discrete state space. Inspired from existing works for multi-agent (constrained) motion planning [1]–[3], [21], [22], [30], [32], we make deployment decisions to solve the monitoring

problem via an integer program. While integer optimization-based approaches provide significant modeling flexibility (e.g., enforcing collision avoidance, handling different motion constraints), they typically involve significant computational effort that may cause scalability issues. A popular approach is to rely on a receding horizon formulation that alleviates some of the computational burden at the cost of optimality [1], [2], [30]. Using physical experiments and simulations, we show that our approach is suitable for small to medium-sized teams.

Notation: Let $\mathbb{R}_{\geq 0}$ denote the set of non-negative real numbers. For a function $f : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$, a scalar $\theta \in \mathbb{R}$, and an operator $\sim \in \{\leq, =, \geq\}$, define the corresponding set $\{f \sim \theta\} = \{t \in \mathbb{R}_{\geq 0} \mid f(t) \sim \theta\}$. Let f' and f'' denote the first and second derivatives of f . For a finite set $\mathcal{S}$, let $2^{\mathcal{S}}$ denote the power set, the set of all subsets of $\mathcal{S}$. For an interval $\mathcal{I} \subset \mathbb{R}_{\geq 0}$, we denote its length by $|\mathcal{I}|$.

II. PROBLEM STATEMENT

We outline the primary research questions this paper aims to address, and formalize them as problem statements at the end of this section.

Research Question 1. (DATA-DRIVEN MONITORING) *How can energy-constrained mobile robots use limited online data to detect critical events in a changing environment?*

Research Question 2. (THEORETICAL GUARANTEES) *When can we guarantee that the intervals used to localize critical events remain bounded and capture all critical events?*

Consider a search team consisting of autonomous, heterogeneous mobile robots, and a search environment modeled as a grid $\mathcal{G}$. We partition $\mathcal{G}$ into three sets: $\mathcal{G} = \mathcal{S} \cup \mathcal{O} \cup \mathcal{R}$. $\mathcal{S}$ is the set of candidate cells to be monitored, $\mathcal{O}$ is the set of cells that must not be visited by the search team (e.g., obstacles or no-fly zones), and $\mathcal{R}$ is the remaining set of grid cells that may be visited by the search team but do not need monitoring. The goal of the spatiotemporal monitoring task is to use finite data collected by the search team to identify the so-called *occurrence set* for every cell in $\mathcal{S}$.

Definition 1. (CRITICAL EVENTS AND OCCURRENCE SET) *Let $f_l : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ be an a priori unknown function associated with each cell $l \in \mathcal{S}$, and let $\theta \in \mathbb{R}$ be a user-specified threshold. A critical event occurs at a cell l at time $t \geq 0$ when $f_l(t) = \theta$. The set $\{f_l = \theta\}$ is the occurrence set, the set of all critical events at l .*

The spatio-temporal monitoring problem thus seeks a collection of intervals in space and time $\{\mathcal{I}_l\}_{l \in \mathcal{S}}$. For each cell $l \in \mathcal{S}$, $\mathcal{I}_l \triangleq \{\mathcal{I}_1, \mathcal{I}_2, \dots, \mathcal{I}_{|\mathcal{I}_l|}\}$ is a collection of time intervals ($\mathcal{I}_i \subset \mathbb{R}_{\geq 0}$), such that for a (small) burn-in time $T_B > 0$,

$$\forall t_l \in \{f_l = \theta\} \cap \{t \geq T_B\}, \exists \mathcal{I} \in \mathcal{I}_l \text{ s.t. } t_l \in \mathcal{I}. \quad (1)$$

Informally, (1) ensures that $\{\mathcal{I}_l\}_{l \in \mathcal{S}}$ contains all critical events in all cells in $\mathcal{S}$ that occur after the burn-in time T_B . The burn-in time allows the search team to collect initial data to initialize a monitoring algorithm.

To facilitate the construction of $\{\mathcal{I}_l\}_{l \in \mathcal{S}}$, we assume that $f_l \in \mathcal{F}$ for each $l \in \mathcal{S}$, where $\mathcal{F}$ is a known function class that encodes available prior information about the monitoring task. We do not assume any spatial correlation on the collected data, and our assumption on the temporal correlation in the collected data may be suitably captured via an appropriate choice of $\mathcal{F}$. As an example, consider Lipschitz continuous functions,

$$\mathcal{F}_L = \left\{ f : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R} \mid \begin{array}{l} \forall t_1, t_2 \in \mathbb{R}_{\geq 0}, \\ |f(t_1) - f(t_2)| \leq L|t_1 - t_2| \end{array} \right\}, \quad (2)$$

for a known Lipschitz constant $L > 0$. Here, a small L makes $f \in \mathcal{F}_L$ slow-varying, since $|f'| \leq L$ for a differentiable f . See Appendix A for other examples of $\mathcal{F}$.

Search team and its physical constraints: We assume that the search team consists of $N_D \in \mathbb{N}$ mobile sensors (e.g. drones) and $N_C \in \mathbb{N}$ (possibly mobile) charging stations. At any time, the set $\mathcal{C} \subseteq \mathcal{R}$ denotes the current locations of the stations. When the stations are mobile, their movements are restricted to $\mathcal{R}$ (e.g., ground vehicles may be restricted to travel on road networks).

We plan for the movements of the search team in discrete time steps, denoted by $k \in \mathbb{N}$. Here, each discrete time step takes $\Delta t > 0$ time in continuous time.

A *sensor path* is a sequence of cells visited by a sensor, $\mathcal{P} = \{\mathcal{P}_0, \mathcal{P}_1, \dots\}$ where $\mathcal{P}_k \in \mathcal{S} \cup \mathcal{R}$, $\mathcal{P}_{k+1} \in \mathcal{N}_{\text{sensor}}(\mathcal{P}_k)$ for all k and $\mathcal{N}_{\text{sensor}} : \mathcal{S} \cup \mathcal{R} \rightarrow 2^{\mathcal{S} \cup \mathcal{R}}$ as the set of neighboring cells that a sensor at $\mathcal{P}_k$ can visit at $k+1$. Similarly, a *station path* is a sequence of cells visited by a station, $\mathcal{Q} = \{\mathcal{Q}_0, \mathcal{Q}_1, \dots\}$ where $\mathcal{Q}_k \in \mathcal{R}$ and $\mathcal{Q}_{k+1} \in \mathcal{N}_{\text{station}}(\mathcal{Q}_k)$ and $\mathcal{N}_{\text{station}} : \mathcal{R} \rightarrow 2^{\mathcal{R}}$. Observe that $\mathcal{N}_{\text{sensor}}$ and $\mathcal{N}_{\text{station}}$ ensure that the sensors and stations avoid visiting $\mathcal{O}$ and that the stations move within $\mathcal{R}$. We define $\mathcal{N}_{\text{station}}(l) = \{l\}$ for problems where the stations are static. A sensor may move to a neighboring cell at every time step k , but a station moves more slowly and can reach a neighboring cell only within K_{speed} time steps for some $K_{\text{speed}} \in \mathbb{N}$, $K_{\text{speed}} \geq 2$. In other words, the sensors are more agile than the stations.

Additionally, the sensors have limited available energy (e.g., fuel or battery). We model the physical limitation as a constraint on the number of consecutive moves a sensor can execute before recharging at a station. Specifically, let $K_{\text{charge}} \in \mathbb{N}$ be the bound on these moves. Then, starting from energy level $E^{(i)} \in \mathbb{N}_{[0, K_{\text{charge}}]}$, the energy level of sensor i at time step $k+1$ is given by the update rule,

$$E_{k+1}^{(i)} = \begin{cases} K_{\text{charge}}, & \text{sensor } i \text{ meets with a station,} \\ E_k^{(i)} - 1, & \text{otherwise,} \end{cases} \quad (3)$$

and we enforce,

$$E_k^{(i)} \geq 0, \quad \forall k \in \mathbb{N}, \forall i \in \mathbb{N}_{[1, N_D]}. \quad (4)$$

In summary, the physical constraints on the search team are characterized by the tuple $(\mathcal{N}_{\text{sensor}}, \mathcal{N}_{\text{station}}, K_{\text{speed}}, K_{\text{charge}})$. In practice, this abstraction can be instantiated from robot-specific parameters and motion primitives: $\mathcal{N}_{\text{sensor}}, \mathcal{N}_{\text{station}}$ define admissible one-step transitions of the sensing robots and charging stations, K_{speed} captures the mobility of the stations relative to the sensors, and K_{charge} determines the

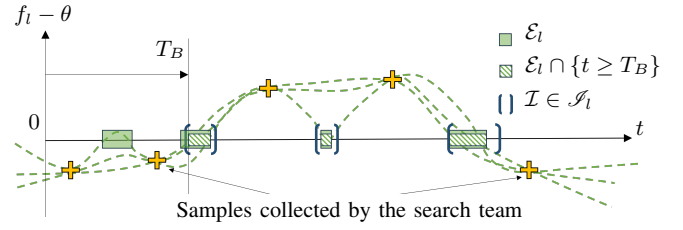


Fig. 2: Illustration of the set of potential critical events $\mathcal{E}_l$, its restriction to events beyond T_B , and an outer-approximating interval collection $\mathcal{I}_l$ for some cell $l \in \mathcal{S}$. The dashed lines illustrate various $f_l \in \mathcal{F}$ that is consistent with the samples (+) collected by the search team, and these functions together characterize $\mathcal{E}_l$ (5). In Problem 1, we define $T_B = K_B \Delta t$.

sensor endurance in discrete planning steps. The resulting framework therefore serves as a high-level planning layer that can be combined with robot-specific estimation, control, and trajectory-tracking modules, as demonstrated in the hardware experiments using drones (see Section VI).

We assume that recharging of the mobile sensors is nearly instantaneous relative to Δt . In practice, this may be realized either through rapid battery swapping within a small fraction of Δt , or by maintaining redundant sensors at the stations that can take over when a depleted sensor arrives. This assumption may be relaxed to account for finite recharging time at the cost of reduced guarantees.

Definition 2. (SEARCH DEPLOYMENT) A search deployment $\mathcal{P} = \{\mathcal{P}^{(i)}, \mathcal{Q}^{(j)}\}_{i \in \mathbb{N}_{[1, N_D]}, j \in \mathbb{N}_{[1, N_C]}}$ is a collection of paths, originating from the current location of each sensor and station on $\mathcal{G}$, that satisfies the physical constraints characterized by the tuple $(\mathcal{N}_{\text{sensor}}, \mathcal{N}_{\text{station}}, K_{\text{speed}}, K_{\text{charge}})$.

By Definition 2, when every sensor and station in the search team is assigned a path from $\mathcal{P}$ based on $\mathcal{N}_{\text{sensor}}, \mathcal{N}_{\text{station}}$, and K_{speed} , the sensors meet a station before exhausting the energy bound K_{charge} by (3) and (4).

Finally, we define the possible set of critical events consistent with the finite data collected by the team. Let $\mathcal{K}_l \subset \mathbb{N}$ denote the time steps when the search team visited cell l , and let $\{v_{l,k}\}_{k \in \mathcal{K}_l}$ be the set of finite samples collected by team at cell l at times $k \in \mathcal{K}_l$. Define $\mathcal{E}_l \subset \mathbb{R}_{\geq 0}$ for each $l \in \mathcal{S}$,

$$\mathcal{E}_l = \bigcup_{f_l \in \mathcal{F}} \{t \mid \forall k \in \mathcal{K}_l, f_l(k\Delta t) = v_{l,k}, f_l(t) = \theta\}, \quad (5)$$

where $\mathcal{E}_l$ contains $\{f_l = \theta\}$. By construction, every critical event at l is included in $\mathcal{E}_l$, but $\mathcal{E}_l$ may include time steps that are not critical events. Since f_l and $\{f_l = \theta\}$ are *a priori* unknown, we focus on (outer-)approximating $\mathcal{E}_l$, the set of *potential* critical events that considers every $f_l \in \mathcal{F}$ consistent with the data collected online $\{v_{l,k}\}_{k \in \mathcal{K}_l}$ (see Figure 2).

We now formalize Research Questions 1 and 2 as follows.

Problem 1. Given a search environment and team with physical constraints, a threshold θ , a function class $\mathcal{F}$, and a mission horizon $K_{\text{mission}} \in \mathbb{N}$, design an iterative algorithm that returns (a small) $K_B \in \mathbb{N}$ and an interval

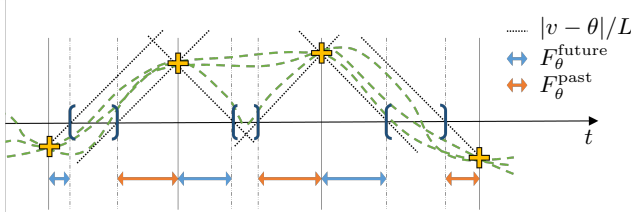


Fig. 3: Illustration of F_θ^{future} (6) and F_θ^{past} (7) for Lipschitz $\mathcal{F}_L$ (2), along with the intervals $\mathcal{I}$ (16) that include all potential critical events that can occur given the samples (+) collected by the search team. The functions F_θ^{future} and F_θ^{past} characterize the earliest and latest times at which a potential critical event close to the current sampling time, and helps in approximating $\mathcal{E}_l$ in (5). See also Figure 2.

collection $\{\mathcal{I}_l\}_{l \in \mathcal{S}}$ that outer-approximates the occurrence set $\mathcal{E}_l \cap \{K_B \Delta t \leq t \leq K_{\text{mission}} \Delta t\}$ for each $l \in \mathcal{S}$.

The burn-in time K_B accounts for the possibility of missing some critical events early in the monitoring task while collecting initial data for the algorithm. In Figure 2, $T_B = K_B \Delta t$.

Problem 2. Provide sufficient conditions under which the algorithm designed for Problem 1 outer-approximates the occurrence set with intervals of bounded length, and each interval is non-trivial (they each include at least one potential critical event).

Problem 2 seeks sufficient conditions under which each interval returned by the proposed algorithm is exhaustive in its characterization of potential critical events, and bounds the approximation error (interval length) in estimating the locations and instants of potential critical events.

III. DATA-DRIVEN SPATIO-TEMPORAL MONITORING

We now address Problem 1. First, we generate intervals that characterize the locations and instants of critical events using $\mathcal{F}$ and the data collected online. Next, we propose an optimization-based strategy to design the search deployment and the interval-based approximation. We implement the optimization problem using an *integer linear program*. To incorporate the online data, we consider a receding horizon implementation of the search deployment. We conclude this section with a summary of the approach.

A. Data-driven inference of critical events and occurrence sets

For a threshold $\theta \in \mathbb{R}$ and any $f \in \mathcal{F}$, we define functions $F_\theta^{\text{future}}, F_\theta^{\text{past}} : \mathbb{R}_{\geq 0} \times \mathbb{R} \rightarrow \mathbb{R}_{\geq 0} \cup \{\infty\}$,

$$F_\theta^{\text{future}}(t, v) \triangleq -t + \inf \left\{ \tau \mid \begin{array}{l} \tau \geq t, \exists f \in \mathcal{F}, \\ f(t) = v, f(\tau) = \theta \end{array} \right\}, \quad (6)$$

$$F_\theta^{\text{past}}(t, v) \triangleq t - \sup \left\{ \tau \mid \begin{array}{l} \tau \leq t, \exists f \in \mathcal{F}, \\ f(t) = v, f(\tau) = \theta \end{array} \right\}. \quad (7)$$

where $\inf$ and $\sup$ of an empty set are $+\infty$ and $-\infty$, respectively. For any $f \in \mathcal{F}$ and $f(t) = v$, $t + F_\theta^{\text{future}}(t, v)$ (and $t - F_\theta^{\text{past}}(t, v)$) is the closest time in future (and past)

relative to t at which f could have become critical, i.e., given (t, v) , it is possible that

$$t + F_\theta^{\text{future}}(t, v), t - F_\theta^{\text{past}}(t, v) \in \{f = \theta\}. \quad (8)$$

The above inference serves as the basis of the following two propositions, whose implications are illustrated in Figure 3 for a Lipschitz function $\mathcal{F}_L$.

Proposition 1. (INFERRING OCCURRENCE SETS) For any $f \in \mathcal{F}$, consider samples $v_1 = f(t_1)$ and $v_2 = f(t_2)$ obtained at times $t_1, t_2 \in \mathbb{R}_{\geq 0}$ with $t_1 < t_2$. Then,

$$\begin{aligned} \{t \in [t_1, t_2] \mid \exists f \in \mathcal{F}, f(t_1) = v_1, f(t_2) = v_2, f(t) = \theta\} \\ \subseteq [t_1 + F_\theta^{\text{future}}(t_1, v_1), t_2 - F_\theta^{\text{past}}(t_2, v_2)]. \end{aligned} \quad (9)$$

From Proposition 1, we can construct $\{\mathcal{I}_l\}_{l \in \mathcal{S}}$ that approximates $\mathcal{E}_l$ (5) using F_θ^{future} and F_θ^{past} .

Proposition 2. (LIPSCHITZ FUNCTIONS) For a Lipschitz function class $\mathcal{F}_L$ (2) with a known Lipschitz constant $L > 0$,

$$\begin{aligned} F_\theta^{\text{future}}(\cdot, v) &= \frac{|\theta - v|}{L}, \\ F_\theta^{\text{past}}(t, v) &= \begin{cases} F_\theta^{\text{future}}(t, v), & F_\theta^{\text{future}}(t, v) \leq t \\ +\infty, & \text{otherwise} \end{cases} \end{aligned}$$

Also, the interval (9) is exact for $\mathcal{F}_L$.

See Appendix B for the proofs of Propositions 1 and 2.

We defined F_θ^{future} and F_θ^{past} based on a single sample (t, v) since it is sufficient for Lipschitz functions (Proposition 2). Single-sample estimators also help reduce the burn-in time K_B in Problem 1 since we obtain non-trivial estimates of $\mathcal{E}_l$ with just two samples for every cell l (Proposition 1). However, they can be overly conservative and may cause large approximation errors in $\mathcal{I}$, since Proposition 1 only guarantees an outer-approximation. We explore more informative F_θ^{future} and F_θ^{past} that can incorporate more past data or higher-order information in Appendix A4.

B. High-level discussion of the approach

We now set up an optimization problem to design a search deployment based on Proposition 1 that satisfies (1) while respecting the physical constraints on the search team. We first provide a high-level discussion of the corresponding optimization problem involved in designing a search deployment $\mathcal{P}$ at each iteration. Then, we incrementally build the interval collection $\{\mathcal{I}_l\}_{l \in \mathcal{S}}$ based on the data collected by executing $\mathcal{P}$.

Design of search deployment: We design a search deployment $\mathcal{P}$ to be executed for the current time step $k_{\text{curr}} \in \mathbb{N}$. We design $\mathcal{P}$ such that the search team visits all cells $l \in \mathcal{S}$ at least once, starting from the current team locations and energy levels. For each l , $k_l^{\text{last}} \in \mathbb{N}$ denotes the set of last time step

where l was visited yielding a sample $v_l^{\text{last}} = f_l(k_l^{\text{last}} \Delta t)$. We generate $\mathcal{P}$ using the following optimization problem,

$$\min_{\mathcal{P}, W} W \quad (10a)$$

$$\text{s. t. Deployment } \mathcal{P} \text{ with } (\mathbb{D}_{k_{\text{curr}}}, \mathbb{C}_{k_{\text{curr}}}, \mathbb{E}_{k_{\text{curr}}}), \quad (10b)$$

$$\forall l \in \mathcal{S}, \exists \mathcal{P} \in \mathcal{P}, \exists K_{\mathcal{P}, l} \in \mathbb{N}_{[1, |\mathcal{P}|]}, l = \mathcal{P}_{K_{\mathcal{P}, l}}, \quad (10c)$$

$$\forall l \in \mathcal{S}, \exists \mathcal{P} \in \mathcal{P}, k_{\text{curr}} + K_{\mathcal{P}, l} \leq k_l^{\text{last}} + K_{\theta}^{\text{deadline}}(k_l^{\text{last}}, v_l^{\text{last}}) + W, \quad (10d)$$

where $(\mathbb{D}_{k_{\text{curr}}}, \mathbb{C}_{k_{\text{curr}}}, \mathbb{E}_{k_{\text{curr}}})$ denote the current search team configuration and energy levels, and $K_{\theta}^{\text{deadline}}(k, v)$ uses (9),

$$\mathbb{D}_{k_{\text{curr}}} \triangleq \{(i, l) : \text{sensor } i \text{ is at cell } l \in \mathcal{S} \cup \mathcal{R} \text{ at } k_{\text{curr}}\}, \quad (11a)$$

$$\mathbb{C}_{k_{\text{curr}}} \triangleq \{(j, l) : \text{station } j \text{ is at cell } l \in \mathcal{R} \text{ at } k_{\text{curr}}\}, \quad (11b)$$

$$\mathbb{E}_{k_{\text{curr}}} \triangleq \{(i, E^{(i)}) : \text{energy of sensor } i \text{ at } k_{\text{curr}} \text{ is } E^{(i)}\}, \quad (11c)$$

$$K_{\theta}^{\text{deadline}}(k, v) \triangleq \left\lfloor \frac{F_{\theta}^{\text{future}}(k \Delta t, v)}{\Delta t} \right\rfloor. \quad (11d)$$

In (10), $W \in \mathbb{N}$ is a (slack) variable that must be minimized.

The constraints (10b) and (10c) require the search team to visit every cell in $\mathcal{S}$ subject to their physical constraints. Specifically, (10b) enforces the energy constraints (4) and the motion constraints on the search team arising from $\mathcal{N}_{\text{sensor}}, \mathcal{N}_{\text{station}}$, and K_{speed} . From (10c), every cell $l \in \mathcal{S}$ is visited by at least one sensor when the team executes $\mathcal{P}$ at time step $k_l^{\text{new}} = k_{\text{curr}} + K_{\mathcal{P}, l}$.

The constraint (10d) is motivated by Proposition 1. We can rule out the occurrence of any critical events at l at all time t ,

$$t_l^{\text{last}} < t < t_l^{\text{last}} + F_{\theta}^{\text{future}}(t_l^{\text{last}}, v_l), \quad (12)$$

with $t_l^{\text{last}} = k_l^{\text{last}} \Delta t$ as the last sample time instant at cell l (see Figure 3). In discrete time, we express (12) as follows for any $k \in \mathbb{N}$,

$$k_l^{\text{last}} < k < k_l^{\text{last}} + \left\lfloor \frac{F_{\theta}^{\text{future}}(k_l^{\text{last}} \Delta t, v_l^{\text{last}})}{\Delta t} \right\rfloor. \quad (13)$$

Consequently, we can rule out the existence of a critical event between $[k_l^{\text{last}} \Delta t, k_l^{\text{new}} \Delta t)$, provided

$$k_l^{\text{new}} \leq k_l^{\text{last}} + \left\lfloor \frac{F_{\theta}^{\text{future}}(k_l^{\text{last}} \Delta t, v_l^{\text{last}})}{\Delta t} \right\rfloor. \quad (14)$$

Since (14) does not involve v^{new} , (14) may be used to schedule k_l^{new} for different cells l , and prioritize visits to different cells in $\mathcal{S}$. However, (14) may not hold for all $l \in \mathcal{S}$ due to the physical constraints on the search team. Therefore, (10d) relaxes (14) for each cell l using a slack variable $W \in \mathbb{N}$, which is subsequently minimized (10a).

The optimal value W^* for any feasible $\mathcal{P}$ to (10),

$$W^*(\mathcal{P}) = \max_{l \in \mathcal{S}} \min_{\mathcal{P} \in \mathcal{P}} \left(\frac{k_{\text{curr}} + K_{\mathcal{P}, l}}{-k_l^{\text{last}} - K_{\theta}^{\text{deadline}}(k_l^{\text{last}}, v_l^{\text{last}})} \right)_+, \quad (15)$$

where $x_+ = \max(0, x)$ is the positive part of any $x \in \mathbb{R}$. We obtain (15) from the observation that (10d) are the only constraints in (10) that involve W .

TABLE I: Decision variables and notation for the integer-programming formulation in Section III used to design the search deployment.

Symbol	Description
$x_{ikl} \in \{0, 1\}$	$x_{ikl} = 1 \Leftrightarrow$ sensor i is at cell l at step k
$y_{jkl} \in \{0, 1\}$	$y_{jkl} = 1 \Leftrightarrow$ station j is at cell l at step k
$z_{ik}^{\text{charge}} \in \{0, 1\}$	$z_{ik}^{\text{charge}} = 1 \Rightarrow$ sensor i recharges at step k
$z_{kl}^{\text{sample}} \in \{0, 1\}$	$z_{kl}^{\text{sample}} = 1 \Rightarrow$ cell l is sampled at step k
$\delta_{kl} \in \{0, 1\}$	$\delta_{kl} = 1 \Leftrightarrow$ cell l remains unvisited before step k
$W \in \mathbb{N}$	Slack variable relaxing the sampling deadlines
N_D, N_C	Number of mobile sensors and charging stations
$\mathbb{D}_{k_{\text{curr}}}, \mathbb{C}_{k_{\text{curr}}}, \mathbb{E}_{k_{\text{curr}}}$	Current sensor/station locations, sensor energy
$\mathcal{N}_{\text{sensor}}(l), \mathcal{N}_{\text{station}}(l)$	Cell-dependent sensor/station-neighborhood map
$K_{\text{speed}}, K_{\text{charge}}$	Speed and energy bounds on the search team
$\mathcal{P}^*, N_H$	Optimal deployment $\mathcal{P}^*$ of horizon N_H
$\mathcal{P}_{\text{init}}, N_{H, \text{init}}$	Initial deployment $\mathcal{P}_{\text{init}}$ of horizon $N_{H, \text{init}}$

Construction of critical event approximations: For any $l \in \mathcal{S}$, let $v_l^{\text{new}} = f_l(k_l^{\text{new}} \Delta t)$ denote the sample collected by the search team at cell l at k_l^{new} . At k_l^{new} , define $\mathcal{I}_l \subset \mathbb{R}$,

$$\mathcal{I}_l \triangleq [k_l^{\text{last}} \Delta t + F_{\theta}^{\text{future}}(k_l^{\text{last}} \Delta t, v_l^{\text{last}}), k_l^{\text{new}} \Delta t - F_{\theta}^{\text{past}}(k_l^{\text{new}} \Delta t, v_l^{\text{new}})]. \quad (16)$$

By Proposition 1, $\mathcal{I}_l$ (see Figure 3) outer-approximates the set of all potential critical events for any $f_l \in \mathcal{F}$ that can occur at cell l between the visits at $k_l^{\text{last}} \Delta t$ and $k_l^{\text{new}} \Delta t$, i.e.,

$$\left\{ t \in [k_l^{\text{last}} \Delta t, k_l^{\text{new}} \Delta t] \mid \begin{array}{l} \exists f \in \mathcal{F}, f(t) = \theta, \\ f(k_l^{\text{last}} \Delta t) = v_l^{\text{last}}, f(k_l^{\text{new}} \Delta t) = v_l^{\text{new}} \end{array} \right\} \subseteq \mathcal{I}_l.$$

Here, we have restricted $f \in \mathcal{F}$ that are consistent with observed data, i.e., $f \in \mathcal{F}$ that yield samples v_l^{last} and v_l^{new} at $k_l^{\text{last}} \Delta t$ and $k_l^{\text{new}} \Delta t$, respectively.

C. Implementation of (10) via an integer linear program

We now provide a collection of integer linear constraints to enforce (10b)–(10d). Let $N_H \in \mathbb{N}$ be the planning horizon that is chosen *a priori* to be long enough to admit a search deployment that satisfy (10b)–(10d). Table I summarizes the decision variables and notation used in this section.

We add the binary variables,

$$x_{ikl}, y_{jkl}, z_{ik}^{\text{charge}}, z_{kl}^{\text{sample}}, \delta_{kl} \in \{0, 1\}, \quad (17)$$

defined for every

$$\text{Sensor index: } i \in \mathcal{D} \triangleq \mathbb{N}_{[1, N_D]}, \quad (18a)$$

$$\text{Station index: } j \in \mathcal{C} \triangleq \mathbb{N}_{[1, N_C]}, \quad (18b)$$

$$\text{Time step: } k \in \mathcal{H} \triangleq \mathbb{N}_{[0, N_H]}, \quad (18c)$$

$$\text{Grid cell location: } l \in \mathcal{S} \cup \mathcal{R}. \quad (18d)$$

To simplify notation, the current time step k_{curr} is $k = 0$.

Motion constraints (10b): We set up the binary variables x so that $x_{ikl} = 1$ if sensor i is at grid cell l at time step k , and

$x_{ikl} = 0$ otherwise. Similarly, $y_{jkl} = 1$ if station j is at grid cell l at time step k , and $y_{jkl} = 0$ otherwise. Then,

$$x_{i0l} = 1, \quad \forall (i, l) \in \mathbb{D}_{k_{\text{curr}}}, \quad (19a)$$

$$y_{j0l} = 1, \quad \forall (j, l) \in \mathbb{C}_{k_{\text{curr}}}, \quad (19b)$$

$$x_{i(k+1)l} \leq \sum_{m \in \mathcal{N}_{\text{sensor}}(l)} x_{ikm}, \quad \begin{aligned} &\forall i \in \mathcal{D}, \forall l \in \mathcal{S} \cup \mathcal{R}, \\ &\forall k \in \mathcal{H} \setminus \{N_H\}, \end{aligned} \quad (19c)$$

$$y_{j(k+1)l} \leq \sum_{m \in \mathcal{N}_{\text{station}, k}(l)} y_{jkm}, \quad \begin{aligned} &\forall j \in \mathcal{C}, \forall l \in \mathcal{R}, \\ &\forall k \in \mathcal{H} \setminus \{N_H\}, \end{aligned} \quad (19d)$$

$$\sum_{i \in \mathcal{D}} x_{ikl} \leq 1 + N_D \mathbb{I}_{\mathcal{R}}(l) \sum_{j \in \mathcal{C}} y_{jkl}, \quad \forall k \in \mathcal{H}, \forall l \in \mathcal{S} \cup \mathcal{R}, \quad (19e)$$

$$\sum_{j \in \mathcal{C}} y_{jkl} \leq 1, \quad \forall k \in \mathcal{H}, \forall l \in \mathcal{R}, \quad (19f)$$

enforce the motion constraints in (10b), with the indicator $\mathbb{I}_{\mathcal{R}}(l) = 1$ if $l \in \mathcal{R}$ and 0 otherwise, and

$$\mathcal{N}_{\text{station}, k}(l) = \begin{cases} \mathcal{N}_{\text{station}}(l), & k \text{ is a multiple of } K_{\text{speed}}, \\ \{l\}, & \text{otherwise.} \end{cases} \quad (20)$$

Constraints (19a) and (19b) are the locations $\mathbb{D}_{k_{\text{curr}}}$ and $\mathbb{C}_{k_{\text{curr}}}$ of the sensors and stations at k_{curr} (11a) and (11b).

Constraints (19c) and (19d) are the physical constraints on the stations and sensors and the agility difference (20). By design, the sensors and stations never visit $\mathcal{O}$, and the stations remain in $\mathcal{R}$.

Constraints (19e) and (19f) are collision avoidance requirements that no two sensors and no two stations occupy the same grid cell. Constraint (19e) is relaxed over $\mathcal{R}$ to allow multiple sensors to visit a station.

Charging constraints (10b): We require sensors to meet stations often to satisfy the energy constraint (4), given their initial energy level $\mathbb{E}_{k_{\text{curr}}}$ (11c) and the energy update rule (3). The constraints,

$$1 + \sum_{j \in \mathcal{C}} y_{jkl} - x_{ikl} \geq z_{ik}^{\text{charge}}, \quad \begin{aligned} &\forall i \in \mathcal{D}, \forall l \in \mathcal{S} \cup \mathcal{R}, \\ &\forall k \in \mathcal{H}. \end{aligned} \quad (21a)$$

$$\sum_{k=0}^{E^{(i)}} z_{ik}^{\text{charge}} \geq 1, \quad \forall i \in \mathcal{D}, \quad (21b)$$

$$\sum_{k'=0}^{K_{\text{charge}}} z_{i \min(k+k', N_H)}^{\text{charge}} \geq 1, \quad \begin{aligned} &\forall i \in \mathcal{D}, \\ &\forall k \in \mathcal{H}, \end{aligned} \quad (21c)$$

enforce the charging constraints in (10b). Specifically, (21a) encodes that the sensor i recharges at time k by meeting with some station in $\mathcal{C}$, whenever $z_{ik}^{\text{charge}} = 1$. It simplifies to $\sum_{j \in \mathcal{C}} y_{jkl} \geq x_{ikl}$ when $z_{ik}^{\text{charge}} = 1$ and becomes redundant when $z_{ik}^{\text{charge}} = 0$.

Constraints (21b) and (21c) encode (4). Here, (21b) ensures that there exists $k \in \mathbb{N}_{[0, E^{(i)}]}$ such that $z_{ik}^{\text{charge}} = 1$ for every sensor i that has an *initial* energy level of $E^{(i)}$ at k_{curr} (11c). Constraint (21c) ensures that, throughout the execution of $\mathcal{P}$, every sensor i performs at most K_{charge} moves depleting energy (4), using a *moving window* constraint to ensure that $z_{ik'}^{\text{charge}} = 1$ for some $k' \in \mathbb{N}_{[k, k+K_{\text{charge}}]}$ for every $k \in \mathbb{N}_{[0, N_H]}$. When drones and stations are allowed to retrace their steps, it is sufficient to enforce (21c) for all $k \leq N_H - \left\lfloor \frac{K_{\text{charge}}}{2} \right\rfloor$ for recursive feasibility.

Coverage constraints (10c): We utilize a *counting mechanism* to characterize $K_{\mathcal{P}, l}$ in (10c) using (17). For every $l \in \mathcal{S}$, we associate a binary sequence $\{\delta_{kl}\}_{k \in \mathcal{H}}$ with every cell $l \in \mathcal{S}$ that is initialized to one, satisfies $\delta_{(k+1)l} \leq \delta_{kl}$ for all k (non-increasing sequence), and switches to zero after the visit time $K_{\mathcal{P}, l}$. Additionally, we encode z_{kl}^{sample} such that $z_{kl}^{\text{sample}} = 1$ implies that the search team visits cell l at time step k , resulting in

$$z_{0l}^{\text{sample}} = 0, \quad \delta_{0l} = 1, \quad (22a)$$

$$\delta_{kl} - z_{(k+1)l}^{\text{sample}} \leq \delta_{(k+1)l} \leq \delta_{kl}, \quad \forall k \in \mathcal{H} \setminus \{N_H\}, \quad (22b)$$

$$\sum_{i \in \mathcal{D}} x_{i(k+1)l} \geq z_{(k+1)l}^{\text{sample}}, \quad \forall k \in \mathcal{H} \setminus \{N_H\}, \quad (22c)$$

$$\sum_{k \in \mathcal{H}} z_{kl}^{\text{sample}} \geq 1. \quad (22d)$$

While (22a)–(22c) encode the desired behavior of δ and z^{sample} , (22d) ensures that every cell in $\mathcal{S}$ is visited at least once within the planning horizon. For each $l \in \mathcal{S}$ and sensor path $\mathcal{P} \in \mathcal{P}$ visiting l , $K_{\mathcal{P}, l}$ in (10c) is characterized by,

$$K_{\mathcal{P}, l} = \sum_{k \in \mathcal{H}} \delta_{kl}. \quad (23)$$

Sampling constraints (10d): Using $K_{\theta}^{\text{deadline}}$ (11d), $K_{\mathcal{P}, l}$ (23), and a subset of data collected so far $(k_l^{\text{last}}, v_l^{\text{last}})_{l \in \mathcal{S}}$, we enforce (10d) as,

$$k_{\text{curr}} + K_{\mathcal{P}, l} \leq k_l^{\text{last}} + K_{\theta}^{\text{deadline}}(k_l^{\text{last}}, v_l^{\text{last}}) + W. \quad (24)$$

As noted in Section III-B, W is an integer (slack) variable that relaxes (12) to prioritize the visitation of cells in $\mathcal{S}$ with a physically-feasible search deployment. Based on the obtained data v_l^{new} , we can construct $\mathcal{I}_l$ as discussed in (16).

Overall integer linear program: Summarizing, we implement (10) as an integer linear program,

$$\begin{aligned} \min. \quad & W \\ \text{s. t.} \quad & W \in \mathbb{N}, \text{ Binary variables (17),} \\ & \text{Motion constraints (19) and (20),} \\ & \text{Charging constraints (21),} \\ & \text{Coverage constraints (22),} \\ & \text{Sampling constraints (24).} \end{aligned} \quad (25)$$

We determine the paths for the search team from the solution $\{x_{ikl}^*, y_{jkl}^*\}$ of (25), which in turn characterizes the search deployment $\mathcal{P}^*$ that solves (10). The total number of binary decision variables in (25) grows linearly with the team size N_D, N_C , the planning horizon N_H , and the environment size $|\mathcal{S} \cup \mathcal{R}|$, see (17) and (18). The solution time for (25) scales exponentially in worst-case, due to the combinatorial nature of the search space. In practice, the runtime is typically dominated by the collision avoidance constraints (19e)–(19f) and the coverage constraints (22). They tightly couple the otherwise independent sensor trajectories, and may increase the computational effort needed to find feasible solutions to (25) with increasing team size or the planning horizon.

In Section V, we show that (25) can typically be solved to optimality for small to medium-sized environments and team sizes in under 1 minute using off-the-shelf solvers. For substantially larger environments or tighter real-time requirements, faster but suboptimal solutions to (25) may be obtained via heuristics in team deployment design [34].

D. Initialization and burn-in time computation

Note that (25) requires $\{(k_l^{\text{last}}, v_l^{\text{last}})\}_{l \in \mathcal{S}}$ to define $K_\theta^{\text{deadline}}$ in (24). At the start of the monitoring task, we use a *data-independent, initial search deployment* $\mathcal{P}_{\text{init}}$ to collect $\{(k_l^{\text{last}}, v_l^{\text{last}})\}_{l \in \mathcal{S}}$. We compute $\mathcal{P}_{\text{init}}$ by,

$$\min_{\mathcal{P}} \left(\max_{l \in \mathcal{S}, \mathcal{P} \in \mathcal{P}} K_{\mathcal{P},l} \right) \quad (26a)$$

$$\text{s. t. } \mathcal{P} \text{ is a search deployment with } (\mathbb{D}_0, \mathbb{C}_0, \mathbb{E}_0), \quad (26b)$$

$$\forall l \in \mathcal{S}, \exists \mathcal{P} \in \mathcal{P}, \exists K_{\mathcal{P},l} \in \mathbb{N}_{[1,|\mathcal{P}|]} \text{ with } l = \mathcal{P}_{K_{\mathcal{P},l}}, \quad (26c)$$

$$\mathcal{P} \text{ terminates with every sensor at a station.} \quad (26d)$$

The solution of (26) $\mathcal{P}_{\text{init}}$ is a search deployment that takes the smallest number of time steps to visit all cells in $\mathcal{S}$. Here, (26b) and (26c) are identical to (10b) and (10c), and (26d) ensures that $\mathcal{P}_{\text{init}}$ terminates with the team recharged for the next search deployment.

Similarly to (25), for a sufficiently long planning horizon N_H , we implement (26) as an integer linear program. Using (21a), we enforce (26d) by fixing z^{charge} at $k = N_H$,

$$z_{iN_H}^{\text{charge}} = 1, \quad \forall i \in \mathcal{D}. \quad (27)$$

Using (23) and similar arguments used to obtain (25), (26) is

$$\begin{aligned} \min. \quad & \max_{l \in \mathcal{S}} (\sum_{k \in \mathcal{H}} \delta_{kl}) \\ \text{s. t.} \quad & (19), (20), (21), (22), (27), \end{aligned} \quad (28)$$

with binary variables (17). Define $N_{H,\text{init}}$ as the smallest planning horizon N_H for which (28) is feasible, with

$$N_{H,\text{init}} \triangleq \max_{\mathcal{P} \in \mathcal{P}_{\text{init}}} |\mathcal{P}|. \quad (29)$$

Since (28) is data-independent, we may compute $\mathcal{P}_{\text{init}}$ and $N_{H,\text{init}}$ *offline*.

We set the burn-in time $K_B = N_{H,\text{init}}$, since our approach may not detect all critical event occurrences during the execution of $\mathcal{P}_{\text{init}}$.

E. Overall approach

Algorithm 1 summarizes our solution to Problem 1.

First, we compute $\mathcal{P}_{\text{init}}$ and $N_{H,\text{init}}$ by solving (28) offline. We obtain the initial data $(k_l^{\text{last}}, v_l^{\text{last}})$ for every cell $l \in \mathcal{S}$ by executing $\mathcal{P}_{\text{init}}$. Next, we solve (25) at each time step (each iteration of Algorithm 1) to obtain $\mathcal{P}_{k_{\text{curr}}}^*$. Unlike $\mathcal{P}_{\text{init}}$ which is executed to its conclusion, we execute only the first move of $\mathcal{P}_{k_{\text{curr}}}^*$, and re-solve (25) based on newly available sample data. We incrementally update a collection of intervals $\mathcal{I}$ that approximates $\{\mathcal{E}_l \cap \{K_B \Delta t \leq t \leq K_{\text{mission}} \Delta t\}\}_{l \in \mathcal{S}}$, where $K_B \triangleq N_{H,\text{init}}$ (see Section III-D).

Recall that $\mathcal{I}_l \in \mathcal{I}_l$ defined in Step 13 is lower bounded by $(k_l^{\text{last}} + K_\theta^{\text{deadline}}(k_l^{\text{last}}, v_l^{\text{last}}))\Delta t$. Given a mission horizon K_{mission} , Step 6 engages the search team for monitoring until all subsequent intervals for every $l \in \mathcal{S}$ are guaranteed to start beyond K_{mission} .

For solving (25) in Step 8, Step 7 chooses a planning horizon $N_H = K_{\text{charge}} + \max_{l \in \mathcal{S}} K_\theta^{\text{deadline}}(k_l^{\text{last}}, v_l^{\text{last}})$. While the considered planning horizon may still be short for the search team to visit all cells in $\mathcal{S}$, we did not observe any infeasibility in our validations.

Algorithm 1 Data-driven spatio-temporal monitoring

Input: Search grid $\mathcal{G} = \mathcal{S} \cup \mathcal{O} \cup \mathcal{R}$, search team locations $(\mathbb{D}, \mathbb{C})$ and constraints $(\mathcal{N}_{\text{sensor}}, \mathcal{N}_{\text{station}}, K_{\text{speed}}, K_{\text{charge}})$, functions F_θ^{future} and F_θ^{past} based on function class $\mathcal{F}$ and threshold θ , mission horizon K_{mission} .

Output: $\{\mathcal{I}_l\}_{l \in \mathcal{S}}$ to solve Problem 1, and burn-in time K_B .

(Offline):

1: Compute $\mathcal{P}_{\text{init}}$ and $N_{H,\text{init}}$ using (28) and (29).

(Online):

2: Set $k_{\text{curr}} \leftarrow 0$, team locations $\mathbb{D}_{k_{\text{curr}}} \leftarrow \mathbb{D}$, $\mathbb{C}_{k_{\text{curr}}} \leftarrow \mathbb{C}$, and sensor energy levels $\mathbb{E}_{k_{\text{curr}}} \leftarrow \{K_{\text{charge}}\}_{i \in \mathcal{D}}$.

3: Execute $\mathcal{P}_{\text{init}}$ which takes $N_{H,\text{init}}$ time steps.

4: For each $l \in \mathcal{S}$, $k_l^{\text{last}} \leftarrow \max_{\mathcal{P} \in \mathcal{P}_{\text{init}}} K_{\mathcal{P},l}$ and $v_l^{\text{last}} \leftarrow f_l(k_l^{\text{last}} \Delta t)$.

5: Set $K_B \leftarrow N_{H,\text{init}}$ and set $k_{\text{curr}} \leftarrow N_{H,\text{init}}$.

6: **while** $\min_{l \in \mathcal{S}} (k_l^{\text{last}} + K_\theta^{\text{deadline}}(k_l^{\text{last}}, v_l^{\text{last}})) \leq K_{\text{mission}}$ **do**

7: Set $N_H \leftarrow K_{\text{charge}} + \max_{l \in \mathcal{S}} K_\theta^{\text{deadline}}(k_l^{\text{last}}, v_l^{\text{last}})$.

8: Compute $\mathcal{P}_{k_{\text{curr}}}^*$ by solving (25) with N_H based on $(\mathbb{D}_{k_{\text{curr}}}, \mathbb{C}_{k_{\text{curr}}}, \mathbb{E}_{k_{\text{curr}}})$, and $\{(k_l^{\text{last}}, v_l^{\text{last}})\}_{l \in \mathcal{S}}$.

9: Move every sensor i to $\mathcal{P}_1^{(i)} \in \mathcal{P}_{k_{\text{curr}}}^*$, and every station j to $\mathcal{Q}_1^{(j)} \in \mathcal{P}_{k_{\text{curr}}}^*$

10: **for** each $l \in \mathcal{S}$ visited by the team $l \in \cup_{i \in \mathcal{D}} \mathcal{P}_1^{(i)}$ **do**

11: Get $v_l^{\text{new}} \leftarrow f_l(k_{\text{curr}} \Delta t)$, and set $k_l^{\text{new}} \leftarrow k_{\text{curr}}$.

12: Compute $\mathcal{I}_l$ using (16).

13: Update $\mathcal{I}_l \leftarrow \mathcal{I}_l \cup \{\mathcal{I}_l\}$, when $\mathcal{I}_l \neq \emptyset$.

14: **end for**

15: Update sets $(\mathbb{D}_{k_{\text{curr}}}, \mathbb{C}_{k_{\text{curr}}}, \mathbb{E}_{k_{\text{curr}}})$ based on the current locations of the search team and (3).

16: Set $k_l^{\text{last}} \leftarrow k_l^{\text{new}}$ and $v_l^{\text{last}} \leftarrow v_l^{\text{new}}$ for every $l \in \mathcal{S}$.

17: Update $k_{\text{curr}} \leftarrow k_{\text{curr}} + 1$.

18: **end while**

Without imposing any additional restrictions on $\mathcal{F}$, there may always exist $f_l \in \mathcal{F}$ that is consistent with the data collected at l so far, such that $K_\theta^{\text{deadline}} = 0$ at all time steps — an *adversarial* choice. From (15), for such an adversarial choice of f_l for every $l \in \mathcal{S}$,

$$W^*(\mathcal{P}) = \max_{l \in \mathcal{S}} \min_{\mathcal{P} \in \mathcal{P}} \left(((k_{\text{curr}} + K_{\mathcal{P},l}) - k_l^{\text{last}})_+ \right). \quad (30)$$

In other words, $W^*(\mathcal{P})$ in (30) is the largest gap, among $l \in \mathcal{S}$, between the current measurement $k_{\text{curr}} + K_{\mathcal{P},l}$ and the past measurement k_l^{last} at $l \in \mathcal{S}$ taken by the team under Algorithm 1, where f_l is chosen adversarially for every $l \in \mathcal{S}$. Typically, $K_\theta^{\text{deadline}} > 0$ for every $l \in \mathcal{S}$ for informative $\mathcal{F}$, which often lowers W^* (see (15)).

Also, Algorithm 1 typically returns disjoint intervals (16) due to non-negative F_θ^{future} and F_θ^{past} . However, two adjacent intervals may touch for some $l \in \mathcal{S}$ at time $k_l^{\text{last}} \Delta t$, when the corresponding new data $v_l^{\text{last}} = \theta$.

Remark 1. We can partially alleviate the computational burden of solving an integer program (25) online in Step 8 by executing a user-specified number of steps (greater than one) of $\mathcal{P}_{k_{\text{curr}}}^*$ in Step 9. We do not pursue such an extension for the sake of brevity.

Remark 2. We briefly discuss several extensions of Algorithm 1 in Appendix A, including other definitions of critical events, mitigating the effect of noisy samples, incorporating higher-order information from $\mathcal{F}$, tightening Proposition 1 using multi-sample inference, and overcoming partial characterization of $\mathcal{F}$, e.g., unknown L in $\mathcal{F}_L$.

IV. THEORETICAL GUARANTEES UNDER SYNCHRONOUS CHARGING CONSTRAINT

In this section, we provide a theoretical analysis of the performance of Algorithm 1 for a set of instances of Problem 1. Specifically, we focus on the setting of Problem 1 where the stations are static ($\mathcal{N}_{\text{station}}(l) = \{l\}$ for any $l \in \mathcal{R}$) and all sensors charge synchronously, i.e., meet the stations at the same time step for charging.

To design search deployments with synchronous charging without specifying the time steps for charging, we introduce an additional constraint in (25) (Step 8 of Algorithm 1) and solve,

$$\begin{aligned} \min. \quad & \text{Objective in (25)} \\ \text{s. t.} \quad & \text{Constraints in (25),} \\ & \forall i \in \mathcal{D}, k \in \mathcal{H}, \quad \sum_{i \in \mathcal{D}} z_{ik}^{\text{charge}} \geq N_D z_{ik}^{\text{charge}}. \end{aligned} \quad (31a)$$

Here, (31a) requires all sensors in $\mathcal{D}$ to get charged at time k , whenever $z_{ik}^{\text{charge}} = 1$ for any $i \in \mathcal{D}$.

For the considered setting, Algorithm 1 admits several theoretical guarantees for a general $\mathcal{F}$, including recursive feasibility, an upper bound on the length of the intervals in $\{\mathcal{I}_l\}_{l \in \mathcal{S}}$, exhaustiveness (the intervals in $\mathcal{I}_l$ together cover $\mathcal{E}_l$), non-triviality (each interval $\mathcal{I}_l$ has a non-empty overlap with $\mathcal{E}_l$), and anytime execution.

Theorem 1. Consider Problem 1 with static stations and synchronous charging and Algorithm 1. Let $\mathcal{P}_{\text{init}}^{\text{no-recharge}}$ in Step 1 be such that $\mathcal{P}_{\text{init}}^{\text{no-recharge}}$ requires the sensor team to recharge only at the end of the deployment, $N_{H,\text{init}}^{\text{no-recharge}} \triangleq \max_{\mathcal{P} \in \mathcal{P}_{\text{init}}^{\text{no-recharge}}} |\mathcal{P}|$. Let all subsequent $\mathcal{P}_{k_{\text{curr}}}$ in Step 8 be computed by (31). Then, the following statements hold true:

- 1) RECURSIVE FEASIBILITY: The integer program (31) solved in Step 8 is always feasible for $N_H \geq K_{\text{charge}} + N_{H,\text{init}}^{\text{no-recharge}}$.
- 2) BOUND ON THE OPTIMUM VALUE OF (31) AND INTERVAL LENGTH: The optimal value of (31) is bounded,

$$0 \leq W^* \leq K_{\text{charge}} + N_{H,\text{init}}^{\text{no-recharge}},$$

and, for each $l \in \mathcal{S}$, $\max_{\mathcal{I} \in \mathcal{I}_l} |\mathcal{I}| \leq (K_{\text{charge}} + N_{H,\text{init}}^{\text{no-recharge}}) \Delta t$.

- 3) EXHAUSTIVENESS AND NON-TRIVIALITY: For each $l \in \mathcal{S}$,

$$\mathcal{E}_l \cap \{K_B \Delta t \leq t \leq K_{\text{mission}} \Delta t\} \subseteq \bigcup_{\mathcal{I} \in \mathcal{I}_l} \mathcal{I}. \quad (32)$$

Additionally, for each $l \in \mathcal{S}$, every interval $\mathcal{I} \in \mathcal{I}_l$ includes a (potential) critical event in $\mathcal{E}_l$, i.e., $\mathcal{I} \cap \mathcal{E}_l \neq \emptyset$.

- 4) ANYTIME EXECUTION: Consider a pre-mature termination of Algorithm 1 at some $k_{\text{curr}} > N_{H,\text{init}}$. Then, for each $l \in \mathcal{S}$,

$$\mathcal{E}_l \cap \{K_B \Delta t \leq t < K_l^{\text{last}} \Delta t\} \subseteq \bigcup_{\mathcal{I} \in \mathcal{I}_l} \mathcal{I}, \quad (33)$$

with $K_l^{\text{last}} \triangleq k_l^{\text{last}} + K_{\theta}^{\text{deadline}}(k_l^{\text{last}}, v_l^{\text{last}})$ for each $l \in \mathcal{S}$.

Proof. Proof of 1): We provide a proof by induction. Since (24) is a soft constraint (due to W), (31) is feasible at k_{curr} , provided

motion constraints including collision avoidance (19), (20), charging constraints (21), coverage constraints (22), and synchronous charging (31a) are together feasible at $k_{\text{curr}} \geq N_{H,\text{init}}^{\text{no-recharge}}$.

Observe that $\mathcal{P}_{\text{init}}^{\text{no-recharge}}$ satisfies all of these constraints by design. However, $\mathcal{P}_{\text{init}}^{\text{no-recharge}}$ requires only $N_{H,\text{init}}^{\text{no-recharge}} < N_H$ steps for execution, and all sensors to be at their corresponding stations at the start of the execution. The proof of 2) consequently focuses on proving the feasibility of a three-phased motion plan — bring the sensors to the stations irrespective of where they are, execute $\mathcal{P}_{\text{init}}^{\text{no-recharge}}$, and then stay at the stations for the rest of the planning horizon N_H .

The base case ($k_{\text{curr}} = N_{H,\text{init}}^{\text{no-recharge}}$) is true by construction, since all sensors start at the station at $k_{\text{curr}} = N_{H,\text{init}}^{\text{no-recharge}}$. We can construct a feasible $\mathcal{P}$ for (31) of length N_H using $\mathcal{P}_{\text{init}}^{\text{no-recharge}}$ by requiring the sensors wait at their respective stations for $k \in [N_{H,\text{init}}^{\text{no-recharge}} + 1, N_H]$.

Assume for induction that (31) has a feasible solution at some $k'_{\text{curr}} > N_{H,\text{init}}^{\text{no-recharge}}$. Then, we must show that (31) also has a feasible solution at $k'_{\text{curr}} + 1$ to complete the proof of 1).

A feasible solution for $k'_{\text{curr}} + 1$ may be constructed by executing the following three-phase solution: first continue executing the search deployment from k'_{curr} until all sensors reach the stations for their first recharge, which takes at most K_{charge} steps due to (21). Since $\mathcal{P}_{k'_{\text{curr}}}$ was designed using (31), sensors may have to charge more than once but all sensors charge synchronously due to (31a). In the second phase, we execute $\mathcal{P}_{\text{init}}^{\text{no-recharge}}$ to ensure satisfaction of the coverage constraints (22), since the first phase solution may not satisfy (22). In the third phase, the sensors wait at the stations for the rest of the planning horizon, similarly to the base case. The solution constructed, which we denote as $\mathcal{P}_{k'_{\text{curr}}+1}^{\text{feas}}$, satisfies all constraints of (31) at $k'_{\text{curr}} + 1$, which completes the proof.

Proof of 2): Let $\mathcal{P}_{k_{\text{curr}}}^*$ and $W^*(\mathcal{P}_{k_{\text{curr}}}^*)$ (15) correspond to the optimal solution to (31) at any $k_{\text{curr}} \in \mathbb{N}$. From the proof of 1), a search deployment $\mathcal{P}$ exists that ensures that every cell can be visited by $k_{\text{curr}} + K_{\text{charge}} + N_{H,\text{init}}^{\text{no-recharge}}$. Consequently, $W^*(\mathcal{P}_{k_{\text{curr}}}^*) \leq K_{\text{charge}} + N_{H,\text{init}}^{\text{no-recharge}}$.

For any k_{curr} and $\mathcal{P} \in \mathcal{P}_{k_{\text{curr}}}^*$, the team takes a new sample at $l \in \mathcal{S}$ at $k_l^{\text{new}} = k_{\text{curr}}$ in Step 11 of Algorithm 1. Then, the new interval $\mathcal{I}$ constructed in Step 13 of Algorithm 1 at k_l^{new} satisfies,

$$\begin{aligned} \frac{|\mathcal{I}|}{\Delta t} &\leq \left(\frac{k_l^{\text{new}} - k_l^{\text{last}} - \frac{F_{\theta}^{\text{future}}(k_l^{\text{last}} \Delta t, v_l^{\text{last}})}{\Delta t}}{-\frac{F_{\theta}^{\text{past}}(k_l^{\text{new}} \Delta t, v_l^{\text{new}})}{\Delta t}} \right) \\ &\leq k_{\text{curr}} - k_l^{\text{last}} \leq W^*(\mathcal{P}_{k_{\text{curr}}}^*). \end{aligned} \quad (34)$$

We obtained the bound in (34) by simplifying the interval definition in Proposition 1 using the observation that $F_{\theta}^{\text{future}}$ and F_{θ}^{past} are non-negative, $k^{\text{new}} = k_{\text{curr}}$, and finally applying (10d). Here, $\mathcal{I}$ covers any possible critical event that may have occurred at l in $[k_l^{\text{last}} \Delta t, k_l^{\text{new}} \Delta t]$. This completes the proof of 2).

Proof of 3): Follows from Proposition 1 and (16) used in Algorithm 1, and that it may take $K_B = N_{H,\text{init}} = N_{H,\text{init}}^{\text{no-recharge}}$ time steps to get the first sample for every $l \in \mathcal{S}$ (see Section III-D).

Proof of 4): Follows from the observation in (12). Specifically, at any time k_{curr} when the algorithm is terminated, we can guarantee that no critical event has occurred at any $l \in \mathcal{S}$ between $k_l^{\text{last}} \Delta t$ and $K_l^{\text{last}} \Delta t$ defined in (33). $\square$

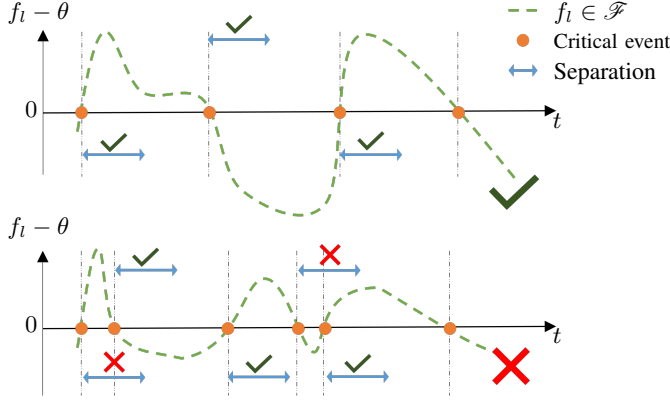


Fig. 4: Illustration of the assumption of sufficiently infrequent critical events associated with the unknown function $f_l \in \mathcal{F}$ for any cell $l \in \mathcal{S}$ used in Corollary 1. The assumption in Corollary 1 allows the function at the top but not the function at the bottom. Specifically, the function in the top has critical events (crossings) that occur at least $K_{\text{charge}} + N_{H,\text{init}}^{\text{no-recharge}}$ spaced apart, while the function at the bottom has some critical events that occur too frequently.

We summarize the implications of Theorem 1. First, Theorem 1 shows that Algorithm 1 is recursively feasible for a sufficiently long planning horizon, which holds even if Algorithm 1 is modified as in Remark 1. Second, Theorem 1 shows that the intervals returned by Algorithm 1 cannot be arbitrarily long. Third, building on Proposition 1, Theorem 1 guarantees that Algorithm 1 does not miss any critical event after the burn-in time $T_B = K_B \Delta t$, and yields an interval collection that satisfies (1) based on the data collected online and prior information available about $\mathcal{F}$. However, Theorem 1 does not exclude the possibility that the outer-approximation $\mathcal{I}_l$ is loose, i.e., some intervals in $\{\mathcal{I}_l\}_{l \in \mathcal{S}}$ do not include any critical event. Fourth, Algorithm 1 shows that pre-mature termination still provides non-trivial solutions to Problem 1. The following corollary provides a sufficient condition under which the outer-approximation $\mathcal{I}_l$ is tight.

Corollary 1. (UNIQUENESS WHEN CRITICAL EVENTS ARE SUFFICIENTLY INFREQUENT) *Consider the setting in Theorem 1. Additionally, let threshold θ be such that for every $f \in \mathcal{F}$, any two critical event occurrences $t_1, t_2 \in \mathcal{E}$ are well-separated, i.e., $|t_1 - t_2| > (K_{\text{charge}} + N_{H,\text{init}}^{\text{no-recharge}}) \Delta t$. Then, Algorithm 1 returns $\{\mathcal{I}_l\}_{l \in \mathcal{S}}$ such that at most one critical event occurs in every $\mathcal{I}_l$.*

Proof. Assume for contradiction, that there is some $l \in \mathcal{S}$ and $\mathcal{I} \in \mathcal{I}_l$ that has two critical events. Then, $|\mathcal{I}| > (K_{\text{charge}} + N_{H,\text{init}}^{\text{no-recharge}}) \Delta t$, which contradicts Theorem 1.2. $\square$

Corollary 1 shows that each interval in $\{\mathcal{I}_l\}_{l \in \mathcal{S}}$ has at most one critical event in slowly-varying environments (see Figure 4). In conjunction with Theorem 1.3, the setting of Corollary 1 guarantees that at most one critical event may occur between every pair of successive samples in $v_l^{\text{last}}, v_l^{\text{new}}$. Thus, we address Problem 2 using Theorem 1 and Corollary 1.

Theorem 1 assumes the existence of $\mathcal{D}_{\text{init}}^{\text{no-recharge}}$ that does not require recharging during the course of deployment. For a

fixed K_{charge} , the assumption in Theorem 1 typically becomes easier to satisfy with teams with larger number of sensors, since $N_{H,\text{init}}$ usually decreases when the number of sensors increases. If there is flexibility in designing K_{charge} , we can first compute $\mathcal{D}_{\text{init}}^{\text{no-recharge}}$ that satisfies Theorem 1 by modifying (28). Specifically, we modify (26) by omitting z^{charge} in (17) and the terminal constraint (27), and replacing the charging constraints (21) by,

$$\sum_{j \in \mathcal{C}} y_{jkl} + x_{ikl} \leq 1, \quad \forall k \in \mathcal{H} \setminus \{N_H\}, \quad (35a)$$

$$\sum_{j \in \mathcal{C}} y_{jkl} \geq x_{ikl}, \quad k = N_H. \quad (35b)$$

Here, (35a) ensures that sensors do not meet any station during the course of the search deployment, and (35b) ensures that all sensors meet the stations at N_H . Then, for any $K_{\text{charge}} \geq N_H$, we satisfy all requirements of Theorem 1.

We expect the performance of Algorithm 1 to degrade gracefully when the assumptions in Theorem 1 are violated. For example, consider the scenario where the local evolution of f_l at some cell l is faster than permitted by the temporal regularity encoded in $\mathcal{F}$, where $f_l \notin \mathcal{F}$ but $f_l \in \mathcal{F}^+$ for some broader function class $\mathcal{F}^+ \supset \mathcal{F}$. Then, Algorithm 1 using $\mathcal{F}$ will generate shorter estimation intervals (16) at l than their counterparts based on $\mathcal{F}^+$, since $F_{\theta}^{\text{future}}, F_{\theta}^{\text{past}}$ according to $\mathcal{F}$ will be larger than their counterparts based on $\mathcal{F}^+$. In such a scenario, there is a higher possibility of missed detections at l , but these events of missed detection are expected to increase gradually as the local evolution of f_l becomes faster.

V. SIMULATION EXPERIMENTS

We now present numerical simulations to validate our approach. The results explore the time-optimality and the scalability of the proposed algorithm when varying sizes of search teams and search environments for different environment functions. We show the efficacy of the proposed approach in detecting critical events in a timely manner, and compare with persistent coverage and Gaussian process-based solutions.

We use an Ubuntu 20.04 LTS workstation with an AMD Ryzen 9 9590X 16-core CPU, an NVIDIA GeForce GTX TITAN Black GPU, and 128GB of RAM for all simulations and experiments. We solve the integer programs in Algorithm 1 using an off-the-shelf solver GUROBI [35].

In the simulations, we considered different environments with $|\mathcal{G}| = \{16, 36, 64, 100\}$ each constituting a $\sqrt{|\mathcal{G}|} \times \sqrt{|\mathcal{G}|}$ grid, including $|\mathcal{O}| = \{0, 2, 8, 28\}$ obstacles, respectively. We select the set of cells on which charging can occur $\mathcal{R}$ as the first column of a tabular $\mathcal{G}$, with $|\mathcal{R}| = \sqrt{|\mathcal{G}|}$ and choose $K_{\text{charge}} = \{10, 20, 30, 40\}$ respectively. We considered the following search team configurations with different numbers of drone sensors (D) and ground charging stations (CS): $\{2D+2CS, 4D+2CS, 6D+3CS, 8D+4CS, 10D+5CS\}$. Since the stations are not static, the considered simulation setting is outside the scope of Theorem 1. Regardless, we set $N_H = K_{\text{charge}} + N_{H,\text{init}}$ and empirically demonstrate that Algorithm 1 retains most of the guarantees of Theorem 1, including recursive feasibility, bounded interval lengths in $\{\mathcal{I}_l\}_{l \in \mathcal{S}}$ and exhaustiveness.

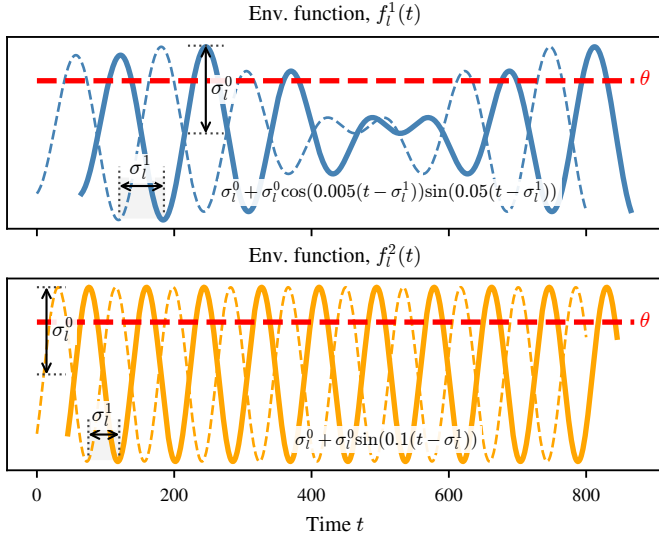


Fig. 5: Parametric function representations for how cell values change. Here, the peak value σ^0 and delay σ^1 are cell-specific parameters.

We chose the mission horizon to be $K_{\text{mission}} = 400$, but the actual missions may last longer (or shorter) than K_{mission} (see Step 6 of Algorithm 1).

We used two separate, parameterized functions to implement the unknown function f_l , with cell-specific tuning parameters $\sigma_l^0, \sigma_l^1 \in \mathbb{R}$ that control the non-uniform delays and peak values (see Figure 5). The functions $f_l^1(\cdot) \in \mathcal{F}_1$ and $f_l^2(\cdot) \in \mathcal{F}_2$ are,

$$f_l^1(t) = \sigma_l^0 + \sigma_l^0 \cos(0.005(t - \sigma_l^1)) \sin(0.05(t - \sigma_l^1)), \quad (36)$$

$$f_l^2(t) = \sigma_l^0 + \sigma_l^0 \sin(0.1(t - \sigma_l^1)). \quad (37)$$

To ensure that f^2 has more frequent peaks and hence critical events than f^1 , we chose $\sigma_l^0 \in [6, 10]$, $\sigma_l^1 = 150$ in (36), and chose $\sigma_l^0 \in [3.5, 5]$, $\sigma_l^1 \in \{10, 11\}$ in (37). While f^1 varies between $[0, 20]$, f^2 takes the values within $[0, 10]$. We set threshold θ for two functions as $\theta = 16$ and 8 respectively. The choice of f, θ may be viewed as an approximation of a periodic flooding scenario, with θ denoting a (rising) water level that characterizes an *ascending critical event*.

Applying Algorithm 1 directly to detect ascending critical events may include extraneous intervals that approximate critical events arising from falling water levels (non-ascending critical events). To specifically detect ascending critical events, we can modify Step 13 in Algorithm 1 to update $\mathcal{J}_l$ only when $\mathcal{I} \neq \emptyset$ and $v_l^{\text{last}} \leq \theta \leq v_l^{\text{new}}$. See Appendix A1 for an extension of Corollary 1, where we can guarantee that the modified Algorithm 1 returns a unique interval for every (potential) ascending critical event.

A. Minimum time to cover the sampling environment

Figure 6 shows the variation in $N_{H, \text{init}}^{\text{no-recharge}}$ with varying environment and team sizes. Recall that $N_{H, \text{init}}^{\text{no-recharge}}$ is the time to cover the search environment in a single charge, and it may be computed offline by modifying (28) with (35) (see

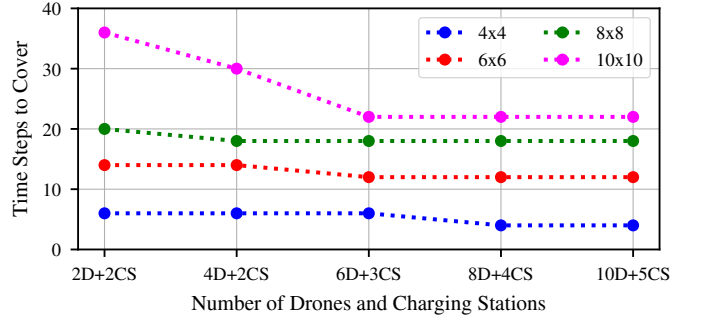


Fig. 6: Variation in the time steps required to cover the search environment in a single charge for different environment and team sizes. As expected, the time needed decreases with increasing team and decreasing environment sizes.

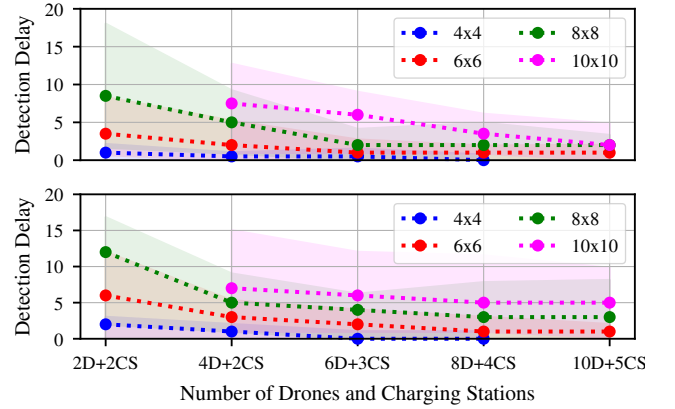


Fig. 7: (0.1, 0.5, 0.9)-quantiles of the detection delays, i.e., the time difference between when a cell becomes critical and when the subsequent sampling occurs, for different environment functions (*top* f_l^1 and *bottom* f_l^2), environment sizes, and search team sizes. The detection delays decrease with increasing team and decreasing environment sizes.

Section III-D). As expected, the time to complete visiting each candidate cell in the environment is smaller for increasing search team sizes and decreasing environment sizes.

B. Performance of the algorithm for varying size of teams and environments under different environment functions

To measure the performance of Algorithm 1, we define *detection delay* as the difference between the time instant at which f_l crosses θ from below and the next time instant at which the cell was sampled. This difference encodes the delay in detecting the critical events. We defer the discussion of exhaustiveness of Algorithm 1 to Section V-C.

Figure 7 shows the variation in the detection delay across the mission horizon and cells for different mission settings and both environment functions. As expected, the detection delays typically decrease with increasing team sizes and decreasing environment sizes. However, fast-changing environments f^2 exhibit a smaller decrease in detection delays, as compared to slow-changing environments f^1 .

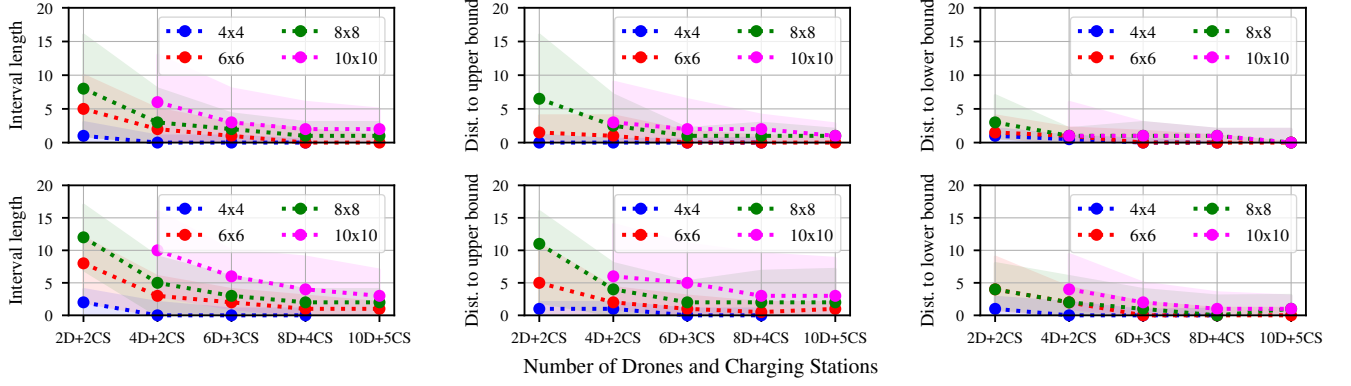


Fig. 8: $(0.1, 0.5, 0.9)$ –quantiles of the lengths of the estimation intervals (*left*) for different environment functions (*top* f_t^1 and *bottom* f_t^2), environment sizes, and search team sizes. The *middle* and *right* plots show the $(0.1, 0.5, 0.9)$ –quantiles of the distance of the critical event times to the upper and lower interval limits respectively for the different settings. The interval lengths decrease with increasing team and decreasing environment sizes.

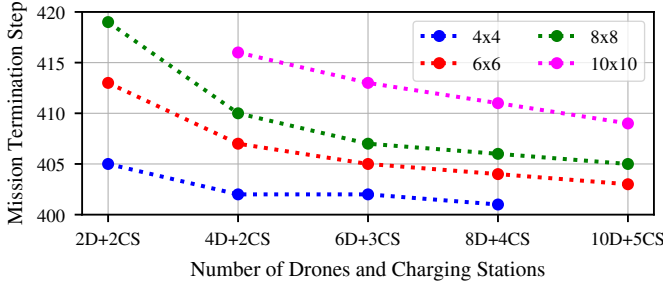


Fig. 9: Termination steps for environment function $f_t^2(t)$ under different environment and search team settings with the nominal mission horizon of $K_{\text{mission}} = 400$.

Figure 8 shows the length of all estimation intervals returned by the algorithm and how close the actual critical event times are to either bounds of the estimation intervals. Recall that the interval lengths may be viewed as the uncertainty (in time) in the occurrence of critical events, where the uncertainty arises from the limited information available to the search team. Similar to Figure 7, the estimation intervals shrink for larger search teams, smaller environments, and slow-changing environments f_t^1 , as expected.

Figure 9 shows the termination time of Algorithm 1 for f_t^2 under varying team and environment sizes. Here, certain cells have critical events close to $K_{\text{mission}}\Delta t$, and consequently Algorithm 1 runs beyond the mission horizon to ensure no critical event is missed. As expected, the number of excess steps needed to complete the monitoring task decreases with increasing team sizes and decreasing environment sizes.

Figure 10 reports the computation times for each problem setting. While the computation times increased with larger team sizes, most of them are under 1 minute.

C. Benefit of adaptivity: Comparison with the data-agnostic, minimum-time coverage-based persistent monitoring

We compare the proposed monitoring algorithm with a conventional persistent monitoring approach that deploys pre-defined paths independently from the sampled data (see [10],

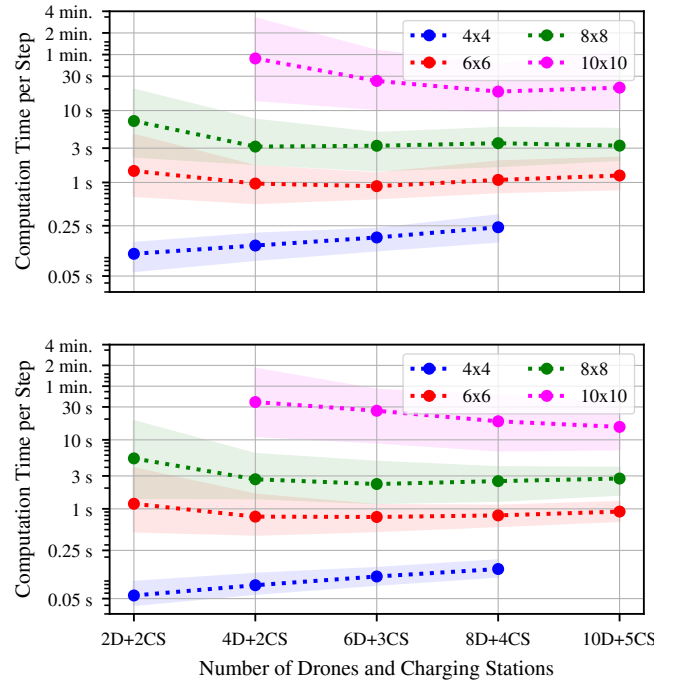


Fig. 10: $(0.1, 0.5, 0.9)$ –quantiles of the computation times for each planning instance for different environment functions (*top* f_t^1 and *bottom* f_t^2), environment sizes, and search team sizes. Vertical axis is in logarithmic scale.

[23]–[25] for similar approaches). Specifically, we compared Algorithm 1 with an approach that repeated $\mathcal{P}_{\text{init}}^{\text{no-recharge}}$, the team deployment that is agnostic to the data collected online and covers the search environment in minimum time under the physical constraints. We refer to this baseline as the *minimum-time coverage* solution.

Tables II and III show that the proposed approach detected *all* critical events successfully, except for 2D+2CS for both f_t^1, f_t^2 with $|\mathcal{G}| = 64$ (where it timed out), and 4D+2CS for f_t^2 with $|\mathcal{G}| = 100$ (where it detected 81% of the critical events).

We did not face any feasibility issues in these simulations, demonstrating the recursive feasibility of Algorithm 1. While the minimum-time coverage solution provided non-trivial detections in all scenarios, it significantly underperformed compared to the proposed approach in scenarios with larger team sizes and environments, and fast-changing environment $f_l^2(\cdot)$. We attribute the higher performance of the proposed approach in such scenarios to its ability to adapt the team deployment decisions based on the data collected online.

Figure 11 demonstrates the differences between the proposed approach and the baseline for the 10×10 search environment with a search team $4D+2CS$. Specifically, we analyze the flight duration for each drone and the median sampling period for the cells. As expected, the proposed approach distributes the load on the search team more equitably compared to the baseline due to its use of an iterative, optimization-based formulation. Additionally, the sampling period of the proposed approach is more dispersed as compared to the baseline, possibly due to its data-driven nature.

Figure 12 shows that the effect of team sizes on the (median) detection delay (see Section V-B) is more significant in the proposed approach as compared to the baseline, especially for slow-changing environment $f_l^1(\cdot)$. In conjunction with Tables II and III, Figure 12 indicates that the proposed approach coordinates the sensors better than the baseline and is more effective in detecting critical events in a timely manner as compared to the baseline.

D. Benefit of interval-based inference: Comparison with Gaussian process-based monitoring

We now compare the conservativeness of the proposed interval-based approach and the two baselines — minimum-time coverage which uses intervals (see Section V-C), and a baseline which is interval-free and uses Gaussian process (GP). Similarly to [14], [17], the GP-based baseline maintains a temporal GP model for each cell $l \in \mathcal{S}$, and prioritizes cells to visit based on the predicted time of the next critical event. Other learning-based or adaptive monitoring methods reviewed in Section I are less directly comparable to our setting because they often exploit spatial-correlation models or statistical priors that we do not assume are available, or they do not jointly address critical-event detection with the hard motion and energy constraints considered here.

GP-based baseline replaces $K_\theta^{\text{deadline}}$ in Steps 6–8 of Algorithm 1 to compute a GP-based deadline $K_{\theta,l}^{\text{GP}}(k)$ for each cell l ,

$$K_{\theta,l}^{\text{GP}}(k) \triangleq \max \left\{ 0, \left\lceil \frac{t_l^{\text{GP}}(k) - k\Delta t}{\Delta t} \right\rceil \right\}, \quad (38)$$

$$t_l^{\text{GP}}(k) \triangleq \inf \{ t \geq k\Delta t \mid \mu_l(t - \Delta t) < \theta \leq \mu_l(t) \}, \quad (39)$$

where $\mu_l(\cdot)$ denotes the posterior mean of the GP for cell l . When the predicate of (39) is empty, we set $t_l^{\text{GP}}(k) = +\infty$. The baseline then uses (25) to design the deployment, where the sampling constraint (24) is replaced with,

$$k_{\text{curr}} + K_{\mathcal{P},l} \leq k_l^{\text{last}} + K_{\theta,l}^{\text{GP}}(k_l^{\text{last}}) + W. \quad (40)$$

We considered a squared exponential kernel for the GPs [14], and tuned the hyperparameters to ensure that f_l^1, f_l^2 lie within the 95%-confidence interval of the GP. We used the challenging setting of monitoring a 10×10 grid with $4D+2CS$ to study the conservativeness of the approaches. Specifically, we investigated the detection delay (see Section V-B) for all approaches, and the distance of the critical event to the corresponding interval boundaries for interval-based approaches (see Figure 8).

Figure 13 shows that the proposed approach and GP-based baseline have moderate to high occurrence of zero-detection delay, i.e., they typically visit a cell at the time step at which it becomes critical. For both f_l^1, f_l^2 , the proposed approach has higher occurrences of *tight detection*—the critical event lies on either interval boundary—than the corresponding GP-based baseline. The proposed approach and GP-based baseline have similar median detection delay, but the proposed approach has much lower, worst-case detection delay than the GP-based baseline (see Table IV and Figure 13). Also, Figure 13 (top) shows that the minimum-time coverage baseline has higher tight detection occurrences as compared to the interval-free GP-based baseline, even though the minimum-time coverage baseline has fewer zero-detection delay occurrences. Together, these observations demonstrate the value of using intervals for inferring spatio-temporal critical events. Additionally, the non-zero occurrences of tight detections for the proposed approach empirically demonstrate that (16) is tight for the considered setting up to a margin of Δt .

VI. HARDWARE EXPERIMENTS

We also validated Algorithm 1 in a physical experiment using Crazyflie drones. We considered the fast-changing environment function $f_l^2(t)$ in (37).

Figure 14 shows the testbed used for the physical experiments. We used a projector to project various colors, representing different water levels corresponding to f^2 , on to the floor of the laboratory. We set up the drones with cameras as sensors to monitor the environment by taking pictures and determining the water level by analyzing the color of the sampled cell in real-time using the OpenCV library. We conducted the experiment in a testing environment of $12\text{ft} \times 12\text{ft}$ motion-capture space identical to the simulation environment with $|\mathcal{G}| = 36$ cells, using an OptiTrack system with 12 cameras for localization. We used CrazySwarm2 package for the low-level control of the drones [36]. We set up the drone-camera configurations to allow near-vertical images. We flew the drones at 0.7 m height to avoid shadows in the environment, and considered static charging stations.

Similarly to the results for the 6×6 search environment in Section V-B, the physical search team achieved a 100% critical-event detection rate and had a median detection delay of 6.0 time steps. Additionally, the online computation time to compute $\mathcal{P}$ was under 1 second in most time steps (see Figure 10, bottom).

VII. CONCLUSIONS

We addressed the two central questions of this work: how to develop a data-driven, spatio-temporal monitoring framework

TABLE II: Percentage of critical events (crossing from below) detected for the environment function $f_l^1(t)$ for the proposed approach and a baseline approach (minimum-time persistent coverage). Here, $\checkmark$ refers to 100%, and T/O refers to time-out, i.e., the computation of (25) did not finish within 1000 seconds.

Env. $\mathcal{G}$	2 Drones + 2 CS		4 Drones + 2 CS		6 Drones + 3 CS		8 Drones + 4 CS		10 Drones + 5 CS	
	Proposed	Baseline	Proposed	Baseline	Proposed	Baseline	Proposed	Baseline	Proposed	Baseline
8×8	95%	98%	$\checkmark$							
10×10	T/O	89%	$\checkmark$	90%	$\checkmark$	96%	$\checkmark$	97%	$\checkmark$	96%

TABLE III: Percentage of critical events (crossing from below) detected for the environment function $f_l^2(t)$ for the proposed approach and a baseline approach (minimum-time persistent coverage). Here, $\checkmark$ refers to 100%, and T/O refers to time-out, i.e., the computation of (25) did not finish within 1000 seconds.

Env. $\mathcal{G}$	2 Drones + 2 CS		4 Drones + 2 CS		6 Drones + 3 CS		8 Drones + 4 CS		10 Drones + 5 CS	
	Proposed	Baseline	Proposed	Baseline	Proposed	Baseline	Proposed	Baseline	Proposed	Baseline
8×8	89%	91%	$\checkmark$							
10×10	T/O	53%	81%	58%	$\checkmark$	85%	$\checkmark$	84%	$\checkmark$	85%

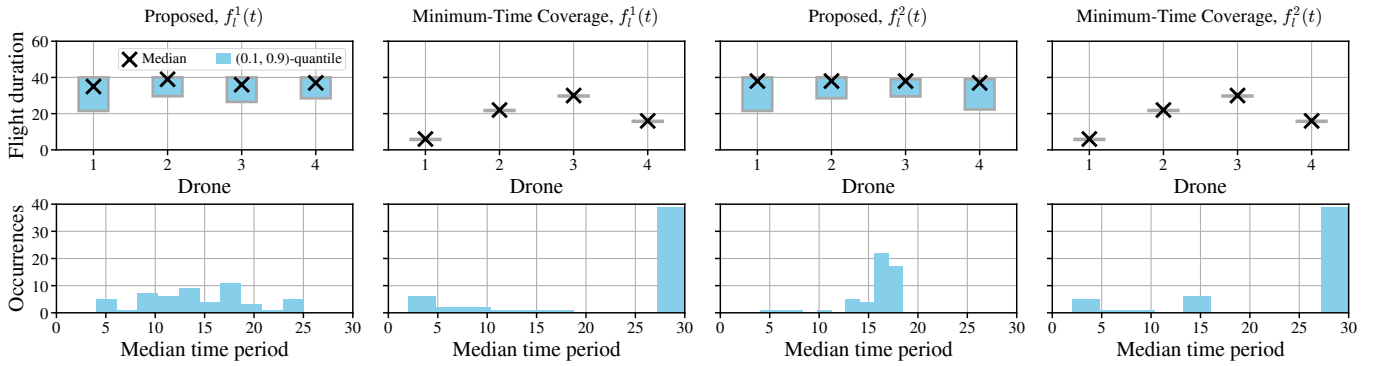


Fig. 11: (0.1, 0.5, 0.9)–quantiles of flight durations between two charging instances (*top*) and median sampling periods (*bottom*) for the cells in a 10×10 search environment. There are no quantiles in the flight duration of the min-time coverage results since they are data-independent and hence, deterministic.

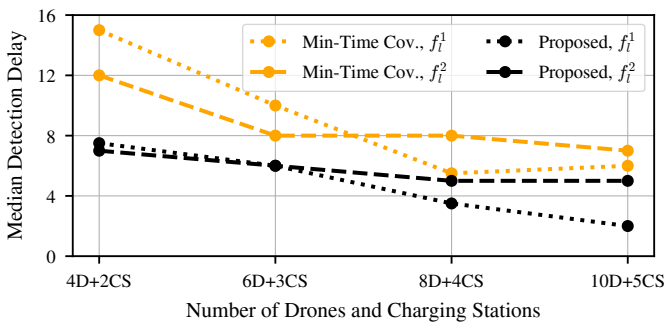


Fig. 12: Comparison of the proposed approach (Algorithm 1) and the baseline (minimum-time coverage) in a 10×10 search environment for f_l^1, f_l^2 .

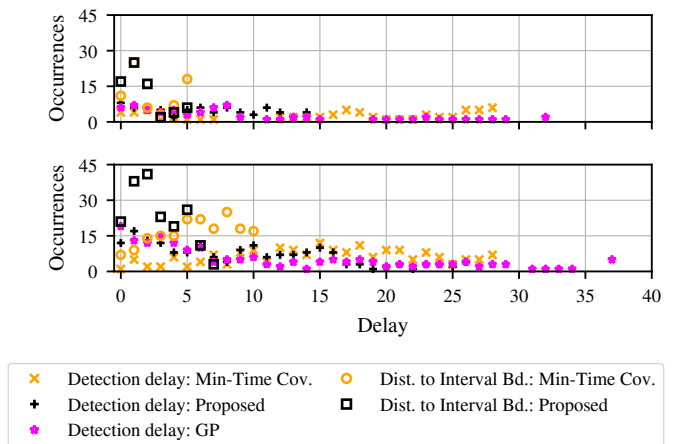


Fig. 13: Detection delay for the proposed approach (Algorithm 1) and the baselines (minimum-time coverage, interval-based baseline and a GP-based, interval-free baseline), and the distance to interval boundaries for the interval-based approaches in a 10×10 search environment with 4D+2CS team for f_l^1 (top) and f_l^2 (bottom).

for heterogeneous, energy-constrained robot teams, and when its interval-valued estimates admit formal guarantees. Our approach combines interval-based inference of critical events with anytime, receding-horizon integer program planning, enabling online deployment updates from newly collected data while respecting motion and charging constraints. Our

TABLE IV: Median (and (0.1, 0.9)-quantiles) of the detection delay and the distance to interval boundaries in a 10×10 search environment with 4D+2CS team. Bold values indicate best performance (lower values).

Env. fun.	Detection delay			Distance to interval boundary	
	GP	Min-Time Cov.	Proposed	Min-Time Cov.	Proposed
f_1^1	7.0 (1.0, 24.1)	17.0 (1.0, 27.0)	6.0 (0.0, 12.0)	1.0 (0.0, 6.0)	1.0 (0.0, 4.0)
f_1^2	6.0 (0.0, 26.0)	15.5 (6.0, 25.0)	8.5 (1.0, 18.9)	6.0 (2.0, 9.0)	2.0 (0.0, 5.0)

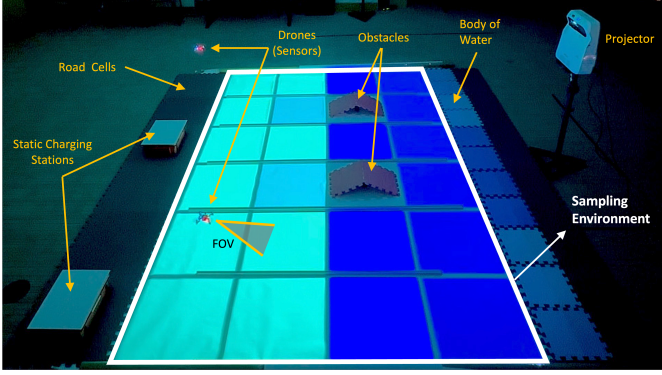


Fig. 14: Snapshot of the testbed used for physical experiments. See <https://youtu.be/Z-v2gqXjmY8> for a video of the physical experiments.

approach achieves lower detection delay than a data-agnostic, interval-based, minimum-time coverage baseline, while maintaining generally high critical-event detection rates (81%–100%). Specifically, in a fast-varying 10×10 environment, the proposed approach achieved 100% detection when using at least six drones and 81% when using four drones, while the two-drone case timed out. A focused comparison with an adaptive, interval-free, GP-based baseline further illustrates the benefit of interval-based inference in improving detection performance and reducing conservativeness. We also establish sufficient conditions for recursive feasibility, bounded interval lengths, exhaustiveness, non-triviality, and anytime execution. Across simulations, most online planning instances were solved in under 1 minute and typically on the order of seconds, and hardware experiments with Crazyflie drones further demonstrate practical feasibility of the approach.

The current framework is limited by two main factors. First, some of the presented theoretical guarantees, for example, exhaustiveness and non-triviality, rely on nearly instantaneous recharging. When charging times are non-negligible, the proposed approach can be extended by either increasing Δt or incorporating charging delays explicitly into the integer linear program (25). These extensions may produce longer intervals, but they do not fundamentally change the modeling framework. Second, while the integer program formulation offers substantial modeling flexibility, it may lead to computational difficulties when dealing with large search environments. Future work will therefore focus on incorporating realistic charging times and developing more scalable heuristics in team deployment design for large-scale environments and team sizes, for example by drawing on ideas from electric-vehicle

routing [34].

REFERENCES

- [1] D. Drew, “Multi-agent systems for search and rescue applications,” *Curr. Rob. Rep.*, vol. 2, pp. 189–200, 2021.
- [2] J. Queralta et al., “Collaborative multi-robot search and rescue: Planning, coordination, perception, and active vision,” *IEEE Access*, vol. 8, pp. 191 617–191 643, 2020.
- [3] M. Dunbabin and L. Marques, “Robots for environmental monitoring: Significant advancements and applications,” *IEEE Rob. Auto. Mag.*, vol. 19, no. 1, pp. 24–39, 2012.
- [4] J. Kim, S. Kim, C. Ju, and H. Son, “Unmanned aerial vehicles in agriculture: A review of perspective of platform, control, and applications,” *IEEE Access*, vol. 7, pp. 105 100–105 115, 2019.
- [5] E. Seraj and M. Gombolay, “Coordinated control of UAVs for human-centered active sensing of wildfires,” in *Amer. Ctrl. Conf.*, 2020, pp. 1845–1852.
- [6] A. Krause and C. Guestrin, “Near-optimal observation selection using submodular functions,” in *AAAI*, vol. 7, 2007, pp. 1650–1654.
- [7] F. Bullo, J. Cortés, and S. Martinez, *Distributed control of robotic networks: a mathematical approach to motion coordination algorithms*. Princeton Univ. Press, 2009, vol. 27.
- [8] M. Coffey and A. Pierson, “Covering dynamic demand with multi-resource heterogeneous teams,” in *IEEE Int’l Conf. Intel. Rob. Syst.*, 2023, pp. 11 127–11 134.
- [9] R. Bajcsy, Y. Aloimonos, and J. K. Tsotsos, “Revisiting active perception,” *Auto. Rob.*, vol. 42, pp. 177–196, 2017.
- [10] A. Kapoutsis, S. Chatzichristofis, and E. Kosmatopoulos, “DARP: Divide areas algorithm for optimal multi-robot coverage path planning,” *J. Int. Rob. Syst.*, vol. 86, pp. 663–680, 2017.
- [11] G. Best, J. Faigl, and R. Fitch, “Online planning for multi-robot active perception with self-organising maps,” *Auto. Rob.*, vol. 42, pp. 715–738, 2018.
- [12] R. Marchant and F. Ramos, “Bayesian optimisation for intelligent environmental monitoring,” in *IEEE Int’l Conf. Intel. Rob. Syst.*, 2012, pp. 2242–2249.
- [13] A. Viseras, R. Wiedemann, C. Manss, L. Magel, J. Mueller, D. Shutin, and L. Merino, “Decentralized multi-agent exploration with online-learning of Gaussian processes,” in *IEEE Int’l Conf. Rob. Autom.*, 2016, pp. 4222–4229.
- [14] K. Jakkala and S. Akella, “Multi-robot informative path planning from regression with sparse Gaussian processes,” in *IEEE Int’l Conf. Rob. Autom.*, 2024, pp. 12 382–12 388.
- [15] A. Munir, A. Dutta, and R. Parasuraman, “Energy-aware coverage planning for heterogeneous multi-robot system,” in *Int’l Symp. Dist. Auto. Rob. Syst.*, 2024, pp. 210–227.
- [16] K. Naveed, D. Agrawal, R. Kumar, and D. Panagou, “Adaptive ergodic search with energy-aware scheduling for persistent multi-robot missions,” *Auto. Rob.*, vol. 49, p. 27, 2025.
- [17] K. Naveed, H. Lee, and D. Panagou, “Multi-robot allocation for information gathering in non-uniform spatiotemporal environments,” in *Amer. Ctrl. Conf.*, 2026, pp. 3481–3488.
- [18] C. Baykal, G. Rosman, S. Clatici, and D. Rus, “Persistent surveillance of events with unknown, time-varying statistics,” in *IEEE Int’l Conf. Rob. Autom.*, 2017, pp. 2682–2689.
- [19] J. Yu, S. Karaman, and D. Rus, “Persistent monitoring of events with stochastic arrivals at multiple stations,” *IEEE Trans. Rob.*, vol. 31, no. 3, pp. 521–535, 2015.
- [20] B. Du, K. Qian, H. Iqbal, C. Claudel, and D. Sun, “Multi-robot dynamical source seeking in unknown environments,” in *IEEE Int’l Conf. Rob. Autom.*, 2021, pp. 9036–9042.
- [21] S. Nayak, M. Greiff, A. Raghunathan, S. Di Cairano, and A. Vinod, “Data-driven monitoring with mobile sensors and charging stations using multi-arm bandits and coordinated motion planners,” in *Amer. Ctrl. Conf.*, 2024, pp. 299–305.
- [22] X. Lin, S. Nayak, S. Di Cairano, and A. Vinod, “Data-driven spatial classification using multi-arm bandits for monitoring with energy-constrained mobile robots,” *IEEE Trans. Ctrl. Syst. Tech.*, pp. 1–9, 2026.
- [23] E. Rolf, D. Fridovich-Keil, M. Simchowitz, B. Recht, and C. Tomlin, “A successive-elimination approach to adaptive robotic source seeking,” *IEEE Trans. Rob.*, vol. 37, no. 1, pp. 34–47, 2020.
- [24] N. Agmon, N. Hazon, and G. Kaminka, “Constructing spanning trees for efficient multi-robot coverage,” in *IEEE Int’l Conf. Rob. Autom.*, 2006, pp. 1698–1703.

- [25] J. Li and P. Zhang, “New approximation algorithms for the rooted budgeted cycle cover problem,” *Theor. Comp. Sci.*, vol. 940, pp. 283–295, 2023.
- [26] K. A. Pant, J. Kim, J. M. Goppert, and I. Hwang, “Energy-constrained resilient multirobot coverage control,” *IEEE Trans. Ctrl. Syst. Tech.*, 2026.
- [27] L. Zhang, J. Zhao, E. Lamon, Y. Wang, and X. Hong, “Energy efficient multi-robot task allocation constrained by time window and precedence,” *IEEE Trans. Autom. Sci. Eng.*, pp. 18 162–18 173, 2025.
- [28] C. Zhang, C. Xu, G. Li, and B. He, “A distributed task allocation approach for multi-UAV persistent monitoring in dynamic environments,” *Sci. Rep.*, vol. 15, p. 6437, 2025.
- [29] S. Gasche, C. Kallies, A. Himmel, and R. Findeisen, “Energy aware and safe path planning for unmanned aircraft systems,” *arXiv preprint arXiv:2504.03271*, 2025.
- [30] A. Buyukkocak, D. Aksaray, and Y. Yazıcıoğlu, “Energy-aware planning of heterogeneous multi-agent systems for serving cooperative tasks with temporal logic specifications,” in *IEEE Int’l Conf. Intel. Rob. Syst.*, 2023, pp. 8659–8665.
- [31] X. Meng, C. Cassandras, X. Sun, and K. Xu, “The price of decentralization in cooperative coverage problems with energy-constrained agents,” *IEEE Trans. Ctrl. Netw. Syst.*, vol. 9, no. 2, pp. 956–965, 2022.
- [32] T. Kundu and I. Saha, “Mobile recharger path planning and recharge scheduling in a multi-robot environment,” in *IEEE Int’l Conf. Intel. Rob. Syst.*, 2021, pp. 3635–3642.
- [33] H. Fouad and G. Beltrame, “Energy autonomy for robot systems with constrained resources,” *IEEE Trans. Rob.*, vol. 38, no. 6, pp. 3675–3693, 2022.
- [34] X. Martin, M. Escoto, A. Guerrero, and A. Juan, “Battery management in electric vehicle routing problems: A review,” *Energies*, vol. 17, no. 5, p. 1141, 2024.
- [35] Gurobi Optimization, LLC, “Gurobi Optimizer Reference Manual,” 2024. [Online]. Available: <https://www.gurobi.com>
- [36] J. Preiss, W. Honig, G. Sukhatme, and N. Ayanian, “Crazyswarm: A large nano-quadcopter swarm,” in *IEEE Int’l Conf. Rob. Autom.*, 2017, pp. 3299–3304.
- [37] Y. Nesterov *et al.*, *Lectures on convex optimization*. Springer, 2018.
- [38] J. Huang, S. Roberts, and J.-P. Calliess, “On the sample complexity of Lipschitz constant estimation,” *Tran. Mach. Learn. Res.*, 2023.

APPENDIX

A. Extensions of Algorithm 1

We outline several generalizations of Algorithm 1, including detecting events where f_l crosses θ from below, handling multiple thresholds, incorporating bounded measurement noise and more informative (e.g., higher-order) samples, and modifying the algorithm when $\mathcal{F}$ is only partially known.

1) *Ascending critical events*: In certain applications, it may be desirable to specifically characterize *ascending* critical events, i.e., critical events occurring at a cell $l \in \mathcal{S}$ at time $t \geq 0$ when f_l hits the threshold θ from *below*, instead of the more inclusive occurrence set considered in Definition 1. Formally, we define $\mathcal{A}_l$ as an *ascending occurrence set* of all such critical events at cell l as,

$$\mathcal{A}_l \triangleq \bigcup_{f_l \in \mathcal{F}} \left\{ t \in \mathbb{R}_{\geq 0} \left| \begin{array}{l} \forall k \in \mathcal{K}, f_l(k\Delta t) = v_{l,k}, \\ f_l(t^-) < f_l(t) = \theta < f_l(t^+) \end{array} \right. \right\}. \quad (41)$$

For some $\delta > 0$, $f_l(t^-) < \theta$ implies that $f_l(\tau) < \theta$ for every τ with $t - \delta \leq \tau < t$, and $f_l(t^+) > \theta$ implies that $f_l(\tau) > \theta$ for every τ with $t < \tau \leq t + \delta$. In (41), $\{v_{l,k}\}_{k \in \mathcal{K}_l}$ denotes the samples collected by the search team as defined in (5).

Algorithm 1, without any modifications, provides an interval-based outer-approximation to $\mathcal{A}_l$, since $\mathcal{A}_l \subseteq \mathcal{E}_l$, and does not differentiate between ascending critical events and non-ascending critical events. Recall that the intermediate value theorem states that a continuous function on a closed

interval takes on every value between its values at the endpoints of that interval. Consequently, we can modify Step 13 of Algorithm 1 and update $\mathcal{A}_l$ only when $\mathcal{I} \neq \emptyset$ and $v_l^{\text{last}}, v_l^{\text{new}}$ are below and above the threshold θ respectively. While such a modification may miss critical events in rapidly-changing environments, no critical events are missed in sufficiently slowly-varying environments as shown in the following proposition.

Proposition 3. (DETECTING ASCENDING CRITICAL EVENTS) *Consider Algorithm 1 with the modified Step 13 — update $\mathcal{A}_l$ with $\mathcal{A}_l \cup \{\mathcal{I}_l\}$, when $\mathcal{I}_l \neq \emptyset$ and $v_l^{\text{last}} \leq \theta \leq v_l^{\text{new}}$. Under the setting in Corollary 1 and for $\mathcal{F}$ containing only continuous functions, Algorithm 1 returns $\{\mathcal{A}_l\}_{l \in \mathcal{S}}$ which satisfies*

$$\mathcal{A}_l \cap \{K_B \Delta t \leq t \leq K_{\text{mission}} \Delta t\} \subseteq \bigcup_{\mathcal{I} \in \mathcal{A}_l} \mathcal{I}, \quad (42)$$

and every interval $\mathcal{I} \in \mathcal{A}_l$ includes a (potential) ascending critical event in $\mathcal{A}_l$, i.e., $\mathcal{I} \cap \mathcal{A}_l \neq \emptyset$. Additionally, the exhaustiveness of the returned interval collection $\mathcal{A}$ holds (like Theorem 1.3).

Proof. We will omit subscript l for notational brevity. For any cell $l \in \mathcal{S}$, we have to consider four possibilities: Case 1 ($v^{\text{last}} > \theta, v^{\text{new}} < \theta$), Case 2 ($v^{\text{last}} > \theta, v^{\text{new}} \geq \theta$), Case 3 ($v^{\text{last}} \leq \theta, v^{\text{new}} < \theta$), and Case 4 ($v^{\text{last}} \leq \theta, v^{\text{new}} \geq \theta$). The modified algorithm updates $\mathcal{A}_l$ only when $\mathcal{I}$ defined by (16) is non-empty and when $(v^{\text{last}}, v^{\text{new}})$ satisfies Case 4. To demonstrate correct detection of all ascending events, we need to show that Cases 1, 2, and 3 do not have any ascending critical events, and critical events (if any) in Case 4 are ascending critical events. We focus on critical events between any pair of samples v^{last} (excluding) and v^{new} (including), since critical events at v^{last} may be handled by the preceding pair.

Consider Case 1. Assume, for contradiction, that an ascending critical event occurred at some $v^{\text{last}} < t \leq v^{\text{new}}$. Then, by (41), $f_l(t) = \theta$ and $f_l(\tau) < \theta$ for some $v^{\text{last}} < \tau < t$. Then, by the intermediate value theorem and the assumption that $\mathcal{F}$ contains only continuous functions, there exists a critical event t' with $f_l(t') = \theta$ and $v^{\text{last}} < t' \leq \tau$, since $f_l(\tau) < \theta < f_l(v^{\text{last}})$. Since $t' \neq t$, we have two critical events t, t' in the time interval $[v^{\text{last}}, v^{\text{new}}]$, which contradicts Corollary 1.

Similar arguments rule out ascending critical events in Cases 2 and 3. Case 4 follows from the intermediate value theorem. $\square$

2) *Monitoring for exiting a pre-defined interval*: In certain applications, it may be desirable to define a critical event as the time instant where $f_l \in \mathcal{F}$ exits a pre-defined interval $\Theta = [\theta^{\text{down}}, \theta^{\text{up}}] \subset \mathbb{R}$. Similarly to (5) and (41), we define the occurrence set for such critical events given the sample data $\{v_{l,k}\}_{k \in \mathcal{K}}$ as,

$$\mathcal{E}_{l,\Theta} \triangleq \bigcup_{f_l \in \mathcal{F}} \left\{ t \in \mathbb{R}_{\geq 0} \left| \begin{array}{l} \forall k \in \mathcal{K}, f_l(k\Delta t) = v_{l,k}, \\ f_l(t) \in \{\theta^{\text{up}}, \theta^{\text{down}}\} \end{array} \right. \right\}. \quad (43)$$

Using (6) and (7), we define $F_{\Theta}^{\text{future}}$ and F_{Θ}^{past} for inferring

$\mathcal{E}_{l,\Theta}$ as follows,

$$F_{\Theta}^{\text{future}}(t, v) \triangleq -t + \inf \left\{ \tau \left| \begin{array}{l} \tau \geq t, \exists f \in \mathcal{F}, \\ f(t) = v, f(\tau) \in \{\theta^{\text{up}}, \theta^{\text{down}}\} \end{array} \right. \right\},$$

$$= \min (F_{\theta^{\text{up}}}^{\text{future}}(t, v), F_{\theta^{\text{down}}}^{\text{future}}(t, v)), \quad (44)$$

$$F_{\Theta}^{\text{past}}(t, v) \triangleq t - \sup \left\{ \tau \left| \begin{array}{l} \tau \leq t, \exists f \in \mathcal{F}, \\ f(t) = v, f(\tau) \in \{\theta^{\text{up}}, \theta^{\text{down}}\} \end{array} \right. \right\},$$

$$= \min (F_{\theta^{\text{up}}}^{\text{past}}(t, v), F_{\theta^{\text{down}}}^{\text{past}}(t, v)). \quad (45)$$

Similarly to $F_{\theta}^{\text{future}}$ and F_{θ}^{past} in (6) and (7), $F_{\Theta}^{\text{future}}$ and F_{Θ}^{past} are the closest relative time instants to a potential critical event. For the Lipschitz $\mathcal{F}_L$, $F_{\Theta}^{\text{future}}(\cdot, v)$ and F_{Θ}^{past} have closed-form expressions based on Proposition 2.

Algorithm 1 provides an interval approximation of $\mathcal{E}_{l,\Theta}$ by considering both thresholds, when we use $F_{\Theta}^{\text{future}}$ and F_{Θ}^{past} instead of $F_{\theta}^{\text{future}}$ and F_{θ}^{past} . Specifically, we use (44) and (45) to modify the definition of $K_{\theta}^{\text{deadline}}$ (11d), and obtain an outer-approximating interval $\mathcal{I}_l$ using (16).

3) *Noisy sampling*: Throughout the discussion so far, we have assumed perfect sensing. We now extend Algorithm 1 to handle the case where each visit to a cell $l \in \mathcal{S}$ returns a noisy sample of $f_l \in \mathcal{F}$.

Specifically, a visit to cell $l \in \mathcal{S}$ by the search team at the step k_l^{new} produces a noisy measurement of f_l ,

$$v_{l,\gamma}^{\text{new}} = f_l(k_l^{\text{new}} \Delta t) + \gamma_{k_l^{\text{new}}}, \quad (46)$$

where $\gamma \in \Gamma$ is an unknown (possibly time-varying) sensor noise corrupting the measurement of f_l . We assume that Γ is a compact interval in $\mathbb{R}$. Similarly to Problem 1, we consider the problem of approximating the occurrence set $\mathcal{E}_l \cap \{K_B \Delta t \leq t \leq K_{\text{mission}} \Delta t\}$ for each $l \in \mathcal{S}$.

In order to apply Algorithm 1 for the noisy setting (46), we define $F_{\theta, \text{noisy}}^{\text{future}}$ and $F_{\theta, \text{noisy}}^{\text{past}}$ as follows,

$$F_{\theta, \text{noisy}}^{\text{future}}(t, v) \triangleq -t + \inf \left\{ \tau \left| \begin{array}{l} \tau \geq t, \exists f \in \mathcal{F}, \exists \gamma_1, \gamma_2 \in \Gamma, \\ f(t) + \gamma_1 = v, f(\tau) + \gamma_2 = \theta \end{array} \right. \right\}, \quad (47)$$

$$= \inf_{\gamma_1, \gamma_2 \in \Gamma} F_{\theta - \gamma_2}^{\text{future}}(t, v - \gamma_1) \quad (48)$$

$$F_{\theta, \text{noisy}}^{\text{past}}(t, v) \triangleq t - \sup \left\{ \tau \left| \begin{array}{l} \tau \leq t, \exists f \in \mathcal{F}, \exists \gamma_1, \gamma_2 \in \Gamma, \\ f(t) + \gamma_1 = v, f(\tau) + \gamma_2 = \theta \end{array} \right. \right\}, \quad (49)$$

$$= \inf_{\gamma_1, \gamma_2 \in \Gamma} F_{\theta - \gamma_2}^{\text{past}}(t, v - \gamma_1). \quad (50)$$

Here, (47) and (49) incorporates the corruption due to the sensor noise into (6) and (7). Then, we cast $F_{\theta, \text{noisy}}^{\text{future}}$ and $F_{\theta, \text{noisy}}^{\text{past}}$ in terms of modified versions $F_{\theta}^{\text{future}}$ and F_{θ}^{past} by rearranging the relevant terms. Informally, $F_{\theta, \text{noisy}}^{\text{future}}$ and $F_{\theta, \text{noisy}}^{\text{past}}$ change the threshold θ and the last observed value v to be used in $F_{\theta}^{\text{future}}$ and F_{θ}^{past} based on the noise limits such that the gap between t and the closest possible occurrence of a critical event is minimized.

Let $\Gamma = [\gamma^-, \gamma^+]$ for some finite scalars $\gamma^- \leq \gamma^+$, and define the interval $\Gamma_{\oplus}$ as follows,

$$\Gamma_{\oplus} = \{\gamma_1 - \gamma_2 | \gamma_1, \gamma_2 \in \Gamma\} = [\gamma^- - \gamma^+, \gamma^+ - \gamma^-]. \quad (51)$$

Similarly to Proposition 2, (48) and (50) have closed-form expressions for Lipschitz $\mathcal{F}_L$. Specifically,

$$F_{\theta, \text{noisy}}^{\text{future}}(t, v) = \inf_{\gamma_1, \gamma_2 \in \Gamma} \frac{|\theta - \gamma_2 - v + \gamma_1|}{L} = \inf_{\gamma \in \Gamma_{\oplus}} \frac{|\theta - v + \gamma|}{L}$$

$$= \begin{cases} 0, & v - \theta \in \Gamma_{\oplus}, \\ \min \left(\frac{|\theta - v + \gamma^- - \gamma^+|}{L}, \frac{|\theta - v + \gamma^+ - \gamma^-|}{L} \right), & \text{otherwise.} \end{cases}$$

Here, we used the closed-form expression in Proposition 2 to characterize (48) and then used simple calculus arguments to obtain a closed-form solution to $\inf_{\gamma \in \Gamma_{\oplus}} |\gamma + x|$ for a fixed x . Similar arguments can be used to derive a closed-form expression for $F_{\theta, \text{noisy}}^{\text{past}}$ as well.

Algorithm 1 provides an interval approximation of the occurrence set in Problem 1 despite the noisy samples (46) when using $F_{\theta, \text{noisy}}^{\text{future}}$ and $F_{\theta, \text{noisy}}^{\text{past}}$ in place of $F_{\theta}^{\text{future}}$ and F_{θ}^{past} . Specifically, we use (48) and (50) to modify the definition of $K_{\theta}^{\text{deadline}}$ (11d), and to outer-approximate $\mathcal{I}_l$ using (16).

4) *Higher-order information and multi-sample inference*: Throughout the discussion so far, we have considered access to only zero-order information and used the last sample value when inferring critical events. However, our approach can also be applied when we have access to higher-order information and/or we use the data earlier than the last sample during inference. In such settings, our approach may require additional computational effort and memory, but it may yield more accurate intervals owing to the additional restrictions placed on the function class $\mathcal{F}$.

As a concrete example, consider $\mathcal{F}_G$, the set of twice-differentiable functions with bounded curvature $G > 0$,

$$\mathcal{F}_G = \left\{ f : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R} \left| \begin{array}{l} f \text{ is twice-differentiable,} \\ \forall t \in \mathbb{R}_{\geq 0}, |f''(t)| \leq G \end{array} \right. \right\}.$$

We assume access to the first-order information of the form $(k_{\text{curr}}, v_{k_{\text{curr}}}, g_{k_{\text{curr}}})$ whenever the search team samples a cell l at time step k_{curr} with $v_{k_{\text{curr}}} = f_l(k_{\text{curr}} \Delta t)$, $g_{k_{\text{curr}}} = f'_l(k_{\text{curr}} \Delta t)$, and seek to use a finite history of samples $\mathcal{V}_l = \{(k, v_k, g_k)\}_{k \in \mathcal{K}}$, $\mathcal{K} \subset \mathbb{N}$ when designing $\mathcal{P}$ using (25).

To incorporate the additional information available, we define $F_{\theta}^{\text{future}}$ (6) and F_{θ}^{past} (7) as,

$$F_{\theta}^{\text{future}}(t, \mathcal{V}_l) \triangleq -t + \inf \left\{ \tau \left| \begin{array}{l} \tau \geq t, \exists f \in \mathcal{F}_G, f(\tau) = \theta \\ \forall k \in \mathcal{V}_l, f(k \Delta t) = v_k, f'(k \Delta t) = g_k, \end{array} \right. \right\}, \quad (52a)$$

$$F_{\theta}^{\text{past}}(t, \mathcal{V}_l) \triangleq t - \sup \left\{ \tau \left| \begin{array}{l} \tau \leq t, \exists f \in \mathcal{F}_G, f(\tau) = \theta \\ \forall k \in \mathcal{V}_l, f(k \Delta t) = v_k, f'(k \Delta t) = g_k, \end{array} \right. \right\}. \quad (52b)$$

While (52) in their current form are not tractable, we can use properties of $\mathcal{F}_G$ to obtain tractable lower-bounds. From [37],

$$|f(\tau) - f(t) - f'(t)(\tau - t)| \leq \frac{G}{2}(\tau - t)^2, \quad (53)$$

for any $f \in \mathcal{F}_G$ and $t, \tau \in \mathbb{R}_{\geq 0}$. Using (53), we obtain a lower bound for $F_\theta^{\text{future}}(t, \mathcal{V}_l)$ and $F_\theta^{\text{past}}(t, \mathcal{V}_l)$,

$$\begin{aligned} F_\theta^{\text{future}}(t, \mathcal{V}_l) \geq \quad & \min. \quad \tau - t \\ & \text{s. t.} \quad \tau \geq t, \\ & \forall (k, v, g) \in \mathcal{V}_l, \quad |\theta - v - g(\tau - k\Delta t)| \\ & \leq \frac{G}{2}(\tau - k\Delta t)^2 \\ F_\theta^{\text{past}}(t, \mathcal{V}_l) \geq \quad & \min. \quad t - \tau \\ & \text{s. t.} \quad \tau \leq t, \\ & \forall (k, v, g) \in \mathcal{V}_l, \quad |\theta - v - g(\tau - k\Delta t)| \\ & \leq \frac{G}{2}(\tau - k\Delta t)^2 \end{aligned} \quad (54)$$

Algorithm 1 solves Problem 1 for the considered setting using (54), modifying the definition of $K_\theta^{\text{deadline}}$ (11d), and obtaining an outer-approximating interval $\mathcal{I}_l$ using (16).

5) *Partial characterization of the function class $\mathcal{F}$* : An important setting in monitoring tasks is where only partial information of $\mathcal{F}$ is available *a priori*. For example, expert domain knowledge may inform us that f_l is Lipschitz (2), but we may not know the true Lipschitz constant L *a priori*. Existing literature provides several plug-in estimators, including the so-called Strongin estimators as well as least-squares based batch estimators of the Lipschitz constant (see [38] for an overview).

To handle the limited prior information on $\mathcal{F}$, we use a two-stage approach. First, we repeatedly execute $\mathcal{P}_{\text{init}}$ to estimate the unknown parameters, increasing K_B in Algorithm 1 (Steps 3–5) until the estimates are reliable; then we run the remainder of Algorithm 1 using these estimated parameters.

B. Proofs of the theoretical results

Proof of Proposition 1. The proof follows from the definition of F_θ^{future} and F_θ^{past} in (6) and (7), respectively. Let t_1, v_1, t_2, v_2 be such that $t_1 \leq t_2$ and $[t_1, t_2] \cap \mathcal{E}_l \neq \emptyset$. Consider any $t_c \in [t_1, t_2] \cap \mathcal{E}_l$. Then, $t_c \geq t_1 + F_\theta^{\text{future}}(t_1, v_1)$ and $t_c \leq t_2 - F_\theta^{\text{past}}(t_2, v_2)$. In other words, $t_c \in [t_1 + F_\theta^{\text{future}}(t_1, v_1), t_2 - F_\theta^{\text{past}}(t_2, v_2)]$, which completes the proof. $\square$

Proof of Proposition 2. The proof follows from (2). Note $f(t + F_\theta^{\text{future}}(t, v)) = \theta$ by (6). Moreover, since $f(\cdot)$ is Lipschitz, $|f(t + F_\theta^{\text{future}}(t, v)) - f(t)| \leq LF_\theta^{\text{future}}(t, v)$, which yields $F_\theta^{\text{future}}(t, v) \geq \frac{|\theta - v|}{L}$. Then, $F_\theta^{\text{future}}(t, v) = \frac{|\theta - v|}{L}$ from (6). Moreover, due to symmetry in Lipschitz $\mathcal{F}$, $F_\theta^{\text{past}}(t, v)$ is equal to $F_\theta^{\text{future}}(t, v)$ when $F_\theta^{\text{future}}(t, v) \leq t$, and it is undefined otherwise as time is defined in $\mathbb{R}_{\geq 0}$. The exactness of the interval inferred in Proposition 1 for Lipschitz $\mathcal{F}_L$ also follows from the uniqueness of the lower bound $\frac{|\theta - v|}{L}$ owing to linearity. $\square$



for autonomous multi-robot systems.

Ali Tevfik Buyukkocak received his B.S. and M.S. degrees in Aerospace Engineering from Middle East Technical University, Turkey in 2015 and 2018, respectively, and the Ph.D. degree in Aerospace Engineering and Mechanics from the University of Minnesota, Twin Cities, Minneapolis, MN, USA in 2024. He is currently a postdoctoral researcher at the Department of Electrical and Computer Engineering at Northeastern University, Boston, MA, USA. His research focuses on developing scalable, resilient, and provably correct planning and control algorithms



strategies for complex mechatronic systems, in automotive, factory automation, transportation systems, and aerospace. His research interests include model predictive control, constrained control, path planning, hybrid systems, optimization, and particle filtering. He has authored/coauthored more than 200 peer-reviewed papers in journals and conference proceedings and more than 100 patents. Dr. Di Cairano was the Chair of the IEEE CSS Technical Committee on Automotive Controls and of the IEEE CSS Standing Committee on Standards. He was the inaugural Chair of the IEEE CCTA Editorial Board and was an Associate Editor of the IEEE TRANS. CONTROL SYSTEMS TECHNOLOGY.

Stefano Di Cairano (Fellow, IEEE) received the Master's (Laurea) and the Ph.D. degrees in information engineering in 2004 and 2008, respectively, from the University of Siena, Italy. During 2008–2011, he was with Powertrain Control R&A, Ford Research and Advanced Engineering, Dearborn, MI, USA. Since 2011, he is with Mitsubishi Electric Research Laboratories, Cambridge, MA, USA, where he is currently a Deputy Director, and a Distinguished Research Scientist. His research focuses on optimization-based control and decision-making



was awarded the Best Student Paper Award in the 2017 ACM Hybrid Systems: Computation and Control Conference, and was the finalist for the Best Paper Award in the 2018 ACM Hybrid Systems: Computation and Control Conference. He is a member of the IEEE CSS Conference Editorial Board.

Abraham P. Vinod (Senior Member, IEEE) received the B.Tech. and M.Tech. degrees in electrical engineering from the Indian Institute of Technology, Madras (IITM), Chennai, India, in 2014 and the Ph.D. degree in electrical engineering from the University of New Mexico, Albuquerque, NM, USA, in 2018. He is currently working as a Principal Research Scientist at Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA, USA. His research interests are in the areas of constrained control, multi-agent systems, and learning. Dr. Vinod