A Hierarchical Approach with Crisis Mitigation for Multi-Robot Spatio-Temporal Restoration

Amel Nestor Docena
Dartmouth College
Hanover, USA
amel.nestor.docena.gr@dartmouth.edu

Alberto Quattrini Li
Dartmouth College
Hanover, USA
alberto.quattrini.li@dartmouth.edu

ABSTRACT

Recent research has addressed the problem of spatio-temporal areas restoration for a single robot with limited battery life, deployed in a known environment, focusing on optimally preserving decaying area properties above a critical threshold. This paper extends this NP-hard problem to the multi-robot case, a challenge that remains open in the literature. We further enhance the problem's realism by incorporating spatial correlations between areas and introducing "surge periods" – intervals where widespread property decay is predicted to overwhelm individual robots. To solve this problem in real-time, we devise a hierarchical approach. A central planner creates jurisdictions of areas by clustering them based on relevant attributes, and then assign each jurisdiction to a "home" robot. Each robot then locally manages its jurisdiction using a state-of-the-art greedy heuristic. To handle surge periods, we design a dynamic replanning method based on discounted losses. This method (1) identifies jurisdictions where home robots require assistance, (2) creates temporary "surge clusters" from highly urgent areas, (3) forms a team of surge robots to service these clusters, and (4) determines when these robots should return to their home jurisdictions. Our extensive experiments using a robotic simulator show that our method operates in real-time, outperforming state-of-the-art methods adapted for this problem, even during surge periods. This work presents the first centralized framework with dynamic re-planning to efficiently solve the multi-robot persistent restoration problem, with potential positive impact for robotics in environmental monitoring and other critical applications.

KEYWORDS

persistent preservation; environment monitoring; multi-robot task allocation; hierarchical clustering with replanning; crisis mitigation

ACM Reference Format:
Amel Nestor Docena and Alberto Quattrini Li. 2026. A Hierarchical Approach with Crisis Mitigation for Multi-Robot Spatio-Temporal Restoration . In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 9 pages. https://doi.org/10.65109/UYQK9511

1 INTRODUCTION

This paper addresses the *Multi-Robot Spatio-Temporal Area Restoration (MR-STAR)* problem (Fig. 1), where a team of autonomous mobile robots must maintain a set of designated areas. In each area, a metric of interest, such as air quality, naturally degrades over time due to varying external factors (e.g., human activity). The robots, constrained by limited battery life, must efficiently travel to these areas to restore the metric to a safe level, with the ability to recharge at a central station. The challenge is amplified by two realistic factors: 1) spatial correlations, where the values in one area exhibit a statistical dependency on the values of neighboring areas, and 2) "surge periods", where restoration demand from specific areas can temporarily overwhelm the robots. The objective is to devise a control strategy that minimizes the total time any area is in an unsafe state (i.e., below a critical threshold). Key applications include surface disinfection [19] and precision weed control in agriculture [31].

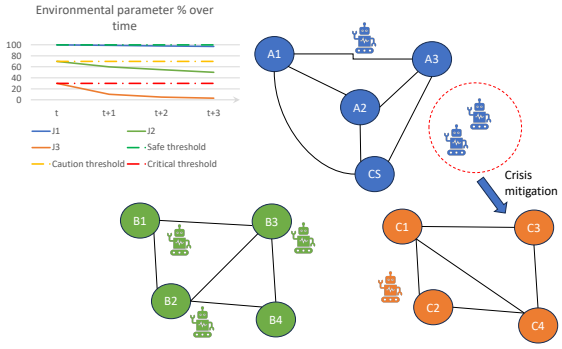


Figure 1: In a known environment where spatio-temporal properties are decaying with varying rates, we propose a hierarchical clustering approach with re-planning to persistently restore areas from falling below a critical threshold: assigning robots to home jurisdictions based on spatio-temporal properties, and potentially re-assigning them to other jurisdictions temporarily to help restore areas in case of crisis.

The MR-STAR problem is NP-hard, as the single-robot case simplifies to the Traveling Salesperson Problem (TSP) without temporal decay or battery constraints. Since finding global optima is intractable for non-trivial instances, the challenge lies in developing scalable algorithms that produce high-quality, sub-optimal solutions efficiently.

The most closely related single-agent problem to STAR is the Time-Varying Reward Orienteering Problem (TR-OP), where an agent maximizes rewards within a time budget. Solutions for TR-OP have included dynamic programming [17] and heuristic search [2], and a greedy algorithm for the single-robot STAR problem was

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/UYQK9511

shown to outperform these methods [5]. While TR-OP has been extended to multi-robot monitoring scenarios [18], these approaches do not address the unique "restoration" objective of STAR. To our knowledge, no work has directly addressed the multi-agent STAR problem, where the goal is to minimize time below a threshold rather than maximize a collected reward.

To address the MR-STAR problem, we design a hierarchical, emergency-inspired framework. By partitioning the environment into jurisdictions and enabling dynamic re-assignment during surge periods, we provide the following contributions:

- We propose a hierarchical algorithm that solves MR-STAR in polynomial time. It first clusters areas to create smaller sub-problems, which are then managed by individual robots using a state-of-the-art heuristic.
- We introduce a dynamic re-assignment mechanism to handle "surge periods", enabling the system to redirect robots to assist overwhelmed agents in urgent areas.
- We provide extensive empirical validation showing our approach significantly outperforms relevant baselines and scales effectively with problem size.

This work provides a foundational framework on multi-robot systems for persistent spatio-temporal restoration, opening new avenues for applying autonomous robots to critical environmental maintenance and monitoring tasks.

2 RELATED LITERATURE

While the MR-STAR problem is novel, our work builds upon research in several related areas, including the single-robot STAR problem, multi-robot monitoring, and dynamic task allocation under crisis conditions. In this section, we review key methodologies from these domains, highlighting the similarities and differences that motivate our hierarchical approach.

STAR-related work. Research on the single-robot STAR problem has established a strong foundation for our work. Initially, Docena and Quattrini Li [5] proposed a real-time, search-based strategy that used a novel heuristic to approximate future losses, but assumed that the decay functions were known. In a subsequent extension, Docena and Quattrini Li [6] relaxed this assumption by incorporating uncertainty. They introduced a dual-component algorithm that combines a risk-averse decision-making process with a time-series model to continuously learn and forecast changing decay rates. In this paper, we address the open problem of the multi-robot case.

TR-OP-related work. The Orienteering Problem with Time-Varying Rewards (TR-OP) is closely related to STAR but has three fundamental differences. First, their objectives diverge: TR-OP seeks to maximize a collected reward from a single tour [2, 17], whereas STAR aims to continuously minimize opportunity costs from perpetual, decaying properties. Second, their visit constraints differ: TR-OP assumes locations are visited at most once, while STAR requires persistent monitoring and allows for strategic re-visits to prevent properties from falling below a critical threshold. Finally, TR-OP assumes a static environment where rewards are fixed, precluding the need for the dynamic re-planning that is central to our approach in handling unforeseen changes and surge periods.

Multi-robot monitoring work. Our work is broadly related to the rich literature on multi-robot monitoring. This includes work in exploration and coverage of known environments [4, 28, 29], as well as persistent patrolling and information gathering [1, 12, 23]. While these strategies are effective, they are often designed for static environments or phenomena.

A more closely related subset of work addresses dynamic environments through adaptive sampling [15, 24]. The key distinction in our MR-STAR problem is that robots are not merely observers; they are active agents whose primary goal is to restore a decaying property, not just monitor it.

Job scheduling. MR-STAR can also be viewed through the lens of multi-agent job scheduling, where each area represents a recurring job that must be serviced by an agent, which can be a robot [10, 20] or CPUs [8]. In this context, the primary goals are often to ensure workload balance among agents and to meet deadline satisfaction for each job, which is analogous to STAR's requirement to keep properties above a critical threshold. While effective for many task allocation problems, we will demonstrate in our experiments that a standard job scheduling approach, which prioritizes temporal deadlines, is outperformed by our method that specifically optimizes for minimizing opportunity costs in a dynamic environment.

MDP. Modeling MR-STAR as a Markov Decision Process (MDP) defines states by area properties and actions by robot assignments. Since exact solutions are computationally intractable [26], we implement an approximate MDP as a practical experimental baseline.

Heirarchical approach. Our methodology is inspired by the hierarchical "cluster-first, route-second" paradigm, an effective strategy in multi-robot task allocation [3, 30] and the Orienteering Problem [7, 9]. This divide-and-conquer approach partitions the environment into smaller, more manageable sub-problems (clusters), which are assigned to individual agents. Each agent then solves its local sub-problem, often with a final fine-tuning step to improve the global solution.

Replanning to respond to crisis. Our dynamic re-assignment mechanism draws from crisis management and emergency response research, where real-time re-planning is essential for handling unforeseen events, e.g., disaster rescue [21]. We treat "surge periods" as localized crises requiring dynamic resource reallocation. Our experiments demonstrate that adaptive re-planning is critical for maintaining area properties during high-demand spikes.

3 PROBLEM STATEMENT

About the environment. In a known environment $W \subset \mathbb{R}^2$, potentially with obstacles $W_o \subset W$, autonomous mobile robots $\mathbf{r} = \{r_1, r_2, ..., r_m\}$ are deployed and operate in free space $W_f \subset W$. Within the environment, there is a charging station where robots can charge simultaneously, indexed as $0 \in W_f$, and areas of interest $J = \{1, 2, ..., n\} \subset W_f$, each with an environmental property F_j that decays over time due to exogenous factors, and therefore needs to be monitored and restored from falling below a critical threshold $z \in \mathbb{R}$. (The case where F_j grows can be formulated as the inverse.)

By prior survey of areas or domain knowledge, as in some environmental studies [16, 22], $F_j = g(t, \delta_j) \in \mathbb{R}$, for $j \in J$, follows a decay function that is monotonic to t with a corresponding rate $\delta_j > 0$. Furthermore, we consider the real-world phenomenon of *spatial spillover* [11], where the evolution of decay in some areas can be correlated to each other due to extraneous factors.

About the robots. We assume that $m < n$, and so task allocation must be identified among $\mathbf{r}$ and J. Each robot can be assigned with either two tasks (or visits): to restore an area, which includes gathering data on F and then restoring it back to safe level; or to recharge its battery level $b_r \in \mathbb{R}$ back to max level at the charging station. Each robot travels with velocity v, is equipped with LiDAR for navigation and a device that restores F. We consider the case where $\mathbf{r}$ are *homogeneous*: all have equal capacities and capabilities. Furthermore, only one robot is needed to restore an area, a single-task single-robot case (ST-SR) [14].

Central planner. We assume oracle knowledge over the decay rates, mirroring scenarios with historical prior data. In future study, we will relax this assumption to incorporate uncertainty. As robots collect data on F during visits, these observations will be used to update the modeled decay function.

Schedule. A schedule $D = \{(r \in \mathbf{r}, D_r)\}$ maps each robot r to its assigned schedule $D_r = \{d_{(1,r)}, d_{(2,r)}, ..., d_{(k,r)}\}$, where D_r is an ordered set of visits, with length $k_r = |D_r| \geq 0$, to either restore an area or charge a robot's battery. The entire schedule D has a total number of assigned visits $K = \sum_{r \in \mathbf{r}} k_r$.

Communication. We assume no restriction on communication, and agents are able to communicate information with each other unobstructed. (A real-world example would be an infrastructure with fail-safe WiFi coverage of the entire environment.)

Objective. The goal of MR-STAR: given the current state, find a *schedule of visits* D^* for $\mathbf{r}$ that minimizes the time that F_j of areas is below z and such that b for each robot does not fully deplete.

4 PROPOSED APPROACH

The paper follows a similar modeling as the single-robot STAR problem [5]. Here, we highlight the main elements.

Environment abstraction. We represent the environment as a weighted graph $G = (V, E)$, where the vertices $V = \{0\} \cup J$, and weighted edges E represent connectivity between vertices wherein the weights are the corresponding distances. These distances can be measured by a path planner, (e.g., using A^* on an inflated map to account for robot footprint), and stored in a (distance) matrix $\mathbf{H}$. Given the robot's velocity v, we then measure the duration it takes to travel among nodes in V, storing in a (duration) matrix $\mathbf{T} = \frac{\mathbf{H}}{v}$. As mentioned previously, each area $j \in V$ has a temporal property F_j that decays according to a decay function $g(t, \delta_j)$.

State dynamics. The state s is composed of the current time T, the set of time elapsed since last restored for all areas $\mathbf{t}$, the location of the robots $\mathbf{l}$, their battery levels $\mathbf{b}$, and the decay rates of the areas δ. The action for each robot would be either to restore an area $j \in J$, which includes travel to and restoration of F_j, or charge up its own battery b_r. Restoring an area takes an amount of time dependent on the restoration rate [5].

4.1 Objective

Loss function. As T progresses, F_j decays toward z if left unrestored. The loss function $L(t_j, \delta_j)$ is inversely related to F.

Minimize opportunity cost. Each committed visit by a robot therefore incurs a loss. We measure the opportunity cost of a schedule of visits as the total losses of taking D given s as:

$$Q_D = \sum_{j \in J} L(t_j, \delta_j) \in \mathbb{R}. \quad (1)$$

We seek a schedule D^* that minimizes the opportunity cost over a sliding temporal window, updated at each replanning interval. Equivalently, by minimizing the total losses, we are minimizing the decay of F across areas as much as possible, so as the duration an area would stay critical, (which is the original problem objective). This combinatorial problem is exponential in the number of robots, areas, and planning horizon; even small-sized instances proved intractable for brute-force search within a feasible timeframe.

Our approach then is to *distribute* areas to robots. Each robot then *locally manages* its assigned cluster.

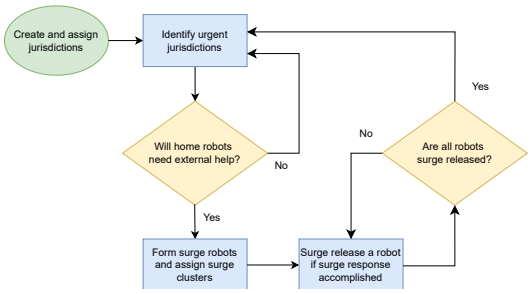


Figure 2: Our proposed method begins with the central planner creating jurisdictions for each robot, clustering areas based on relevant attributes. The central then monitors whether there are overloaded jurisdictions. If home robots assigned to these jurisdictions would benefit from external help, the central forms a team of surge robots and creates surge clusters, assigning the former to the latter. The central then releases a surge robot back to its home jurisdiction once it accomplishes its surge response. If all of the surge robots have been released, the crisis has been responded to, and the central resumes monitoring for overloaded jurisdictions.

4.2 Hierarchical clustering-based approach with replanning

Key idea (Fig. 2). A central planner creates and assigns jurisdiction for each robot. As the robot visits areas and collects information, the central planner constantly checks whether a crisis has occurred. If it occurred, the central forms a team of *surge* robots to respond to the crisis. After a successful crisis response, the surge members return to their original jurisdiction. (Note that with oracle knowledge, the information collection is not needed to update belief.)

4.2.1 Central-level: Hierarchical clustering. Creation of home jurisdictions per robot. The central clusters the J areas into $\mathbf{C} = \{C_1, C_2, ..., C_m\}$ jurisdictions, (i.e., equal to the number of m robots), using relevant attributes $\mathcal{A}$, (e.g., spatial coordinates, decay rates, and other features that may be related to the temporal properties). In our implementation, we used K-means clustering as the clustering technique. The central then computes some planning metrics

and then assigns each robot to a jurisdiction, sending information about the assignment and relevant metrics.

Pre-compute planning metrics. We define a function $\bar{\tau}(\cdot)$ that takes in an array/matrix and measures the average value. The central estimates some measures needed for decision-making:

(1) The rate of duration traveled per unit battery consumed, b_{rate}, based on historical data of travel duration and battery consumption; (trials can be run if no data available).
(2) The duration matrix of each jurisdiction, $\mathbf{T}_C$, sliced from the global duration matrix $\mathbf{T}$ correspondingly.
(3) The average duration an area j decays if left un-restored measured as $\bar{\tau}(\mathbf{T}[\,:i \neq j])$; that is, we take the average of all the columns in $\mathbf{T}$ except for column j. What we are measuring is the average time a robot is visiting vertices other than area j, hence area j is decaying for this duration.
(4) The average duration a robot would take to reach an area j coming from all other vertices, measured as as $\bar{\tau}(\mathbf{T}[\,:j])$; i.e., the average of column j.
(5) The average duration to visit any area in a jurisdiction, measured as $\bar{\tau}(\mathbf{T}_C)$; i.e., the average value of $\mathbf{T}_C$, equivalently the average time it takes to reach a vertex within that cluster.

For brevity, we denote (3) as $\bar{\tau}(\neg j)$, (4) as $\bar{\tau}(j)$, and (5) as $\bar{\tau}(C)$. In all our estimates, we assume that the duration of a visit is based primarily on travel, (i.e., based on $\mathbf{T}$).

Assign robot to a cluster. We identify the *anchor* of a cluster as the area $j \in C$ that has the minimum F in that cluster. We then assign a robot to a cluster based on its distance from that cluster's anchor via Hungarian assignment. (We likewise consider a greedy assignment. In our experiments, we show that a Hungarian assignment scales comparably well to a greedy assignment.) We shall use this metric for assignment, (i.e., distance to an anchor), in both home jurisdictions and surge clusters (introduced in later subsections).

4.2.2 Robot-level: Heuristic decision-making to manage assigned cluster. Given a cluster of areas C, the robot employs the Best-forecasted Visit Greedy (BFVG) algorithm [5] on that cluster to come up with a decision on which areas to visit in $O(|C|^2 k)$ time, where k is a user-defined parameter on the number of visits to forecast. The algorithm takes the visit d that minimizes a heuristic that measures the *discounted* opportunity cost of a visit:

$$\hat{Q}_d = \sum_{j \in J} L(t_{(j,1)}, \delta_j) + \sum_{i=2}^{k} [\gamma^{i-1} \sum_{j \in J} L(\hat{t}_{(j,i)}, \delta_j)] \in \mathbb{R}$$

where the first summand is its immediate opportunity cost; while, the second summand, discounted by γ, is its forecasted opportunity costs supposing r takes $k-1$ visits afterwards.

We now have robots assigned to their respective jurisdictions for persistent restoration. Up next, we include replanning for the possibility of robots responding to crisis.

4.2.3 Central-level: Crisis mitigation. We incorporate in our approach the handling of the real-world scenario of *surge periods* when majority of the areas in a jurisdiction are predicted to become critical, and may overwhelm the assigned home robot. We thus consider sending some home robots from other jurisdictions temporarily to help out.

Crisis detection. We classify a jurisdiction C as *highly urgent* if it experiences *internal overload* and *external help* would be feasible.

Internal overload: For a length of $\tilde{k}$ visits (user-defined), the central forecasts decayed measure of each area from time elapsed t_j:

$$\hat{F}_j = g(\hat{t}_j, \delta_j), \tag{2}$$

where $\hat{t}_j = t_j + \tilde{k} \cdot \bar{\tau}(\neg j)$ is the forecasted time elapsed that area j has decayed, on average. (Note that $\tilde{k}$ is a central-level parameter, similar to the k parameter at the robot-level, allowing user flexibility in non-myopic decision-making.) We define an area as *highly urgent* if its $F_j \leq z$, and *predicted* highly urgent in the next $\tilde{k}$ visits if $\hat{F}_j \leq z$. Denote the set of predicted highly urgent areas in jurisdiction C as Ω_C. We mark C as *overloaded* if majority of its areas (i.e., more than half) are predicted highly urgent and there is already at least one area that is actually highly urgent:

$$|\Omega_C| \geq \frac{|C|}{2} \wedge \exists j \in C : (F_j \leq z) \Rightarrow \text{C is overloaded} \tag{3}$$

External feasibility: We then assess whether the home robot of C could actually handle the overload by the time we expect an available external (or non-home) robot could arrive to help.

(1) *Assess home robot.* We estimate the number of areas the home robot r could visit given its battery as: $\left\lfloor \frac{b_r \cdot b_{\text{rate}}}{\bar{\tau}(C)} \right\rfloor$; i.e., the travel duration its battery can cover divided by the average duration to visit an area in C. Thus: $\min\left(|\Omega_C|, \left\lfloor \frac{b_r \cdot b_{\text{rate}}}{\bar{\tau}(C)} \right\rfloor\right)$ is the number of predicted highly urgent areas the home robot could feasibly visit.
(2) *Assess external help.* We determine the anchor of C as the area with the most predicted decay. For each available external robot, we estimate the battery consumed in traveling [5] to the anchor, and whether its battery can cover this consumption. The minimum of all these feasible durations, denoted as t_e, is our estimated duration an available robot would take to arrive. Dividing by $\bar{\tau}(C)$, we approximate the number of visits the home robot would have made, on average, by the time external help could arrive: $\left\lfloor \frac{t_e}{\bar{\tau}(C)} \right\rfloor$.
(3) *Evaluate feasibility.* We thus assess C as *externally feasible* for help if there is still some predicted highly urgent area that an external robot can restore when it arrives (otherwise, the home robots manage C by default):

$$\left\lfloor \frac{t_e}{\bar{\tau}(C)} \right\rfloor < \min\left(|\Omega_C|, \left\lfloor \frac{b_r \cdot b_{\text{rate}}}{\bar{\tau}(C)} \right\rfloor\right) \Rightarrow \text{C is externally feasible} \tag{4}$$

If C is both overloaded and externally feasible, we mark that jurisdiction as *highly urgent*. Let $\mathcal{M}$ be the set of these marked jurisdictions, and Γ the set of predicted highly urgent areas in $\mathcal{M}$.

Formation of surge robots. Let $\mathbf{r}_{\text{b}}$ be the corresponding home robots of $\mathcal{M}$, while $\mathcal{P}$ the available external robots. We form a team of *surge robots* initialized as $\mathbf{r}_{\text{surge}} = \mathbf{r}_{\text{b}}$. Now if $|\mathbf{r}_{\text{surge}}| < |\Gamma|$, there is some predicted highly urgent area that some $x \in \mathcal{P}$ can potentially help out. We thus evaluate the value of sending robot x. Note that this is different from evaluating external feasibility (Eq. 4), because here we evaluate the net value of sending a robot for

surge response; while, in external feasibility we are computing for the duration external help would arrive.

Computing present value of losses: For a given cluster of areas C, we measure the *present value* of the total losses if left unrestored as:

$$PV(C) = \sum_{j \in C} \frac{1}{(1+\gamma^h)} L(\hat{t}_j, \delta_j), \tag{5}$$

where $\hat{t}_j$ is forecasted as in Eq. (2), and $h = |C|$, the minimum number of visits to restore all the areas in C.

(1) *Net value of surge response:* It is estimated as $PV_{\text{surge}} = \frac{PV(\Gamma)}{\max(1,|\mathcal{P}|)}$. While for each robot $x \in \mathcal{P}$, we estimate the present value of its respective home jurisdiction, $PV_{\text{home}} = PV(C_x)$. If $PV_{\text{home}} < PV_{\text{surge}}$, then the valuation of sending the robot away temporarily for a surge response is better than the valuation of keeping it home.

(2) *Cost of surge response:* We further check whether the battery can cover the cost of surge response. The expected size of a surge response, assuming Γ would be distributed uniformly among $\mathcal{P}$, is $\left\lfloor \frac{|\Gamma|}{\max(1,|\mathcal{P}|)} \right\rfloor$. We estimate the travel cost to a highly urgent area as the average of all $\bar{\tau}(j)$, $j \in \Gamma$ divided by the b_{rate}. We thus estimate the expected *surge cost* as the expected size of a surge response times the travel cost to a highly urgent area.

(3) *Surge team inclusion:* If robot x has enough battery to cover this cost, we include x among *viable* external robots:

$$\mathbf{r}_p = \left\{ x \in \mathcal{P} : b_x \geq \left\lfloor \frac{|\Gamma|}{\max(1,|\mathcal{P}|)} \right\rfloor \cdot \left(\frac{\sum_{j \in \Gamma} \bar{\tau}(j)}{|\Gamma|} \right) \cdot \left(\frac{1}{b_{\text{rate}}} \right) \right\} \tag{6}$$

The team of surge robots is thus composed of the home robots plus the viable after valuation:

$$\mathbf{r}_{\text{surge}} = \mathbf{r}_{\text{b}} \cup \mathbf{r}_{\text{p}} \tag{7}$$

Creation of surge clusters and surge assignment. We form *surge clusters*, $\mathbf{C}_{\text{surge}}$, by clustering Γ into $\min(|\mathbf{r}_{\text{surge}}|, |\Gamma|)$ clusters. Similar to Subs. 4.2.1, we assign a surge robot to a surge cluster by the distance of the former to the anchor of the latter.

Surge binding and release. We bind a surge robot, locking the areas from being served by other robots. Once the surge cluster is not classified as internally overloaded, we release the surge robot back to its home jurisdiction, unlocking the areas the now released surge robot had served, updating the serviceable areas of the respective home jurisdictions to which these unlocked areas belong.

Crisis mitigated. We consider the crisis mitigated if all surge robots have been released.

4.3 Hierarchical Best-forecasted Visit Greedy Algorithm with Crisis Mitigation (H-BFVG-CM)

We propose Algorithm 1 for the MR-STAR problem. Lines 1-3 of Algorithm 1 comprises the creation and assignment of jurisdictions; while, Lines 4-23 is the constant check for crisis and corresponding mitigation. (In our experiments we show that this replanning further improves the performance of the hierarchical approach.) The algorithm assumes two user-defined parameters used for non-myopic decision-making: $\tilde{k}$ at the central-level while k at the robot-level. The higher value, the more steps ahead will be predicted but with an increase of computation time. In the experiments, we set $\tilde{k}$ and k based on preliminary trials. We included the supporting subroutines in the supplementary document on github.

Algorithm 1 Hierarchical Best-forecasted Visit Greedy Algorithm with Crisis Mitigation (H-BFVG-CM)

Require: robots $\mathbf{r}$ and velocity v; areas J and decay rates δ; clustering attributes $\mathcal{A}$; distance matrix H; critical threshold z; central forecast $\tilde{k}$; robot forecast k; discount γ

```
1:  C ← CreateClusters(J, r, A)
2:  Compute planning metrics: b_rate; {∀C ∈ C : T_C, τ̄(C)}; {∀j ∈ J : τ̄(j), τ̄(¬j)}
3:  AssignClusters(C, r, k)
4:  Initialize state s = {T, t, l, b, δ}
5:  mitigating ← False
6:  while system running do
7:      Ω, F̂, t̂ ← PredictHighlyUrgentAreas(k̃, τ̄, z)
8:      if mitigating is False then
9:          M ← ∅                                          ▷ highly urgent jurisdictions
10:         for each home cluster C ∈ C do
11:             Ω_C ← C ∩ Ω                                ▷ predicted highly urgent areas in C
12:             if InternalOverload(C, Ω_C) then
13:                 if ExternalFeasibility(C, F̂, Ω_C) then
14:                     M ← M ∪ {C}
15:         if M ≠ ∅ then
16:             Γ ← ⋃_{C∈M} Ω_C                            ▷ predicted highly urgent areas in highly urgent jurisdictions
17:             r_surge ← FormSurgeRobots(M, Γ, t̂)
18:             if |r_surge| ≠ ∅ then
19:                 C_surge ← CreateClusters(Γ, r_surge, A)
20:                 AssignClusters(C_surge, r_surge, k)    ▷ Hungarian or Greedy
21:                 BindSurgeRobots(r_surge)               ▷ locks
22:                 mitigating ← True
23:     else
24:         ReleaseSurgeRobots(r_surge, Ω)                 ▷ releases
25:         if all r_surge released then
26:             mitigating ← False
27:     Update state s
```

Runtime. We analyze the runtime of each component.

S0: Pre-computing preliminaries. Computing $\bar{\tau}$ takes $O(n)$; so for all areas, $O(n^2)$.

S1: Creating jurisdictions: We employ K-means clustering, which takes $O(qnm|\mathcal{A}|)$ time, where q is the number of iterations for convergence, n number of areas to be clustered, m the number of clusters, and $|\mathcal{A}|$ the number of attributes.

S2: Assignment of robot to a cluster. We employ Hungarian assignment, which has $O(m^3)$. If instead a greedy assignment is employed, we have $O(m^2)$. (In our experiments, we show that the Hungarian assignment scales well relative to the greedy assignment.)

S3: Prediction of highly urgent areas. Prediction per area (Eq. 2) takes $O(1)$; so for all areas it only takes $O(n)$.

S4: Crisis detection. We go through m clusters: For internal overload (Eq. 3), the check takes $O(1)$ and $O(n)$, so $O(mn)$. If overloaded, we check for external feasibility (Eq. 4): the check takes $O(1)$ over m robots; thus $O(m^2)$. Overall, this component takes $O(mn + m^2)$ if overloaded; otherwise, only $O(mn)$.

S5: Crisis mitigation. For valuation, calculating an area's future loss is $O(1)$. Summing PV_{home} across n/m areas for all m robots yields $O(n)$, while PV_{surge} is $O(n)$. Comparing them is $O(1)$. For costs, calculating averages is $O(n)$ with $O(1)$ checks; repeated for m robots, this is $O(mn)$. Thus, the total complexity is $O(mn)$.

S0 and S1 are called once. Within the loop, we constantly check for S3. S4 is run only when there is no ongoing crisis mitigation. We call S2 whenever we need to assign a robot to a cluster. And S5 only when we detected a crisis.

Baseline	4×24	8×64	10×100
Mixed portfolio	42% (p=0.00)	38% (p=0.00)	34% (p=0.00)
Solo distributed	19% (0.00)	17% (0.00)	20% (0.00)
Global greedy	42% (0.00)	39% (0.00)	35% (0.00)
Job scheduling	20% (0.00)	16% (0.00)	20% (0.00)
MDP	49% (0.00)	39% (0.00)	38% (0.00)

Table 1: (With spatial correlation) Improvement of proposed H-BFVG over baselines in average duration areas are below critical threshold for m robots $\times$ n areas repeated 1,000 trials, with Welch one-tailed t-test for statistical significance.

5 DISCUSSION: BOUNDED OPPORTUNITY COST

Bounded opportunity cost per jurisdiction. [5] provided a strict upper bound on the opportunity cost of a robot r's schedule, given a set of areas to restore, and schedule length. Given the state s and a jurisdiction C_r of areas, we bound the opportunity cost of the home robot's schedule D_r with length k_r as:

$$Q_{D_r} < (|C_r| - 1) \sum_{i=1}^{k_r} \gamma^{i-1} L([\max(\mathbf{t}) + i \max(\mathbf{T})], \max(\delta)).$$

Set this upper bound as Φ_r.

Bounded opportunity cost overall. Thus, for the entire schedule of robots D:

$$Q_D < \sum_{r \in \mathbf{r}} \Phi_r.$$

Bounded opportunity cost after surge response. Since after a surge response all highly urgent areas in Γ would have been restored, we subtract from the upper bound of a jurisdiction the losses that have been restored therein:

$$Q_{D_r} < \Phi_r - \sum_{j \in \Gamma \cap C_r} L(\hat{t}_j, \delta_j).$$

Summing across all jurisdictions:

$$Q_D < \sum_{r \in \mathbf{r}} \left(\Phi_r - \sum_{j \in \Gamma \cap C_r} L(\hat{t}_j, \delta_j) \right).$$

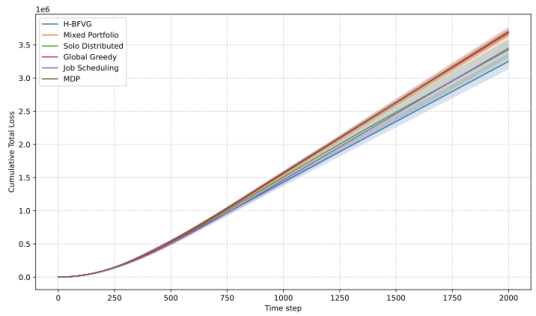
6 EXPERIMENTS

Experiments were conducted using a custom simulator for rapid prototyping and Stage 2D [27] to account for motion and sensor noise. Our open-source Python implementation runs on ROS[1] and was tested on an Intel i7-10750H CPU (2.96GHz) with 16GB RAM.

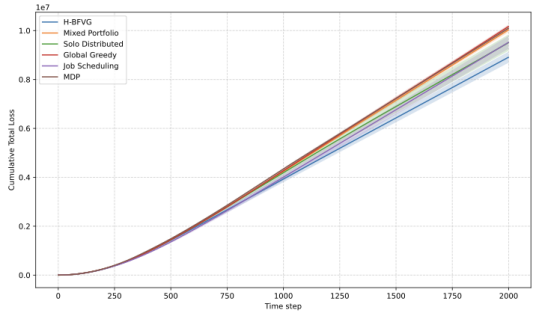
6.1 Extensive Quantitative Evaluation

Experimental setting. We evaluated our method in a 500x500 unit open environment to systematically assess performance without the confounding factor of complex navigation. We divided the map into four quadrants, distributing the areas among the quadrants evenly and randomly placed. Each area has $F_j \in [0, 100]$ and a critical threshold $z = 50$. Key environmental parameters were varied across trials, particularly **spatial correlation** with two cases: decay rates of areas within same quadrant are/are not correlated.

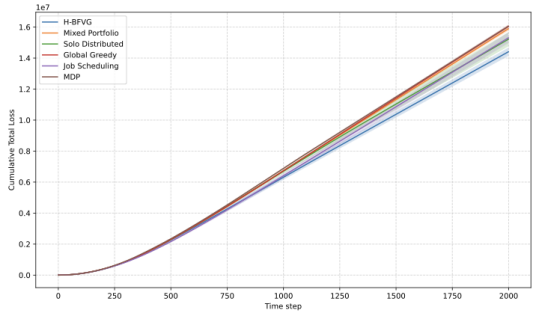
[1]https://github.com/greenguy13/intermittent_preservation.git



(a) Total losses: 4 robots $\times$ 24 areas



(b) 8 robots $\times$ 64 areas

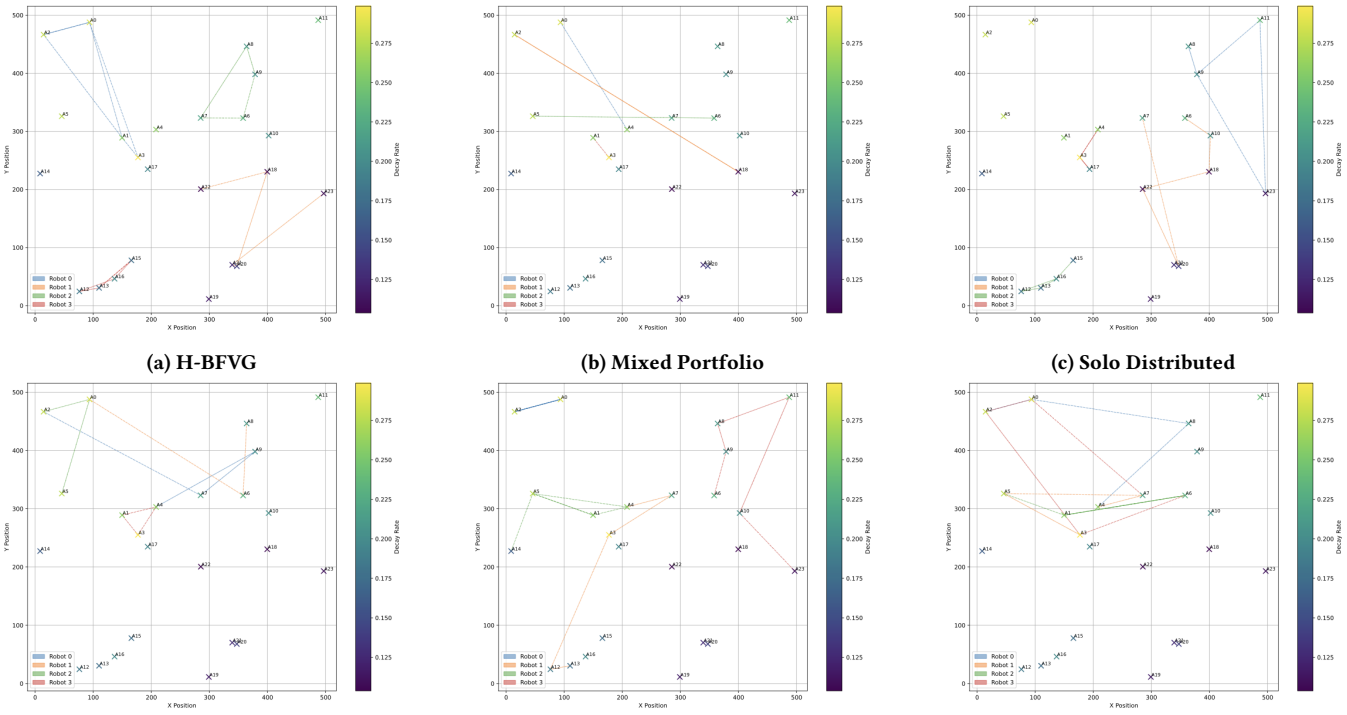


(c) 10 robots $\times$ 100 areas

Figure 3: (With spatial correlation) Proposed H-BFVG has best overall performance over baselines, with least total losses across m robots $\times$ n areas repeated 1,000 trials.

We benchmarked our proposed algorithm against five baselines covering different types of approaches, running 1,000 trials for each configuration of m robots and n areas: 4, 8, 10 robots $\times$ 6, 8, 10 areas.

(1) *Proposed approach:* We first evaluate the hierarchical clustering approach, (i.e., Lines 1-3 of Algorithm 1), denoted as H-BFVG. We used K-means clustering wherein the attributes $\mathcal{A}$ used for clustering are the spatial coordinates and decay rates of areas. (In the next subsection, we evaluate the improvement in performance when responding to crisis.)
(2) *Mixed portfolio:* This baseline builds balanced risk portfolios by assigning every robot a mix of "fast" (high-risk) and "slow" (low-risk) decaying areas. Areas are sorted by decay rate and split into high-risk and low-risk sets. Assignments are



(a) H-BFVG (b) Mixed Portfolio (c) Solo Distributed

(d) Global Greedy (e) Job Scheduling (f) MDP

Figure 4: Routing behavior snapshot 4 robots × 24 areas under spatial correlation.

made via an alternating fill strategy (one high-risk, then one low-risk), selecting the area spatially closest to the last assignment to maintain local coherence. The intuition behind this approach is that jurisdictions should be more balanced, allowing robots to revisit more frequently higher-risk areas while having lower-risk ones visited from time to time.

(3) *Solo distributed:* A sequence of n visits for a single robot computed brute-force if tractable, otherwise greedily; then that sequence of visits distributed among the m robots.
(4) *Global greedy:* The central planner greedily assigns the most critical area to the best-suited robot at each planning.
(5) *Job scheduling:* We model STAR as a deadline-driven scheduling problem. Areas are sorted by urgency (time until critical threshold), treating this duration as a "deadline". For each area, we estimate the time of arrival for all robots; the area is assigned to the robot with the minimum ETA. Robots execute their accumulated task queues sequentially.
(6) *MDP (approximate):* We implemented with key components: i) Benefit matrix: predicted loss saved minus travel penalties for each robot-area pair. ii) Action generation: uses Hungarian assignment on the benefit matrix to generate candidate joint actions. iii) Rollout simulation: simulates future epochs for each candidate, which includes event-driven decay, restoration upon arrival, and discounted long-term rewards. The plan with the highest rollout value is selected.

6.1.1 Results. Performance. The proposed approach has the best overall performance, with minimum cumulative losses (statistically significant at 1% confidence, Fig. 3). Our approach has also the least duration that an area stays below the critical threshold (significant at 1%, Table 1). The same is observed even without assuming spatial correlation (included in the supplementary document on github).

Routing behavior. The routing paths that the robots followed provide insights on the properties of each algorithm (Fig. 4). With our proposed approach, the areas appear to be well clustered, allowing for efficient routing, thereby restoration. The other baselines, however, appear to plan paths that overlap, resulting in inefficiencies.

6.1.2 Ablation. We evaluate the impact of having crisis mitigation in our proposed method. We simulate a *surge moment* where areas in quadrants Q1 and Q2 are decaying 100% faster than those in Q3 and Q4. We implemented Algorithm 1 with tuned $\tilde{k} = 3$, $k = 1$, and $\gamma = 0.75$ based on preliminary trials—denoted H-BFVG-CM.

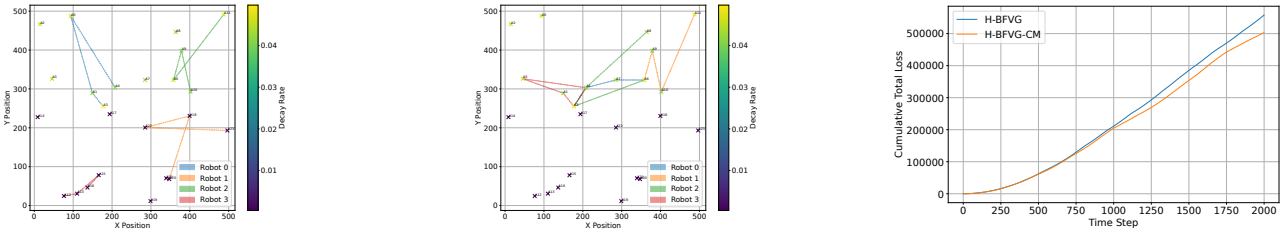
Results. There is improvement in performance by employing crisis mitigation than without, (see Fig. 5). As shown by the routing behavior, robots are able to collaborate to help mitigate the crisis.

6.2 Experiments in Realistic Robot Simulator

Experimental setting. We ran experiments on Stage [27], a 2D robotic simulator with motion and sensor noise.

We considered realistic maps (Fig. 6): "sdr site b" (referred here as 'sdr-b') from the Radish repository [13], as a realistic indoor environment; and "grass" (as 'cluttered'), characterized by cluttered obstacles, from the robotics simulator repository, MRESim [25].

Spatial correlation: To ensure obvious consequences of committed visits, as in [5], we divided the map into four quadrants, evenly



(a) Routing behavior snapshot H-BFVG **(b) Routing behavior snapshot H-BFVG-CM** **(c) Improved results with crisis mitigation**

Figure 5: Improvement in performance with crisis mitigation during surge periods, 4 robots × 24 areas w spatial correlation.

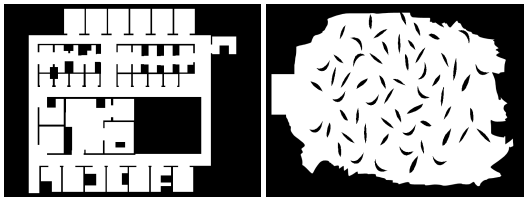
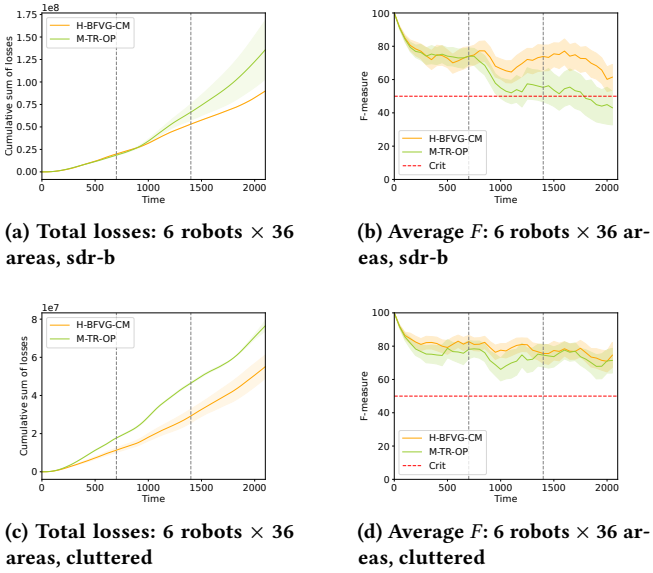


Figure 6: Maps: sdr-b (left), cluttered (right); 80m x 60m.

and randomly distributing the areas among the quadrants. Areas within Q1 (top-right) have exponential decay rates within the range $(0.00100, 0.00125]$; Q2 (top-left), $(0.00150, 0.00175]$; Q3 (bottom-L), $(0.00200, 0.00225]$; and Q4 (bottom-R), $(0.00230, 0.00255]$. Without intervention, $F_j \in [0, 100]$ in this environment will hit the critical threshold $z = 50$ three to eight times.

Surge periods: There will be at least two ($T = 700, 1400$) over the entire time horizon, where all areas will have hit the critical threshold if left unrestored. We benchmarked our proposed hierarchical algorithm with crisis mitigation (H-BFVG-CM) against a strong baseline adapted from the most relevant literature, a multi-agent Time-Varying Reward Orienteering Problem (M-TR-OP) solver. This allows for a direct comparison between our restoration-focused objective and a more traditional reward-maximization approach. We ran the experiment with 6 robots × 36 areas, repeated 5 trials.

Results. The proposed approach performs better than the state of the art, preserving F above the critical threshold better (Fig. 7).



(a) Total losses: 6 robots × 36 areas, sdr-b **(b) Average F: 6 robots × 36 areas, sdr-b**

(c) Total losses: 6 robots × 36 areas, cluttered **(d) Average F: 6 robots × 36 areas, cluttered**

Figure 7: Our proposed approach H-BFVG-CM outperforms existing related state of the art, preserving F above the critical threshold (red dashed line) better, especially during surge periods (dashed vertical lines).

6.3 Scalability

We show that Algorithm 1 scales well, measuring the scalability of its components: creation of jurisdictions, assignment of a cluster to a robot, crisis detection, and crisis mitigation (Fig. 8). Our proposed approach, with all components, can execute on the order of a few seconds or less, even with a large number of robots and areas.

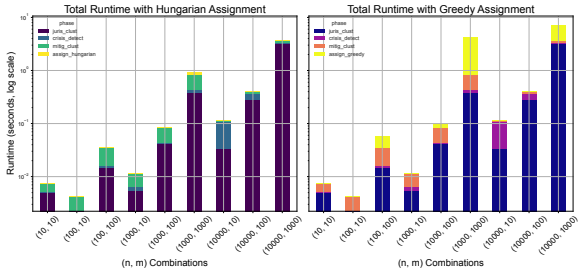


Figure 8: Scalability of Algorithm 1, with n areas and m robots.

7 FUTURE WORK AND CONCLUSIONS

We assumed perfect decay knowledge and unrestricted communication. Future work will incorporate online learning for environmental uncertainty and decentralized control for communication constraints.

In summary, we introduced the Multi-Robot Spatio-Temporal Area Restoration (MR-STAR) problem, a novel, NP-hard challenge in persistent autonomy. We presented a hierarchical algorithm with a dynamic re-assignment mechanism to handle "surge periods", inspired by emergency response systems. Our experiments confirmed that this approach is scalable, suitable for real-time application, and outperforms existing methods from related domains. This work provides a foundational framework for deploying autonomous systems in critical environmental restoration tasks.

ACKNOWLEDGMENTS

This work is supported in part by the NSF CNS-2144624, OIA1923004.

REFERENCES

[1] Nicola Basilico and Stefano Carpin. 2020. Balancing unpredictability and coverage in adversarial patrolling settings. In *Workshop on the Algorithmic Foundations of Robotics (WAFR)*. 762–777.

[2] Chao Cao, Jinyun Xu, Ji Zhang, Hoset Choset, and Zhongqiang Ren. 2024. Heuristic Search for the Orienteering Problem with Time-Varying Rewards. In *AAAI Proceedings of the 17th International Symposium on Combinatorial Search*, Vol. 17. https://doi.org/10.1609/socs.v17i1.31537

[3] Hamza Chakraa, François Guérin, Edouard Leclercq, and Dimitri Lefebvre. 2023. Optimization techniques for Multi-Robot Task Allocation problems: Review on the state-of-the-art. *Robotics and Autonomous Systems* (2023).

[4] Howie Choset. 2000. Coverage of known spaces: The boustrophedon cellular decomposition. *Autonomous Robots* 9 (2000), 247–253.

[5] Amel Nestor Docena and Alberto Quattrini Li. 2024. Search-based Strategy for Spatio-Temporal Environmental Property Restoration. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

[6] Amel Nestor Docena and Alberto Quattrini Li. 2025. Persistent Preservation of a Spatio-temporal Environment Under Uncertainty. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

[7] Almiqdad Elzein and Gianni A Di Caro. 2022. A clustering metaheuristic for large orienteering problems. *Plos one* 17, 7 (2022).

[8] Yiqin Gao, Li Han, Jing Liu, Yves Robert, and Frédéric Vivien. 2024. Minimizing energy consumption for real-time tasks on heterogeneous platforms under deadline and reliability constraints. *Algorithmica* 86, 10 (2024), 3079–3114.

[9] Damianos Gavalas, Charalampos Konstantopoulos, Konstantinos Mastakas, Grammati Pantziou, and Yiannis Tasoulas. 2013. Cluster-based heuristics for the team orienteering problem with time windows. In *Experimental Algorithms: International Symposium*. 390–401.

[10] Matthew C Gombolay, Ronald J Wilcox, and Julie A Shah. 2018. Fast scheduling of robot teams performing tasks with temporospatial constraints. *IEEE Transactions on Robotics* 34, 1 (2018), 220–239.

[11] Yu Hao, Zhiqiang Gai, Guanpeng Yan, Haitao Wu, and Muhammad Irfan. 2021. The spatial spillover effect and nonlinear relationship analysis between environmental decentralization, government corruption and air pollution: Evidence from China. *Science of The Total Environment* 763 (2021), 144183. https://doi.org/10.1016/j.scitotenv.2020.144183

[12] Geoffrey A Hollinger and Gaurav S Sukhatme. 2014. Sampling-based robotic information gathering algorithms. *The International Journal of Robotics Research* 33, 9 (2014), 1271–1287.

[13] Andrew Howard. 2003. The robotics data set repository (radish). *http://radish.sourceforge.net/* (2003).

[14] G. Ayorkor Korsah, Anthony Stentz, and M. Bernardine Dias. 2013. A comprehensive taxonomy for multi-robot task allocation. *Int. J. Rob. Res.* (2013).

[15] Nicholas RJ Lawrance, Jen Jen Chung, and Geoffrey A Hollinger. 2017. Fast marching adaptive sampling. *IEEE Robotics and Automation Letters* 2, 2 (2017), 696–703.

[16] Shi V Liu, Fu-lin Chen, and Jianping Xue. 2019. A meta-analysis of selected near-road air pollutants based on concentration decay rates. *Heliyon* 5, 8 (2019).

[17] Zhibei Ma, Kai Yin, Lantao Liu, and Gaurav S. Sukhatme. 2017. A spatio-temporal representation for the orienteering problem with time-varying profits. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 6785–6792. https://doi.org/10.1109/IROS.2017.8206597

[18] Ariella Mansfield, Sandeep Manjanna, Douglas G Macharet, and M Ani Hsieh. 2021. Multi-robot scheduling for environmental monitoring as a team orienteering problem. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 6398–6404.

[19] Ishaan Mehta, Hao-Ya Hsueh, Sharareh Taghipour, Wenbin Li, and Sajad Saeedi. 2023. UV disinfection robots: a review. *Robotics and autonomous systems* 161 (2023), 104332.

[20] Ernesto Nunes and Maria Gini. 2015. Multi-robot auctions for allocation of tasks with temporal constraints. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 29.

[21] Sarvapali D Ramchurn, Feng Wu, Wenchao Jiang, Joel E Fischer, Steve Reece, Stephen Roberts, Tom Rodden, Chris Greenhalgh, and Nicholas R Jennings. 2016. Human–agent collaboration for disaster response. *Autonomous Agents and Multi-Agent Systems* 30, 1 (2016), 82–111.

[22] James L Repace, Wayne R Ott, and Neil E Klepeis. 1998. Indoor air pollution from cigar smoke. *Smoking and Tobacco Control Monograph 9. Cigars—Health Effects and Trends* (1998), 161–179.

[23] Mac Schwager, Philip Dames, Daniela Rus, and Vijay Kumar. 2017. A multi-robot control policy for information gathering in the presence of unknown hazards. In *International Symposium of Robotics Research (ISRR)*. 455–472.

[24] Ryan N Smith, Mac Schwager, Stephen L Smith, Burton H Jones, Daniela Rus, and Gaurav S Sukhatme. 2011. Persistent ocean monitoring with underwater gliders: Adapting sampling resolution. *Journal of Field Robotics* 28, 5 (2011), 714–741.

[25] Victor Spirin, Stephen Cameron, and Julian De Hoog. 2014. Time preference for information in multi-agent exploration with limited communication. In *Towards Autonomous Robotic Systems: (TAROS)*. Springer, 34–45.

[26] Aviv Tamar, Shie Mannor, and Huan Xu. 2014. Scaling up robust MDPs using function approximation. In *International conference on machine learning*. PMLR, 181–189.

[27] Richard Vaughan. 2008. Massively multi-robot simulation in stage. *Swarm intelligence* 2 (2008), 189–208.

[28] Anqi Xu, Chatavut Viriyasuthee, and Ioannis Rekleitis. 2014. Efficient complete coverage of a known arbitrary environment with applications to aerial operations. *Autonomous Robots* 36 (2014), 365–381.

[29] Brian Yamauchi. 1997. A frontier-based approach for autonomous exploration. In *IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA). Towards New Computational Principles for Robotics and Automation*. 146–151.

[30] Han Zhang, Mingze Yao, Ziang Liu, Jiaoyang Li, Lucas Terr, Shao-Hung Chan, TK Satish Kumar, and Sven Koenig. 2021. A hierarchical approach to multi-agent path finding. In *Proceedings of the International Symposium on Combinatorial Search*, Vol. 12. 209–211.

[31] Wen Zhang, Zhonghua Miao, Nan Li, Chuangxin He, and Teng Sun. 2022. Review of current robotic approaches for precision weed management. *Current robotics reports* 3, 3 (2022), 139–151.